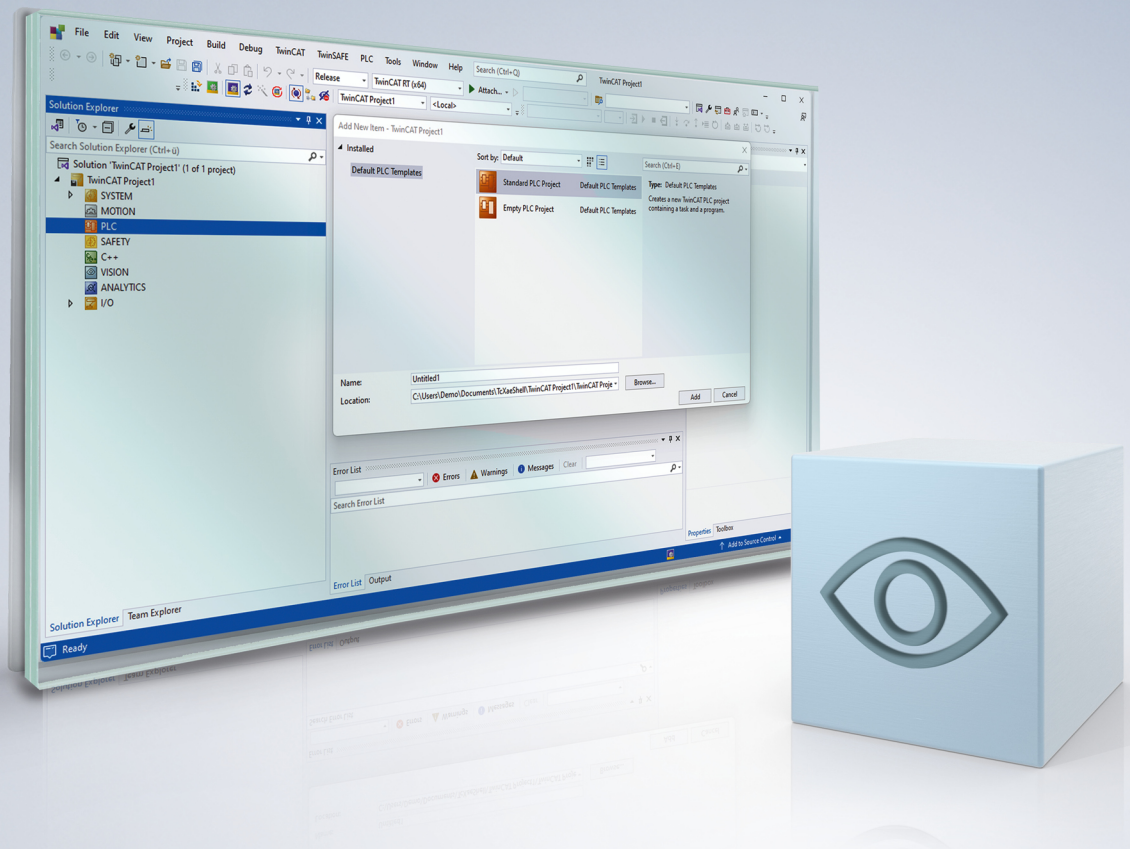
BECKHOFF New Automation Technology

手册 | ZH

TF7000 - TF7810

TwinCAT 3 | Vision



2026-10-29 | 版本： 2.6.0

目录

1 前言 7
1.1 文档说明 7
1.2 安全信息 8
1.3 信息安全说明 8
1.4 文档发行状态 9
2 概述 10
2.1 系统组件 10
2.2 授权模式 12
3 安装 14
3.1 系统要求 14
3.2 版本概览 15
3.3 TwinCAT 3.1 Build 4024 安装说明 17
3.4 TwinCAT 3.1 Build 4026 及以上版本的安装说明 22
3.5 23
3.6 使用 TwinCAT 3 Usermode Runtime 23
3.7 授权 24
4 第一步 27
4.1 构建项目 27
4.1.1 创建 TwinCAT 项目 27
4.1.2 选择目标系统 29
4.1.3 系统配置 31
4.2 设置视觉设备 33
4.2.1 显示视觉节点并创建应用程序 33
4.2.2 创建文件源控制 35
4.2.3 创建 GigE Vision 相机 38
4.3 创建 PLC 项目 45
4.3.1 创建 PLC 项目 45
4.3.2 集成 PLC 库 46
4.3.3 编写 PLC 程序 48
4.3.4 初始化功能块 49
4.4 运转 50
4.4.1 激活 TwinCAT 项目并启动 PLC 项目 50
4.4.2 ADS 图像查看中的图像显示 51
5 开发环境 53
5.1 系统配置 53
5.1.1 路由器内存 54
5.1.2 核心内存 55
5.1.3 堆栈大小 56
5.1.4 网络适配器 56
5.1.5 CPU 内核和任务 63
5.1.6 工作任务 66

5.2 视觉节点 ... 68
5.2.1 服务配置 ... 68
5.2.2 记录 ... 70
5.3 应用节点 ... 70
5.4 GigE Vision 相机对象 ... 71
5.4.1 常规 ... 73
5.4.2 配置助手 ... 77
5.4.3 记录/回放 ... 101
5.4.4 相机校准 ... 106
5.4.5 GigE Vision 标准版 ... 120
5.4.6 TcCom 对象 ... 120
5.5 倍福相机对象 ... 122
5.6 文件源对象 ... 123
5.6.1 文件源控制 ... 124
5.6.2 TcCom 对象 ... 126
5.7 视觉工作池 ... 127
5.8 ADS 图像查看 ... 128
5.9 导出/导入节点 ... 130
6 API 参考 ... 132
6.1 PLC ... 132
6.1.1 软件概念 ... 132
6.1.2 数据类型 ... 152
6.1.3 接口 ... 237
6.1.4 功能 ... 429
6.1.5 功能块 ... 1569
6.2 C++ ... 1639
6.2.1 Data Types ... 1640
6.2.2 接口 ... 1719
6.2.3 功能 ... 1882
6.2.4 功能块 ... 2613
7 TwinCAT HMI 软件包 ... 2660
7.1 服务器扩展 ... 2660
7.2 控制 ... 2665
7.2.1 图像查看 ... 2666
7.2.2 颜色 ... 2689
7.2.3 历史 ... 2700
7.2.4 直方图 ... 2707
7.2.5 矩形（交互式） ... 2714
7.2.6 多边形（交互式） ... 2716
7.2.7 帮助功能 ... 2717
7.2.8 数据类型 ... 2720
8 例程和工具 ... 2722
8.1 函数样本 ... 2723

8.1.1 容器基本操作 ... 2723
8.1.2 基本图像操作 ... 2725
8.1.3 编码质量 ... 2728
8.1.4 代码读取 ... 2731
8.1.5 轮廓分析 ... 2747
8.1.6 图像分析 ... 2755
8.1.7 图像颜色和对比度处理 ... 2781
8.1.8 图像过滤 ... 2785
8.1.9 图像分割 ... 2790
8.1.10 测量 ... 2793
8.1.11 OCR ... 2810
8.1.12 自行编写的函数 ... 2814
8.2 功能块样本 ... 2816
8.2.1 相机寄存器访问 ... 2816
8.2.2 文件访问 ... 2818
8.2.3 图像采集 ... 2820
8.3 助手样本 ... 2822
8.3.1 校准助手 ... 2822
8.4 相机配置样本 ... 2828
8.4.1 感兴趣区域（ROI） ... 2828
8.4.2 像素合并 ... 2829
8.4.3 图像采集和触发 ... 2831
8.5 神经网络兼容性检查 ... 2837
9 附录 ... 2840
9.1 常见问题解答 ... 2840
9.2 见重要路径 ... 2841
9.3 故障排除 ... 2842
9.3.1 TwinCAT 系统启动 ... 2842
9.3.2 PLC 运行时环境 ... 2845
9.3.3 助手 ... 2849
9.3.4 TwinCAT HMI 软件包 ... 2852
9.3.5 相机通信 ... 2853
9.3.6 Support Information Report ... 2855
9.4 ADS 返回代码 ... 2857
9.5 技术支持和服务 ... 2861
9.6 第三方组件 ... 2862

1 前言

1.1 文档说明

本说明仅适用于熟悉国家标准且经过培训的控制和自动化工程专家。
在安装和调试组件时，必须遵循文档和以下说明及解释。
操作人员应具备相关资质，并始终使用最新的生效文档。

相关负责人员必须确保所述产品的应用或使用符合所有安全要求，包括所有相关法律、法规、准则和标准。

免责声明

本文档经过精心准备。然而，所述产品正在不断开发中。
我们保留随时修改和更改本文档的权利，恕不另行通知。
不得依据本文档中的数据、图表和说明对已供货产品的修改提出赔偿。

商标

Beckhoff®、ATRO®、EtherCAT®、EtherCAT G®、EtherCAT G10®、EtherCAT P®、MX-System®、Safety over EtherCAT®、TC/BSD®、 TwinCAT®、TwinCAT/BSD®、TwinSAFE®、XFC®、XPlanar® 和 XTS®是 Beckhoff Automation GmbH
的注册商标并由其授权使用。本出版物中所使用的其它名称可能是商标名称，任何第三方出于其自身目的使用它们可能会侵犯商标所有者的权利。





EtherCAT®是注册商标和专利技术，由 Beckhoff Automation GmbH 授权使用。

版权所有

© Beckhoff Automation GmbH。
未经明确授权，不得复制、分发、使用和传播本文档内容。
违者将被追究赔偿责任。Beckhoff Automation GmbH 保留所有发明、实用新型和外观设计专利权。

第三方商标

本文档可能使用了第三方商标。有关商标信息，可以访问：https://www.beckhoff.com/trademarks。

1.2 安全信息

安全规范

为了确保您的使用安全，请务必仔细阅读
并遵守本文档中每个产品的安全使用说明。

责任免除

所有组件在供货时都配有适合应用的特定硬件和软件配置。严禁未按文档所述修改硬件或软件配置，否则，德国倍福自动化有限公司对由此产生的后果不承担责任。

人员资格

本说明仅供熟悉适用国家标准的控制、自动化和驱动工程专家使用。

警示性词语

文档中使用的警示信号词分类如下。为避免人身伤害和财产损失，请阅读并遵守安全和警告注意事项。

人身伤害警告

⚠ 危险
存在死亡或重伤的高度风险。

⚠ 警告
存在死亡或重伤的中度风险。

⚠ 谨慎
存在可能导致中度或轻度伤害的低度风险。

财产或环境损害警告

注意
可能会损坏环境、设备或数据。

操作产品的信息

这些信息包括：
有关产品的操作、帮助或进一步信息的建议。

1.3 信息安全说明

Beckhoff Automation GmbH & Co.KG (简称 Beckhoff) 的产品，只要可以在线访问，都配备了安全功能，支持工厂、系统、机器和网络的安全运行。尽管配备了安全功能，但为了保护相应的工厂、系统、机器和网络免受网络威胁，必须建立、实施和不断更新整个操作安全概念。Beckhoff 所销售的产品只是整个安全概念的一部分。客户有责任防止第三方未经授权访问其设备、系统、机器和网络。它们只有在采取了适当的保护措施的情况下，方可与公司网络或互联网连接。

此外，还应遵守 Beckhoff 关于采取适当保护措施的建议。关于信息安全和工业安全的更多信息，请访问本公司网站 https://www.beckhoff.com/secguide。

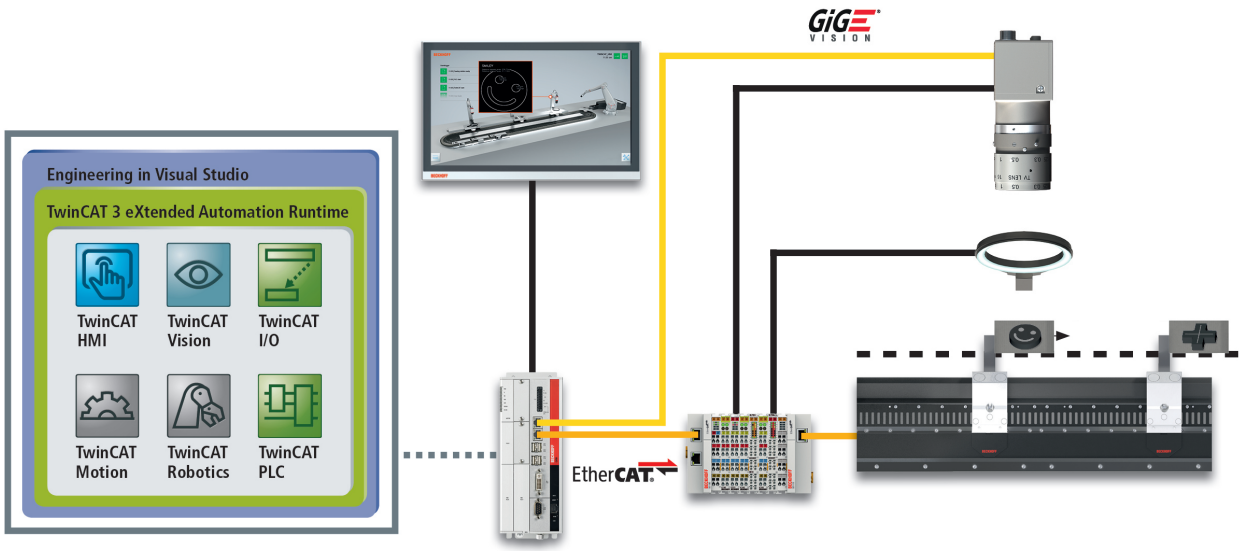
Beckhoff 的产品和解决方案持续进行改进。这也适用于安全功能。鉴于持续进行改进，Beckhoff 明确建议始终保持产品的最新状态，并在产品更新可用后马上进行安装。使用过时的或不支持的产品版本可能会增加网络威胁的风险。

如需了解 Beckhoff 产品信息安全的信息，请订阅 https://www.beckhoff.com/secinfo 上的 RSS 源。

1.4 文档发行状态

版本	修改
2.6.x	**新内容：** 各种功能 [▶ 132] Bandwidth management for GigE Vision [▶ 62]
2.5.x	**新内容：** 巨型帧 [▶ 59] TimestampFormatting 函数 [▶ 2706] 直方图 [▶ 2707] 编码质量 [▶ 2728] 编码检测 [▶ 2731] DotCode （点码）读取 [▶ 2743] 带旋转的Walsh模板匹配 [▶ 2771] Support Information Report [▶ 2855] 各种功能块、函数、数据类型和接口 [▶ 132]
2.4.x	**新内容：** 历史 [▶ 2688] 统计信息 [▶ 2689] 历史 [▶ 2700]
2.3.x	**新内容：** 各种数据类型和功能 安装/使用 TwinCAT 3 Usermode Runtime [▶ 23]
2.2.x	**新内容：** 神经网络和神经网络 C++
2.1.x	**新内容：** 各种功能、编码质量 [▶ 820]和 编码质量 C++ [▶ 2189]函数 **新内容：** TwinCAT 3.1 Build 4026 安装说明 [▶ 22]
2.0.x	**新内容：** 各种功能块、函数、数据类型和接口 [▶ 132] **新内容：** C++ 的 API [▶ 1639]
1.5.x	**新内容：** 各种功能块和函数

2 概述



TwinCAT Vision 是 TwinCAT 3 中的机器视觉功能模块。诸如目标检测、识别和测量等任务可直接在 PLC 中实时执行。通过将图像处理集成到 TwinCAT 平台中，可以实现高度同步的控制过程和极短的响应时间。应用开发由各种现有 TwinCAT Engineering 工具提供支持。

图像处理序列可直接在 PLC 代码中使用 Tc3_Vision 库以 IEC61131-3 语言进行编程。另外，也可以在 TwinCAT 3 C++ 中进行编程。可以使用图像显示工具直接检查中间结果。

TwinCAT Vision 具有实时功能，因为图像处理算法直接在 TwinCAT 3 运行时环境中与控制器同步执行。此外，指定适用多核的算法可以在多个 CPU 核上自动并行执行。

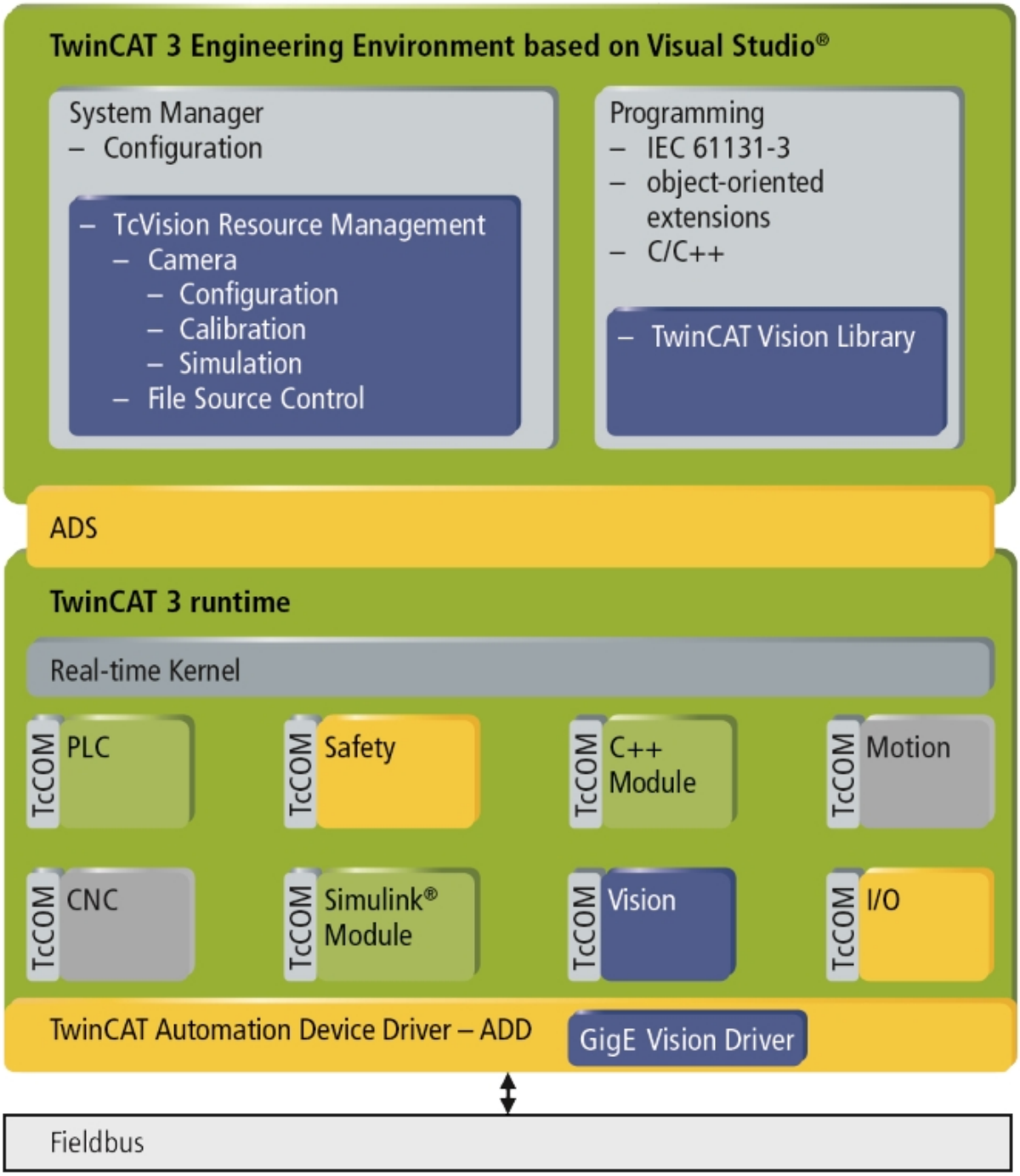
通过 GigE Vision 接口可连接大量工业相机。熟悉的 TwinCAT 开发环境可用于配置和标定相机，以及系统设置和调试。

各章提供有关以下主题的详细信息：

- 系统组件 [▶ 10]，用于了解 TwinCAT Vision 的系统结构
- 安装 [▶ 14]，提供安装和授权说明
- 第一步 [▶ 27]，提供对 TwinCAT Vision 的简单介绍
- Development environment [▶ 53]，有关相机配置、标定和仿真的信息
- API 参考 [▶ 132]（软件文档）
- 一些图像处理序列的示例 [▶ 2722]
- 故障排除须知 [▶ 2842]

2.1 系统组件

与其他 TwinCAT 组件一样，TwinCAT Vision 也集成到基于 Visual Studio 的 TwinCAT 开发环境和具有实时功能的 TwinCAT 运行时环境中。下文对 TwinCAT Vision 系统的组件进行了说明。



配置助手

- 工业相机可通过 GigE Vision [▶ 120] 接口连接。助手 [▶ 71]可提供相机配置、标定和仿真服务。
- 图像可通过文件源 [▶ 123]元件从文件系统输入 PLC 运行时。
- 在 TwinCAT 实时设置中必须针对 TwinCAT Vision 进行核和路由器内存的配置 [▶ 53]。
- TwinCAT Vision 可以利用 TwinCAT 的多核支持来并行执行某些算法。相应设置在 Vision Job Pool [▶ 127] 中进行。

库

TwinCAT Vision 提供一个包含大量算法的库，可用于图像处理序列的编程。它还包含作为相机和文件系统接口的功能块。API 参考 [▶ 132]包含所有库元件的详细说明。

调试工具

TwinCAT Vision 提供以下图像处理序列调试机构：

- ADS Image Watch [▶ 128] 可用于实时可视化相机图像和中间结果。它通过 ADS 将图像从 PLC 运行时传输到开发环境，并在其中显示。
- 所有 TwinCAT Vision 功能均根据 ADS 返回代码 [▶ 2857]（请参见 HRESULT [▶ 135]）提供有关其执行状态的信息。
- 所有 TwinCAT 调试功能（如断点等）均可用于编程。

GigE Vision 驱动程序

TwinCAT Vision 包括一个具有实时功能的GigE Vision [▶ 120]驱动程序。它用于相机配置的开发环境和TwinCAT运行时。它通过UDP/IP与工业相机进行通信，基于GigE Vision标准。该标准用于查找和配置网络中的相机，并接收来自相机的图像数据。

TwinCAT Vision服务

对于某些功能，TwinCAT Vision 必须在运行时与外部组件进行实时通信。例如，文件源控制从文件系统加载图像。TwinCAT Vision Service [▶ 68]是 TwinCAT 运行时与外部 Vision 组件之间的一个集中的、基于ADS 的通信接口。

TwinCAT HMI 视觉扩展

TwinCAT HMI 视觉扩展 [▶ 2660]是TwinCAT HMI的一个扩展，可将图像从 PLC 传输到 TwinCAT HMI。来自 PLC 的图像可以通过 HmiImageControl 直接显示在基于网络的 HMI 中。

TwinCAT HMI 视觉控制

TwinCAT HMI 视觉控制 [▶ 2665]包提供了视觉专用控制，作为TwinCAT HMI的扩展。例如，它包括一个控件，除了图像显示外，控件还提供许多工具和信息显示。这些功能和显示可以很容易地通过配置进行调整，并直接使用，而不需要额外编程。

2.2 授权模式

TwinCAT Vision 的 TF7xxx 产品包括相机连接许可证和图像处理库。请按照以下说明 [▶ 24]生成许可证。使用 TwinCAT Vision 进行测试或开发时，可以创建 7 天试用版的许可证。

有关当前可用的产品许可证，请参阅产品网站。

TF700x | TwinCAT 3 GigE Vision Connector

通过 TwinCAT 3 GigE Vision Connector，可以集成 GigE Vision 相机以及倍福相机。TF7000 的基本许可证始终是必需的，可以连接两台相机。如有需要，可通过扩展许可证来增加容量：

TF7000	基本许可证包括 2 个相机接口 （始终需要）
TF7001	扩展包，用于 2 个附加接口
TF7002	扩展包，用于 4 个附加接口
TF7003	扩展包，用于 8 个附加接口

扩展许可证可以组合使用，因此最多可以连接 2+2+4+8=16 台相机。

例如，11 个相机接口需要以下许可证：

- TF7000 Base（包括 2 个相机接口）
- TF7001 2 相机
- TF7003 8 相机

→ 现在总计可以连接 12 台相机。

TF7020 | TwinCAT 3 Vision Beckhoff Camera Connector

TwinCAT 3 Vision Beckhoff Camera Connector 最多可集成 64 台倍福相机。其他制造商生产的相机不能使用此许可证。

TF7100-TF7810 | TwinCAT 3 视觉算法库

TF7100-TF7810 产品许可证可在“Tc3_Vision”库中提供图像处理功能。使用该库需要基本许可证 TF7100。它提供对基本算法的访问权限，如滤波操作、格式转换和轮廓跟踪。这已经使许多应用得以实现。扩展许可证补充了表中所述的功能范围。功能与许可证的分配可参见 API 参考。

TF7100	“Tc3_Vision”库的基本许可证。包含用于解决图像处理任务的算法，如代数运算、滤波、傅立叶分析、彩色图像处理、分割、轮廓和 Blob 分析、结果显示以及用于读写相机参数的算法。
TF7200	针对二维对象匹配扩展包: 系统能够基于学习过的参考模板、轮廓特征、关键点或其他属性，对目标对象进行识别与比对（如模板匹配、关键点检测及描述符匹配） 常见的应用是对象分类。
TF7250	Code Reading（读码）扩展包: 检测和读取一维条码和二维码。
TF7255	Code Quality（编码质量）扩展包: 一维条码和二维条码质量评估。
TF7260	OCR 扩展包: 用于光学字符识别的函数，可识别图像区域中的字符并返回识别出的字符串。
TF7300	Metrology 2D（测量 2D）扩展包: 用于光学测量几何对象属性（距离、半径等）的工具。 • 以亚像素精度对测量图像进行边缘定位 • 补偿光学几何畸变 **（需预先对相机进行校准）** • 像素坐标与真实世界坐标之间的转换 **（需预先对相机进行校准）**
TF7800	Machine Learning（机器学习）扩展包: 用于培训和执行各种经典 ML 模型的函数。
TF7810	Neural Network（神经网络）扩展包: 用于以 Open Neural Network Exchange（开放神经网络交换，ONNX）格式执行训练好的神经网络的函数。 包含许可证 TF3800、TF3810 和 TF7800。

自动确定所需的视觉库许可证

只有在编译相应的 TwinCAT 项目时，才能确定所需的许可证。同时，还要检查代码中声明了哪些功能块，使用了哪些函数。

因此，请注意，在项目变更的情况下（临时通知），可能需要添加或移除所需的许可证。如果已添加新的许可证，则必须通过 **Activate Configuration（激活配置）**来加载变更项。在这种情况下，无法进行在线更改。

3 安装

本章包含 TwinCAT Vision 的系统要求 [▶ 14]、版本概述 [▶ 15]、以及安装 [▶ 17]和授权 [▶ 24]说明。

3.1 系统要求

IPC 硬件

技术数据	描述
TwinCAT 3 平台等级	最低 P50 性能增强版，如 Intel® 4 核 Atom CPU
网卡	TwinCAT 实时以太网兼容千兆网卡（用于连接 GigE Vision 相机） **至少应有一个专供相机使用的网络端口。建议每个相机使用单独的网络端口。**
主存	纯运行时系统建议至少 4 GB 开发系统建议至少 8 GB 对于在运行时系统上使用记录/回放 [▶ 101]数据流的情况，建议至少 8 GB
硬盘	建议使用读/写速率高的 SSD 硬盘，尤其是在 PLC 和硬盘之间需要交换大量数据的情况。

相机

技术数据	描述
接口	必须支持标准版 GigE Vision [▶ 120]。（通常标有 GigE Vision 徽标） **TwinCAT GigE Vision Connector 已根据标准版 GigE Vision 2.0 获得认证。但这并不意味着支持该版本的所有相机或 GigE Vision 功能。** 在 TwinCAT 3 User Mode Runtime 中使用相机时，必须遵守使用 TwinCAT 3 User Mode Runtime [▶ 23] 一章中所述的要点。

软件

技术数据	描述
操作系统	Windows 10（仅 64 位！） Windows 11，TwinCAT 3.1 Build 4026 及以上版本
支持的 Visual Studio 版本	2017（社区版，专业版） 2019（社区版，专业版） 2022（社区版，专业版），TwinCAT 3.1 Build 4026 及以上 TwinCAT XAE Shell (TcXaeShell) TwinCAT XAE Shell (TcXaeShell64)，TwinCAT 3.1 Build 4026 及以上版本
TwinCAT	最低版本 3.1.4024.59 最低版本 3.1.4026.11
.NET Framework	最低版本 4.7.2（仅用于开发环境 [▶ 53]）

TwinCAT HMI Server 扩展包

技术数据	描述
.NET	最低版本 .NET Desktop Runtime 8.0

3.2 版本概览

TwinCAT 3.1.4024 设置	TwinCAT 3.1.4026 工作负载	注释
-	5.9.19	
4.0.8.4	5.8.4	
4.0.7.4	5.7.4	
4.0.6.5	5.6.5	新增 Neural Network（神经网络）。
4.0.5.10	5.5.10	新增倍福 Camera Connector 和 Code Quality（编码质量）。
4.0.4.8	-	新增 C++ API、OCR 型号和 Machine Learning（机器学习）。
4.0.3.5	-	已删除在编译过程中或激活配置期间对初始化命令的自动检查。
4.0.2.13	-	
4.0.1.3	-	安装更新设置后会删除存储库文件。 **如果您想保留该版本，请在安装新设置之前保存“%TWINCAT3DIR%\3.1\Repository\Beckhoff Automation GmbH”文件夹，并在安装后将其复制回去。**

注意

兼容性

如果在一个系统上安装了多个 TwinCAT Vision 版本，或者开发和运行时环境位于不同的系统上，则必须确保使用的所有组件版本均相同。

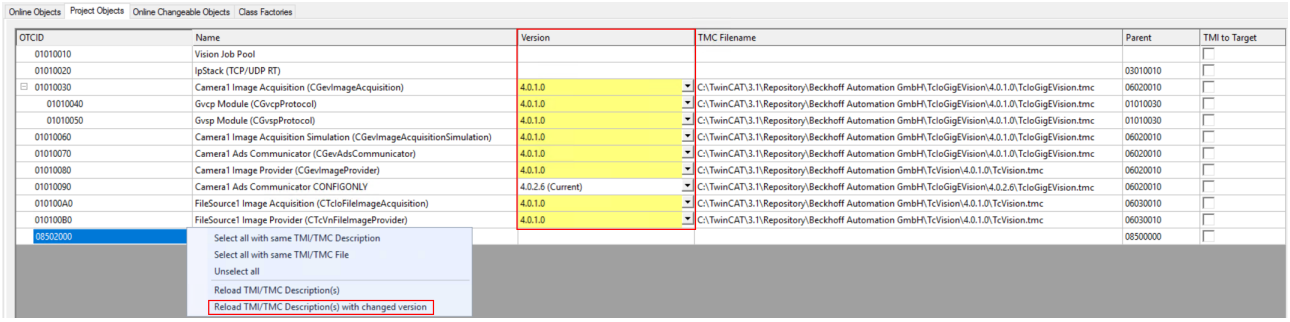
TwinCAT Vision Service 和 Engineering Extensions（工程扩展包）仅在最新版本中提供。因此，这些组件决定了要使用的 TcCOM 和库版本。不允许混合版本，否则会导致故障。

版本选择提示

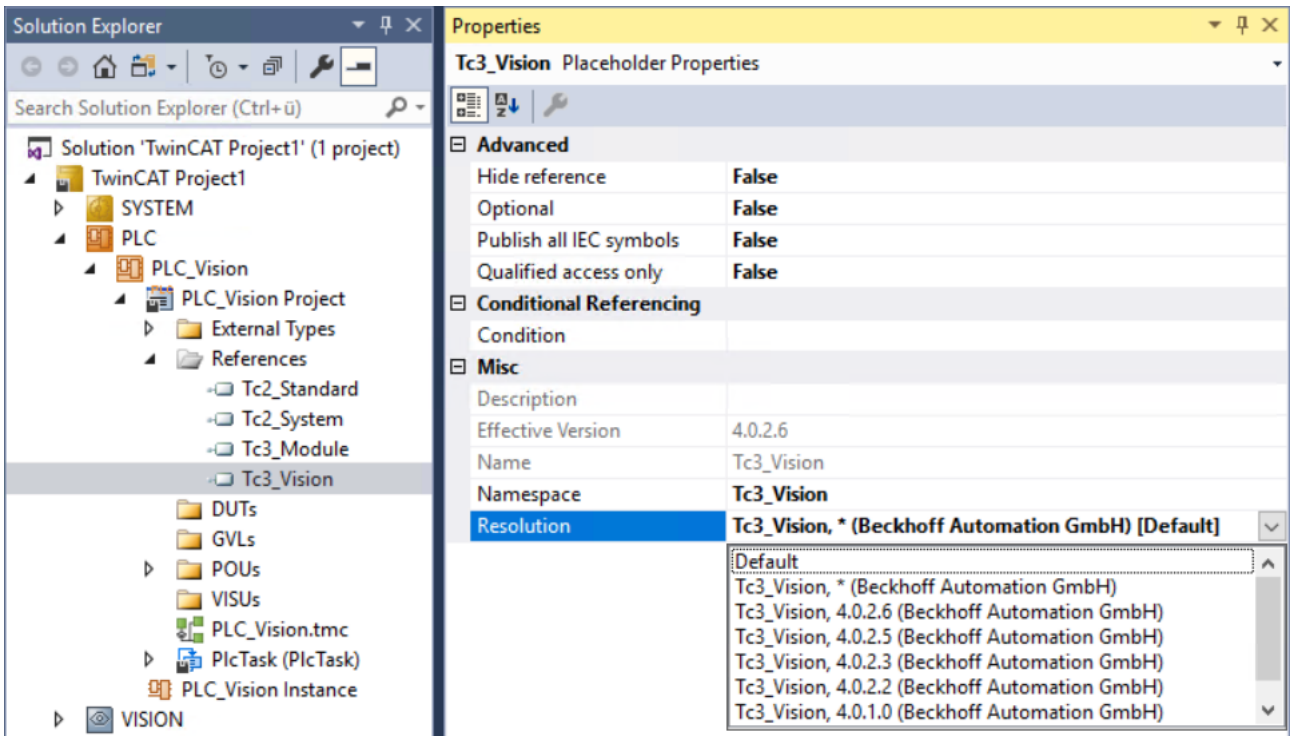
如果要使用特定版本的 TwinCAT Vision，则必须将项目的所有 TcCOM 对象更新到相应版本。Ads Communicator CONFIGONLY 是个例外。该模块仅在 Configuration Assistant（配置助手）内部使用，必须始终拥有与扩展相匹配的最新版本。因此，该模块会在加载项目时自动设置最新版本。

要检查或更改 TcCOM 版本：

1. 打开 **SYSTEM > TcCOM Objects > Project Objects（系统 > TcCOM 对象 > 项目对象）**窗口。
2. 要更改，请为所有对象选择所需版本。
3. 右键点击第一列中的一个对象 ID。
4. 选择 **Reload TMI/TMC Description(s) with changed version（用更改后的版本重新加载 TMI/TMC 说明）**。



此外，PLC 项目中必须使用与所选 TcCOM 对象版本相对应的相同 Tc3_Vision 库版本。因此，在 Tc3_Vision 库的属性中检查有效使用的版本，或对其进行相应调整。



TwinCAT Vision Service 的当前版本号以及 Engineering Extensions（工程设计扩展包）的当前版本号可在 Vision 节点下的 服务配置 [▶ 68] 中找到。

注意

更新后的在线更改

如果在开发系统上更新 TwinCAT Vision，从而在使用旧版本的目标系统上进行了 Online Change（在线更改），这可能会导致接口指针（图像和容器）版本冲突。

删除旧版本

要删除以 4.x.x.x 开头的旧版本，需要两个步骤：

1. 首先，您必须删除“%TWINCAT3DIR%\3.1\Repository\Beckhoff Automation GmbH”或“C:\ProgramData\Beckhoff\TwinCAT\3.1\Repository\Beckhoff Automation GmbH”下的 Tc3_Vision、TcIoGigEVision 和 TcVision 目录中的相应版本文件夹。
2. 要删除相应的 PLC 库版本，还必须利用 Library Repository（库存储库）对话框将其卸载。

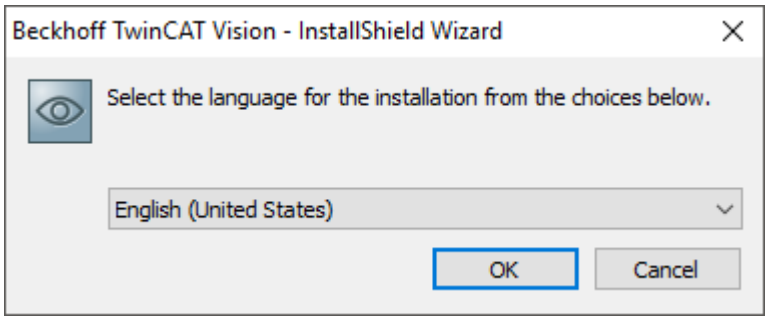
要删除以 5.x.x 开头的旧版本，必须使用以下命令指定版本，卸载相应的软件包。

- 要从 TcIoGigEVision 存储库卸载：
```
tcpkg uninstall TwinCAT.XAE.TMX.GigEVision=5.x.x
```
- 要从 Tc3_Vision 和 TcVision 存储库卸载：
```
tcpkg uninstall TwinCAT.XAE.TMX.Vision=5.x.x
```
- 要卸载经典 OCR 型号：
```
tcpkg uninstall TwinCAT.XAR.OcrModelClassic=5.x.x
```
- 要卸载编码检测模型：
```
tcpkg uninstall TwinCAT.XAR.VisionCodeDetectionModel=5.x.x
```
- 要卸载 5.7.4 及之前版本中交付的过时共享资源，您必须首先从 VisionCommon 软件包中执行 `uninstall`。在此之后，对于 VisionCommon 5.7.4 及之前版本，以及 Vision 软件包的更新版本，必须执行一次 `repair`：
```
tcpkg uninstall TwinCAT.XAE.VisionCommon=5.x.x
# until version 5.7.4
tcpkg repair TwinCAT.XAE.VisionCommon
# newer versions
tcpkg repair TwinCAT.XAE.Vision
```

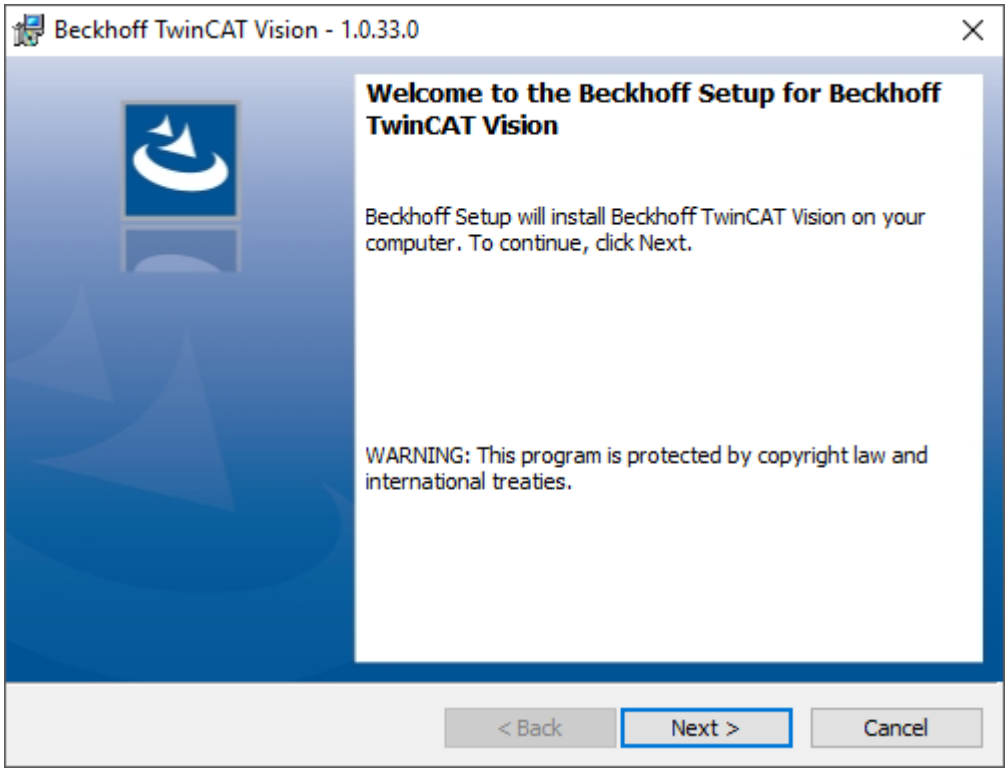
3.3 TwinCAT 3.1 Build 4024 安装说明

在 Windows 操作系统上安装 TwinCAT Vision 包含以下步骤：

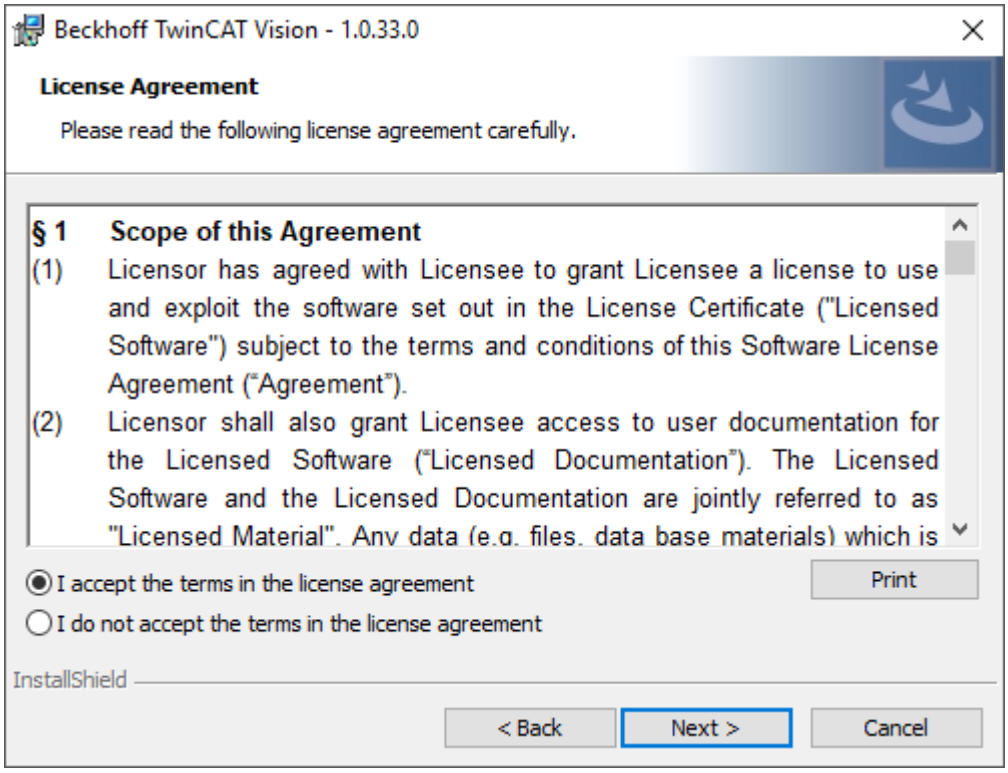
1. 确保 TwinCAT 处于配置模式，且未打开任何 Visual Studio 实例或 TcXaeShell。
2. 以管理员身份运行安装文件 TF7xxx-Vision.exe。为此，请右键点击文件并选择 **Run As Admin（以管理员身份运行）**。点击 **Yes（确定）**确认。
3. 选择安装语言。然后点击 **OK（确认）**。



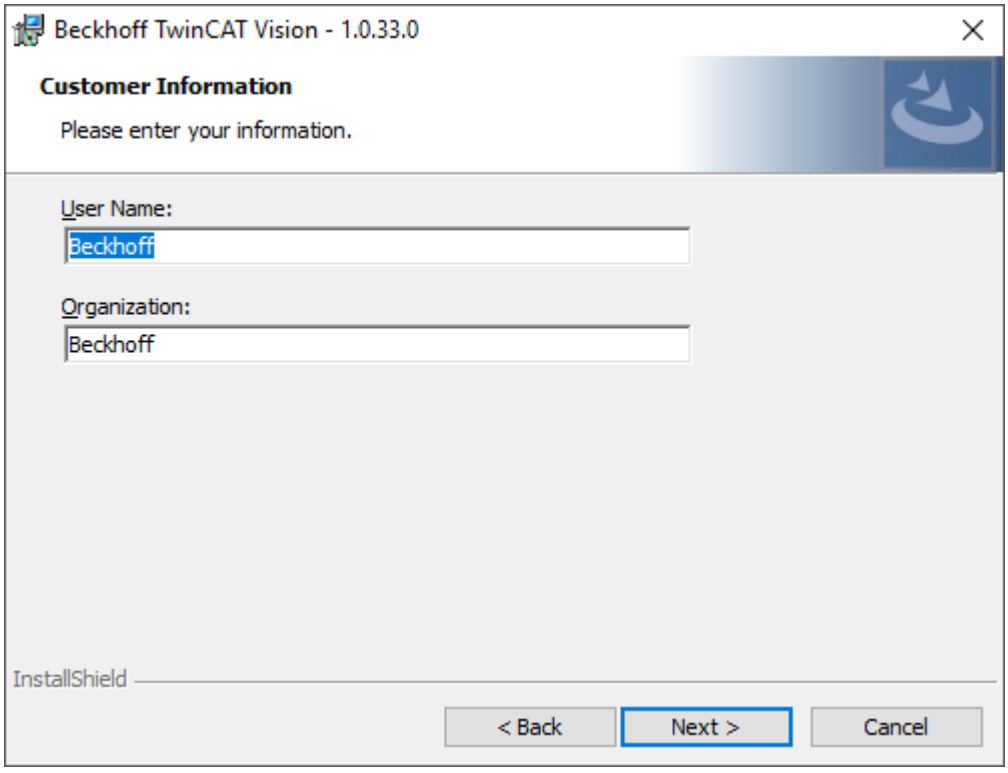
4. 等待安装准备过程结束，然后点击 **Next（下一步）**。



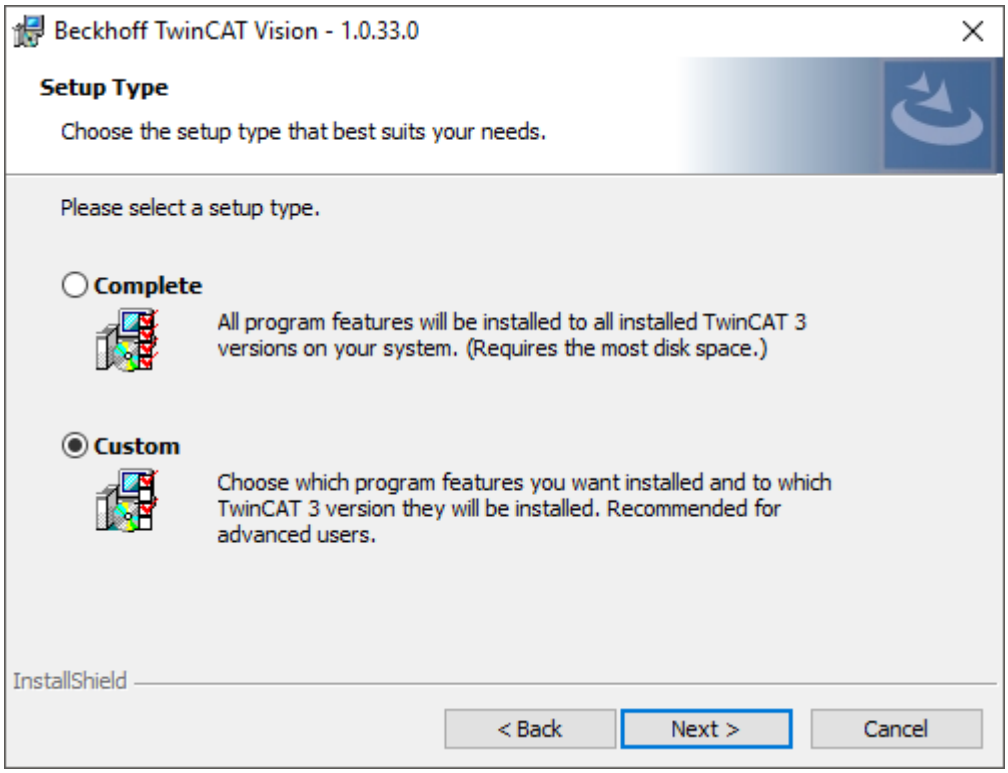
5. 通过勾选 **I accept the terms of the license agreement（我接受许可协议的条款）**选项，确保您同意许可协议。如有需要，您可以**打印**许可协议。然后点击 **Next（下一步）**。



6. 输入用户名和公司作为用户信息。然后点击 **Next（下一步）**。

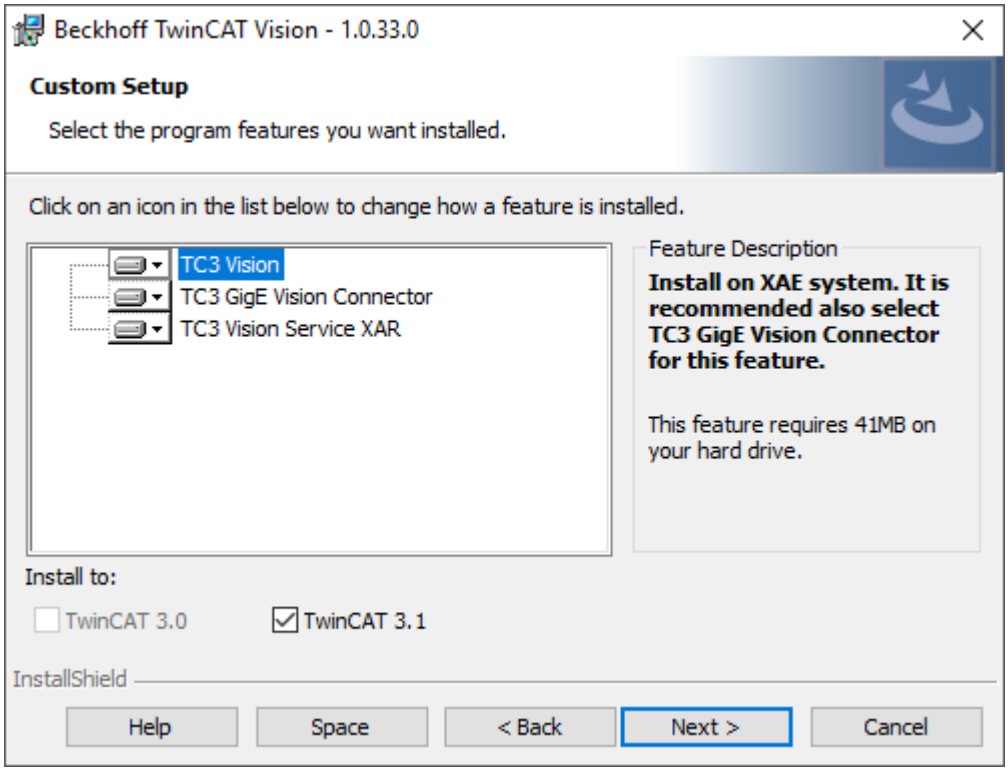


7. 在 **Complete（完整）**或 **Custom（自定义）**两种设置类型中进行选择。然后点击 **Next（下一步）**。



— 完整设置会安装所有 TwinCAT Vision 组件，需要最多的存储空间。建议 XAE 系统选择该选项。由于纯运行时系统上只能安装 Vision Service，其他选项无法选择，因此这种情况也推荐选择该选项。
— 自定义设置可让您在下一步中选择要安装的 TwinCAT Vision 组件。

8. 对于下列每个 TwinCAT Vision 组件，请选择是否安装。此外，请选择至少一个待安装所选组件的 TwinCAT 3 版本。然后点击 **Next（下一步）**。



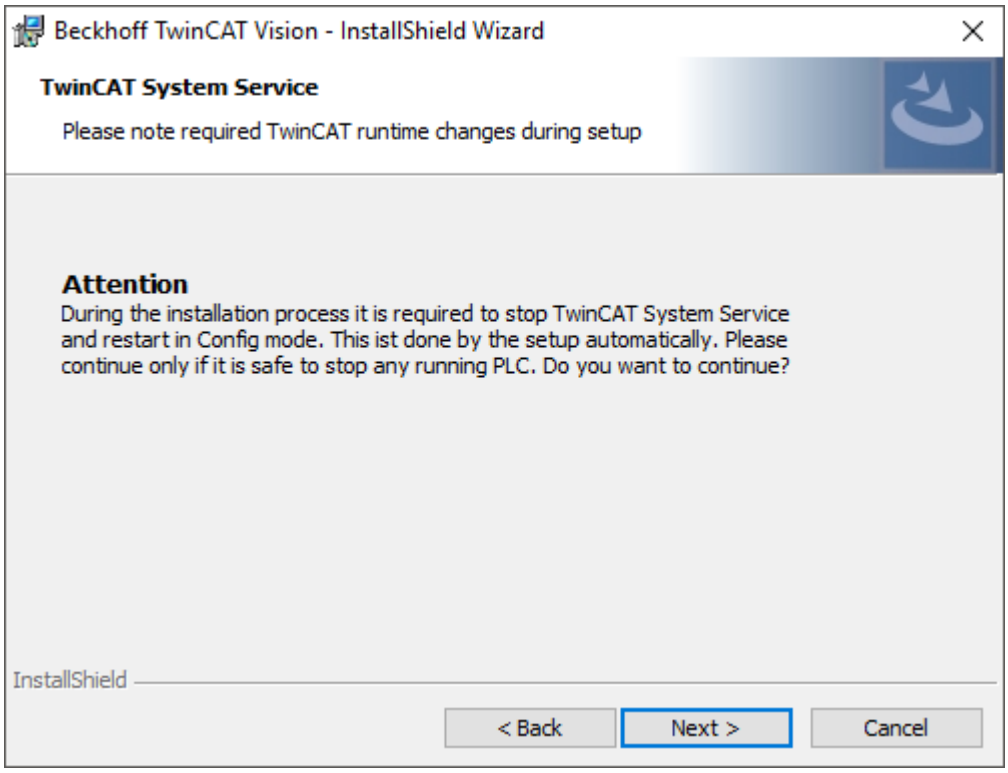
— **TC3 Vision**：为开发 TwinCAT Vision 应用所必需，并应安装在 XAE 系统上。

— **TC3 GigE Vision 连接器**：为使用 GigE Vision 设备进行开发所必需，并应安装在 XAE 系统上。
— **TC3 Vision Service XAR**：包含 TwinCAT Vision Service，且必须安装在 TwinCAT Vision 应用的目标系统上。

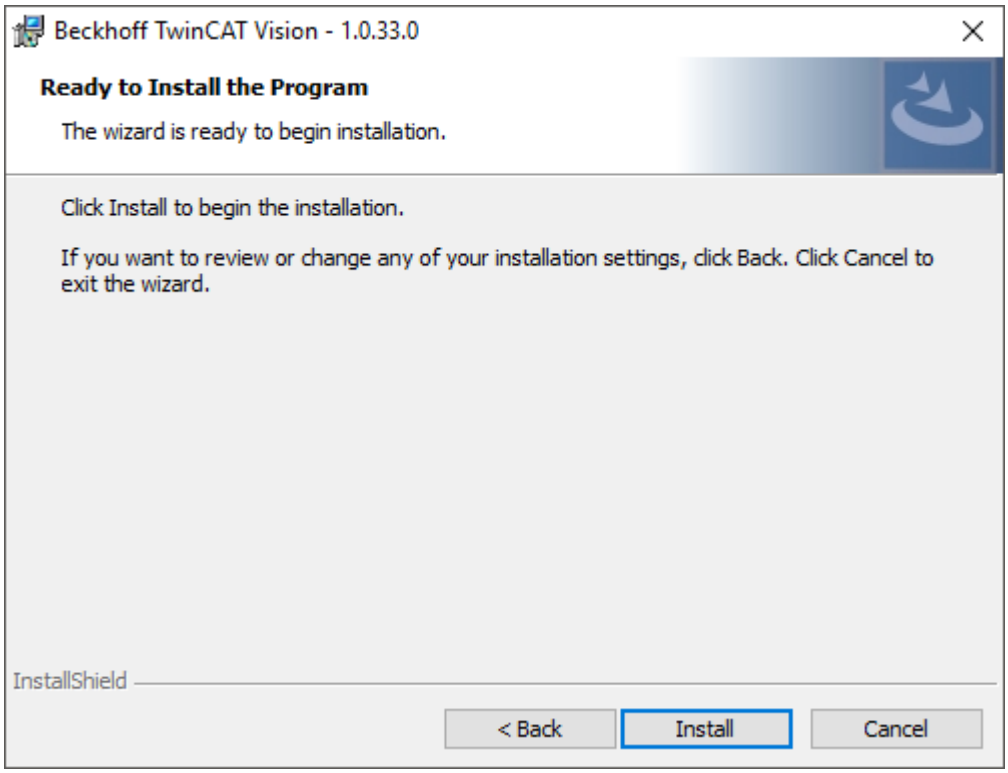
说明

目标系统上必须安装 Vision Service。相反，在激活 TwinCAT 项目时，TC3 Vision 和 TC3 GigE Vision 连接器的组件总是自动从开发系统复制到目标系统。因此，它们并不一定要安装在目标系统上。

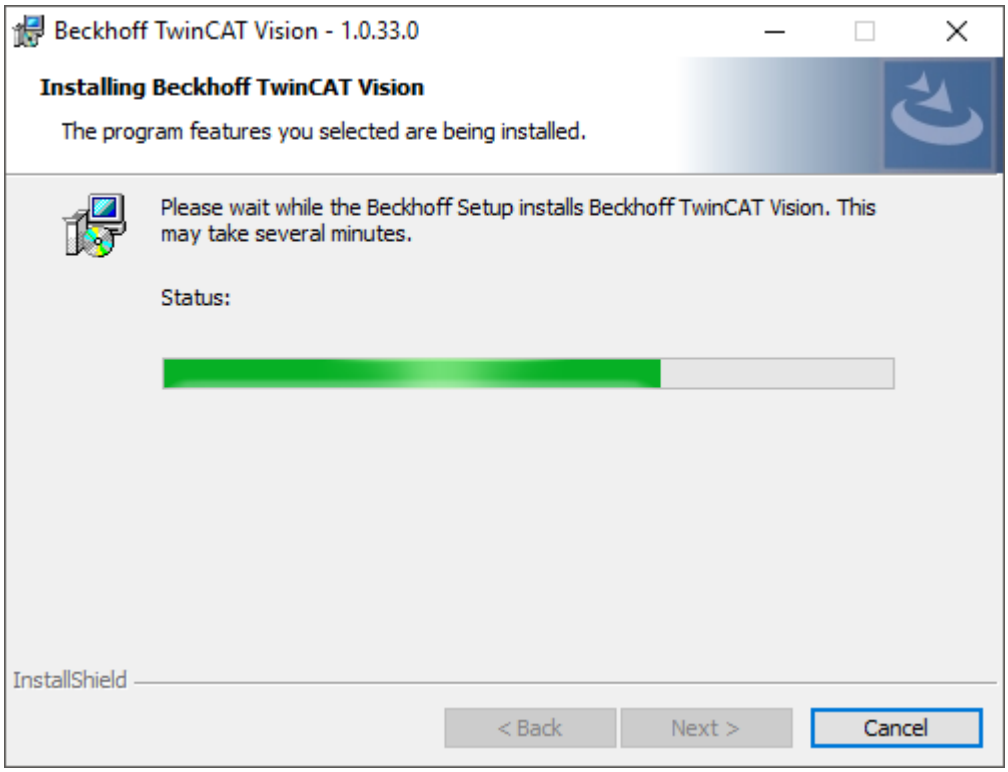
9. 请注意，在接下来的安装过程中，TwinCAT 系统服务将停止并以配置模式重新启动。只有当运行中的 PLC 系统可以停止时，才能继续安装。在这种情况下，点击 **Next（下一步）**。



10. 确保您同意所选的安装设置。然后点击 **Next（下一步）**。

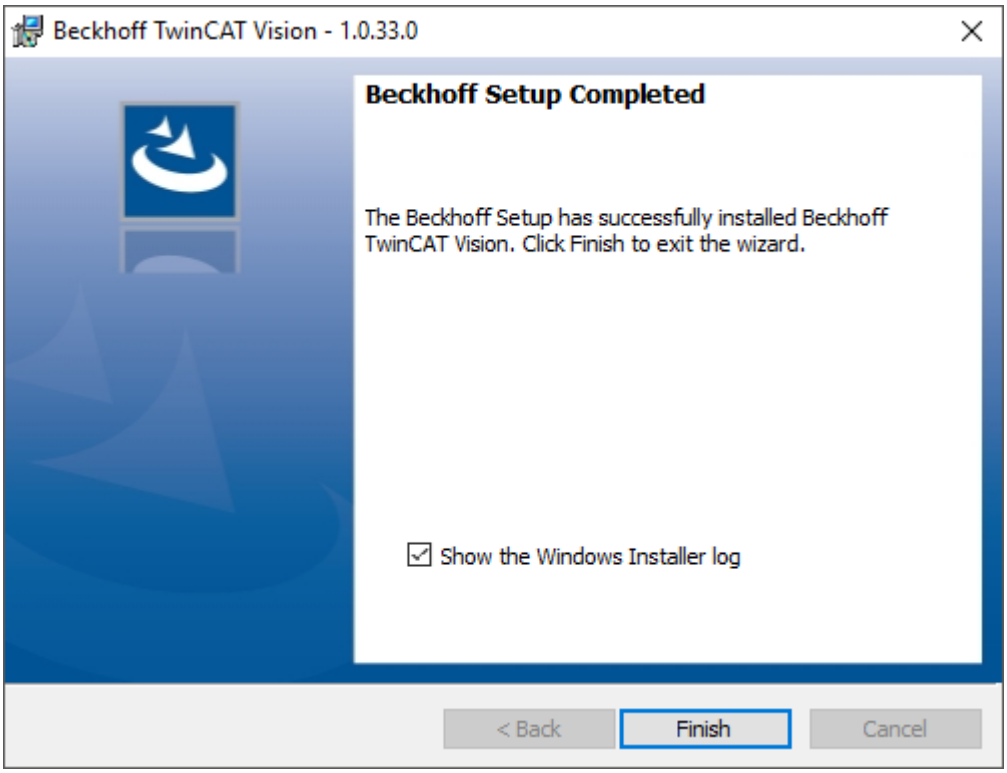


11. 等待所选 TwinCAT Vision 组件安装完毕。



12. 可能会出现一个对话框，显示为完成安装而必须关闭的应用列表。选择自动关闭应用还是在安装后重启 PC。

⇨ 安装现已完成。可以选择显示安装的日志文件。点击 **Finish（完成）**。



TwinCAT Vision 安装成功后必须先获得许可方可使用。该程序在许可 [▶ 24]一章中有所描述。

有关版本和更新现有项目的信息，请访问 版本概览 [▶ 15]。

如果安装失败，您有以下选择：

- 使用故障排除 [▶ 2842]解决问题。
- 联系客户支持部并提供安装日志。安装日志 TF7xxx-Vision.exe.log 位于路径：C:\Users\<username>\ 中的以下文件夹中

3.4 TwinCAT 3.1 Build 4026 及以上版本的安装说明

TwinCAT Package Manager

如果在 Microsoft Windows 操作系统上使用 TwinCAT 3.1 Build 4026（及更高版本），则可以通过 TwinCAT Package Manager 安装 TwinCAT Vision，请参见 Installing TwinCAT 3.1 Build 4026（安装 TwinCAT 3.1 Build 4026）。

通常情况下，可通过相应的 Workload（工作负载）安装 TwinCAT Vision；然而，您也可以单独安装 Workload（工作负载）中包含的软件包。本文档简要介绍了通过工作负载进行安装的过程。

命令行程序 TcPkg

您可以使用 TcPkg **C**ommand **L**ine **I**nterface（命令行界面，CLI）来显示系统上的可用工作负载：

```
tcpkg list -t workload
```

若要仅显示 TwinCAT Vision Workloads（工作负载），可以使用此命令：

```
tcpkg list TF7
```

您可以使用以下命令安装 TF7000-TF7810 Vision 功能的 Workloads（工作负载）。

```
tcpkg install TF7xxx.Vision.XAE
tcpkg install TF7xxx.Vision.XAR
```

TwinCAT Package Manager UI

您可以使用 **U**ser **I**nterface（用户界面，UI）显示所有可用的工作负载，并根据需要进行安装。
为此，请按照界面中的相应说明操作。

3.5

3.6 使用 TwinCAT 3 Usermode Runtime

TwinCAT 3.1 Build 4026 可提供 Usermode Runtime。要将 TwinCAT Vision 与 TC170x | TwinCAT 3 Usermode Runtime 结合使用，必须遵守下述配置。有关 TwinCAT 3 Usermode Runtime 的安装、功能和局限性的基本说明，请参见相关文档。

启动 Vision 服务

TwinCAT 3 Vision 使用 Vision 服务实现与硬盘的所有交互。例如，如果需要加载或保存图像（这也适用于 FileSource 的使用），则必须由 TwinCAT 3 Usermode Runtime 启动 Vision 服务。

为此，必须在 Usermode Runtime 的 3.1\Target 目录下创建包含以下内容的文件 StartManConfig.xml，如果有扩展路径，则在默认路径下创建：“C:\ProgramData\Beckhoff\TwinCAT\3.1\Runtimes\UmRT_Default\3.1\Target”。

```
<StartMan>
    <TwinCATServers>
        <TwinCATServer Enabled="1">
            <Name>TcVnService</Name>
            <Path>C:\Program Files
(x86)\Beckhoff\TwinCAT\3.1\Components\Vision\TcVision\TcVnService.exe</Path>
        </TwinCATServer>
    </TwinCATServers>
</StartMan>
```

使用相机

要通过 RT 以太网适配器集成相机，必须以管理员权限启动 Usermode Runtime：

1. 例如，以管理员身份启动 Windows PowerShell
2. 切换到 Usermode Runtime 文件夹，默认路径为“C:\ProgramData\Beckhoff\TwinCAT\3.1\Runtimes\UmRT_Default”，然后执行“Start.bat”：
   ```
   cd C:\ProgramData\Beckhoff\TwinCAT\3.1\Runtimes\UmRT_Default
   .\Start.bat
   ```

⇨ 然后可通过 RT 以太网适配器集成相机。

由于 TwinCAT 3 Usermode Runtime 无法保证具有确定的运行时行为，因此相机连接可能会出现问题，具体取决于所使用的硬件。请参阅 相机通信 [▶ 2853] 上的诊断选项。

如果出现问题，可能需要增加 CameraTcCOM-GVCP [▶ 100] 模块的以下参数：

- `TransmissionTimeout`
- `HeartbeatRate`
- `DiscoveryTimeout`

请注意，如果 GVCP 模块中的超时时间增加，那么 操作和设置 [▶ 87] 中的 ADS 通信超时时间可能也需要增加。超时时间应大于 MaxTimeouts 和 TransmissionTimeout 的乘积。

增加路由器内存

TwinCAT Usermode Runtime 的路由器内存可以在 TwinCAT 系统配置中进行设置。然而，可调整的大小受 Usermode Runtime 配置的堆内存限制。因此，所选堆内存必须大于要设置的路由器内存。堆内存可以在 Usermode Runtime 的 \3.1 目录中的 TcRegistry.xml 中进行调整，默认路径为：“C:\ProgramData\Beckhoff\TwinCAT\3.1\Runtimes\UmRT_Default\3.1\TcRegistry.xml”

为此，必须对 XML 文件进行如下调整：

1. 停止 Usermode Runtime。

2. 打开 TcRegistry.xml。
3. 调整 XML 文件中的 `<Value Name="HeapMemSizeMB" Type="DW">{size in MB}</Value>` 条目。
4. 大小以兆字节为单位进行设置。在本示例中，这些数据为 1024 MB。

```
---snipped---
<Key Name="System">
    <Value Name="RunAsDevice" Type="DW">1</Value>
    <Value Name="AmsNetId" Type="BIN">C0A804010101</Value>
    <Value Name="HeapMemSizeMB" Type="DW">1024</Value>
---snipped---
```

3.7 授权

TwinCAT 3 功能可以作为完整版或 7 天测试版激活。这两种许可证类型都可以通过 TwinCAT 3 开发环境（XAE）激活。

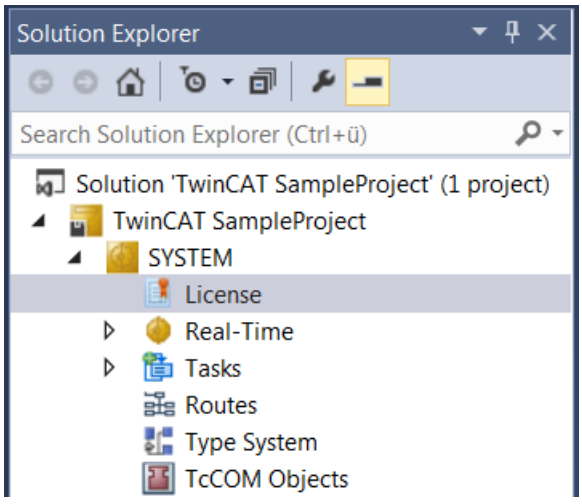
授权使用 TwinCAT 3 功能的完整版本

关于许可证完整版本的程序描述，可参见倍福信息系统的文档“TwinCAT 3 授权”。

授权一个 TwinCAT 3 功能的 7 天测试版本

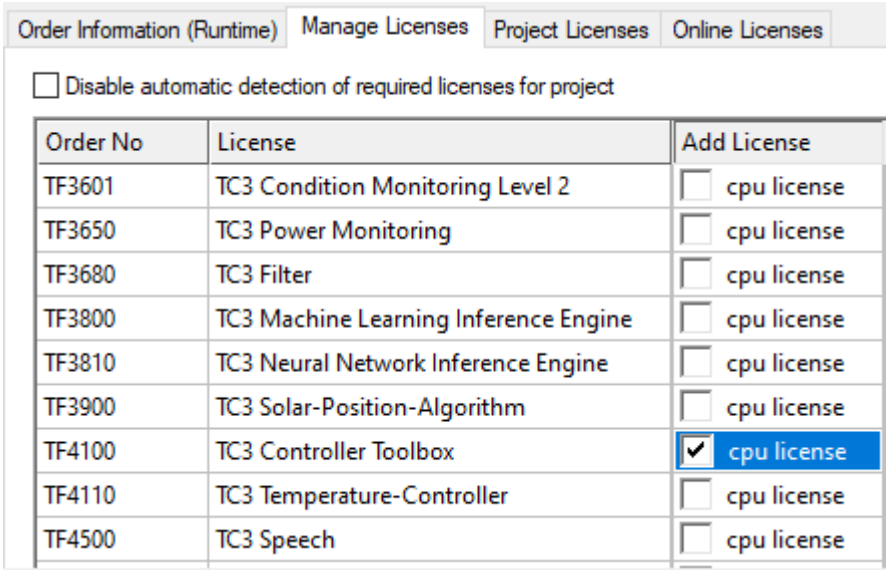
对于TwinCAT 3 许可证加密狗无法启用 7 天测试版本。

1. 启动 TwinCAT 3 开发环境（XAE）。
2. 打开一个现有的 TwinCAT 3 项目或创建一个新项目。
3. 如果想为一个远程设备激活授权，请设置所需的目标系统。为此，从工具栏的**选择目标系统**下拉列表中选择目标系统。
   ⇨ 授权设置总是指选定的目标系统。在目标系统上激活项目时，自动将相应的 TwinCAT 3 许可证复制到该系统中。
4. 在**解决方案资源管理器**中，双击**系统**子目录的**许可证**。

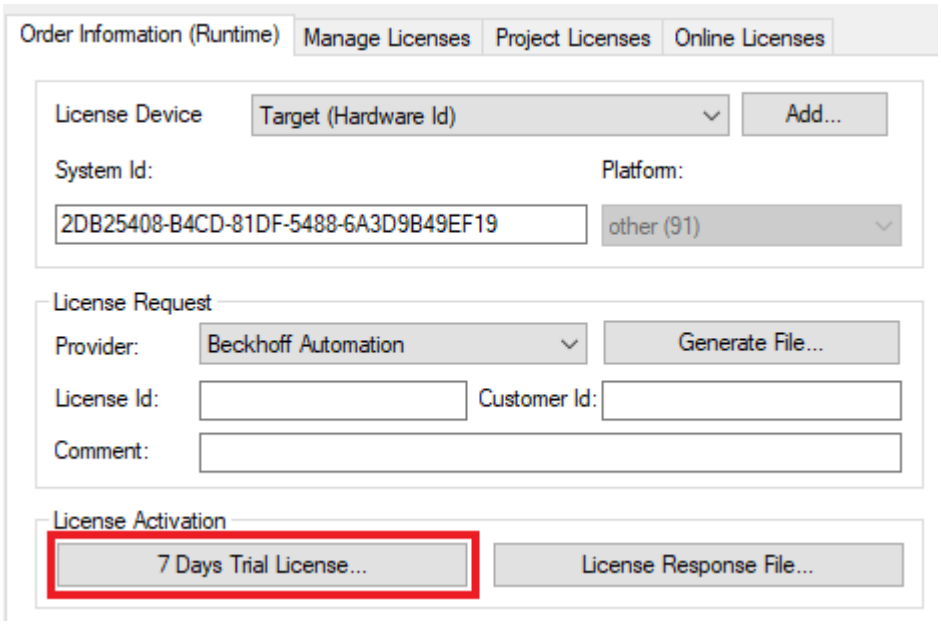


⇨ TwinCAT 3 许可证管理器打开。

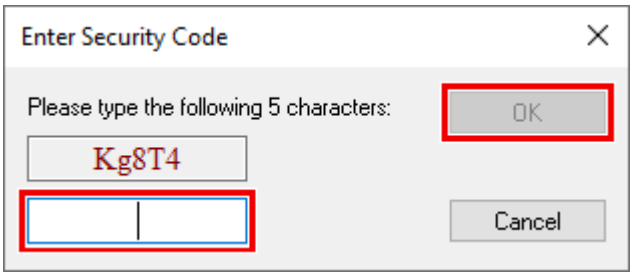
5. 打开**管理许可证**选项卡。在**添加许可证**栏中，选中希望添加到项目的许可证的复选框（例如“TF4100 TC3 控制器工具箱”）。



6. 打开**订单信息（运行时间）**选项卡。
   ⇨ 在许可证的表格概览中，以前选择的许可证显示为“缺失”状态。
7. 点击**7 天试用许可证...**，以激活 7 天试用许可证。



   ⇨ 一个对话框打开，提示输入对话框中显示的安全代码。



8. 准确输入显示的代码并确认输入。
9. 确认随后的对话框，表明激活成功。
   ⇨ 在许可证的表格概览中，许可证状态现在显示许可证的到期日。
10. 重新启动 TwinCAT 系统。

⇨ 7 天试用版启用。

4 第一步

以下指南对 TwinCAT Vision 进行了介绍（成功安装 [▶ 14]后）。

1. 构建项目 [▶ 27]
   - 创建 TwinCAT 项目 [▶ 27]
   - 选择目标系统 [▶ 29]
   - 系统配置 [▶ 31]
2. 显示一个设置视觉设备 [▶ 33]，并创建一个视觉设备
   - 显示视觉节点并创建应用程序 [▶ 33]
   - 创建文件源控制 [▶ 35]或
   - 创建 GigE Vision 相机 [▶ 38]
3. 创建 PLC 项目 [▶ 45]
   - 创建 PLC 项目 [▶ 45]
   - 集成 PLC 库 [▶ 46]
   - 编写 PLC 程序 [▶ 48]
   - 初始化功能块 [▶ 49]
4. 运转 [▶ 50]
   - 激活 TwinCAT 项目并启动 PLC 项目 [▶ 50]
   - ADS 图像查看中的图像显示 [▶ 51]
   - 下载示例项目：https://github.com/Beckhoff/TF7xxx_Samples

关于更多详细信息，可参见开发环境 [▶ 53]、API 参考 [▶ 132]和样本 [▶ 2722]。

4.1 构建项目

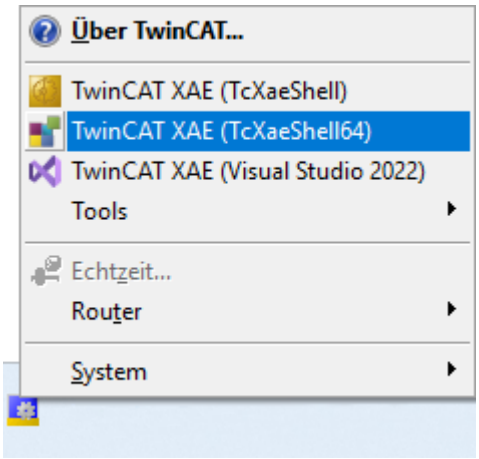
为 TwinCAT Vision 设置系统的视频示例：

视频：https://infosys.beckhoff.com/content/1033/tf7xxx_tc3_vision/Resources/5906036107.mp4

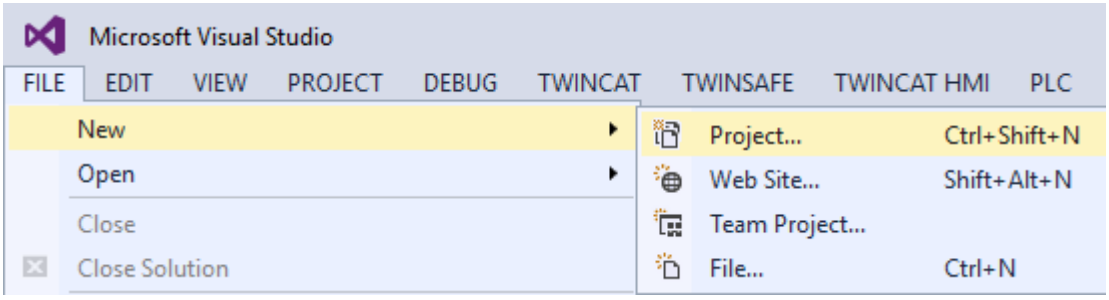
4.1.1 创建 TwinCAT 项目

TwinCAT Vision 完全集成在 TwinCAT 系统中。因此，一个标准的 TwinCAT 项目是每个 TwinCAT Vision 应用的出发点：

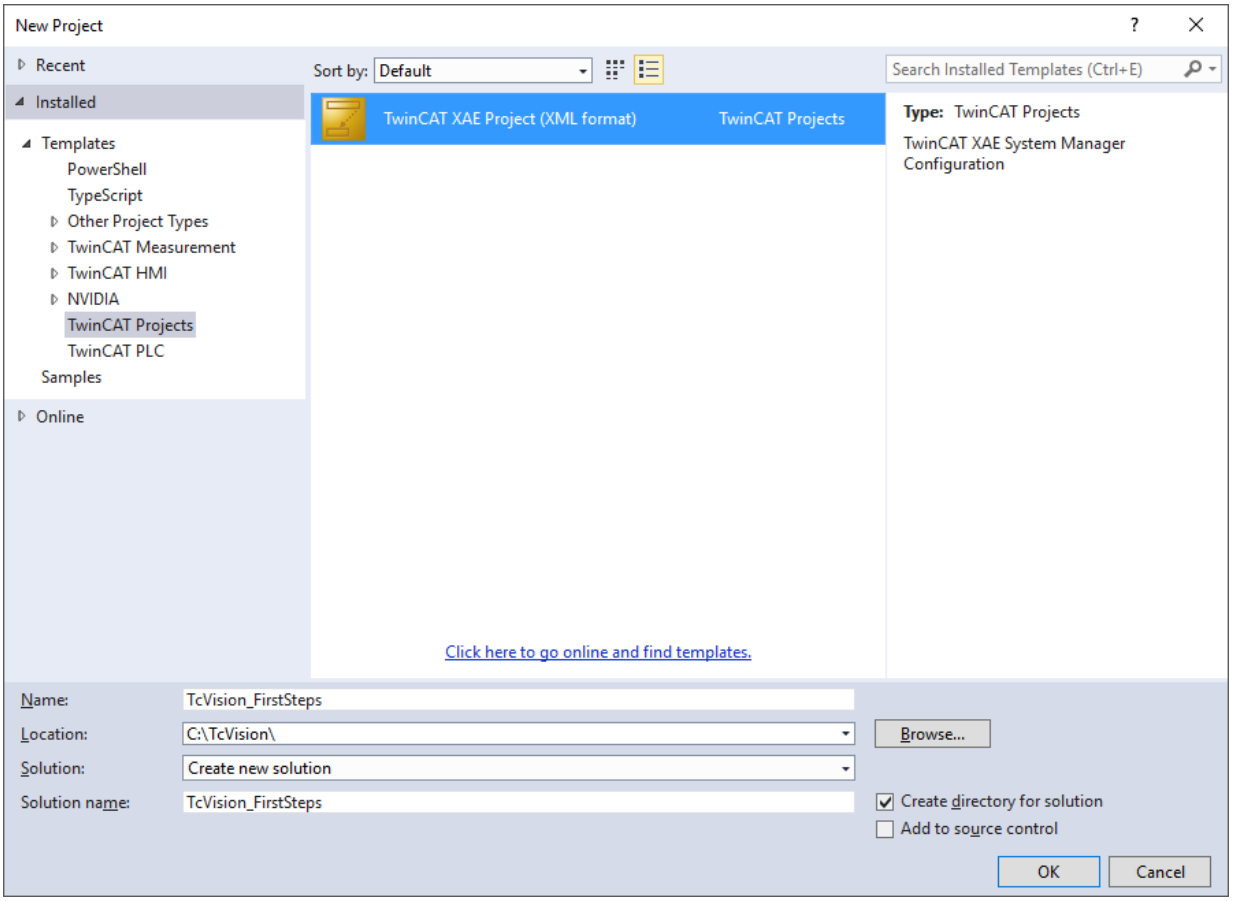
1. 通过集成 TwinCAT 打开 Visual Studio。



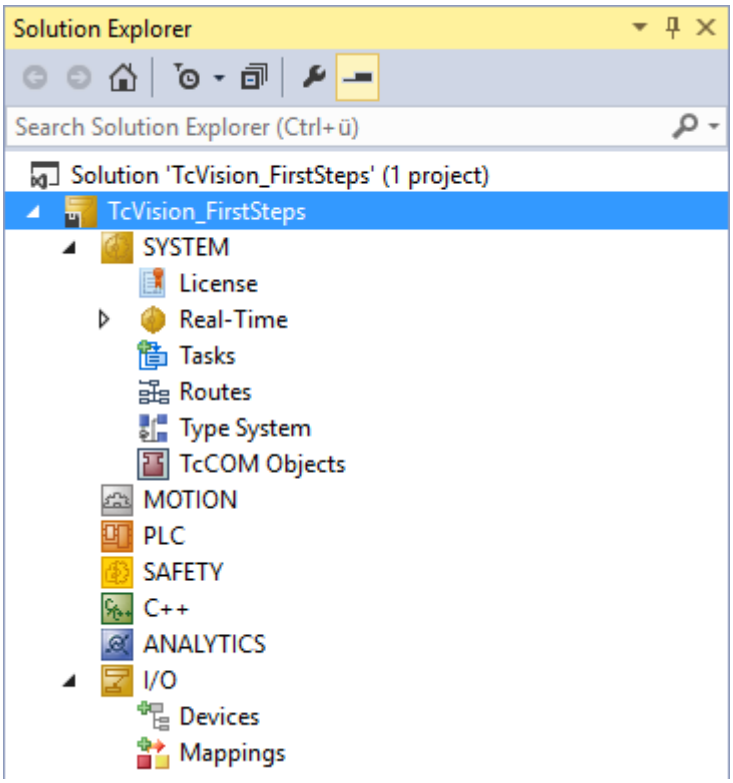
2. 打开**新/项目**对话框。



3. 选择名称和地点。



⇨ TwinCAT 项目已创建。



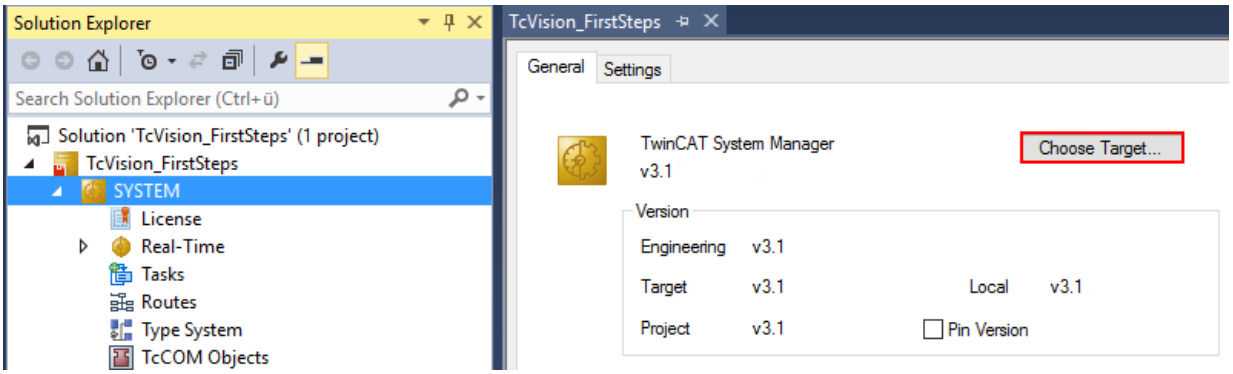
下一步

选择目标系统 [▶ 29]

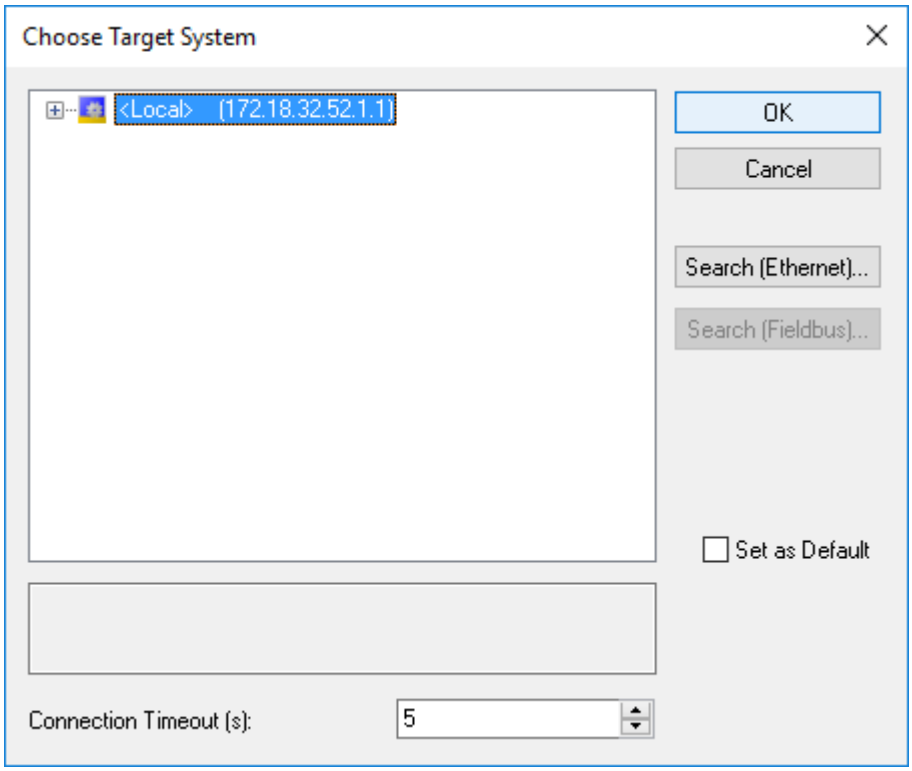
4.1.2 选择目标系统

TwinCAT 区分了开发系统和运行时间系统。如果你的开发系统不是你的运行时系统，创建一个通往运行时系统的路线并选择路线。例如，这可以通过以下步骤完成：

1. 点击**解决方案资源管理器 > 系统 > 通用 > 选择目标...**
   选择目标系统对话框打开，其中列出了所有从开发系统中存在路线的目标系统。

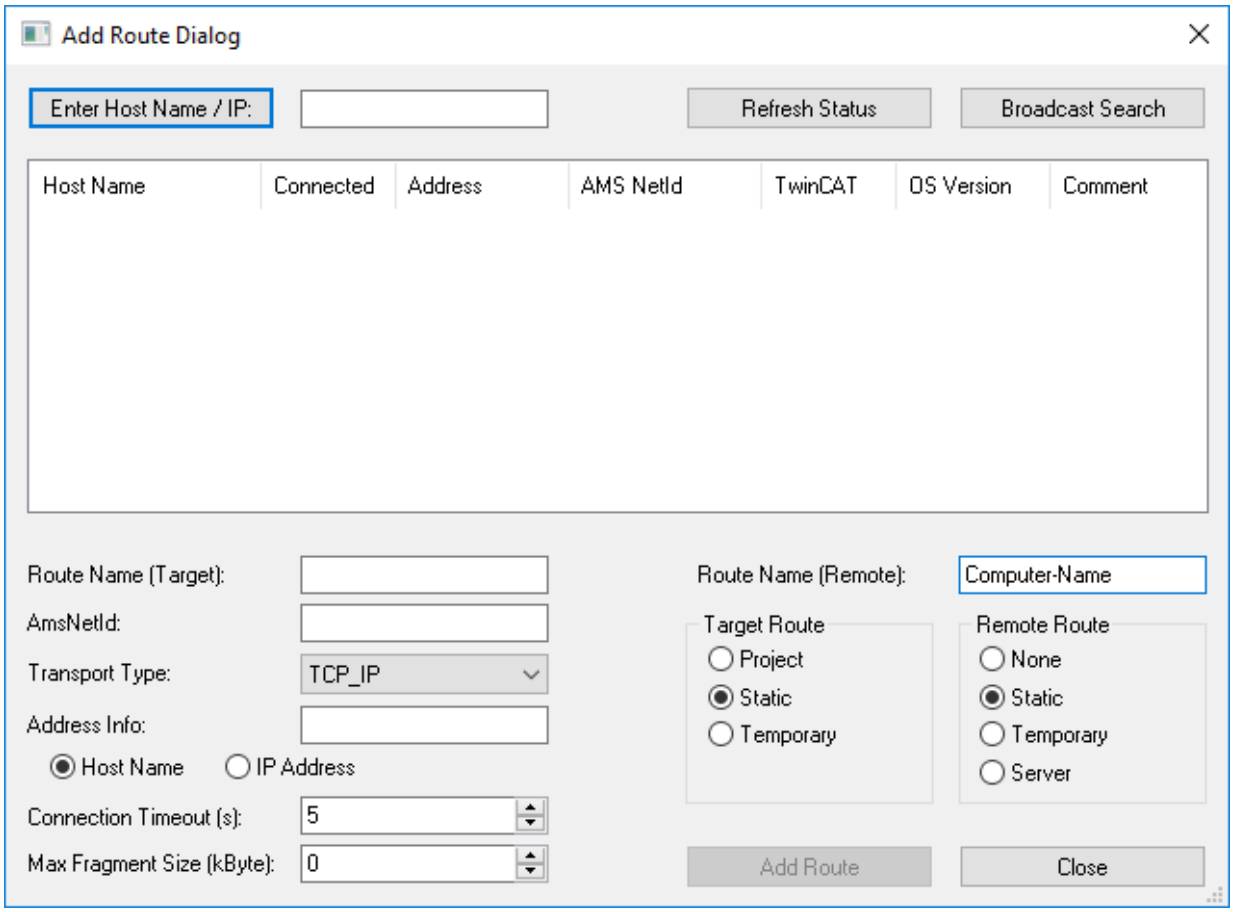


2. 如果目标系统尚未列出，即不存在通往系统的路线，点击**搜索**。

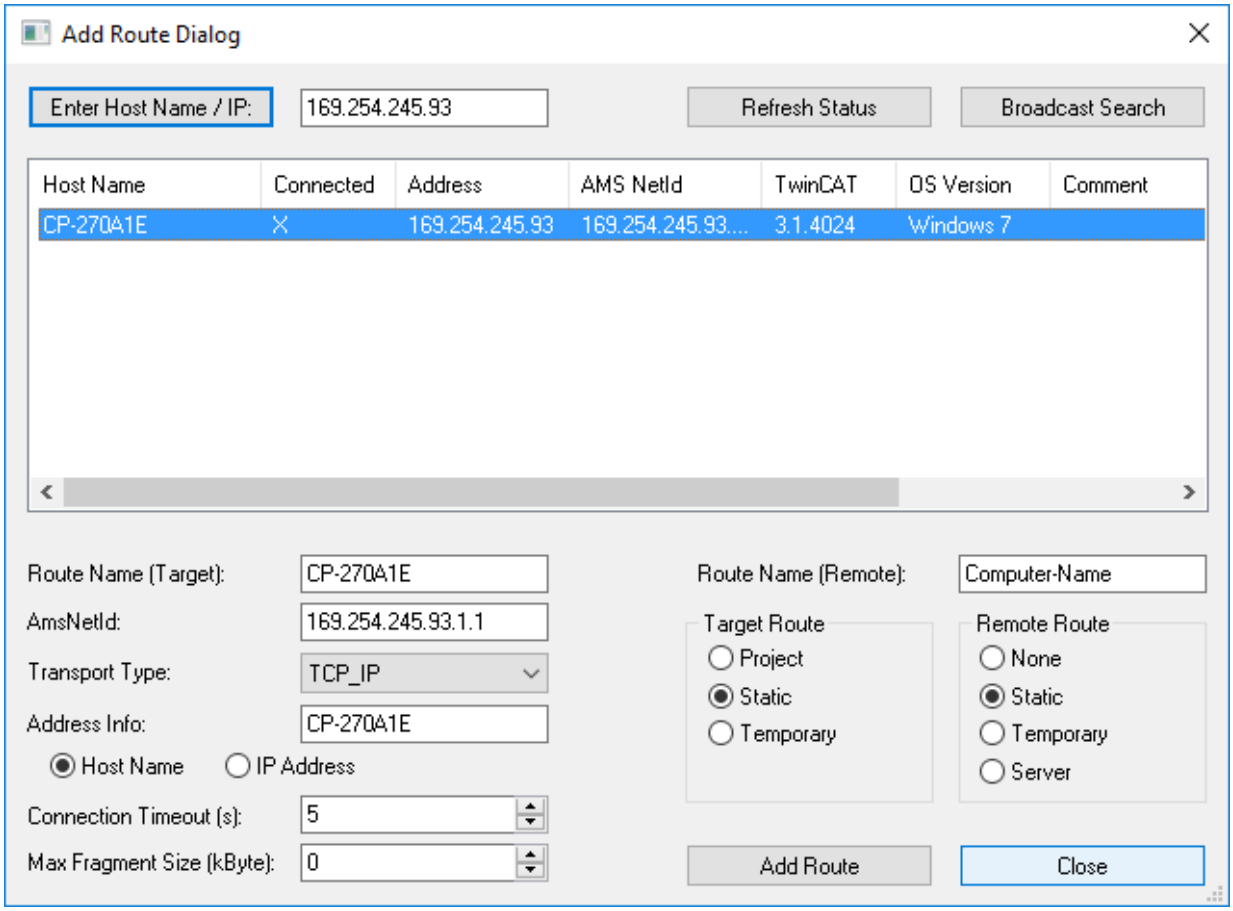


“添加路线”对话框打开，而在这里可以添加更多路线。

3. 在**添加路线对话框**中，点击**广播搜索**，以搜索网络中的 TwinCAT 系统。或者，通过**输入主机名称/IP**，搜索设备名称或 IP 地址。然后显示找到的任何系统。



4. 从列表中选择一个系统，然后点击**添加路线**，为其创建一个路线。显示的信息和设置选项在添加路线对话框章中有详细描述。新路线现在出现在**选择目标系统**对话框中。



5. 在**选择目标系统**对话框中选择目标系统，并通过**确定**确认。

下一步

系统配置 [▶ 31]

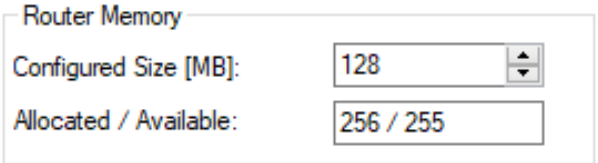
4.1.3 系统配置

系统配置取决于目标系统，详见 系统配置 [▶ 53] 章节。

增加路由器内存

例如，TwinCAT Vision 在创建图像时需要动态内存分配。这在路由器内存 [▶ 54]中进行。因此，必须配置足够大的路由器内存。

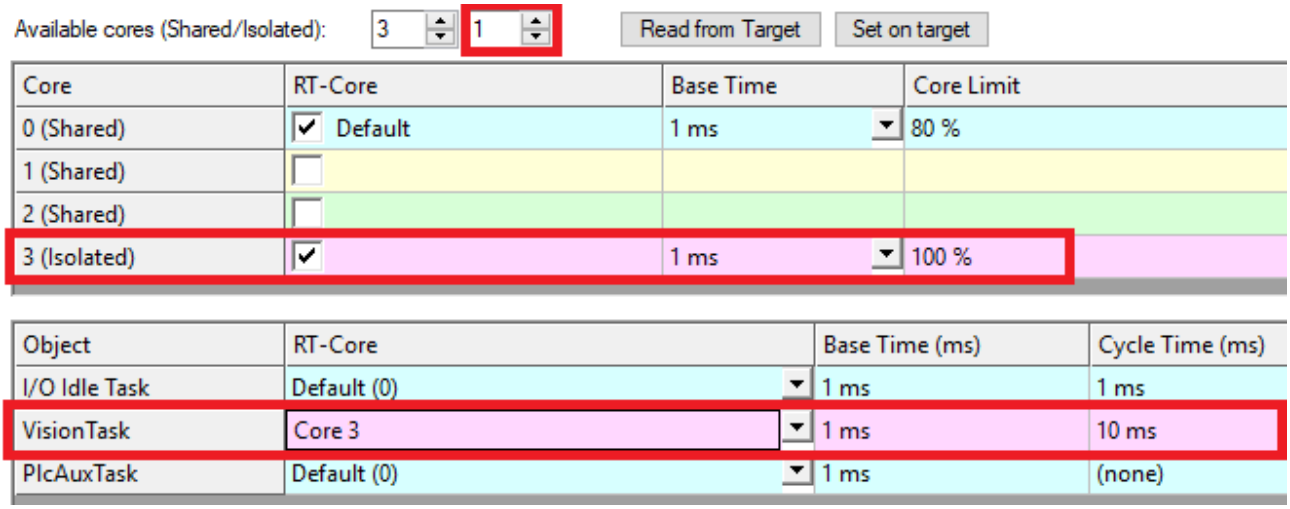
这可以在 TwinCAT 项目中的 **SYSTEM（系统） > Real-Time（实时） > Settings（设置） > Router Memory (Mbyte)（路由器内存，兆字节）**下完成。



然后必须激活配置  。如果路由器内存减少，则必须重新启动目标系统。

选择 CPU 核和任务

在多核系统中，TwinCAT 支持区分 Windows 系统分享核和独立核。使用 Windows 核时，操作系统和 TwinCAT 应用共享处理器时间。实时部分可限制为 10% 到 90%。与此相反，独立核可完全用于 TwinCAT 应用。因此，建议 Vision 应用使用独立核。在初始测试中，也可以使用分核而没有任何问题。



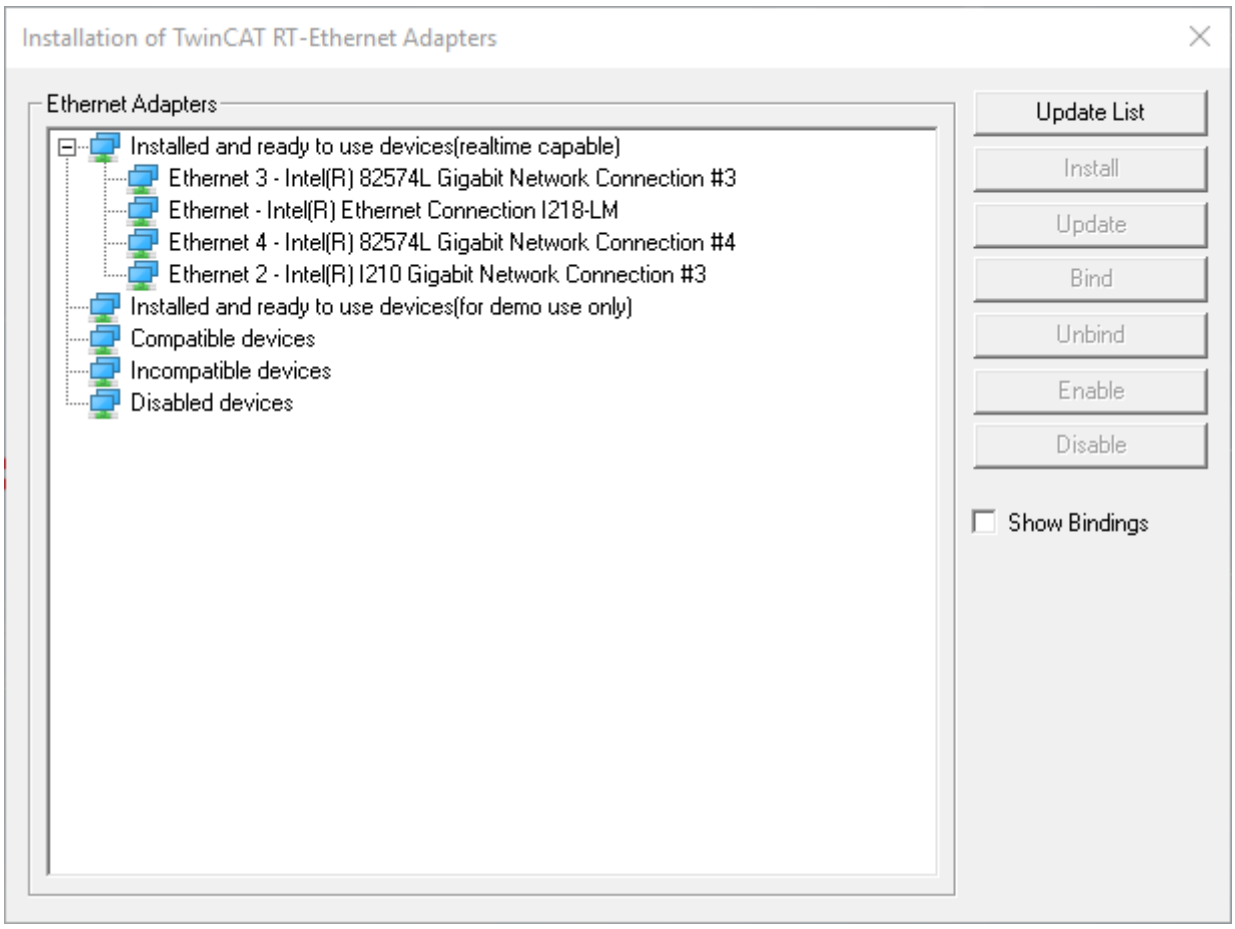
核配置在 TwinCAT 项目树中的 **SYSTEM（系统） > Real-Time（实时） > Settings（设置）** 下进行。详细配置说明见 CPU 内核和任务 [▶ 63] 章节。本章还介绍了如何更改执行任务的循环时间。

然后必须激活配置  。此外，在重新定义分核和独立核时，必须重新启动目标系统。

安装实时以太网驱动程序

要将 GigE Vision 相机连接到 TwinCAT Vision，必须在所使用的网络适配器上安装 TwinCAT RT 以太网驱动程序。该驱动程序可确保 TwinCAT 在运行时期间使用网络适配器。详情请参见 网络适配器 [▶ 56] 章节。

在开发环境中，可通过 **TWINCAT > Show Realtime Ethernet Compatible Devices...（显示实时以太网兼容设备……）** 调用安装助手。在目标系统上，可以通过文件 C:\TwinCAT\3.1\System\TcRteInstall.exe 启动该助手。可为打开对话框中的 **Compatible devices（兼容设备）** 下所列的网络适配器安装 RT 以太网驱动程序。



确保要连接相机的所有网络适配器现在都在**已安装即用设备（支持实时功能）**列表中。

下一步

显示视觉节点并创建应用程序 [▶ 33]

4.2 设置视觉设备

使用文件源控制的视频示例：

视频：https://infosys.beckhoff.com/content/1033/tf7xxx_tc3_vision/Resources/5931182859.mp4

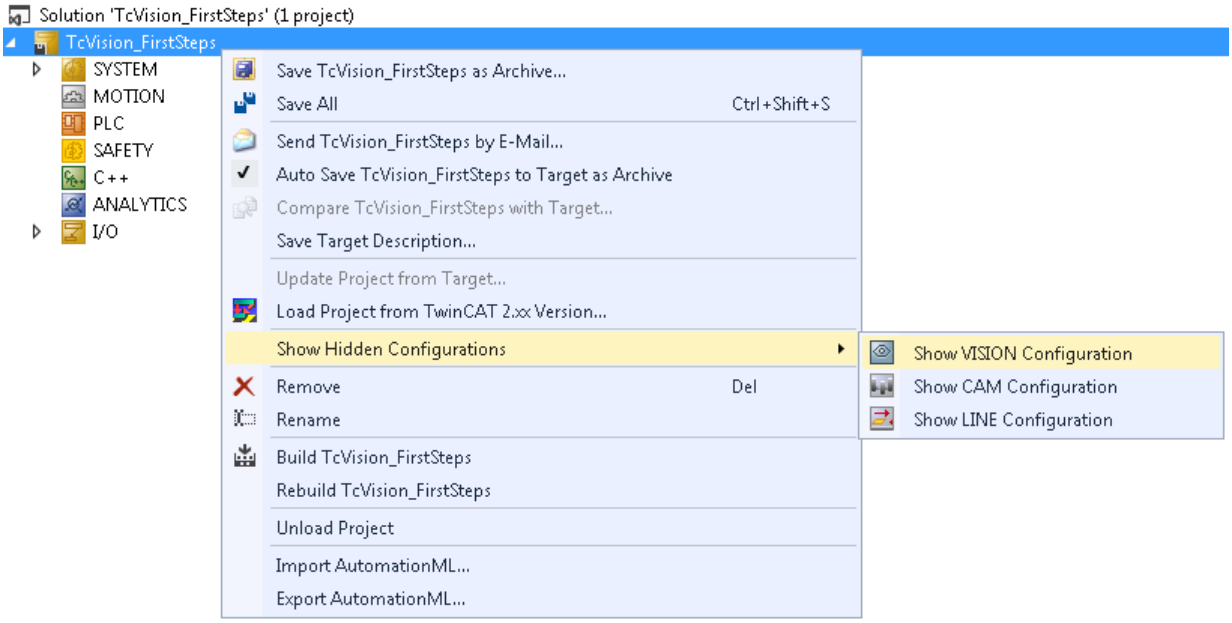
创建相机实例的视频示例：

视频：https://infosys.beckhoff.com/content/1033/tf7xxx_tc3_vision/Resources/5931185163.mp4

4.2.1 显示视觉节点并创建应用程序

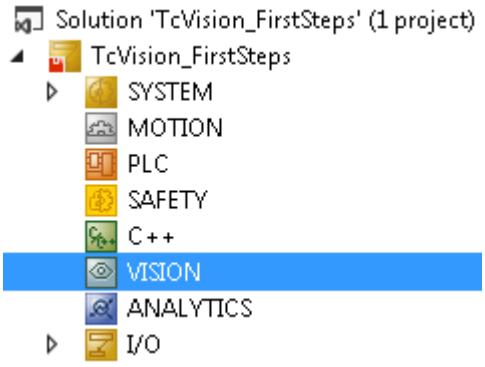
在 TwinCAT 项目中有一个可用于 Vision 配置的 Vision 节点。

1. 如果 TwinCAT 项目中未显示 Vision 节点，请右键点击 Solution Explorer（解决方案资源管理器）>**Show Hidden Configurations（显示隐藏配置）** > **Show VISION Configuration（显示 VISION 配置）**

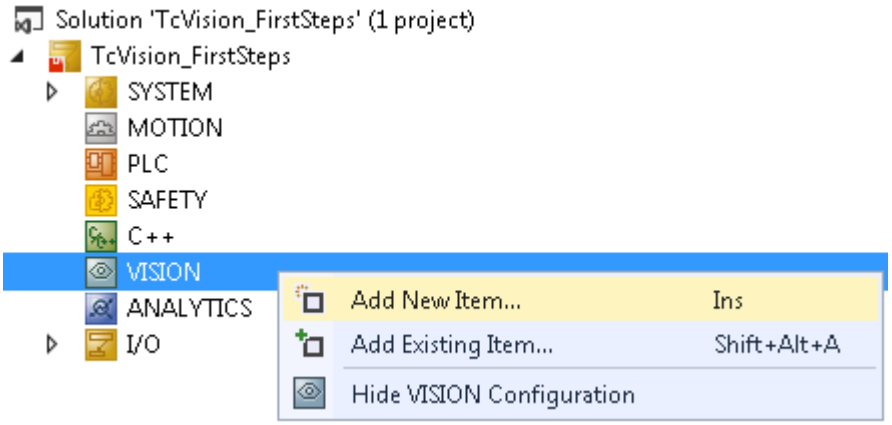


中的 TwinCAT 项目。

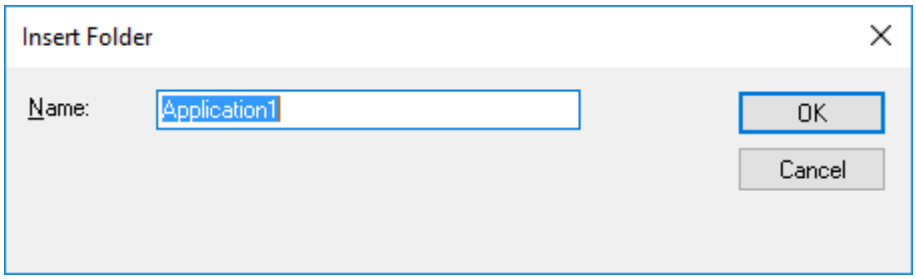
⇨ Vision 节点现在可见。



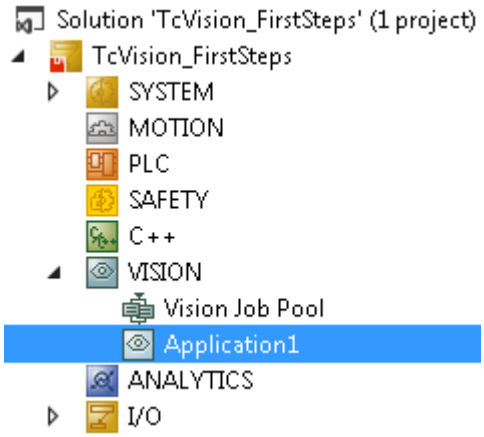
2. 在 Vision 节点内，文件夹（也称为应用）用于对图像处理设备进行分组。右键点击 **Vision Node（Vision 节点）** > **Add New Item（添加新项目）**。



3. 选择一个名称并使用 **OK（确定）**确认。



⇨ 文件夹已创建



此外，您还可以在此找到更多信息：

- 视觉节点 [▶ 68]
- 应用节点 [▶ 70]

下一步

下一步可创建一个可用于将图像加载到 PLC 的 Vision 设备。既可以是真正的 GigE Vision 相机，也可以是用于加载现有图像文件的文件源控制器：

创建 GigE Vision 相机 [▶ 38] 或者 创建文件源控制 [▶ 35]

4.2.2 创建文件源控制

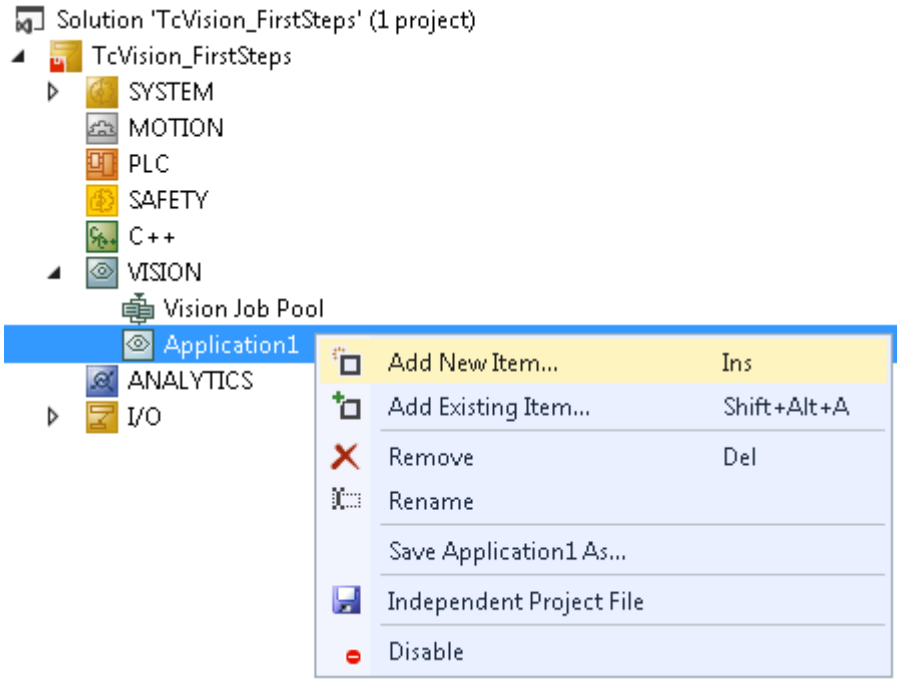
以下步骤中描述了创建文件源 [▶ 123]对象。这可以用来将存储的图像加载到 PLC 中。为此，目标系统必须处于配置模式。点击 TwinCAT XAE Base 工具栏中的符号  或点击**TWINCAT 下 Visual Studio 菜单**中的**重新启动 TwinCAT（配置模式）**。

或者，可以创建 GigE Vision 相机 [▶ 38]。

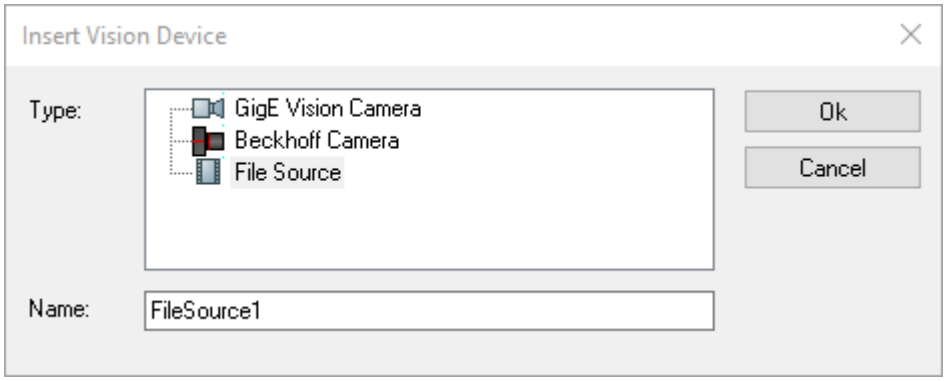
按以下步骤创建文件源对象：

1. 点击之前创建的视觉节点下的应用程序。

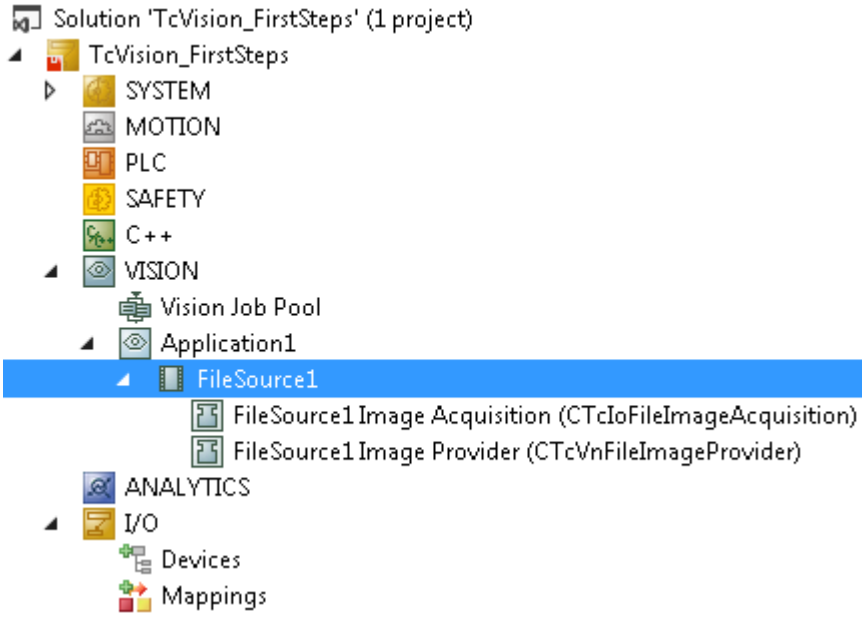
2. 右键单击并选择**添加新项目...**



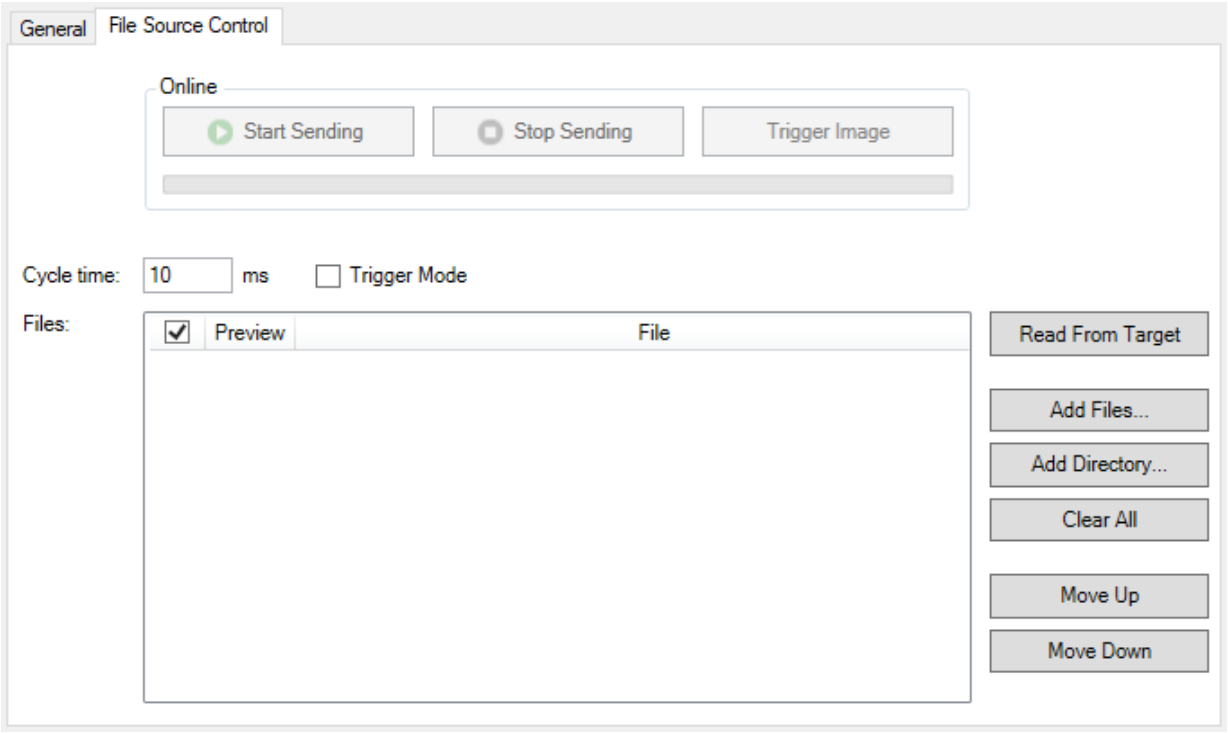
3. 选择设备类型**文件源**，必要时调整名称，点击**确定**。



⇨ 现在，文件源控制已经在应用程序下面创建，且可以使用。

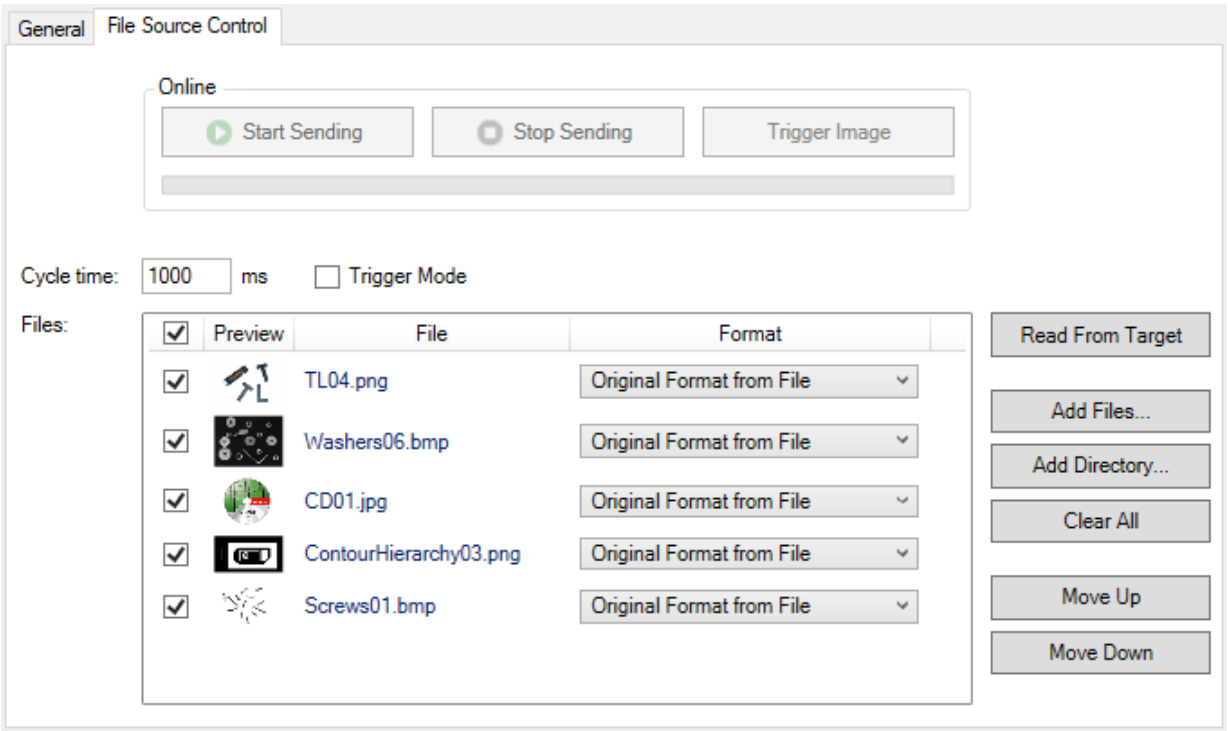


4. 双击项目树中应用程序节点下的**文件源对象**。在打开的窗口中，选择**文件源控制**选项卡，并实施相关设置。



5. 单击**从目标读取**。项目中的文件源控制 [▶ 124]与所选目标系统上存储的任何图像进行比较。
6. 通过**添加文件...**，通过**添加目录**...或通过拖放将图像添加到文件源控制。如有必要，可通过项目下载下载样本图片：第一步 [▶ 27]
7. 如有必要，调整文件源对象向 PLC 发送图像的周期时间。

⇨ 文件源对象可以像这样使用：



下一步

创建 PLC 项目 [▶ 45]

4.2.3 创建 GigE Vision 相机

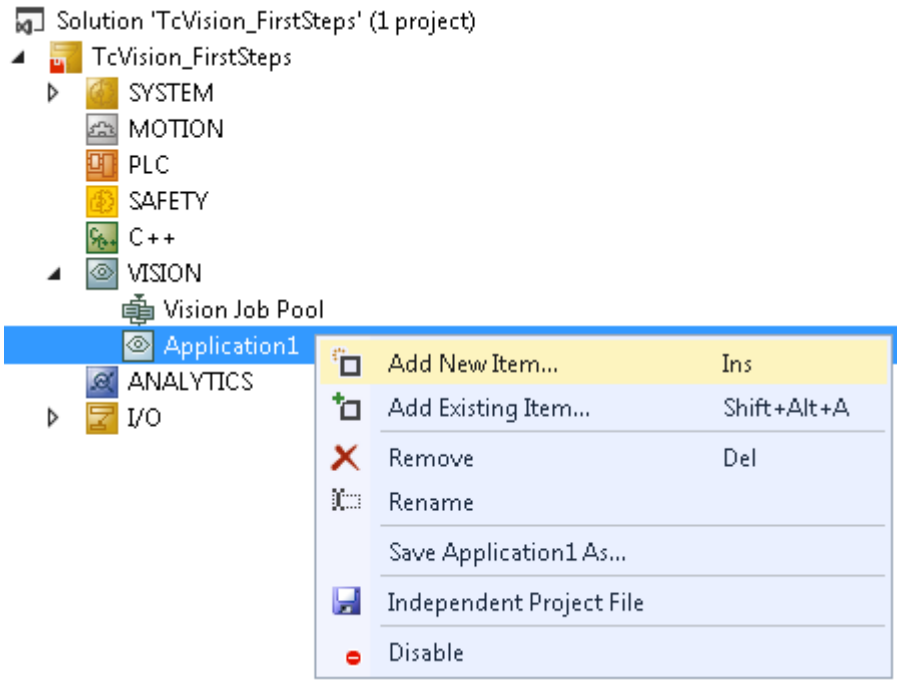
以下步骤介绍的是如何创建 GigE Vision 相机对象 [▶ 71]，该对象也可用于创建倍福相机 [▶ 122]对象。这需要将 GigE Vision 相机连接到 PC，并为相应的网络端口安装 TwinCAT RT 以太网适配器，具体如 系统配置 [▶ 31] 中所述。

另外，目标系统必须处于 Config Mode（配置模式）。点击 TwinCAT XAE Base 工具栏中的  符号，或点击 **Visual Studio menu > TWINCAT（Visual Studio 菜单 > TWINCAT）**中的 **Restart TwinCAT (Config Mode)（重新启动 TwinCAT（配置模式））**。

或者，您也可以创建 File Source Control（文件源控制器） [▶ 35]。

相机创建步骤如下：

1. 点击先前创建 Vision 节点下的一个应用。
2. 右键点击并选择 **Add new item...（添加新项目…）**



3. 选择设备类型 **GigE Vision Camera 或倍福相机**。如有需要，可调整相机名称和 GigEVision Camera Options（GigEVision 相机选项），然后点击 **OK**。

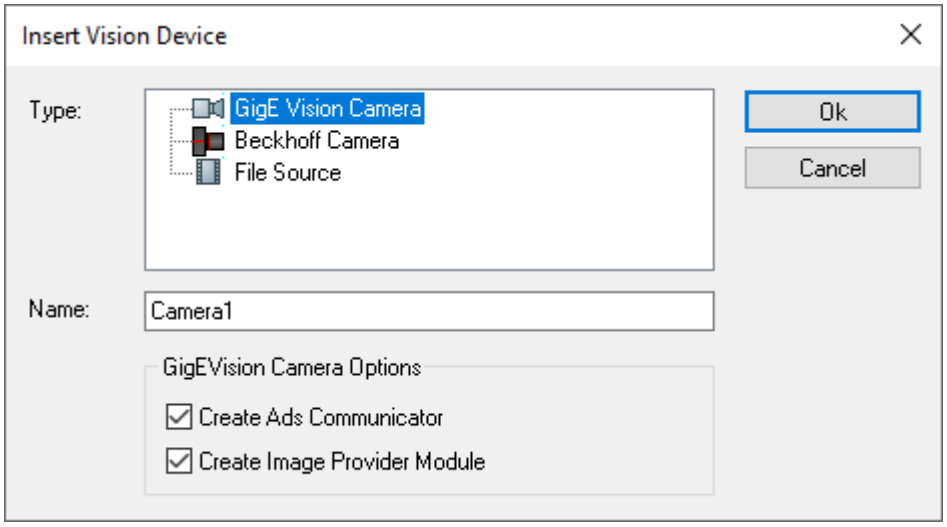
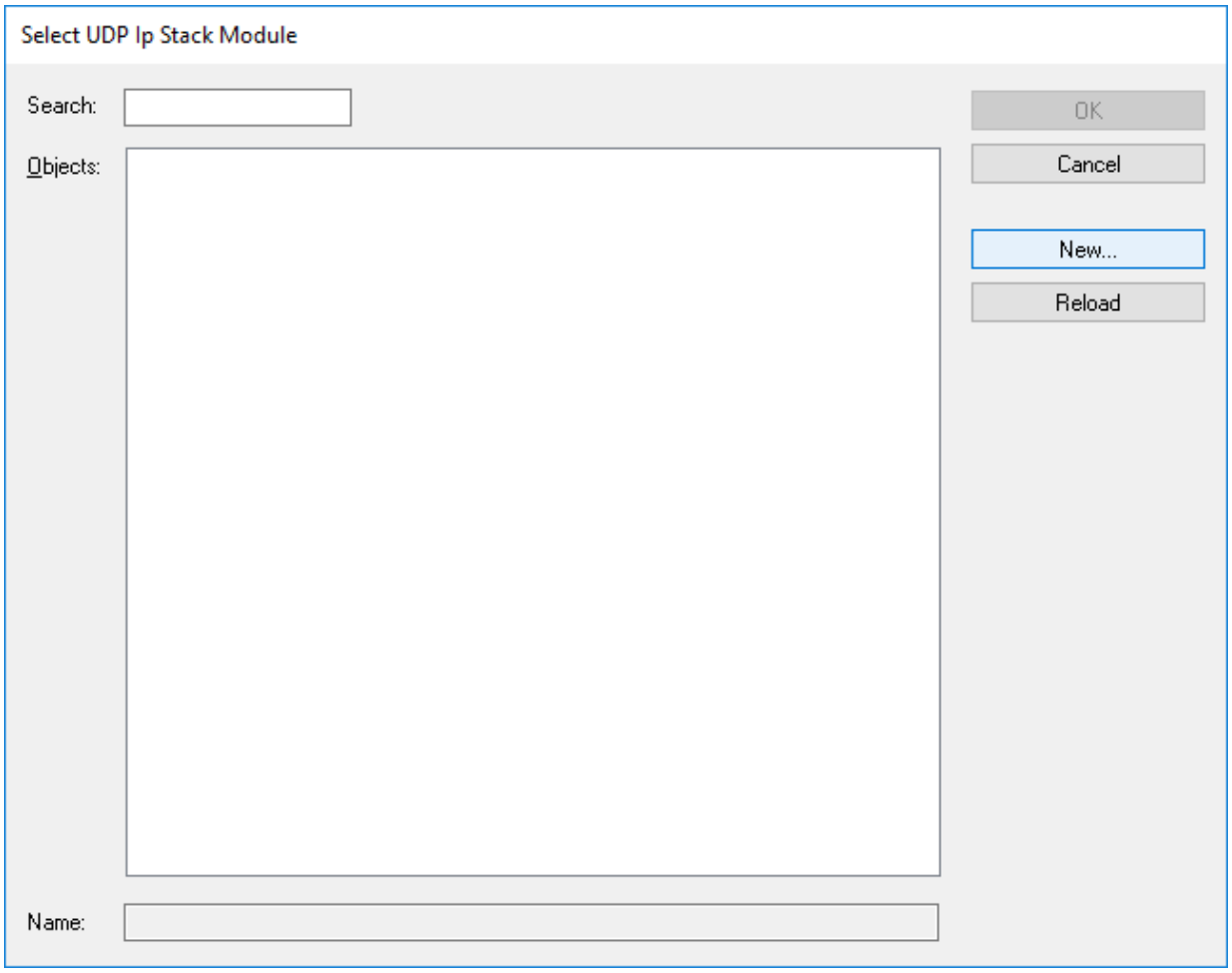


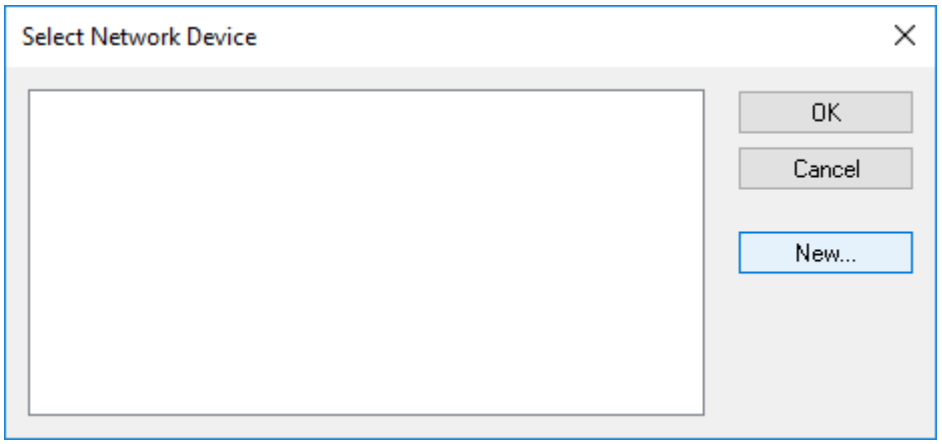
Image Provider 是 PLC 功能块的接口，因此通常始终是必要的（请参见 Camera details（相机详情） [▶ 120]）。

只有通过相机对象中的 Record/Playback（记录/回放） [▶ 101]选项卡记录数据流时才需要 Ads Communicator。
如果不记录数据流，则可禁用此选项。这样就不需要 Ads Communicator 模块和相应的 VnTask ADS。

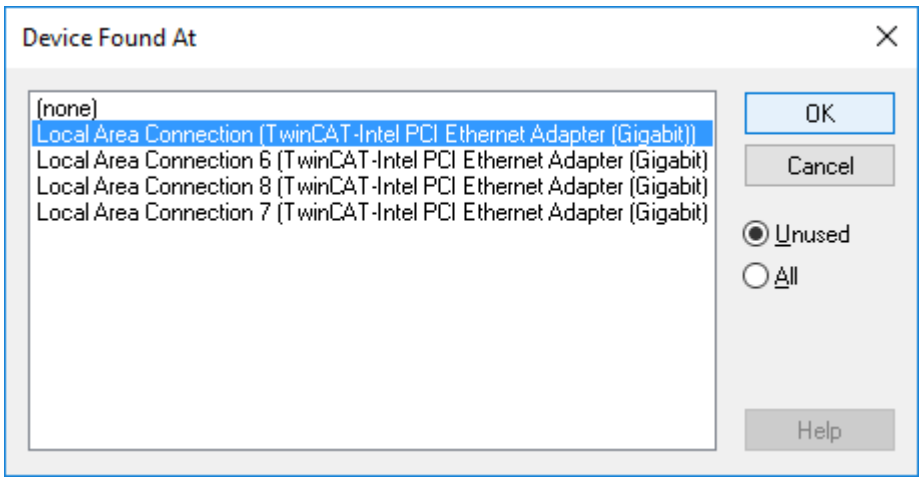
4. GigE Vision 相机通过 UDP/IP 进行通信。对于所使用的网络端口，需要在 I/O 节点中安装一个带有相应 IP 栈的合适网络设备。点击 **New...（新建…）** 进行创建。
只能针对网卡的每个物理端口创建一个带有一个 IP 栈的网络设备。



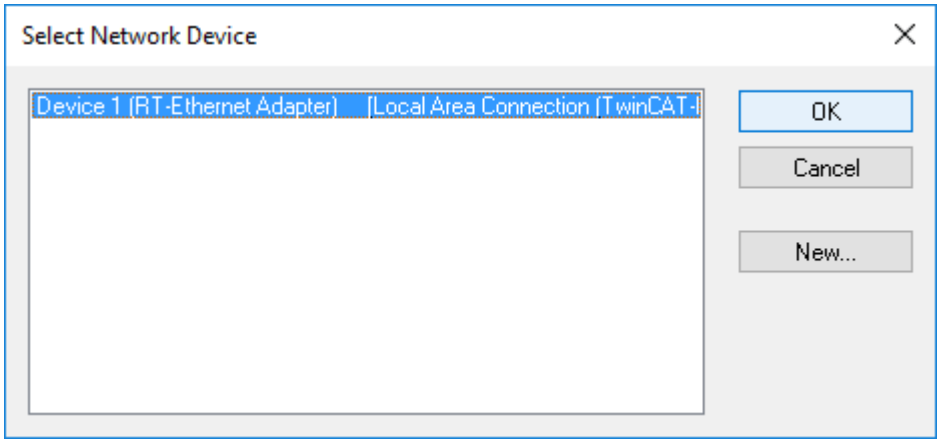
5. 要创建新的网络设备，请点击 **Select Network Device（选择网络设备）** 对话框中的 **NEW...（新建…）** 。



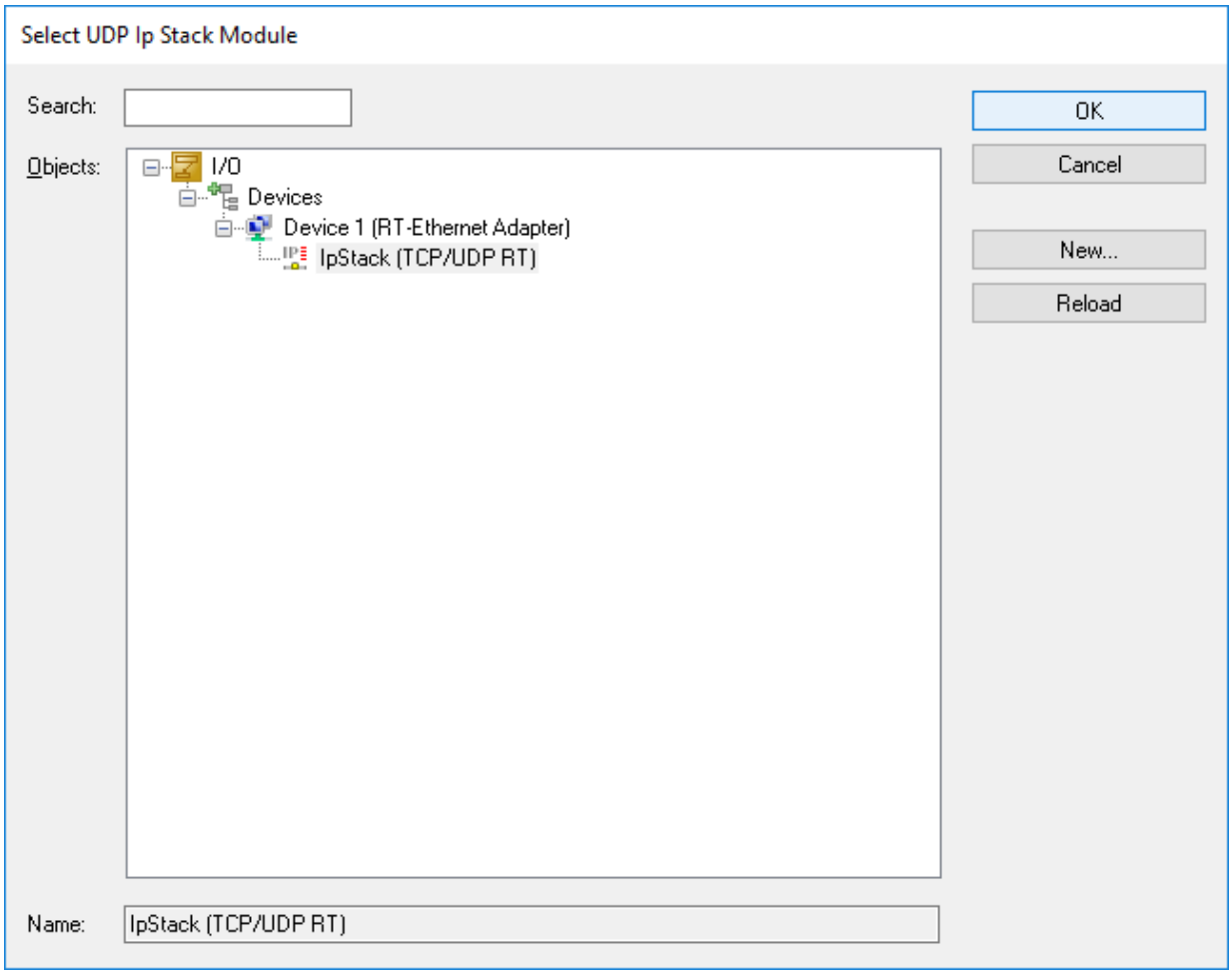
6. 在 **Device Found At（设备找到位置）**对话框中，选择相机所连接的网络端口，然后点击 **OK**。
   只显示已安装 TwinCAT RT 以太网驱动程序的网络端口。



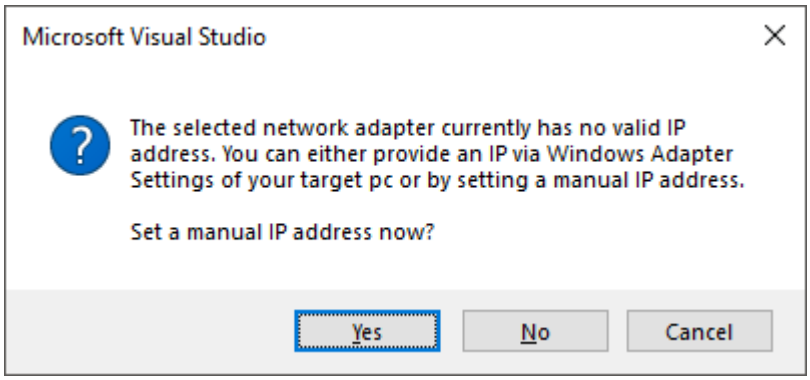
7. 现已在 I/O 配置中创建网络设备。在 **Select Network Device（选择网络设备）**对话框中选择它，然后点击 OK。这样就为设备创建了一个 IP 栈。



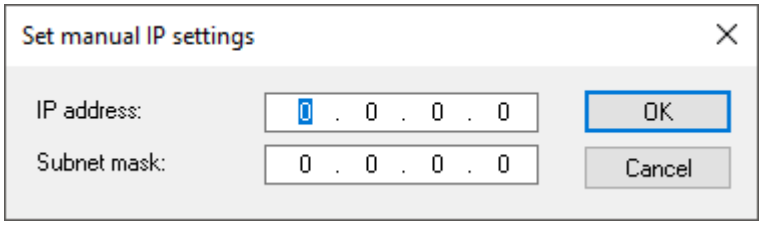
8. 选择 IP 栈，然后点击 **OK**。



9. 如果尚未为网络端口分配 IPv4 地址，请回答是否要分配固定 IP 地址。

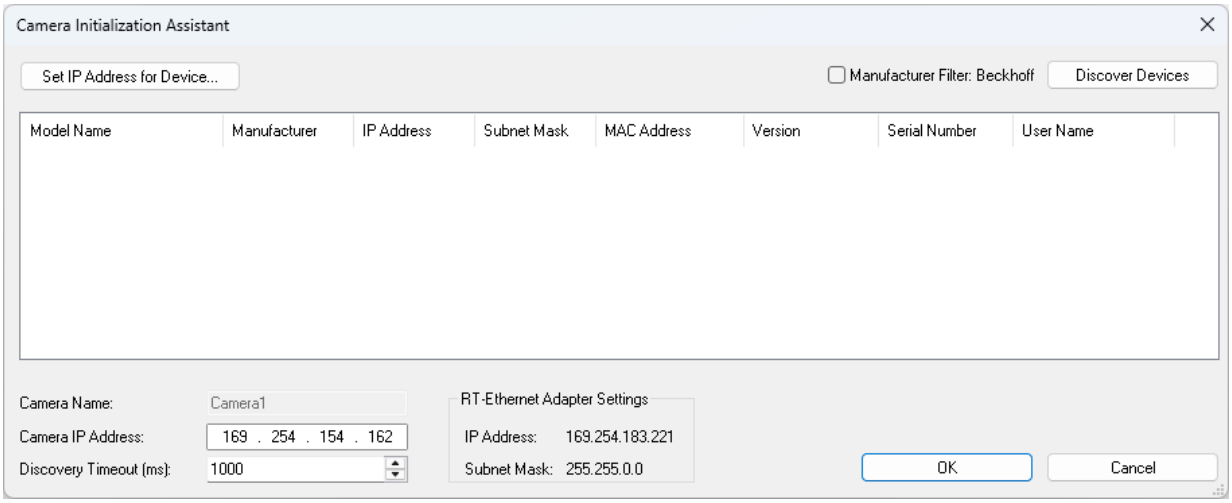


如果是，请在以下对话框中设置 IP 地址。

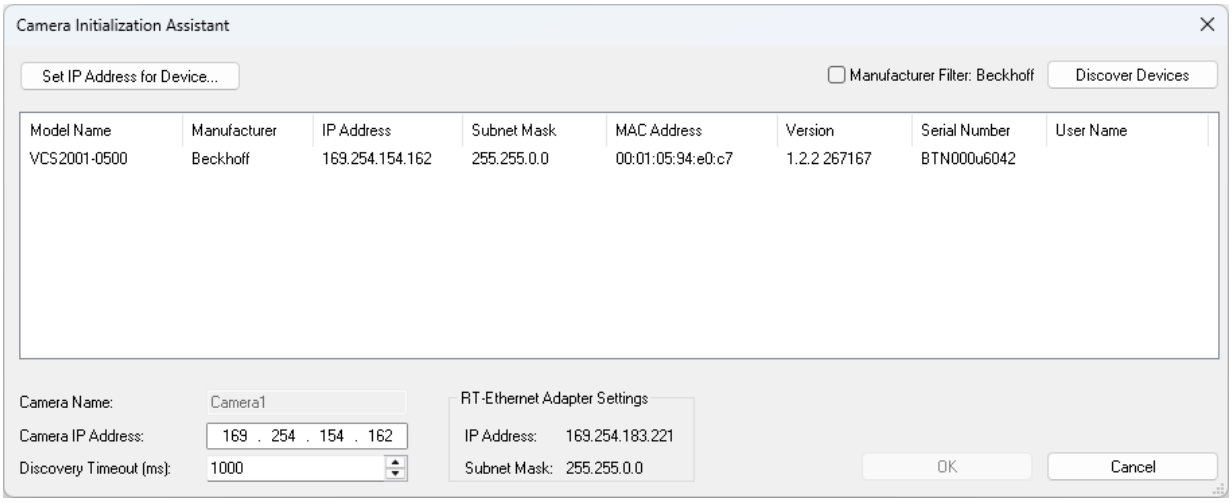


然后 IP 地址和子网掩码将保存在网卡的 TwinCAT IP 配置中（请参见 Network configuration（网络配置） [▶ 38]）。

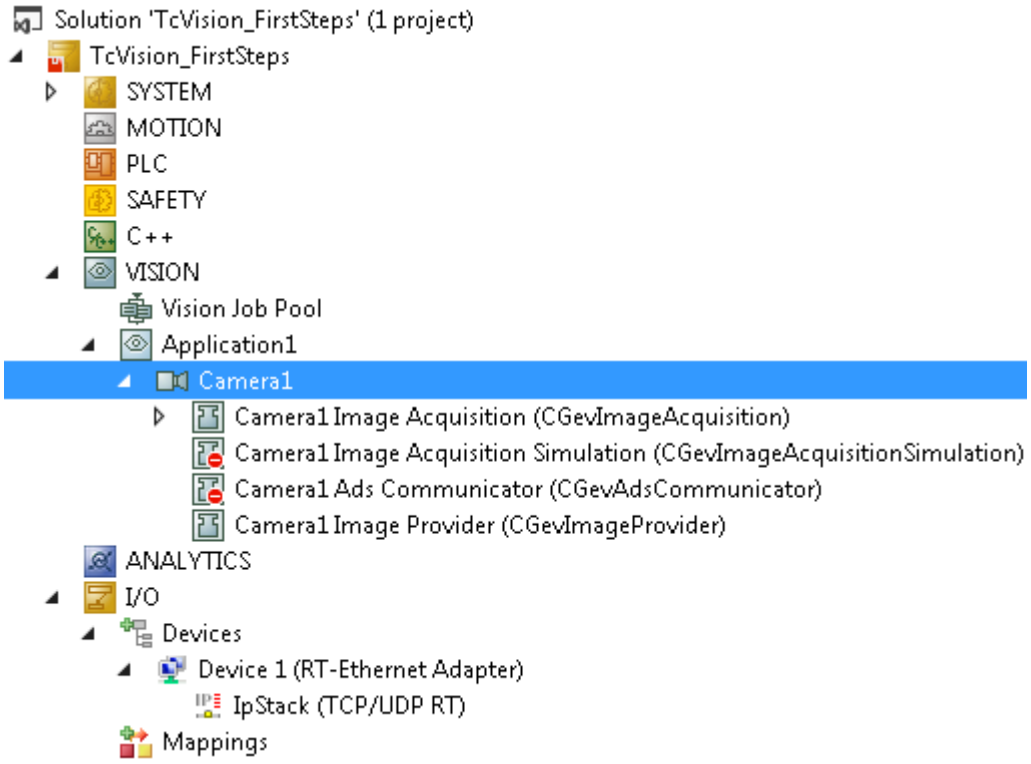
10. 在 **Camera Initialization Assistant（相机初始化助手）**中，点击 **Discover Devices（发现设备）**。所选网络端口将搜索连接相机。



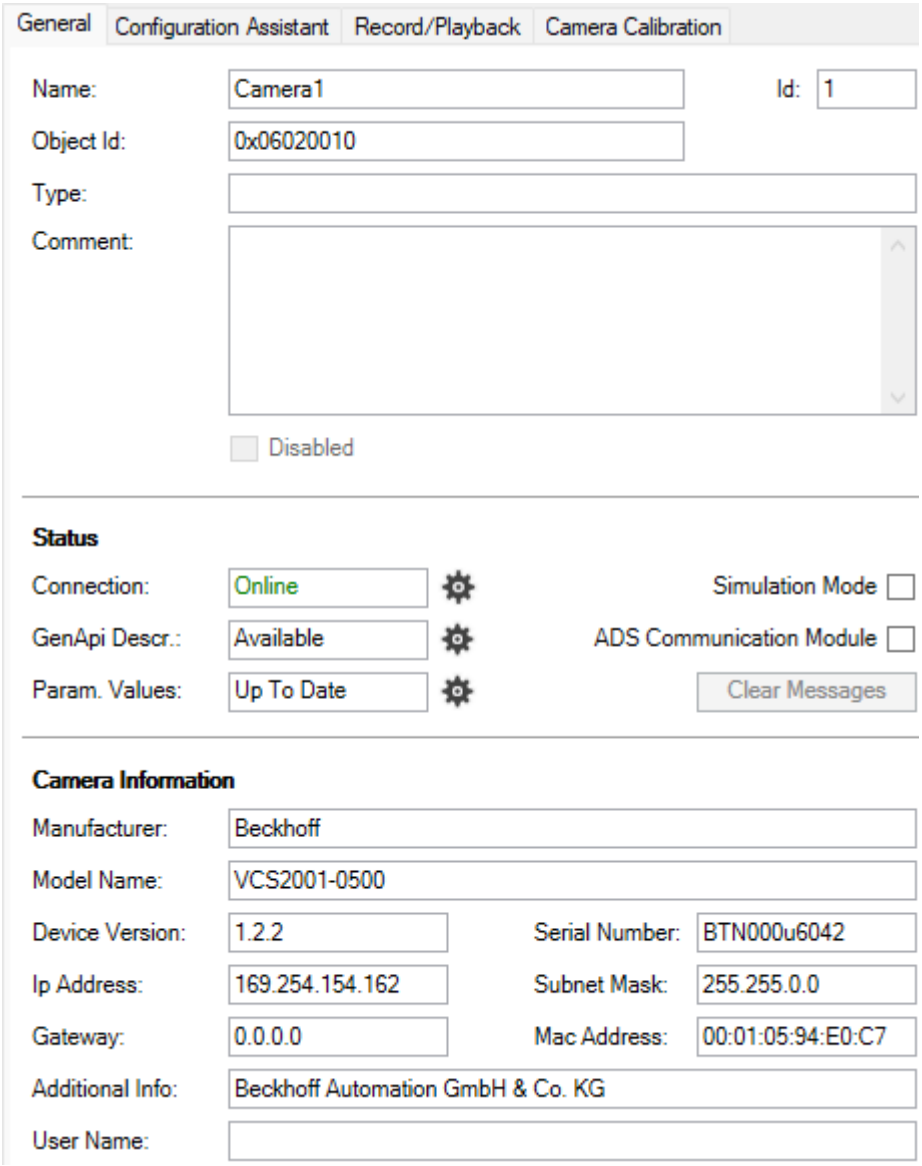
11. 根据 Manufacturer Filter（制造商筛选）设置的不同，会显示找到的所有相机。选择所需的相机并点击 **OK**。



⇨ 相机现已在应用下创建并可使用。



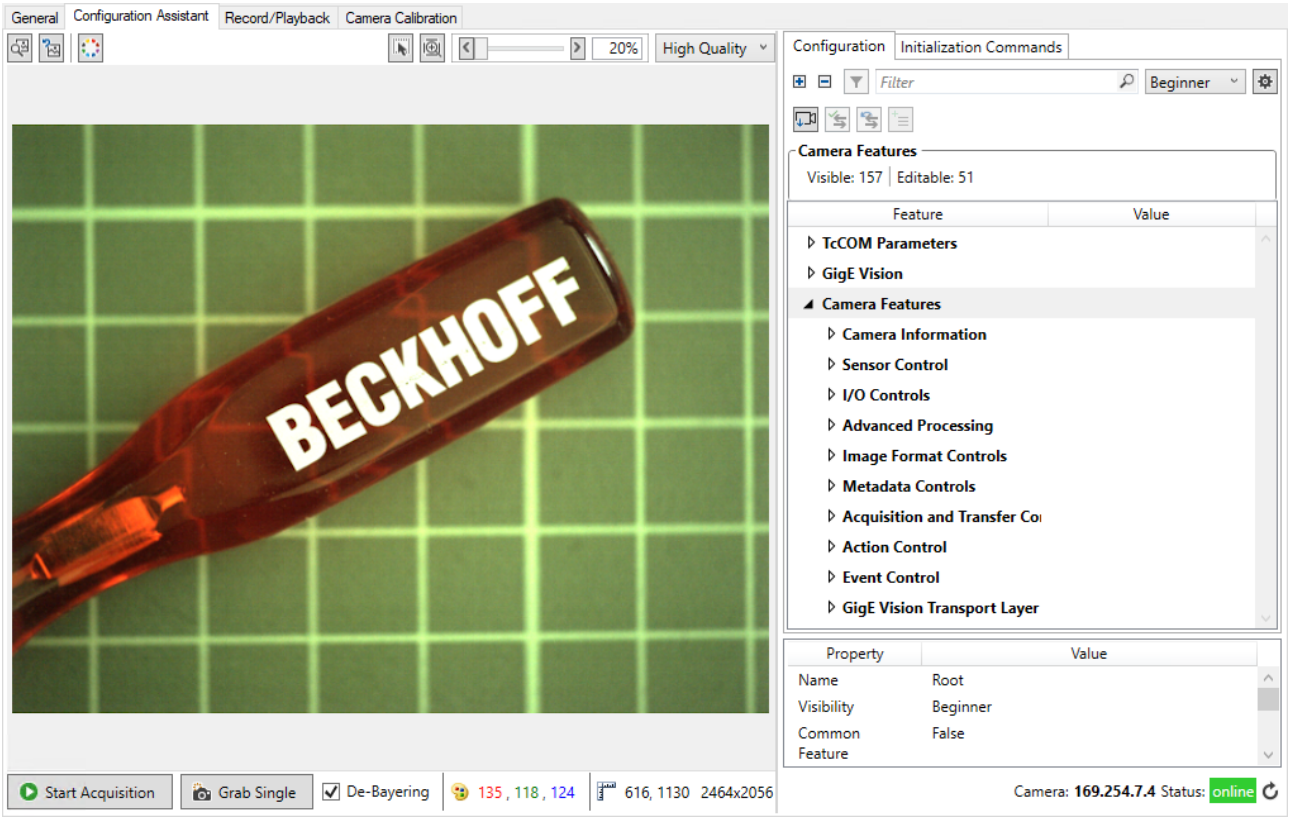
双击相机节点 [▶ 71]可打开一个窗口，在其中可验证相机连接。常规 [▶ 73] 选项卡可提供有关相机的一般信息。



在 配置助手 [▶ 77] 中，您可以使用 **Start Acquisition（开始采集）**查看相机的实时图像，并对相机进行配置设置。

另请参见：相机配置样本 [▶ 2828]

如果要应用更改的参数并在每次启动时将其写入相机，请在最后创建 初始化命令 [▶ 89]。



下一步

创建 PLC 项目 [▶ 45]

4.3 创建 PLC 项目

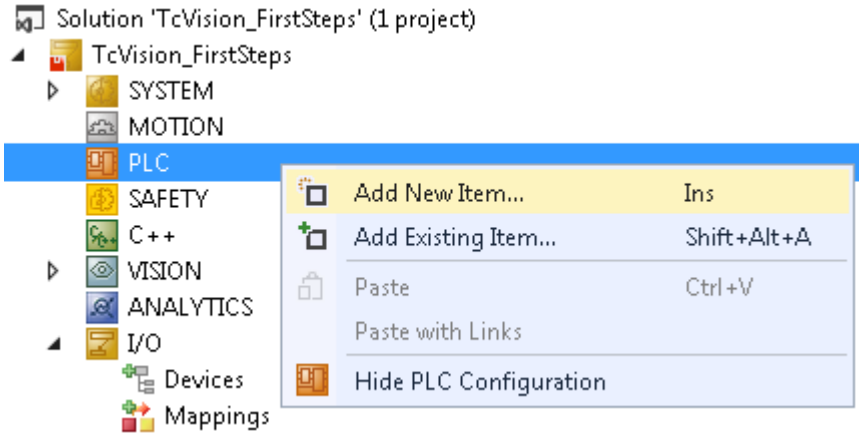
通过 TwinCAT Vision 创建 PLC 项目的视频示例：

视频：https://infosys.beckhoff.com/content/1033/tf7xxx_tc3_vision/Resources/5977084043.mp4

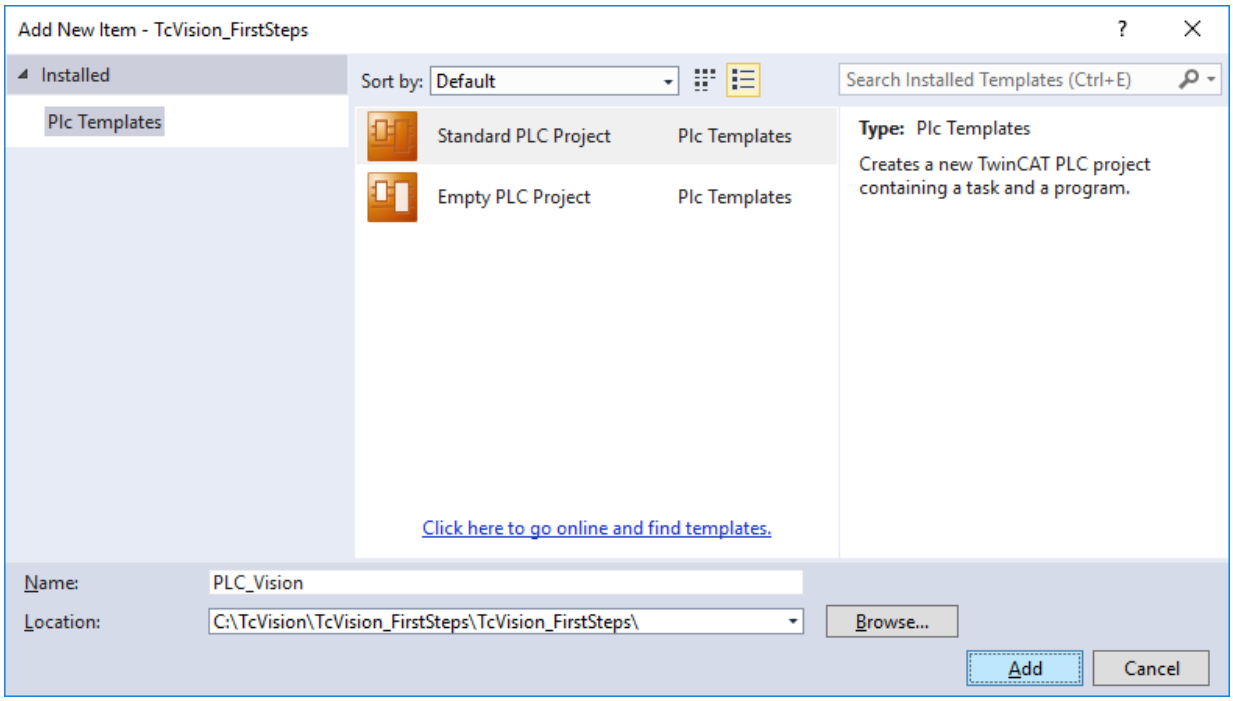
4.3.1 创建 PLC 项目

与一般机器控制的编程一样，图像处理的编程也在 PLC 中进行。为此，TwinCAT 项目必须通过 PLC 项目扩展。

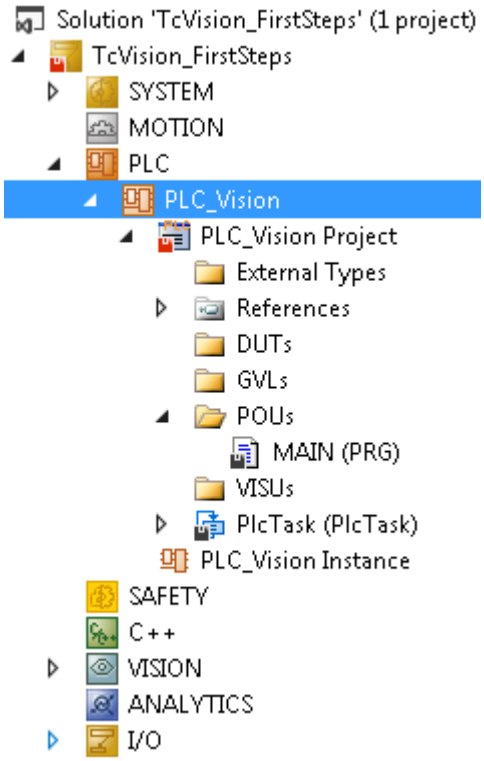
1. 右键单击解决方案资源管理器中的 PLC 节点，然后选择**添加新项目...**



2. 在**添加新项目**对话框中选择名称和位置。



⇨ PLC 项目已创建。



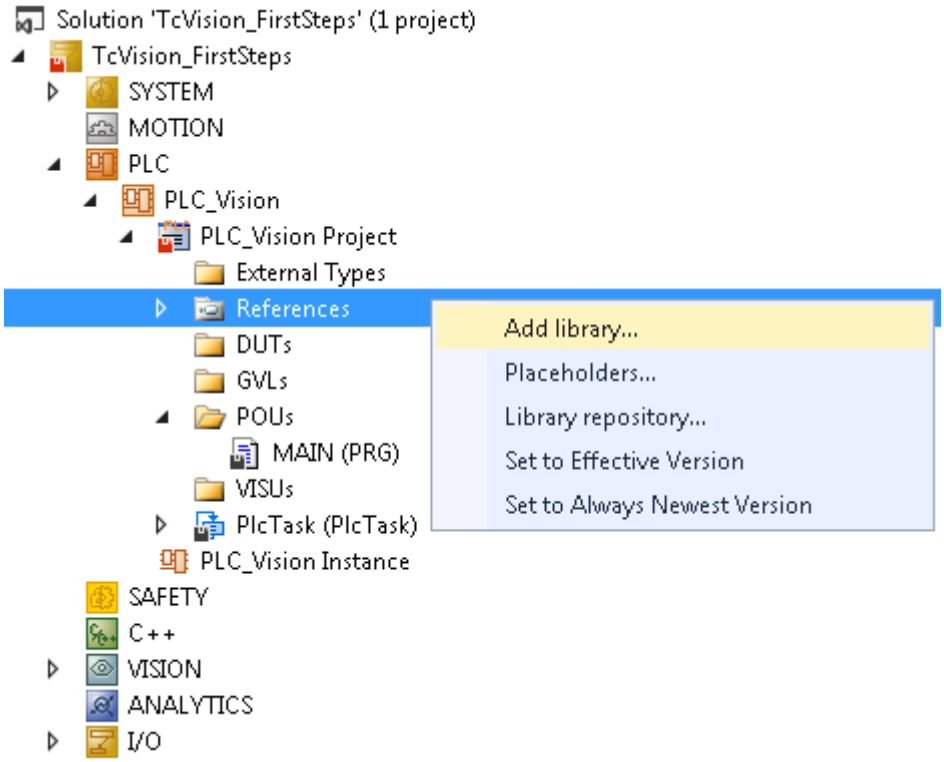
下一步

集成 PLC 库 [▶ 46]

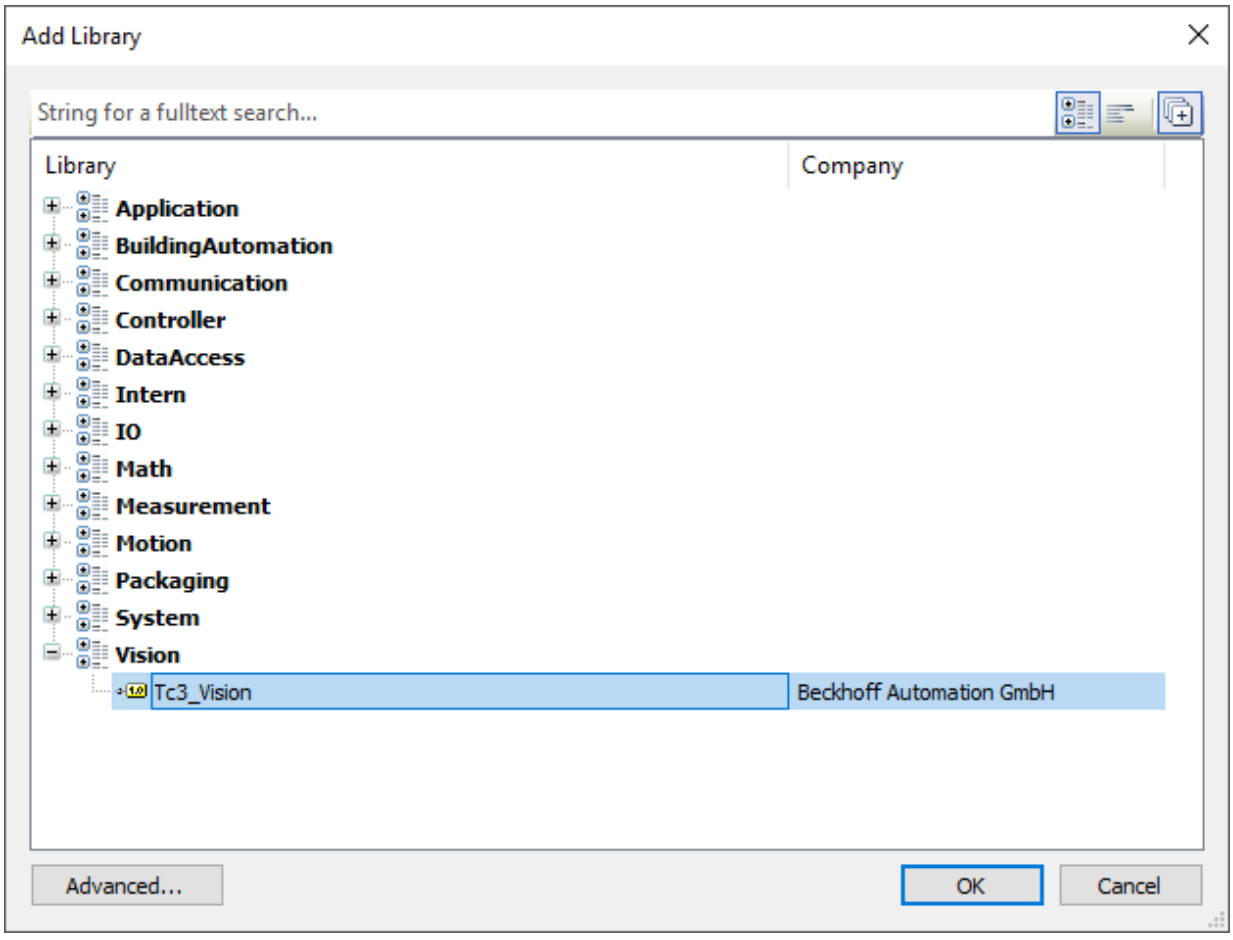
4.3.2 集成 PLC 库

Tc3_Vision 库包含图像处理所需的数据类型、功能和功能块。

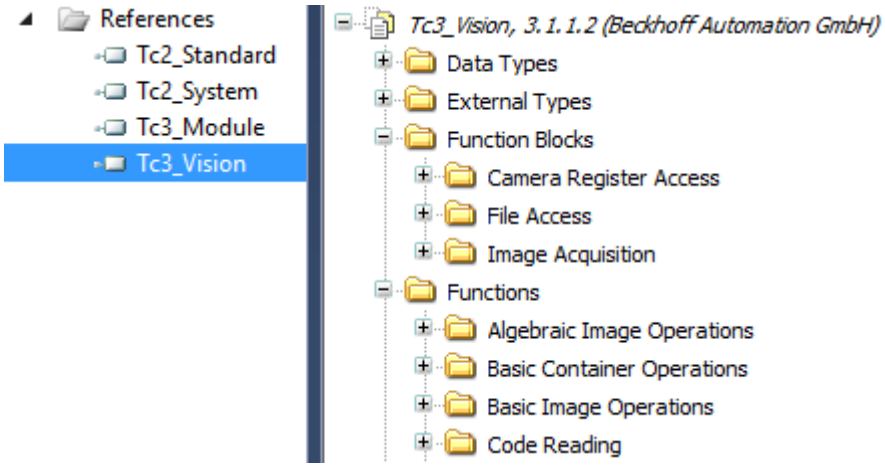
1. 如需将其添加到 PLC 项目中，右键单击**参考**，然后选择**添加库...**



2. 选择**Tc3_Vision**库，并通过**确定**进行确认。



⇨ 现在库已添加，且可以双击查看。



下一步

编写 PLC 程序 [▶ 48]

4.3.3 编写 PLC 程序

简单 PLC 程序用于从相机或文件源连续采集图像，包括以下内容：

变量

```
PROGRAM MAIN
VAR
    hr                  :    HRESULT;
    fbCamera            :    FB_VN_SimpleCameraControl;
    eState              :    ETcVnCameraState;
    ipImageIn           :    ITcVnImage;
    ipImageInDisp       :    ITcVnDisplayableImage;
    nNewImageCounter    :    UINT;
END_VAR
```

名称	类型	描述
hr	HRESULT	Tc3_Vision库的所有功能和许多方法都返回一个 HRESULT 类型的结果代码，用于错误处理。关于背后概念和单个代码含义的更多信息，可参见 HRESULT [▶ 135]。
fbCamera	FB_VN_SimpleCameraControl	每个相机和文件源控制都由一个功能块实例表示。在最简单的情况下，这是一个FB_VN_SimpleCameraControl [▶ 1630]的实例，其中涵盖了相机和文件源控制的共同属性，且因此可以与两种设备类型相联系。另外，FB_VN_GevCameraControl [▶ 1609]可用于相机，而FB_VN_FileSourceControl [▶ 1601]用于文件源控制。
eState	ETcVnCameraState	每个视觉设备都有一个类型为ETcVnCameraState [▶ 162]的状态。
ipImageIn	ITcVnImage	处理的图像使用ITcVnImage [▶ 407]变量进行管理。在这个例子中，相机输入图像的访问通过ipImageIn获取。
ipImageInDisp	ITcVnDisplayableImage	显示的图像通过ITcVnDisplayableImages [▶ 407]管理。

带有状态机的程序代码

每个视觉设备都有一个内部状态，TCVN_CS_INITIAL是初始状态。由此，可以使用StartAcquisition方法将设备带入TCVN_CS_ACQUIRING状态。在这种状态下，图像可以被捕捉和传输。如果一个图像完全存储在存储器中，可以通过fbCamera.GetCurrentImage(ipImageIn)检索到。

为了检查一个图像是否被完全接收并可以被检索，IF 查询查询该方法的HRESULT以及图像内存区域的指针是否不为 0。之后，可以开始分析图像。例如，为了检查并表明可以检索到图像，计数器变量nNewImageCounter将递增。

F_VN_TransformIntoDisplayableImage [▶ 818]然后将图像指针移到ipImageInDisp，以显示图像。这样，就不能再通过ipImageIn检索图像了。另外，可以通过F_VN_CopyIntoDisplayableImage [▶ 781]创建副本，以显示和编辑图像。

```
eState := fbCamera.GetState();

IF eState = TCVN_CS_ERROR THEN
    hr := fbCamera.Reset();

ELSIF eState < TCVN_CS_ACQUIRING THEN
    hr := fbCamera.StartAcquisition();

ELSIF eState = TCVN_CS_ACQUIRING THEN
    hr := fbCamera.GetCurrentImage(ipImageIn);

    // Check if new Image was received
    IF SUCCEEDED(hr) AND ipImageIn <> 0 THEN
        nNewImageCounter := nNewImageCounter + 1;

        // Place to call vision algorithms
        hr := F_VN_TransformIntoDisplayableImage (ipImageIn, ipImageInDisp, hr);
    END_IF
END_IF
```

关于状态机的更多信息，可查看 API 章节图像采集 [▶ 1599]。

更多信息

- API 参考 [▶ 132]
- 图像采集的功能块 [▶ 1599]
- 图像 [▶ 143]

下一步

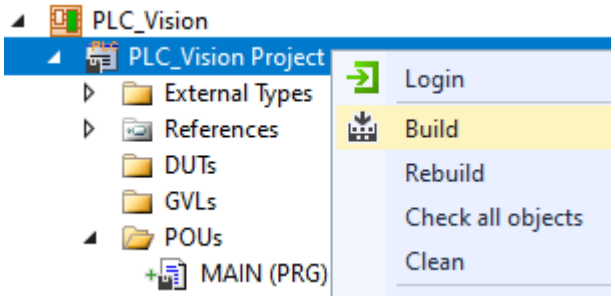
初始化功能块 [▶ 49]

4.3.4 初始化功能块

在配置了视觉设备和创建 PLC 程序之后，现在必须将视觉设备与 PLC 中适当的相机变量联系起来。

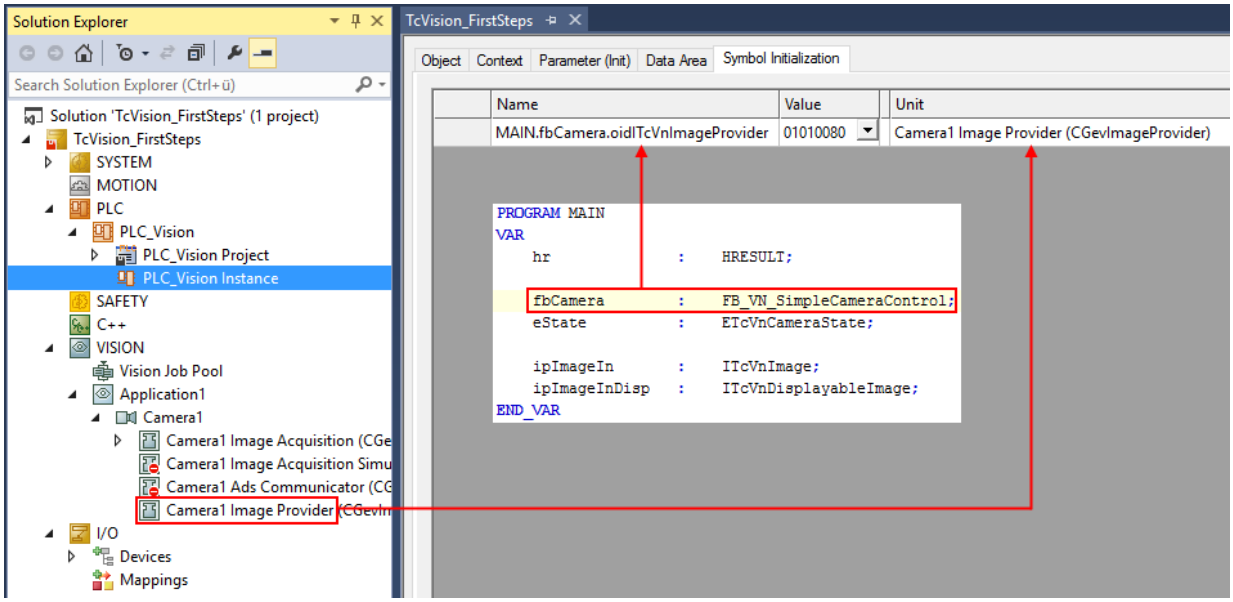
PLC 程序的功能块FB_VN_SimpleCameraControl [▶ 1630]的实例fbCamera必须与视觉设备的图像提供者连接。请注意：

1. 通过右键单击 PLC 项目并选择**构建**，建立PLC 程序 [▶ 48]。确保在操作过程中不会发生错误。如果发生错误，**符号初始化**选项卡出现在 PLC 实例和其下方fbCamera实例中。



2. 双击**PLC 项目实例**，然后打开**符号初始化**选项卡。

3. 通过“数值”栏的下拉菜单，将你的视觉设备的图像提供者分配给符号 `MAIN.fbCamera.oidITcVnImageProvider`。这是为图像中的相机对象显示的，但它与文件源对象的工作方式完全相同。



下一步

激活 TwinCAT 项目并启动 PLC 项目 [▶ 50]

4.4 运转

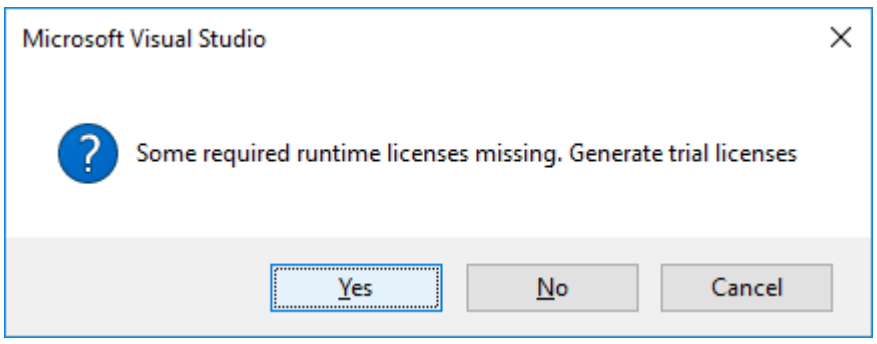
执行 TwinCAT Vision PLC 项目的视频示例：

视频：https://infosys.beckhoff.com/content/1033/tf7xxx_tc3_vision/Resources/5977086347.mp4

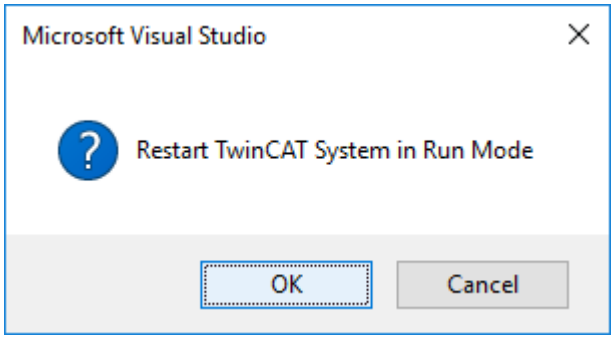
4.4.1 激活 TwinCAT 项目并启动 PLC 项目

可通过执行以下步骤执行项目：

1. 通过 或通过 Visual Studio 菜单**TwinCAT** >**激活配置**激活配置。
2. 如果目标系统上尚未激活许可证，会出现以下对话框。对于最初测试，可通过**是**，并输入安全代码，创建一个试用许可证。



3. 然后会问您是否要重新启动 TwinCAT。通过**是**确认。另外，可以通过 或通过 Visual Studio 菜单 **TwinCAT** >**重新启动 TwinCAT 系统**触发重启。



4. 如果 PLC 未设置为自动启动，请手动启动。为此，通过 或通过 Visual Studio 菜单的**PLC** >**登录**登录 PLC。

5. 然后通过 或通过 Visual Studio 菜单的**PLC** >**启动**启动 PLC。

下一步

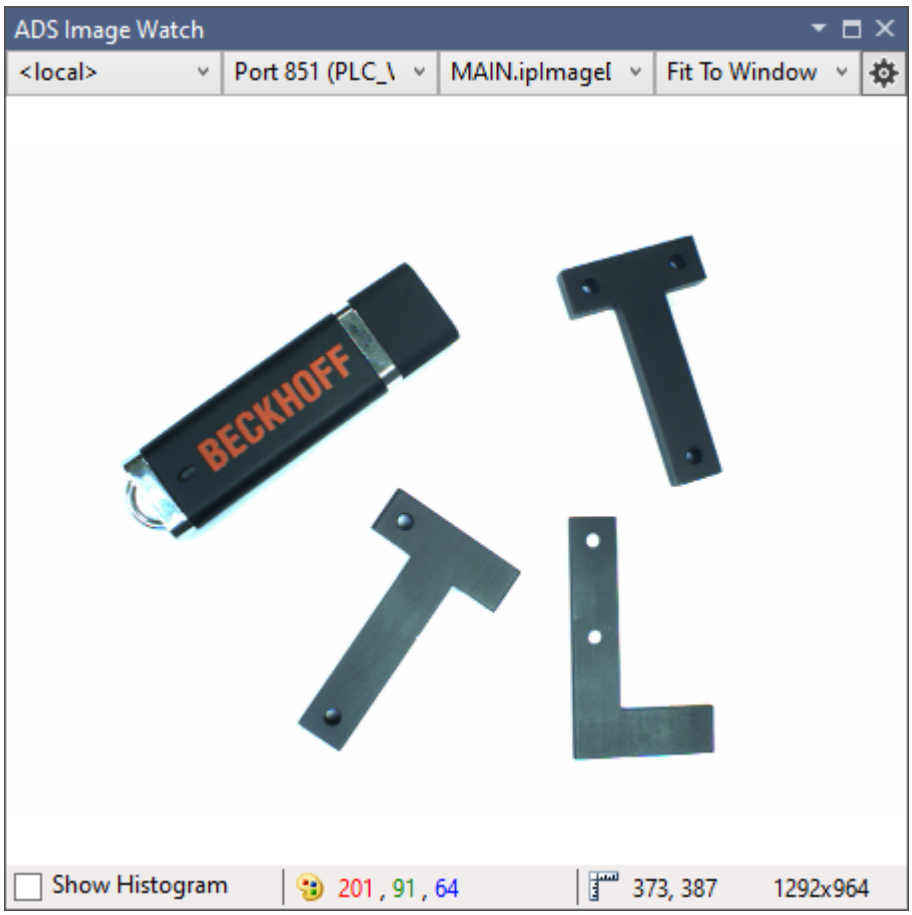
ADS 图像查看中的图像显示 [▶ 51]

4.4.2 ADS 图像查看中的图像显示

PLC 中类型为`ITcVnDisplayableImage`的图像可以显示在ADS 图像查看中。您可以在 Visual Studio 菜单 **TwinCAT > Windows > ADS 图像查看**中找到该窗口。

如需选择待显示图像，可从左到右设置以下参数：

- 目标系统
- 端口
- 图像的 ADS 符号
- 图像缩放

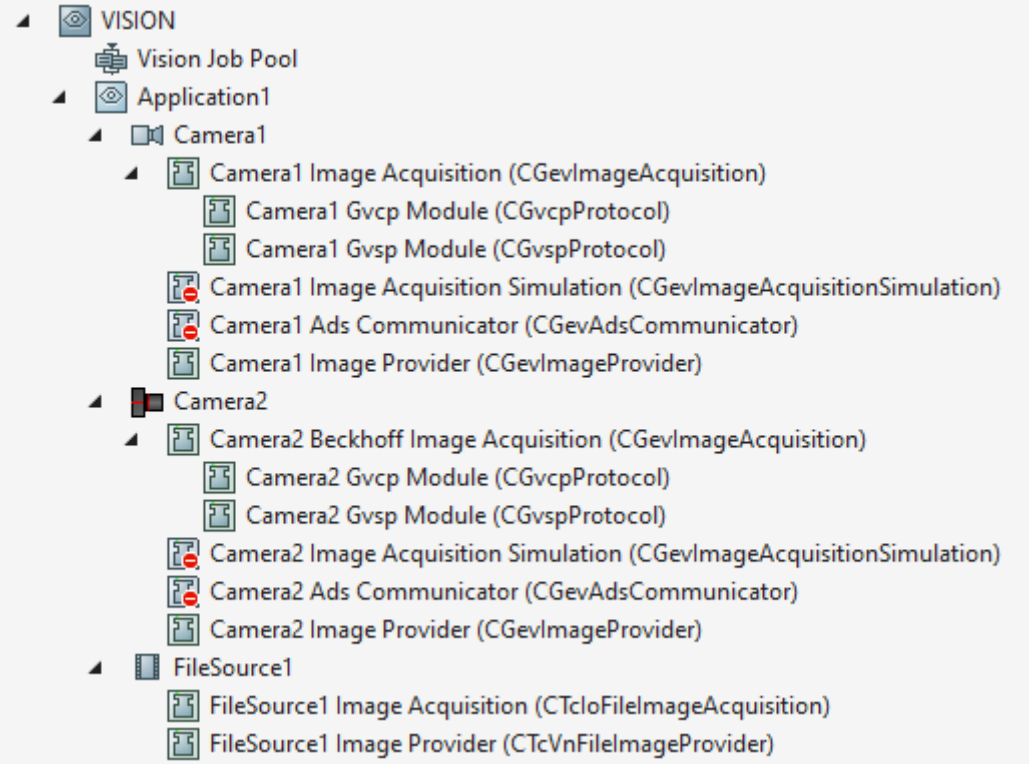


除了图像内容外，还可以在 ADS 图像查看中读取以下信息：

- 图像的直方图
- 当前光标位置的颜色/强度值
- 当前光标位置的像素坐标
- 图像大小

5 开发环境

TwinCAT Vision 完全集成在 TwinCAT 3 开发环境中。从本质上讲，TwinCAT Vision 通过另一个节点扩展 TwinCAT 系统，在该节点下可创建和配置 Vision 组件。使用了以下配置元素：



- VISION 节点 [▶ 68]可提供影响系统上所有 Vision 组件的常规配置选项。其中包括所有其他配置元素。
- Vision Job Pool（Vision 工作吃） [▶ 127]可用于定义并行执行 Vision 函数的工作任务。每个系统都有一个单独的 Vision Job Pool（Vision 工作池）。
- 一个应用节点 [▶ 70]为多个 Vision 设备（GigE Vision 相机和文件源）提供接口，形成一个应用单元。可以导出和导入应用的整个配置。每个 TwinCAT 项目可以有多个应用。
- GigE Vision 相机节点 [▶ 71]包含一个物理 GigE Vision 相机的配置。每个应用可以有多个 GigE Vision 相机节点。
- 倍福相机节点 [▶ 122]包含一个物理倍福相机的配置。每个应用可以有多个倍福相机节点。
- 文件源节点 [▶ 123]包含目标系统硬盘上的图像与 PLC 内图像采集模块之间接口的配置。每个应用可以有多个文件源节点。

与系统组件进行通信

GigE Vision 相机节点、倍福相机节点和文件源节点均包含 TcCOM 模块，可用于与各种系统组件进行通信。

此外，开发环境还具有一个 ADS Image Watch（ADS 图像查看） [▶ 128]窗口元素，可通过 ADS 显示来自 PLC 的图像。此外，为实现 Vision 组件的正确运行，需要常规系统配置 [▶ 53]。

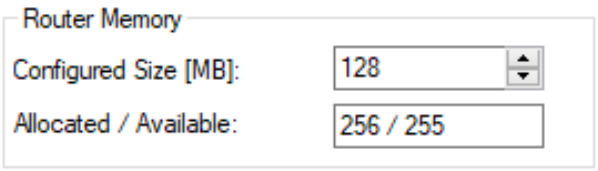
为了能够在 TwinCAT HMI 中实时显示来自 PLC 的图像，可使用相应的 HMI 扩展 [▶ 2660]。

5.1 系统配置

以下章节描述了 TwinCAT Vision 特有的设置，应根据系统和项目进行或检查。

5.1.1 路由器内存

与传统控制系统相比，图像处理应用需要大量内存，因为即使是单一图像也可能包含几兆字节的数据。在 TwinCAT Vision 中，所有动态数据（请参见接口指针 [▶ 132]）均分配在路由器内存中。因此，相应地配置路由器内存大小以及 TwinCAT 3.1 4026 的 核心内存 [▶ 55] 尤为重要。可以在 **Settings（设置）** 选项卡中实时节点下的 TwinCAT 系统配置中对其进行设置：



配置大小	设置当前项目所需的路由器内存。
分配	系统上当前分配的路由器内存。
可用	系统上当前分配路由器内存中仍可自由使用的部分。（其余部分已在 TwinCAT 中使用）。 **如果重启 TwinCAT 后可用内存没有完全恢复，则表明存在内存泄漏。在这种情况下，请重新启动系统并检查程序是否存在内存泄漏。**

TwinCAT 3.1 4024 中路由器内存的最大可调大小为 1024 MB。TwinCAT 3.1 4026 及以上版本d的可用大小基本上取决于可用主内存，且限制为 65535 MB。实际所需内存大小主要取决于 PLC 程序中动态内存的使用情况（如图像 [▶ 143]和容器 [▶ 146]）。

注意

路由器内存的大小

在配置路由器内存时，必须注意路由器内存将从主内存中减去，因此也不再可供操作系统使用。因此，我们建议在主内存大小低于 4 GB 的系统中避免使用 1024 MB 的大小。对于外部项目，在激活配置前一定要检查路由器内存设置，必要时减少内存。

估计所需的内存大小

路由器内存的所需大小取决于许多因素。通过计算动态分配的内存（通过接口指针 [▶ 132]），可以得到 TwinCAT 项目视觉部分的粗略估计。

一个图像的存储空间M（单位：字节）可以通过纳入图像大小（宽度w和高度h）、通道数C和元素类型的字节数$M = w * h * C * sizeof(ElementType)$来计算。

对于一个字符串或矢量类型的容器，存储空间M（单位：字节）可以通过分配的元素数量N和元素类型的字节数$M = N * sizeof(ElementType)$来计算。

必须注意数据的复制和发布方式。如果复制了一个图像或生成了一个新的图像，两个实例的存储空间当然必须包括在计算中。此外，路由器本身和其他项目部分的内存需求也必须包括在计算中。当通过 ADS 检索图像或其他数据时，这些都会被复制，所以也必须考虑到额外的存储空间。

注意

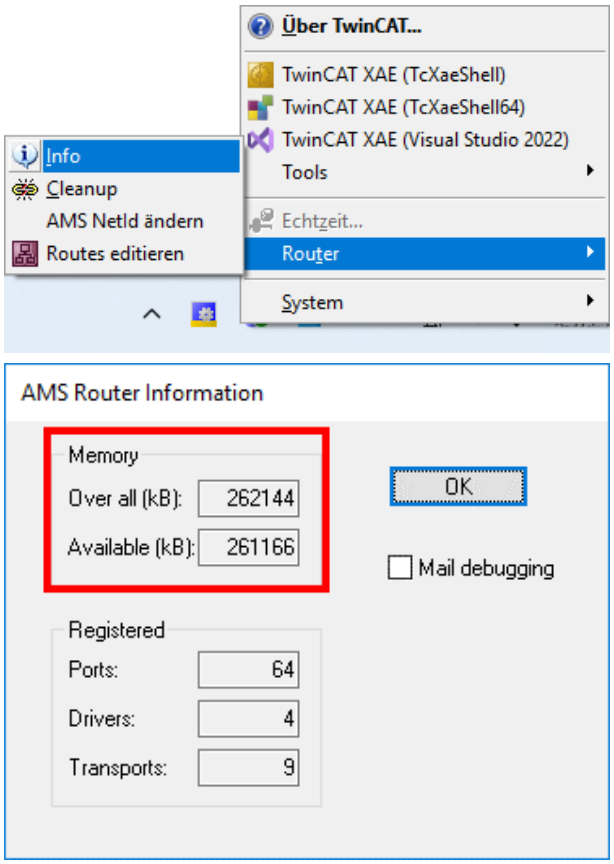
带有安全系数的内存值

始终用一个安全系数来扩展计算的内存值。还要考虑到项目中其他模块的内存需求。

为了不在路由器内存中使用不必要的大量空间，你应该注意不要做任何不必要的深度拷贝，并再次释放不再需要的对象（见接口指针 [▶ 132]）。为了了解实际使用了多少内存，可以对路由器的内存进行实时监控，具体方法如下。

监测路由器的内存

即使在项目中设置了路由器内存，这也是一个全系统的设置。因此，可以在目标系统上查看项目中设置的、目前在系统上使用的路由器内存。为此，请右击 Windows 任务栏中的 TwinCAT 图标，并选择**路由器 > 信息**。



注意

减少可用的路由器内存

可用的路由器内存可能通过异常或未释放的接口指针而减少。因此，在开发过程中应检查内存，特别是在出现错误的情况下。如果没有足够的内存，请重新启动整个系统，然后检查你的程序。

5.1.2 核心内存

从 TwinCAT 3.1 4026 版开始，除了路由器内存外，还可以为每个实时核心预留单独的内存区域。该核心的所有内存请求首先由该内存处理，只有在内存不足时，才会向路由器请求内存。有关更多设置选项和详细信息，请参见 Real-Time Settings（实时设置）项下的 TwinCAT 工程设计文档。

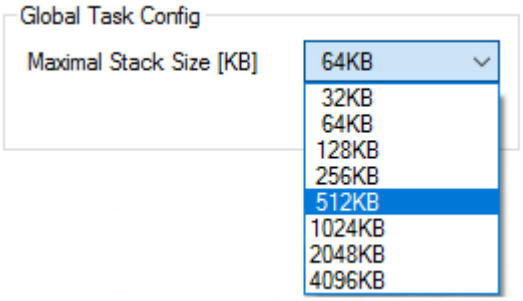
由于 TwinCAT Vision 通常使用大量动态数据（如图像），因此建议使用核心内存并相应设置其大小。这减少了对普通路由器内存的访问，从而通过并行内存访问加快了内存分配。

Core	RT-Core	Base Time	Core Limit	Latency Warning	Core Memory	Core Frequency
0 (Shared)	☐					Base Frequency
1 (Shared)	☐					Base Frequency
2 (Shared)	☐					Base Frequency
3 (Shared)	☑	1 ms	80 %	(none)	512 KB with 2 KB limit	Base Frequency
4 (Shared)	☑ Default	1 ms	80 %	(none)	512 KB with 2 KB limit	Base Frequency
5 (Shared)	☐					Base Frequency
6 (Isolated)	☑	1 ms	100 %	(none)	16 MB	Base Frequency
7 (Isolated)	☑	1 ms	100 %	(none)	64 MB	Base Frequency

5.1.3 堆栈大小

默认的 64KB 大小通常对图像处理应用来说是足够的。在使用较大的图像格式或特殊功能时，如F_VN_Clahe [▶ 1246]，该数值太小，必须增加堆栈大小。那么建议直接使用 512 KB 或更大。

它可以在 TwinCAT 系统配置中实时节点下的**设置**选项卡中设置：



5.1.4 网络适配器

在选择、配置和使用网卡时，有几个方面需要考虑。各个主题分为三个部分，并在以下章节中进行说明：

基本设置

- 首先，必须选择我们的 TwinCAT 实时以太网驱动程序所支持的用于连接相机的合适网卡，以便安装驱动程序并用于实时操作。安装驱动程序所需的步骤在安装 TwinCAT 实时以太网驱动程序 [▶ 56]一章中进行说明。
- 然后，应为此网卡停用所有不必要的网络协议，并调整 IP 设置，以确保相机通信无故障。所需的相关配置步骤（包括在 TwinCAT 项目的 IP 栈中设置一个持久 IP 地址）均在 IP 配置和网络协议 [▶ 58] 一章中进行说明。

高级设置

- 在以高数据速率进行图像传输时，可能需要使用巨型帧。相关要求、配置以及对系统行为的影响均在巨型帧 [▶ 59]一章中进行说明。
- 如果要在网卡的同一端口上操作多台相机，则必须对 IP 栈进行额外调整。相关配置在在一个端口上操作多台相机 [▶ 62]一章中进行说明。

诊断

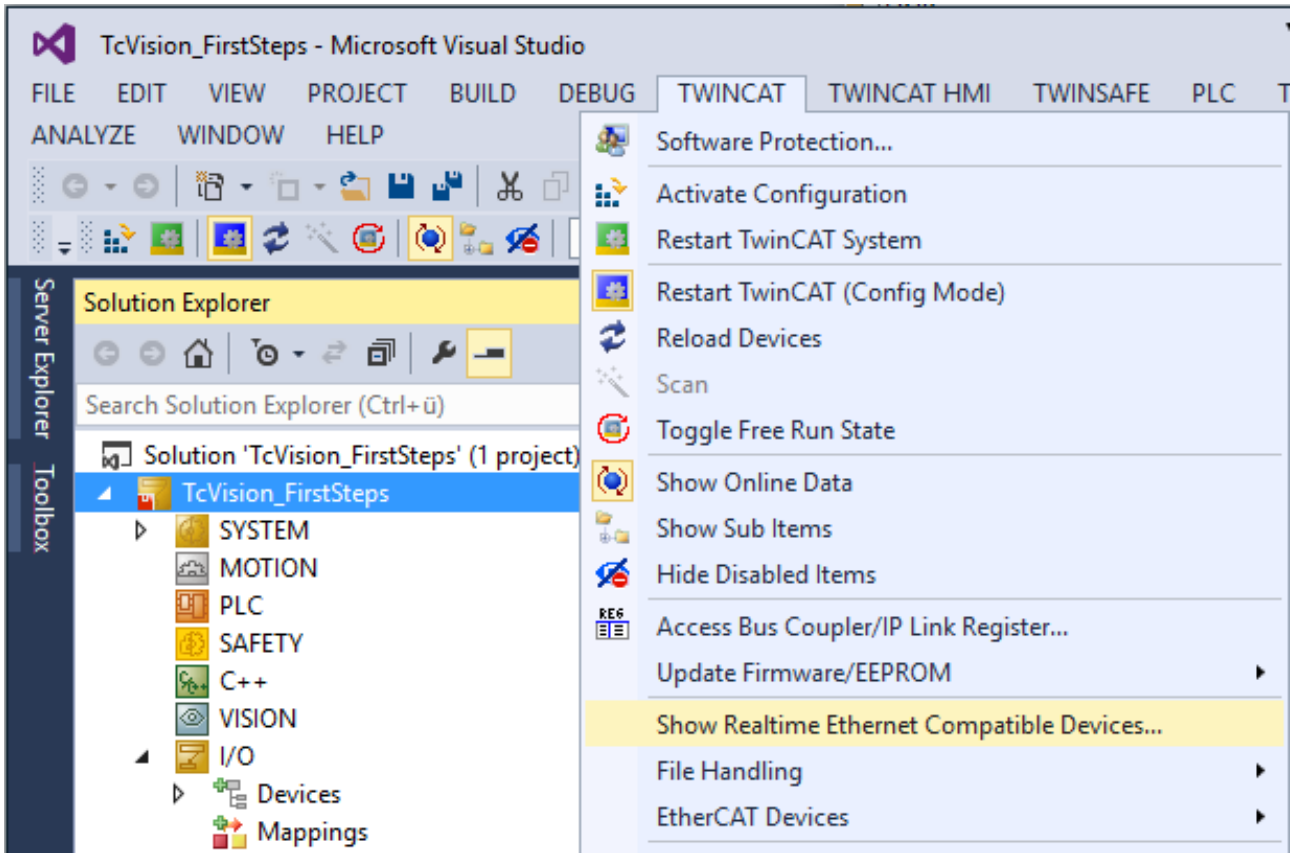
- 此外，还提供用于诊断网络通信的选项。这些内容在网络诊断 [▶ 63]一章中进行说明。

还请参阅有关此

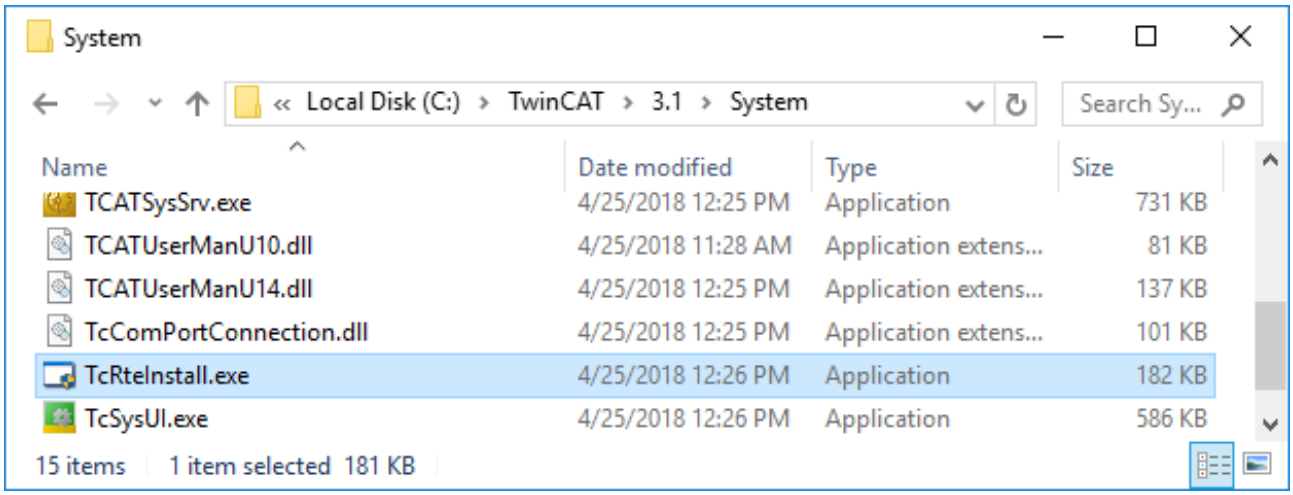
[▶ 62]

5.1.4.1 TwinCAT 实时以太网驱动程序的安装

与连接 EtherCAT 设备类似，必须在相应的网卡上安装 TwinCAT 实时以太网驱动程序，才能实现与 GigE Vision 设备的连接。要打开安装对话框，请点击 Visual Studio 菜单中的 **TWINCAT > Show real-time Ethernet Compatible Devices（TWINCAT > 显示实时以太网兼容设备）**。



或者，您也可以在目标系统上调用文件“C:\Program Files (x86)\Beckhoff\TwinCAT\3.1\System\TcRteInstall.exe”；此操作也会打开相应的安装助手。在 TwinCAT 3.1.4024 中，该文件默认位于“C:\TwinCAT\3.1\System\”下。

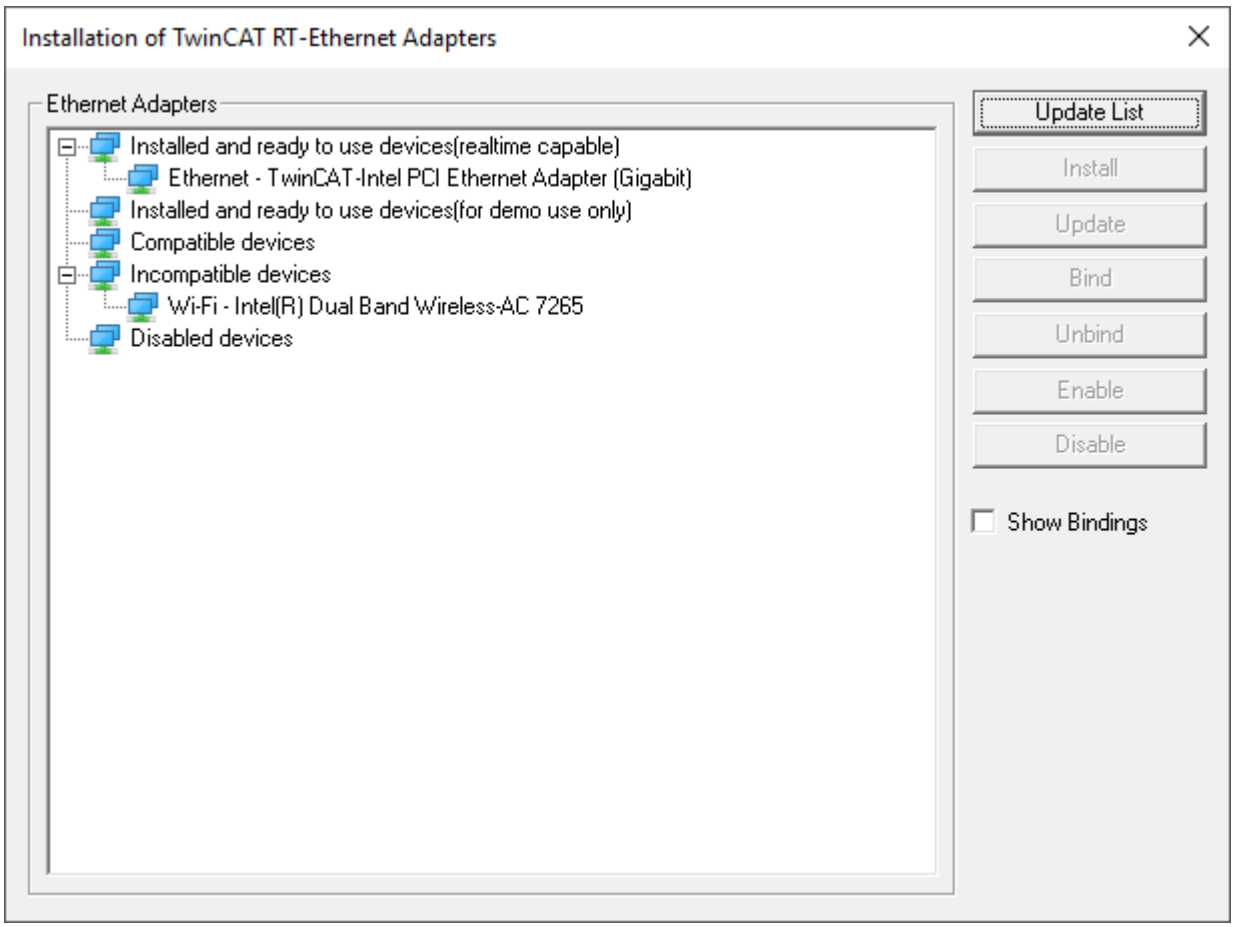


在安装助手中，确保在 **Installed and ready to use devices (real-time capable)（已安装即用设备（支持实时功能））**下列出相应的网卡。如果不是这种情况，请从**兼容设备**列表中选择一个合适的适配器，然后点击 **Install（安装）**。

注意

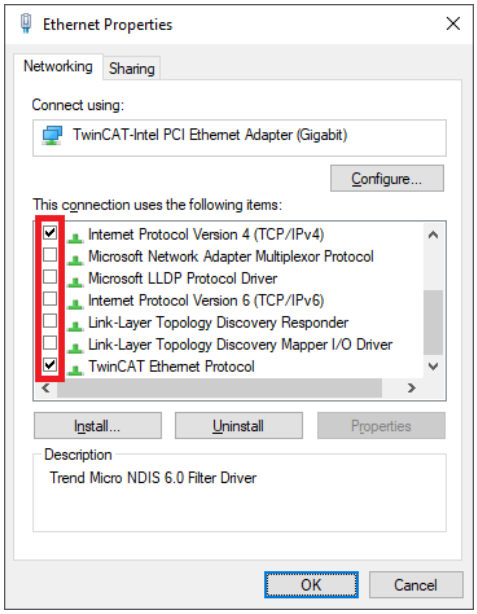
仅供演示使用

这种状态不足以保证 GigE Vision 设备的稳定连接连接和运行。如果安装后网卡只列在该组下，则必须使用其他网卡。



5.1.4.2 IP 配置和网络协议

建议将网卡限制为仅用于 TwinCAT 以太网协议，并为相机和网卡分配一个静态 IP 地址。如果多个协议都可以访问同一个网卡，那么这些协议也可能会发送数据包，从而对相机通信造成干扰。相关设置可在网卡的 Windows 属性中找到。



只有在将 DHCP 用于相机连接时才需要互联网协议版本 4（IPv4）。首次设置相机时，自动分配链路本地地址 (LLA) 会有所帮助。在所有其他情况下或完成初始相机配置后，也可以禁用 IPv4 协议并在 TwinCAT 的 IP 栈中手动配置静态 IP 地址，如下所述。

注意

禁用其他网络协议

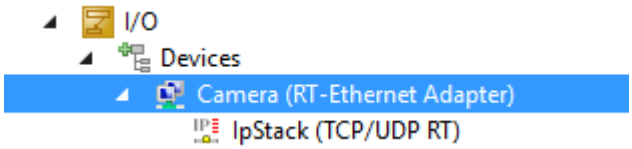
如果可能，在用于相机通信的所有以太网卡上，停用除 TwinCAT 以太网协议之外的所有网络协议。否则，这可能会导致与 GigE Vision 相机的通信出现中断和延迟。

在 TwinCAT RT 以太网卡中设置静态IP 地址

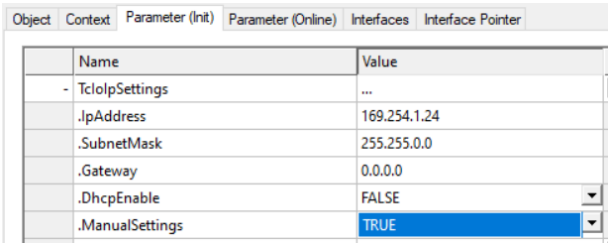
要在 TwinCAT 中为网卡手动设置静态 IP 地址，请按以下步骤操作：

如果 IPC 不为网卡自动分配 IP 地址，而您希望为网卡手动分配静态 IP 地址，则可在 TwinCAT 中按以下步骤操作：

1. 在 I/O 配置中打开 RT 以太网卡的 **IP 栈**。



2. 选择 **Parameter（参数）（Init）**选项卡，点击 `TcIoIpSettings` 前面的**加号（+）**显示更多参数。
3. 指定参数 `.IpAddress` 和 `.SubnetMask`。
4. 将参数 `.ManualSettings` 设置为 `TRUE`。
5. 务必加载或激活它们，设置才能生效。根据应用场景的不同，在 TwinCAT 中可以通过 **Reload Devices（重新加载设备）**、**Activate configuration（激活配置）**或 **Restart TwinCAT（重新启动 TwinCAT）（配置）**来实现。



如果在创建相机对象 [▶ 38]时手动分配 IP 地址，则会自动分配相同的设置。

注意

静态 IP 地址

为防止网卡的 IP 地址发生变化，请在 TwinCAT 项目的 IP 栈中指定一个静态 IP 地址。

5.1.4.3 巨型帧

标准以太网帧支持的最大有效载荷（最大传输单元 MTU）为 1500 字节。巨型帧能够使每个数据包传输的有效载荷大幅增加（通常可达 9000 字节），从而在传输大量数据（如图像数据）或在较高数据速率下传输时，显著提升性能。

优势如下：

- 效率更高：有效载荷与协议开销之比显著提高。
- 数据包处理工作量更少：每个图像的数据包更少，从而降低了处理数据包时的 CPU 负载。

要在相机通信中使用巨型帧，需要“TwinCAT-Intel® PCI™ Ethernet adapter（Gigabit）V2”驱动程序（TcI2xx）或“TwinCAT-Intel® PCI™ Ethernet adapter（10 Gigabit）”驱动程序（TcIXgbe），这两个驱动程序分别支持 4088 字节和 9014 字节的巨型数据包。驱动程序仅适用于兼容的 Intel® 控制器（例如 I210、I219 和 I226），完整列表请参见支持的网络控制器部分。

注意

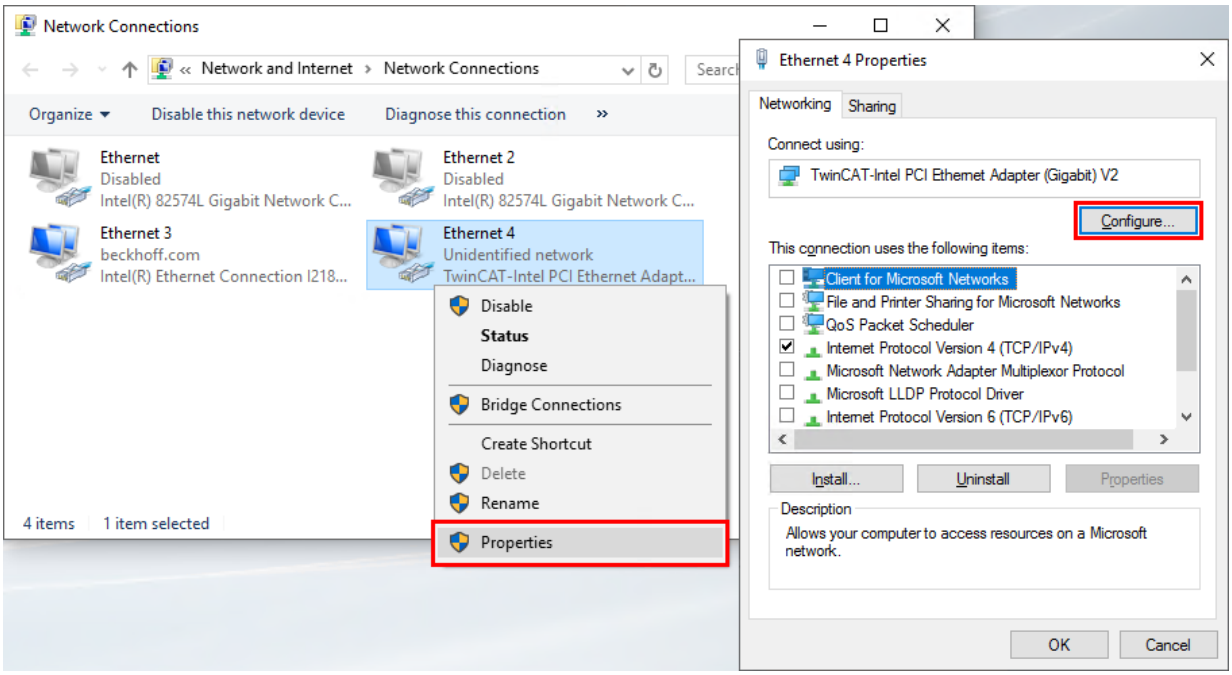
通讯网络中的完整巨型帧支持

从相机到 IPC 的整个数据通讯网络路径（包括所有交换机和网络接口卡）都必须支持巨型帧并为其进行主动配置。不支持巨型帧的网络组件可能会丢弃较大的数据包，从而导致无法确定的通信行为。在许多情况下，具体行为表现是图像传输失败，但其他通信功能正常。

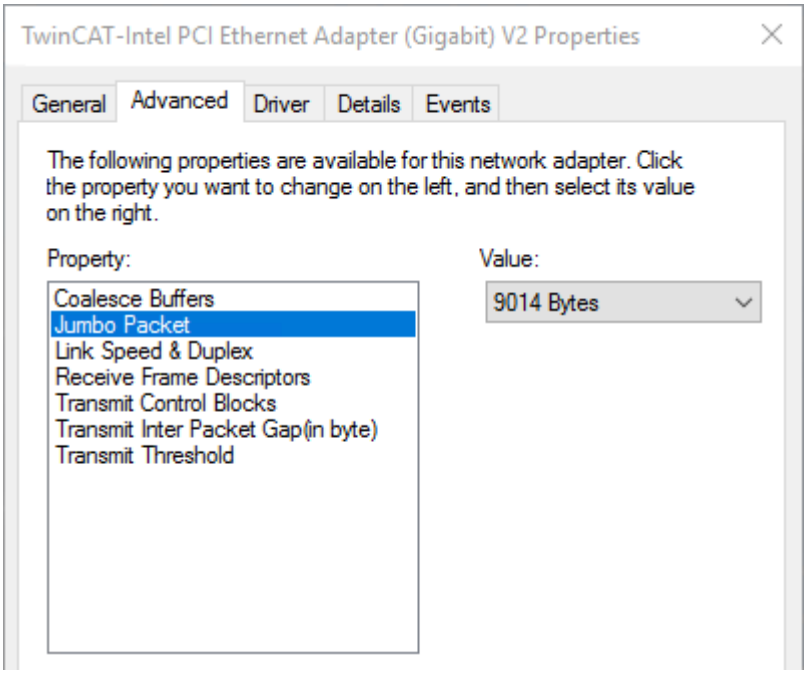
配置网卡

✓ 要安装 TwinCAT RT 以太网驱动程序，请遵循安装 TwinCAT 实时以太网驱动程序 [▶ 56]一章中的步骤进行操作。

1. 驱动程序安装完成后，在 IPC 的 Windows 窗口中打开“网络连接”（例如，通过 Windows 键 + R 和 ncpa.cpl 命令）
2. 右键点击所需的网卡并选择 **Properties（属性）**。
3. 在新窗口中，点击 **Configure...（配置...）**按钮



4. 在 **Advanced（高级）**选项卡中，将 **Jumbo Packet（巨型数据包）**设置为 9014 字节。此项设置仅限定了网卡能够接收的最大数据包大小。您仍可毫无问题地继续处理较小的数据包，因为该值仅限定了上限。

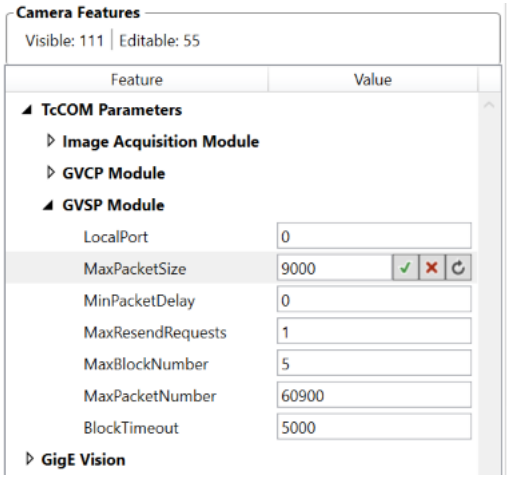


⇨ 网卡现已被设置为使用所选数据包大小的巨型帧。

相机对象的配置

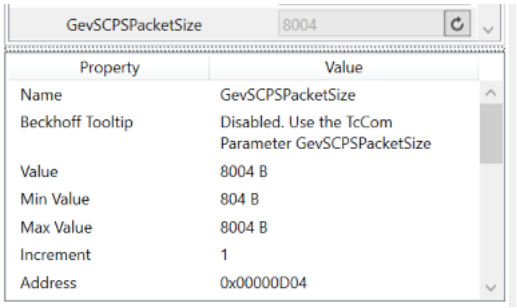
要在 TwinCAT Vision 中配置 GigE Vision 相机的数据包大小（`MaxPacketSize`），请按以下步骤操作：

1. 在 TwinCAT 项目中，打开所需 GigE Vision 相机的配置助手 [▶ 77]。
2. 在配置树中的 **TcCOM Parameters > GVSP Module > MaxPacketSize（TcCOM 参数 > GVSP 模块 > MaxPacketSize）**下，输入 9000 字节（其中 14 字节预留给标准以太网报头）。这仅仅限定了可能的最大数据包大小。系统随后会自动查询实际可用的值。如果相机支持更大的值，则使用输入的 9000 字节；否则使用相机所支持的最大值。



3. 点击  **Reload Devices（重新加载设备）**，重新加载配置。

4. 在配置树中的 **GigE Vision > GevSCPSPacketSize** 下，检查该值是否已正确传输，或者已确定的最大值是多少。在这种情况下，相机支持的最大数据包大小为 8004 字节。



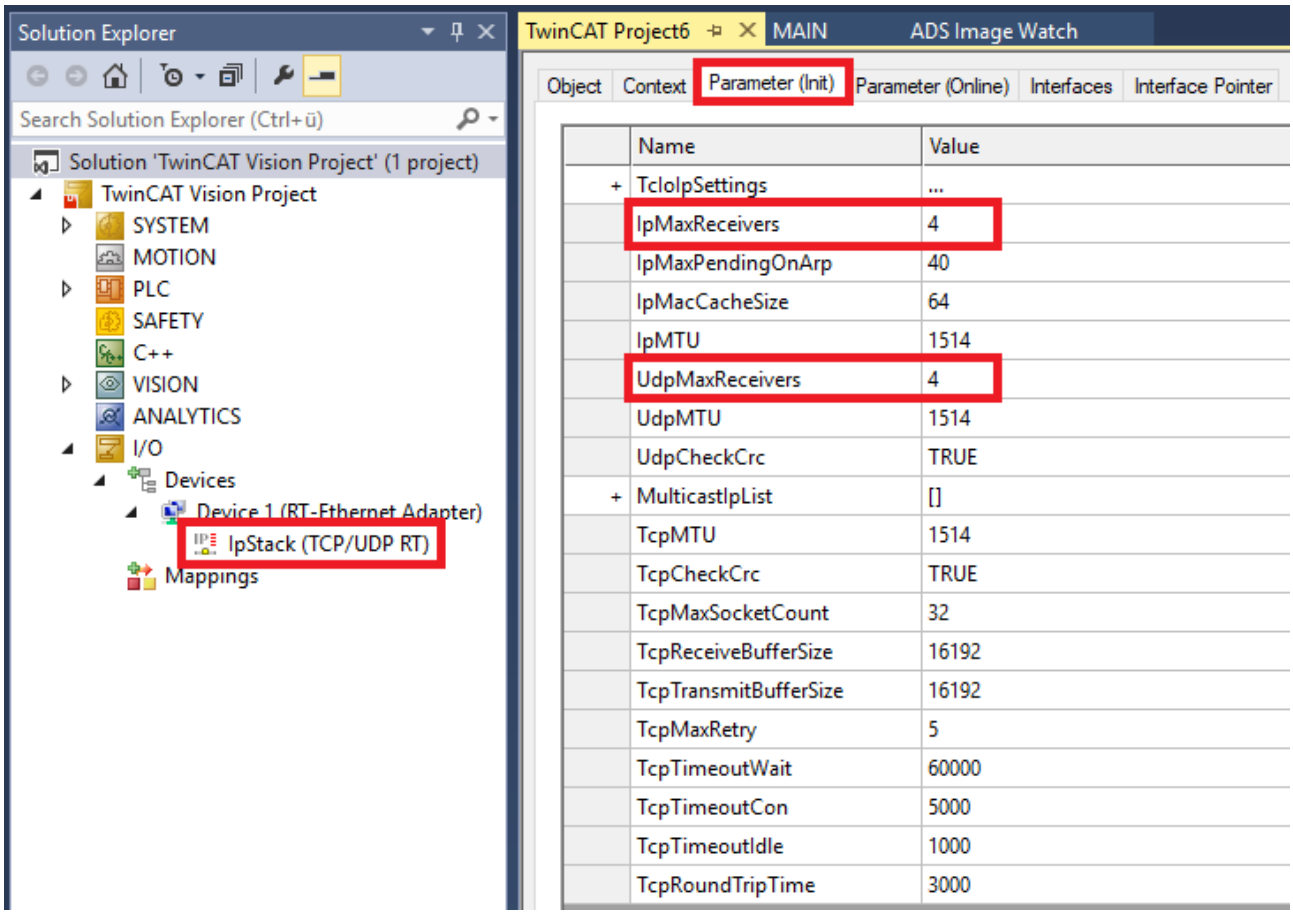
⇨ 相机现已被设置为使用配置的 `MaxPacketSize` 的巨型帧。

5.1.4.4 在一个端口上操作多台相机

如果您想要在同一个 RT 以太网卡上操作多台相机，则必须确保 UDP/IP 设置允许此操作。导航至 RT 以太网设备下的 UDP/IP 栈，然后选择 **Parameter (Init)（参数（Init））** 选项卡。

在出现的参数列表中，设置 `IpMaxReceivers` 和 `UdpMaxReceivers`，使其对应于通过该网卡连接的相机数量的两倍。这是因为每台相机有两个接收器（GVCP 模块和 GVSP 模块）。

默认设置支持两台相机连接，因此参数被设置为 `2 * 2 = 4`。



5.1.4.5

还请参阅有关此

- GigE Vision 标准版 [▶ 120]
- TcCOM 参数 [▶ 99]

巨型帧 [▶ 59]

5.1.4.6 网络诊断

如果连接出现问题，特别是发送或接收帧异常时，您可以在 RT 以太网卡中找到包含统计信息的选项卡，并在 IpStack 的 **Parameter (Online)（参数（在线））** 选项卡中找到含有各项计数器信息的内容。有关详细信息，请参见 相机通信 [▶ 2853] 的附录/故障排除部分。

混杂模式

混杂模式是 TwinCAT RT 以太网卡中的一项设置。使用 Wireshark® 工具进行调试时，需要启用此模式。在与 GigE Vision 设备通信时，此模式可能会引发问题。因此，仅应在创建 Wireshark® 记录时临时开启该模式，然后立即将其关闭。

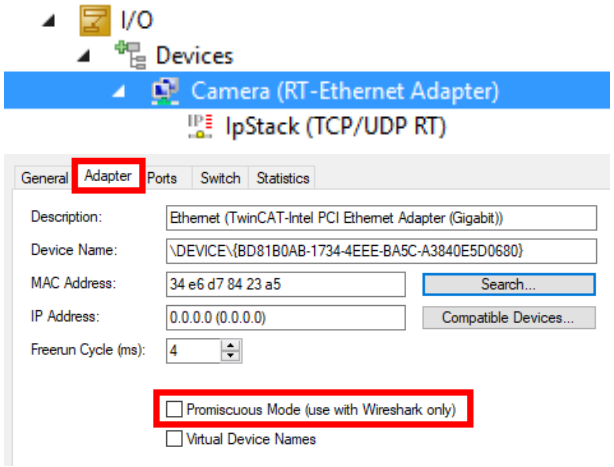
注意

避免使用混杂模式

在与 GigE Vision 设备通信时，此模式可能会引发问题。因此，在生产作业中使用 GigE Vision 设备时，切勿启用混杂模式！

通过在 TwinCAT 项目中打开相应的 RT 以太网卡并切换到 **Adapter（网卡）** 选项卡，您可以检查此设置。务必禁用 **Promiscuous Mode (use with Wireshark® only)（混杂模式（仅与 Wireshark® 一起使用））** 的复选框。

务必加载或激活它们，设置才能生效。根据应用场景的不同，在 TwinCAT 中可以通过 **Reload Devices（重新加载设备）**、**Activate configuration（激活配置）**或 **Restart TwinCAT（重新启动 TwinCAT）（配置）**来实现。



5.1.5 CPU 内核和任务

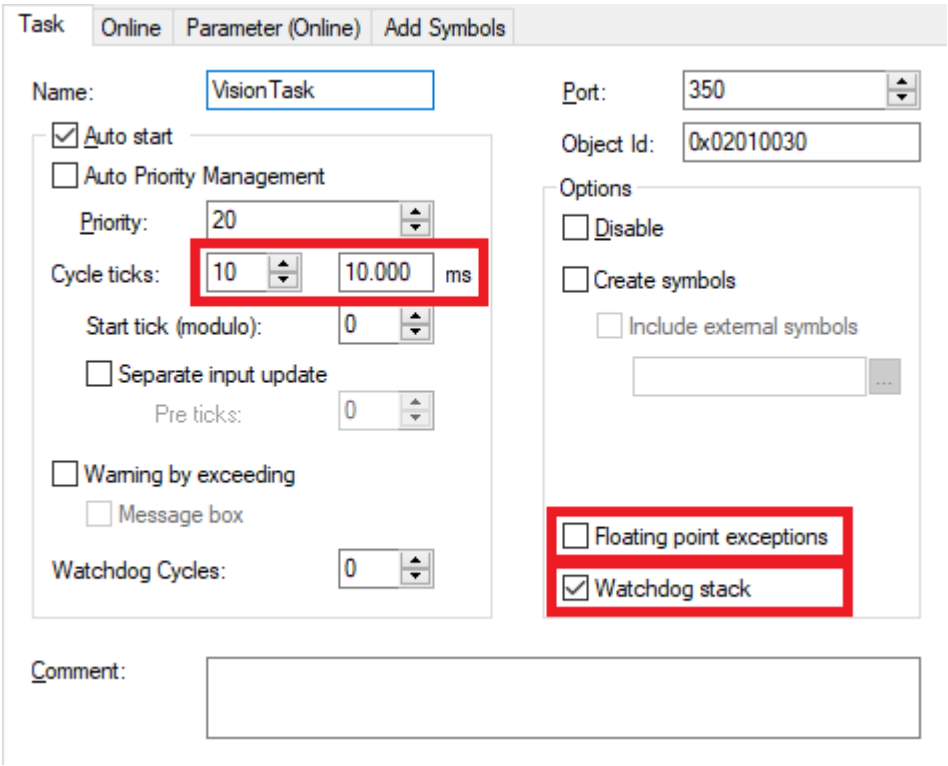
任务和 CPU 核的配置不为 TwinCAT Vision 所特有，对于所有 TwinCAT 组件均相同。有关如何操作的说明，请参见实时设置和任务设置。下文将介绍与 TwinCAT Vision 有关的问题和注意事项。

TwinCAT Vision 应用需要用于各种用途的各种任务：

- 执行 PLC 的循环任务
- 操作 Vision 设备的循环任务
- Job task 用于并行 PLC 执行
  （可选；此类任务将在工作任务一章中单独描述）。

配置 PLC 任务

基本上，TwinCAT Vision 所需的所有循环任务均自动创建。创建 PLC 时，系统会生成一个循环时间为 10 ms 的任务。这意味着 PLC 中的图像处理序列每 10 ms 重复一次。您可以在 **SYSTEM（系统） > Tasks（任务） > [TaskName]（[任务名称]）** 下更改任务设置。



循环刻度	循环刻度乘以所选 CPU 核的基准时间（默认为 1 ms）等于上述 PLC 的循环时间。
浮点异常	该设置指定 TwinCAT 是否检查浮点异常。 对于 Vision 应用，建议停用此选项。除此以外，这些错误信息没有说明异常的确切原因。
Watchdog（看门狗）栈	如果要在分配的 PLC 程序中使用 Watchdog（看门狗） [▶ 140]，则必须激活此设置。

⚠ 警告

浮点异常

对于主要执行 Vision 命令的 PLC 任务，强烈建议停用“Floating point exceptions（浮点异常）”选项。否则，某些 TwinCAT Vision API 功能可能会出现不必要的错误，导致系统中止。但是，如果运动控制或安全组件也在同一个 PLC 任务中执行，则必须激活此选项。

注意

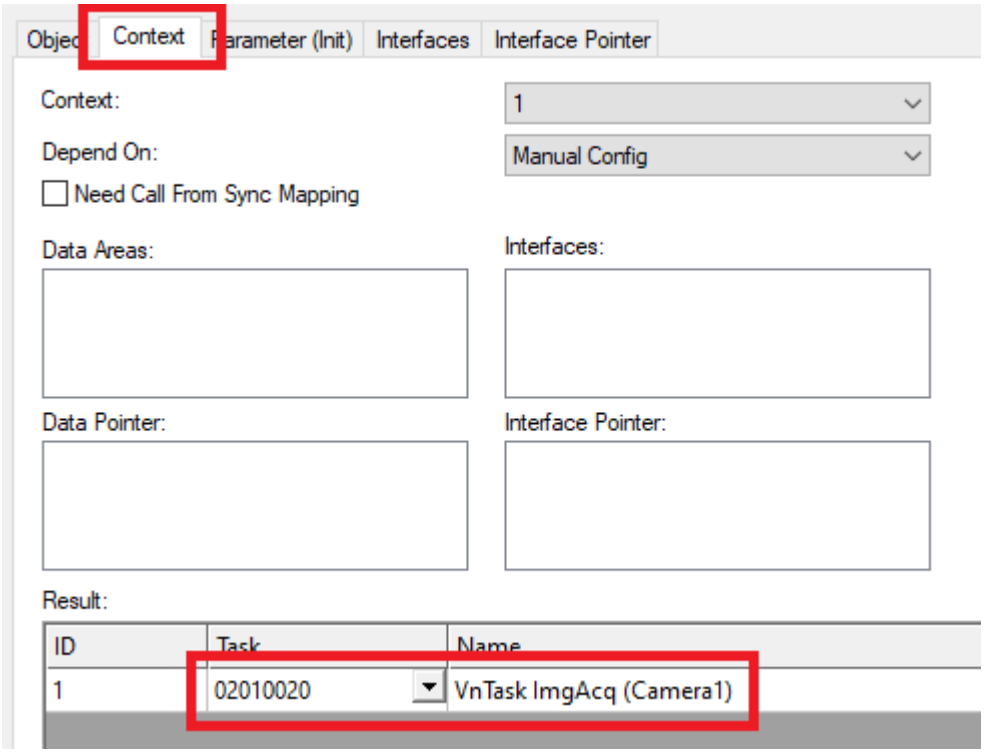
工作任务中的浮点异常

如果还为 Vision 应用使用工作任务 [▶ 66]，并且在 PLC 任务中禁用了“浮点异常”，则也必须在分配的工作任务中禁用此选项。否则仍会出现异常和系统崩溃。

设备任务配置

创建 Vision 设备时会自动生成所需任务。这些任务的循环时间可以如上所述进行调整。在大多数情况下，建议不要更改这些任务的循环时间。只有在特殊情况下才需要这样做。

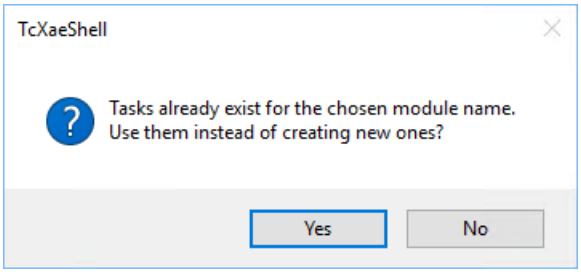
如果需要调整 Vision 设备与任务的链接，可在 GigE Vision 相机 [▶ 120]或文件源 [▶ 126]对象的相应 TcCOM 对象的 **Context（上下文）** 选项卡中进行：



如果使用下列 TcCOM 对象，则必须将它们与循环任务相链接：

- 相机图像采集 [▶ 121]
- 相机图像采集仿真 [▶ 121]
- ADS 通讯器 [▶ 121]
- 文件图像采集 [▶ 126]

如果在创建 Vision 设备时所选设备名称下已经存在合适的任务，则可以选择重复使用这些任务。在这种情况下会出现以下窗口；点击 **Yes（是）** 以确认重复使用：



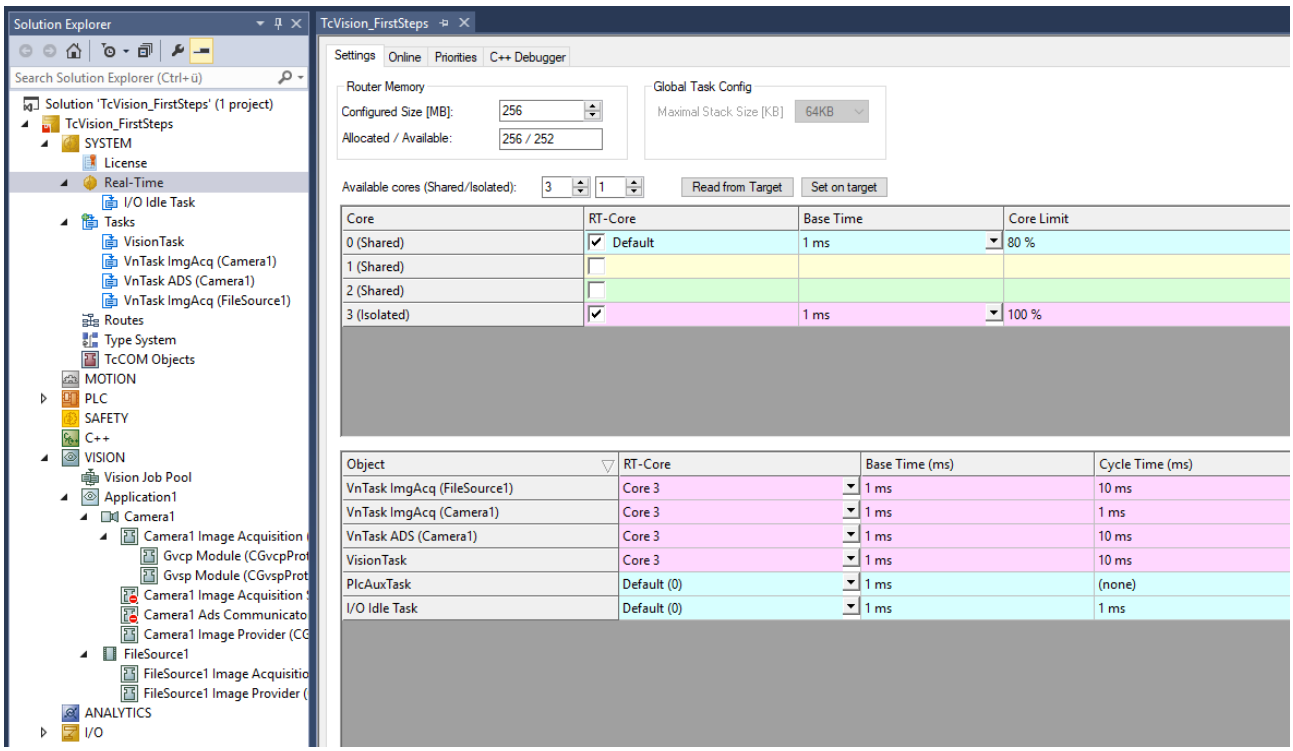
重复使用设备任务

在重复使用设备任务时，应确保它们不会同时被多个 Vision 设备使用。

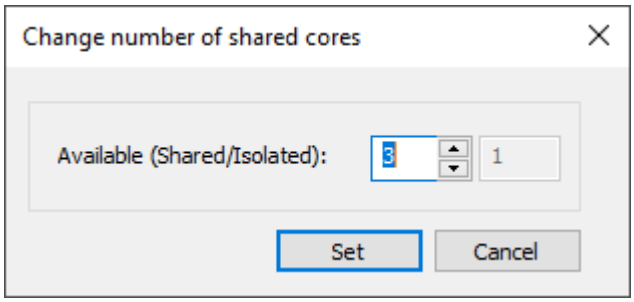
分配给 CPU 核

将任务分配给 CPU 核的操作在 TwinCAT 项目中的 **SYSTEM（系统）> Real-Time（实时）> Settings（设置）** 下进行。在多核系统中，TwinCAT 支持区分 Windows 系统分享核和独立核。使用 Windows 核时，操作系统和 TwinCAT 应用共享处理器时间。实时部分可限制为 10% 到 90%。与此相反，独立核可完全用于 TwinCAT 应用。因此，建议 Vision 应用使用独立核。

要显示目标系统当前的 CPU 核配置，请点击 **Read from Target（从目标读取）**。



要更改目标系统的 CPU 核配置，请点击 **Set on Target（在目标上设置）**并选择所需的分享核和独立核分配。



之后必须重新启动目标系统。

建议

我们建议为 Vision 任务（设备和 PLC）至少使用一个专用独立核。如果由于保存或加载图像（通过相机流、文件源控制器或直接从 PLC）而导致 TcVnService 负载较大，我们建议将一个核完全留用于 Windows，或严格限制一个核的实时部分。

如果为一个核分配了多个任务，则必须考虑优先级。一般来说，应为循环时间较短的任务分配较高优先级。

5.1.6 工作任务

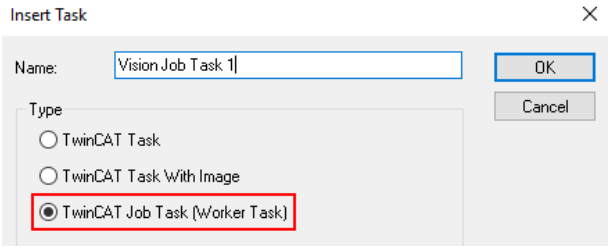
工作任务被 TwinCAT Vision 用于并行化功能。根据使用的系统、其配置、使用的 API 功能、图像内容和工作任务的数量，这可能导致执行时间的缩短。在API 参考资料 [▶ 132]中支持的功能通过注释“可以使用可用的 TwinCAT 工作任务来执行并行代码区域”进行标注。视觉工作池 [▶ 127]用于指定哪些工作任务被 TwinCAT Vision 使用。工作任务只需要创建和配置一次；此后，并行化将在运行时自动进行。

工作任务的优点

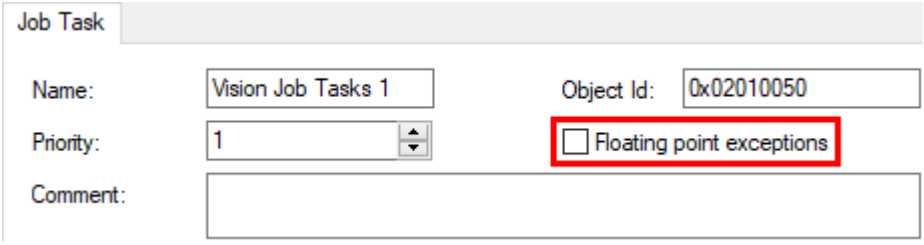
工作任务的实际优点必须在个别情况下检查。这取决于使用的系统、其配置、使用的 AP I功能、图像格式、图像内容和工作任务的数量。

使用

与其他任务一样，工作任务可以通过以下方式创建：**Solution Explorer > SYSTEM >（解决方案资源管理器 > 系统 >）**，右键点击 **Tasks > Add new item...（任务 > 添加新项目...）**。在后面的 **Insert Task Dialog（插入任务对话框）**中，选择类型 **TwinCAT Job Task (Worker Task)**。



然后，与所有其他任务一样，可以通过以下方式进行相应的 CPU 核分配：**Solution Explorer > SYSTEM > Real-Time（解决方案资源管理器 > 系统 > 实时）**。请遵守限制和建议 [▶ 67]部分中的说明。



注意

浮点异常

该设置指定 TwinCAT 是否检查浮点异常。此处应选择与相应 PLC 任务的配置 [▶ 63]中相同的选项。否则可能会出现异常和系统崩溃。

TwinCAT Vision 使用哪些工作任务可以通过 视觉工作池 [▶ 127] 的参数列表进行分配。一旦在解决方案资源管理器的 Vision 节点下创建了应用程序，即可在该节点下找到 Vision Job Pool。



优先次序

Vision Job Pool 访问的优先次序

如果多个并行执行的程序共享 Vision Job Pool，通常具有最高任务优先级的程序将获得对工作任务的访问权。不过，这种分配是动态的，也就是说，如果当前具有最高优先级的程序正处于顺序处理部分，则较低优先级的程序可以使用这些工作任务。一旦更高优先级的程序能够并行处理任务，较低优先级程序的任务就会被抢占。不过，较低优先级程序的主任务能够自行终止其已并行启动的任务，因此无需等待工作任务再次变为可用。

多个任务共享一个 CPU 核

如“限制和建议”章节所述，一个工作任务不应与其他任务共享同一个核心。否则，将遵循任务优先级规则。

限制条件

- 目前，工作任务仅能用于 TwinCAT Vision。
- 不得将多个任务分配给同一个 CPU 核。
- 工作任务不应与 Vision PLC 任务分配给同一个 CPU 核。
- 工作任务不应与 PlcAuxTask 分配给同一个 CPU 核。

建议

- 每个工作任务都应分配给一个专用的隔离 CPU 核。
- 不过，如果另一个循环任务与工作任务在同一个 CPU 核上运行，则该循环任务的优先级应高于工作任务的优先级。需要注意的是，这会限制并行化，在最坏的情况下，甚至会导致无法实现并行化。
- 不应将任何工作任务分配给“默认”CPU 核。

如果循环 TwinCAT 任务或 Windows 操作系统同时要求在 CPU 核上计算时间，则会因工作任务的负载变化而影响分析。

5.2 视觉节点

与图像处理相关的组件通过 VISION 节点及其子元件进行配置。它有用于服务配置 [▶ 68]和日志记录 [▶ 70]的选项卡。

显示 VISION 节点

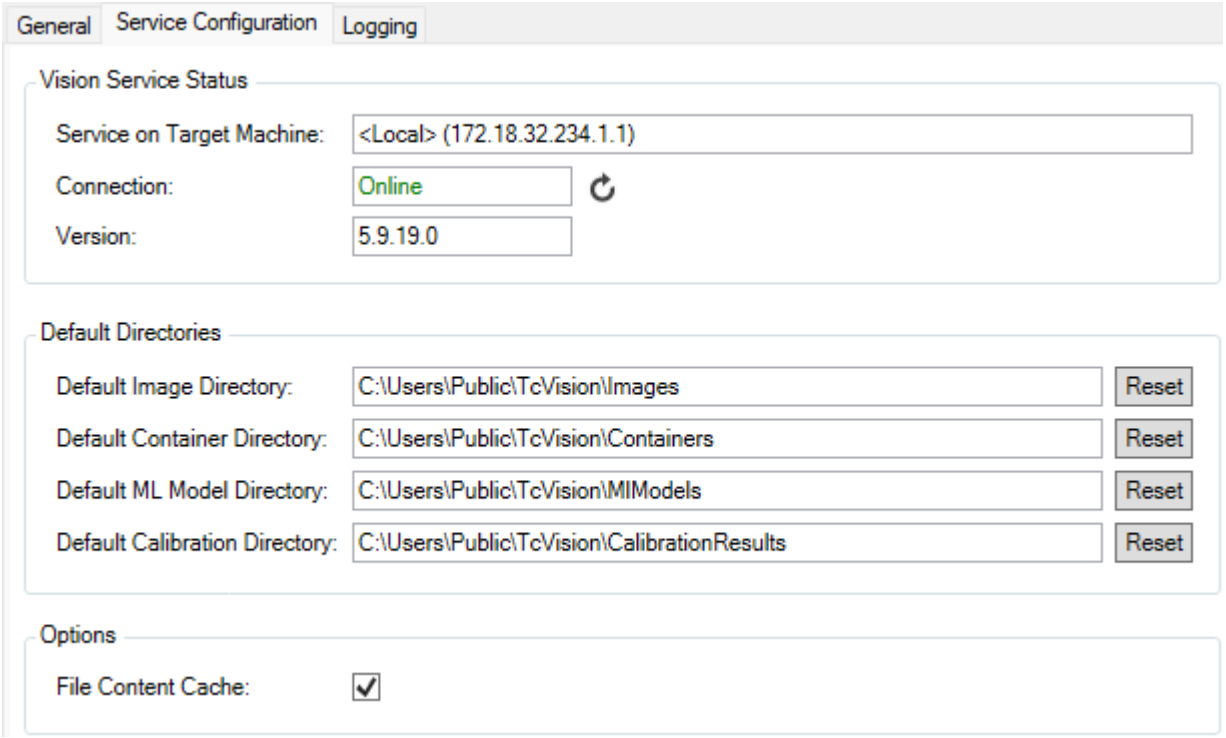
如果 TwinCAT 项目中未显示 VISION 节点，请右键点击 TwinCAT 项目并选择 **Show Hidden Configurations（显示隐藏配置） > Show VISION Configuration（显示 VISION 配置）**，以显示 Vision 节点。下一步是创建应用节点 [▶ 70]。

5.2.1 服务配置

Vision Service 为某些实时组件（如功能块）提供非实时对应组件。为此，Vision Service 执行一个 ADS 服务器，支持从实时环境和开发环境进行访问。

Vision Service 支持 TwinCAT Vision API 功能块从实时环境间接访问硬盘，例如加载或保存图像。

Vision Service 在 **VISION > Service Configuration（服务配置）**下进行配置。



Vision Service 状态

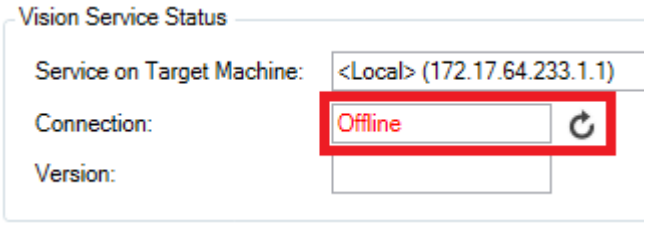
目标机构上的服务	运行 TwinCAT Vision Service 的目标系统。
连接	开发环境与服务之间的连接状态。这是衡量服务是否正常运行的良好指标。

版本	TwinCAT Vision Service 的版本号。

首次使用前，必须在目标系统上安装 Vision Service，请参见设置说明 [▶ 17]。

手动启动 Vision Service

Vision Service 通常与 TwinCAT 一起自动启动。但是，如果 Vision Service 处于脱机状态，则可以通过重启 TwinCAT 来重新激活（点击  或 ）。如果这样做不起作用，或者 Service 反复启动失败，则应在 TwinCAT 中重新注册；请参见：视觉服务不启动 [▶ 2843]。



默认目录

这些路径定义目标系统上的点，默认情况下，TwinCAT Vision 应在这些点上搜索文件并保存。

默认图像目录	功能块读写图像时使用的默认路径： FB_VN_ReadImage [▶ 1590] FB_VN_WriteImage [▶ 1596]
默认容器目录	功能块读写容器时使用的默认路径： FB_VN_ReadContainer [▶ 1589] FB_VN_WriteContainer [▶ 1595]
默认 ML 模型目录	功能块读写 ML 模型时使用的默认路径： FB_VN_ReadNeuralNetwork [▶ 1598] FB_VN_ReadMlModel [▶ 1591] FB_VN_WriteMlModel [▶ 1597]
默认标定目录	功能块读写标定结果时默认使用的路径： FB_VN_ReadCalibrationResult [▶ 1587] FB_VN_WriteCalibrationResult [▶ 1593]

所有重要路径的完整列表请参见 见重要路径 [▶ 2841] 部分。

Vision Service 日志

所有重要的服务事件均会写入目标系统上的日志文件 TcVnService.log，其位于 C:\ProgramData\Beckhoff\TcVnService 下。如果遇到有关 TwinCAT Vision Service 的支持问题，必须同时发送日志文件和行为发生时间。开发系统还配备一个适用于 Vision 节点下所有组件的日志记录器 [▶ 70]。

Vision Service 缓存设置

默认情况下，Vision Service 会缓存其读取的图像，以便在下次检索时更快提供请求图像。这种优化运行模式可满足大多数应用的要求，因此值得推荐。

如果图像内容发生变化，但文件名保持不变，则可以通过 **File Content Cache（文件内容缓存）**选项停用图像内容的缓存。

退出 Vision Service

要更改设置，必须首先通过任务管理器终止 TwinCAT Vision Service (TcVnService.exe)，否则将无法应用更改。之后，可在配置模式下重新启动该服务。

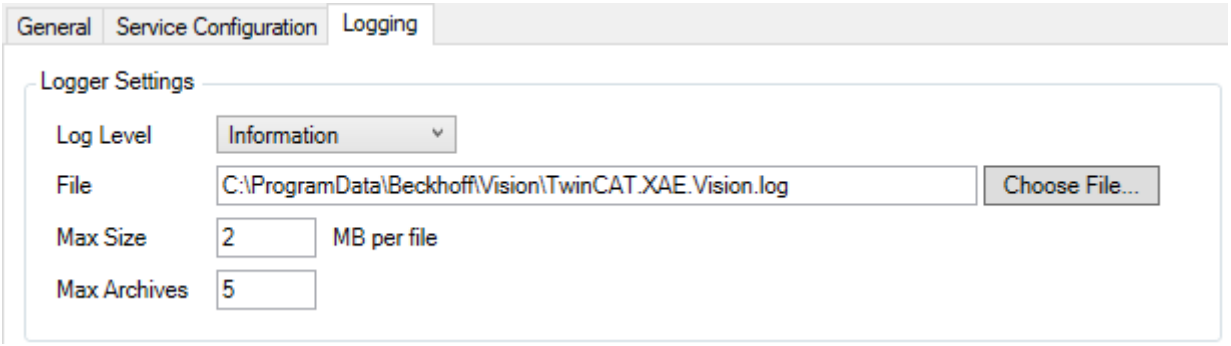
注意

数据不一致

确保同一文件不会被多个进程同时访问，否则会导致文件损坏或数据不一致。

5.2.2 记录

Vision 节点有一个日志记录器，根据日志级别的不同，它可以记录 VISION 节点下 TwinCAT Vision 组件的所有重要事件。设置可在 **VISION > Logging（日志记录）** 下找到。



可提供下列设置：

日志级别

级别	描述
错误	只记录错误信息。
警告	记录错误信息和警告。
信息	记录错误信息、警告和重要事件。[默认]
详细资讯	记录错误信息、警告和所有重要事件。

日志文件

默认情况下，名称为 TwinCAT.XAE.Vision.log 的日志文件保存在 C:\ProgramData\Beckhoff\Vision 目录中。这可以在 **Logging（日志记录）** 选项卡中更改。日志文件的最大大小和文件的最大数量也是如此。如果达到限制，将覆盖旧数据。

如想有关于 Vision 节点内行为的支持问询，请提供日志文件和行为发生时间。

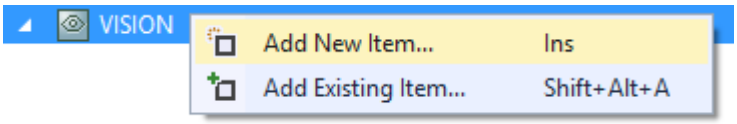
其他日志记录器

Vision Service [▶ 68] 的目标系统上还有一个日志记录器。此外，还有一个用于 TwinCAT Vision 安装的日志文件。所有相关日志记录路径均列于 见重要路径 [▶ 2841] 部分。

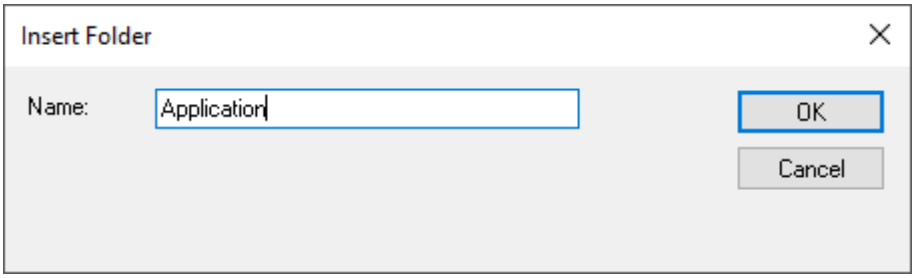
5.3 应用节点

一个 TwinCAT Vision 配置可以包含多个应用节点，所有这些节点都位于 VISION 节点内。创建应用节点的步骤如下：

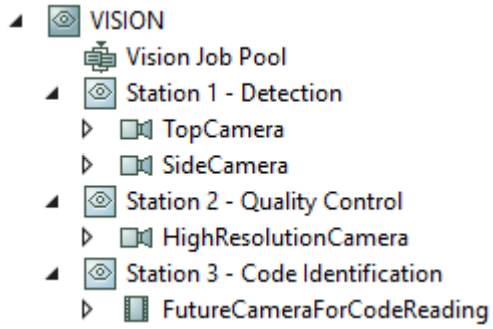
1. 右键点击 **VISION node（VISION 节点）→Add New Item...（添加新项目……）**



2. 输入应用节点的名称，然后点击 **OK（确定）**。



GigE Vision 相机节点 [▶ 71]和文件源节点 [▶ 123]在应用节点下创建。应用节点的目的是整合与主题相关的 Vision 设备。下图显示了一个包含多个 Vision 应用的结构示例：



此外，应用节点还支持单个 XTI 文件中多个 Vision 设备配置的一般导出/导入 [▶ 130]。

5.4 GigE Vision 相机对象

每个 GigE Vision 相机对象代表一个项目中的一台相机。一个项目可以包含多个相机对象。

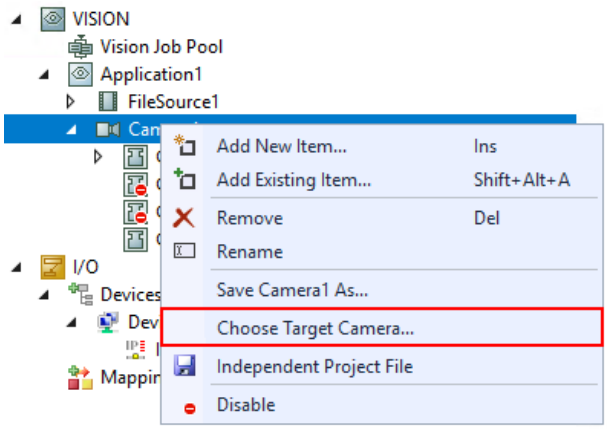
创建 GigE Vision 相机对象

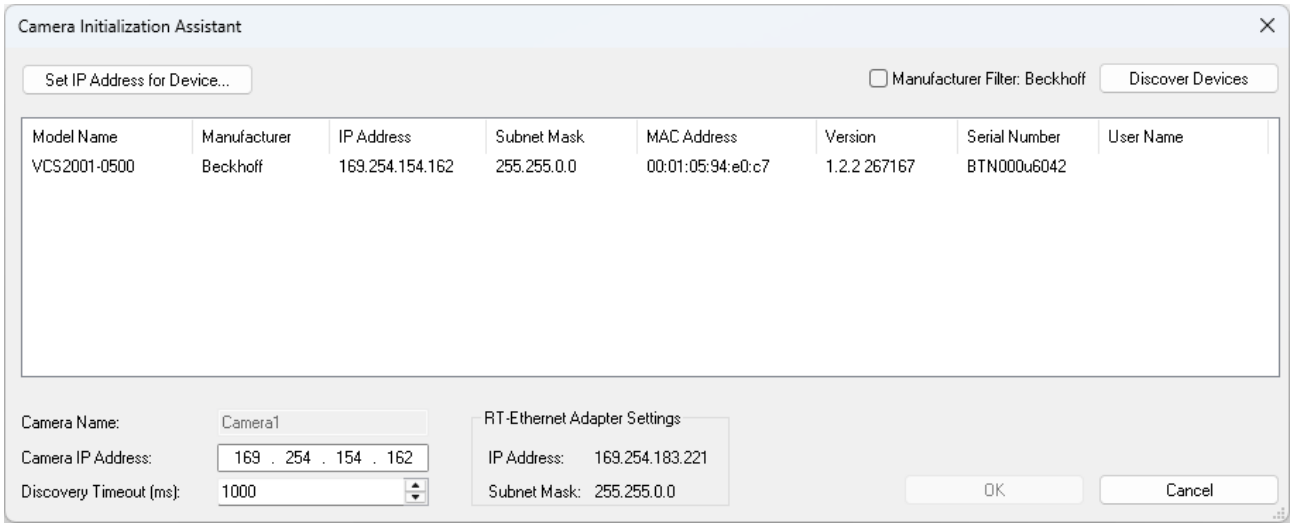
创建相机的说明请参见 创建 GigE Vision 相机 [▶ 38] 项下的 First Steps（第一步） [▶ 27]。在此请参阅网卡 [▶ 56]和设备任务 [▶ 63]配置的注意事项。

创建相机后，系统会自动尝试从相机读取 GenAPI 文件。此文件包含相机参数树的完整说明（请参见 GigE Vision 标准版 [▶ 120]）。在此说明的基础上，配置树便会显示在 配置助手 [▶ 77] 中，而相机也会读取当前的参数值。

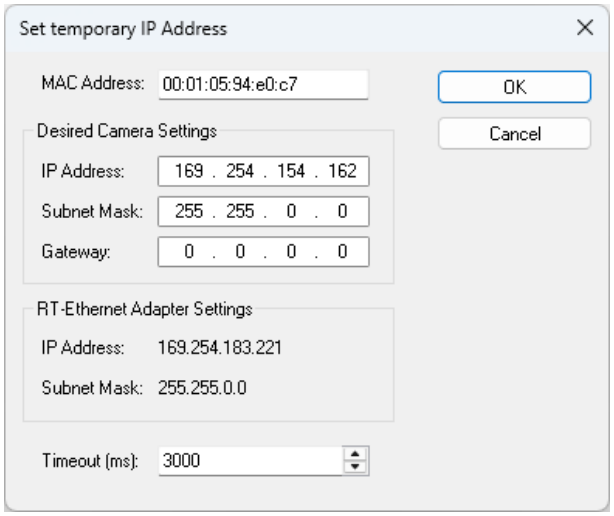
选择连接的相机

要将连接到目标系统的相机与 GigE Vision 相机对象相连，可右键单击相应的相机对象并选择 **Choose Target Camera（选择目标相机）**，打开 **Camera Initialization Assistant（相机初始化助手）**。或者，也可以通过 常规 [▶ 73] 选项卡上的 Connection Options（连接选项）打开此对话框。





相机名称	显示所选相机对象的名称。
相机 IP 地址	保存在 TcCOM 参数中的 IP 地址。该条目将在选择相机时被覆盖，并在模块中更新为 **OK**。
发现超时	Discovery Timeout（发现超时）描述的是与相应网络端口相连的相机作出响应所允许的时间。 单位：毫秒 (ms) 默认：1000
设置设备的 IP 地址	打开以下 Set IP Address（设置 IP 地址）对话框，可临时设置相机的 IP 地址和子网掩码。仅更改（强制更改）当前正在使用的 IP 地址，此项更改并非永久生效。指定的 IP 地址和子网掩码将保存在 Image Acquisition Object（图像采集对象） [▶ 99]的 TcCOM 参数`CameraIPAddress` 中。
制造商筛选：倍福	启用筛选，使结果表中只显示倍福制造的相机。
发现设备	搜索程序从 **Discover Devices（发现设备）**开始。
找到的相机连接结果表	读出找到的相机列表并显示以下相机信息： 型号名称、制造商、IP 地址、子网掩码、MAC 地址、版本、序列号和用户标识
RT 以太网卡设置	连接到相机的 IPC 上网卡的当前 IP 地址和子网掩码。



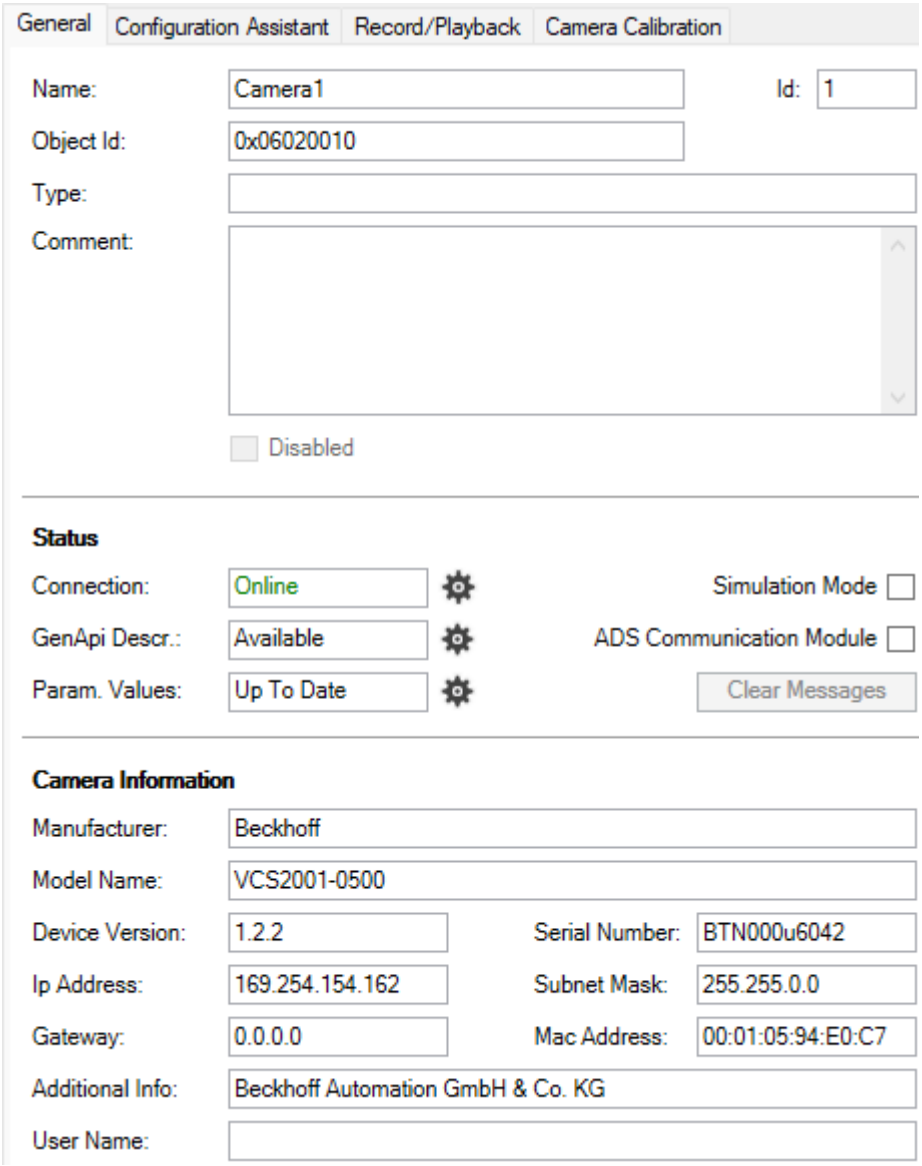
MAC 地址	所选相机的 MAC 地址。
所需的相机设置	输入相机要设置的所需 IP 地址、子网掩码和网关。
RT 以太网卡设置	连接到相机的 IPC 上网卡的当前 IP 地址和子网掩码。
超时	相机响应强制 IP 设置命令的超时时间。

导出/导入 GigE Vision 相机对象

如果要保存相机的配置或将其导入到项目中，请按照 Exporting/importing configurations（导出/导入配置）[▶ 130]一章中的说明进行操作。

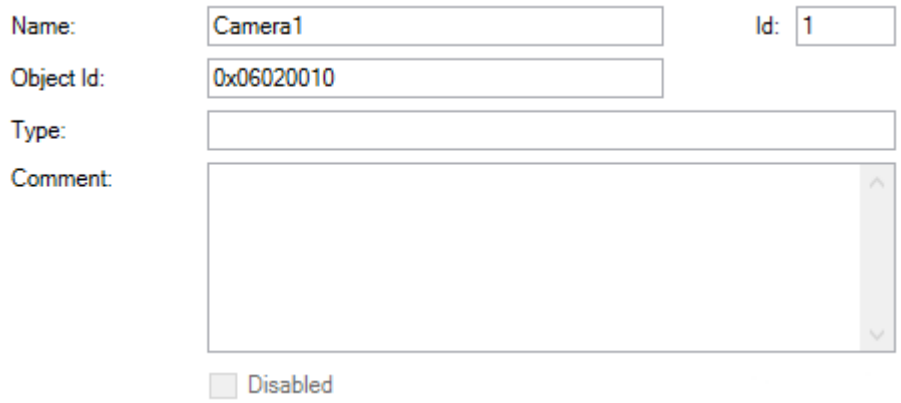
5.4.1 常规

General（常规）选项卡提供了 TwinCAT 中相机对象的概览、已连接的相机、正在使用的任务，以及 RT 以太网卡或 IpStack 的相关信息。



常规对象信息

上半部分包含有关相机对象的常规信息：



名称	选定的对象名称。 **不要重复分配对象名称！**
对象 ID	自动分配的 32 位识别号码，在项目内部是唯一的。
类型	对象类型。

注释	用户可自由编辑的注释字段。
Id	相机模块的识别号码，按顺序分配。如果某个相机模块被删除，该识别号码将再次变为可用，并可以将其分配给新的相机模块。

相机实例的状态

中间部分包含相机实例的状态信息：



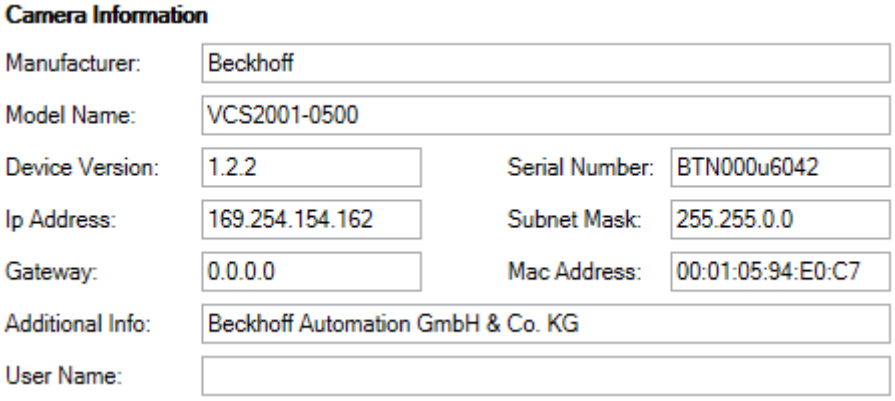
下表说明了各个元素：

连接	显示与相机的连接状态。 通过  按钮，可使用 **Refresh Camera Status（刷新相机状态）**更新显示内容。此外，还可以使用 **Choose Target Camera（选择目标相机）**打开用于选择相机的对话框 可能的状态： • **Unknown（未知）**：无法检查与相机的连接情况。相机模块可能未处于运行状态，或者相机模块的创建可能未完成。 • **Online（在线）**：已建立与相机的连接。 • **Offline（离线）**：没有连接相机。 • **Disabled（已禁用）**：相机节点已禁用。与相机没有通信。 • **Not in Config Mode（非配置模式）**：此处的连接状态仅在 TwinCAT 配置模式下才能显示。 • **Loading（加载中）**：相机模块正在初始化。
GenApi 描述	这指明了 GenApi 结构（配置助手 [▶ 77] 中的功能树）是否已成功创建。状态选项： • **Available（可用）**：GenApi 结构已成功创建。 • **Not Available（不可用）**：GenApi 结构未创建。无法显示任何功能。 按钮  还允许执行以下操作： • **Read GenApi Description from Camera（从相机读取 GenApi 描述）**：读取相机的 GenApi 描述。同时，配置树 [▶ 81]中的所有功能将被重置。 • **Export GenApi Description（导出 GenApi 描述）**：相机的 GenApi 描述被保存到本地。

参数 值	状态选项： • **Up To Date（最新）**：相机的所有功能均有取值且处于最新状态（相机在线）。 • **Complete（完整）**：相机的所有功能均有取值，但这些取值可能已过时（相机离线）。 • **Incomplete（不完整）**：并非相机的所有功能都有取值。 按钮  还允许执行以下操作： • **Write Local Changes（写入本地更改）**：所有偏离相机值（在配置树 [▶ 81]中以橙色标记）的本地参数值都会写入相机。 • **Revert Local Changes（还原本地更改）**：功能树中任何尚未写入相机的本地更改将被重置。
仿真模式	在仿真模式下，可以将来自录制的相机视频流的图像发送到 PLC，而非由真实相机提供图像（参见记录/回放 [▶ 101]）。 如果仿真模式已激活，则图像采集对象将自动取消激活，同时图像采集仿真对象将被激活。 如果仿真模式已取消激活，则图像采集对象将自动激活，同时图像采集仿真对象将被取消激活。 相应的对象链接（参见相机详情 [▶ 120]）将各自自动进行调整。 默认：禁用。
ADS 通信模块	录制相机视频流所需，参见 记录/回放 [▶ 101]。 此复选框仅用于激活或取消激活相机对象的 Ads Communicator 对象。 **请注意，如果 Ads Communicator 对象已激活，相机采集的图像仍会在短时间内被内部引用，因此无法转换为可显示图像。** 更多详细信息，请参见可显示图像 [▶ 145]。 默认：禁用。
清除消息	使用此按钮可以删除此相机对象写入的消息。 仅当存在消息时，此按钮才会启用。

相机信息

下半部分会显示从相机中读取到的相机常规信息（例如相机的制造商和型号名称）。此信息也可在配置树 [▶ 81]中找到。

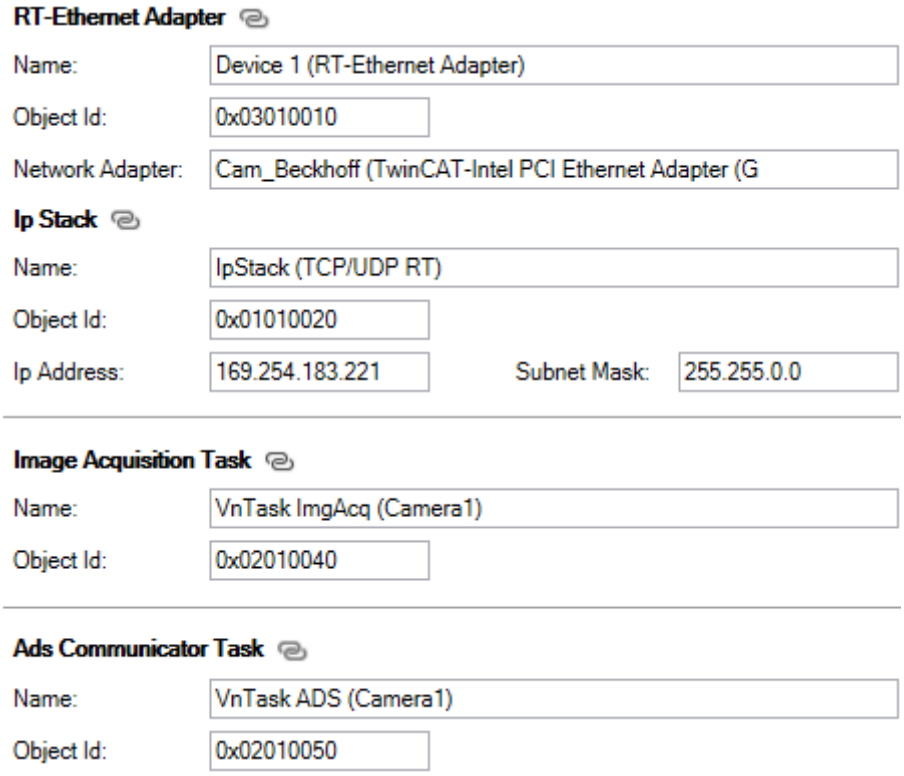


有关链接对象的信息

此部分包含有关所使用的任务以及 RT 以太网卡或 IP 栈的信息。每个对象都会显示相应的名称和对象 ID。自 TwinCAT 3.1.4026.20 起，名称旁边还会显示一个链接符号，可用于在 System Manager 中直接跳转到链接对象。

对于 RT 以太网卡，还会显示所使用的网卡的名称（由操作系统识别）。在 IP 栈下方会显示当前分配的 IP 地址。

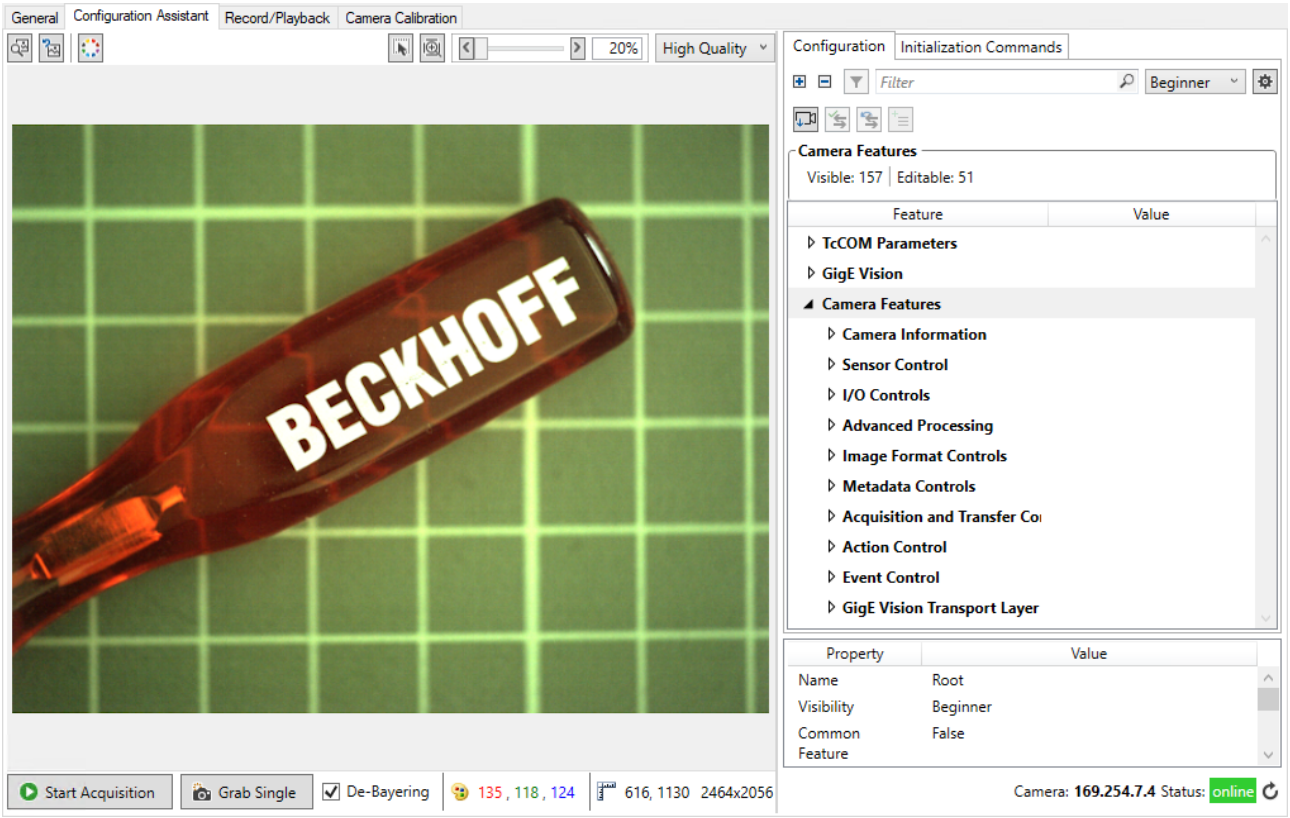
打开相机对象时，根据所使用的每个对象 ID 在 System Manager 中进行搜索。如果未找到该对象或者如果对象 ID 本身为 0，则会显示一个警告符号以代替链接，并且对象 ID 会显示为橙色。



5.4.2 配置助手

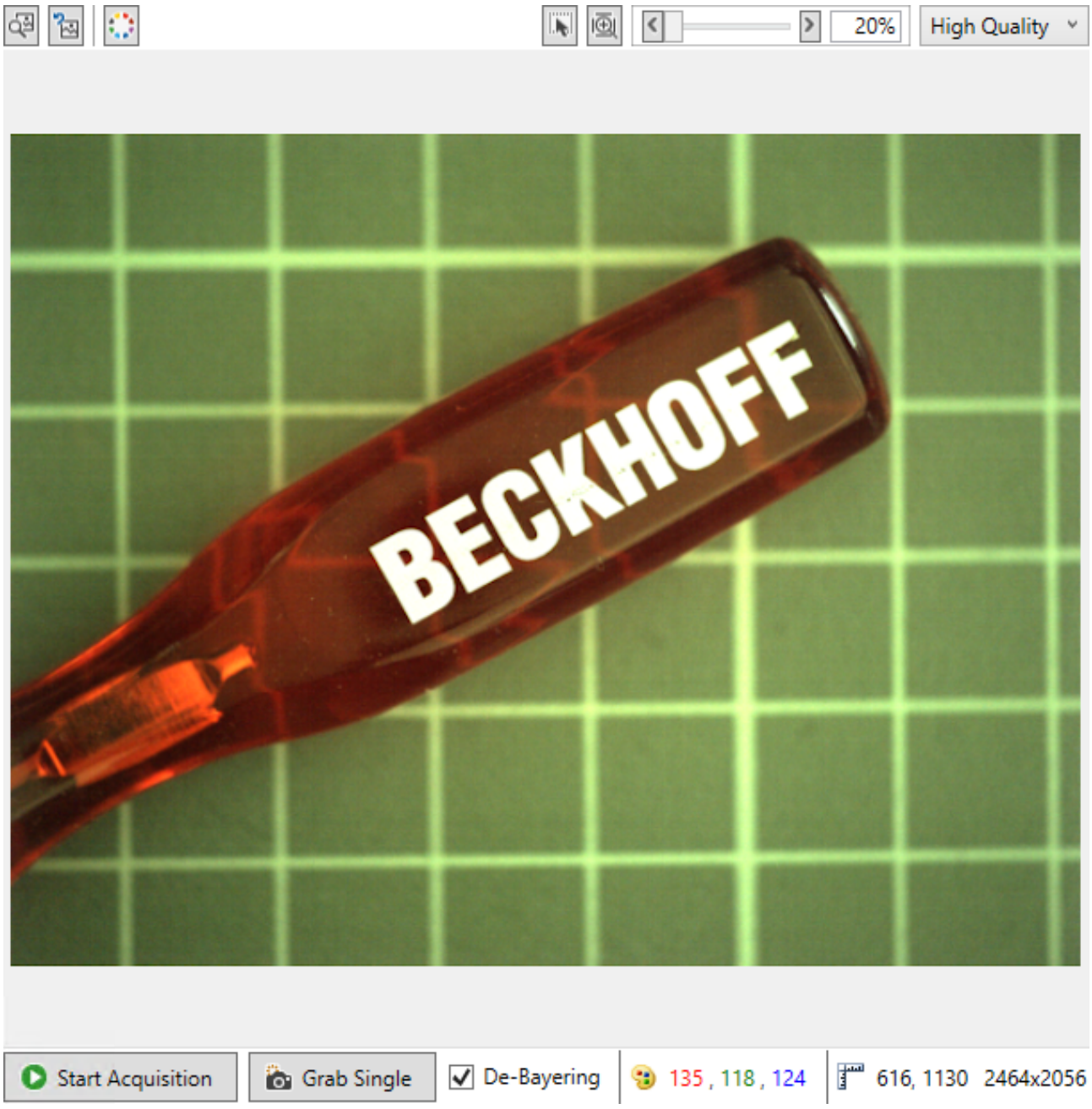
相机参数和通信行为都可以在**配置助手**选项卡上进行配置。配置助手有四个部分：

- 图片预览 [▶ 78]
- 配置树 [▶ 81]
- 初始化命令 [▶ 89]
- 行动和设置 [▶ 87]
- TcCOM 参数 [▶ 99]



5.4.2.1 图片预览

Configuration Assistant（配置助手）左侧区域包括一个预览窗口和一些操作元件。



在该图像预览中，只能正确显示 8 位整数图像。

下栏

预览窗口下方是开始/停止预览的控制器，以及光标在图像上的位置信息：

开始/停止采集	用于开始或停止采集图像的按钮。
抓取单个	用于触发单个图像的按钮。
做贝叶斯处理 De-Bayering	Bayer 格式的图像可以作为 RGB 图像进行预览。此设置默认根据所使用的像素格式进行设置，并且可以手动取消激活。 如果在相机功能中选择了 Bayer 像素格式，但接收到的是不同的格式，则会显示一条警告，提示格式不匹配。

	如果启用此选项，则会转换相机拍摄的下一个图像，而不会转换当前图像。
像素值	此处表示图像上鼠标光标位置处像素的颜色或强度值。
光标位置	显示鼠标光标在图像上的位置。请注意，图像的零点始终位于左上方。
图像大小	此处显示的图像大小以像素为单位。

图像采集和触发设置

确保相机的图像采集和触发设置是以相机能对命令作出反应的方式选择的。
例如，如果相机的“触发”模式已激活，则相机不会在图像采集开始时自动发送图像。详情请参见示例 图像采集和触发 [▶ 2831]。

上栏

在预览图像上方有一些控制器，用于方便相机配置和调整图像显示。

相机配置

设置 ROI	点击该按钮后，按住鼠标左键可直接在预览图像中绘制 ROI。此 ROI 会被写入相机参数，因此下一个图像发送时就会包含所设置的 ROI。
重置 ROI	点击该按钮后，先前设置的 ROI 会被删除，即图像大小和偏移量会被设置为相机的默认值。
白平衡	点击操作将在彩色相机上执行白平衡。相机必须支持以下 GenAPI 参数：BalanceRatioSelector 和 BalanceRatio（或旧相机的 BalanceRatioAbs）。此外，相机图像中必须包含一个白色（非曝光过度或反光）物体，如一张纸，以便进行白平衡。 在白平衡之后，必要的数值会保存在命名的 GenAPI 参数中，并可在 Initialization Commands（初始化命令） [▶ 89]或 UserSets 中采用。相机上的自动持久存储可能是相关相机的特有功能，但不是标准功能。 该按钮只有在存在有效颜色格式（三通道图像为 RGB 或单通道图像为 Bayer 格式）时才会激活。

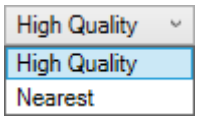
显示图像大小



图像显示尺寸的调整选项。使用滑块或在文本字段中输入数值，可在原始尺寸的 20% 到 6400% 之间缩放图像。点击左键后，按住鼠标左键可直接在预览图像中拖动矩形区域。然后相应调整缩放比例，以便在预览中显示所选区域。点击左起第二个 **"Fit to window"**（调整至窗口大小）按钮，便会自动选择显示尺寸，从而使预览尽可能充分利用屏幕上的可用空间。

插补

在此处，您可以选择不同的插补方法，将图像从原始尺寸缩放到显示尺寸：



高质量	高质量插值，边缘和像素过渡平滑。
局部最近	通过采用局部最近的像素值进行插补，从而直观地显示单个像素并对其进行分隔。此方法可用于检查像素值。

右键点击图像

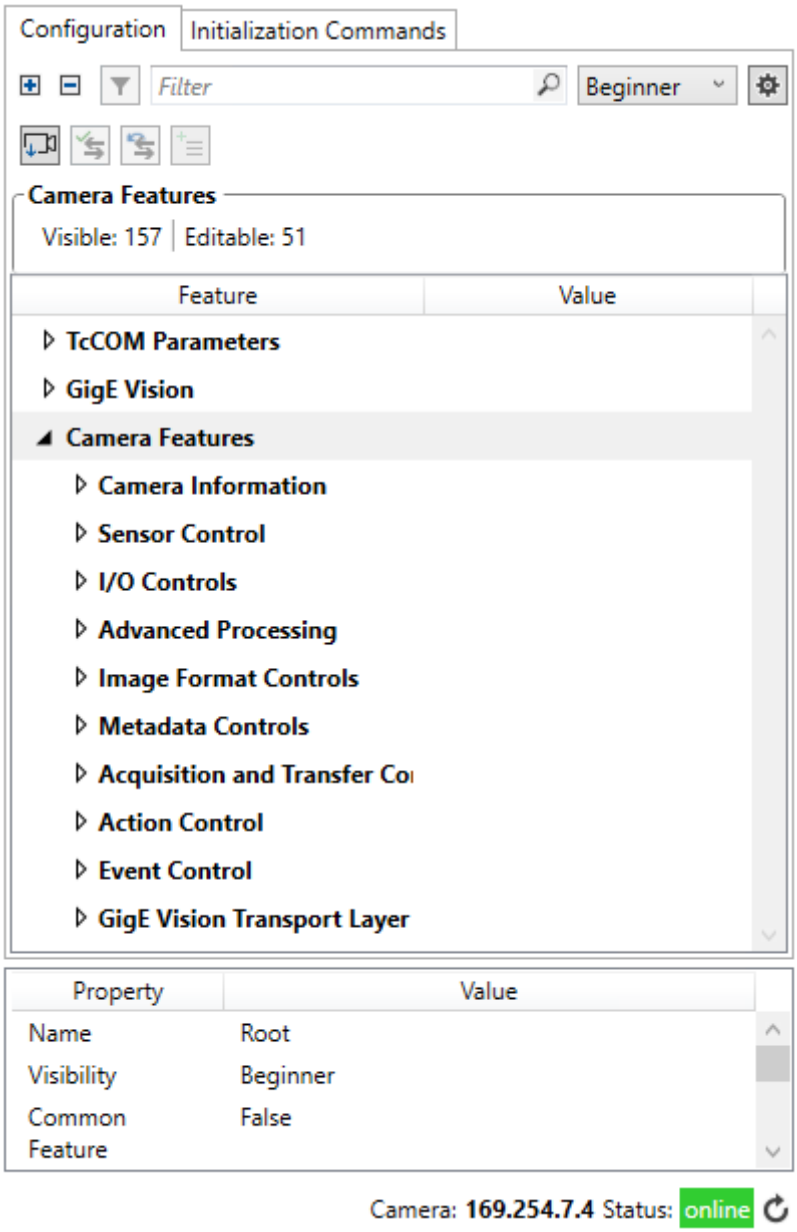


开始采集	开始图像采集。
停止采集	停止图像采集。
保存当前图像	将图像保存为位图、JPEG 或 PNG。

5.4.2.2 配置

Configuration（配置）选项卡由 3 个区域组成，分别如下所述：

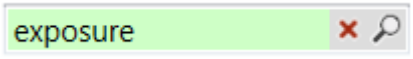
- 筛选器和控制器
- 配置树
- 属性窗口



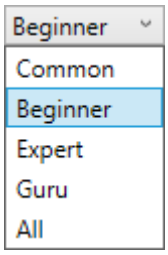
筛选条目



借助筛选选项，可以在配置树和初始化命令列表中快速轻松的找到各功能：

- 点击  符号可展开所有树元素。所有参数均可见。
- 点击  符号可折叠所有树元素。只显示主要组别。
- 点击  图标会对树项目进行筛选，只显示有变化的行。该按钮只有在显示更改时才会启用。
- 搜索行 

会按特定字符串或（十六进制）地址对参数进行筛选。输入字符串时，文本框背景为白色，同时会出现一个用于删除或重置筛选器的附加按钮。如果通过按 Enter（回车键）或点击筛选器按钮确认条目，那么文本框也会以绿色高亮显示。这表示当前字符串是否与激活的筛选器相对应，因此也是树元素的视图。
- 根据 GenICam 标准，可选择 **Beginner（初级）**、**Expert（专家）**和 **Guru（大师）**三个可视级别。每个功能的可视级别都存储在相机的 GenApi 文件中。此外，还有 **All（全部）**条目，可显示所有条目，包括不可见的功能。还有一个 **Common（常用）**条目，只会显示预定义或自定义的功能。



常用功能列表

Common Features List（常用功能列表）是预定义列表，包含配置相机最常用或最经常使用的功能。预定义寄存器使用 GenICam Standard Features Naming Convention（标准功能命名规则，SFNC）中的名称。因此，根据各相机基本支持的功能以及是否使用了标准名称，编号可能会有所不同。

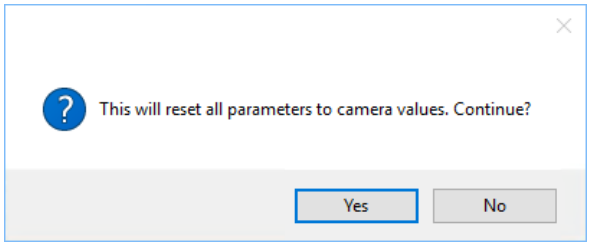
所包含的功能列表可单独扩展或缩减。要添加相机功能，请选择一个条目并点击  按钮进行添加。如果列表中已包含该功能，则可使用  按钮将其删除。所有更改都会保存在 TwinCAT 项目中。无法添加类别和选择器。

可通过操作菜单中的“Reset Common Features configuration（重置常用功能配置）”条目将列表重置为预定义状态。配置命令和初始化命令都可以选择“Common（常用）”列表作为滤波器。

控制器



这些按钮的功能从左到右依次为：

读取相机参数	所有参数值均从相机读取并写入配置树。这必须首先通过以下对话框确认： 
写入本地更改	所有偏离相机值（在配置树中以橙色标记）的本地参数值都会写入相机。
还原本地更改	功能树中任何尚未写入相机的本地更改将被重置。

配置树

配置树包含三个主要组别：

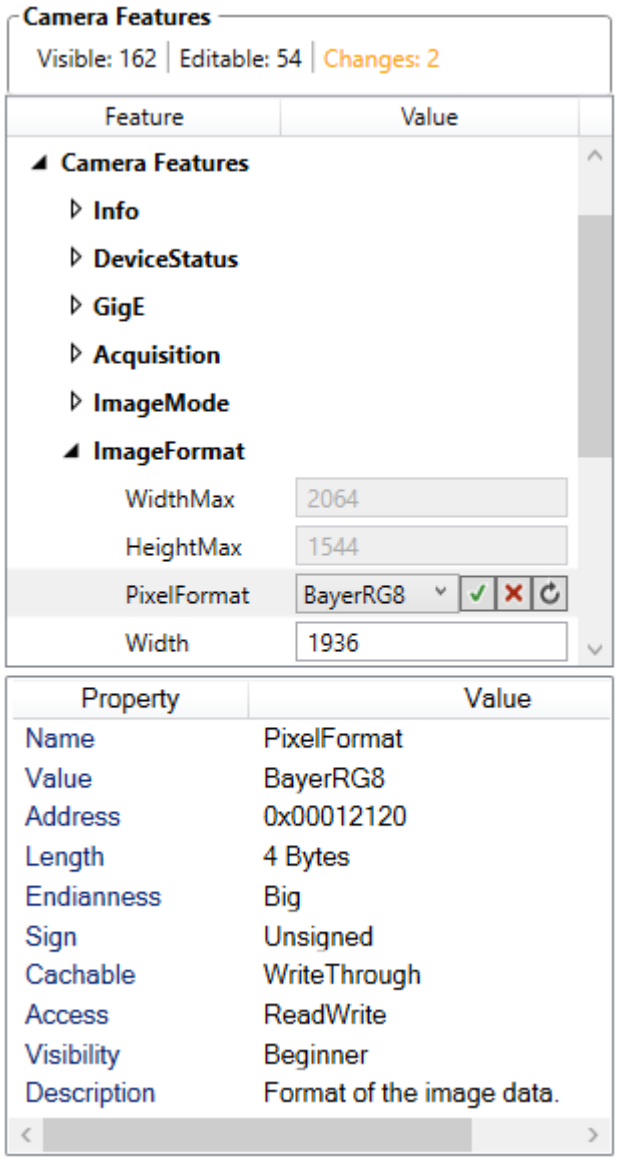


- **TcCOM 参数**：对 TwinCAT 与所连接相机之间的通信行为进行参数设置。有关这些参数的解释，请参见 TcCOM 参数 [▶ 99] 章节。
- **GigE Vision**：根据 GigE Vision 标准得出的相机相关信息，以及 IP 配置选项。根据相机的不同，这些设置也可能在 Camera Features（相机功能）树中提供。
- **Camera Features（相机功能）**：相机参数设置基于 GenICam 标准。所有相机功能的描述都由相机自动从 GenAPI 文件中读取。功能可以是一个可执行命令、一个（可改变）参数或一个组织元件。有关某些设置（如图像采集/触发、合并读出、曝光时间等），请参见 相机配置样本 [▶ 2828]。
  有关特定相机参数的详细信息，请参见相应的供应商文档。此外，GenICam 标准 [▶ 120]还描述了 Camera Feature（相机功能）树的结构方式。

Camera Feature（相机功能）信息

配置表上方显示有关 Camera Feature（相机功能）的三项信息：

- 可见：所有当前可见或显示功能的数量，具体取决于激活的筛选器设置。
- 可编辑：可编辑的“可见”功能数。这个数字一般取决于当前相机的参数设置，因此会有所不同。
- 更改：Camera Feature（相机功能）树中的更改总数。只有存在更改时才会显示该数字，与筛选器设置是否激活无关。



如果点击配置树中的某个功能，该功能的属性会显示在树下方。

参数设置：

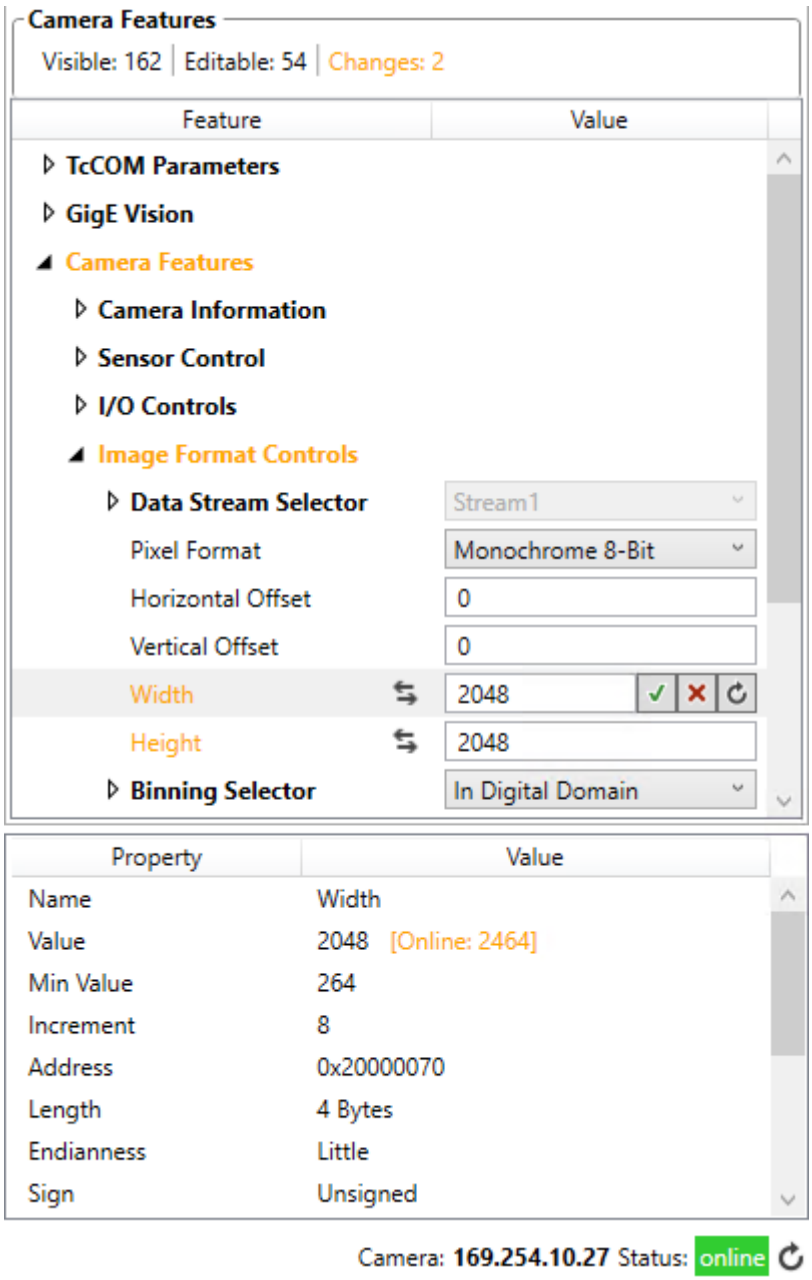
可更改特征可通过文本框、滑块、组合框或复选框进行更改。只读特征显示为灰色。每个可更改配置元件都有以下按钮：

写入值
将本地更改写入相机。

重置值
如果本地值与相机值之间存在差异，则删除本地更改。

加载值
从相机加载参数的当前值。

如果设置的参数值与相机上的实际值有偏差，配置树中的参数和更高级别元件将被标记为橙色。此外，有更改的行会用图标标出，偏差也会显示在属性窗口中。表格上方橙色的“更改”注释一般表示列表中的更改。

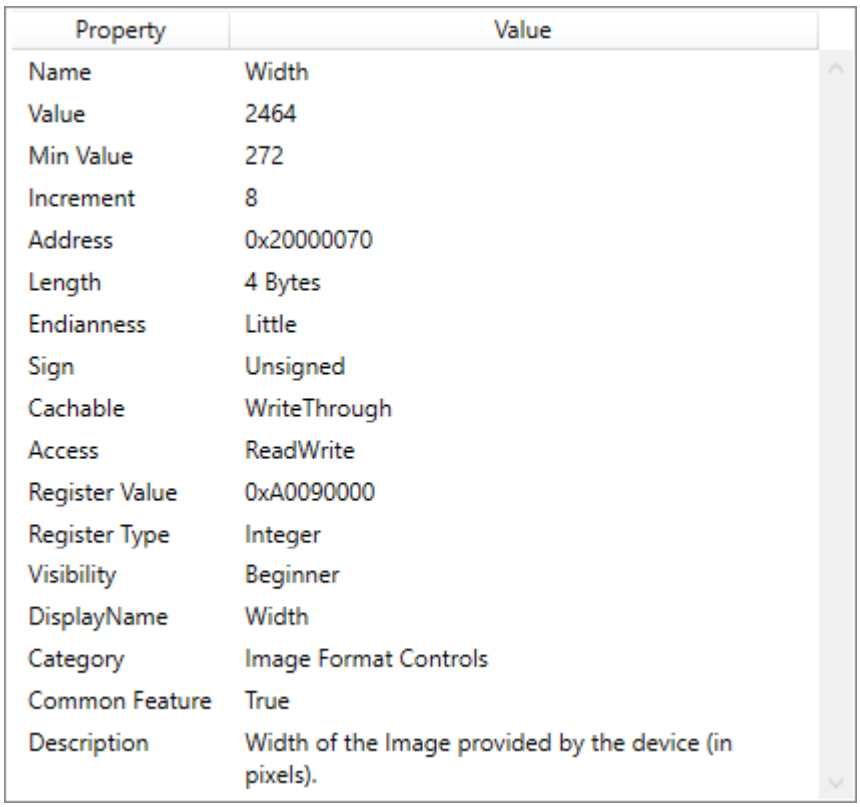


离线更改

如果相机处于脱机状态或目标处于运行模式，也可以更改配置树中的参数。在这种情况下，参数不会直接写入相机，而只会保存在项目中。因此，可以先离线更改参数，之后再与相机同步。

功能属性

单独功能的属性显示在配置树下方。其中包括各相机寄存器的大小和地址、访问权限和说明。



Property	Value
Name	Width
Value	2464
Min Value	272
Increment	8
Address	0x20000070
Length	4 Bytes
Endianness	Little
Sign	Unsigned
Cachable	WriteThrough
Access	ReadWrite
Register Value	0xA0090000
Register Type	Integer
Visibility	Beginner
DisplayName	Width
Category	Image Format Controls
Common Feature	True
Description	Width of the Image provided by the device (in pixels).

属性	描述
名称	参数名称。
倍福工具提示	包含附加信息，仅在特殊情况下显示。例如，“Acquisition（采集）”命令已禁用，因为只需使用用于开始/停止预览的控制器。
值	参数的本地值。如果本地值与相机值有偏差，相机值还会显示为橙色。
最小值	最小设定值。只有在 GenApi 中为功能指定最小值时才会显示。
最大值	最大设定值。只有在 GenApi 中为功能指定最大值时才会显示。
增量	数值变化的增量。只有在 GenApi 中为功能指定最小值时才会显示。
地址	相机上参数的十六进制地址
长度	参数的大小（单位：字节）
字节顺序	参数的字节序列。使用功能块 FB_VN_ReadRegister [▶ 1572] 和 FB_VN_WriteRegister [▶ 1578] 时必须遵守该属性。
签名	• **已签名**：参数已签名 • **未签名**：参数未签名
可缓存	指定未连接相机时是否缓存参数值。 • **写透**：值既写入相机，也写入项目缓存和项目内存。 • **无缓存**：根据相机的 GenAPI 描述，此相机参数的值不应保存在本地。因此，如果没有相机连接，此参数可能为空。
访问	在相机上读写此参数的访问权限。
寄存器值	参数的寄存器值。如果是已知类型，则这可能与显示值有偏差。示例： 名称：PixelFormat 值：BayerRG8 寄存器值：13
位掩码	使用的位掩膜，只显示掩膜寄存器。
寄存器类型	寄存器的数据类型。该类型表示使用哪个读/写功能块。
可见性	参数的可见级别。可能值：**Beginner（初级）**、**Expert（专家）**、**Guru（大师）**和 **All（全部）**

属性	描述
	参数的可见性可以通过 筛选器条目 [▶ 82]进行设置。
显示名称	要显示的功能名称。
类别	对功能进行排序的类别名称。
常用功能	表示该功能是否包含在 Common Feature（常用功能）列表中。
描述	参数的含义

例如，如果要从 PLC 读取或更改参数，就需要这些属性。基本上，必须注意地址、长度、字节顺序和寄存器值。相应的功能块可在 Camera Register Access（相机寄存器访问）下的 API 参考中找到。

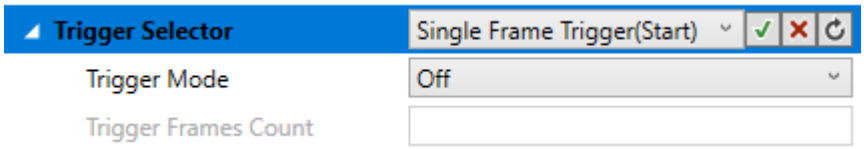
根据寄存器的数据类型，可以添加更多属性，如 Min（最小值）/Max（最大值）/Increment（增量）。

选择器

选择器是用于分组和选择多个子功能的参数。要使用子功能，必须首先将选择器参数设置为适合的参数（在配置助手和 PLC 中）。

如果选择器参数没有地址属性，则不指向相机寄存器。它的作用只是在配置助手中提供更清晰的显示。在这种情况下，可以直接通过 PLC 更改子参数。

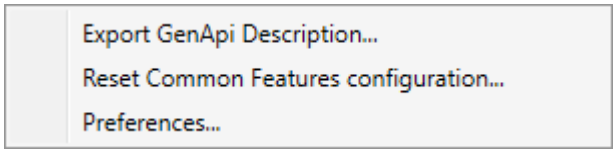
为了在创建初始化命令时考虑选择器的所有子参数，会自动对所有子功能进行迭代。相应的设置载于 操作和设置 [▶ 87]。选择器的一个示例是触发选择器：



5.4.2.3 操作和设置

操作

右上方是操作按钮 ，可用于进一步操作或进行附加设置：

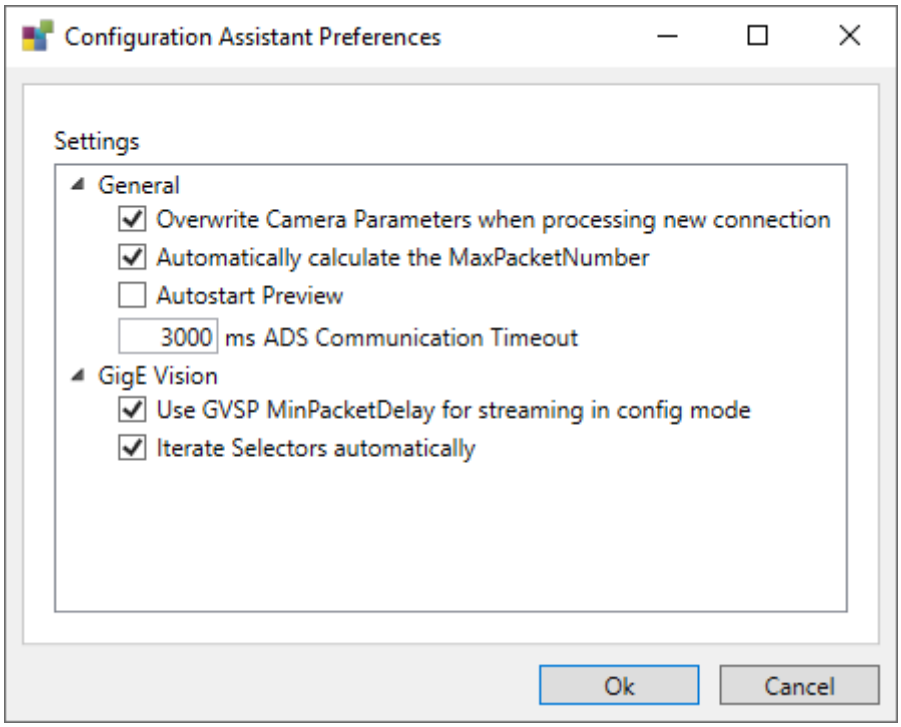


导出 GenApi 描述……	相机的 GenAPI 描述存储在本地指定的路径中。
重置 Common Features（常用功能）配置……	将 Common Features（常用功能）列表重置为预定义状态。
偏好……	下述设置窗口打开。

无法加载已保存 GenAPI 文件

无法加载已保存 GenAPI 文件，因为需要与相机连接才能读取某些相机功能。

设置



加载项目时覆盖相机参数	此设置控制在加载项目时是否应将保存的相机值直接写入相机。这样可以确保相机自动处于配置项目时最后保存的状态。 这只有在加载项目后首次初始化相机对象时才会起作用，前提是当时相机已连接且 TwinCAT 处于配置模式。 默认：已激活
自动计算 MaxPacketNumber	如果激活此选项，则会在读出所有相机参数后自动确定图像所需的（和允许的）最大数据包数。计算基于图像大小和像素格式。只有当计算值大于当前值时，才会将其写入负责的 GVSP 对象。该功能可根据实际图像数据量自动调整为待接收图像预留的空间。 默认：已激活。
自动启动预览	如果激活此选项，则在可能情况下，打开 Configuration Assistant（配置助手） [▶ 77]时会自动显示相机的实时图像。 默认：禁用。
ADS 通信超时	Communication Timeout（通信超时）与 TwinCAT 中相机实例内部的通信有关。只有当 GVCP 或 GVSP 模块中的其他超时也需要增加时才需要对其进行增加。 单位：毫秒 (ms) 默认：3,000（与相机相适应）
在配置模式下为实现流式传输使用 GVSP MinPacketDelay	如果激活该选项，来自 GVSP 模块的 MinPacketDelay 将用于减少 Configuration Mode（配置模式）下的带宽。如果关闭此选项，相机的帧率反而会降低。 默认：已激活。

自动迭代选择器	激活此选项后，作为选择器的相机枚举参数会自动迭代。 选择器参数的值用作寻址其他参数的索引。通过改变一个选择器值，可以相应改变从属参数，从而实现相机参数设置的一致性。 如果没有这种迭代，就只能描述当前所选选择器的参数。 这既适用于配置树 [▶ 81]中的设置，也适用于创建 Initialization Commands（初始化命令） [▶ 89] 的情况。 默认：已激活。

注意

对初始化命令的影响

如果更改了 **Iterate Selectors automatically（自动迭代选择器）**选项，则必须重新创建 Initialization Commands（初始化命令） [▶ 89]。点击 Initialization Commands （初始化命令）选项卡上的 Create（创建）按钮可重新创建初始化命令。

5.4.2.4 初始化命令

初始化命令的目的是为相机保存特定的参数配置，且如果需要，还可以将相机设置为这种配置。这个功能十分必要，因为相机在重启后总是反映出交付状态。初始化可以在启动 TwinCAT 时自动进行，或者也可以通过 PLC 中的方法调用来手动进行。

初始化命令与 EtherCAT 设备的启动列表或用户设置相类似。用户设置是 GenAPI 的一个功能，可以保存和自动加载相机中的配置。用户设置并非 GigE Vision 标准所要求，因此并非所有相机都提供。

为了能够为所有 GigE Vision 相机统一提供这种功能，TwinCAT Vision 通过初始化命令提供了可能性。当然，如果可用，也可以用用户设置作为替代。

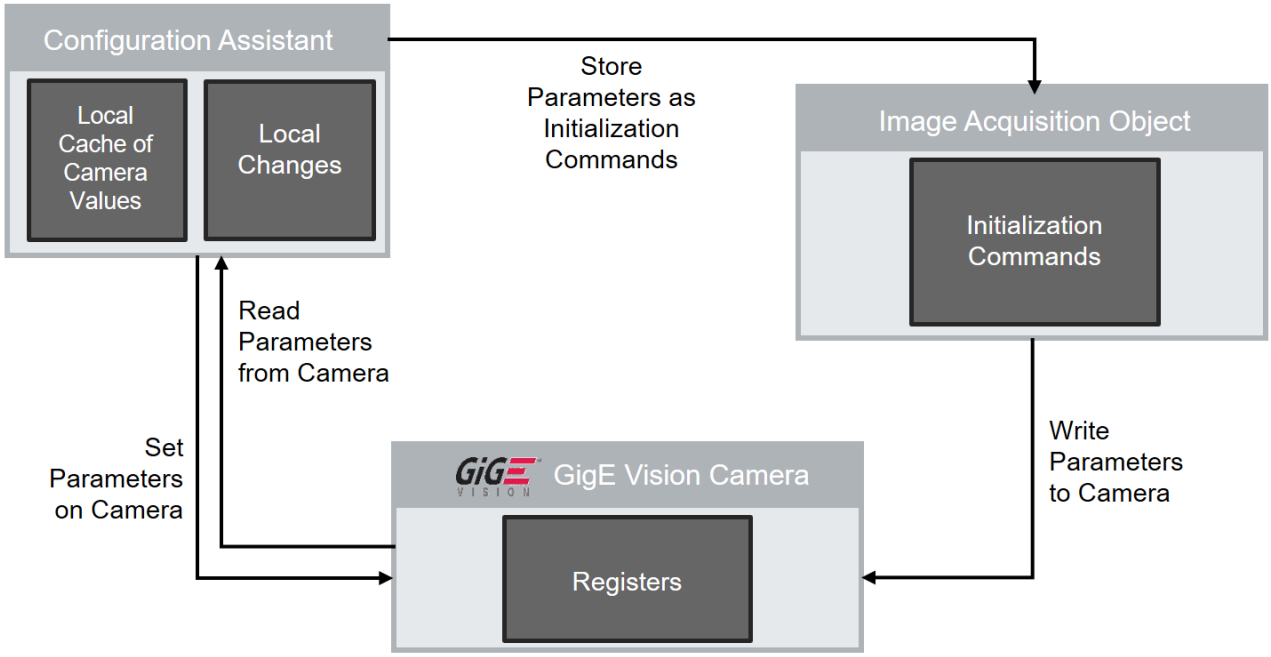
除了这些一般信息外，初始化命令可以按主题划分为以下几个方面：

- 配置 [▶ 90]
- 创建 [▶ 94]
- 故障排除 [▶ 96]

启动过程后的相机状态

在 GigE Vision 相机启动后，它的配置中默认包含出厂设置。为了恢复以前的配置状态，最初必须通过用户设置、初始化命令或在后台由相机助手将相应的参数写入相机。

初始化命令的作用可以描述如下：



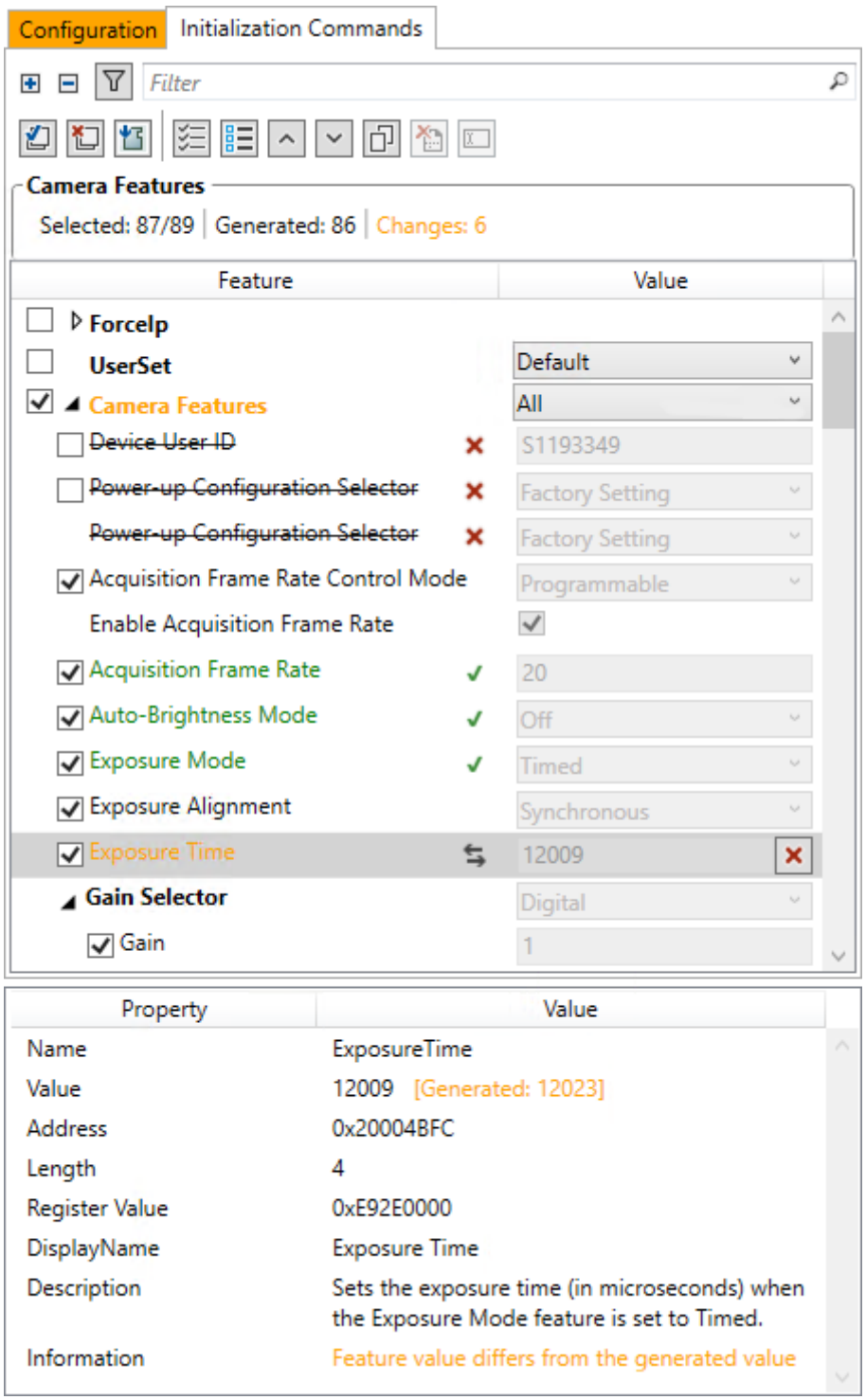
读写操作的触发方式如下：

操作	触发器
从相机中读取参数	• 设置相机后的初始读数 • 重启 TcCOM 对象时读取（例如，当点击**加载设备**时） • 点击**读取相机参数**
向相机写入个别参数	• 在配置树 [▶ 81]中，通过点击**写入值**，对参数进行单独更改和写入。 • 点击**写入本地更改** • 如果在设置 [▶ 87]中**加载项目时覆盖相机参数**激活，在加载 TwinCAT 项目后首次连接相机时
根据当前参数值在相机对象中创建初始化命令	• 在初始化命令选项卡中点击**创建并测试所选的初始化命令**
将初始化命令中的参数写入相机	• 如果图像采集 TcCOM 对象被切换到 OP 状态（启动 TwinCAT），取决于图像采集 [▶ 99]对象中的参数 `InitializationAutoMode`。 • 通过功能块FB_VN_GevCameraControl [▶ 1609]的一个方法调用

5.4.2.4.1 配置

Initialization Commands（初始化命令）选项卡由 3 个区域组成，分别如下所述：

- 筛选器和控制器
- 配置列表
- 属性窗口

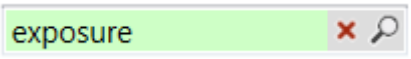


筛选条目



借助筛选选项，可以在配置树和初始化命令列表中快速轻松的找到各功能：

- 点击  符号可展开所有树元素。所有参数均可见。
- 点击  符号可折叠所有树元素。只显示主要组别。
- 点击  图标会对树项目进行筛选，只显示有变化的行。该按钮只有在显示更改时才会启用。

- 搜索行 

会按特定字符串或（十六进制）地址对参数进行筛选。输入字符串时，文本框背景为白色，同时会出现一个用于删除或重置筛选器的附加按钮。如果通过按 Enter（回车键）或点击筛选器按钮确认条目，那么文本框也会以绿色高亮显示。这表示当前字符串是否与激活的筛选器相对应，因此也是树元素的视图。

控制器

左侧一组按钮  可提供基本功能。从左到右的功能依次为：

按钮	描述
创建并测试选定初始化命令	初始化命令使用配置列表的当前值创建。 **参数也会写入相机并回读以供测试。**
删除更改和初始化命令	在 TcCOM 对象中创建和存储的初始化命令将被删除。这也适用于对列表所做的所有其他更改。 **这会将整个列表重置为初始状态。**
应用生成的 Initialization Command（初始化命令）	在 TcCOM 对象中创建和存储的 Initialization Command（初始化命令）会被写入相机。

右侧一组按钮  可提供特殊功能，仅对 Camera Register（相机寄存器）节点下的条目有效。从左到右的功能依次为：

按钮	描述
选择 Camera Register（相机寄存器）类别的所有功能	Camera Register（相机寄存器）类别下的所有条目均会被选中。
取消选中 Camera Register（相机寄存器）类别的所有功能	取消选择 Camera Register（相机寄存器）类别下的所有条目。
向上移动元素	选中元素将上移一个列表位置。
向下移动元素	选中元素将下移一个列表位置。
复制元素	创建所选元素的副本。设定值也会被复制。 **复制条目的值以后不能更改。**
删除复制元素	删除先前复制元素。
重命名复制元素	打开一个窗口，以重命名复制项目。

配置列表

配置列表包含三个主要组别：

- **ForceIP**：如果此选项已激活，则在运行时初始化相机时会执行 ForceIp。使用来自图像采集模块 [▶ 99]的 CameraIPAddress。由于命令会发送到相机相应的 Mac 地址，因此在更换相机时，必须使用更改后的地址重新生成初始化命令。使用的数值仅在此处显示。默认：禁用。
- **用户设置**：某些相机具有用户设置功能，可加载存储在相机中的参数图像，以便对相机进行初始化。如果相机提供此功能，则可在此处选择所需的用户设置。如果该选项已激活，则选择和加载相应用户设置的命令会写入初始化命令。默认：禁用。
- **Camera Features（相机功能）**：在该组中，根据所选滤波器显示寄存器。使用 **All（全部）**功能，可列出在读取 GenApi 文件中声明为可写入并且在当前相机配置中可用的所有寄存器。这与存储的可见性无关，因此也会列出不可见的寄存器，这些寄存器在配置树中只在“All（全部）”可见级别下显示。**Common（常用）**条目显示的是 Common Feature（常用功能）列表中的所有可写寄存器，包括所需选择器。初始顺序与 GenApi 文件中的顺序一致。如果相机功能选项已激活，则在其下面选择的所有寄存器都将用于创建初始化命令。可以选择或取消选择单个寄存器，并通过相应的操作元件进行更改。地址相同且仅掩码不同的寄存器仅用复选框显示，因为它们不能独立写入。对已创建初始化命令的更改会

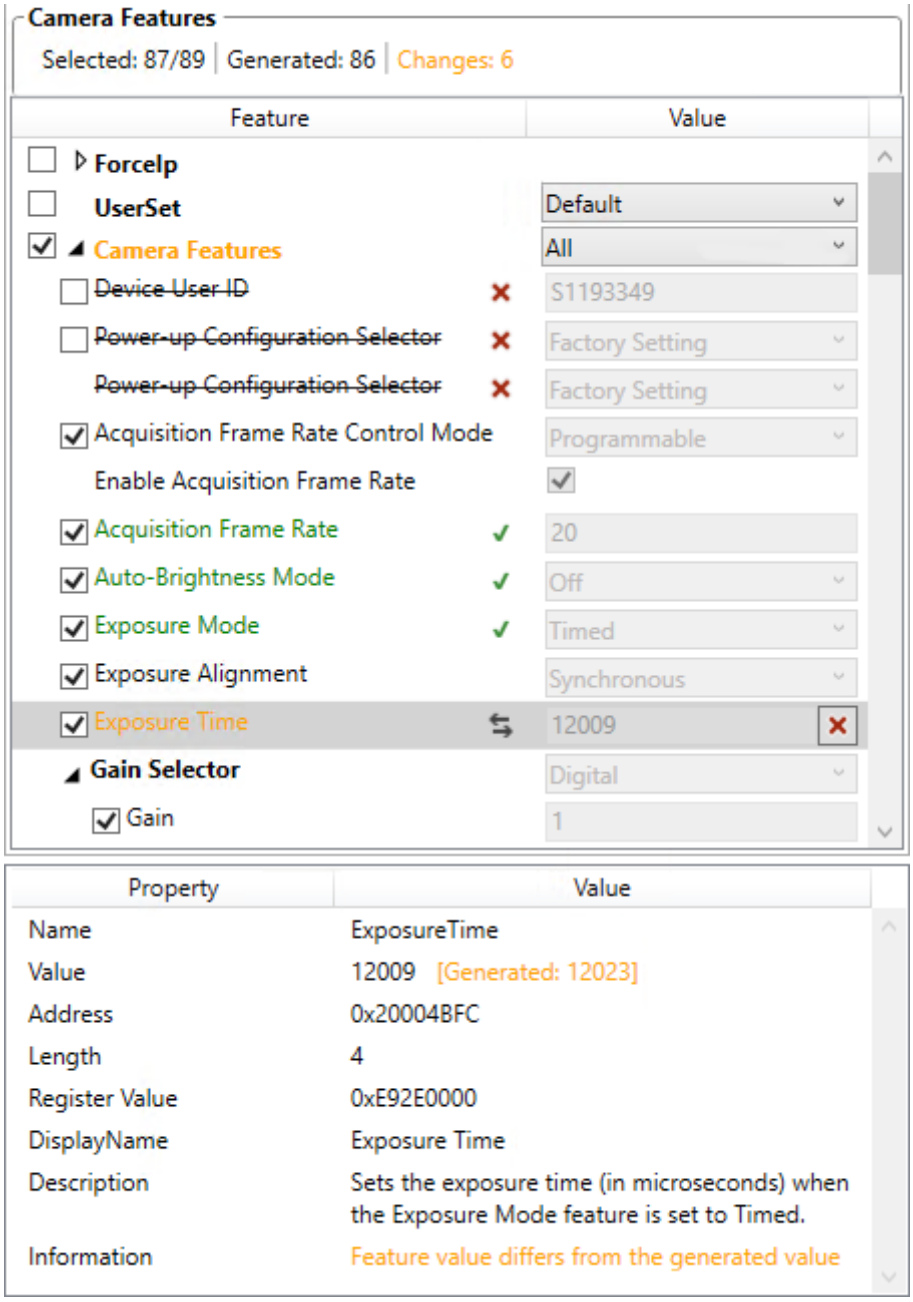
用颜色和图形标识。默认：已激活。**Reset Value（重置值）**按钮可用于将单个值的变化重置为生成值。**Apply generated Initialization Commands（应用生成的初始化命令）**按钮可用于将所有值重置为已生成状态。

当选择或取消选择其中一个组别或更改组别下的设置时，必须重新创建初始化命令。使用用户设置时，初始化命令包含必要的用户设置命令，而非参数。用户设置和相机寄存器选项互斥。

Camera Feature（相机功能）信息

配置列表上方显示有关相机功能的三项信息：

- 已选择：从最大可用功能数中选择的所有当前相机功能数。这通常取决于当前相机的参数设置，因此会有所不同。
- 已生成：项目中存储的最近创建初始化命令的数量。
- 更改：相机功能列表中的更改总数。只有存在更改时才会显示该数字，与筛选器设置是否激活无关。

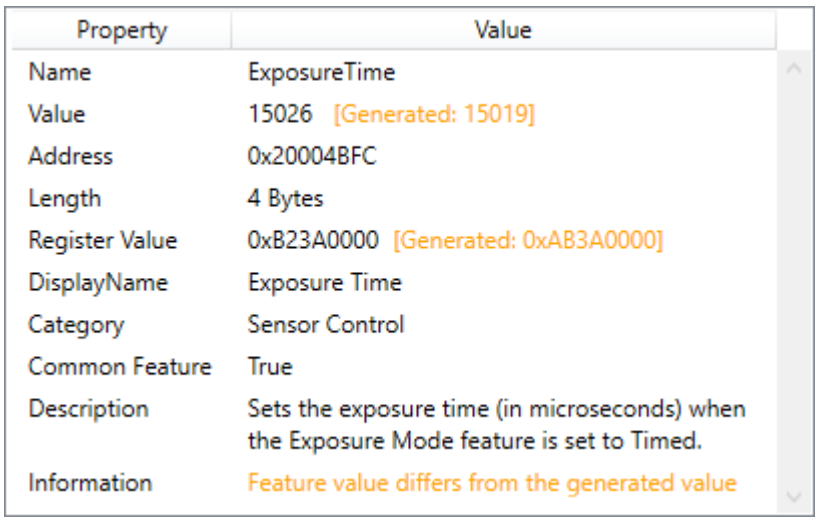


选择器功能

如果在设置 [▶ 87]中激活了**自动迭代选择器**，则“选择器”功能的每个值都有一个单独的组别。在该组别中，会显示所有取决于选择器值且可写入的寄存器。因此，组别和寄存器名称可能出现不止一次，在这种情况下，值或寄存器地址会有所不同。如果设置未启用，则每个选择器只有一个条目。

属性窗口

Properties（属性）窗口可显示所选功能的以下信息：



属性	描述
名称	参数名称。
值	参数的本地值。如果本地值与创建的初始化命令值不同，则该功能显示为橙色。
地址	相机上参数的十六进制地址。
长度	参数的大小（单位：字节）。
寄存器值	参数的寄存器值。
命令包括	写入命令中汇总的功能列表。例如，掩膜寄存器，其识别方法是在一个复选框下可以找到两个或多个条目。
显示名称	功能显示名称。
类别	对功能进行排序的类别名称。
常用功能	表示该功能是否包含在 Common Feature（常用功能）列表中。
描述	参数的含义。
信息	有关值是否不同（橙色），功能已添加（绿色）或删除（黑色，删除线）的信息。

5.4.2.4.2 创建

初始化命令的创建只能通过初始化命令选项卡上的按钮**创建并测试选定的初始化命令**启动。在此之前，应该检查设置或更改，以确保所有需要的寄存器都已选中或者不需要的寄存器已取消选择并且寄存器的值与操作所需的设置相对应。

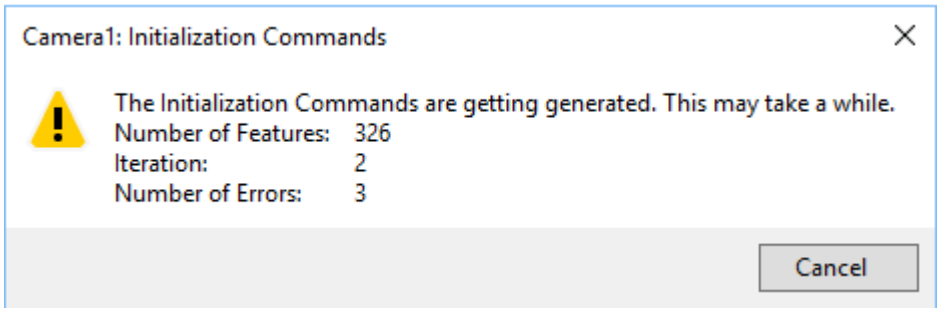
创建步骤

初始化命令的创建包括几个在后台执行的动作。

- 测试向所连接的相机写入初始化命令。
- 测试从所连接的相机中读取先前写入的初始化命令。
- 如果检测到数值差异或错误，会出现一个包含详细信息的列表。如果没有发生关键错误，用户仍然可以接受无错误的初始化命令。

- 将初始化命令数组写入相机的图像采集 TcCOM 对象中。

通过测试写入和随后读取，可以验证相机是否接受初始化命令。在生成过程中会显示以下窗口之一：



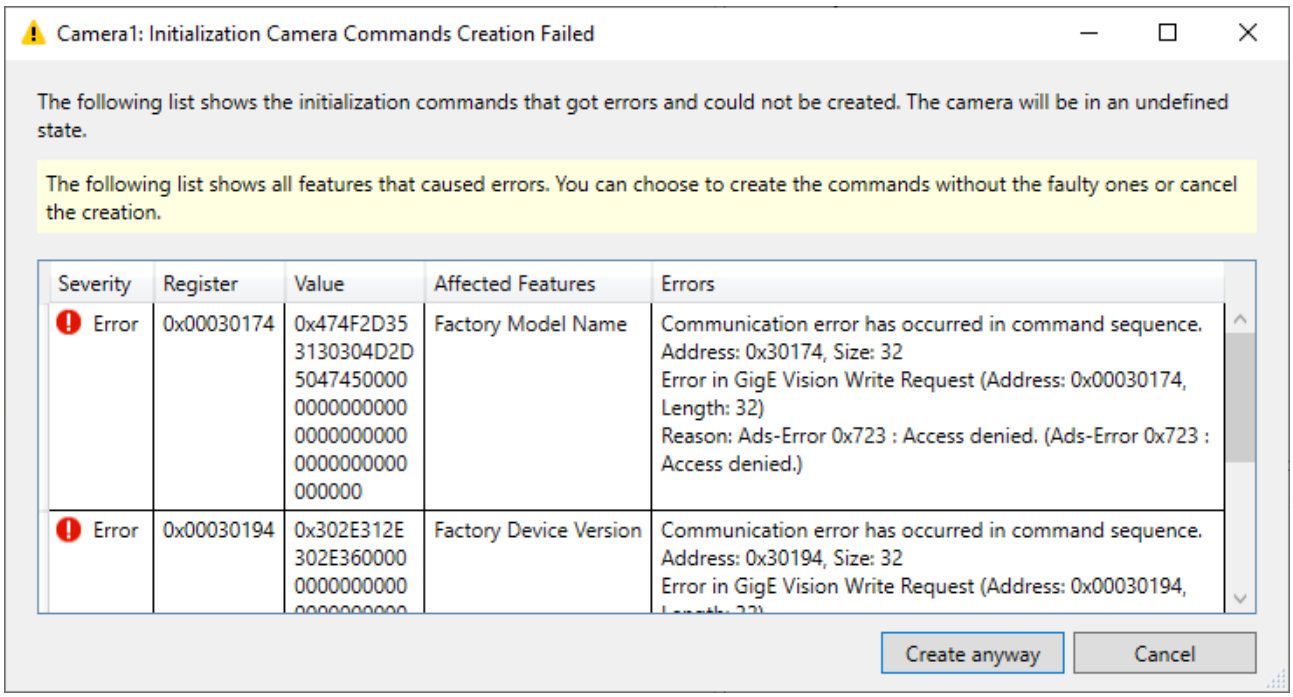
显示以下状态信息：

特征数量	创建初始化命令的参数数量。
迭代	以前尝试向相机写入初始化命令的次数。在成功的情况下，该数值为“1”。如果失败，这个值会增加，同时在没有错误参数的情况下再次尝试将初始化命令成功地写入相机。通常情况下，不应发生超过两次的迭代；然而，如果发生，必须停用图像采集的 TcCOM 参数中的串联。
错误数量	已发生的错误数量。这些在下面的窗口中会有更详细的描述。

如果向相机写入初始化命令的工作非常迅速和成功，该窗口又立即消失且不可见。

消息

在成功创建初始化命令后，相机（由于测试写入）完全具有以后在操作中写入相机的配置。如果在测试向相机写入初始化命令时出现错误，这些错误将显示在以下窗口中：



附图 1: 创建初始化命令时的非关键性错误

只要没有发生重大错误，就可以通过点击**仍然创建**创建并保存初始化命令。在这种情况下，错误的参数将被排除在初始化命令之外，因此以后不会写入相机。

在发生重大错误的情况下，初始化命令无法保存。因此，无法点击**仍然创建**。首先，必须纠正错误的原因。

这些错误情况及其纠正方法如下：故障排除 [▶ 96]。

创建初始化命令时的警告

除了错误之外，还可能出现警告。这些大多只是表明相机没有接受和纠正某个数值。

5.4.2.4.3 故障排除

对于基本相机通信和参数化以及初始化命令创建，主要涉及三个部分：

- GenApi 设备描述文件的内容
- 相机行为
- 以及GevImageAcquisition [▶ 121]对象行为。

GenApi 设备描述文件

GigE Vision [▶ 120]规范包括 GenICam 规范的 GenApi 模块，该模块描述了相机所支持的功能。这种描述采取 XML 设备描述文件的形式，必须按照规范的要求创建。相机提供的 GenApi 描述在收到后会受到检查。如果包含不符合规定的数字，则会报告警告或错误。

- 例如，如果未定义的寄存器位于保留的引导区中，就会发出警告。这些寄存器随后被忽略，但这通常不会导致操作上的限制。在个别情况下，必须检查是否需要所报告的寄存器。
- 例如，如果描述的访问权限与相机的行为不匹配，就会报告错误。也就是说，相机在读取标记为可读的寄存器或写入标记为可写的寄存器时返回一个错误。
- 如果 GenApi 描述包含完全不符合要求的段落，在极少数情况下，相机可能根本无法运行。

相机行为

如前所述，如果相机行为与 GenApi 的描述不同以及如果寄存器包含相互依赖关系或相互影响，相机的行为会产生影响。

- 例如，寄存器可能被其他寄存器锁定，因此这些寄存器必须首先被激活或停用。
- 寄存器的值可能会受到其他寄存器或公式的影响，这会造成限制最大值等情况。
- 当其他寄存器被写入时，相机可以改变寄存器的值。例如，相机可能在写入像素格式时自动重置 ROI。

参考制造商的文件

如果相机行为存在问题，请与制造商的文件进行精确比较。例如，个别读写寄存器的写入可能被相机错误拒绝。

GevImageAcquisition 对象的行为

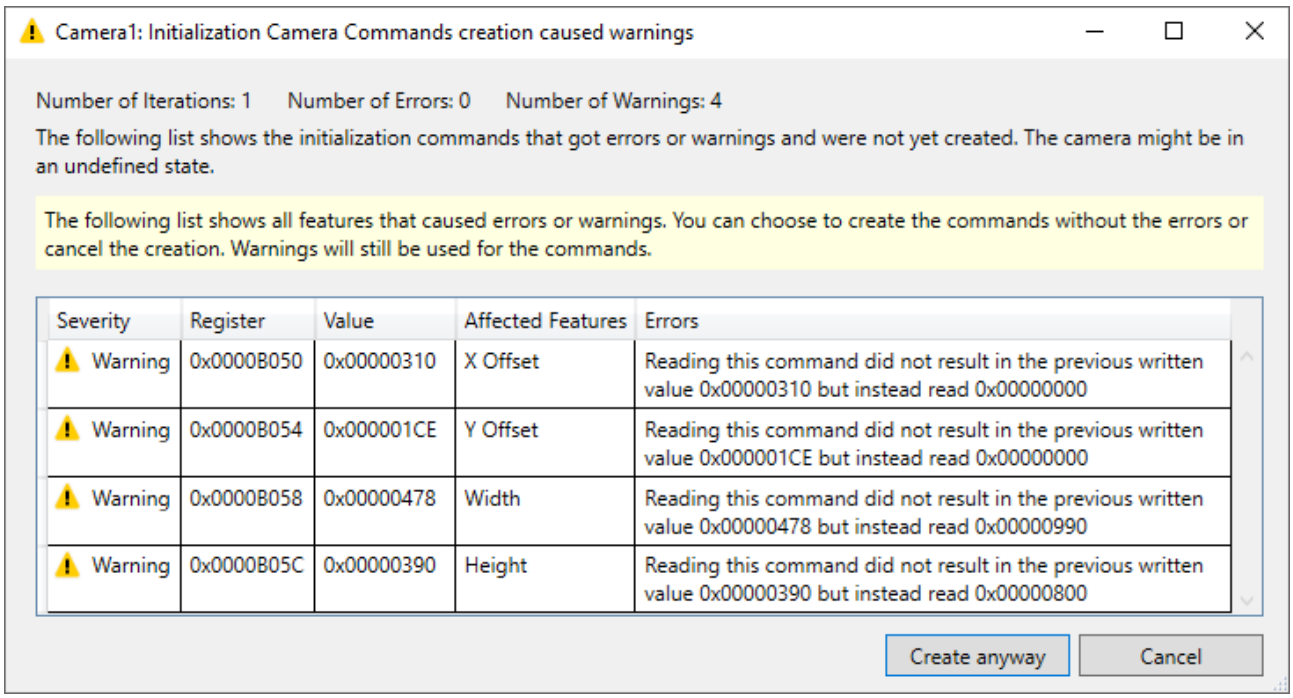
GevImageAcquisition 对象作为 TC3 GigE Vision 连接器产品的一部分通过了 GigE Vision 认证，因此其行为符合标准。串联命令（如果相机支持）和迭代选择器在设置中默认为激活。这意味着，在与相机通信时，所有可能的寄存器组合都被读取或写入。这偶尔会因为命令的数量或时间而导致问题。为了补救这个问题，你可以在图像采集模块中禁用**启用串联命令**或者在设置中禁用**自动迭代选择器**。

消息

上面描述的组件及其行为对初始化命令的创建或可能的消息有不同的影响。

警告

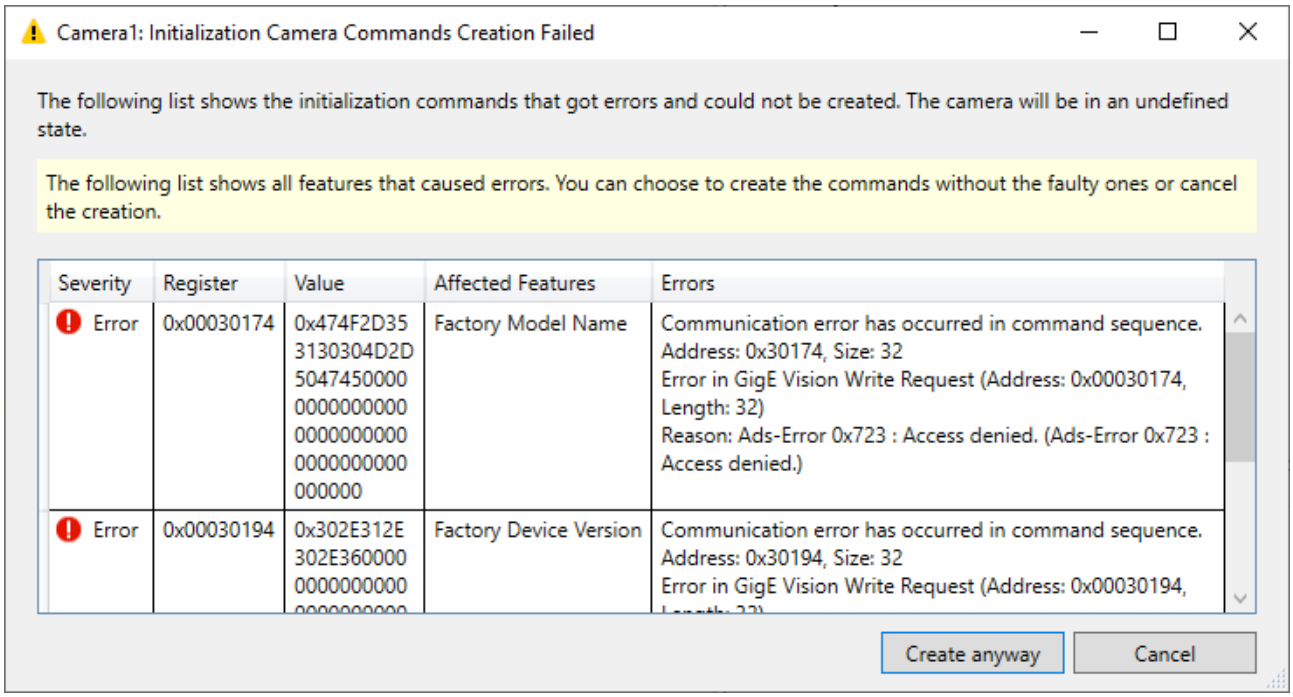
如果读回的数值与之前写入的数值不同，则显示警告。这些差异可能由不同的相机行为或寄存器之间的依赖关系造成。这并不意味着相机不能用初始化命令工作。因此，警告对通过**仍然创建**进行创建没有影响。之后，你应该测试报告的寄存器是否有正确的值以及相机的行为是否符合要求。



一个可能出现警告信息的例子是当设置了一个 ROI 时。根据相机中的当前值，警告会发生，因为根据 GenApi 的顺序，偏移量首先写入。如果需要写入大于 0 的偏移量，宽度或高度必须相应地小于最大值。在这种情况下，相机就会重新设置所有的 ROI 值，造成差异。在这种情况下，可以改变配置列表中 ROI 寄存器的顺序，使宽度和高度在两个偏移量之前写入。在特殊情况下，如果需要，也可以复制一个寄存器，移动顺序，从而多次写入。

错误

如果个别寄存器数值无法写入，则报告错误。如上所述，这可能有各种原因，如极限值、地址、写入权限或写入时公式计算不正确或更改。在通过**仍然创建**进行创建时，受影响的寄存器被省略，并在配置列表中经过核对。随后，应测试被省略的寄存器是否对相机功能有影响。



例如，从截图中可以看出，如果只是型号名称更改，或者寄存器没有更改且设有合适的默认值，那么尽管存在这个错误信息，相机还是可以不受限制地使用。

重大错误

只有在相机在一定时间内没有反应时才会发生重大错误。然后该过程被中止，并出现以下信息之一。在这些情况下，必须调整相应的超时。

超时

强制 IP 或 GVCP 超时

在写入初始化命令的过程中，如果配置了强制 IP 命令，也会发送该命令。如果各相机用此命令设置 IP 的时间很长，可能会发生以下超时错误：

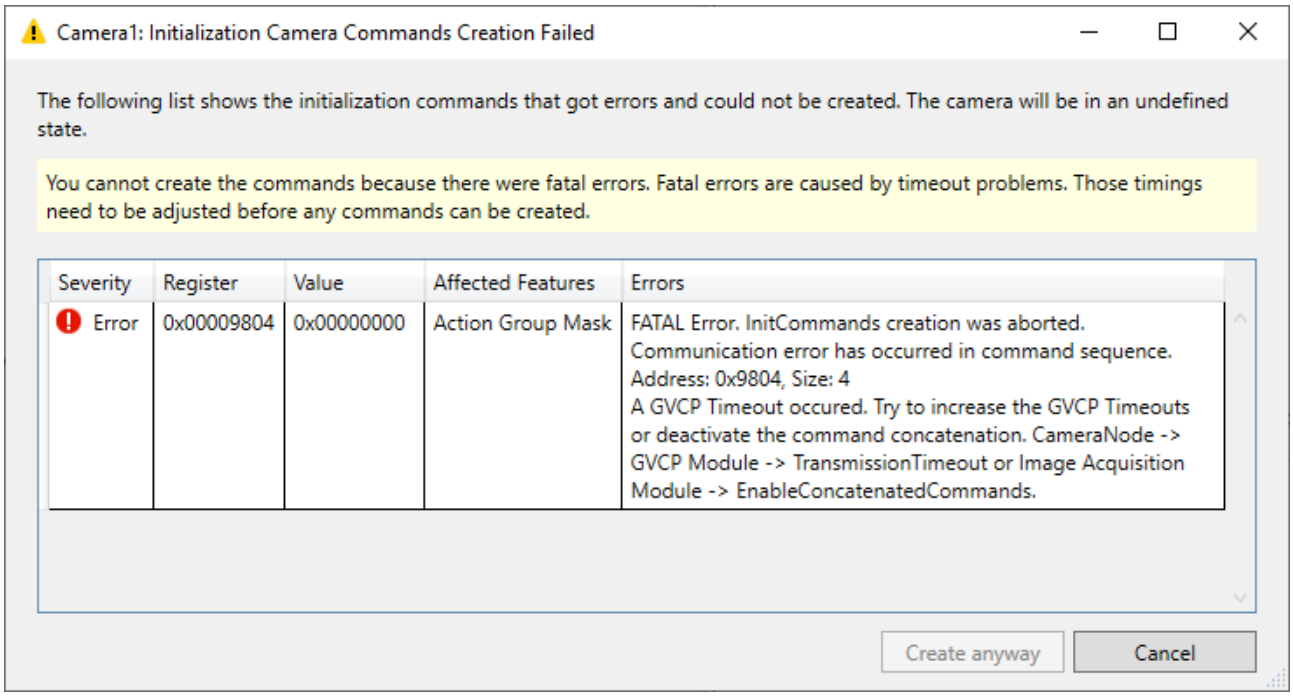


附图 2: 强制 IP 过程中的错误

永久 IP 地址

如果不使用强制 IP，建议设置一个永久 IP 地址。

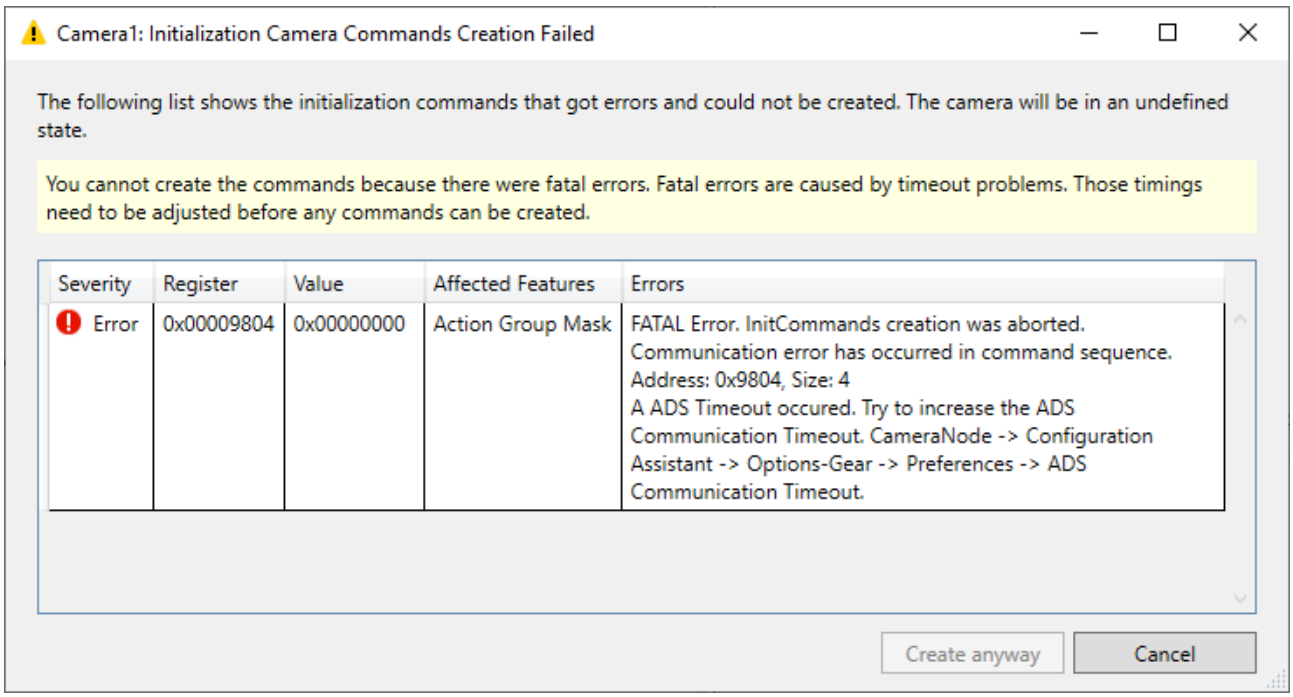
在这种情况下，**强制 IP**可以在配置列表 [▶ 90]中停用。另外，在 GVCP 超时的情况下，`TransmissionTimeout`和/或`MaxTimeouts`可以从GVCP 模块 [▶ 100]在 TcCOM 参数中增加。请注意，设置窗口 [▶ 87]中的`ADS Communication Timeout`应始终至少等于前面两个参数的乘积。



附图 3: 创建初始化命令时出现严重的 GVCP 超时错误

ADS 超时

在 ADS 超时的情况下，在设置窗口 [▶ 87]增加参数`ADS Communication Timeout`。



附图 4: 创建初始化命令时出现严重 ADS 超时错误

5.4.2.5 TcCOM 参数

三个 TcCOM 模块图像采集、GVCP 和 GVSP 的参数描述如下。调整这些参数可以帮助某些相机建立一个稳定的连接。应使用配置树 [▶ 81]以改变参数。

最初，此处仅涉及参数`InitializationAutoMode`。然而，在出现连接或初始化问题时，可能需要设置其余参数。

在参数更改的情况下重新加载

为了使参数更改生效，必须重新加载相应的对象。

图像采集模块

特征	描述
相机 IP 地址	相机的 IP 地址。这个 IP 地址与强制 IP 命令一起使用。如果这个 IP 地址改变，初始化命令 [▶ 89]必须重新创建，以使更改生效。此外，配置 [▶ 90]中的相应选项必须为强制 IP 命令激活。
初始化超时	向相机传输初始化参数的最长时间。根据所使用的相机，传输时间可能更长，在这种情况下，必须对参数进行相应调整。 单位：毫秒（ms） 默认值：2,000
初始化自动模式	选择图像采集 TcCOM 对象自动执行相机初始化的方式。如果相机没有被初始化，就无法与之通信。 **AUTOINIT_SO**：相机在 TcCOM 对象的启动阶段被初始化。在这个阶段，仅有限时间可用。如果使用了初始化参数，由于潜在的超时，初始化通常不能在这个启动阶段进行。在这种情况下，即使选择了 AUTOINIT_SO，也会应用 AUTOINIT_AFTER_SO 的行为。

特征	描述
	AUTOINIT_AFTER_SO：TcCOM 对象在没有初始化相机的情况下进入 OP 状态。只有当从 PLC 打开与相机的连接时（例如用 `FB_VN_SimpleCameraControl.StartAcquisition()` 或用 `FB_VN_GevCameraControl.OpenCamera()`），相机才会自动初始化。 **NO_AUTOINIT**：相机只有在调用 `GevCameraController.InitializeCamera()` 时才会被初始化。该选项旨在针对需要完全控制初始化以便在某个时间执行的情况。 标准：AUTOINIT_AFTER_SO.
启用串联命令	选择在向相机写入多个命令（例如初始化参数）时，是将相关命令链接起来还是单独发送。 命令的链接使传输更快，但可能会导致某些相机出现问题。 默认：激活。

GVCP 模块

特征	描述
本地端口	本地 UDP 端口，GVCP 模块通过该端口进行通信。如果该值为零，则使用自动生成的端口。 默认值：0
最大超时量	在 GVCP 数据包被丢弃之前可能发生的最大传输超时数。每次超时后，GVCP 数据包会被重新发送。如果达到最大的超时数，就会输出一个错误。 默认值：3（适应于相机）
传输超时	在发送一个 GVCP 数据包后，等待响应。如果响应时间长于传输超时时间，则认为数据包没有到达，在这种情况下，数据包会被重新传输。 单位：毫秒（ms） 默认值：600（适应于相机） **传输超时应小于心跳速率。**
心跳频率	应用程序的控制通道向相机发送一个心跳信号，以检查连接是否仍然有效。心跳频率描述了距离下一个信号发送的时间。 单位：毫秒（ms） 默认值：1,000 **心跳速率不应被任意设置得过高，因为许多相机在几秒钟没有心跳后就会触发超时。**
发现超时	发现超时描述了等待发现命令响应所需的时间。在网络中搜索相机 [▶ 71]，就会发送发现命令。 单位：毫秒（ms） 默认值：1,000

GVSP 模块

功能	描述
LocalPort	GVCP 模块通信所通过的本地 UDP 端口。如果该值为零，则使用自动生成的端口。 默认：0
MaxPacketSize	**MaxPacketSize** 描述的是 GVSP 数据包的最大数据包大小。以太网帧头不包括在此大小中。

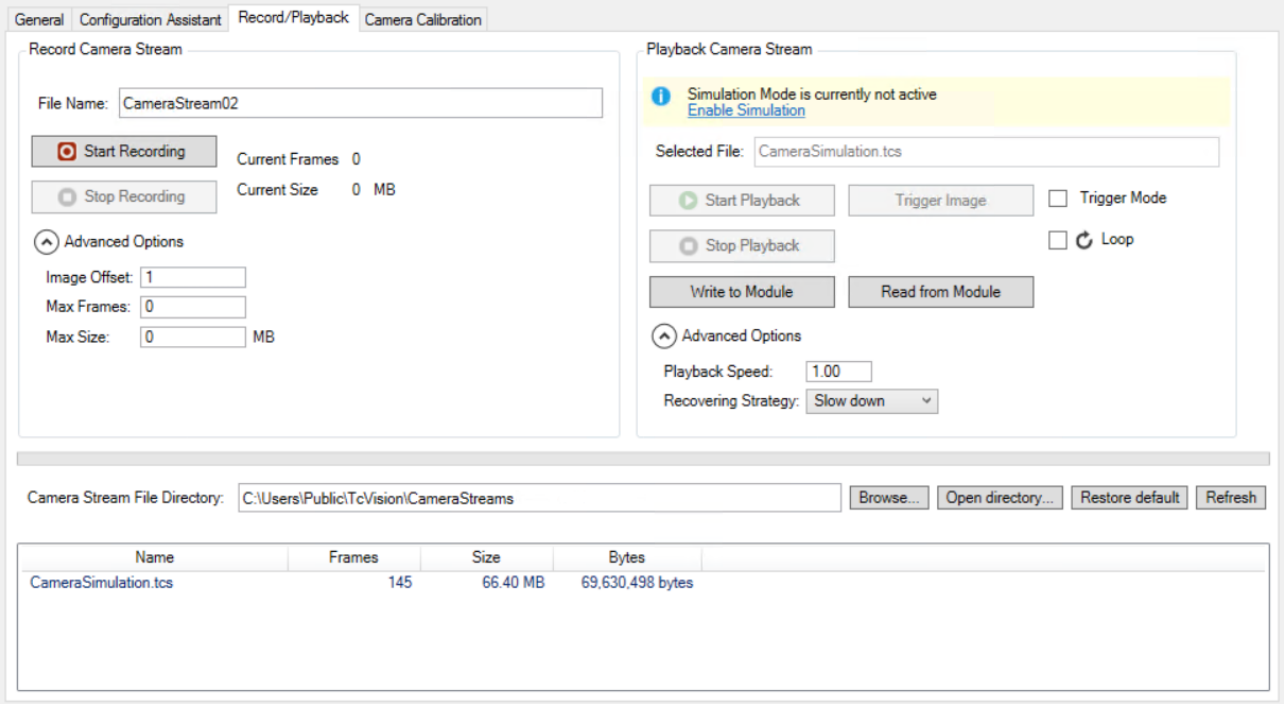
功能	描述
	单位：字节 默认：1,500 **根据安装的 TwinCAT 实时以太网驱动程序和在属性中设置的 Jumbo 数据包大小，最多可使用 9000 字节。**
MinPacketDelay	如果增加 **MinPacketDelay**，那么 GVSP 数据包就会人为延迟。这样就可以减少带宽。 单位：时间戳计数器单位 默认：0
MaxResendRequests	单个 GVSP 数据包出现故障或耗时过长时的最大重发请求次数。 图像在由多个数据包组成的 GVSP 块中传输。如果某个数据块的数据包无法成功传输，则图像无效。 默认：1
MaxBlockNumber	可同时处理的最大 GVSP 块数。如果值为零，则数字不受限制。 默认：5
MaxPacketNumber	一个数据块中 GVSP 数据包的最大数量。包含更多数据包的数据块不能被解释为有效图像。 默认：25,000 **如果在 Preferences（首选项）设置中激活“Automatically calculate the MaxPacketNumber（自动计算 MaxPacketNumber）”选项，则会自动计算该数字。**
BlockTimeout	**BlockTimeout** 描述了一个数据块中单个数据包之间可能间隔的时间。 单位：微秒 (µs) 默认：5,000

5.4.3 记录/回放

在**记录/回放**选项卡上可以记录相机流，并输入到 PLC。这对开发很有帮助，以便在桌面上记录相机流，并在以后的时间里反复进行离线回放。因此，相机硬件的真实行为通过原始图像进行模拟。

要么激活模拟模式，在这种情况下可以回放相机流，要么停用模拟模式，在这种情况下可以记录相机流。不可能同时进行这两项工作。

相机流由单个图像组成，这些图像与相应的时间标记一起以自己的格式（.tcs）保存。通过额外保存的时间标记，机器上的相机的时间行为可以模拟出应用的真实性。图像数据正是以相机发送的格式保存的，不需要进一步转换。



相机要求

相机必须支持时间戳并在图像标题中输入有效值。每个图像都需要时间戳，以便能够模拟时间行为。

系统要求

使用快速的固态硬盘和至少 8GB 的内存来记录和回放相机流。如果使用较慢的存储介质，必要时可通过参数**图像偏移**降低图像速率。

禁用/激活 TcCOM 对象

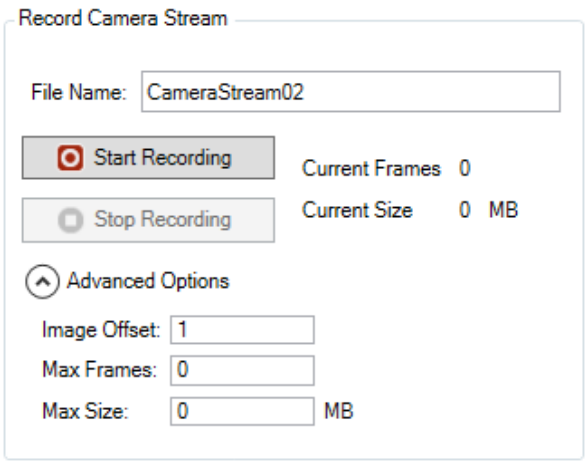
相机 TcCOM 对象的激活/停用不应手动进行，因为在后台从模拟模式和 Ads 通信更改时，必须调整更多链接。

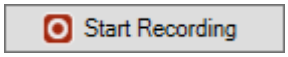
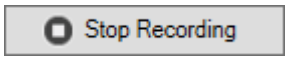
记录/回放助手的三个区域（记录、回放、流文件）解释如下。

记录相机流：

相机流的记录只有在相机连接激活的情况下才能进行。因此，相机实例 [▶ 71]的模拟模式必须禁用，并选择标准采集。另外，还需要 Ads 通信器 TcCOM 模块。

记录相机流的设置位于记录相机流区域：



文件名称	相机流文件名的文本框。用于保存文件的路径是在选择文件路径和流 [▶ 105]中定义。
	用于开始记录的按钮。只有在可以记录的情况下才激活。
	用于停止记录的按钮。仅在记录期间激活。
当前帧数	当前记录的相机流中的图像数量。
当前尺寸	当前记录的相机流的文件大小。
高级选项	显示和隐藏高级选项的选项。
图像偏移	指定图像记录顺序。根据存储介质的写入速度、图像大小和每秒帧数的不同，可能需要跳过图像。 默认值：1（即尝试记录每张图像）。
最大帧数	待记录相机流的最大允许图像数。一旦达到这个数量的图像，记录就会自动结束。 默认：0（即没有限制）。
最大尺寸	待记录相机流的最大允许文件大小，单位：MB。一旦达到这个文件大小，就会自动停止记录。 默认：0（即没有限制）。

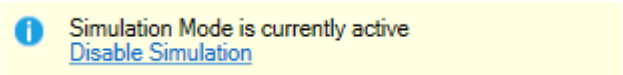
是否存在足够的自由存储空间?

在记录数据流之前，确保你的目标系统有足够的存储空间，或者必要时通过**最大帧数**和**最大尺寸**限制数据流大小。

必须满足以下条件，才能记录数据流：

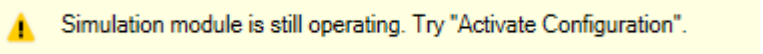
- 模拟模式已禁用。
- Ads 通信对象存在并已激活。
- TwinCAT 处于运行模式。
- 相机必须处于ACQUIRING状态。

如果不符合这些条件，可能会出现以下说明和建议解决方案：

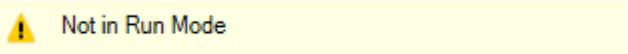


- 模拟模式是指可以回放数据流但不记录的一种状态。该状态可以通过注释中的**禁用模拟**链接进行更改。

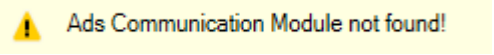
  这也需要重新激活配置  。

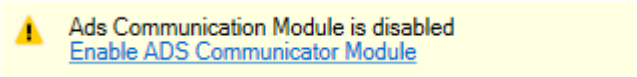


- 模拟模式已被关闭，但该更改只有在配置被激活后才会应用。

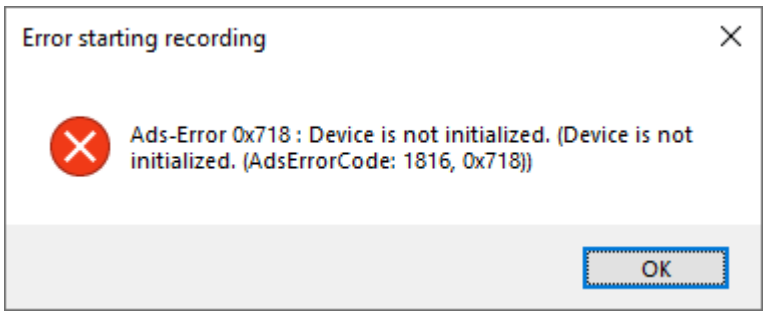


- 只有在 TwinCAT 运行模式下，才能对相机流进行记录。在所有其他 TwinCAT 状态下都会显示该警告。





- 只有在与相机有开放连接的情况下，才有可能记录相机流，因为在记录之前会从相机中读取数据。因此，相机此时应该已经处于ACQUIRING状态。



- Ads 通信模块是记录相机流所必需的。它可以通过说明中的链接激活。如果尚不存在 ADS 通信器对象，可以通过相机对象上的**添加新项目...**和常规选项卡 [▶ 73]中的**尝试修复**进行添加。激活配置后，该更改被接受 。

提前激活 ADS 通信器模块

如果计划以后在不首先停止 TwinCAT 的情况下记录相机流（例如在机器运行期间），必须提前激活 ADS 通信器模块。

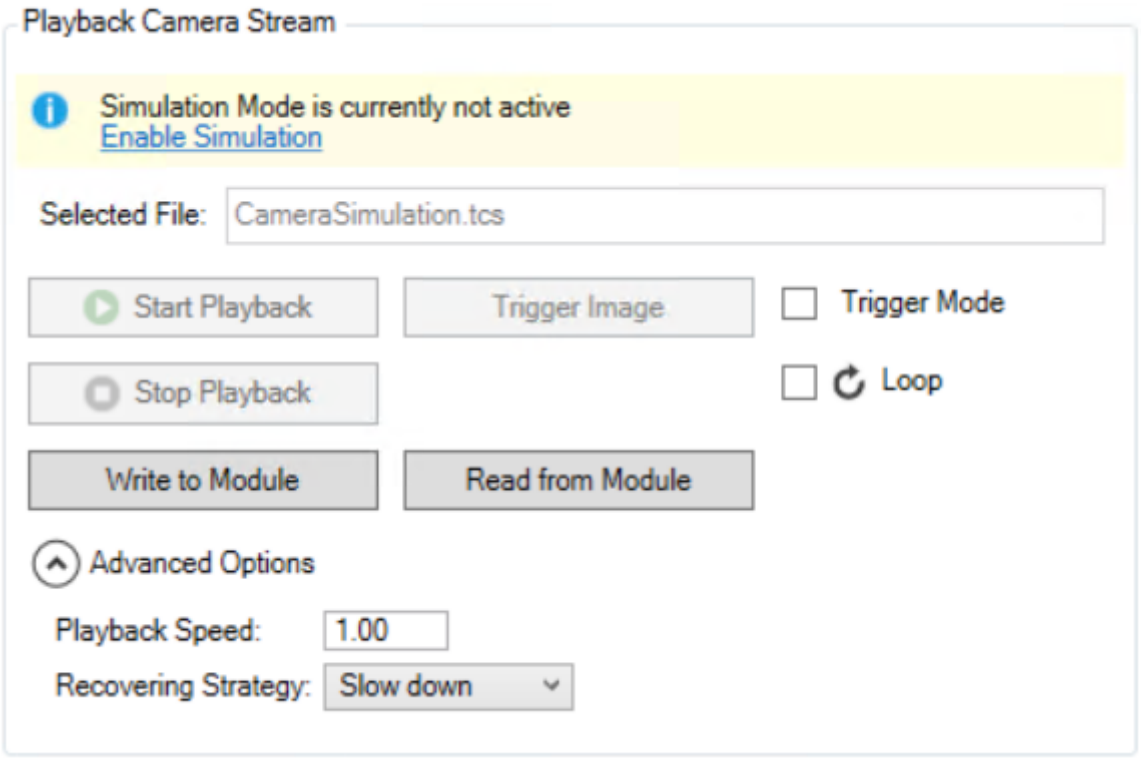
如果 Ads 通信器对象已激活，可能来自相机的图像在短时间内仍被对象内部引用，且因此不能直接转化为可显示的图像。关于更多详细信息，请参见可显示的图像 [▶ 145]。

回放相机流：

为了回放相机流，必须激活相机的模拟模式；这可以通过相机对象的常规 [▶ 73]选项卡或下面提到的信息来实现。

如果通过写入模块将配置写入 TcCom 对象并保存在项目中，那么这些数值在项目激活后就已经可用，且模拟会随着 PLC 的开始采集命令直接开始。这样做的好处是，不必每次都手动重置设置和启动数据流。

回放相机流的设置位于**回放相机流**区域：



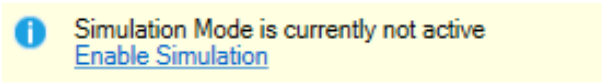
选定的文件	用于显示所选数据流文件的文本框。从可用数据流的列表中进行选择。
开始回放	用于开始回放的按钮。只有在可以回放的情况下才启用。回放总是从图像 1 开始，无论数据流之前在哪里停止。
停止回放	用于停止回放的按钮。只在回放数据流时启用。

触发模式	选择触发模式，在这种模式下，数据流中的帧可以被单独触发。
触发图像	用于触发图像的按钮。仅在选择了触发模式且回放数据流时有效。
循环	选择数据流回放一次或无限循环的选项。
写入模块	将当前设置写入 TcCom 模块。存储内容有回放模式、回放速度、回放策略、循环播放和所选数据流文件的路径。
从模块中读取	从模块中读取存储的值并设置这些设置。
高级选项	显示/隐藏高级设置。
回放速度	回放速度的加速/减速系数。（>1：加速；<1：减速） 默认值：1.00 = 速度与记录相符
恢复策略	在无法以指定速度回放的情况下的策略： • 放慢速度：回放速度延迟（默认设置） • 跳过图像：跳过图像 • 滞后时中止：终止回放（不建议）

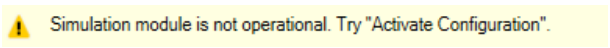
必须满足以下条件，才能回放数据流：

- 模拟模式启用。
- TwinCAT 处于运行模式。

以下说明和警告可能会出现在选定文件名的文本字段上方：



- 模拟模式是数据流可以回放的状态。这个状态可以通过**启用模拟**链接和随后激活配置  达到。



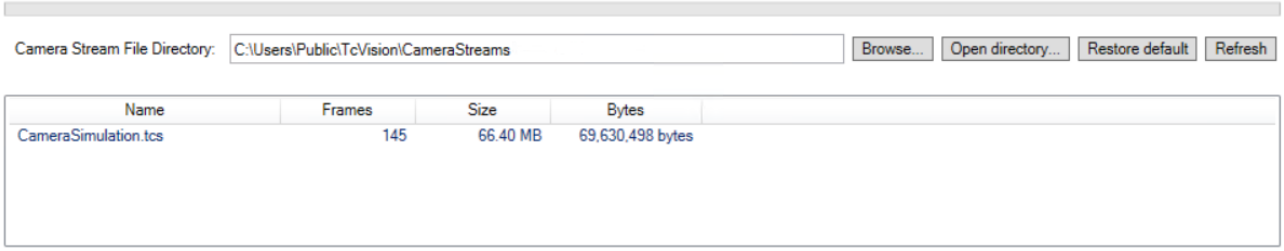
- 模拟模式已启用，但更改仅在配置启用后生效。



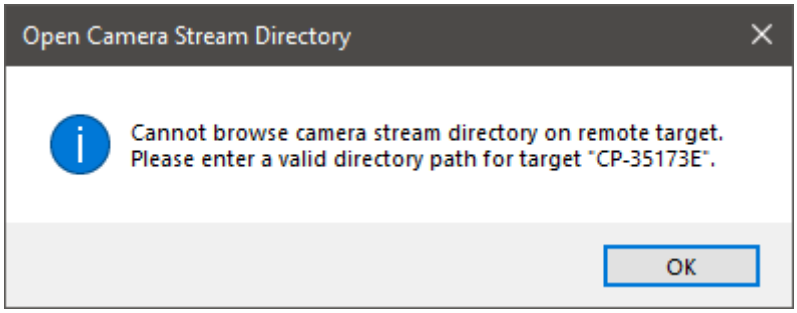
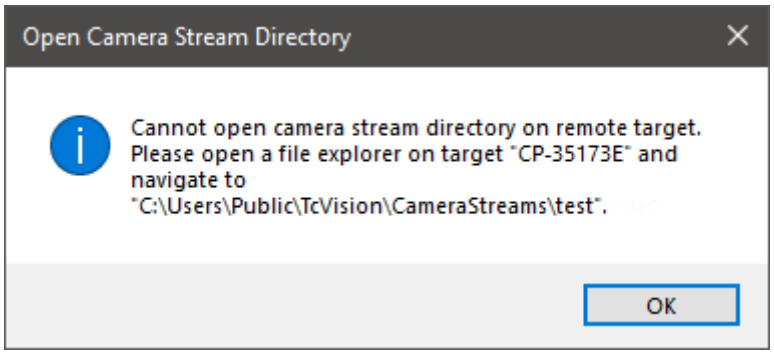
- 仅在 TwinCAT 运行模式下才可以回放相机流。在所有其他 TwinCAT 状态下都会显示该警告。

选择文件路径和数据流

文件路径和数据流的选择如下所示：

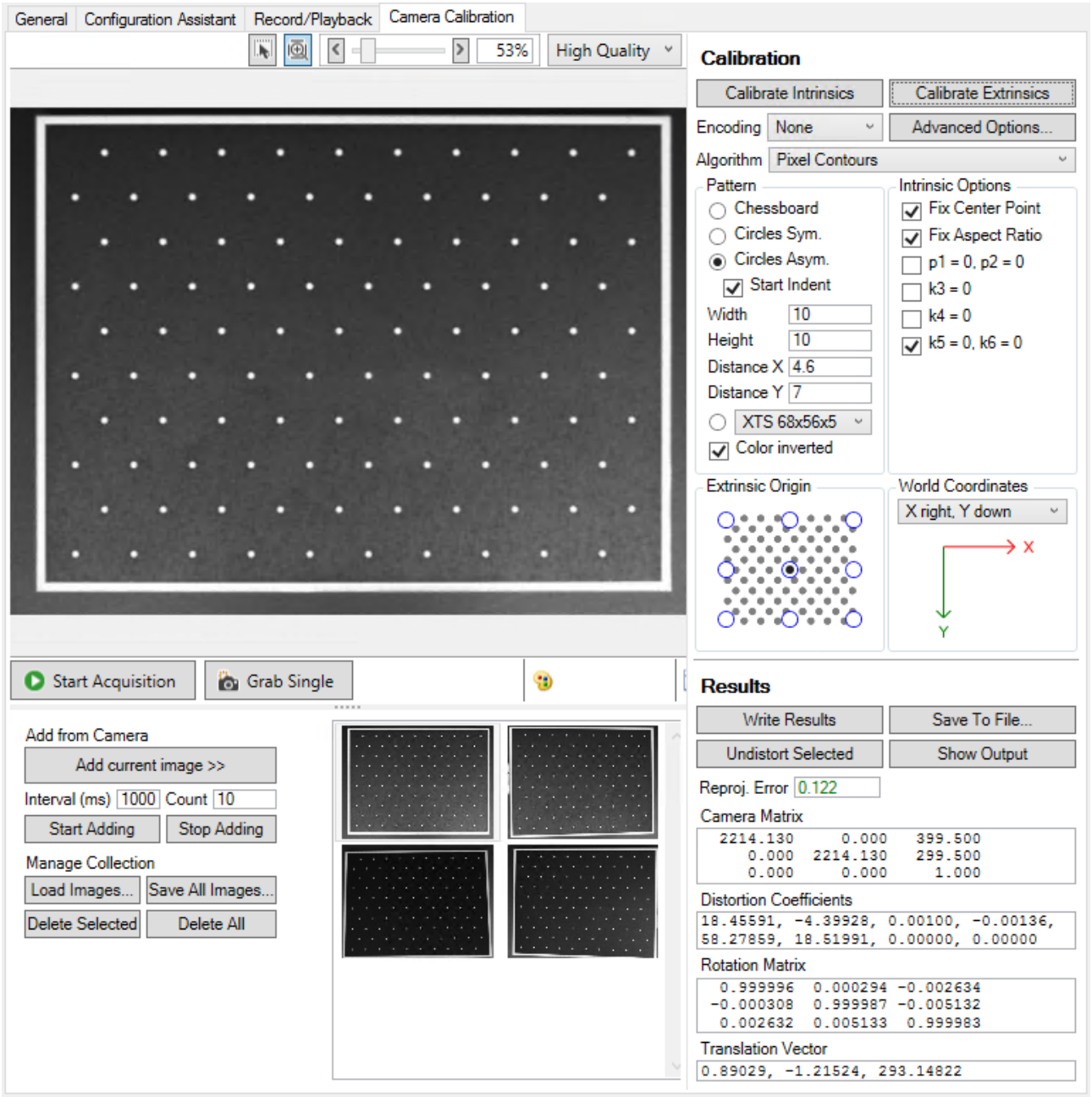


	录制或回放的状态栏
相机流文件目录	目前设置的文件路径。这个路径既用于读取已经记录的数据流，也用于写入新的数据流。子文件夹也会被搜索数据流文件。 **文件路径始终指的是目标系统，而不是开发系统。** **确保这个文件路径不受写入过滤器或类似元素的影响。** 见重要路径 [▶ 2841]。
浏览...	该按钮可打开选择窗口，用于浏览目标文件夹。

	仅在开发系统也是目标系统时，才有可能。 否则会出现以下信息： 
打开目录...	用于在本地文件资源管理器中打开指定文件路径的按钮。这也可以通过在文件列表中的自由区域内双击来完成。 **仅在开发系统也是目标系统时，才有可能。** 否则会出现以下信息： 
恢复默认	用于设置默认文件路径的按钮。
刷新	用于刷新数据流列表的按钮。
名称	所选文件路径中所有相机流的列表，也用于回放时的数据流选择。
帧	各个相机流所包含的图像数量。
大小	各个相机流的文件大小，单位：MB。
字节	大小规格（单位：字节）。

5.4.4 相机校准

区域扫描相机可以在相机校准选项卡上校准模式的帮助下进行几何校准。



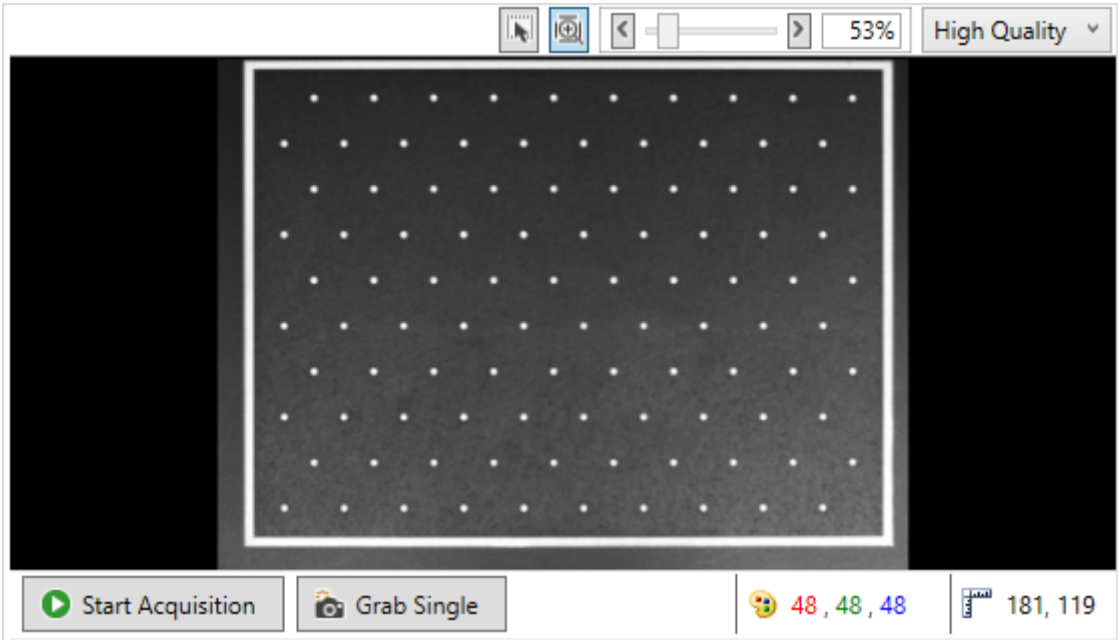
在区域扫描相机的几何相机校准过程中，成像过程由图像重建。为此，图像中的点需要在现实世界中的位置是已知的。具有定义点的校准模式可用，可从中提取图像。

支持的校准模式：

- 棋盘图案 [▶ 114]
- 对称圆形图案 [▶ 116]
- 不对称的圆圈图案 [▶ 116]
- 单个圆圈图案 [▶ 118]

图像采集/选择

如同在配置助手 [▶ 77]中一样，配有一个图像采集的预览图像，上面设有显示选项，下面设有图像采集的控制按钮：



显示预览图像

图像总是按照从相机接收的方式显示。因此，Bayer 格式的图像将被显示为未编码的灰度图像。在这一点上，设置编码还没有应用到图像上。编码只与校准过程有关，但与预览图像无关。

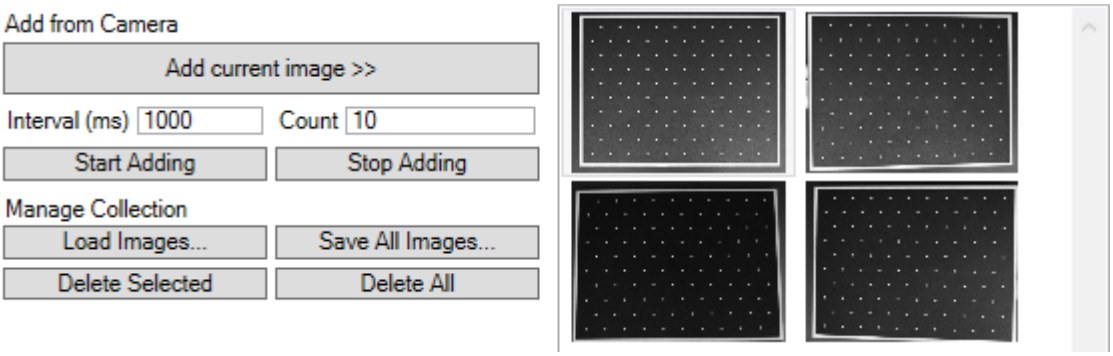
在预览窗口上方：

- **预览图像的插值：**
  组合框，用于选择是否要对预览图像进行高质量或最近邻的插值。该选择仅指预览。校准基于原始图像数据。
- **预览图像的大小**
  如果选择了**适应窗口**，图像会被缩放到预览窗口的大小。另外，也可以通过滑块指定显示尺寸。

在预览窗口下面：

- **开始采集/停止采集**
  按钮用于开始或停止图像采集。相机的数据流模式必须处于激活状态。
- **抓取单图**
  按钮，用于触发单个图像。
- **像素值**
  显示图像上鼠标光标位置的像素的颜色值。
- **光标位置**
  显示鼠标光标在图像上的位置。请注意，图像的零点总是在左上方。

如果校准模式在相机的视场内，可以选择相应的图像进行校准。控制按钮位于左边的预览图像下方，而校准图像集合位于右边。



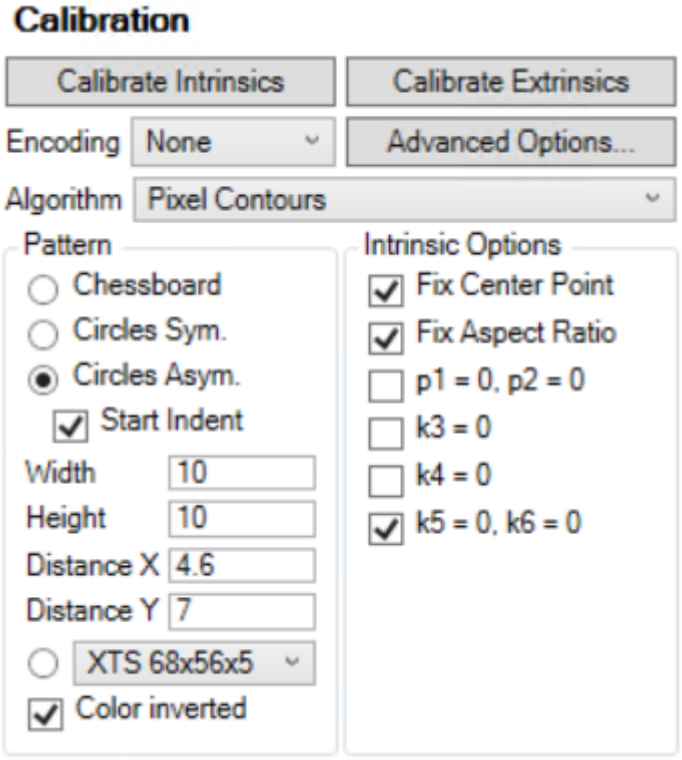
- **添加当前图像 >>**
  添加当前图像将当前显示的相机图像添加到集合中进行校准。

- **开始/停止添加**
  一系列图像可以被添加到集合，而不是单个图像。图像系列通过**开始添加**启动。随后，按照**间隔**设定的时间间隔添加图像，直到达到**计数**输入的数字或通过**停止添加**终止图像的添加。
- **加载图像...**
  作为图像采集的替代方法，可以通过**加载图像**加载保存的图像。请注意，这些都是通过相同的相机系统、相同的相机设置、相同的镜头设置和相同的校准模式记录。
- **保存所有图片...**
  保存所有图片可以用来保存整个图片集。
- **删除所选**
  删除所选将所选的图像从集合中删除。
- **删除所有**
  删除所有将所有图像从集合中删除。

图片格式

目前，仅可使用 Mono8、RGB24 和 Bayer8 图像。

设置和计算



编码

对于测量任务，建议使用单色相机。然而，如果需要通过 Bayer 模式校准彩色相机，必须通过**编码**指定相应的 Bayer 模式。这在默认情况下自动选择。

算法

对于圆形图案检测，可以选择不同的边缘定位功能。根据图像质量，可以使用子像素功能来改善结果。然而，这也意味着整个过程需要更多时间。关于子像素功能的更多信息，请参见测量一章。

- **像素轮廓**
  基于轮廓
- **子像素插值**
  用子像素进行插值
- **子像素误差近似**
  误差近似后的子像素

标定模板的定义

在**模板**下选择和定义使用的标定模版：

- **棋盘**
  使用棋盘模板
- **对称圆圈**
  使用对称圆圈模板
- **不对称圆圈**
  使用不对称圆圈模板
  - **开始缩进**
    用于指定圆圈模板首行缩进的复选框。在没有复选标记的标准系统中，预计不缩进行。
- **宽度**
  水平方向的特征点数量
- **高度**
  垂直方向的特征点数量
- **距离 X**
  水平方向特征点在世界坐标系中的距离
- **距离 Y**
  垂直方向特征点在世界坐标系中的距离
- **单个圆圈模板的下拉菜单**
  选择单个圆圈模板（只有在指定文件夹中找到相应文件时才会显示条目）
- **颜色反转**
  用于选择图像颜色反转的复选框。默认情况下（无复选标记），白色背景上显示黑色对象。

对称模板

对于完全对称的模板，找到点的分配可能与定义点有偏差，或者坐标原点可能位于不同的象限。如果所选图像的模板定向与定义不同，也会出现这种情况。

因此，在完成标定程序后，应检查坐标系是否与您的规格相对应。

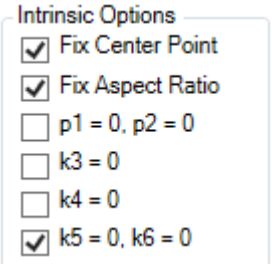
检查填充圆圈

如果是圆圈模板，还要检查圆圈是否填充了相应的颜色。如果在这一点上出现偏差，相应点就不会被识别为模板的一部分。

有关模板和参数设置的信息，请参见支持的标定模板类型：

- 棋盘图案 [▶ 114]
- 对称圆形图案 [▶ 116]
- 不对称的圆圈图案 [▶ 116]
- 单个圆圈图案 [▶ 118]

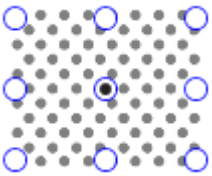
内在选项



- **固定中心点：**主点对应于图像的中心。
- **固定长宽比：**假设相机传感器有方形像素，因此像素的长宽比在 x 和 y 方向上相同。
- **p1=0，p2=0：**假定不存在切向畸变，相应的系数设置为 0。
- **k3=0：**径向畸变系数 k_3假定为 0

- **k4=0**：径向畸变系数 k_4假定为 0
- **k5=0, k6=0**：径向畸变系数 k_5和 k_6 假定为 0

外在原点



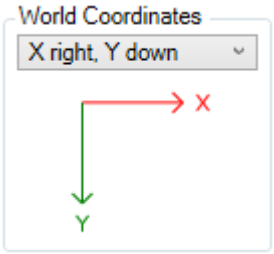
外在原点指定校准模式中全局坐标系的原点。

根据所选择的校准模式，会显示相应的背景图像。

原点

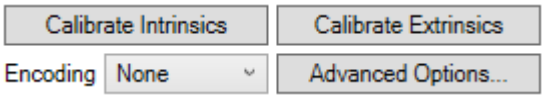
选定的原点是指校准模式的对应点，而不是指整个图像。

全局坐标



全局坐标设置全局坐标系的方向。这可以用来设置一个与图像坐标系相比不同的方向。在所示的默认设置下，两个坐标系相匹配。

操作



校准内在

校准内在计算相机矩阵和畸变系数。所有选择的图像都用于此目的。这些都必须包含所定义的校准模式的全部内容。如果不是这种情况，就会出现负的**再投影。显示错误**。

校准外在

校准外在计算旋转矩阵和平移矢量。这方面的一个先决条件是内在校准（相机矩阵和畸变系数）和选择图像，其中二维校准图案的点位于随后的测量平面内。

校准的持续时间

校准的时间随着校准模式中参考点的数量和校准图像的数量而增加。因此，在选择校准模式时，确保持续时间保持在可接受的范围内。

结果的偏差

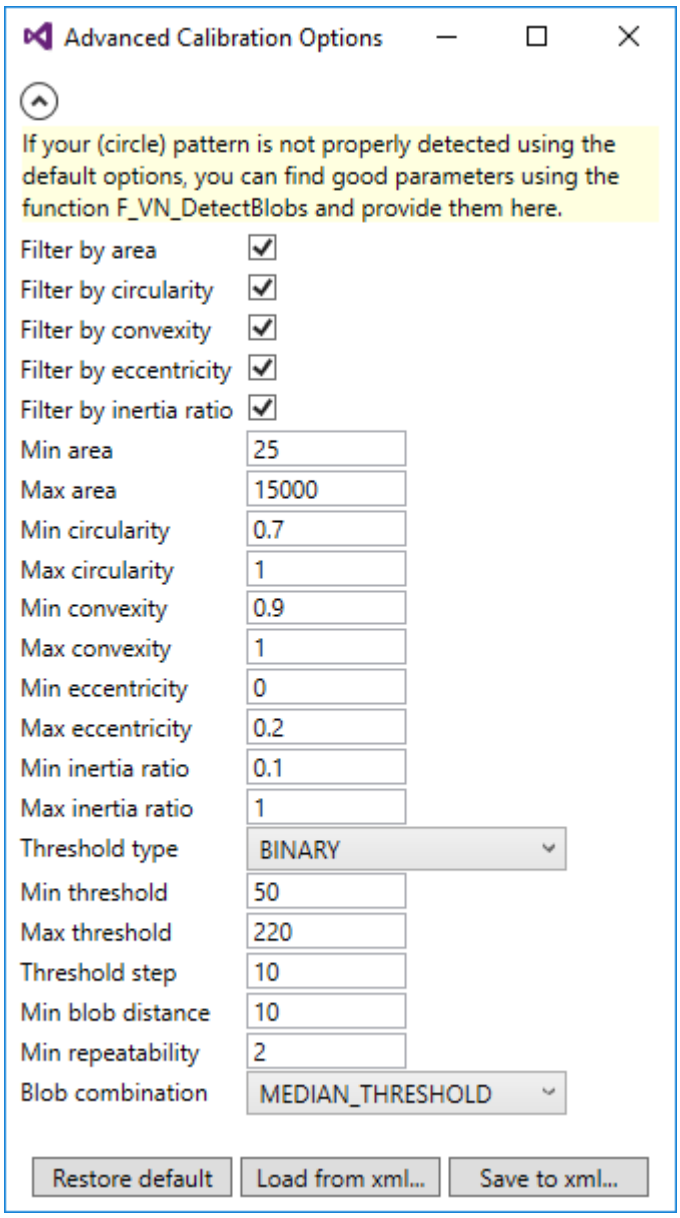
由于所使用的算法，校准的结果在不同的调用中在小数位范围内可能彼此略有不同。这涉及到校准助手中的校准结果和 PLC 中的校准结果，以及两个校准变体之间的校准结果。

将设置转移到 TwinCAT 项目中

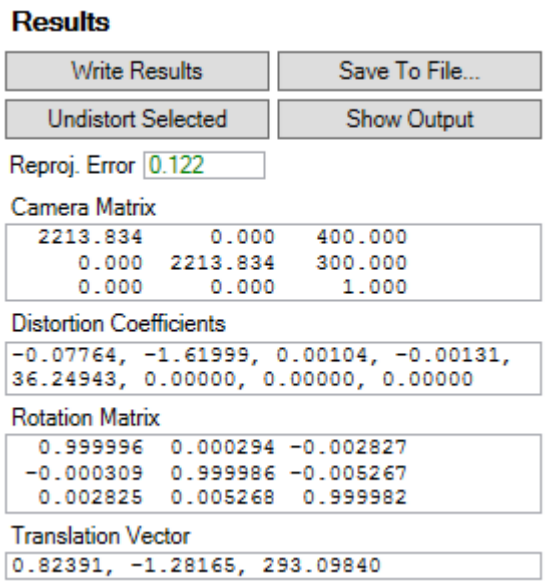
仅在执行**校准内在**或**校准外在**功能后，设置的值和选项才会应用于 TwinCAT 项目。如果事后保存项目，这些值将在再次打开项目时被重新加载。否则，将始终使用默认设置。

高级选项

圆形图案识别可以通过**高级校准选项**重新进行参数化。仅在默认设置存在困难的情况下才可以这样做。为了更好地理解设置参数，请参见函数的参数化F_VN_DetectBlobs [▶ 1171]。



结果



再投影 误差

再投影误差表示内部标定的结果：

- **>0**：有效结果
  - **<1**：良好结果
  - **>2**：不准确结果
- **<0**：无结果和具体错误编号：
  - **-100**：未找到图像
    预览窗口下方的图像集合用于内部标定。不考虑相机的实时图像。
  - **-200**：至少在一个图像中找不到模板。
    预览窗口下方集合中的所有图像必须显示完整模板，包括所有特征点。为了更好地进行分析，会在 C:\ProgramData\Beckhoff\Vision_CalibrationAssistantOutput 下创建结果图像，显示找到的特征点。
  - **-300**：现有图像中均不包含所选模板。
    请参见适用于 -200 的描述，此外，建议检查模板配置。
  - **-400**：单个标定模板的用户定义描述文件无效。
  - **-500**：发生了一个错误。

相机矩阵

相机矩阵描述从三维到二维的图像过程。

$$CameraMatrix = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

- 参数 f_x 和 f_y 表示 x 和 y 方向上的焦距。
- 参数 c_x 和 c_y 表示光轴与图像的交点，称为主点。理想情况下，其应位于图像中心。如果在 **Intrinisic Options（内部选项）** 下选择了 **Fix Center Point（固定中心点）**，参数 c_x 和 c_y 将自动设置为图像的中心。

畸变系数

光学系统的畸变会导致图像中发生几何畸变。这些问题可以在坐标转换过程中通过**几何畸变系数**得到部分补偿。

$$DistortionCoefficients = \left[k_1, k_2, p_1, p_2, k_3, k_4, k_5, k_6\right]$$

- 系数 $\mathbf{k}_1$ **至** $\mathbf{k}_6$ 描述图像中的径向几何畸变。
- 系数 $\mathbf{p}_1$ 和 $\mathbf{p}_2$ 描述图像中的切向几何畸变。

旋转矩阵

旋转矩阵描述相机坐标系相对于世界坐标系的定向，反之亦然。

$$RotationMatrix = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

平移向量

平移向量描述相机坐标系的坐标原点相对于世界坐标系的位移，反之亦然。

$$TranslationVector = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

写入结果

结果将写入相机的图像提供程序 TcCOM 对象。这意味着可以通过功能块 F_VN_GevCameraControl [▶ 1609] 从 PLC 调用它们。

即使尚未进行标定，也会写入设定标定模板的描述。这可通过 GetCalibPatternRef [▶ 1615] 方法在 PLC 中进行检索。另外，也可通过 F_VN_GenerateCalibrationPatternReferencePoints [▶ 1615] 直接在 PLC 中生成标定模板的描述。

保存到文件……

结果可以保存在 json 文件中。此外，还可通过 FB_VN_ReadCalibrationResult [▶ 1588] 将结果读入 PLC。

已选择去几何畸变

几何畸变系数应用于所选图像，结果显示在单独窗口中。

显示输出

打开目录 C:\ProgramData\Beckhoff\Vision_CalibrationAssistantOutput。创建结果图像是为了能够分析发现了哪些特征点以及如何分配这些特征点。

PLC 中的替代方案

如果必须在机器运行时对相机进行校准，在 PLC 中可使用函数F_VN_CalibrateCamera [▶ 1071]进行校准。

在 PLC 中进行校准

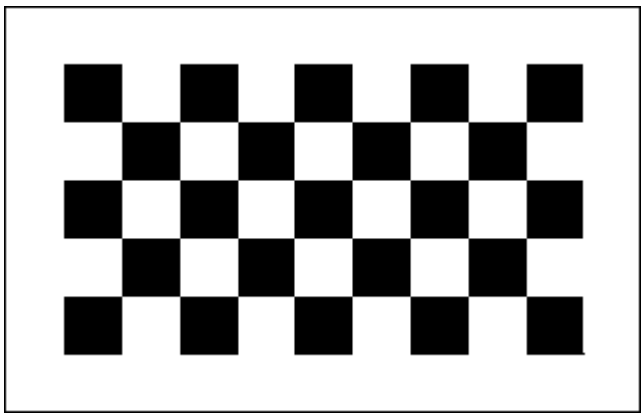
校准的持续时间可能比一个 PLC 周期长得多。因此，在 PLC 中进行校准时，确保在你的机器序列中不会出现问题。

校准助手样本

校准助手 [▶ 2822]

5.4.4.1 棋盘图案

棋盘图案由浅色和深色的方块交替组成，它们共同构成一个矩形图案。参见下面包含 4 x 8 交叉图案的棋盘示例：



对该图案的要求

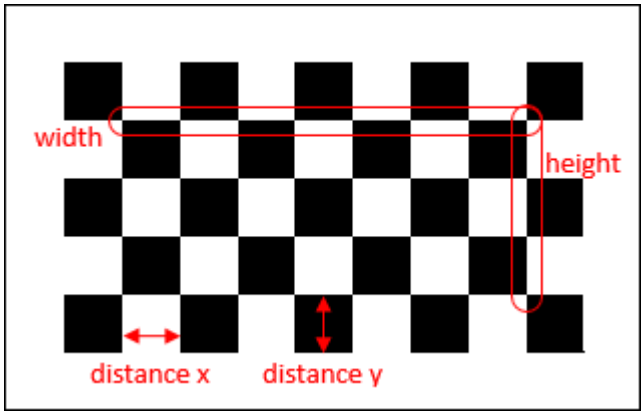
- 图案应用于尽可能平坦的表面（平面），例如玻璃板。
- 浅色和深色的方块必须
  - 以交替的方式排列，
  - 尺寸相同
  - 并且是方形。
- 一个图案必须包含至少 16 个完全可见的方块，且这些方块共同构成至少 3×3 的交叉图案。
- 图案周围的背景色必须形成一个‘安静区’。

对图像采集的要求

- 曝光
  - 图案照度均匀
  - 画面中的明暗方块对比良好
  - 在交叉口没有过度曝光导致的开口
- 数字
  - 一个方块必须至少由 15x15 像素表示。

相机校准助手 [▶ 106]中的配置

- **宽度**
  对应于水平方向的交叉点的数量。
- **高度**
  对应于垂直方向上的交点数量。
- **距离 x**
  对应于水平方向上两个交叉点之间的距离。
- **距离 y**
  对应于垂直方向上两个交叉点之间的距离。



5.4.4.2 对称圆形图案

对称圆形图案由均匀地排列在行和列中的圆形组成。

其他支持的圆形图案有：不对称圆形图案 [▶ 116]和独立圆形图案 [▶ 118]。

对该图案的要求

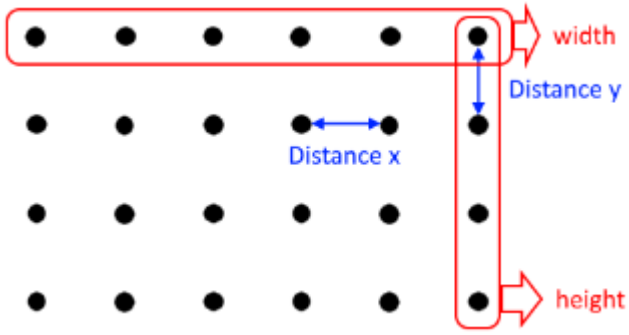
- 图案应用于尽可能平坦的表面（平面），例如玻璃板。
- 圆圈的大小应尽可能相等。
- 一个样本必须由至少 9 个点组成。
- 圆心必须在水平和垂直方向的相同距离上应用。这个距离必须足够大，以确保圆圈不会相互接触，并被识别为独立的对象。
- 图案周围的背景色必须形成一个‘安静区’。

对图像采集的要求

- 曝光
  - 图案照度均匀。
  - 圆圈和背景之间的对比良好。
- 数字
  - 透视畸变应保持在最低限度。
  - 必须遵守或适应**高级校准选项**的标准规格。默认情况下，一个圆必须由至少 25 个像素且最大 15000 像素的区域来表示。这个区域不能太小，否则就会接收到来自背景的干扰。

相机校准助手 [▶ 106]中的配置

- **宽度**
  对应于每行的圆圈数量。
- **高度**
  对应于每列的圆圈数量。
- **距离 x**
  对应于水平方向上两个圆心之间的距离。
- **距离 y**
  对应于垂直方向上两个圆心的距离。



- **颜色反转**
  默认情况下，假定深色圆圈搭配浅色背景。如果图案由浅色圆圈搭配深色背景组成，则可以通过**颜色反转**选项反转颜色。

5.4.4.3 不对称的圆圈图案

不对称圆圈图案类似于对称圆圈图案 [▶ 116]，只是每隔一行就偏移半列。这使得两行之间的距离更小，允许在同一区域设置更多的线条。

图案要求

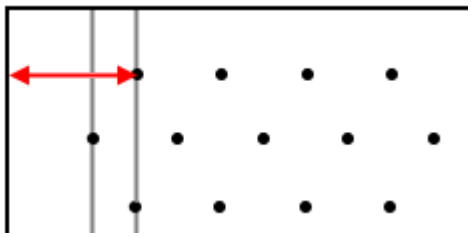
- 图案应用于尽可能平坦的表面（平面），例如玻璃板。
- 圆圈的大小应尽可能相等。
- 一个样本必须由至少 9 个点组成。
- 圆心必须在水平和垂直方向的相同距离上应用。这个距离必须足够大，以确保圆圈不会相互接触，并被识别为独立的对象。
- 图案周围的背景色必须形成一个‘安静区’。

对图像采集的要求

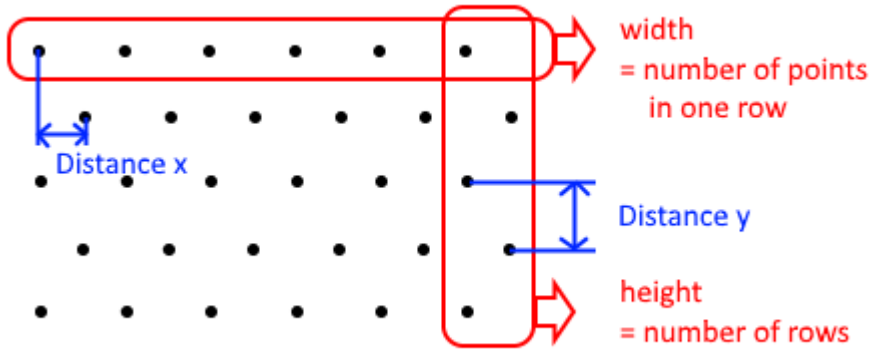
- 曝光
  - 图案照度均匀。
  - 圆圈和背景之间的对比良好。
- 数字
  - 透视畸变应保持在最低限度。
  - 必须遵守或适应**高级校准选项**的标准规格。默认情况下，一个圆必须由至少 25 个像素且最大 15000 像素的区域来表示。这个区域不能太小，否则就会接收到来自背景的干扰。

相机校准助手 [▶ 106]中的配置

- **开始缩进**
  第一行相对于第二行缩进。



- **宽度**
  对应于每行的圆圈数量。
- **高度**
  对应于每列的圆圈数量。
- **距离 x**
  对应于水平方向（跨行）上到最近圆心的距离。
- **距离 y**
  对应于垂直方向上到最近圆心的距离（在一列内）。



- **颜色反转**
  在在深色背景上使用浅色图案时。

5.4.4.4 单个圆圈图案

单个圆圈模板由多个圆圈组成，用户可以选择自定义位置，从而对单个圆圈组件进行排列。这也支持使用复杂的定制 3D 模板进行标定。

模板要求

- 模板应尽可能应用于平坦（平面）表面
- 圆圈大小应尽可能相等。
- 一个模板必须由至少 9 个点组成。
- 距离必须足够大，以确保圆圈不会相互接触，并会被识别为独立物体。
- 模板周围的背景颜色必须形成一个“安静区”。

图像采集要求

- 曝光
  - 模板的均匀照明。
  - 圆圈与背景对比鲜明。
- 图
  - 透视失真应保持在最低限度。
  - 必须遵守或调整**高级标定选项**的标准规格。默认情况下，一个圆圈必须用至少 25 像素、最多 15000 像素的区域来表示。区域不能太小，否则会接收到背景干扰。

描述文件要求

- 模板必须以 XML 文件的形式描述，并存储在开发计算机的以下文件夹中：C:\Users\Public\TcVision\CalibrationPattern
- XML 文件必须根据 %TwinCAT3DIR%\Components\Vision\TcVision 文件夹中的 TcVnCalibrationPattern.xsd 模式进行验证。
- 描述文件必须至少包含显示名称、原点和圆点的二维或三维坐标。

中心点

由于中心点是由点的描述自动给出，因此原点规格只是充当补充信息，在计算中不进行评估。

二维模板

当指定二维模板时，在内部使用 0 的 Z 坐标以进行计算。

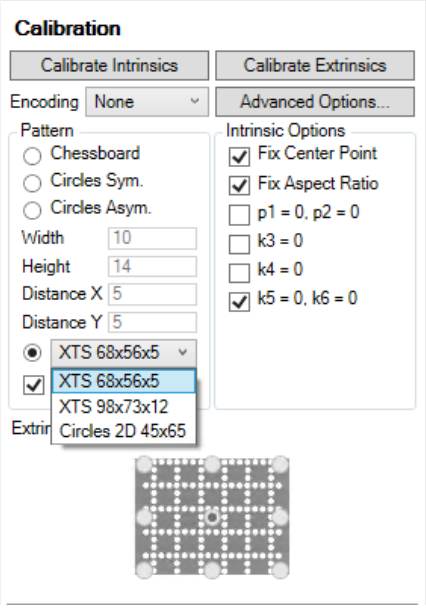
常量偏移量

常量偏移量可以作为具有常量 Z 值的三维模板来实现。

相机标定助手 [▶ 106]中的配置

如果在指定文件夹中找到有效 XML 描述文件，则会用从该文件中读取的显示名称显示这些文件。

- **单个圆圈模板的下拉菜单**
  选择单个圆圈模板。
- **颜色反转**
  用于选择图像颜色反转的复选框。默认情况下（无复选标记），白色背景上显示黑色对象。
- **外部原点**
  该值将自动设置为针对所选模板存储在文件中的值，无法通过用户界面更改。
  - 如果在描述文件中指定了包括文件扩展名在内的图像名称，且图像文件位于同一文件夹中，则会相应显示标定模板示意图的可选背景图像。



单个圆圈模板描述文件的样本示例

以下代码块可用作创建您自己描述文件的模板，并包含已完成元数据和二维点描述的样本。三维点描述也包括在内，但已被注释掉。如果需要三维，则必须用 Pattern3D 块完全取代 Pattern2D 块，即不允许混合。

点分配

如果点按行或列排序，并与标定图像中点的位置直接匹配，这会很有帮助。这样可以更快且更可靠地分配点。

调整坐标系

用于校准的坐标系必须与图像的坐标系相对应，在 X 轴上从左到右，在 Y 轴上从上到下。这也适用于在 PLC 中进行手动校准以及使用 F_VN_SortDetectedPatternPoints 函数。

```
<?xml version="1.0" encoding="UTF-8"?>
<TcVnCalibrationPattern xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="file:///C:/TwinCAT/3.1/Components/Vision/TcVision/
TcVnCalibrationPattern.xsd">
    <MetaData>
        <DisplayName>12345</DisplayName>
        <Origin>Center</Origin>
        <PreviewImage>PreviewImageSample.png</PreviewImage>
        <Description>This is a calibration pattern</Description>
        <Revision>V1.0</Revision>
        <Width>10</Width>
        <Height>10</Height>
        <Depth>4</Depth>
        <NumberOfPoints>9</NumberOfPoints>
        <PointDiameter>1</PointDiameter>
    </MetaData>
    <Pattern2D>
        <!-- 1st row -->
        <Point2D x="-5" y="-5"/>
        <Point2D x="-0" y="-5"/>
        <Point2D x="5" y="-5"/>
        <!-- 2nd row -->
        <Point2D x="-5" y="0"/>
        <Point2D x="0" y="0"/>
        <Point2D x="5" y="0"/>
        <!-- 3rd row -->
        <Point2D x="-5" y="5"/>
        <Point2D x="0" y="5"/>
        <Point2D x="5" y="5"/>
    </Pattern2D>
    <!-- <Pattern3D> -->
        <!-- 1st row -->
        <!-- <Point3D x="-5" y="-5" z="-2"/> -->
        <!-- <Point3D x="-0" y="-5" z="-2"/> -->
        <!-- <Point3D x="5" y="-5" z="-2"/> -->
        <!-- 2nd row -->
        <!-- <Point3D x="-5" y="0" z="0"/> -->
        <!-- <Point3D x="0" y="0" z="0"/> -->
        <!-- <Point3D x="5" y="0" z="0"/> -->
        <!-- 3rd row -->
        <!-- <Point3D x="-5" y="5" z="2"/> -->
        <!-- <Point3D x="0" y="5" z="2"/> -->
```

```
        <!-- <Point3D x="5" y="5" z="2"/> -->
    <!-- </Pattern3D> -->
</TcVnCalibrationPattern>
```

5.4.5 GigE Vision 标准版

TwinCAT Vision 或产品 TC3 GigE Vision Connector 已根据 GigE Vision 标准版（版本 2.0）获得认证。因此，TwinCAT Vision 可以与所有兼容 GigE Vision 的相机配合使用。由于每个版本的 GigE Vision 标准版都细分为必备功能和可选功能，因此认证并不意味着支持该版本的所有功能。由于相机本身也可能具有不符合 GigE Vision 标准版的附加功能（例如数据压缩），因此无法对全力支持做出声明。因此，建议对所需功能分别进行比较和测试。

GigE Vision 标准版

GigE Vision 是工业图像处理的标准接口。便于配置和操作工业相机。GigE 代表千兆以太网。通过使用千兆以太网通信协议，GigE Vision 可受益于以下功能：

- 默认传输速率为 1 千兆比特/秒（可扩展为 2.5 千兆比特、5 千兆比特和 10 千兆比特）。
- 电缆长度最大可达 100 米，无需放大。
- 使用千兆以太网的标准硬件和软件接口。

GigE Vision 标准版定义了通过 UDP/IP 与兼容设备的通信，由以下四个要素组成：

- GigE Vision Control Protocol（GigE Vision 控制协议，GVCP）定义了 GigE Vision 设备的寻址方式。它规定了在 PC 和相机之间传输图像和配置数据的数据通道和机制。
- GigE Vision Stream Protocol（GigE Vision 数据流协议，GVSP）规定了用于将图像从相机传输到 PC 的不同数据类型和传输方法。Packet Resend（数据包重发）选项也可用于修复传输错误。
- GigE Device Discovery Mechanism（GigE 设备发现机制）可以通过搜索查询找到网络中的相机。
- XML 文件包含 GenAPI 说明，其中定义了相机的所有公开函数。本说明以 GenICam 标准版为基础。

GenICam 标准

关于相机 GenICam 标准和 GenAPI 描述的信息，可查看EVMA 网站。

兼容性

如何了解相机是否与 TwinCAT Vision 兼容：查看产品说明中是否明确命名为“GigE Vision”或显示 GigE Vision 徽标：

“GigE”对比“GigE Vision”

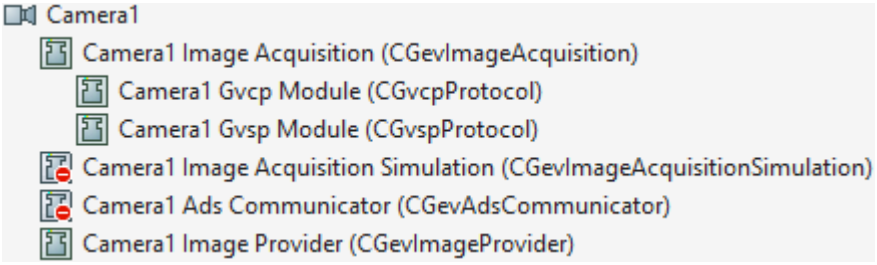
注意“GigE”与“GigE Vision”之间的区别！仅支持符合 GigE Vision 标准的相机。

不支持与其他设备连接

此外，GigE Vision 标准版还允许与其他设备（如闪光灯控制器）连接。TwinCAT Vision 目前不支持此功能。

5.4.6 TcCom 对象

在 TwinCAT 项目中，每个 GigE Vision 相机实例均包含若干 TcCOM 对象：



由 TwinCAT Vision 自动管理的 TcCOM 对象

TcCOM 对象始终由 TwinCAT Vision 自动管理，无需手动调整或配置。因此，以下信息主要用于提供更全面的技术理解。

循环任务

如果 TcCOM 对象 `GevImageAcquisition`、`GevImageAcquisitionSimulation` 和 `GevAdsCommunicator` 被激活，则必须将它们与循环任务相链接。相应配置载于对象的 **Context（上下文）** 选项卡中。通常情况下，无需手动链接任务，因为创建 Vision 设备时所需的任务会自动创建和链接。有关创建和链接任务的详细信息，请参见 CPU 内核和任务 [▶ 63] 章节。

GigE Vision 图像采集

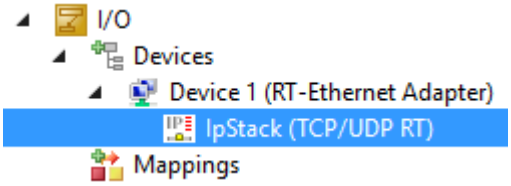
GevImageAcquisition 对象协调与 GigE Vision 相机的通信。它整合并管理实现 GigE Vision [▶ 120] 标准的两个协议的两个子对象：

- GigE Vision 控制协议（GVCP）
- GigE Vision 流协议 (GVSP)

有关相机通信的诊断信息载于两个子对象的 **Parameter (Online)（参数，联机）** 选项卡。

它还在适当的网络适配器上使用 UDP/IP 栈与 GigE Vision 相机通信。诊断信息也可以在这些对象中检索到。有关这两个诊断选项的详细信息，请参见位于相机通信 [▶ 2853]的“附录/故障排除”部分。

在 TwinCAT 项目的 I/O 节点下可以找到 RT 以太网适配器和 IP 栈：



图像采集和两个协议（GVCP 和 GVSP）的参数设置在配置树 [▶ 81]中的配置助手 [▶ 77]中进行。有关参数的描述，请参见 TcCOM 参数 [▶ 99] 章节。相机仿真激活时，此对象将被停用。

GigE Vision 图像采集模拟

GevImageAcquisitionSimulation 对象模拟了 GigE Vision 相机数据流行为，且默认情况下停用。如果模拟模式处于激活状态，GevImageAcquisition 对象反而会被停用。

启用块 ID 校正	该设置可用于明确指出应使用人工生成的连续块 ID，而不是原始 ID。 默认：禁用

GigE Vision Ads 通信器

GevAdsCommunicator 对象在记录相机流 [▶ 101]时需要。它通过 ADS 将相应的图像从路由器内存发送到 TwinCAT Vision 服务。相比通过功能块 FB_VN_WriteImage 保存图像，记录功能需要单独的 TcCOM 对象，因为记录是独立于 PLC 处理而运行。

在配置模式下，再使用一个 GevAdsCommunicator 模块的实例，以便所有相机助手能够与相机进行通信。

发布图片	通过该设置，Ads 通信器每隔 n 个图像就会向 TwinCAT Vision 服务发送一次。该设置在激活数据流记录时自动设置，并取决于记录/回放选项卡 [▶ 101] 的**图像偏移**参数。它不应手动设置。 默认值：0

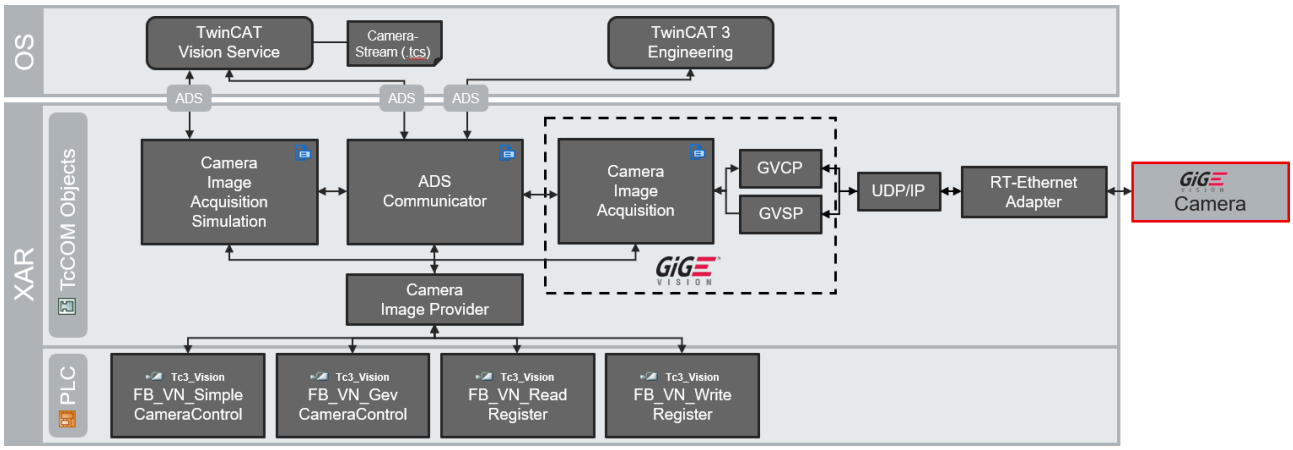
GigE Vision 图像提供者

GevImageProvider 对象代表相机和 PLC 中相应功能块之间的接口。其中包括 `FB_VN_GevCameraControl`、`FB_VN_SimpleCameraControl`、`FB_VN_ReadRegister`和 `FB_VN_WriteRegister`以及`FB_VN_ReadMemory`和`FB_VN_WriteMemory`。为此，项目实例的符号初始化选项卡中的对象必须与所需功能块的 PLC 符号相连接。如果项目中存在多个相机实例，确保根据名称选择正确的 GevImageProvider 模块。

图像队列大小	如果 PLC 中的功能块不直接接受所采集图像，需要进行缓冲的采集图像数量。 默认值：1

链接 TcCOM 对象

下面的概览图显示了相互通信或检索数据的不同对象之间的简化关系。



附图 5: GigE Vision 相机功能的架构

TcCOM 对象的链接根据模拟模式和 Ads 通信器对象的设置而有所不同（见常规选项卡 [▶ 73]）。关于相应的链接设置，可查看各个对象的**接口指针**选项卡。默认情况下，该链接根据相机助手中的设置自动创建，而不应手动更改。如果链接被错误更改或者 TcCOM 对象后来被添加到相机中，则可以在**常规**选项卡中单击**尝试修复**修复链接。



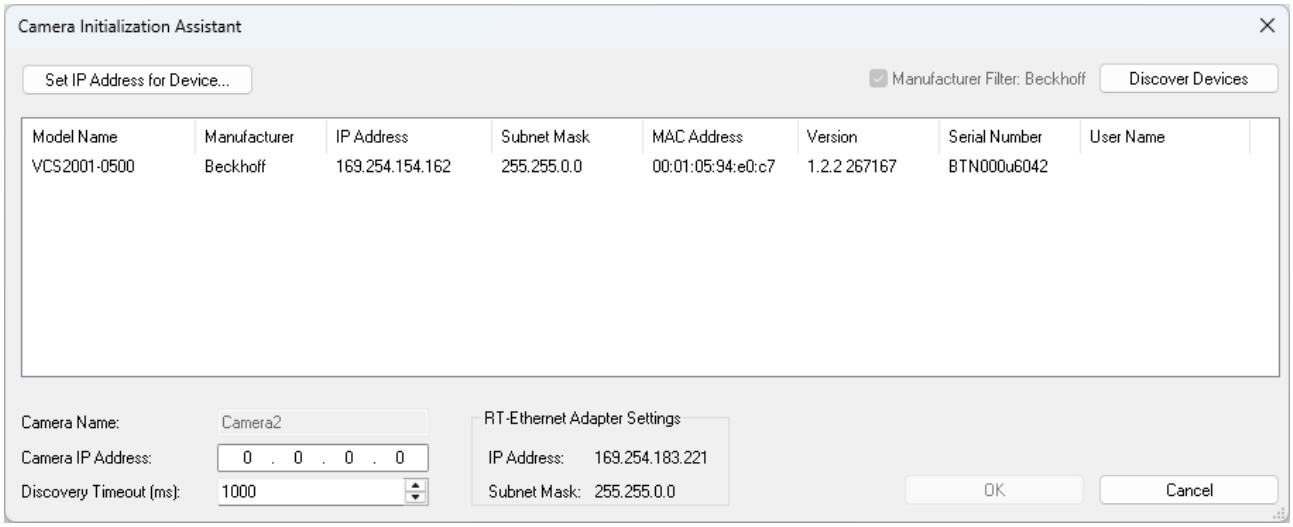
附图 6: 关于 TcCOM 对象的无效链接的说明

5.5 倍福相机对象

每个倍福相机对象都代表项目中的一台相机，并以不同的名称和图标加以标识。一个项目可以包含多个相机对象。由于配置和操作与 GigE Vision 相机对象 [▶ 71]相同，详细信息可在本章相应页面找到。



只有倍福的相机才能与倍福相机对象连接并进行操作。因此，制造商筛选器会在 Camera Initialization Assistant（相机初始化助手）中始终处于激活状态，用户无法取消选择。在建立相机连接时，也要检查这一点，以便在 Configuration Assistant（配置助手）和 Run Mode（运行模式）中仅使用倍福的相机。



5.6 文件源对象

文件源代表相机的另一个图像源。它使存储的图像可以从文件系统中加载到 TwinCAT 实时系统中，而不是实时图像。请注意，与 GigE Vision 相机对象的记录/回放 [▶ 101]功能相比，文件源不是用于加载连续的相机流，而是用于加载单个图像。

只要图片是以标准的图片格式保存，如 bmp、jpg、png，就可以使用。

压缩图像格式

当使用压缩图像格式时，这取决于所应用的方法是否有损，因为图像数据之后将不再对应于相机的原始图像数据。因此，建议使用 BMP 作为存储格式。

灰度图像可以是彩色图像

看起来像灰度图像的图像可以用几个通道保存。在调用转换功能时注意这一点。

功能块FB_VN_FileSourceControl [▶ 1601]和FB_VN_SimpleCameraControl [▶ 1630]的存在是为了能在 PLC 中接收图像。与文件源控制的通信通过这些功能块进行，类似于与相机的通信方式。另外，将单个图像加载到 TwinCAT 实时中的另一个选择功能块FB_VN_ReadImage [▶ 1590]。

图像传输模式

文件源控制有两种提供图像的模式：

1. 数据流模式：像流式相机一样，以指定周期提供新的图像。
2. 触发模式：通过触发信号提供新的图像。这可以通过软件触发器或手动启动按钮触发。

设置和图像选择在文件源控制 [▶ 124]下完成。

创建文件源节点

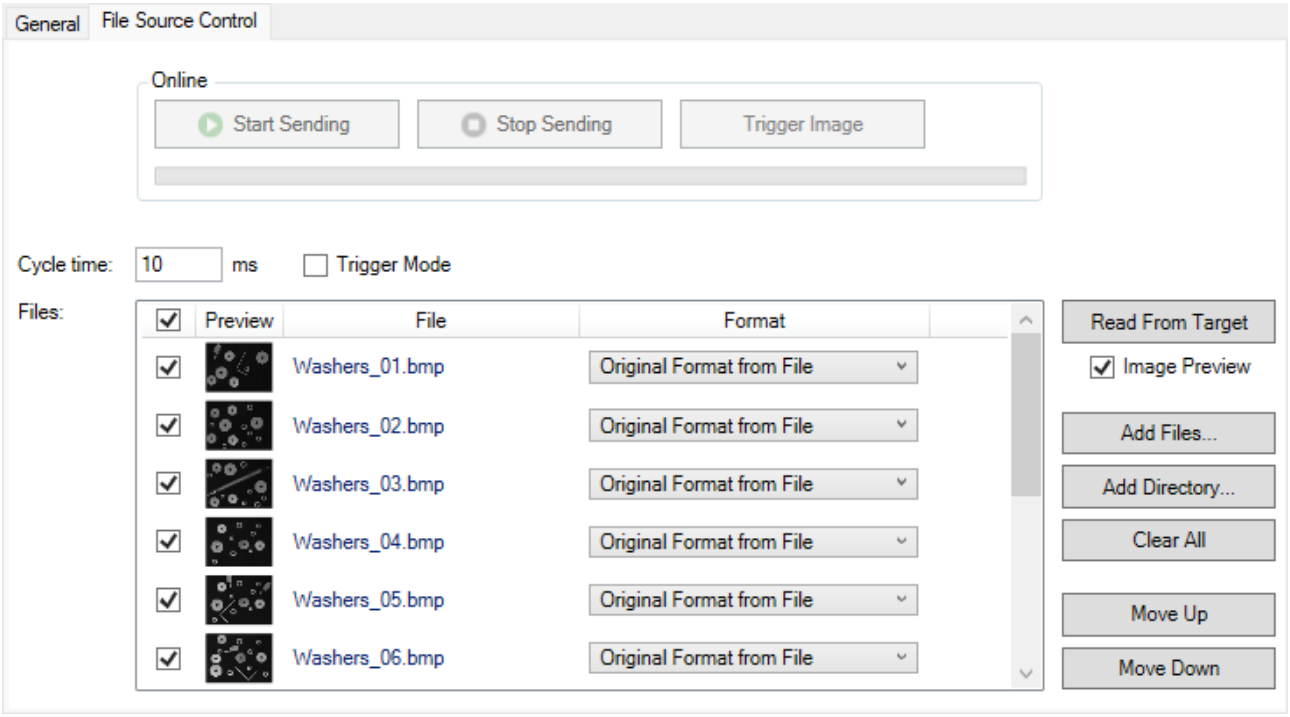
关于创建文件源节点的信息，可参见创建文件源控制 [▶ 35]下第一步 [▶ 27]。

文件源节点的导出/导入

如果想要保存文件源的配置或将其导入项目，请按照导出/导入配置 [▶ 130]一章中的说明进行操作。

5.6.1 文件源控制

文件源控制的配置在同名的选项卡上进行：
视觉节点 > 文件源节点 > 文件源控制选项卡。



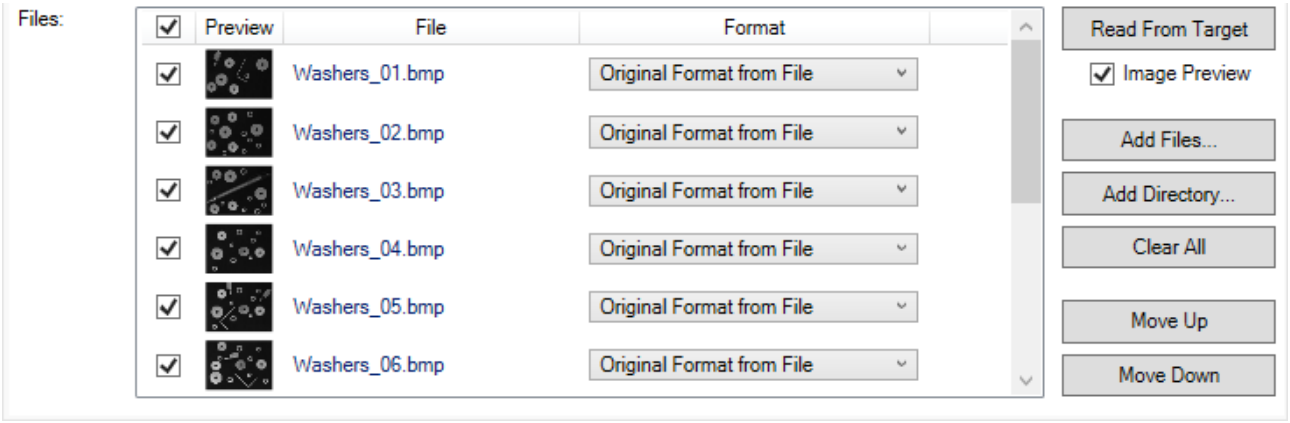
附图 7: 文件源控制

管理图像选择

文件源控制器所使用的图像通过文件列表和相关控制器进行管理：

从目标读取

创建文件源后，必须按一次 **Read from Target（从目标读取）**并激活配置，以便添加图像。



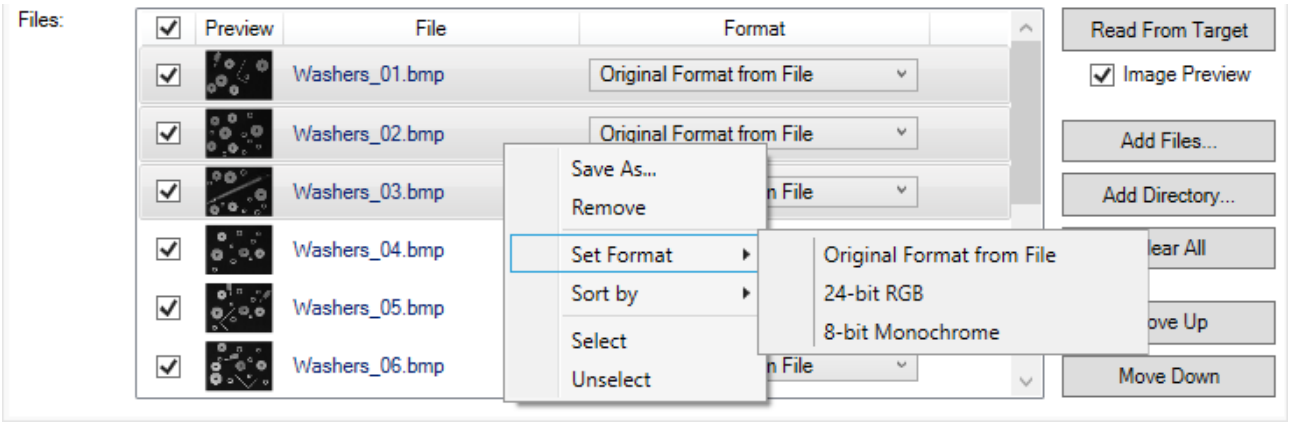
附图 8: 在文件源控制器中选择图像

复选框	第一列中的复选框指示是否使用图像。
预览	预览图像的大小取决于列宽。
文件	图像的文件名，包括数据类型。
Format（格式）	默认情况下，图像以保存的格式发送。或者，可以指定预先将其转换为单通道灰度图像或 3 通道 RGB 图像，然后再发送。 **我们建议您明确指定每个图像的格式。**

从目标读取	系统会读取相关信息，以了解哪些图像存储在目标系统上，以供关联的文件源控制器使用。
图像预览	选择是否在列表中显示预览图像。如果您有大量图像或尺寸较大的图像，我们建议取消选择预览，因为目标系统加载并显示所有预览图像可能会花费很长时间。
添加文件	通过文件资源管理器可以选择开发系统中的图像。然后，这些图像会在后台被复制到目标系统，从而独立于开发系统在目标系统中使用。请注意，只能使用文件名唯一的文件。此外，这里只能添加图像格式的文件，而不能添加视频格式的文件。如果您希望将相机流（*.tcs）加载到 PLC 中，请使用 Record/Playback（录制/回放）[▶ 101]功能。 **除非另有配置，否则图像文件将存储在目标系统的“C:\Users\Public\TcVision\FileSources”路径下。确保相应路径不受诸如 WriteFilter 等其他软件的影响。**
添加目录	通过文件资源管理器可以选择开发系统的一个目录，该目录下的所有图像都将被使用。
全部清除	将从文件列表中移除所有图像。
向上移动/向下移动	选中图像在文件列表中向上/向下移动一个位置。

上下文菜单

在文件列表中，使用 **CTRL+点击**可以选择多个图像。右键点击其中一个选中图像，即可调整所有选中图像的设置：

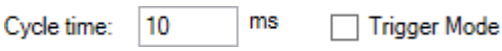


附图 9: 在文件源控制器中选择图像的上下文菜单

另存为...	打开一个文件资源管理器对话框，用于保存选中的图像。
Remove（删除）	删除选中的图像。
设置格式	为所有选中的图像设置图像格式： • **文件原始格式：**使用图像存储时所采用的格式。 • **24 位 RGB：**文件源以 3 通道 RGB 图像的格式发送图像。 • **8 位单色：**文件源以单通道灰度图像的格式发送图像。
排序依据	按文件列表对所有图像进行排序： • **名称（升序）：**按名称升序排列 • **名称（降序）：**按名称降序排列
选择	选择高亮显示的图像，以便通过文件源控制器发送。
取消选择	取消选择高亮显示的图像，使其不会通过文件源控制器发送。

图像传输设置

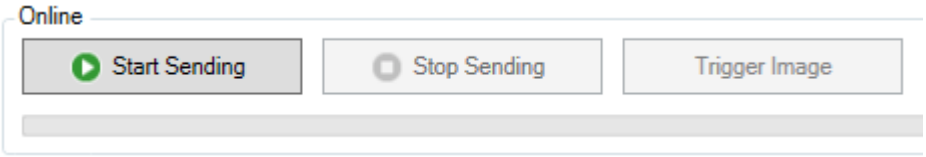
像流式相机一样，文件源控制可以在指定的时间周期或调用软件触发器后提供新的图像。



附图 10: 在文件源控制中设置选项

周期时间	如果没有选择触发模式，图像将在周期时间的间隔内被发送到实时环境中。根据文件列表，图像的顺序被考虑在内。这个参数与相机的帧率相当。 **加载图像的最大速度取决于系统和图像的情况。周期时间只是期望值。**
触发模式	如果选择了触发模式，则在调用相应的软件触发器后会发送图像。通过标准的软件触发器，遵守根据文件列表的图像顺序。另外，也可以根据文件名触发特定的图像。

在线

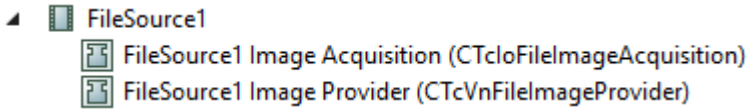


附图 11: 文件源控制中的操作

开始发送	PLC 中的**开始发送**对应于功能块FB_VN_FileSourceControl [▶ 1601]的开始采集 [▶ 1605]方法。如果没有选择触发器，则图像将在**周期时间**间隔通过文件源控制发送。如果选择了触发器，文件源控制会对软件触发器做出反应，并根据触发信号发送图像。
停止发送	PLC 中的**停止发送**对应于功能块FB_VN_FileSourceControl [▶ 1601]的停止采集 [▶ 1606]方法，并可终止通过文件源控制发送图像。
触发图像	PLC 中的**触发图像**对应于功能块FB_VN_FileSourceControl [▶ 1601]的触发图像 [▶ 1608]方法。这可以用来手动触发图像。这个按钮仅在触发模式启用且通过文件源传输图像功能激活的情况下才可用。

5.6.2 TcCom 对象

在 TwinCAT 项目中，每个 FileSource 实例均包含两个 TcCOM 对象：



TwinCAT Vision 自动管理 TcCOM 对象

TcCOM 对象始终由 TwinCAT Vision 自动管理，无需手动调整或配置。因此，以下信息主要用于提供更全面的技术理解。

文件图像采集

TcIoFileImageAcquisition 对象为 FileSource 实例协调从硬盘加载图像。这个对象包含以下参数（除了 `CycleTime`和`TriggerMode`，这些参数可以在文件源控制 [▶ 124]中设置）：

基地目录	目标系统上的文件路径，添加到选择列表中的图像将存储在该路径下。 标准：C:\Users\PublicTcVision\FileSources\<UniqueId>
服务超时	与TwinCAT Vision Service [▶ 68]通信的超时（单位：毫秒）。 默认值：5000

循环任务

文件图像采集对象必须与循环任务相联系。关于相应配置，可查看对象的**上下文**选项卡。通常情况下，无需手动连接任务，因为创建 Vision 设备时需要的任务会自动创建和连接。关于创建和链接任务的详细信息，可查看CPU 内核和任务 [▶ 63]章节。

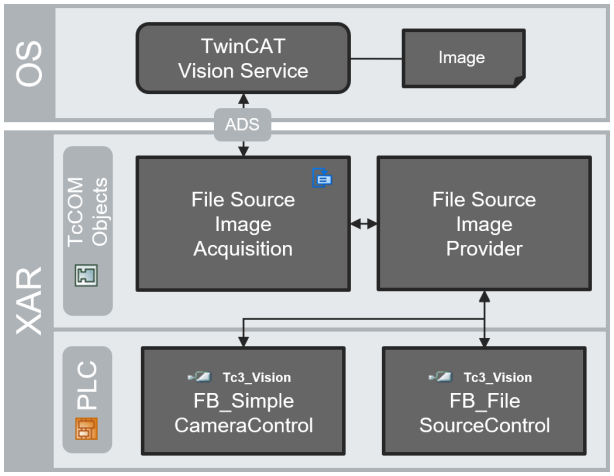
文件图像提供者

这个对象代表了 PLC 中文件源对象和文件源功能块之间的接口。功能块FB_VN_FileSourceControl [▶ 1601]或FB_VN_SimpleCameraControl [▶ 1630]的 ObjectID 变量必须在编译 PLC 代码后在项目实例的**符号初始化**选项卡上与相应的文件图像提供者 TcCOM 对象链接。如果项目中存在几个文件源实例，必须注意根据名称选择正确的对象。这个对象包含以下参数：

图像队列大小	如果 PLC 中的功能块不直接接受已加载的图像，则要对其进行缓冲的数量。 默认值：1

链接 TcCOM 对象

对于文件源设备，仅图像提供者必须与图像采集链接。默认情况下，这个链接会自动创建，且不应手动更改。下面的概览图显示了相互通信或检索数据的不同对象之间的简化关系。



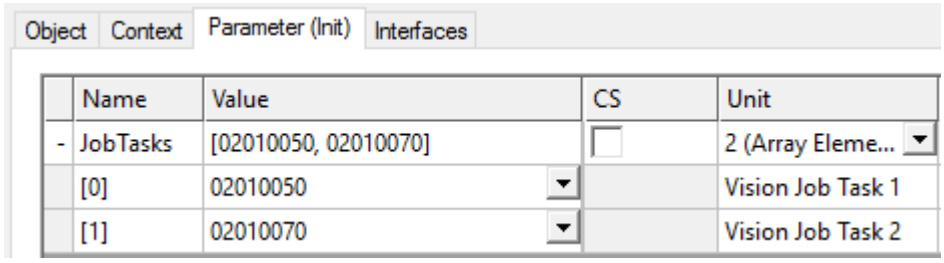
附图 12: 文件源功能的架构

5.7 视觉工作池

视觉工作池用于针对图像处理任务管理工作任务 [▶ 66]。这使得选定的 TwinCAT Vision 算法可以并行执行，而这些算法在API 参考资料 [▶ 132]中由以下补充内容标记：“可使用可用的 TwinCAT 工作任务执行并行代码区域”。



点击**视觉工作池**，打开一个窗口。工作任务的分配在这个窗口的**参数（初始化）**选项卡上进行。在第一行，**单位**栏指示需要分配多少工作任务。然后，**工作任务**下的数组元素可以通过**值**列中的下拉菜单分配一个工作任务。每个工作任务仅可分配给一个元素。

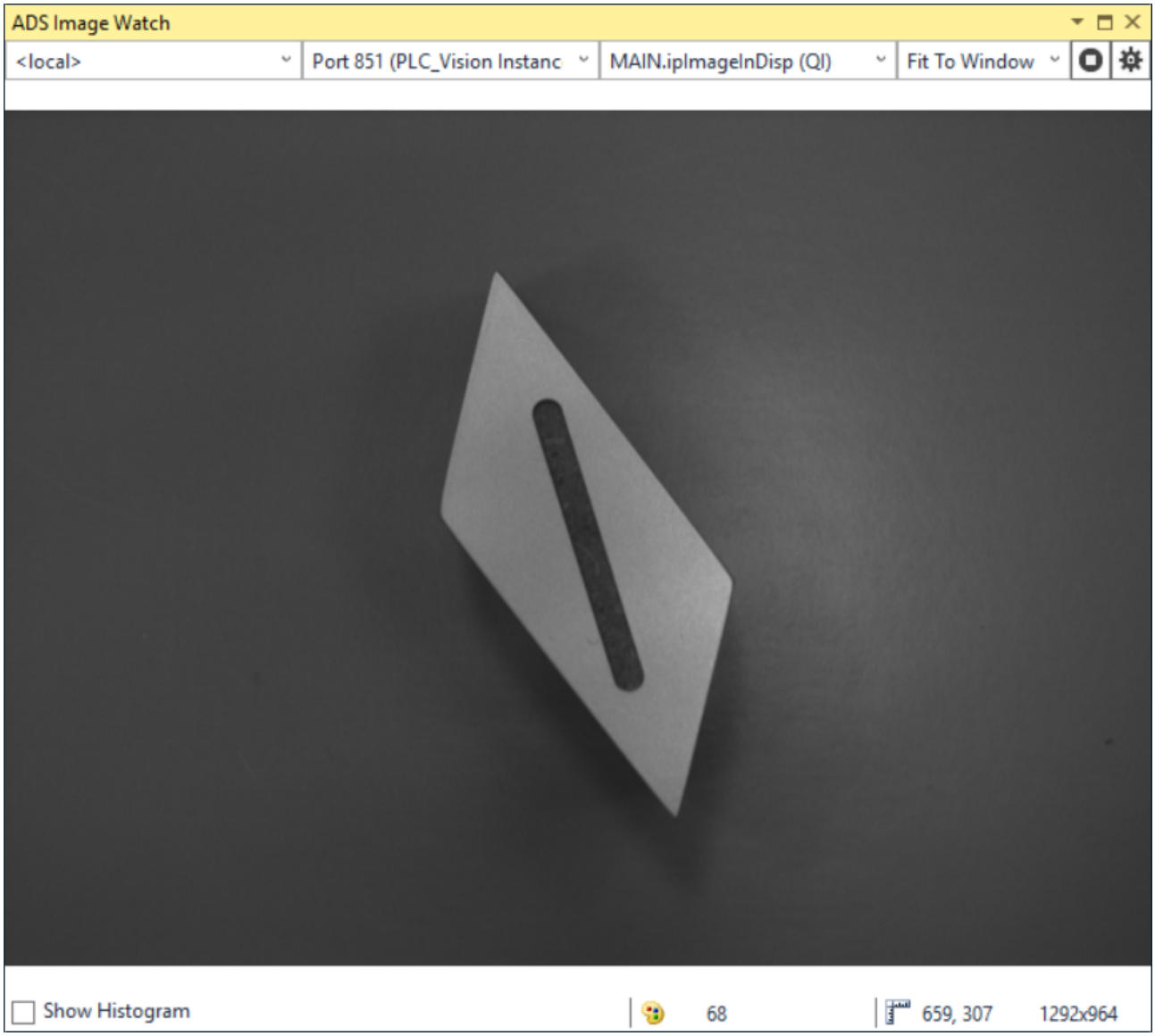


注意

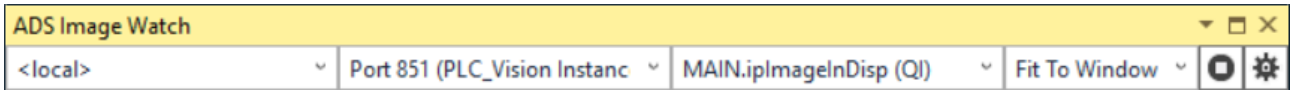
在创建工作任务时，请务必参考限制和建议 [▶ 67]。

5.8 ADS 图像查看

通过 ADS 图像查看，类型ITcVnDisplayableImage [▶ 407]的图像可以从 PLC 中检索到并在开发环境中显示。
通过 Visual Studio 菜单**TwinCAT > Windows > ADS 图像查看**打开 ADS 图像查看。



图像选择



图像选择通过选择区域从左到右进行。

目标系统	ADS 设备通过ADS-AmsNetId 和 ADS 端口寻址。ADS-AmsNetId 明白地确定了必须首先选择的目标系统。默认情况下选择本地系统。然而，可以选择任何已经创建了路线的系统；见选择目标系统 [▶ 29]。
端口	系统中的各个设备由 ADS 端口号识别。为 PLC 运行时间定义了一个固定的端口号范围，因此选择列表已根据这些端口号预先过滤。如果在 ADS 图像查看中选择后目标系统上只存在一个 PLC 端口，则会自动选择 PLC 端口。

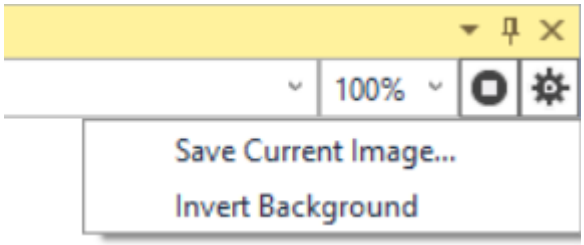
图像变量	所有类型为ITcVnDisplayableImage [▶ 407]的变量均显示在此处且可以选中，无论该变量当时是否指向受支持图像格式 [▶ 129]的图像。
大小	如果没有更多设置，图像会被缩放到当前窗口的大小（**适应窗口**）。然而，它可以选择性地被设置为相对于原始图像的尺寸。请注意，如果选择的尺寸小于原始图像，则在通过 ADS 传输之前图像会被实时环境中的PlcAux 任务缩小。如果之后保存图像，则图像不不具有原始尺寸。
停止/播放	交互停止/播放按钮可用于停止或重新启动图像显示的更新。如果暂停图标可见，则更新处于活动状态，并持续接收图像。如果播放图标可见，说明更新已停止，并保持最后一张图片的显示。不再接收更多的图像。 这使得分析图像和每个细节成为可能，例如通过放大。

图像缩放

如果选择的尺寸不是 100%，显示的图像将被插值。对于**适应窗口**或缩小，使用线性插值，对于放大，使用最近邻插值。

根据图像内容或像素值的不同，可能会出现偏差。如果需要对个别像素进行详细分析，应因此将尺寸设置为 100%。

选项



通过右上方图像大小旁边的**行动**图标，然后使用**保存当前图像**，可以保存当前显示的图像。

此外，显示图像周围的背景可以通过**反转背景**在明暗之间改变。这对非常明亮或黑暗的图像很有帮助，这样可以清楚地看到图像内容或图像边缘，并通过与背景的高度对比来进行区分。

图像大小

图像被保存为 ADS 图像查看从 PLC 查询的尺寸。为了以原始尺寸保存图像，必须将图像至少设置为 100%，或将窗口至少拉到原始图像尺寸。如果图像大小被设置为大于 100%，图像也将以原始尺寸保存。如果设置较小，使用**适应窗口**或窗口太小，图像将以相应的缩小尺寸存储。

支持的图像格式

在 ADS 图像查看中，以下图像格式可以在下列色彩空间中显示：

- **1 通道图像**被解释为**灰度图像**。支持所有像素类型为枚举ETcVnElementType [▶ 191]的图像。
- **3 通道图像**被解释为**RGB 彩色图像**。支持每通道 8 位或 16 位的图像；这对应于像素类型 [▶ 191]USINT、SINT、UINT 和 INT。
- **4 通道图像**被解释为**RGBA 彩色图像**。这里支持每通道 8 位/像素的图像；这对应于像素类型 [▶ 191]USINT 和 SINT。

其他图像格式

其他图像格式和色彩空间可以在 PLC 中作为 ITcVnImage 处理。然而，为了显示它们，它们必须首先通过 F_VN_ConvertColorSpace 转换为可显示的格式之一，然后再转换为 ITcVnDisplayableImage。如果不这样做，图像要么以错误的颜色表示方式显示，要么根本就不显示。

显示 16 位图像

16 位图像作为 8 位图像从 PLC 传输到 ADS 图像查看。因此，像素值只在状态栏中以 256 的增量显示（0、256、512 等），但还是与 16 位有关：（0：最小，65.280：最大）。

显示（L）REAL 图像

（L）REAL 图像作为 8 位图像从 PLC 传输到 ADS 图像查看。为此，数值范围[-1，+1]被缩放为[-127，+127]并以整数传输。在这些限制之外的浮点值将被解释为 -1 或 +1。在 ADS 图像查看中，待传输的像素值再次除以 127，以便在[-1, +1]范围内显示浮点值。

状态栏

图像下方的状态栏显示以下信息（从左到右）：

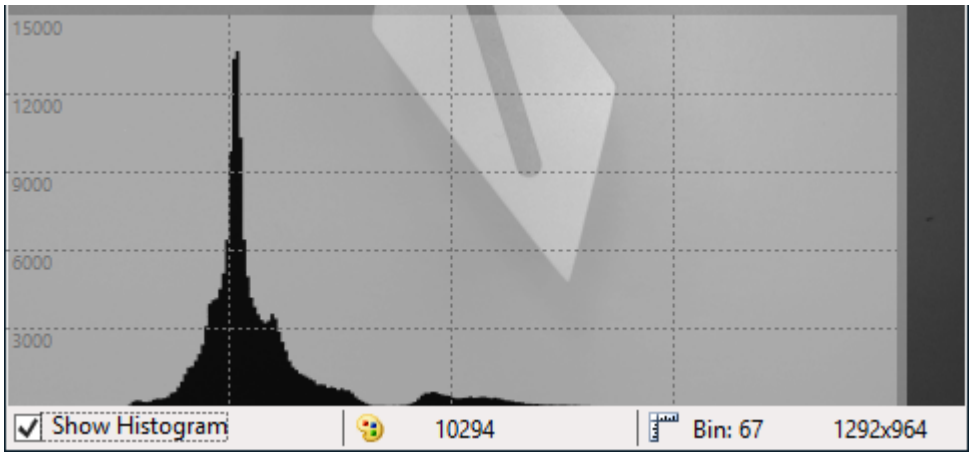


附图 13: ADS 图像查看中的状态栏

- 图像上鼠标指针位置的颜色值/强度值
  或直方图中鼠标指针位置的相应容器中的像素数
- 在图像上鼠标指针位置的图像坐标
  或直方图中鼠标指针位置的容器
- 图像分辨率

直方图

当**显示直方图**复选框被选中时，图像的直方图在图像的左下方显示。这可以有几个通道叠加显示，具体取决于图像类型。



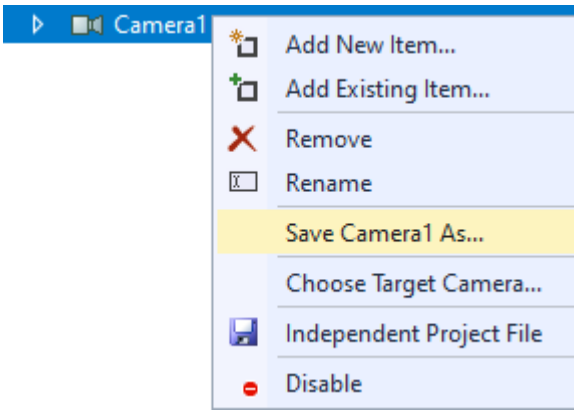
附图 14: ADS 图像查看中的直方图

5.9 导出/导入节点

TwinCAT 提供将 I/O 或 MOTION 元件等单独配置节点保存在单独的 XTI 文件中或从该等文件中导入的选项。使用应用节点以及 GigE Vision 相机和文件源对象也可以实现这一点。这样，Vision 设备的配置可以独立于项目保存，并在其他地方重新导入。

导出

1. 右键点击要导出的节点，并选择 **Save <Name> As...（将<名称>另存为……）**

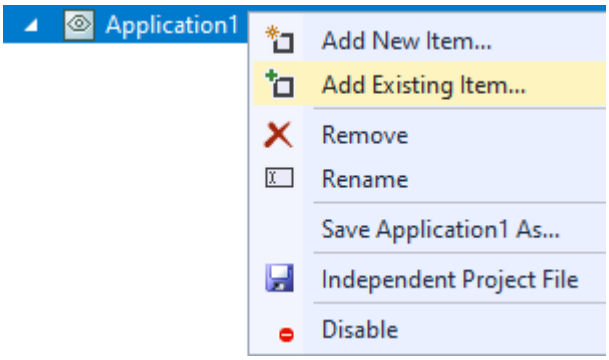


2. 为 XTI 文件选择一个位置。

导入

根据要导入整个应用节点还是只导入单个 Vision 设备，应在 VISION 节点或相应的应用节点上执行以下步骤。

3. 右键点击要导入配置的节点，然后点击 **Add Existing Item...（添加现有项……）**



4. 选择要导入的 XTI 文件。
5. 如有必要，为导入的 Vision 设备应用特定于系统的设置（如网络适配器）。

复制相机配置

请注意，只有在相机型号和固件相同的情况下才能保证为多个相机成功传输相同的配置。否则可能会出现问题，例如相机寄存器地址改变。

独立项目文件

也可以将配置节点保存在项目中的一个独立文件中，而不是完全导出。在这种情况下，相应节点的配置不会保存在 TwinCAT 项目文件中，而是保存在一个单独的 XTI 文件中。例如，这有助于版本控制系统，当修改配置时，不会更改整个项目文件。

6 API 参考

本章介绍 TwinCAT Vision API，它为 TwinCAT 3 Vision 库的编程使用提供了选项。由于 TwinCAT Vision 可以在 TwinCAT 3 PLC 和 TwinCAT 3 C++ 中使用，因此这两个文档构成了基础。因此，所有特定于 TwinCAT Vision 的描述均分为以下两个子章节。

6.1 PLC

本章包含有关在 PLC 中编程使用库的 TwinCAT Vision API 描述。由于 TwinCAT Vision 带来了一些创新，因此对软件概念进行了明确说明：

- 软件概念 [▶ 132]
- 数据类型 [▶ 152]
  （别名、常量、数组、枚举、结构）
- 接口 [▶ 237]
  （图像、容器等）
- 函数
  （图像处理）
- 功能块
  （与相机和文件系统通信、图像处理）

6.1.1 软件概念

本章介绍 TwinCAT Vision 带来的创新或特殊特征。TwinCAT 3 中的PLC 编程原理以及参考编程载于一般 TwinCAT 3 文档。在 PLC 的描述中，有一章介绍TwinCAT 3 编程约定。其中包含的建议和说明以及优缺点可支持实现程序结构标准化，对象、变量、实例命名一致性，以及代码易读性和易懂性，从而简化程序的开发、使用和维护。

对库模块的明确访问

如果 Tc3_Vision 库要包含（引用）到另一个库中，并因此在一个项目中出现多次，则必须通过命名空间作为标识符的前缀进行明确的访问。默认，例如 `hr := Tc3_Vision.F_VN_TransformIntoDisplayableImage(ipImageIn, ipImageInDisp, hr);`

有关更多信息和设置选项，请参见 PLC 项目设置和库属性。

⚠ 警告

使用多个任务时访问相同数据

例如，如果多个任务要访问同一图像，则必须根据这些准则同步这些访问。否则，就有可能出现数据集不一致的情况，从而导致不可预见的后果和程序故障。

6.1.1.1 接口指针

TwinCAT Vision 需要动态内存区域，以保存图像和容器等。这些图像和容器由类型为ITcVnImage [▶ 407]和ITcVnContainer [▶ 365]的变量表示。由于它们是指向动态内存区域的接口，这些变量被称为接口指针。

通过 TwinCAT Vision API 功能，相关的内存管理在功能中被自动控制，且此处无需注意。

例如，创建一个新的图像（800 x 600 像素的类型TCVN_ET_USINT，包含一个通道）：

```
VAR
    hr          : HRESULT;
    ipImage     : ITcVnImage;
END_VAR

hr := F_VN_CreateImage(ipImage, 800, 600, TCVN_ET_USINT, 1, hr);
```

这个图像现在要被转换成 3 个通道的 RGB 图像。因此需要 3 倍的内存。下面的函数调用足以实现这一点：

```
hr := F_VN_ConvertColorSpace(ipImage, ipImage, TCVN_CST_GRAY_TO_RGB, hr);
```

在内部，它释放了先前的内存区域，并以合适的大小分配了新的内存。因此，接口指针是管理 PLC 中动态需要内存的对象的一种简单方法。

⚠ 警告

误用的后果

如不遵循以下关于使用接口指针的说明，可能会导致**系统完全崩溃**和**内存泄漏**!

参考计数器

然而，一旦变量不再专门作为 TwinCAT Vision API 功能的功能参数使用，则需要更准确地熟悉接口指针的概念。

务必了解，接口指针变量只是一个指向内存中的对象的指针。如果你把一个变量赋值给另一个变量，只有这个内存区域的指针被复制，而不是数据本身。

这意味着，如果两个变量中一个变量的数据被更改，另一个也会自动更改。
如果其中一个变量中的内存被释放（如上面的例子F_VN_ConvertColorSpace [▶ 1248]），而第二个变量仍然有其指向这个内存区域的指针，访问这个已经释放的内存往往会导致系统崩溃。

由于这个原因，内存中的对象都有一个参考计数器。只要一个变量被分配给另一个变量，这个引用计数器就必须增加。如果一个新的指针被分配给一个变量，那么必须再次减少参考计数器。

当参考计数器达到 0 时，内存被释放。因此，相应的数据被完全删除。

为了表明一个指针变量是否指向一个有效的内存区域，如果这个指针不再使用（参考计数器因此被减少），这个指针必须始终被设置为 0。默认情况下，当程序启动时，PLC 中所有创建的变量（因此也包括指针）最初被设置为 0。

接口指针的管理

在 TwinCAT Vision API 功能中（用F_VN_...标记），参考计数器由接口指针自动管理。因此，在使用它们时不需要特别注意。然而，如果接口指针被手动复制或使用，例如，作为功能参数，则需要手动管理（见下文）。

检查接口指针的有效性

由于接口指针值 0 基本上只对应一个无效的内存区域，而且对它的访问也会导致严重的系统错误，所以务必在每次使用前检查该值是否为非零。

在 TwinCAT Vision API 功能（`F_VN_...`）中，这种检查和相应的处理在内部自动进行。然而，对于下面示例中的`TcRelease()`等接口方法不可行，因为如果变量为 0，调用这些方法已经导致了错误。因此，必须事先明确检查变量。

⚠ 警告

对无效内存区域的方法访问导致系统崩溃

对于接口方法，内部检查不可行，因为相应的对象必须存在才能调用该方法。

即使在方法调用之外，也要确保相应的对象存在（接口指针应该是非零）。

此外，系统崩溃不一定总是发生。如果无效的内存区域碰巧保持未使用状态，系统崩溃可能不会发生。

启用接口指针

接口指针变量的启用包括检查该变量是否已为 0，否则将减少引用计数器并将该变量设置为 0。由于这是一个经常用例，因此提供了一个函数 `FW_SafeRelease`，只需一次调用即可执行所有必要步骤：

```
hr := FW_SafeRelease(ADR(ipImage));
```

`FW_SafeRelease` 函数的内部运行模式通过以下代码片段进行说明：

```
IF ipImage <> 0 THEN        // Check if interface pointer is already = 0
                            // If not, call .TcRelease() and set it to 0
    ipImage.TcRelease();    // decrement reference count
    ipImage := 0;           // set to 0
END_IF
```

复制接口指针

复制接口指针时，必须手动增加其引用计数器。通过 := 操作符进行直接分配，以及将接口指针作为输入、输出或返回值传输的函数/方法/功能块调用，都属于复制程序。通过 I/O 图像进行的传输也包括在内。可以使用以下方法增加引用计数器：

```
ipImage.TcAddRef(); // increment reference count
```

此外，复制前必须确保要写入的接口指针已启用。否则会出现内存泄漏。例如，复制接口指针的手动程序如下所示：

```
IF ipSrcImage <> 0 THEN                 // check if source pointer is valid
    FW_SafeRelease(ADR(ipDestImage));   // release destination pointer
    ipDestImage := ipSrcImage;          // assign source pointer to destination pointer
    ipSrcImage.TcAddRef();              // increment reference count
END_IF
```

请注意，只复制接口指针，而非底层图像。

经验法则

在处理接口指针时，请遵循以下经验法则。

接口指针是否是（可显示）图像、容器、迭代器等并不重要。— 它们在样本中可以相互替换。

如果调用接口指针上的方法

必须确保接口指针有效
（检查 <> 0）：

```
IF ipImage <> 0 THEN
    ipImage.GetImageInfo(stImageInfo);
END_IF
```

一般情况下，建议使用替代 F_VN_ 函数（如果存在的话），而不是接口方法，因为内部会自动执行检查：

```
hr := F_VN_GetImageInfo(ipImage, stImageInfo, hr);
```

如果在同一个表达式中执行检查和调用接口方法，则必须使用 AND_THEN 而非 AND。只有在这种情况下，如果前部返回 TRUE，后部才会被执行：

```
IF ipIterator <> 0 AND_THEN ipIterator.CheckIfEnd() <> S_OK THEN
    ipIterator.GetContainer(ADR(ipElement));
END_IF
```

手动复制接口指针

事先确保要写入的接口指针已启用。复制后增加接口指针的引用计数器：

```
FW_SafeRelease(ADR(ipDestImage));
IF ipSrcImage <> 0 THEN
    ipDestImage := ipSrcImage;
    ipSrcImage.TcAddRef();
END_IF
```

在 VAR_INPUT 中声明接口指针

对于函数和方法，什么都不需要观察。对于程序和功能块，变量中的指针会在调用后保留，因此应在 POU 结束时设置为 0：

```
VAR_INPUT
    ipSrcImage: ITcVnImage;
END_VAR
//… functional code
ipSrcImage := 0;
```

这样就能确保下次调用时无法再访问内存区域，因为它可能已经被释放出去。在 POU 中增加引用计数器并不是一种解决方案，因为新分配会覆盖先前指针，因此无法释放，从而导致内存泄漏。如果要保留接口指针，则应将其复制到本地变量。

将接口指针声明为 VAR 中的引用

在函数和方法结束时释放接口指针。结束 POU 后，接口指针不再存在，因此必须先减少引用计数器。对于程序和功能块，一旦不再需要接口指针，就应立即将其释放：

```
VAR
    ipImageWork: ITcVnImage;
END_VAR
//… functional code
hr := FW_SafeRelease(ADR(ipImageWork));
```

不建议 — 在 VAR_OUTPUT 中声明接口指针

不建议在 VAR_OUTPUT 中声明接口指针，因为输出的使用可选，因此可能会发生内存泄漏。
在程序和功能块中，变量中的指针即使在调用后也会保留，这样在外部发生变化时，也可以在功能块中访问已释放内存区域。
为了防止出现这种情况，必须始终确保使用输出，如果是程序和功能块，则必须在复制输出变量后增加引用计数器或将输出变量设置为 0。
这样做非常费力，而且容易出错，因为您必须要注意在代码的每一个点上正确实现。因此，建议将其作为 VAR_INPUT 中的引用进行转移。

注意

内存泄漏

不建议在 VAR_OUTPUT 中使用接口指针。未观察到上述点可能会导致内存泄漏和系统崩溃。

将接口指针声明为 VAR_INPUT 中的引用

如果要在 POU 内将对接口指针的更改（如分配新图像/容器）传递到外部，以便在那里进一步使用，则应使用作为引用的传输。

```
VAR_INPUT
    ipDestImage: REFERENCE TO ITcVnImage;
END_VAR
```

例程 自行编写的函数 [▶ 2814] 解释了在自写函数和方法中接口指针的正确使用。

误用的后果

如果一个接口指针变量被复制，但参考计数器没有同时调整`TcAddRef()`，就会有两个指针；但是，只有其中一个被程序所知。因此，一旦两个接口指针变量中的一个被释放，那么这两个接口指针变量所指向的数据就会被删除。释放可以通过手动执行`FW_SafeRelease`，也可以通过改写变量的 TwinCAT Vision API 功能来实现。如果剩下的接口指针变量再被方法调用访问，系统将**崩溃**，如上所述。

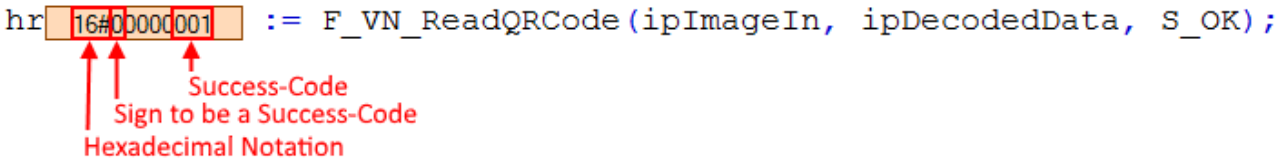
如果与此相反，接口指针变量在没有事先释放的情况下被改写（例如通过`:=` 赋值），这将导致**内存泄漏**。例如，如果接口指针被用作方法变量，并且在结束方法之前没有释放，就会发生这种情况。在 AMS 路由器的信息窗口中可以找到内存泄漏的指示（见路由器内存 [▶ 54]）。如果那里标记为可用的内存随着时间的推移而减少，这表明有内存泄漏。

6.1.1.2 HRESULT

所有 TwinCAT Vision 功能在执行后都会返回一个 HRESULT。它的值表示执行成功与否。

成功代码

成功的执行由正数代码表示。在十六进制的符号中，第一个数字在 0 和 7 之间。



经常出现的成功代码是：

代码	名称	描述
16#000	S_OK	功能已成功执行

代码	名称	描述
16#001	S_FALSE	功能已成功执行，但没有达到完整的结果。(例如，在图像中没有发现代码时，代码读取功能会出现这种情况。)
16#203	S_PENDING	异步方法已经启动，但还没有结果（例如，在第一次调用 fbCameraControl.StartAcquisition() 时出现；例如，在第二次调用 S_OK 返回时出现）
16#256	S_WATCHDOGTIMEOUT	函数被看门狗中止。

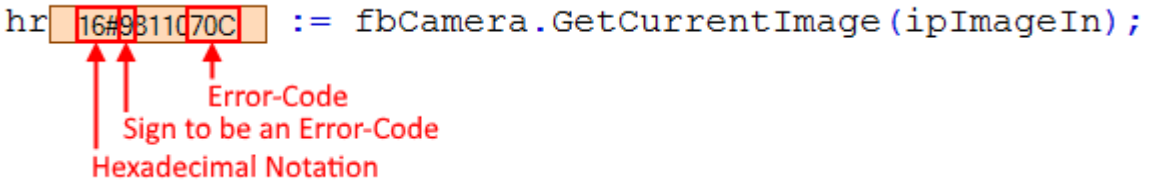
错误代码

失败的执行由负数代码表示，或者在十六进制符号中，第一个数字>=8。代码的最后几个数字对应于ADS 返回代码 [▶ 2857]。

注意

错误代码

如果返回了错误代码，表示功能的所有返回结果无效，且因此不得使用。



TwinCAT Vision API 元素的常见错误代码：

十六进制	十进制	名称	描述
16#70A	1802	NOMEMORY	内存不足
16#70B	1803	INVALIDPARM	参数值无效
16#70C	1804	NOTFOUND	未找到（文件、图像...）
16#70E	1806	INCOMPATIBLE	对象不匹配
16#712	1810	INVALIDSTATE	一个 FB 的方法在不允许的状态下被调用。
16#719	1817	TIMEOUT	超时
16#71A	1818	NOINTERFACE	接口查询失败
16#71B	1819	INVALIDINTERFACE	请求了错误的接口
16#71D	1821	INVALIDOBJID	对象 ID 无效。
16#734	1844	OUTOFRANGE	超出有效范围。

关于完整列表，请参见ADS 返回代码 [▶ 2857]。

应用

如果出现错误，HRESULT为负（错误代码），否则为正（成功代码）。使用FAILED ()和SUCCEEDED ()功能进行检查。

```
PROGRAM MAIN
VAR
    hr      : HRESULT := S_OK;
END_VAR

IF SUCCEEDED(hr) THEN
    //code
END_IF
```

限制性查询

请注意，上述查询仅检查功能是否成功执行，而不是检查是否获得正确或期望的结果。

例如，如果在图像中没有发现或读到代码，则代码读取功能返回成功代码S_FALSE（16#001）。为了检测这种情况，必须将hr与S_OK或S_FALSE直接进行比较。相反，如果待测量的对象从图像中丢失，则测量功能返回NOTFOUND（16#70C）。

大多数 TwinCAT Vision 功能都希望HRESULT作为最后一个输入参数，并且只有这是一个成功代码且从而确认到目前为止没有发生过错误时才会被执行。否则，相应的功能不会被执行，而是返回之前的HRESULT。这确保了，在HRESULT从一个功能传递到另一个功能时，输出的是第一个发生的错误代码，而不是后续的错误。

成功代码

成功代码（如S_FALSE）不被被下一个功能转发和覆盖。如有必要，必须明确检查各功能的 HRESULT 值。

访问接口方法

如果一个功能要创建一个接口指针，该功能的 HRESULT 指示接口指针现在是否存在。此外，必须检查接口指针以确定其是否为零。

⚠ 警告

对无效内存区域的方法访问导致系统崩溃

对于接口方法，内部检查不可行，因为相应的对象必须存在才能调用该方法。

即使在方法调用之外，也要确保相应的对象存在（接口指针不为 0），并且程序还没有返回错误代码。

这是一个安全的查询，以使用一个新创建的接口指针：

```
IF SUCCEEDED(hr) AND ipImage <> 0 THEN
    hr := ipImage.GetWidth(nWidth);
END_IF
```

集成到用户定义的功能块中

HRESULT可以被集成到用户定义的功能块中，如下所示：

```
METHOD MyMethod : HRESULT
VAR_INPUT
    hr : HRESULT
END_VAR

IF FAILED(hr) THEN
    MyMethod := hr; //skip the whole method
ELSE
    (*
        code
        code
        code
    *)
    MyMethod := S_OK; // or some error code if something went wrong
END_IF
```

这在样本自行编写的函数 [▶ 2814]中进行了更详细的解释。

提取 ADS 的返回代码

例如，如果需要在人机界面中显示纯粹的 ADS 返回代码，可以从HRESULT以数字和文本形式提取，如下所示：

```
PROGRAM MAIN
VAR
    hr              :   HRESULT;
    nReturnCode     :   DWORD;
    sReturnCode     :   STRING;
END_VAR

nReturnCode := DINT_TO_DWORD(hr) AND 16#FFF;
sReturnCode := DWORD_TO_HEXSTR(nReturnCode, 3, FALSE);
```

为此需要使用 Tc2_Utilities 库。

6.1.1.3 机器学习

在机器学习中，优化模型以解决一项任务。这个过程被称为训练。在训练模型的帮助下，可以自动进行复杂的数据分析。这意味着手动创建的复杂程序结构可以被取代。机器学习模型在 TwinCAT 3 实时环境中使用所提供的机器学习功能创建、训练和执行。它可以解决的任务多种多样，大致可分为五个任务领域（分类、异常检测、回归分析、聚类分析和特征转换）。根据任务的不同，有不同的模型类型可供选择，每种类型都有一般或特定于任务的优缺点。

在机器学习中，有监督学习和无监督学习之分。监督学习需要额外的信息，如标注（如标记）数据集，以便模型可以学习到已知类别（分类）或所需目标变量的映射。然后，模型可以对未知数据的类别或目标变量做出预测。相比之下，无监督学习无需额外信息，完全基于功能本身。

所有分类和回归模型一般都可归入监督学习。相比之下，所有异常检测和聚类模型都被可归入无监督学习。这两个选项都可用于特征转换，线性判别分析法基于监督学习，主成分分析基于无监督学习。

以下对任务区域进行的模型类型分配有助于选择合适的模型。需要注意的是，此处具体展示的是 TwinCAT Vision 中机器学习的处理，并不具有普遍性。

模型类型	分类	异常检测	回归	聚类	特征转换
提升分类器（提升）	x				
K 均值聚类++ (KMPP)		x		x	
K 最近邻算法 (KNN)	x	x	x		
Linde-Buzo-Gray (LBG)		x		x	
线性判别分析 (LDA)					x
普通贝叶斯分类器 (NBC)	x	x			
主成分分析 (PCA)					x
随机森林 (RTrees)	x		x		
简化 TopoART (STA)	x	x	x	x	
支持向量机 (SVM)	x	x	x		
支持向量机 – 随机梯度下降 (SVM-SGD)	x				

只有使用机器学习组的 `F_VN_Create` 函数之一创建的模型才能使用。可通过功能块 FB_VN_WriteMlModel [▶ 1597] 和 FB_VN_ReadMlModel [▶ 1591] 将以这种方式创建和随后训练的模型保存或加载到硬盘上。不打算导入外部训练模型，如 ONNX 文件。

机器学习的基本程序：

- 分析问题，定义预期结果：
  - 除其他事项外，这还决定了应用范围和可使用的模型类型。
- 收集数据：
  - 训练、验证和测试数据
  - 数据必须包含适合解决问题的信息。
  - 数据必须涵盖以后可能出现的所有变量和功能。
  - 如果是监督学习，还必须对数据进行标注。

- 选择并提取合适的特征：
  - 哪些特征是独特的、特殊的，或可为差异化提供相关信息？
  - 应避免冗余特征。
  - 特征转换在此可能会有所帮助。
  - 最后，应进行功能归一化处理，以便使不同特征的值范围具有可比性，并满足模型的要求。
- 确定合适的模型和超参数（验证和优化）。
  （用确定的超参数在训练和验证数据上训练选定模型）。
- 使用之前未使用的独立数据进行最终测试。

数据必须以样本的形式提供。样本是由特征组成的向量。这些特征既可以是相关图像的有形变量（如颜色值）或轮廓值（如重心形式）等，也可以是抽象变量（如 PCA 的结果），它们以有意义的方式对图像进行统计描述，但用户无法直接解读。对于监督模型的训练，每个样本均需要额外的信息，如类别分配。多个样本的集合称为一个批次。

根据模型类型，可提供用于样本和/或批量训练和预测的函数。有些模型可以重新训练，以便在运行时期间进一步优化模型或使其适应新条件。

特征提取

特征提取涉及从图像数据中提取量化特征，是机器学习解决方案的重要组成部分。必须区分是否可以通过函数直接确定特征，是否需要多个函数组合来确定，或者是否可以利用对象的逻辑特征（如孔数）进行区分。您可以在 特征提取功能 [▶ 139] 章节中找到一份示例清单。

特征归一化

对特征进行缩放以实现归一化主要是为了确保所有特征对模型学习过程的贡献相同。数值较大的特征会在距离计算中占主导地位，使基于距离的模型对这些特征更加敏感。为了避免这种情况并满足模型的要求，我们通常建议执行特征归一化。

函数 F_VN_GetFeatureScales [▶ 1419] 提供三种缩放选项（ETcVnFeatureScalingType [▶ 194]）。您还可以创建和使用自己的缩放比例。使用函数 F_VN_FeatureScaling [▶ 1408] 应用缩放。

相同特征提取和归一化

需要注意的是，所有处理步骤都必须在训练集训练期间和测试集测试模型时进行。因此必须在两个位置执行函数，必要时还必须保存参数以供日后使用。

6.1.1.3.1 特征提取功能

该列表包含适合特征提取的函数。该列表仅为示例，并不详尽。

图像特征

- F_VN_ImageAverage [▶ 1224]
- F_VN_ImageAverageStdDev [▶ 1225]
- F_VN_ImageMoments [▶ 1230]
  - F_VN_HuMomentInvariants [▶ 1523]
- F_VN_CountNonZeroPixels [▶ 1217]
- F_VN_HistogramExp [▶ 1254]
- F_VN_ImageCenterOfMass [▶ 1227]
- F_VN_MaxPixelValue [▶ 1231]
- F_VN_MinPixelValue [▶ 1233]
- F_VN_RegionOrientation [▶ 1235]

轮廓特征

- F_VN_ContourArea [▶ 965]

- F_VN_ContourCircularity [▶ 967]
- F_VN_ContourRoundness [▶ 976]
- F_VN_ContourMoments [▶ 972]
  - F_VN_HuMomentInvariants [▶ 1523]
- F_VN_ContourConvexity [▶ 967]
- F_VN_ContourPerimeter [▶ 975]
- F_VN_ContourOrientation [▶ 973]
- F_VN_ContourCenterOfMass [▶ 966]
- F_VN_ContourEccentricity [▶ 968]
- F_VN_ContourElongation [▶ 969]
- F_VN_ContourExtremePoint [▶ 970]
- F_VN_ContourInertiaRatio [▶ 971]
- F_VN_ConvexHullPoints [▶ 977]
- F_VN_ConvexityDefects [▶ 978]
- F_VN_FourierDescriptors [▶ 987]
- F_VN_EnclosingCircle [▶ 981]
- F_VN_FitEllipse [▶ 984]
- F_VN_EnclosingRectangle [▶ 982]

图像纹理

- F_VN_Granulometry [▶ 1567]
- F_VN_HaralickFeatures [▶ 1568]

关键点

- F_VN_KeyPointsAGAST [▶ 1340]
- F_VN_KeyPointsAndDescriptorsAKAZE [▶ 1342]
- F_VN_KeyPointsAndDescriptorsORB [▶ 1348]
- F_VN_KeyPointsFAST [▶ 1350]
- F_VN_KeyPointsGFTT [▶ 1352]
- F_VN_KeyPointsSB [▶ 1355]

6.1.1.4 看门狗

由于许多图像处理算法的运行时取决于图像内容，因此在不利情况下，函数调用的时间会比平时长很多。

例如，光照条件的变化、图像中出现意外物体以及其他因素都可能导致这种情况，从而极大地影响 F_VN_FindContours() [▶ 1179] 等函数的计算时间。例如，如果在正常条件下只能找到 10 个轮廓，那么在其他照明条件下可能会有 100 个或更多。

这可能导致循环超限，必须不惜一切代价避免，因为这会导致未定义行为。

为此，Watchdog（看门狗）很有帮助。Watchdog（看门狗）可以在一定时间后中止单个 Vision 函数或包含多个函数的整个代码段。某些 TwinCAT Vision 函数可以在执行过程中中止，并返回到那个时候为止计算出的部分结果。如果多个函数受 Watchdog（看门狗）监控，则在指定时间过后，Watchdog（看门狗）范围内的所有剩余函数都会被跳过。不过，随后的函数（Watchdog（看门狗）区域之后）会再次正常执行。

有两种函数可用于定义 Watchdog（看门狗）监控区域的起始，它们在指定的开始计数时间上有所不同：

- F_VN_StartAbsWatchdog [▶ 994] 定义相对于任务循环开始的绝对中止时间
- F_VN_StartRelWatchdog [▶ 996] 定义相对于 Watchdog（看门狗）区域开始的相对中止时间

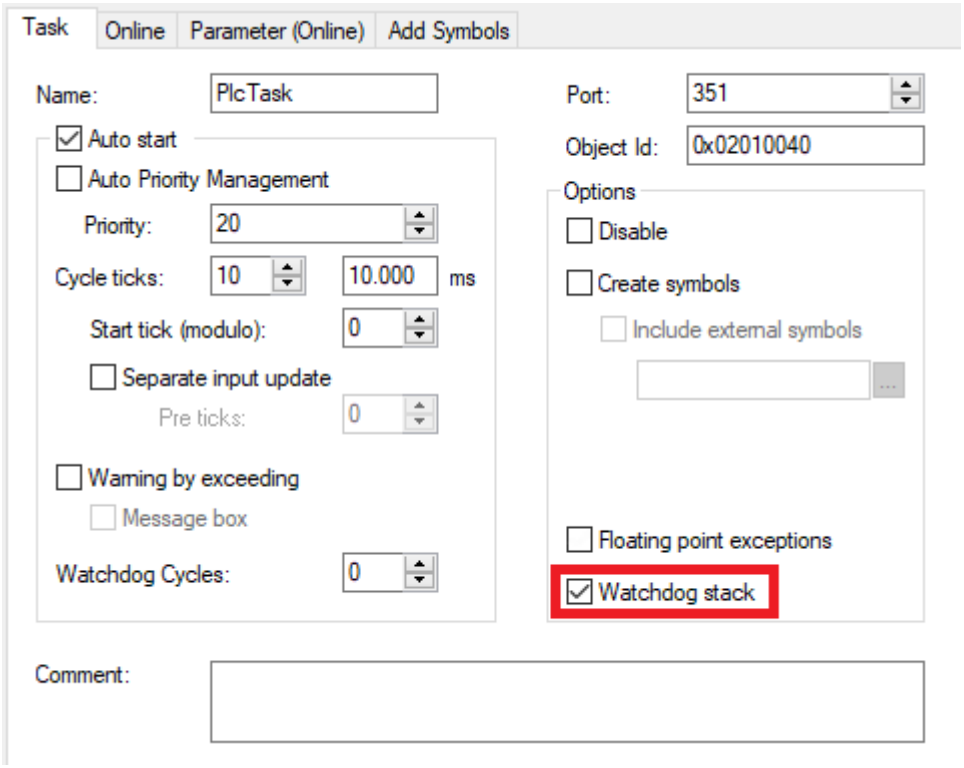
在这两种情况下，区域结束都由函数 F_VN_StopWatchdog [▶ 998] 定义，其可选择返回处理组件和所需时间。

注意

具有相对中止时间的 Watchdog（看门狗）

在使用 F_VN_StartRelWatchdog 时，必须注意中止时间不要设置过长，否则会继续出现循环过冲。因此，要观察相应任务的循环时间、调用函数之前已经耗费的时间以及 F_VN_StopWatchdog 之后到任务结束还需要的时间。

为了让 Watchdog（看门狗）监控函数，必须在相应的执行任务上启用“Watchdog stack（看门狗栈）”选项。



例程

在下方示例中，Watchdog（看门狗）被启动，相对于 Watchdog（看门狗）函数调用的停止时间为 10 ms。例如，如果循环时间为 20 ms，调用 F_VN_StartRelWatchdog [▶ 996] 时当前任务循环时间已过去 4 ms，而 Watchdog（看门狗）启动时的停止时间 tStop 相对于当前时间为 10 ms，则 Watchdog（看门狗）会在当前循环时间过去 14 ms 后停止受监控函数（即 Watchdog（看门狗）启动和停止之间调用的任何函数）。

```
VAR
    ipImage    : ITcVnImage := 0;
    ipContours : ITcVnContainer := 0;

    // watchdog runtime info
    nFunctionsMonitored  : ULINT;
    nFractionProcessed   : UDINT;
    tRest                : DINT;
END_VAR

(* imagine some other functions that use 4ms up to here *)

hr := F_VN_StartRelWatchdog(10000, hr); // 10ms
    hr := F_VN_Threshold(ipImage, ipImage, 120, 255, TCVN_TT_BINARY, hr);
    hr := F_VN_FindContours(ipImage, ipContours, hr);
hr := F_VN_StopWatchdog(nFunctionsMonitored => nFunctionsMonitored, nFractionProcessed => nFractionP
rocessed, tRest => tRest);
```

现在有两种可能情况：

- 两个函数均及时终止 – 这是正常情况。

假设 F_VN_Threshold [▶ 1322] 需要 1 ms，F_VN_FindContours [▶ 1179] 需要 5 ms。当调用 F_VN_StopWatchdog [▶ 998] 时，剩余 4 ms。Watchdog（看门狗）的运行时信息是：

```
nFunctionsMonitored = 2
nFractionProcessed = 100 // in %
tRest = 4000 // in us, equals 4ms
```

- Watchdog（看门狗）必须干预。

情景：函数 F_VN_Threshold [▶ 1322] 与图像内容无关（只与像素数有关），因此只需 1 ms，但照明条件发生了不利变化，因此 F_VN_FindContours 所需的时间将超过 9 ms。因此 Watchdog（看门狗）停止了 F_VN_FindContours [▶ 1179]，但还是返回了目前为止找到的轮廓。Watchdog（看门狗）的运行时信息如下所示：

```
nFunctionsMonitored = 2
nFractionProcessed = 70
tRest = -50
```

在这种情况下，当函数中止时，计算出的估计处理百分比为 70%。剩余时间为负值，即计划的停止时间超出了 50 us，因此在函数 F_VN_StopWatchdog [▶ 998] 之后，从任务循环时间开始已经过去了 14050 us，而不是计划的 14000 us。这种超限是由于要继续使用已经计算过的函数的部分结果。因此，一方面，算法只能在特定点中止，另一方面，必须整理并返回之前的结果。最大超限一般取决于具体函数和图像内容。因此，在程序中，应始终选择终止时间，以便在任务循环结束时保留安全缓冲。

监控函数

以下函数在 Watchdog（看门狗）终止时提供部分结果：

- F_VN_AdaptiveThreshold(Exp) [▶ 1316]
- F_VN_ConvertColorSpace [▶ 1248]
- F_VN_DetectBlobs(Exp) [▶ 1171]
- F_VN_FindContourHierarchyExp [▶ 1175]
- F_VN_FindContours(Exp) [▶ 1179]
- F_VN_GradeDataMatrixCode(Exp) [▶ 829]
- F_VN_GradeQRCode(Exp) [▶ 834]
- F_VN_HoughCircle(Exp) [▶ 1194]
- F_VN_ImageAbsoluteAverage [▶ 1237]
- F_VN_LocateCircularArc(Exp) [▶ 1464]
- F_VN_LocateEdge(Exp) [▶ 1470]
- F_VN_LocateEdges(Exp) [▶ 1476]
- F_VN_LocateEllipse(Exp) [▶ 1485]
- F_VN_MatchTemplate(Exp) [▶ 1201]
- F_VN_MatchTemplateAndEvaluate(Exp) [▶ 1203]
- F_VN_MeasureAngleBetweenEdges(Exp) [▶ 1490]
- F_VN_MeasureEdgeDistance(Exp) [▶ 1496]
- F_VN_MeasureMinEdgeDistance(Exp) [▶ 1505]
- F_VN_NormalizeImageForDisplay [▶ 1259]
- F_VN_ReadDataMatrixCode(Exp) [▶ 850]
- F_VN_ReadDataMatrixCode(Exp(2)) [▶ 853]
- F_VN_ReferenceColorSimilarity(Exp)_ITcVnColorModel (_ITcVnMlModel) [▶ 1260]
- F_VN_ReferenceColorSimilarity(Exp)_TcVnVector3_LREAL [▶ 1262]
- F_VN_ResizeImage(Exp) [▶ 1153]
- F_VN_RotateImageExp(2) [▶ 1158]
- F_VN_Threshold [▶ 1322]

- F_VN_TrainImageColor(Exp(2)) [▶ 1265]
- F_VN_TrainImageColor(Exp(2))_(ITcVnMlModel) [▶ 1275]
- F_VN_UprightInnerRectangle(Exp) [▶ 1239]
- F_VN_WarpAffine(Exp) [▶ 1159]
- F_VN_WarpPerspective(Exp) [▶ 1164]

其他函数在执行过程中无法中止，但如果 Watchdog（看门狗）时间已过，则会跳过。

示例

- 带看门狗监控的斑点检测 [▶ 2755]

6.1.1.5 图像处理

在以下各章中，将参照 TwinCAT Vision 对图像处理的一些概念进行一般和具体说明。它们将帮助你了解许多 API 功能的操作模式。

6.1.1.5.1 图像

在 TwinCAT Vision 中，图像由图像对象处理，这些对象可以实现各种接口，如ITcVnImage [▶ 407]。通过接口指针可以访问图像对象的实际图像数据，然后在 PLC 中用于处理。这些图像可以由各种像素类型和一些通道组成，并且可以有几乎任意的尺寸。由于处理来自实时和动态的大小，图像数据位于路由器内存中。相机图像在收到后直接存储在那里，而来自硬盘的图像必须首先通过文件源或 PLC 功能块加载，并以这种方式最终进入路由器内存。相机对象中的助手对路由器内存的图像不起作用。

查询一个视觉设备的图像接口

在许多情况下，图像通过相机或通过加载一个图像文件创建。以下功能块可用于此目的：

- FB_VN_SimpleCameraControl [▶ 1630]
  可以连接到 GigE Vision 相机实例或文件源实例。
- FB_VN_GevCameraControl [▶ 1610]
  连接到 GigE Vision 相机实例。
- FB_VN_FileSourceControl [▶ 1601]
  连接到文件源实例。

根据视觉设备的设置，可能需要启动过程或图像触发。此处仅显示 PLC（通过它可以访问内存中的图像）中接口指针的查询：

```
VAR
    hr: HRESULT;
    fbVisionDevice: FB_VN_SimpleCameraControl;
END_VAR

hr := fbVisionDevice.GetCurrentImage(ipImageIn);
IF SUCCEEDED(hr) AND ipImageIn <> 0 THEN
    // Process image…
END_IF
```

图像格式

最常用的两种图像格式是单色和 RGB。在相机中，传输的图像格式在`PixelFormat`选项卡下设置。为了减少数据量，从而能够实现更高的传输率，也有打包或过滤的图像格式，如 Bayer 格式。由于这些传输的图像格式不能使用，或者由于过滤器的原因与原始图像不一致，所以在收到后应直接转换这些格式。

- 如果是打包的图像，当调用`GetCurrentImage()`时转换将在内部进行，然后会直接得到解包的图像。
- 在所有其他情况下，就像使用 Bayer 格式一样，应该先用F_VN_ConvertColorSpace [▶ 1248]转换图像。之后，输入的图像不再需要，且可以丢弃。

这一点在使用插值图像信息的函数（如F_VN_CompensateLensDistortion [▶ 1089]、F_VN_WarpAffine [▶ 1159]、F_VN_RotateImageExp [▶ 1157]或诸如受过滤图像干扰的校准函数）时尤其需要注意。因此，一般建议首先将图像读入 PLC，在那里进行转换，然后将所有后续功能应用于原始图像。

创建图像

在有些情况下，需要的图像不是由相机拍摄且也不是从文件系统加载。例如，可以创建一个彩色图像来显示结果。通过指定图像尺寸、像素类型和通道数量，可以为此使用函数F_VN_CreateImage [▶ 784]。这里选择的像素类型TCVN_ET_USINT对应于 8 位/通道的常规图像。请注意，图像所需的内存只被分配，但没有被初始化。因此，可以通过函数F_VN_SetPixels [▶ 812]将图像设置为统一的颜色。

最大图像尺寸

可以创建、接收或处理的最大图像尺寸限制在行或列的 2^20 像素数或图像的总尺寸 2^30 像素，以先到者为准。

```
PROGRAM MAIN
VAR
    ipImage     :   ITcVnImage;
    aColorBlack :   TcVnVector4_LREAL := [0, 0, 0];
    hr          :   HRESULT;
END_VAR

hr := F_VN_CreateImage(
    ipImage     :=  ipImage,
    nWidth      :=  640,
    nHeight     :=  480,
    ePixelType  :=  TCVN_ET_USINT,
    nChannelNum :=  3,
    hrPrev      :=  hr
);
hr := F_VN_SetPixels(ipImage, aColorBlack, hr);
```

或者，可以使用F_VN_CreateImageFromArray [▶ 786]函数从现有的图像数据中创建一个图像。

```
PROGRAM MAIN
VAR
    aImageData  :   ARRAY [0..479, 0..639] OF USINT;
    ipImage     :   ITcVnImage;
    hr          :   HRESULT;
END_VAR

hr := F_VN_CreateImageFromArray(
    pData       :=  ADR(aImageData),
    ipImage     :=  ipImage,
    nWidth      :=  640,
    nHeight     :=  480,
    ePixelType  :=  TCVN_ET_USINT,
    nChannelNum :=  1,
    hrPrev      :=  hr
);
```

访问图像数据

函数F_VN_GetPixel [▶ 803]和F_VN_SetPixel [▶ 811]可用于访问图像的单个像素。这种方法很容易使用。然而，当重复应用于许多像素时，它的运行时间相对较长。在这种情况下，直接访问图像数据阵列比较合适。一个完整的像素行的指针被获取，且数组算子被用来访问各个像素：

```
PROGRAM MAIN
VAR
    ipImage     :   ITcVnImage;
    nHeight     :   UDINT;
    nWidth      :   UDINT;
    y           :   UDINT;
    x           :   UDINT;
    // Pointer type must match pixel type of ipImage!
    pRow        :   POINTER TO USINT;
    nPixelValue :   USINT;
END_VAR

// Create ipImage here or set to an existing image

// Determine width and height of image
hr := F_VN_GetImageWidth(ipImage, nWidth, hr);
hr := F_VN_GetImageHeight(ipImage, nHeight, hr);

// Iterate every pixel of the image
FOR y := 0 TO nHeight - 1 DO
    IF hr = S_OK THEN
        hr := ipImage.GetRowPointer(y, ADR(pRow));
        IF hr = S_OK THEN
            FOR x := 0 TO nWidth - 1 DO
                // Access pixel via pRow[x]
                nPixelValue := pRow[x]; // e.g.
                pRow[x] := nPixelValue; // e.g.
            END_FOR
        END_IF
        // It's important to release the pointer. Otherwise there will be memory leaks.
```

```
        hr := ipImage.ReleaseRowPointer(ADR(pRow));
    END_IF
END_FOR
```

可显示的图像

为了能够通过 ADS 传输和显示图像（例如在ADS 图像查看 [▶ 128]中），必须首先将其转换为类型为 ITcVnDisplayableImage [▶ 407]的可显示图像。

严格区分显示和处理图像是出于以下原因：通过 ADS 的图像传输与 PLC 周期不同步。因此，图像处理功能可以在图像传输的同时进行。为了避免图像传输过程中的内存冲突，因此有必要将内存区域与要显示和处理的图像分开。

两个接口ITcVnImage [▶ 407]和ITcVnDisplayableImage [▶ 407]可防止意外目的的图像访问（无论是处理还是传输）。以下函数可用于转换为可显示图像（ITcVnDisplayableImage [▶ 407]）：

- F_VN_CopyIntoDisplayableImage [▶ 781]
- F_VN_TransformIntoDisplayableImage [▶ 818]
- F_VN_TransformIntoDisplayableImageExp [▶ 819]

F_VN_CopyIntoDisplayableImage [▶ 781]函数创建了图像数据的深度拷贝，并使其成为可显示的图像。如果待显示图像将用于下面的程序序列，务必使用功能：

```
// Processing, e.g.:
hr := F_VN_Threshold(ipImage, ipImage, 128, 255, TCVN_TT_BINARY, hr);

// Show intermediate processing step
hr := F_VN_CopyIntoDisplayableImage(ipImage, ipImageDisp, hr);

// Further processing, e.g.:
hr := F_VN_FindContours(ipImage, ipContours, hr);
```

函数F_VN_TransformIntoDisplayableImage [▶ 818]使用现有的图像数据并释放图像的原始接口指针。这意味着图像数据不能再被处理，且完全可用于图像传输。由于没有创建图像数据的副本，该功能的执行时间要短得多，而且没有使用额外的内存。然而，这个功能仅在没有其他指向图像数据的指针时才起作用。否则，将返回错误代码E_INCOMPATIBLE (16#70E)。如果Ads 通信器对象 [▶ 121]激活（例如数据流记录 [▶ 102]），就会出现这种情况，因为这将在短时间内继续在内部引用该图像。因此，这个功能主要对处理链的末端有用，以（例如）显示已经生成的结果图像。

```
// Put last results on image, e.g.:
hr := F_VN_DrawContours(ipContours, -1, ipImageRes, aColorRed, 5, hr);

// Finally transform result image into a displayable image
hr := F_VN_TransformIntoDisplayableImage(ipImageRes, ipImageResDisp, hr);
```

此外，函数F_VN_TransformIntoDisplayableImageExp [▶ 819]提供了按需创建原始图像深度拷贝的选项。因此，如果可能，会使用节省资源的转换变体，而必要时则使用复制变体。如果GigE Vision 相机 [▶ 71]实例的 AdsCommunicator 对象 [▶ 120]被激活以将图像作为数据流保存到文件系统中，就会出现相应的用例。除了 PLC 中的指针外，AdsCommunicator 对象中还有一个图像数据指针。如果在使用该图像时 PLC 中尚未启用该功能，则必须复制该图像进行显示。

```
hr := fbCamera.GetCurrentImage(ipImageIn);
hr := F_VN_CopyImage(ipImageIn, ipImageWork, hr);

// Reliably show original image and try to reuse image data to avoid deep copy
hr := F_VN_TransformIntoDisplayableImageExp(
    ipSrcImage      :=  ipImageIn,
    ipDestImage     :=  ipImageInDisp,
    bAllowDeepCopy := TRUE,
    hrPrev          :=  hr
);

// Manipulate ipImageWork during processing here ...
```

可显示图像的内存限制

在开发环境中显示的图像（ADS 图像查看或相机助手）与各自的 Visual Studio 实例位于同一 RAM 区域。如果现在希望查看非常多的图像或大尺寸的图像，由于每个实例的最大可用内存，可能会出现瓶颈。

由于图像是从 PLC 中检索出来的，以便在 ADS 图像查看（ADS 通信）中显示，而且这种交换通过路由器内存进行，因此在这一点上必须有足够的内存空间。

元信息

图像的元信息可以通过函数F_VN_GetImageInfo [▶ 800]获得。作为使用该函数的结果，会返回一个类型为TcVnImageInfo [▶ 225]的结构。其中包含的信息可以用于（例如）确保图像具有预期格式，以便进行转换：

```
hr := F_VN_GetImageInfo(ipImageIn, stImageInfo, hr);
IF stImageInfo.stPixelFormat.nChannels = 3 THEN
    hr := F_VN_ConvertColorSpace(ipImageIn, ipImageIn, TCVN_CST_RGB_TO_GRAY, hr);
END_IF
```

6.1.1.5.2 容器

图像处理的一个典型任务是寻找图像中的物体并确定其形状、位置和大小。所发现的物体可以通过其轮廓点等进行描述。轮廓点的数量取决于具体的对象，所以变化很大。因此，使用了动态尺寸的容器来存储这些轮廓。

然而，容器并不限于存储轮廓点，也可以包含各种其他数据类型。在一个容器内，不同的数据类型不能混合。下面解释了处理容器的一些基本概念。

创建容器

容器可以通过单个函数调用创建。仅须定义元素类型（见可用类型 [▶ 152] ）和初始数量：

```
PROGRAM MAIN
VAR
    hr           : HRESULT;
    ipContainer  : ITcVnContainer;
END_VAR

hr := F_VN_CreateContainer(ipContainer, ContainerType_Vector_REAL, 10, hr);
```

在这个示例中，创建了包含 10 个REAL类型元素的新容器，内部组织形式 C++ 矢量。另外，函数F_VN_CopyContainer [▶ 749]可用于创建现有容器的深度拷贝。不仅复制数据的指针，就像分配给另一个变量一样，而且数据被复制到一个新分配的内存区域。

所有可用的容器类型都存储为全局常量 [▶ 152]，并按照以下模式命名：所有名称都以“ContainerType_”开头。接下来是类型，由实际容器类型的描述（基于 C++ 容器）和元素类型（如：REAL、UDINT等等）组成。容器也可以反过来包含更多的容器作为元素。容器结构的每一层在名称中都由各自的容器类型表示，例如“Vector_”或“String_”。因此，带有REAL元素的容器的结果类型名称是ContainerType_Vector_REAL。

添加元素

由于容器能够动态地扩展分配的内存，因此可以在现有的容器中添加额外的元素。函数F_VN_AppendToContainer [▶ 545]和F_VN_InsertIntoContainer [▶ 677]可用于此。

然而，由于内部可能需要分配一个全新的内存区域并复制现有的数据，因此建议事先保留所需的最大内存。因此，新的内存只分配一次，且程序的整体性能更高。为此存在一个函数，即F_VN_ReserveContainerMemory [▶ 768]，它只是保留内存，但不改变容器中元素的数量。

```
// Reserve memory for 100 elements
hr := F_VN_ReserveContainerMemory(ipContainer, 100, hr);

// Append the value 1.23 to the end of the container
hr := F_VN_AppendToContainer_REAL(1.23, ipContainer, hr);

// Insert the value 4.56 into the container at index 2 (3rd element)
hr := F_VN_InsertIntoContainer_REAL(4.56, ipContainer, 2, hr);
```

访问元素

如果仅需要访问容器的几个单独元素，建议使用 F_VN_GetAt_... 或 F_VN_SetAt_... 函数。由于在 PLC 中不可能有功能名过载，所以每个支持的容器类型都有一个单独的功能名。容器中的第一个元素的索引是 0。

```
PROGRAM MAIN
VAR
    hr           : HRESULT;
    ipContainer  : ITcVnContainer;
    fElement     : REAL;
END_VAR

hr := F_VN_GetAt_REAL(ipContainer, fElement, 0, hr); // Get first element
hr := F_VN_SetAt_REAL(fElement, ipContainer, 2, hr); // Set third element
```

使用这些现成功能的好处是，只需要一行代码，且所有的检查（例如，容器类型是否正确以及索引是否过大）都在内部进行。如果想要一个接一个地访问容器中的所有元素，特别是对于大型容器，建议通过迭代器和相关访问接口访问容器元素，以便获得更好的性能：

```
PROGRAM MAIN
VAR
    hr           :   HRESULT;
    ipContainer  :   ITcVnContainer;
    ipIterator   :   ITcVnForwardIterator;
    ipAccess     :   ITcVnAccess_REAL;
    fElement     :   REAL;
END_VAR

hr := F_VN_GetForwardIterator(ipContainer, ipIterator, hr);
IF SUCCEEDED(hr) AND ipIterator <> 0 THEN
    hr := ipIterator.TcQueryInterface(IID_ITcVnAccess_REAL, ADR(ipAccess));
    IF SUCCEEDED(hr) AND ipAccess <> 0 THEN
        WHILE SUCCEEDED(hr) AND ipIterator.CheckIfEnd() <> S_OK DO
            hr := ipAccess.Get(fElement);
            IF SUCCEEDED(hr) THEN
                fElement := fElement + 1;
                hr := ipAccess.Set(fElement);
            END_IF
            IF SUCCEEDED(hr) THEN
                hr := ipIterator.Increment();
            END_IF
        END_WHILE
    END_IF
    hr := FW_SafeRelease(ADR(ipAccess));
END_IF
hr := FW_SafeRelease(ADR(ipIterator));
```

如果容器中的元素也是容器，请使用以下程序代替。

容器的容器

例如，函数F_VN_FindContours [▶ 1179]不仅可以找到一个轮廓，而且可以找到多个轮廓，所以容器可以包含进一步的容器作为元素（在这种情况下，所有内部容器必须包含相同类型的元素）。原则上，这些可以按照上述简单容器完全相同的方式使用。为此，提供了函数F_VN_GetAt_ITcVnContainer [▶ 644]，用于从 PLC 获得任何可使用的内部容器的深度拷贝（从技术上讲，不可能获得原始数据的接口指针）。由于这一层的一般 ITcVnContainer 数据类型，所以无需区分内部容器中包含哪些基本元素类型。这简化了通过迭代器的访问，因为不再需要特殊的访问接口：

```
PROGRAM MAIN
VAR
    hr           : HRESULT;
    ipContainer  : ITcVnContainer;
    ipIterator   : ITcVnForwardIterator;
    ipElement    : ITcVnContainer;
END_VAR

hr := F_VN_GetForwardIterator(ipContainer, ipIterator, hr);
IF SUCCEEDED(hr) AND ipIterator <> 0 THEN
    WHILE SUCCEEDED(hr) AND ipIterator.CheckIfEnd() <> S_OK DO
        // ipElement gets a deep copy!
        hr := F_VN_GetContainer(ipIterator, ipElement, hr);
        IF SUCCEEDED(hr) AND ipElement <> 0 THEN
            // 1. extract information or manipulate ipElement
            // 2. write back changes if ipElement was manipulated
            hr := F_VN_SetContainer(ipIterator, ipElement, hr);
        END_IF
        hr := F_VN_IncrementIterator(ipIterator, hr);
    END_WHILE
    // more efficient to release outside loop, GetContainer handles it inside
    hr := FW_SafeRelease(ADR(ipElement));
END_IF
hr := FW_SafeRelease(ADR(ipIterator));
```

显示容器

目前，如果不采取进一步行动，容器内容不能在 Visual Studio 实时调试中显示。相反，数据可以被导出到一个数组中，并以这种方式查看。

```
VAR
    aArray      :   ARRAY [0..9] OF REAL;
    ipContainer :   ITcVnContainer;
    nBufferSize :   ULINT;
END_VAR

hr := F_VN_ExportContainerSize(ipContainer, nBufferSize, hr);
IF nBufferSize = SIZEOF(aArray) THEN
    hr := F_VN_ExportContainer(
```

```
        ipContainer :=ipContainer,
        pBuffer     :=ADR(aArray),
        nBufferSize :=nBufferSize,
        hr
    );
END_IF
```

覆写容器

一般来说，具有 API 功能的容器可以简单地覆盖。然而，目前，如果现有容器和待写入容器类型不同，则无法覆盖。这是由错误代码 70E（INCOMPATIBLE）表示。

6.1.1.5.3 轮廓

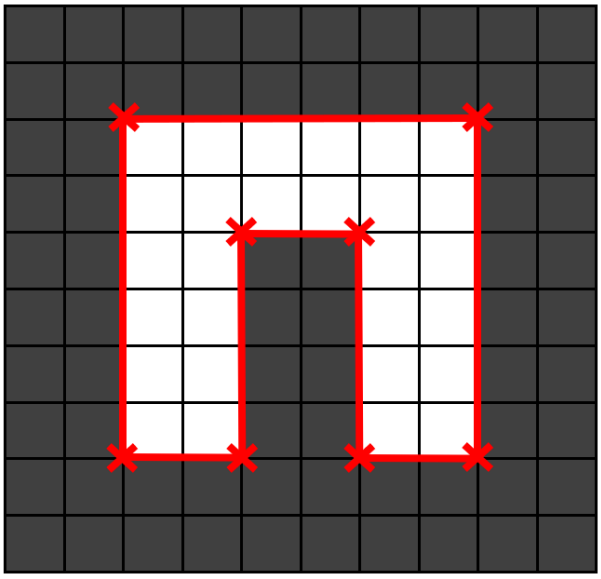
本章介绍了 TwinCAT Vision 中轮廓的使用。

什么是轮廓？

一般来说，轮廓（也叫轮廓线）是一个物体与周围环境的分界线；这在图像处理中是典型的二维处理。轮廓描述了这个物体的形状、大小和位置。在图像处理的范围内，这意味着图像 [▶ 143]中的一个物体；因此，轮廓描述了图像中物体的轮廓线。

轮廓如何显示？

轮廓显示为二维点的集合。如果没有设置 ROI，这些点大多指的是物体被映射的图像的原点。如果把这个领域的所有点都用直线连接起来，就会得到轮廓的图纸。



TwinCAT Vision 中如何显示轮廓？

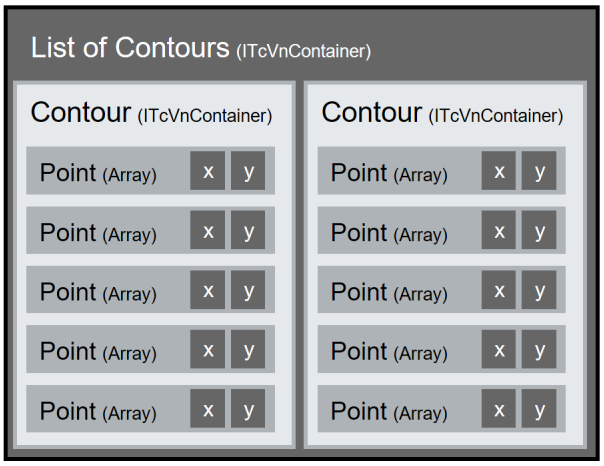
在 TwinCAT Vision 中，动态二维点阵列被处理为点（`TcVnPoint2_DINT/REAL`）的容器 [▶ 146]（`ITcVnContainer`）。然后，相应容器的类型名称为`ContainerType_Vector_TcVnPoint2_DINT/REAL`。

然后，这些点经常直接位于像素网格上（例如在轮廓搜索的情况下）。在这种情况下，`DINT`元素用于保存这些点。然而，也有一些情况下，精度不够，因此必须使用`REAL`（例如在测量的情况下）。这样，这些点就不完全位于一个像素的中心点，而是在其他地方。

绘图功能只能在图像中绘制具有整数元素的轮廓。因此，在有疑问的情况下，必须对容器的元素类型进行转换。

轮廓领域

经常出现的情况是需要同时处理几个轮廓（例如，通常通过F_VN_FindContours [▶ 1179]找到几个轮廓）。这在技术上通过使用二维容器实现。因此，存在轮廓阵列或二维点阵列。然后，相应的类型名称是 `ContainerType_Vector_Vector_TcVnPoint2_DINT/REAL`。



什么是轮廓近似？

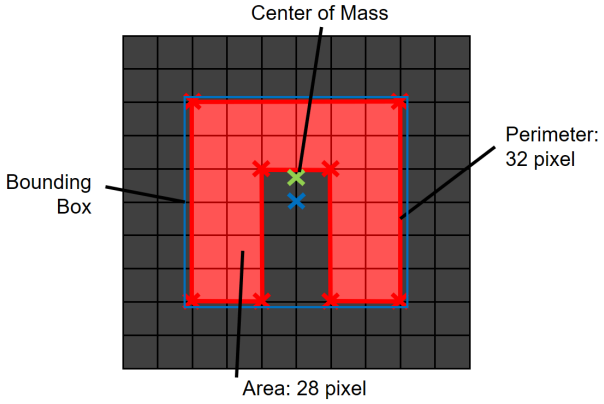
一般来说，在显示一个物体时，总是可以使用非常多的点。然而，这经常并不必要，因为（近似）直线可以通过其起点和终点进行近似。两者之间的点不需要另外保存，因为起点和终点足以定义。

根据所使用的方法和物体的几何形状，近似可能或多或少有所损失。在任何情况下，在不同的近似情况下，几何特征也可以用稍微不同的值来计算。

几何特征

在轮廓上，即从点的阵列中，可以计算出大量单独的特征，而这些特征从几何上描述各自的轮廓。轮廓和相应物体可以被检查并通过这些特征进行分类。这些特征包括：

- 范围
- 表面
- 中心点



封闭和开放的轮廓

轮廓通常封闭。在这种情况下，阵列的第一个和最后一个元素被简单地认为相连。然而，也可能存在开放的轮廓。因此，在技术表现上，开放和封闭轮廓没有区别，而区别在于数据如何解释的相关信息。在 API 的相关点上可相应地查询到各自的轮廓开放或封闭使用。

样本

- 对象检测 [▶ 2755]
- 轮廓分析 [▶ 2747]

6.1.1.5.4 掩码

本章介绍了 TwinCAT Vision 中**掩码**含义。

什么是掩码？

在图像处理方面，掩码是指属于另一个图像 [▶ 143]的掩码图像。掩膜图像具有相同的尺寸，并由一个灰度通道组成。掩膜图像的强度定义了相关图像的哪些像素在某些工作步骤中要被考虑在内。

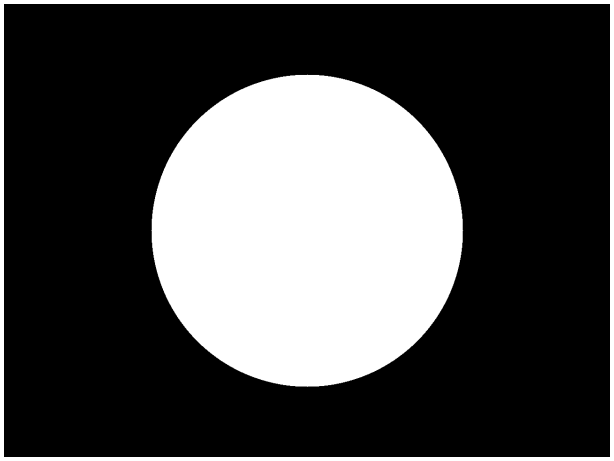
需要掩码做什么？

如果没有进一步的行动，图像将限制为矩形形状。如果在图像操作过程中需要定义矩形以外的形状，可以通过掩码的方式实现。掩码由另一个具有相同形状和大小的图像显示，上面需要考虑的区域（像素）通过不同方式标记出来。

从技术上讲，掩码如何显示？

掩码灰度图像的强度描述了哪些像素需要考虑。原则上，对于强度的解释可能因功能不同而有所偏差。从两方面看，所有数值为 0 的像素通常被解释为不被考虑，而所有不同数值（>=1）的像素被解释为需要考虑。然而，也可以想象到一些情况，其中考虑的程度可以设定得更细。

例如，一个将观察区域限制在图像中的一个圆形区域的掩码如下所示：



当应用该掩码时，相关图像角落里的像素对后续操作没有进一步影响。

样本

举例来说，为图像`ipImageWork`创建一个掩码，也就是说只考虑图像的一个圆形区域：

```
hr := F_VN_GetImageWidth(ipImageWork, nWidth, hr);
hr := F_VN_GetImageHeight(ipImageWork, nHeight, hr);

hr := F_VN_CreateImage(ipImageMask, nWidth, nHeight, TCVN_ET_USINT, 1, hr);
hr := F_VN_SetPixels(ipImageMask, aColorBlack, hr);
hr := F_VN_DrawCircle(nWidth/2, nHeight/2, MIN(nWidth, nHeight)/2, ipImageMask, aColorWhite, -1,
hr);
```

随后计算出掩码区域内图像的平均强度值：

```
hr := F_VN_ImageAverageExp(ipImageWork, aAverage, ipImageMask, hr);
fAverageIntensityInCircle := aAverage[0];
```

如果没有掩码，就不可能在不采取进一步行动的情况下将平均值限制在圆形区域而不是完整的矩形图像上。

关于完整的样本，可查看：任何类型的形状的平均强度 [▶ 2776]。

6.1.1.5.5 感兴趣区域

本章介绍如何在 TwinCAT Vision 中使用**感兴趣区域**。

什么是感兴趣区域？

感兴趣区域 (ROI) 是一个图像暂时被限制在其中的矩形图像区域。图像本身并没有改变，但需要处理的部分尺寸缩小。可以随时将 ROI 重置为完整图像。

为何需要感兴趣区域？

现有图像往往大于需要处理的区域。在这种情况下，应设定 ROI，以获得以下优势：

1. 更好的性能：如果需要处理的图像区域变小，则图像处理时间更短。
2. 减少干扰：ROI 外的干扰图像对象会被排除在处理过程之外，因此不会影响结果。

通过在图像中的不同点逐个设置 ROI，同样可以通过使用 ROI 分别处理多个图像区域。

注意

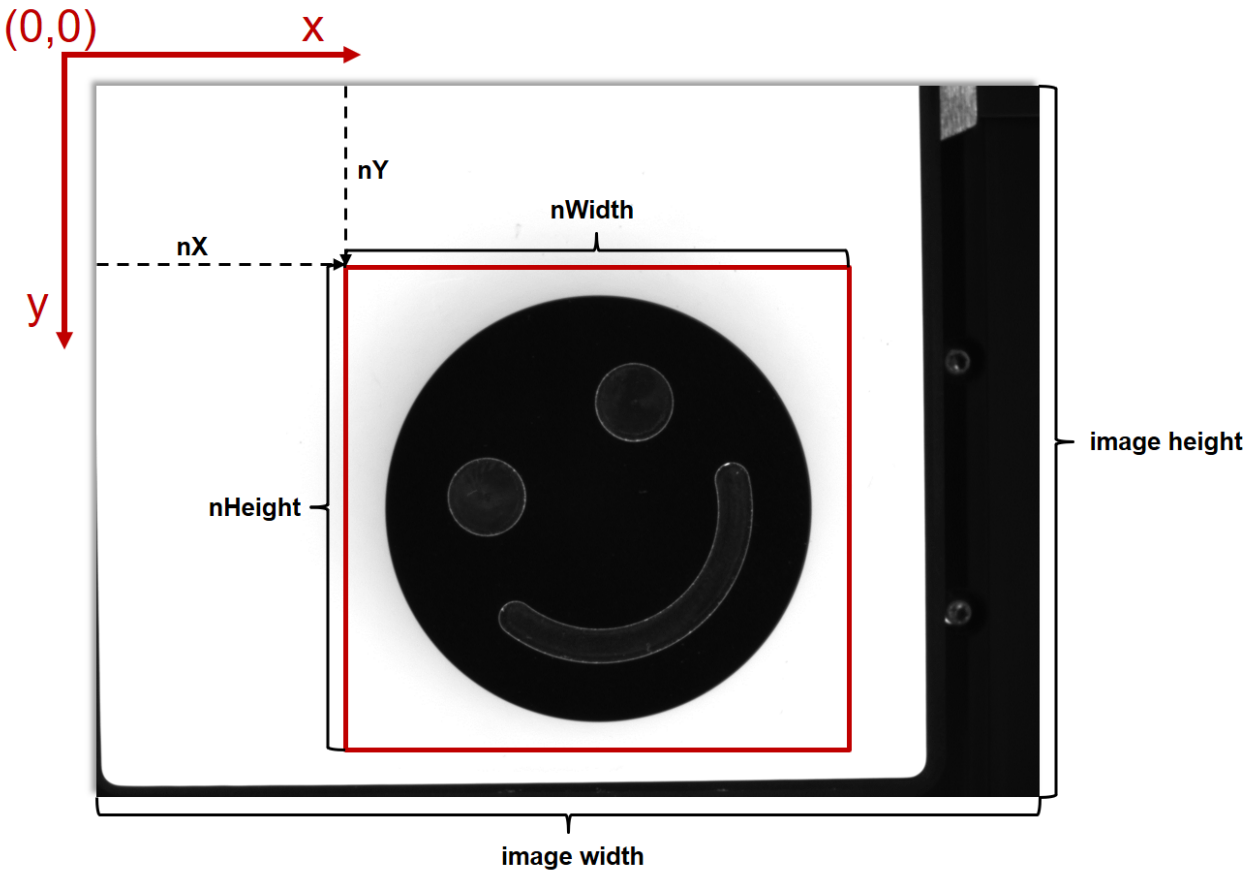
在相机中进行 ROI 配置

如果使用相机作为图像源，最好在其中执行 ROI，以减少需要传输的数据量，并使其尽可能低。根据设置和传感器的不同，可能帧率也会因此提高。请参见相机配置示例：感兴趣区域（ROI） [▶ 2828]

技术上如何显示感兴趣区域？

在 TwinCAT Vision 中，一组 ROI 的相关信息直接存储在相应的图像对象中。ROI 的设置仅包括矩形描述的设置。该区域之外的图像数据仍然存在，但在调用函数时会被忽略。

矩形描述由左上角的位置（`nX` 和`nY`）和矩形的大小（`nWidth` 和 `nHeight`）组成。确保矩形不超出图片边缘（`nWidth ≤ ImageWidth - nX` 和 `nHeight ≤ ImageHeight - nY`）。



要设置 ROI，可以直接传输这些参数，也可以作为 TcVnRectangle_UDINT [▶ 236] 类型的结构进行传输，具体取决于函数：

- F_VN_SetRoi [▶ 814]
- F_VN_SetRoi_TcVnRectangle_UDINT [▶ 816]

函数 F_VN_GetRoi [▶ 805] 可用于检查图像上是否设置了 ROI 以及设置了哪个 ROI。也可以在已经有 ROI 的图像上设置 ROI。如果要将 ROI 放大或重置为完整图像大小，可以使用函数 F_VN_ResetRoi [▶ 810]。

6.1.2 数据类型

共有四种数据类型，每种类型的前缀都是“TcVn”：

- 某些数组类型的别名 [▶ 152]涉及图像处理
- 具有特定数据类型组合和定义数的数组 [▶ 153]经常被图像处理函数使用。
- 枚举 [▶ 154]有助于函数的参数设置，并包含附加前缀“E”
- 结构 [▶ 217]整合多个相关信息片段（如几何描述）

此外，还提供用于描述容器类型的常量 [▶ 152]。

6.1.2.1 别称

名称	类型
GVCP_REGISTER_ADDRESS	UDINT
GVCP_REGISTER_VALUE	UDINT
TcVnPoint	TcVnPoint2_DINT [▶ 152]
TcVnPoint2_DINT	TcVnVector2_DINT [▶ 153]
TcVnPoint2_LREAL	TcVnVector2_LREAL [▶ 153]
TcVnPoint2_REAL	TcVnVector2_REAL [▶ 153]
TcVnPoint3_LREAL	TcVnVector3_LREAL [▶ 153]
TcVnPoint3_REAL	TcVnVector3_REAL [▶ 153]
TcVnRectangle	TcVnRectangle_DINT [▶ 236]

6.1.2.2 常量

以下常量代表了容器 [▶ 146]类型为GUID。对于F_VN_CreateContainer [▶ 752]等函数，需要这些常量。

名称	类型
ContainerType_Vector_SINT	GUID
ContainerType_Vector_USINT	GUID
ContainerType_Vector_INT	GUID
ContainerType_Vector_UINT	GUID
ContainerType_Vector_DINT	GUID
ContainerType_Vector_UDINT	GUID
ContainerType_Vector_ULINT	GUID
ContainerType_Vector_REAL	GUID
ContainerType_Vector_LREAL	GUID
ContainerType_Vector_ITcVnImage	GUID
ContainerType_Vector_TcVnPoint2_DINT	GUID
ContainerType_Vector_TcVnPoint2_REAL	GUID
ContainerType_Vector_TcVnPoint2_LREAL	GUID
ContainerType_Vector_TcVnPoint3_REAL	GUID
ContainerType_Vector_TcVnPoint3_LREAL	GUID
ContainerType_Vector_Vector_TcVnPoint2_DINT	GUID
ContainerType_Vector_Vector_TcVnPoint2_REAL	GUID

名称	类型
ContainerType_Vector_Vector_TcVnPoint3_REAL	GUID
ContainerType_Vector_Vector_UDINT	GUID
ContainerType_Vector_Vector_ULINT	GUID
ContainerType_Vector_Vector_REAL	GUID
ContainerType_Vector_Vector_LREAL	GUID
ContainerType_Vector_TcVnVector2_SINT	GUID
ContainerType_Vector_TcVnVector2_USINT	GUID
ContainerType_Vector_TcVnVector2_INT	GUID
ContainerType_Vector_TcVnVector2_UINT	GUID
ContainerType_Vector_TcVnVector2_DINT	GUID
ContainerType_Vector_TcVnVector2_REAL	GUID
ContainerType_Vector_TcVnVector3_SINT	GUID
ContainerType_Vector_TcVnVector3_USINT	GUID
ContainerType_Vector_TcVnVector3_INT	GUID
ContainerType_Vector_TcVnVector3_UINT	GUID
ContainerType_Vector_TcVnVector3_REAL	GUID
ContainerType_Vector_TcVnVector4_SINT	GUID
ContainerType_Vector_TcVnVector4_USINT	GUID
ContainerType_Vector_TcVnVector4_INT	GUID
ContainerType_Vector_TcVnVector4_UINT	GUID
ContainerType_Vector_TcVnVector4_DINT	GUID
ContainerType_Vector_TcVnVector4_REAL	GUID
ContainerType_Vector_TcVnVector4_LREAL	GUID
ContainerType_Vector_TcVnRectangle_DINT	GUID
ContainerType_Vector_TcVnRotatedRectangle	GUID
ContainerType_Vector_TcVnKeyPoint	GUID
ContainerType_Vector_TcVnDMatch	GUID
ContainerType_Vector_Vector_TcVnDMatch	GUID
ContainerType_String_SINT	GUID
ContainerType_Vector_String_SINT	GUID

6.1.2.3 数组

名称	类型
TcVnArray10_ITcVnImage	array [0..9] of ITcVnImage [▶ 407]
TcVnArray3_Point2_REAL	array [0..2] of TcVnPoint2_REAL [▶ 152]
TcVnArray33_UDINT	array [0..32] of UDINT
TcVnArray4_LREAL	array [0..3] of LREAL
TcVnArray4_Point2_REAL	array [0..3] of TcVnPoint2_REAL [▶ 152]
TcVnArray7_LREAL	array [0..6] of LREAL
TcVnArray8_LREAL	array [0..7] of LREAL
TcVnMatrix2x3_LREAL	array [0..1, 0..2] of LREAL
TcVnMatrix3x3_LREAL	array [0..2, 0..2] of LREAL
TcVnVector2_DINT	array [0..1] of DINT
TcVnVector2_INT	array [0..1] of INT
TcVnVector2_LREAL	array [0..1] of LREAL

名称	类型
TcVnVector2_REAL	array [0..1] of REAL
TcVnVector2_SINT	array [0..1] of SINT
TcVnVector2_UINT	array [0..1] of UINT
TcVnVector2_USINT	array [0..1] of USINT
TcVnVector3_INT	array [0..2] of INT
TcVnVector3_LREAL	array [0..2] of LREAL
TcVnVector3_REAL	array [0..2] of REAL
TcVnVector3_SINT	array [0..2] of SINT
TcVnVector3_UINT	array [0..2] of UINT
TcVnVector3_USINT	array [0..2] of USINT
TcVnVector4_DINT	array [0..3] of DINT
TcVnVector4_INT	array [0..3] of INT
TcVnVector4_LREAL	array [0..3] of LREAL
TcVnVector4_REAL	array [0..3] of REAL
TcVnVector4_SINT	array [0..3] of SINT
TcVnVector4_UINT	array [0..3] of UINT
TcVnVector4_USINT	array [0..3] of USINT

6.1.2.4 枚举

6.1.2.4.1 ETcVnDetectBarcodesWalshOptions

提供多种Walsh条形码检测选项。

语法

定义：

```
TYPE ETcVnDetectBarcodesWalshOptions :
(
    TCVN_DBWO_HORIZONTAL     := 1,
    TCVN_DBWO_VERTICAL       := 2,
    TCVN_DBWO_DIAGONAL_45    := 4,
    TCVN_DBWO_DIAGONAL_135   := 8,
    TCVN_DBWO_ALL_DIRECTIONS := 15
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_DBWO_HORIZONTAL	对水平条形码进行定位。
TCVN_DBWO_VERTICAL	对垂直条形码进行定位。
TCVN_DBWO_DIAGONAL_45	定位旋转了45度的对角条形码。
TCVN_DBWO_DIAGONAL_135	定位旋转了135度的对角条形码。
TCVN_DBWO_ALL_DIRECTIONS	对任意旋转的条形码进行定位。

6.1.2.4.2 ETcVnEdgeSelection

提供边缘选择选项。

语法

定义：

```
TYPE ETcVnEdgeSelection :
(
    TCVN_ES_RISING  := 1,
    TCVN_ES_FALLING := 2,
    TCVN_ES_ANY     := 3
)DINT;
END_TYPE
```

值

名称	描述
TCVN_ES_RISING	仅搜索上升沿，即从暗到亮的过渡
TCVN_ES_FALLING	仅搜索下降沿，即从亮到暗的过渡
TCVN_ES_ANY	搜索任意边缘方向，即上升或下降

6.1.2.4.3 ETcVnFindRoisMethod

提供了ROI提取方法。

语法

定义：

```
TYPE ETcVnFindRoisMethod :
(
    TCVN_FRM_GRID                := 0,
    TCVN_FRM_CONNECTEDCOMPONENTS := 1
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_FRM_GRID	通过在图像上铺设由 stRoiSize 大小单元格组成的规则网格（参见 TcVnParamsFindRois），只保留纹理足够丰富的单元格，从而提取感兴趣区域（ROI）。该方法同时支持二值图像和灰度图像。
TCVN_FRM_CONNECTEDCOMPONENTS	使用连通分量分析提取感兴趣区域（ROI）。对二值图像进行膨胀处理以连接相邻的连通分量，然后进行标注。

6.1.2.4.4 ETcVnJpegSampling

提供用于JPEG编码器的采样因子，以便在色度分辨率与文件大小之间进行权衡。

语法

定义：

```
TYPE ETcVnJpegSampling :
(
    TCVN_JS_444 := 11,
    TCVN_JS_440 := 12,
    TCVN_JS_422 := 21,
    TCVN_JS_420 := 22,
    TCVN_JS_411 := 41,
    TCVN_JS_410 := 42
)DINT;
END_TYPE
```

值

名称	描述
TCVN_JS_444	采样因子 1x1、1x1、1x1（全色度分辨率）
TCVN_JS_440	采样因子 1x2、1x1、1x1
TCVN_JS_422	采样因子 2x1、1x1、1x1

名称	描述
TCVN_JS_420	采样因子 2x2、1x1、1x1（默认）
TCVN_JS_411	采样因子 4x1、1x1、1x1
TCVN_JS_410	采样因子 4x2、1x1、1x1（最低色度分辨率）

6.1.2.4.5 ETcVnMergeRegionsOptions

提供了多种合并区域的选项。

语法

定义：

```
TYPE ETcVnMergeRegionsOptions :
(
    TCVN_MRO_UPRIGHT_BOUNDING_RECT := 1
)ULINT;
END_TYPE
```

值

名称	描述
TCVN_MRO_UPRIGHT_BOUNDING_RECT	将每个组合并为一个与坐标轴对齐（正置）的边界矩形。

6.1.2.4.6 ETcVnPolarizedImageInterpolation

提供了用于偏振图像去马赛克的插值选项。

语法

定义：

```
TYPE ETcVnPolarizedImageInterpolation :
(
    TCVN_PII_NONE             := -1,
    TCVN_PII_NEAREST_NEIGHBOR := 0,
    TCVN_PII_BILINEAR         := 1,
    TCVN_PII_BICUBIC          := 2,
    TCVN_PII_LANCZOS4         := 4
)DINT;
END_TYPE
```

值

名称	描述
TCVN_PII_NONE	不进行插值，即去马赛克处理后生成的图像宽高均为原始值的一半。
TCVN_PII_NEAREST_NEIGHBOR	使用最近邻插值将图像缩放至全分辨率。
TCVN_PII_BILINEAR	使用双线性插值将图像缩放至全分辨率。
TCVN_PII_BICUBIC	图像使用双立方插值法调整为全分辨率。
TCVN_PII_LANCZOS4	图像使用 Lanczos4 插值算法缩放至全分辨率。

6.1.2.4.7 ETcVnSalientFeaturesMethod

提供了计算突出特征的方法。

语法

定义：

```
TYPE ETcVnSalientFeaturesMethod :
(
    TCVN_SFM_WALSH_CODE := 0,
    TCVN_SFM_SOBEL      := 1
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_SFM_WALSH_CODE	基于沃尔什变换（Walsh transform）的码检测。
TCVN_SFM_SOBEL	用于码检测的 Sobel 和 Robinson 罗盘滤波器。

6.1.2.4.8 ETcVn2dCodeSearchStrategy

提供 2d 编码读取的搜索策略（多个 TCVN_CSS_XXX_INVERTED或多个 TCVN_CSS_XXX_FLIPPED不能同时选择）。

语法

定义：

```
TYPE ETcVn2dCodeSearchStrategy :
(
    TCVN_CSS_DEFAULT            := 1,
    TCVN_CSS_ONLY_NOT_INVERTED  := 8,
    TCVN_CSS_FIRST_NOT_INVERTED := 10,
    TCVN_CSS_ONLY_INVERTED      := 12,
    TCVN_CSS_FIRST_INVERTED     := 14,
    TCVN_CSS_ONLY_NOT_FLIPPED   := 64,
    TCVN_CSS_FIRST_NOT_FLIPPED  := 80,
    TCVN_CSS_ONLY_FLIPPED       := 96,
    TCVN_CSS_FIRST_FLIPPED      := 112
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_CSS_DEFAULT	算法使用默认搜索策略搜索编码，所有其它标记将被忽略（取决于编码类型）。
TCVN_CSS_ONLY_NOT_INVERTED	该算法只在非反转图像中搜索编码。
TCVN_CSS_FIRST_NOT_INVERTED	该算法首先在非反转图像中搜索编码，然后在反转图像中搜索编码。
TCVN_CSS_ONLY_INVERTED	该算法只在反转图像中搜索编码。
TCVN_CSS_FIRST_INVERTED	该算法首先在反转图像中搜索编码，然后在非反转图像中搜索编码。
TCVN_CSS_ONLY_NOT_FLIPPED	该算法只在非镜像图像中搜索编码。
TCVN_CSS_FIRST_NOT_FLIPPED	该算法首先在非镜像图像中搜索编码，然后在镜像图像中搜索编码。
TCVN_CSS_ONLY_FLIPPED	该算法只在镜像图像中搜索编码。
TCVN_CSS_FIRST_FLIPPED	该算法首先在镜像图像中搜索编码，然后在非镜像图像中搜索编码。

更多信息

该枚举为读码函数的搜索策略提供了设置选项。关于更多信息，请参见代码读取搜索策略。

相关函数

- F_VN_ReadDataMatrixCodeExp [▶ 852]
- F_VN_ReadQRCodeExp [▶ 866]

6.1.2.4.9 ETcVnAdaptiveThresholdMethod

提供自适应阈值的方法。

语法

定义：

```
TYPE ETcVnAdaptiveThresholdMethod :
(
    TCVN_ATM_MEAN     := 0,
    TCVN_ATM_GAUSSIAN := 1
)DINT;
END_TYPE
```

值

名称	描述
TCVN_ATM_MEAN	阈值等于（x,y）的 nBlockSize x nBlockSize 邻域的平均值再减去 fConstant 。
TCVN_ATM_GAUSSIAN	阈值等于（x,y）的 nBlockSize x nBlockSize 邻域的加权和（与高斯窗互相关）减去 fConstant。

相关函数

- F_VN_AdaptiveThresholdExp [▶ 1317]

6.1.2.4.10 ETcVnBarcodeSearchDirection

提供一维码的搜索方向。

语法

定义：

```
TYPE ETcVnBarcodeSearchDirection :
(
    TCVN_BSD_ANY        := 0,
    TCVN_BSD_HORIZONTAL := 1,
    TCVN_BSD_VERTICAL   := 2
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_BSD_ANY	该算法可在任何支持的方向上搜索编码。
TCVN_BSD_HORIZONTAL	该算法在水平方向上搜索编码。
TCVN_BSD_VERTICAL	该算法在垂直方向上搜索编码。

相关函数

- F_VN_ReadBarcodeExp [▶ 842]

6.1.2.4.11 ETcVnBarcodeType

提供要搜索的一维码类型。

语法

定义：

```
TYPE ETcVnBarcodeType :
(
    TCVN_BT_CODABAR        := 4,
    TCVN_BT_CODE39         := 8,
```

```
    TCVN_BT_CODE93          := 16,
    TCVN_BT_CODE128         := 32,
    TCVN_BT_EAN8            := 128,
    TCVN_BT_EAN13           := 256,
    TCVN_BT_ITF             := 512,
    TCVN_BT_UPCA            := 32768,
    TCVN_BT_UPCE            := 65536,
    TCVN_BT_ANY             := 99260,
    TCVN_BT_CODE39EXTENDED := 262144
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_BT_CODABAR	该算法搜索 Codabar 码。
TCVN_BT_CODE39	该算法搜索 Code-39 编码。
TCVN_BT_CODE93	该算法搜索 Code-93 编码。
TCVN_BT_CODE128	该算法搜索 Code-128 编码。
TCVN_BT_EAN8	该算法搜索 EAN-8 编码。
TCVN_BT_EAN13	该算法搜索 EAN-13 编码。
TCVN_BT_ITF	该算法搜索 ITF 编码。
TCVN_BT_UPCA	该算法搜索 UPC-A 编码。
TCVN_BT_UPCE	该算法搜索 UPC-E 编码。
TCVN_BT_ANY	该算法搜索任何支持的一维码。如果编码类型已知，建议直接选择具体类型。
TCVN_BT_CODE39EXTENDED	该算法搜索 Code-93-Extended 编码。

相关函数

- F_VN_ReadBarcode(Exp) [▶ 842]

6.1.2.4.12 ETcVnBlobCombination

提供多个斑点轮廓模式选择，这些轮廓模式应在多阈值情景下作为返回值（用于 TcVnParamsBlobDetection，而 TcVnParamsBlobDetection 又用于 F_VN_DetectBlobs）。

语法

定义：

```
TYPE ETcVnBlobCombination :
(
    TCVN_BC_SMALLEST          := 0,
    TCVN_BC_LARGEST           := 1,
    TCVN_BC_MIN_THRESHOLD     := 2,
    TCVN_BC_MAX_THRESHOLD     := 3,
    TCVN_BC_MEDIAN_THRESHOLD := 4
)DINT;
END_TYPE
```

值

名称	描述
TCVN_BC_SMALLEST	返回最小的斑点轮廓。
TCVN_BC_LARGEST	返回最大的斑点轮廓。
TCVN_BC_MIN_THRESHOLD	返回最小阈值的斑点轮廓。
TCVN_BC_MAX_THRESHOLD	返回最大阈值的斑点轮廓。
TCVN_BC_MEDIAN_THRESHOLD	返回中值阈值的斑点轮廓。

更多信息

枚举`ETcVnBlobCombination`用于结构TcVnParamsBlobDetection [▶ 229]。

相关函数

- F_VN_CalibrateCamera(Exp) [▶ 1071]
- F_VN_DetectBlobs(Exp) [▶ 1171]

6.1.2.4.13 ETcVnBoostClassifierType

提供不同类型的 Boost 分类器。

语法

定义：

```
TYPE ETcVnBoostClassifierType :
(
    TCVN_BCT_DISCRETE := 0,
    TCVN_BCT_REAL     := 1,
    TCVN_BCT_LOGIT    := 2,
    TCVN_BCT_GENTLE   := 3
)DINT;
END_TYPE
```

值

名称	描述
TCVN_BCT_DISCRETE	Discrete AdaBoost
TCVN_BCT_REAL	Real AdaBoost
TCVN_BCT_LOGIT	LogitBoost
TCVN_BCT_GENTLE	Gentle AdaBoost

6.1.2.4.14 ETcVnBorderInterpolationMethod

提供对非存在像素值插值的实现方法。一方面用于滤波功能，可以在图像边界（滤波掩膜覆盖边界的位置）进行滤波。另一方面，在几何图像变换后，用于插值未定义的像素。

语法

定义：

```
TYPE ETcVnBorderInterpolationMethod :
(
    TCVN_BIM_CONSTANT             := 0,
    TCVN_BIM_REPLICATE            := 1,
    TCVN_BIM_REFLECT              := 2,
    TCVN_BIM_WRAP                 := 3,
    TCVN_BIM_REFLECT_101          := 4,
    TCVN_BIM_DEFAULT              := 4,
    TCVN_BIM_TRANSPARENT          := 5,
    TCVN_BIM_ISOLATED_CONSTANT    := 16,
    TCVN_BIM_ISOLATED_REPLICATE   := 17,
    TCVN_BIM_ISOLATED_REFLECT     := 18,
    TCVN_BIM_ISOLATED_WRAP        := 19,
    TCVN_BIM_ISOLATED_REFLECT_101 := 20
)DINT;
END_TYPE
```

值

名称	描述
TCVN_BIM_CONSTANT	iiiiiii | abcdefgh | iiiiiii（包含某些指定的 i）
TCVN_BIM_REPLICATE	aaaaaaa | abcdefgh | hhhhhhh
TCVN_BIM_REFLECT	gfedcba | abcdefgh | hgfedcb
TCVN_BIM_WRAP	bcdefgh | abcdefgh | abcdefg
TCVN_BIM_REFLECT_101	hgfedcb | abcdefgh | gfedcba
TCVN_BIM_DEFAULT	如果不知道选择哪种方法，请选择此选项（与 REFLECT_101 相同）。

名称	描述
TCVN_BIM_TRANSPARENT	目标图像中的相应像素不会被修改。仅适用于几何图像变换。
TCVN_BIM_ISOLATED_CONSTANT	与 CONSTANT 类似，但将 ROI 作为孤立区域处理，即忽略 ROI 以外的周边图像像素。
TCVN_BIM_ISOLATED_REPLICATE	与 REPLICATE 类似，但将 ROI 作为孤立区域处理，即忽略 ROI 以外的周边图像像素。
TCVN_BIM_ISOLATED_REFLECT	与 REFLECT 类似，但将 ROI 作为孤立区域处理，即忽略 ROI 以外的周边图像像素。
TCVN_BIM_ISOLATED_WRAP	与 WRAP 类似，但将 ROI 作为孤立区域处理，即忽略 ROI 以外的周边图像像素。
TCVN_BIM_ISOLATED_REFLECT_101	与 REFLECT_101 类似，但将 ROI 作为孤立区域处理，即忽略 ROI 以外的周边图像像素。

6.1.2.4.15 ETcVnCalibrationPattern

提供常见的标定板类型。

语法

定义：

```
TYPE ETcVnCalibrationPattern :
(
    TCVN_CP_CIRCLES_SYM         := 0,
    TCVN_CP_CIRCLES_ASYM        := 1,
    TCVN_CP_CIRCLES_ASYM_INDENT := 2
)DINT;
END_TYPE
```

值

名称	描述
TCVN_CP_CIRCLES_SYM	对称圆形板
TCVN_CP_CIRCLES_ASYM	不对称圆形板，即每 2 行缩进一行，起始行不缩进
TCVN_CP_CIRCLES_ASYM_INDENT	非对称圆形板，即每 2 行缩进，起始行缩进

6.1.2.4.16 ETcVnCalibrationPatternOrigin

提供标定板原点。

语法

定义：

```
TYPE ETcVnCalibrationPatternOrigin :
(
    TCVN_CPO_TOPLEFT      := 0,
    TCVN_CPO_TOPCENTER    := 1,
    TCVN_CPO_TOPRIGHT     := 2,
    TCVN_CPO_CENTERLEFT   := 3,
    TCVN_CPO_CENTER       := 4,
    TCVN_CPO_CENTERRIGHT  := 5,
    TCVN_CPO_BOTTOMLEFT   := 6,
    TCVN_CPO_BOTTOMCENTER := 7,
    TCVN_CPO_BOTTOMRIGHT  := 8
)DINT;
END_TYPE
```

值

名称	描述
TCVN_CPO_TOPLEFT	原点位于左上方

名称	描述
TCVN_CPO_TOPCENTER	原点位于顶部中心
TCVN_CPO_TOPRIGHT	原点位于右上方
TCVN_CPO_CENTERLEFT	原点位于左侧中心
TCVN_CPO_CENTER	原点位于中心
TCVN_CPO_CENTERRIGHT	原点位于右侧中心
TCVN_CPO_BOTTOMLEFT	原点位于左下方
TCVN_CPO_BOTTOMCENTER	原点位于底部中心
TCVN_CPO_BOTTOMRIGHT	原点位于右下方

6.1.2.4.17 ETcVnCameraState

指定相机控制器的状态，该控制器控制连接的相机。

语法

定义：

```
TYPE ETcVnCameraState :
(
    TCVN_CS_ERROR             := -1,
    TCVN_CS_INITIAL           := 0,
    TCVN_CS_INITIALIZING      := 1,
    TCVN_CS_INITIALIZED       := 2,
    TCVN_CS_OPENING           := 3,
    TCVN_CS_OPENED            := 4,
    TCVN_CS_STARTACQUISITION  := 5,
    TCVN_CS_ACQUIRING         := 6,
    TCVN_CS_STOPACQUISITION   := 7,
    TCVN_CS_RESETTINGFEATURES := 8,
    TCVN_CS_TRIGGERING        := 9,
    TCVN_CS_CLOSING           := 10
)DINT;
END_TYPE
```

值

名称	描述
TCVN_CS_ERROR	相机控制器处于出错状态（例如，与相机的连接中断）。
TCVN_CS_INITIAL	相机控制器处于初始状态，准备与相机建立连接。
TCVN_CS_INITIALIZING	相机控制器正在初始化相机。
TCVN_CS_INITIALIZED	相机控制器已初始化相机。
TCVN_CS_OPENING	相机控制器正在建立与相机的连接。
TCVN_CS_OPENED	已建立相机连接，相机控制器已准备好开始图像采集。
TCVN_CS_STARTACQUISITION	相机控制器开始采集图像。
TCVN_CS_ACQUIRING	相机正在发送图像（连续采集或手动触发，具体取决于配置）。
TCVN_CS_STOPACQUISITION	相机控制器正在停止图像采集。
TCVN_CS_RESETTINGFEATURES	相机控制器正在重置相机的特征。
TCVN_CS_TRIGGERING	相机控制器正在处理软件触发。
TCVN_CS_CLOSING	相机控制器正在关闭与相机的连接。

更多信息

用于这些相机状态的状态机在各自的图像采集功能块 [▶ 1599]中描述：

- FB_VN_GevCameraControl [▶ 1609]
- FB_VN_FileSourceControl [▶ 1601]

- FB_VN_SimpleCameraControl [▶ 1630]

6.1.2.4.18 ETcVnCircularityCalculationMethod

Offers methods to calculate contour circularity

Syntax

Definition:

```
TYPE ETcVnCircularityCalculationMethod :
(
    TCVN_CCM_DEFAULT                := 0,
    TCVN_CCM_CONVEXHULL_PERIMETER   := 1,
    TCVN_CCM_MAX_RADIUS             := 2,
    TCVN_CCM_ENCLOSINGCIRCLE_RADIUS := 3
)DINT;
END_TYPE
```

Values

Name	Description
TCVN_CCM_DEFAULT	4pi * area / perimeter^2
TCVN_CCM_CONVEXHULL_PERIMETER	4pi * area / (convex hull perimeter)^2
TCVN_CCM_MAX_RADIUS	area / (pi * r^2), r = max distance to the contour's center of mass
TCVN_CCM_ENCLOSINGCIRCLE_RADIUS	area / (pi * r^2), r = radius of the minimum enclosing circle

6.1.2.4.19 ETcVnCircularityCalculationMethod

提供计算轮廓圆度的方法

语法

定义：

```
TYPE ETcVnCircularityCalculationMethod :
(
    TCVN_CCM_DEFAULT                := 0,
    TCVN_CCM_CONVEXHULL_PERIMETER   := 1,
    TCVN_CCM_MAX_RADIUS             := 2,
    TCVN_CCM_ENCLOSINGCIRCLE_RADIUS := 3
)DINT;
END_TYPE
```

值

名称	描述
TCVN_CCM_DEFAULT	4pi * 面积/周长^2
TCVN_CCM_CONVEXHULL_PERIMETER	4pi * 面积/（凸包周长）^2
TCVN_CCM_MAX_RADIUS	面积/（pi * r^2），r = 到轮廓的质心的最大距离
TCVN_CCM_ENCLOSINGCIRCLE_RADIUS	面积/（pi * r^2），r = 最小包围圆的半径

6.1.2.4.20 ETcVnClusteringAlgorithm

提供聚类算法

语法

定义：

```
TYPE ETcVnClusteringAlgorithm :
(
    TCVN_CA_KMEANSPP := 0,
    TCVN_CA_LBG      := 1
)DINT;
END_TYPE
```

值

名称	描述
TCVN_CA_KMEANSPP	KMeans++（固定聚类数）
TCVN_CA_LBG	LBG 变体（动态聚类数）

6.1.2.4.21 ETcVnCodeDetectionModel

提供经过训练的神经网络模型，用于编码检测。

语法

定义：

```
TYPE ETcVnCodeDetectionModel :
(
    TCVN_CDM_DETECT_1_1 := 0x0001100000000001,
    TCVN_CDM_DETECT_1_2 := 0x0001100000000002,
    TCVN_CDM_DETECT_1_3 := 0x0001100000000003,
    TCVN_CDM_DETECT_2_1 := 0x0001100000010001,
    TCVN_CDM_DETECT_2_2 := 0x0001100000010002,
    TCVN_CDM_DETECT_2_3 := 0x0001100000010003,
    TCVN_CDM_DETECT_3_1 := 0x0001100000020001,
    TCVN_CDM_DETECT_3_2 := 0x0001100000020002,
    TCVN_CDM_DETECT_3_3 := 0x0001100000020003
)ULINT;
END_TYPE
```

值

名称	描述
TCVN_CDM_DETECT_1_1	目标检测模型可检测一维条码和二维码，并将二维码分类为QR码、数据矩阵码和 DotCode（点码） 类型。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及相应的编码类型。输入：224×224 RGB 图像。
TCVN_CDM_DETECT_1_2	目标检测模型可检测一维条码和二维码，并将二维码分类为QR码、数据矩阵码和 DotCode（点码） 类型。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及相应的编码类型。输入：352×256 RGB 图像。
TCVN_CDM_DETECT_1_3	目标检测模型可检测一维条码和二维码，并将二维码分类为QR码、数据矩阵码和 DotCode（点码） 类型。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及相应的编码类型。输入：416×352 RGB 图像。
TCVN_CDM_DETECT_2_1	目标检测模型可检测一维条码。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及 TCVN_CT_1D 编码类型。输入：224×224 RGB 图像。
TCVN_CDM_DETECT_2_2	目标检测模型可检测一维条码。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及 TCVN_CT_1D 编码类型。输入：352×256 RGB 图像。

名称	描述
TCVN_CDM_DETECT_2_3	目标检测模型可检测一维条码。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及 TCVN_CT_1D 编码类型。输入：416×352 RGB 图像
TCVN_CDM_DETECT_3_1	目标检测模型可检测二维码。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及 TCVN_CT_2D 编码类型。输入：224×224 RGB 图像。
TCVN_CDM_DETECT_3_2	目标检测模型可检测二维码。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及 TCVN_CT_2D 编码类型。输入：352×256 RGB 图像。
TCVN_CDM_DETECT_3_3	目标检测模型可检测二维码。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及 TCVN_CT_2D 编码类型。输入：416×352 RGB 图像。

6.1.2.4.22 ETcVnCodeType

提供要检测的编码类型。

语法

定义：

```
TYPE ETcVnCodeType :
(
    TCVN_CT_ANY         := 0xFFFFFFFFFFFFFFFF,
    TCVN_CT_1D          := 0x0000FFFFFF000000,
    TCVN_CT_2D          := 0x0000000000FFFFFF,
    TCVN_CT_QR          := 0x0000000000000001,
    TCVN_CT_DATAMATRIX  := 0x0000000000000002,
    TCVN_CT_DOTCODE     := 0x0000000000000004
)ULINT;
END_TYPE
```

值

名称	描述
TCVN_CT_ANY	一维条码和二维码
TCVN_CT_1D	一维条码，例如 ETcVnBarcodeType 中的类型
TCVN_CT_2D	二维编码，例如QR码、数据矩阵码和 DotCode （点码）
TCVN_CT_QR	QR码
TCVN_CT_DATAMATRIX	数据矩阵码
TCVN_CT_DOTCODE	DotCode（点码）

6.1.2.4.23 ETcVnColorMap

提供色图（类似于 GNU Octave/MATLAB 型）。

语法

定义：

```
TYPE ETcVnColorMap :
(
    TCVN_CM_AUTUMN           := 0,
    TCVN_CM_BONE             := 1,
    TCVN_CM_JET              := 2,
    TCVN_CM_WINTER           := 3,
    TCVN_CM_RAINBOW          := 4,
    TCVN_CM_OCEAN            := 5,
    TCVN_CM_SUMMER           := 6,
    TCVN_CM_SPRING           := 7,
    TCVN_CM_COOL             := 8,
    TCVN_CM_HSV              := 9,
    TCVN_CM_PINK             := 10,
    TCVN_CM_HOT              := 11,
    TCVN_CM_PARULA           := 12,
    TCVN_CM_MAGMA            := 13,
    TCVN_CM_INFERNO          := 14,
    TCVN_CM_PLASMA           := 15,
    TCVN_CM_VIRIDIS          := 16,
    TCVN_CM_CIVIDIS          := 17,
    TCVN_CM_TWILIGHT         := 18,
    TCVN_CM_TWILIGHT_SHIFTED := 19,
    TCVN_CM_TURBO            := 20
)DINT;
END_TYPE
```

值

名称	描述
TCVN_CM_AUTUMN	红色 - 橙色 - 黄色
TCVN_CM_BONE	黑色 - 蓝灰色 - 白色
TCVN_CM_JET	深蓝色 - 绿色 - 深红色
TCVN_CM_WINTER	蓝色 - 绿色
TCVN_CM_RAINBOW	红色 - 绿色 - 紫色
TCVN_CM_OCEAN	黑色 - 蓝色 - 白色
TCVN_CM_SUMMER	绿色 - 黄色
TCVN_CM_SPRING	粉色 - 黄色
TCVN_CM_COOL	青色 - 品红色
TCVN_CM_HSV	红色 - 绿色 - 蓝色 - 红色
TCVN_CM_PINK	黑色 - 粉色 - 浅黄色 - 白色
TCVN_CM_HOT	黑色 - 红色 - 黄色 - 白色
TCVN_CM_PARULA	蓝色 - 黄色
TCVN_CM_MAGMA	黑色 - 品红色 - 浅黄色
TCVN_CM_INFERNO	黑色 - 品红色 - 黄色
TCVN_CM_PLASMA	蓝色 - 品红色 - 黄色
TCVN_CM_VIRIDIS	紫色 - 黄色
TCVN_CM_CIVIDIS	蓝色 - 灰色 - 黄色
TCVN_CM_TWILIGHT	白色 - 浅蓝色 - 深紫色 - 浅红色 - 白色
TCVN_CM_TWILIGHT_SHIFTED	深紫色 - 浅蓝色 - 白色 - 浅红色 - 深紫色
TCVN_CM_TURBO	深蓝色 - 浅绿色 - 深红色

6.1.2.4.24 ETcVnColorMapSize

提供色图的大小。

语法

定义：

```
TYPE ETcVnColorMapSize :
(
    TCVN_CMS_256   := 256,
    TCVN_CMS_65536 := 65536
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_CMS_256	256 色，用于 8 位图像（TCVN_ET_USINT）
TCVN_CMS_65536	65536 色，用于 16 位图像（TCVN_ET_UINT）

相关函数

- F_VN_GenerateColorMap [▶ 1249]
- F_VN_GenerateCustomColorMap [▶ 1250]

6.1.2.4.25 ETcVnColorSpaceTransform

提供颜色空间转换。

语法

定义：

```
TYPE ETcVnColorSpaceTransform :
(
    TCVN_CST_BGR_TO_BGRA                    := 0,
    TCVN_CST_RGB_TO_RGBA                    := 0,
    TCVN_CST_BGRA_TO_BGR                    := 1,
    TCVN_CST_RGBA_TO_RGB                    := 1,
    TCVN_CST_BGR_TO_RGBA                    := 2,
    TCVN_CST_RGB_TO_BGRA                    := 2,
    TCVN_CST_BGRA_TO_RGB                    := 3,
    TCVN_CST_RGBA_TO_BGR                    := 3,
    TCVN_CST_BGR_TO_RGB                     := 4,
    TCVN_CST_RGB_TO_BGR                     := 4,
    TCVN_CST_BGRA_TO_RGBA                   := 5,
    TCVN_CST_RGBA_TO_BGRA                   := 5,
    TCVN_CST_BGR_TO_GRAY                    := 6,
    TCVN_CST_RGB_TO_GRAY                    := 7,
    TCVN_CST_GRAY_TO_BGR                    := 8,
    TCVN_CST_GRAY_TO_RGB                    := 8,
    TCVN_CST_GRAY_TO_BGRA                   := 9,
    TCVN_CST_GRAY_TO_RGBA                   := 9,
    TCVN_CST_BGRA_TO_GRAY                   := 10,
    TCVN_CST_RGBA_TO_GRAY                   := 11,
    TCVN_CST_RGB_TO_BGR_565                 := 12,
    TCVN_CST_BGR_TO_BGR_565                 := 13,
    TCVN_CST_BGR_565_TO_RGB                 := 14,
    TCVN_CST_BGR_565_TO_BGR                 := 15,
    TCVN_CST_RGBA_TO_BGR_565                := 16,
    TCVN_CST_BGRA_TO_BGR_565                := 17,
    TCVN_CST_BGR_565_TO_RGBA                := 18,
    TCVN_CST_BGR_565_TO_BGRA                := 19,
    TCVN_CST_GRAY_TO_BGR_565                := 20,
    TCVN_CST_BGR_565_TO_GRAY                := 21,
    TCVN_CST_RGB_TO_BGR_555                 := 22,
    TCVN_CST_BGR_TO_BGR_555                 := 23,
    TCVN_CST_BGR_555_TO_RGB                 := 24,
    TCVN_CST_BGR_555_TO_BGR                 := 25,
    TCVN_CST_RGBA_TO_BGR_555                := 26,
    TCVN_CST_BGRA_TO_BGR_555                := 27,
    TCVN_CST_BGR_555_TO_RGBA                := 28,
    TCVN_CST_BGR_555_TO_BGRA                := 29,
    TCVN_CST_GRAY_TO_BGR_555                := 30,
    TCVN_CST_BGR_555_TO_GRAY                := 31,
    TCVN_CST_BGR_TO_XYZ                     := 32,
    TCVN_CST_RGB_TO_XYZ                     := 33,
    TCVN_CST_XYZ_TO_BGR                     := 34,
    TCVN_CST_XYZ_TO_RGB                     := 35,
    TCVN_CST_BGR_TO_YCRCB                   := 36,
    TCVN_CST_RGB_TO_YCRCB                   := 37,
    TCVN_CST_YCRCB_TO_BGR                   := 38,
    TCVN_CST_YCRCB_TO_RGB                   := 39,
    TCVN_CST_BGR_TO_HSV                     := 40,
    TCVN_CST_RGB_TO_HSV                     := 41,
    TCVN_CST_BGR_TO_LAB                     := 44,
    TCVN_CST_RGB_TO_LAB                     := 45,
    TCVN_CST_BAYER_RG_TO_BGR                := 46,
    TCVN_CST_BAYER_BG_TO_RGB                := 46,
    TCVN_CST_BAYER_GR_TO_BGR                := 47,
    TCVN_CST_BAYER_GB_TO_RGB                := 47,
    TCVN_CST_BAYER_BG_TO_BGR                := 48,
    TCVN_CST_BAYER_RG_TO_RGB                := 48,
    TCVN_CST_BAYER_GB_TO_BGR                := 49,
    TCVN_CST_BAYER_GR_TO_RGB                := 49,
    TCVN_CST_BGR_TO_LUV                     := 50,
    TCVN_CST_RGB_TO_LUV                     := 51,
```

```
    TCVN_CST_BGR_TO_HLS                 := 52,
    TCVN_CST_RGB_TO_HLS                 := 53,
    TCVN_CST_HSV_TO_BGR                 := 54,
    TCVN_CST_HSV_TO_RGB                 := 55,
    TCVN_CST_LAB_TO_BGR                 := 56,
    TCVN_CST_LAB_TO_RGB                 := 57,
    TCVN_CST_LUV_TO_BGR                 := 58,
    TCVN_CST_LUV_TO_RGB                 := 59,
    TCVN_CST_HLS_TO_BGR                 := 60,
    TCVN_CST_HLS_TO_RGB                 := 61,
    TCVN_CST_BAYER_RG_TO_BGR_VNG        := 62,
    TCVN_CST_BAYER_GR_TO_BGR_VNG        := 63,
    TCVN_CST_BAYER_BG_TO_BGR_VNG        := 64,
    TCVN_CST_BAYER_GB_TO_BGR_VNG        := 65,
    TCVN_CST_BAYER_RG_TO_RGB_VNG        := 64,
    TCVN_CST_BAYER_GR_TO_RGB_VNG        := 65,
    TCVN_CST_BAYER_BG_TO_RGB_VNG        := 62,
    TCVN_CST_BAYER_GB_TO_RGB_VNG        := 63,
    TCVN_CST_BGR_TO_HSV_FULL            := 66,
    TCVN_CST_RGB_TO_HSV_FULL            := 67,
    TCVN_CST_BGR_TO_HLS_FULL            := 68,
    TCVN_CST_RGB_TO_HLS_FULL            := 69,
    TCVN_CST_HSV_TO_BGR_FULL            := 70,
    TCVN_CST_HSV_TO_RGB_FULL            := 71,
    TCVN_CST_HLS_TO_BGR_FULL            := 72,
    TCVN_CST_HLS_TO_RGB_FULL            := 73,
    TCVN_CST_LBGR_TO_LAB                := 74,
    TCVN_CST_LRGB_TO_LAB                := 75,
    TCVN_CST_LBGR_TO_LUV                := 76,
    TCVN_CST_LRGB_TO_LUV                := 77,
    TCVN_CST_LAB_TO_LBGR                := 78,
    TCVN_CST_LAB_TO_LRGB                := 79,
    TCVN_CST_LUV_TO_LBGR                := 80,
    TCVN_CST_LUV_TO_LRGB                := 81,
    TCVN_CST_BGR_TO_YUV                 := 82,
    TCVN_CST_RGB_TO_YUV                 := 83,
    TCVN_CST_YUV_TO_BGR                 := 84,
    TCVN_CST_YUV_TO_RGB                 := 85,
    TCVN_CST_BAYER_RG_TO_GRAY           := 86,
    TCVN_CST_BAYER_GR_TO_GRAY           := 87,
    TCVN_CST_BAYER_BG_TO_GRAY           := 88,
    TCVN_CST_BAYER_GB_TO_GRAY           := 89,
    TCVN_CST_YUV_420_NV12_TO_RGB        := 90,
    TCVN_CST_YUV_420_NV12_TO_BGR        := 91,
    TCVN_CST_YUV_420_NV21_TO_RGB        := 92,
    TCVN_CST_YUV_420_NV21_TO_BGR        := 93,
    TCVN_CST_YUV_420_SP_TO_RGB          := 92,
    TCVN_CST_YUV_420_SP_TO_BGR          := 93,
    TCVN_CST_YUV_420_NV12_TO_RGBA       := 94,
    TCVN_CST_YUV_420_NV12_TO_BGRA       := 95,
    TCVN_CST_YUV_420_NV21_TO_RGBA       := 96,
    TCVN_CST_YUV_420_NV21_TO_BGRA       := 97,
    TCVN_CST_YUV_420_SP_TO_RGBA         := 96,
    TCVN_CST_YUV_420_SP_TO_BGRA         := 97,
    TCVN_CST_YUV_420_YV12_TO_RGB        := 98,
    TCVN_CST_YUV_420_YV12_TO_BGR        := 99,
    TCVN_CST_YUV_420_IYUV_TO_RGB        := 100,
    TCVN_CST_YUV_420_IYUV_TO_BGR        := 101,
    TCVN_CST_YUV_420_I420_TO_RGB        := 100,
    TCVN_CST_YUV_420_I420_TO_BGR        := 101,
    TCVN_CST_YUV_420_P_TO_RGB           := 98,
    TCVN_CST_YUV_420_P_TO_BGR           := 99,
    TCVN_CST_YUV_420_YV12_TO_RGBA       := 102,
    TCVN_CST_YUV_420_YV12_TO_BGRA       := 103,
    TCVN_CST_YUV_420_IYUV_TO_RGBA       := 104,
    TCVN_CST_YUV_420_IYUV_TO_BGRA       := 105,
    TCVN_CST_YUV_420_I420_TO_RGBA       := 104,
    TCVN_CST_YUV_420_I420_TO_BGRA       := 105,
    TCVN_CST_YUV_420_P_TO_RGBA          := 102,
    TCVN_CST_YUV_420_P_TO_BGRA          := 103,
    TCVN_CST_YUV_420_TO_GRAY            := 106,
    TCVN_CST_YUV_420_NV21_TO_GRAY       := 106,
    TCVN_CST_YUV_420_NV12_TO_GRAY       := 106,
    TCVN_CST_YUV_420_YV12_TO_GRAY       := 106,
    TCVN_CST_YUV_420_IYUV_TO_GRAY       := 106,
    TCVN_CST_YUV_420_I420_TO_GRAY       := 106,
    TCVN_CST_YUV_420_SP_TO_GRAY         := 106,
    TCVN_CST_YUV_420_P_TO_GRAY          := 106,
    TCVN_CST_YUV_422_UYVY_TO_RGB        := 107,
    TCVN_CST_YUV_422_UYVY_TO_BGR        := 108,
    TCVN_CST_YUV_422_Y422_TO_RGB        := 107,
    TCVN_CST_YUV_422_Y422_TO_BGR        := 108,
    TCVN_CST_YUV_422_UYNV_TO_RGB        := 107,
    TCVN_CST_YUV_422_UYNV_TO_BGR        := 108,
    TCVN_CST_YUV_422_UYVY_TO_RGBA       := 111,
    TCVN_CST_YUV_422_UYVY_TO_BGRA       := 112,
    TCVN_CST_YUV_422_Y422_TO_RGBA       := 111,
    TCVN_CST_YUV_422_Y422_TO_BGRA       := 112,
    TCVN_CST_YUV_422_UYNV_TO_RGBA       := 111,
    TCVN_CST_YUV_422_UYNV_TO_BGRA       := 112,
    TCVN_CST_YUV_422_YUY2_TO_RGB        := 115,
    TCVN_CST_YUV_422_YUY2_TO_BGR        := 116,
    TCVN_CST_YUV_422_YVYU_TO_RGB        := 117,
    TCVN_CST_YUV_422_YVYU_TO_BGR        := 118,
```

```
    TCVN_CST_YUV_422_YUYV_TO_RGB           := 115,
    TCVN_CST_YUV_422_YUYV_TO_BGR           := 116,
    TCVN_CST_YUV_422_YUNV_TO_RGB           := 115,
    TCVN_CST_YUV_422_YUNV_TO_BGR           := 116,
    TCVN_CST_YUV_422_YUY2_TO_RGBA          := 119,
    TCVN_CST_YUV_422_YUY2_TO_BGRA          := 120,
    TCVN_CST_YUV_422_YVYU_TO_RGBA          := 121,
    TCVN_CST_YUV_422_YVYU_TO_BGRA          := 122,
    TCVN_CST_YUV_422_YUYV_TO_RGBA          := 119,
    TCVN_CST_YUV_422_YUYV_TO_BGRA          := 120,
    TCVN_CST_YUV_422_YUNV_TO_RGBA          := 119,
    TCVN_CST_YUV_422_YUNV_TO_BGRA          := 120,
    TCVN_CST_YUV_422_UYVY_TO_GRAY          := 123,
    TCVN_CST_YUV_422_YUY2_TO_GRAY          := 124,
    TCVN_CST_YUV_422_Y422_TO_GRAY          := 123,
    TCVN_CST_YUV_422_UYNV_TO_GRAY          := 123,
    TCVN_CST_YUV_422_YVYU_TO_GRAY          := 124,
    TCVN_CST_YUV_422_YUYV_TO_GRAY          := 124,
    TCVN_CST_YUV_422_YUNV_TO_GRAY          := 124,
    TCVN_CST_RGBA_TO_PREMULTIPLICATED_RGBA := 125,
    TCVN_CST_PREMULTIPLICATED_RGBA_TO_RGBA := 126,
    TCVN_CST_RGB_TO_YUV_420_I420           := 127,
    TCVN_CST_BGR_TO_YUV_420_I420           := 128,
    TCVN_CST_RGB_TO_YUV_420_IYUV           := 127,
    TCVN_CST_BGR_TO_YUV_420_IYUV           := 128,
    TCVN_CST_RGBA_TO_YUV_420_I420          := 129,
    TCVN_CST_BGRA_TO_YUV_420_I420          := 130,
    TCVN_CST_RGBA_TO_YUV_420_IYUV          := 129,
    TCVN_CST_BGRA_TO_YUV_420_IYUV          := 130,
    TCVN_CST_RGB_TO_YUV_420_YV12           := 131,
    TCVN_CST_BGR_TO_YUV_420_YV12           := 132,
    TCVN_CST_RGBA_TO_YUV_420_YV12          := 133,
    TCVN_CST_BGRA_TO_YUV_420_YV12          := 134,
    TCVN_CST_BAYER_RG_TO_BGR_EA            := 135,
    TCVN_CST_BAYER_GR_TO_BGR_EA            := 136,
    TCVN_CST_BAYER_BG_TO_BGR_EA            := 137,
    TCVN_CST_BAYER_GB_TO_BGR_EA            := 138,
    TCVN_CST_BAYER_RG_TO_RGB_EA            := 137,
    TCVN_CST_BAYER_GR_TO_RGB_EA            := 138,
    TCVN_CST_BAYER_BG_TO_RGB_EA            := 135,
    TCVN_CST_BAYER_GB_TO_RGB_EA            := 136,
    TCVN_CST_BAYER_RG_TO_BGRA              := 139,
    TCVN_CST_BAYER_GR_TO_BGRA              := 140,
    TCVN_CST_BAYER_BG_TO_BGRA              := 141,
    TCVN_CST_BAYER_GB_TO_BGRA              := 142,
    TCVN_CST_BAYER_RG_TO_RGBA              := 141,
    TCVN_CST_BAYER_GR_TO_RGBA              := 142,
    TCVN_CST_BAYER_BG_TO_RGBA              := 139,
    TCVN_CST_BAYER_GB_TO_RGBA              := 140,
    TCVN_CST_RGB_TO_YUV_422_UYVY           := 143,
    TCVN_CST_BGR_TO_YUV_422_UYVY           := 144,
    TCVN_CST_RGB_TO_YUV_422_Y422           := 143,
    TCVN_CST_BGR_TO_YUV_422_Y422           := 144,
    TCVN_CST_RGB_TO_YUV_422_UYNV           := 143,
    TCVN_CST_BGR_TO_YUV_422_UYNV           := 144,
    TCVN_CST_RGBA_TO_YUV_422_UYVY          := 145,
    TCVN_CST_BGRA_TO_YUV_422_UYVY          := 146,
    TCVN_CST_RGBA_TO_YUV_422_Y422          := 145,
    TCVN_CST_BGRA_TO_YUV_422_Y422          := 146,
    TCVN_CST_RGBA_TO_YUV_422_UYNV          := 145,
    TCVN_CST_BGRA_TO_YUV_422_UYNV          := 146,
    TCVN_CST_RGB_TO_YUV_422_YUY2           := 147,
    TCVN_CST_BGR_TO_YUV_422_YUY2           := 148,
    TCVN_CST_RGB_TO_YUV_422_YVYU           := 149,
    TCVN_CST_BGR_TO_YUV_422_YVYU           := 150,
    TCVN_CST_RGB_TO_YUV_422_YUYV           := 147,
    TCVN_CST_BGR_TO_YUV_422_YUYV           := 148,
    TCVN_CST_RGB_TO_YUV_422_YUNV           := 147,
    TCVN_CST_BGR_TO_YUV_422_YUNV           := 148,
    TCVN_CST_RGBA_TO_YUV_422_YUY2          := 151,
    TCVN_CST_BGRA_TO_YUV_422_YUY2          := 152,
    TCVN_CST_RGBA_TO_YUV_422_YVYU          := 153,
    TCVN_CST_BGRA_TO_YUV_422_YVYU          := 154,
    TCVN_CST_RGBA_TO_YUV_422_YUYV          := 151,
    TCVN_CST_BGRA_TO_YUV_422_YUYV          := 152,
    TCVN_CST_RGBA_TO_YUV_422_YUNV          := 151,
    TCVN_CST_BGRA_TO_YUV_422_YUNV          := 152,
    TCVN_CST_MAX                           := 155
)DINT;
END_TYPE
```

值

名称	描述
TCVN_CST_BGR_TO_BGRA	将颜色空间从 BGR 转换为 BGRA。
TCVN_CST_RGB_TO_RGBA	将颜色空间从 RGB 转换为 RGBA。
TCVN_CST_BGRA_TO_BGR	将颜色空间从 BGRA 转换为 BGR。
TCVN_CST_RGBA_TO_RGB	将颜色空间从 RGBA 转换为 RGB。

名称	描述
TCVN_CST_BGR_TO_RGBA	将颜色空间从 BGR 转换为 RGBA。
TCVN_CST_RGB_TO_BGRA	将颜色空间从 RGB 转换为 BGRA。
TCVN_CST_BGRA_TO_RGB	将颜色空间从 BGRA 转换为 RGB。
TCVN_CST_RGBA_TO_BGR	将颜色空间从 RGBA 转换为 BGR。
TCVN_CST_BGR_TO_RGB	将颜色空间从 BGR 转换为 RGB。
TCVN_CST_RGB_TO_BGR	将颜色空间从 RGB 转换为 BGR。
TCVN_CST_BGRA_TO_RGBA	将颜色空间从 BGRA 转换为 RGBA。
TCVN_CST_RGBA_TO_BGRA	将颜色空间从 RGBA 转换为 BGRA。
TCVN_CST_BGR_TO_GRAY	将颜色空间从 BGR 转换为灰色。
TCVN_CST_RGB_TO_GRAY	将颜色空间从 RGB 转换为灰色。
TCVN_CST_GRAY_TO_BGR	将颜色空间从灰色转换为 BGR。
TCVN_CST_GRAY_TO_RGB	将颜色空间从灰色转换为 RGB。
TCVN_CST_GRAY_TO_BGRA	将颜色空间从灰色转换为 BGRA。
TCVN_CST_GRAY_TO_RGBA	将颜色空间从灰色转换为 RGBA。
TCVN_CST_BGRA_TO_GRAY	将颜色空间从 BGRA 转换为灰色。
TCVN_CST_RGBA_TO_GRAY	将颜色空间从 RGBA 转换为灰色。
TCVN_CST_RGB_TO_BGR_565	将颜色空间从 RGB（24 位 888）转换为 BGR（16 位 565）。
TCVN_CST_BGR_TO_BGR_565	将颜色空间从 BGR（24 位 888）转换为 BGR（16 位 565）。
TCVN_CST_BGR_565_TO_RGB	将颜色空间从 BGR（16 位 565）转换为 RGB（24 位 888）。
TCVN_CST_BGR_565_TO_BGR	将颜色空间从 BGR（16 位 565）转换为 BGR（24 位 888）。
TCVN_CST_RGBA_TO_BGR_565	将颜色空间从 RGBA（32 位 8888）转换为 BGR（16 位 565）。
TCVN_CST_BGRA_TO_BGR_565	将颜色空间从 BGRA（32 位 8888）转换为 BGR（16 位 565）。
TCVN_CST_BGR_565_TO_RGBA	将颜色空间从 BGR（16 位 565）转换为 RGBA（32 位 8888）。
TCVN_CST_BGR_565_TO_BGRA	将颜色空间从 BGR（16 位 565）转换为 BGRA（32 位 8888）。
TCVN_CST_GRAY_TO_BGR_565	将颜色空间从灰色转换为 BGR（16 位 565）。
TCVN_CST_BGR_565_TO_GRAY	将颜色空间从 BGR（16 位 565）转换为灰色。
TCVN_CST_RGB_TO_BGR_555	将颜色空间从 RGB（24 位 888）转换为 BGR（16 位 555）。
TCVN_CST_BGR_TO_BGR_555	将颜色空间从 BGR（24 位 888）转换为 BGR（16 位 555）。
TCVN_CST_BGR_555_TO_RGB	将颜色空间从 BGR（16 位 555）转换为 RGB（24 位 888）。
TCVN_CST_BGR_555_TO_BGR	将颜色空间从 BGR（16 位 555）转换为 BGR（24 位 888）。
TCVN_CST_RGBA_TO_BGR_555	将颜色空间从 RGBA（32 位 8888）转换为 BGR（16 位 555）。
TCVN_CST_BGRA_TO_BGR_555	将颜色空间从 BGRA（32 位 8888）转换为 BGR（16 位 555）。
TCVN_CST_BGR_555_TO_RGBA	将颜色空间从 BGR（16 位 555）转换为 RGBA（32 位 8888）。

名称	描述
TCVN_CST_BGR_555_TO_BGRA	将颜色空间从 BGR（16 位 555）转换为 BGRA（32 位 8888）。
TCVN_CST_GRAY_TO_BGR_555	将颜色空间从灰色转换为 BGR（16 位 555）。
TCVN_CST_BGR_555_TO_GRAY	将颜色空间从 BGR（16 位 555）转换为灰色。
TCVN_CST_BGR_TO_XYZ	将颜色空间从 BGR 转换为 CIE XYZ（缩放至图像的完整数值范围）。
TCVN_CST_RGB_TO_XYZ	将颜色空间从 RGB 转换为 CIE XYZ（缩放至图像的完整数值范围）。
TCVN_CST_XYZ_TO_BGR	将颜色空间从 CIE XYZ（缩放至图像的完整数值范围）转换为 BGR。
TCVN_CST_XYZ_TO_RGB	将颜色空间从 CIE XYZ（缩放至图像的完整数值范围）转换为 RGB。
TCVN_CST_BGR_TO_YCRCB	将颜色空间从 BGR 转换为 YCrCb（缩放至图像的完整数值范围）。
TCVN_CST_RGB_TO_YCRCB	将颜色空间从 RGB 转换为 YCrCb（缩放至图像的完整数值范围）。
TCVN_CST_YCRCB_TO_BGR	将颜色空间从 YCrCb（缩放至图像的完整数值范围）转换为 BGR。
TCVN_CST_YCRCB_TO_RGB	将颜色空间从 YCrCb（缩放至图像的完整数值范围）转换为 RGB。
TCVN_CST_BGR_TO_HSV	将颜色空间从 BGR 转换为 HSV（支持 USINT 或 REAL 类型的图像；对于 USINT 类型，H 的缩放范围为 0 至 180，S 和 V 的缩放范围为 0 至 255）。
TCVN_CST_RGB_TO_HSV	将颜色空间从 RGB 转换为 HSV（支持 USINT 或 REAL 类型的图像；对于 USINT 类型，H 的缩放范围为 0 至 180，S 和 V 的缩放范围为 0 至 255）。
TCVN_CST_BGR_TO_LAB	将颜色空间从 BGR 转换为 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
TCVN_CST_RGB_TO_LAB	将颜色空间从 RGB 转换为 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
TCVN_CST_BAYER_RG_TO_BGR	将颜色空间从 BayerRG 转换为 BGR（支持 USINT 或 UINT 类型的图像）。
TCVN_CST_BAYER_BG_TO_RGB	将颜色空间从 BayerBG 转换为 RGB（支持 USINT 或 UINT 类型的图像）。
TCVN_CST_BAYER_GR_TO_BGR	将颜色空间从 BayerGR 转换为 BGR（支持 USINT 或 UINT 类型的图像）。
TCVN_CST_BAYER_GB_TO_RGB	将颜色空间从 BayerGB 转换为 RGB（支持 USINT 或 UINT 类型的图像）。
TCVN_CST_BAYER_BG_TO_BGR	将颜色空间从 BayerBG 转换为 BGR（支持 USINT 或 UINT 类型的图像）。
TCVN_CST_BAYER_RG_TO_RGB	将颜色空间从 BayerRG 转换为 RGB（支持 USINT 或 UINT 类型的图像）。
TCVN_CST_BAYER_GB_TO_BGR	将颜色空间从 BayerGB 转换为 BGR（支持 USINT 或 UINT 类型的图像）。
TCVN_CST_BAYER_GR_TO_RGB	将颜色空间从 BayerGR 转换为 RGB（支持 USINT 或 UINT 类型的图像）。
TCVN_CST_BGR_TO_LUV	将颜色空间从 BGR 转换为 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
TCVN_CST_RGB_TO_LUV	将颜色空间从 RGB 转换为 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。

名称	描述
TCVN_CST_BGR_TO_HLS	将颜色空间从 BGR 转换为 HLS（对于 USINT 类型的图像，H 的缩放范围为 0 至 180，L 和 S 的缩放范围为 0 至 255）。
TCVN_CST_RGB_TO_HLS	将颜色空间从 RGB 转换为 HLS（对于 USINT 类型的图像，H 的缩放范围为 0 至 180，L 和 S 的缩放范围为 0 至 255）。
TCVN_CST_HSV_TO_BGR	将颜色空间从 HSV（对于 USINT 类型的图像，H 的缩放范围为 0 至 180，S 和 V 的缩放范围为 0 至 255）转换为 BGR。
TCVN_CST_HSV_TO_RGB	将颜色空间从 HSV（对于 USINT 类型的图像，H 的缩放范围为 0 至 180，S 和 V 的缩放范围为 0 至 255）转换为 RGB。
TCVN_CST_LAB_TO_BGR	将颜色空间从 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 BGR。
TCVN_CST_LAB_TO_RGB	将颜色空间从 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 RGB。
TCVN_CST_LUV_TO_BGR	将颜色空间从 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 BGR。
TCVN_CST_LUV_TO_RGB	将颜色空间从 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 RGB。
TCVN_CST_HLS_TO_BGR	将颜色空间从 HLS（对于 USINT 类型的图像，H 的范围为 0 至 180，L 和 S 的范围为 0 至 255）转换为 BGR。
TCVN_CST_HLS_TO_RGB	将颜色空间从 HLS（对于 USINT 类型的图像，H 的范围为 0 至 180，L 和 S 的范围为 0 至 255）转换为 RGB。
TCVN_CST_BAYER_RG_TO_BGR_VNG	使用可变梯度数将颜色空间从 BayerRG 转换为 BGR（支持 USINT 类型的图像）。
TCVN_CST_BAYER_GR_TO_BGR_VNG	使用可变梯度数将颜色空间从 BayerGR 转换为 BGR（支持 USINT 类型的图像）。
TCVN_CST_BAYER_BG_TO_BGR_VNG	使用可变梯度数将颜色空间从 BayerBG 转换为 BGR（支持 USINT 类型的图像）。
TCVN_CST_BAYER_GB_TO_BGR_VNG	使用可变梯度数将颜色空间从 BayerGB 转换为 BGR（支持 USINT 类型的图像）。
TCVN_CST_BAYER_RG_TO_RGB_VNG	使用可变梯度数将颜色空间从 BayerRG 转换为 RGB（支持 USINT 类型的图像）。
TCVN_CST_BAYER_GR_TO_RGB_VNG	使用可变梯度数将颜色空间从 BayerGR 转换为 RGB（支持 USINT 类型的图像）。
TCVN_CST_BAYER_BG_TO_RGB_VNG	使用可变梯度数将颜色空间从 BayerBG 转换为 RGB（支持 USINT 类型的图像）。
TCVN_CST_BAYER_GB_TO_RGB_VNG	使用可变梯度数将颜色空间从 BayerGB 转换为 RGB（支持 USINT 类型的图像）。
TCVN_CST_BGR_TO_HSV_FULL	将颜色空间从 BGR 转换为 HSV（支持 USINT 或 REAL 类型的图像；对于 USINT 类型，所有通道的缩放范围为 0 至 255）。
TCVN_CST_RGB_TO_HSV_FULL	将颜色空间从 RGB 转换为 HSV（支持 USINT 或 REAL 类型的图像；对于 USINT 类型，所有通道的缩放范围为 0 至 255）。
TCVN_CST_BGR_TO_HLS_FULL	将颜色空间从 BGR 转换为 HLS（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
TCVN_CST_RGB_TO_HLS_FULL	将颜色空间从 RGB 转换为 HLS（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。

名称	描述
TCVN_CST_HSV_TO_BGR_FULL	将颜色空间从 HSV（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 BGR。
TCVN_CST_HSV_TO_RGB_FULL	将颜色空间从 HSV（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 RGB。
TCVN_CST_HLS_TO_BGR_FULL	将颜色空间从 HLS（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 BGR。
TCVN_CST_HLS_TO_RGB_FULL	将颜色空间从 HLS（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 RGB。
TCVN_CST_LBGR_TO_LAB	将颜色空间从 LBGR 转换为 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
TCVN_CST_LRGB_TO_LAB	将颜色空间从 LRGB 转换为 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
TCVN_CST_LBGR_TO_LUV	将颜色空间从 LBGR 转换为 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
TCVN_CST_LRGB_TO_LUV	将颜色空间从 LRGB 转换为 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
TCVN_CST_LAB_TO_LBGR	将颜色空间从 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 LBGR。
TCVN_CST_LAB_TO_LRGB	将颜色空间从 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 LRGB。
TCVN_CST_LUV_TO_LBGR	将颜色空间从 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 LBGR。
TCVN_CST_LUV_TO_LRGB	将颜色空间从 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 LRGB。
TCVN_CST_BGR_TO_YUV	将颜色空间从 BGR 转换为 YUV（缩放至图像的完整数值范围）。
TCVN_CST_RGB_TO_YUV	将颜色空间从 RGB 转换为 YUV（缩放至图像的完整数值范围）。
TCVN_CST_YUV_TO_BGR	将颜色空间从 YUV（缩放至图像的完整数值范围）转换为 BGR。
TCVN_CST_YUV_TO_RGB	将颜色空间从 YUV（缩放至图像的完整数值范围）转换为 RGB。
TCVN_CST_BAYER_RG_TO_GRAY	将颜色空间从 BayerRG 转换为灰度（支持 USINT 或 UINT 类型的图像）。
TCVN_CST_BAYER_GR_TO_GRAY	将颜色空间从 BayerGR 转换为灰度（支持 USINT 或 UINT 类型的图像）。
TCVN_CST_BAYER_BG_TO_GRAY	将颜色空间从 BayerBG 转换为灰度（支持 USINT 或 UINT 类型的图像）。
TCVN_CST_BAYER_GB_TO_GRAY	将颜色空间从 BayerGB 转换为灰度（支持 USINT 或 UINT 类型的图像）。
TCVN_CST_YUV_420_NV12_TO_RGB	将颜色空间从 YUV420 NV12 转换为 RGB。
TCVN_CST_YUV_420_NV12_TO_BGR	将颜色空间从 YUV420 NV12 转换为 BGR。
TCVN_CST_YUV_420_NV21_TO_RGB	将颜色空间从 YUV420 NV21（SP）转换为 RGB。
TCVN_CST_YUV_420_NV21_TO_BGR	将颜色空间从 YUV420 NV21（SP）转换为 BGR。
TCVN_CST_YUV_420_SP_TO_RGB	将颜色空间从 YUV420 NV21（SP）转换为 RGB。
TCVN_CST_YUV_420_SP_TO_BGR	将颜色空间从 YUV420 NV21（SP）转换为 BGR。
TCVN_CST_YUV_420_NV12_TO_RGBA	将颜色空间从 YUV420 NV12 转换为 RGBA。
TCVN_CST_YUV_420_NV12_TO_BGRA	将颜色空间从 YUV420 NV12 转换为 BGRA。
TCVN_CST_YUV_420_NV21_TO_RGBA	将颜色空间从 YUV420 NV21（SP）转换为 RGBA。
TCVN_CST_YUV_420_NV21_TO_BGRA	将颜色空间从 YUV420 NV21（SP）转换为 BGRA。

名称	描述
TCVN_CST_YUV_420_SP_TO_RGBA	将颜色空间从 YUV420 NV21（SP）转换为 RGBA。
TCVN_CST_YUV_420_SP_TO_BGRA	将颜色空间从 YUV420 NV21（SP）转换为 BGRA。
TCVN_CST_YUV_420_YV12_TO_RGB	将颜色空间从 YUV420 YV12（P）转换为 RGB。
TCVN_CST_YUV_420_YV12_TO_BGR	将颜色空间从 YUV420 YV12（P）转换为 BGR。
TCVN_CST_YUV_420_IYUV_TO_RGB	将颜色空间从 YUV420 IYUV（I420）转换为 RGB。
TCVN_CST_YUV_420_IYUV_TO_BGR	将颜色空间从 YUV420 IYUV（I420）转换为 BGR。
TCVN_CST_YUV_420_I420_TO_RGB	将颜色空间从 YUV420 IYUV（I420）转换为 RGB。
TCVN_CST_YUV_420_I420_TO_BGR	将颜色空间从 YUV420 IYUV（I420）转换为 BGR。
TCVN_CST_YUV_420_P_TO_RGB	将颜色空间从 YUV420 YV12（P）转换为 RGB。
TCVN_CST_YUV_420_P_TO_BGR	将颜色空间从 YUV420 YV12（P）转换为 BGR。
TCVN_CST_YUV_420_YV12_TO_RGBA	将颜色空间从 YUV420 YV12（P）转换为 RGBA。
TCVN_CST_YUV_420_YV12_TO_BGRA	将颜色空间从 YUV420 YV12（P）转换为 BGRA。
TCVN_CST_YUV_420_IYUV_TO_RGBA	将颜色空间从 YUV420 IYUV（I420）转换为 RGBA。
TCVN_CST_YUV_420_IYUV_TO_BGRA	将颜色空间从 YUV420 IYUV（I420）转换为 BGRA。
TCVN_CST_YUV_420_I420_TO_RGBA	将颜色空间从 YUV420 IYUV（I420）转换为 RGBA。
TCVN_CST_YUV_420_I420_TO_BGRA	将颜色空间从 YUV420 IYUV（I420）转换为 BGRA。
TCVN_CST_YUV_420_P_TO_RGBA	将颜色空间从 YUV420 YV12（P）转换为 RGBA。
TCVN_CST_YUV_420_P_TO_BGRA	将颜色空间从 YUV420 YV12（P）转换为 BGRA。
TCVN_CST_YUV_420_TO_GRAY	将颜色空间从 YUV420 转换为灰色。
TCVN_CST_YUV_420_NV21_TO_GRAY	将颜色空间从 YUV420 转换为灰色。
TCVN_CST_YUV_420_NV12_TO_GRAY	将颜色空间从 YUV420 转换为灰色。
TCVN_CST_YUV_420_YV12_TO_GRAY	将颜色空间从 YUV420 转换为灰色。
TCVN_CST_YUV_420_IYUV_TO_GRAY	将颜色空间从 YUV420 转换为灰色。
TCVN_CST_YUV_420_I420_TO_GRAY	将颜色空间从 YUV420 转换为灰色。
TCVN_CST_YUV_420_SP_TO_GRAY	将颜色空间从 YUV420 转换为灰色。
TCVN_CST_YUV_420_P_TO_GRAY	将颜色空间从 YUV420 转换为灰色。
TCVN_CST_YUV_422_UYVY_TO_RGB	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 RGB。
TCVN_CST_YUV_422_UYVY_TO_BGR	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 BGR。
TCVN_CST_YUV_422_Y422_TO_RGB	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 RGB。
TCVN_CST_YUV_422_Y422_TO_BGR	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 BGR。
TCVN_CST_YUV_422_UYNV_TO_RGB	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 RGB。
TCVN_CST_YUV_422_UYNV_TO_BGR	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 BGR。
TCVN_CST_YUV_422_UYVY_TO_RGBA	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 RGBA。
TCVN_CST_YUV_422_UYVY_TO_BGRA	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 BGRA。
TCVN_CST_YUV_422_Y422_TO_RGBA	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 RGB。
TCVN_CST_YUV_422_Y422_TO_BGRA	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 BGRA。
TCVN_CST_YUV_422_UYNV_TO_RGBA	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 RGB。

名称	描述
TCVN_CST_YUV_422_UYNV_TO_BGRA	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 BGRA。
TCVN_CST_YUV_422_YUY2_TO_RGB	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 RGB。
TCVN_CST_YUV_422_YUY2_TO_BGR	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 BGR。
TCVN_CST_YUV_422_YVYU_TO_RGB	将颜色空间从 YUV422 YVYU 转换为 RGB。
TCVN_CST_YUV_422_YVYU_TO_BGR	将颜色空间从 YUV422 YVYU 转换为 BGR。
TCVN_CST_YUV_422_YUYV_TO_RGB	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 RGB。
TCVN_CST_YUV_422_YUYV_TO_BGR	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 BGR。
TCVN_CST_YUV_422_YUNV_TO_RGB	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 RGB。
TCVN_CST_YUV_422_YUNV_TO_BGR	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 BGR。
TCVN_CST_YUV_422_YUY2_TO_RGBA	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 RGBA。
TCVN_CST_YUV_422_YUY2_TO_BGRA	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 BGRA。
TCVN_CST_YUV_422_YVYU_TO_RGBA	将颜色空间从 YUV422 YVYU 转换为 RGBA。
TCVN_CST_YUV_422_YVYU_TO_BGRA	将颜色空间从 YUV422 YVYU 转换为 BGRA。
TCVN_CST_YUV_422_YUYV_TO_RGBA	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 RGBA。
TCVN_CST_YUV_422_YUYV_TO_BGRA	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 BGRA。
TCVN_CST_YUV_422_YUNV_TO_RGBA	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 RGBA。
TCVN_CST_YUV_422_YUNV_TO_BGRA	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 BGRA。
TCVN_CST_YUV_422_UYVY_TO_GRAY	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为灰色。
TCVN_CST_YUV_422_YUY2_TO_GRAY	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为灰色。
TCVN_CST_YUV_422_Y422_TO_GRAY	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为灰色。
TCVN_CST_YUV_422_UYNV_TO_GRAY	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为灰色。
TCVN_CST_YUV_422_YVYU_TO_GRAY	将颜色空间从 YUV422 YVYU 转换为灰色。
TCVN_CST_YUV_422_YUYV_TO_GRAY	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为灰色。
TCVN_CST_YUV_422_YUNV_TO_GRAY	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为灰色。
TCVN_CST_RGBA_TO_PREMULTIPLICATED_RGBA	将颜色空间从 RGBA 转换为预乘 RGBA。
TCVN_CST_PREMULTIPLICATED_RGBA_TO_RGBA	将颜色空间从预乘 RGBA 转换为 RGBA。
TCVN_CST_RGB_TO_YUV_420_I420	将颜色空间从 RGB 转换为 YUV420 IYUV（I420）。
TCVN_CST_BGR_TO_YUV_420_I420	将颜色空间从 BGR 转换为 YUV420 IYUV（I420）。
TCVN_CST_RGB_TO_YUV_420_IYUV	将颜色空间从 RGB 转换为 YUV420 IYUV（I420）。
TCVN_CST_BGR_TO_YUV_420_IYUV	将颜色空间从 BGR 转换为 YUV420 IYUV（I420）。
TCVN_CST_RGBA_TO_YUV_420_I420	将颜色空间从 RGBA 转换为 YUV420 IYUV（I420）。

名称	描述
TCVN_CST_BGRA_TO_YUV_420_I420	将颜色空间从 BGRA 转换为 YUV420 IYUV（I420）。
TCVN_CST_RGBA_TO_YUV_420_IYUV	将颜色空间从 RGBA 转换为 YUV420 IYUV（I420）。
TCVN_CST_BGRA_TO_YUV_420_IYUV	将颜色空间从 BGRA 转换为 YUV420 IYUV（I420）。
TCVN_CST_RGB_TO_YUV_420_YV12	将颜色空间从 RGB 转换为 YUV420 YV12（P）。
TCVN_CST_BGR_TO_YUV_420_YV12	将颜色空间从 BGR 转换为 YUV420 YV12（P）。
TCVN_CST_RGBA_TO_YUV_420_YV12	将颜色空间从 RGBA 转换为 YUV420 YV12（P）。
TCVN_CST_BGRA_TO_YUV_420_YV12	将颜色空间从 BGRA 转换为 YUV420 YV12（P）。
TCVN_CST_BAYER_RG_TO_BGR_EA	使用边缘感知算法，将颜色空间从 BayerRG 转换为 BGR。
TCVN_CST_BAYER_GR_TO_BGR_EA	使用边缘感知算法，将颜色空间从 BayerGR 转换为 BGR。
TCVN_CST_BAYER_BG_TO_BGR_EA	使用边缘感知算法，将颜色空间从 BayerBG 转换为 BGR。
TCVN_CST_BAYER_GB_TO_BGR_EA	使用边缘感知算法，将颜色空间从 BayerGB 转换为 BGR。
TCVN_CST_BAYER_RG_TO_RGB_EA	使用边缘感知算法，将颜色空间从 BayerRG 转换为 RGB。
TCVN_CST_BAYER_GR_TO_RGB_EA	使用边缘感知算法，将颜色空间从 BayerGR 转换为 RGB。
TCVN_CST_BAYER_BG_TO_RGB_EA	使用边缘感知算法，将颜色空间从 BayerBG 转换为 RGB。
TCVN_CST_BAYER_GB_TO_RGB_EA	使用边缘感知算法，将颜色空间从 BayerGB 转换为 RGB。
TCVN_CST_BAYER_RG_TO_BGRA	将颜色空间从 BayerRG 转换为 BGRA。
TCVN_CST_BAYER_GR_TO_BGRA	将颜色空间从 BayerGR 转换为 BGRA。
TCVN_CST_BAYER_BG_TO_BGRA	将颜色空间从 BayerBG 转换为 BGRA。
TCVN_CST_BAYER_GB_TO_BGRA	将颜色空间从 BayerGB 转换为 BGRA。
TCVN_CST_BAYER_RG_TO_RGBA	将颜色空间从 BayerRG 转换为 RGBA。
TCVN_CST_BAYER_GR_TO_RGBA	将颜色空间从 BayerGR 转换为 RGBA。
TCVN_CST_BAYER_BG_TO_RGBA	将颜色空间从 BayerBG 转换为 RGBA。
TCVN_CST_BAYER_GB_TO_RGBA	将颜色空间从 BayerGB 转换为 RGBA。
TCVN_CST_RGB_TO_YUV_422_UYVY	将颜色空间从 RGB 转换为 YUV422 UYVY（Y422、UYNV）。
TCVN_CST_BGR_TO_YUV_422_UYVY	将颜色空间从 BGR 转换为 YUV422 UYVY（Y422、UYNV）。
TCVN_CST_RGB_TO_YUV_422_Y422	将颜色空间从 RGB 转换为 YUV422 UYVY（Y422、UYNV）。
TCVN_CST_BGR_TO_YUV_422_Y422	将颜色空间从 BGR 转换为 YUV422 UYVY（Y422、UYNV）。
TCVN_CST_RGB_TO_YUV_422_UYNV	将颜色空间从 RGB 转换为 YUV422 UYVY（Y422、UYNV）。
TCVN_CST_BGR_TO_YUV_422_UYNV	将颜色空间从 BGR 转换为 YUV422 UYVY（Y422、UYNV）。
TCVN_CST_RGBA_TO_YUV_422_UYVY	将颜色空间从 RGBA 转换为 YUV422 UYVY（Y422、UYNV）。
TCVN_CST_BGRA_TO_YUV_422_UYVY	将颜色空间从 BGRA 转换为 YUV422 UYVY（Y422、UYNV）。
TCVN_CST_RGBA_TO_YUV_422_Y422	将颜色空间从 RGBA 转换为 YUV422 UYVY（Y422、UYNV）。

名称	描述
TCVN_CST_BGRA_TO_YUV_422_Y422	将颜色空间从 BGRA 转换为 YUV422 UYVY（Y422、UYNV）。
TCVN_CST_RGBA_TO_YUV_422_UYNV	将颜色空间从 RGBA 转换为 YUV422 UYVY（Y422、UYNV）。
TCVN_CST_BGRA_TO_YUV_422_UYNV	将颜色空间从 BGRA 转换为 YUV422 UYVY（Y422、UYNV）。
TCVN_CST_RGB_TO_YUV_422_YUY2	将颜色空间从 RGB 转换为 YUV422 YUY2（YUYV、YUNV）。
TCVN_CST_BGR_TO_YUV_422_YUY2	将颜色空间从 BGR 转换为 YUV422 YUY2（YUYV、UYNV）。
TCVN_CST_RGB_TO_YUV_422_YVYU	将颜色空间从 RGB 转换为 YUV422 YVYU。
TCVN_CST_BGR_TO_YUV_422_YVYU	将颜色空间从 BGR 转换为 YUV422 YVYU。
TCVN_CST_RGB_TO_YUV_422_YUYV	将颜色空间从 RGB 转换为 YUV422 YUY2（YUYV、YUNV）。
TCVN_CST_BGR_TO_YUV_422_YUYV	将颜色空间从 BGR 转换为 YUV422 YUY2（YUYV、UYNV）。
TCVN_CST_RGB_TO_YUV_422_YUNV	将颜色空间从 RGB 转换为 YUV422 YUY2（YUYV、YUNV）。
TCVN_CST_BGR_TO_YUV_422_YUNV	将颜色空间从 BGR 转换为 YUV422 YUY2（YUYV、UYNV）。
TCVN_CST_RGBA_TO_YUV_422_YUY2	将颜色空间从 RGBA 转换为 YUV422 YUY2（YUYV、YUNV）。
TCVN_CST_BGRA_TO_YUV_422_YUY2	将颜色空间从 BGRA 转换为 YUV422 YUY2（YUYV、UYNV）。
TCVN_CST_RGBA_TO_YUV_422_YVYU	将颜色空间从 RGBA 转换为 YUV422 YVYU。
TCVN_CST_BGRA_TO_YUV_422_YVYU	将颜色空间从 BGRA 转换为 YUV422 YVYU。
TCVN_CST_RGBA_TO_YUV_422_YUYV	将颜色空间从 RGBA 转换为 YUV422 YUY2（YUYV、YUNV）。
TCVN_CST_BGRA_TO_YUV_422_YUYV	将颜色空间从 BGRA 转换为 YUV422 YUY2（YUYV、UYNV）。
TCVN_CST_RGBA_TO_YUV_422_YUNV	将颜色空间从 RGBA 转换为 YUV422 YUY2（YUYV、YUNV）。
TCVN_CST_BGRA_TO_YUV_422_YUNV	将颜色空间从 BGRA 转换为 YUV422 YUY2（YUYV、UYNV）。
TCVN_CST_MAX	仅供内部使用，在添加新值时进行调整。

相关函数

- F_VN_ConvertColorSpace [▶ 1248]

6.1.2.4.26 ETcVnColorTrainingMethod

提供颜色训练方法。

语法

定义：

```
TYPE ETcVnColorTrainingMethod :
(
    TCVN_CTM_LAB := 0,
    TCVN_CTM_RGB := 1
)DINT;
END_TYPE
```

值

名称	描述
TCVN_CTM_LAB	在 CIE L*a*b* 颜色空间中训练颜色。
TCVN_CTM_RGB	在 RGB 颜色空间中训练颜色。

相关函数

- F_VN_TrainImageColorExp [▶ 1268]

6.1.2.4.27 ETcVnComparisonOperator

Offers comparison operators for the SortContainer function.

Syntax

Definition:

```
TYPE ETcVnComparisonOperator :
(
    TCVN_CO_LESS    := 0,
    TCVN_CO_GREATER := 1
)INT;
END_TYPE
```

Values

Name	Description
TCVN_CO_LESS	Sort values ascending
TCVN_CO_GREATER	Sort values descending

6.1.2.4.28 ETcVnConnectedComponentsAlgorithm

提供连通分量算法。

语法

定义：

```
TYPE ETcVnConnectedComponentsAlgorithm :
(
    TCVN_CCA_WU    := 0,
    TCVN_CCA_GRANA := 1
)DINT;
END_TYPE
```

值

名称	描述
TCVN_CCA_WU	SAUF 算法
TCVN_CCA_GRANA	BBDT 算法用于 8 连通，SAUF 算法用于 4 连通

6.1.2.4.29 ETcVnContainerExportFormat

提供容器导出格式。

语法

定义：

```
TYPE ETcVnContainerExportFormat :
(
    TCVN_CEF_XML            := 0,
    TCVN_CEF_XML_SERIALIZED := 1,
    TCVN_CEF_CSV            := 2,
```

```
    TCVN_CEF_BINARY         := 3
)UINT;
END_TYPE
```

值

名称	描述
TCVN_CEF_XML	可人工读取数据的 XML 格式
TCVN_CEF_XML_SERIALIZED	序列化数据的 XML 格式
TCVN_CEF_CSV	CSV 格式（仅限于二维表示，因此不适用于所有容器类型）
TCVN_CEF_BINARY	二值序列化数据（建议用于大型容器）

相关元素

- FB_VN_WriteContainer [▶ 1595]

6.1.2.4.30 ETcVnContourApproximationMethod

提供轮廓近似法。

语法

定义：

```
TYPE ETcVnContourApproximationMethod :
(
    TCVN_CAM_NONE       := 1,
    TCVN_CAM_SIMPLE     := 2,
    TCVN_CAM_TC89_L1    := 3,
    TCVN_CAM_TC89_KCOS  := 4
)DINT;
END_TYPE
```

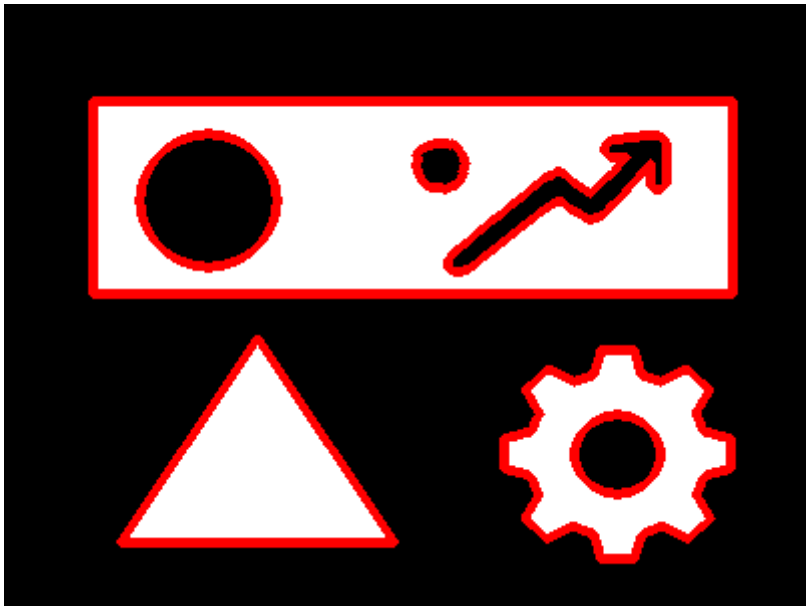
值

名称	描述
TCVN_CAM_NONE	没有近似值，每一个点都被存储，即随后的两个点总是直接相邻（水平、垂直或对角线）。
TCVN_CAM_SIMPLE	对水平、垂直或对角线方向上形成直线的线段进行无损压缩，只保留其端点（例如，将一个直立的矩形压缩为四个角点）。
TCVN_CAM_TC89_L1	使用 Teh-Chin 链近似算法进行压缩（IEEE-paper C. Teh, R. Chin, On the Detection of Dominant Points on Digital Curves, 1989）。
TCVN_CAM_TC89_KCOS	使用 Teh-Chin 链近似算法进行压缩（IEEE-paper C. Teh, R. Chin, On the Detection of Dominant Points on Digital Curves, 1989）。

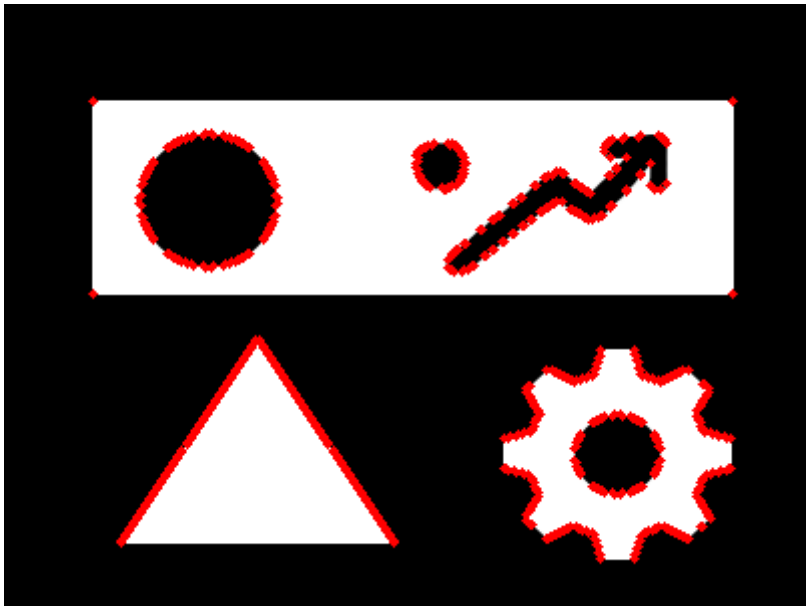
更多信息

这种类型表明是否要简化描述轮廓的点以及如果需要，如何简化。

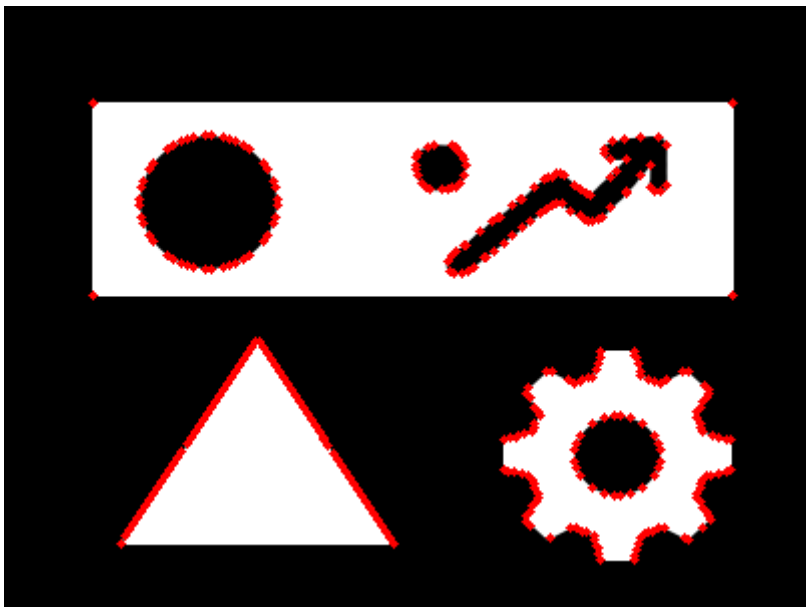
- **TCVN_CAM_NONE**
  未对轮廓描述进行简化，轮廓由所有包围点描述。



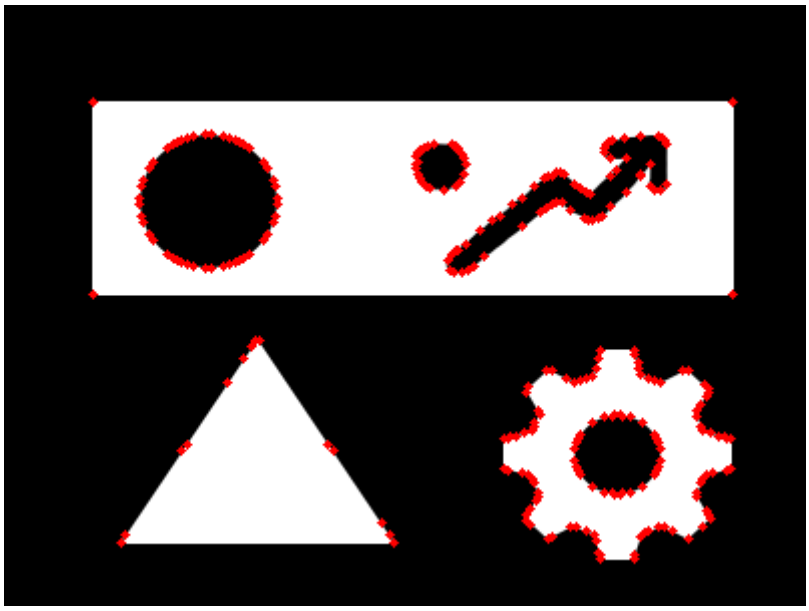
- **TCVN_CAM_SIMPLE**
  无损简化



- **TCVN_CAM_TC89_L1**
  使用 Teh-Chin 链近似算法简化轮廓点（IEEE-paper C. Teh, R. Chin, On the Detection of Dominant Points on Digital Curves, 1989）。



- **TCVN_CAM_TC89_KCOS**
  使用 Teh-Chin 链近似算法简化轮廓点（IEEE-paper C. Teh, R. Chin, On the Detection of Dominant Points on Digital Curves, 1989）。



相关函数

- F_VN_FindContoursExp [▶ 1181]
- F_VN_FindContourHierarchyExp [▶ 1175]

6.1.2.4.31 ETcVnContourRetrievalMode

提供轮廓搜索的检索模式。

语法

定义：

```
TYPE ETcVnContourRetrievalMode :
(
    TCVN_CRM_EXTERNAL             := 0,
    TCVN_CRM_LIST                 := 1,
    TCVN_CRM_CONNECTED_COMPONENTS := 2,
    TCVN_CRM_TREE                 := 3,
    TCVN_CRM_FLOODFILL            := 4
)DINT;
END_TYPE
```

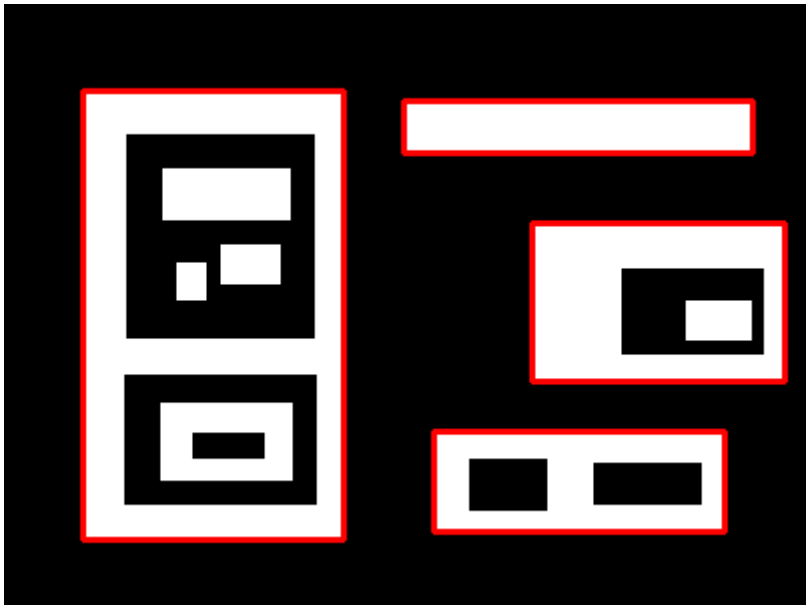
值

名称	描述
TCVN_CRM_EXTERNAL	只返回外部轮廓。
TCVN_CRM_LIST	返回所有找到的轮廓，不考虑其层级。
TCVN_CRM_CONNECTED_COMPONENTS	返回所有具有 2 级层次结构的轮廓（外部轮廓分配为 0 级，内部轮廓为 1 级）。
TCVN_CRM_TREE	返回所有轮廓及其完整层级。
TCVN_CRM_FLOODFILL	返回使用 分水岭算法找到的轮廓（仅适用于 DINT 图像）。

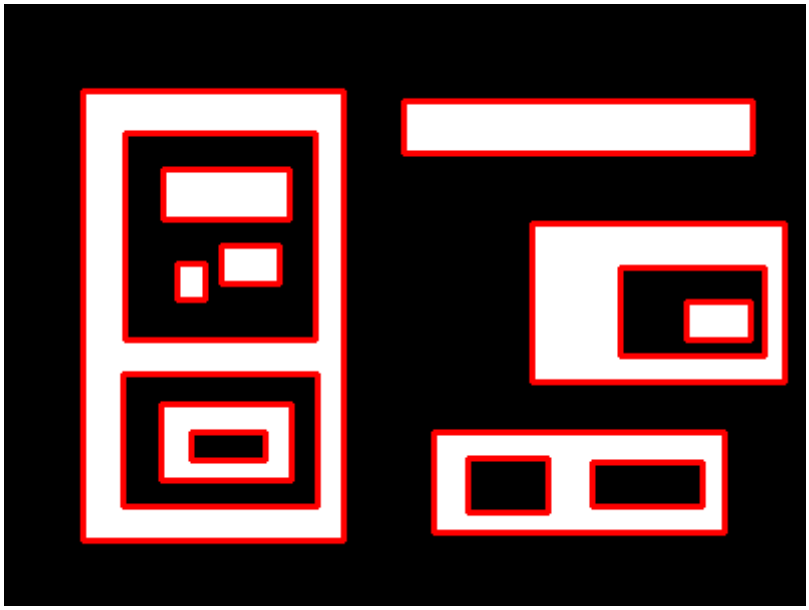
更多信息

该类型指定了在轮廓搜索中如何考虑轮廓层次的问题。

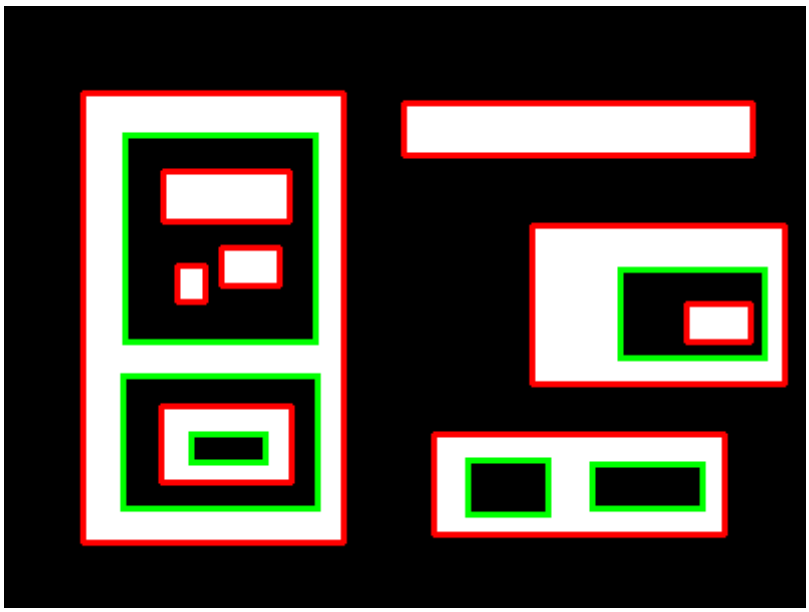
- **TCVN_CRM_EXTERNAL**
  只返回外部轮廓，不考虑内部轮廓。



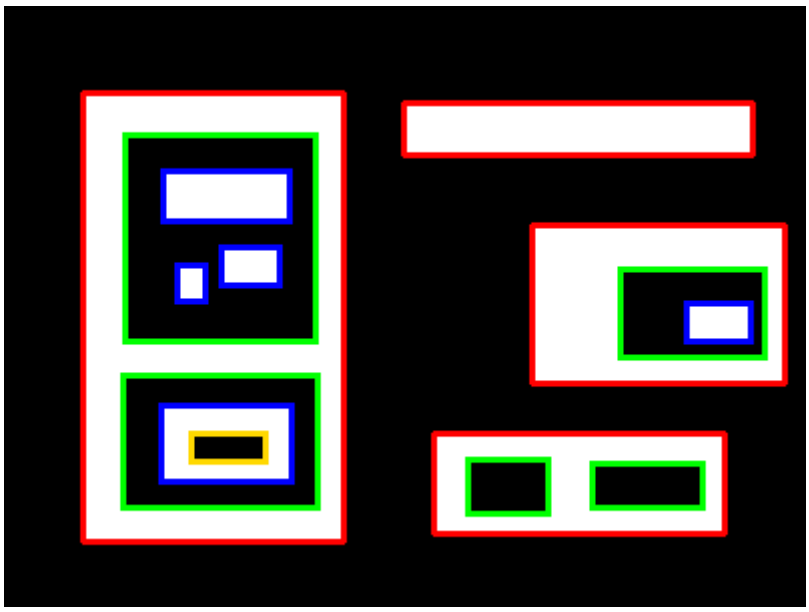
- **TCVN_CRM_LIST**
  所有轮廓都被返回。轮廓层次未考虑在内。



- **TCVN_CRM_CONNECTED_COMPONENTS**
  返回所有的轮廓，并创建一个 2 级层次结构。在图形中，1 级的所有轮廓线都显示为红色，2 级显示为绿色。很明显，在 2 级轮廓之后的下一个内部轮廓又是在 1 级。



- **TCVN_CRM_TREE**
  返回所有的轮廓，并根据树状结构创建一个层次结构。在图像中，1 级是红色，2 级是绿色，3 级是蓝色，4 级是黄色。



相关函数

- F_VN_FindContoursExp [▶ 1181]
- F_VN_FindContourHierarchyExp [▶ 1175]

6.1.2.4.32 ETcVnContoursMatchComparisonMethod

提供轮廓匹配的比较方法。

语法

定义：

```
TYPE ETcVnContoursMatchComparisonMethod :
(
    TCVN_CMCM_CONTOURS_MATCH_I1 := 1,
    TCVN_CMCM_CONTOURS_MATCH_I2 := 2,
    TCVN_CMCM_CONTOURS_MATCH_I3 := 3
)DINT;
END_TYPE
```

值

名称	描述
TCVN_CMCM_CONTOURS_MATCH_I1	使用对等单个特征之间的差异总和进行轮廓比较。
TCVN_CMCM_CONTOURS_MATCH_I2	使用单个特征之间的差异总和进行轮廓比较。
TCVN_CMCM_CONTOURS_MATCH_I3	仅使用单个特征之间的最大差值进行轮廓比较。

6.1.2.4.33 ETcVnDataLayout

提供图像数据布局。

语法

定义：

```
TYPE ETcVnDataLayout :
(
    TCVN_DL_DEFAULT := 0x0,
    TCVN_DL_2D_HW   := 0x12,
```

```
    TCVN_DL_2D_WH   := 0x21,
    TCVN_DL_3D_CHW  := 0x123,
    TCVN_DL_3D_HWC  := 0x231,
    TCVN_DL_4D_NCHW := 0x1234,
    TCVN_DL_4D_NHWC := 0x1342
)ULINT;
END_TYPE
```

值

名称	描述
TCVN_DL_DEFAULT	默认值
TCVN_DL_2D_HW	2D 数据 HW，多通道/单通道图像
TCVN_DL_2D_WH	2D 数据 WH，多通道/单通道图像
TCVN_DL_3D_CHW	3D 数据 CHW，单通道图像
TCVN_DL_3D_HWC	3D 数据 HWC，单通道图像
TCVN_DL_4D_NCHW	4D 数据 NCHW，单通道图像
TCVN_DL_4D_NHWC	4D 数据 NHWC，单通道图像

6.1.2.4.34 ETcVnDataMatrixOptions

Offers Data Matrix code detection options.

Syntax

Definition:

```
TYPE ETcVnDataMatrixOptions :
(
    TCVN_DMO_DEFAULT          := 0,
    TCVN_DMO_BINARY_INPUT     := 1,
    TCVN_DMO_GLOBAL_THRESHOLD := 2,
    TCVN_DMO_BINARY_GRID      := 4,
    TCVN_DMO_STATIC_GRID      := 8,
    TCVN_DMO_REGULAR_COLOR    := 16,
    TCVN_DMO_INVERTED_COLOR   := 32
)ULINT;
END_TYPE
```

Values

Name	Description
TCVN_DMO_DEFAULT	Expects a graylevel image as input. Applies an adaptive local threshold and creates a dynamic grid based on the clock pattern.
TCVN_DMO_BINARY_INPUT	Expects a binary image as input (all pixels must contain 0 or 255, disables internal preprocessing)
TCVN_DMO_GLOBAL_THRESHOLD	Applies a global instead of an adaptive local threshold (requires uniform illumination)
TCVN_DMO_BINARY_GRID	Create the dynamic grid based on the binary instead of the graylevel image
TCVN_DMO_STATIC_GRID	Use an equally spaced static grid instead of a dynamic one
TCVN_DMO_REGULAR_COLOR	Only search for regular color (dark code on light background)
TCVN_DMO_INVERTED_COLOR	Only search for inverted color (light code on dark background)

6.1.2.4.35 ETcVnDataMatrixOptions

提供数据矩阵编码检测选项。

语法

定义：

```
TYPE ETcVnDataMatrixOptions :
(
    TCVN_DMO_DEFAULT          := 0,
    TCVN_DMO_BINARY_INPUT     := 1,
    TCVN_DMO_GLOBAL_THRESHOLD := 2,
    TCVN_DMO_BINARY_GRID      := 4,
    TCVN_DMO_STATIC_GRID      := 8,
    TCVN_DMO_REGULAR_COLOR    := 16,
    TCVN_DMO_INVERTED_COLOR   := 32
)ULINT;
END_TYPE
```

值

名称	描述
TCVN_DMO_DEFAULT	预期将灰度图像作为输入。应用自适应局部阈值，并根据时钟模式创建动态网格。
TCVN_DMO_BINARY_INPUT	预期将二进制图像作为输入（所有像素必须包含 0 或 255，禁用内部预处理）
TCVN_DMO_GLOBAL_THRESHOLD	应用全局阈值而非自适应局部阈值（需要均匀照度）
TCVN_DMO_BINARY_GRID	根据二进制图像而非灰度图像创建动态网格
TCVN_DMO_STATIC_GRID	使用等间距的静态网格，而非动态网格
TCVN_DMO_REGULAR_COLOR	仅搜索常规颜色（浅色背景上的深色编码）
TCVN_DMO_INVERTED_COLOR	仅搜索反色（深色背景上的浅色编码）

6.1.2.4.36 ETcVnDetectPatternPointsOptions

提供 DetectPatternPoints2 的选项。

语法

定义：

```
TYPE ETcVnDetectPatternPointsOptions :
(
    TCVN_DPPO_SYMMETRIC  := 1,
    TCVN_DPPO_ASYMMETRIC := 2,
    TCVN_DPPO_CLUSTERING := 4
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_DPPO_SYMMETRIC	对称网格
TCVN_DPPO_ASYMMETRIC	非对称网格
TCVN_DPPO_CLUSTERING	使用聚类算法进行网格检测（对透视几何畸变的适应性较强，但对背景杂乱的适应性较弱）

6.1.2.4.37 ETcVnDiffusivityTypeKAZE

提供用于特征检测算法 KAZE 和 AKAZE 的扩散类型。

语法

定义：

```
TYPE ETcVnDiffusivityTypeKAZE :
(
    TCVN_DT1_KAZE_PM_G1       := 0,
    TCVN_DT1_KAZE_PM_G2       := 1,
    TCVN_DT1_KAZE_WEICKERT    := 2,
```

```
    TCVN_DT1_KAZE_CHARBONNIER := 3
)DINT;
END_TYPE
```

值

名称	描述
TCVN_DT1_KAZE_PM_G1	使用 Perona-Malik 扩散滤波 g1。
TCVN_DT1_KAZE_PM_G2	使用 Perona-Malik 扩散滤波 g2。
TCVN_DT1_KAZE_WEICKERT	使用 Weickert 扩散滤波。
TCVN_DT1_KAZE_CHARBONNIER	使用 Charbonnier 扩散滤波。

更多信息

枚举`ETcVnDiffusivityTypeKAZE`用于结构TcVnParamsKAZE [▶ 232]。

相关函数

- F_VN_KeyPointsAndDescriptorsKAZEExp [▶ 1347]

6.1.2.4.38 ETcVnDistanceTransformationLabel

提供要构建的标签数组类型。

语法

定义：

```
TYPE ETcVnDistanceTransformationLabel :
(
    TCVN_DTL_CCOMP := 0,
    TCVN_DTL_PIXEL := 1
)DINT;
END_TYPE
```

值

名称	描述
TCVN_DTL_CCOMP	标记连通分量（源图像中零值的每个连通分量和最靠近连通分量的像素都将被分配相同的标签）。
TCVN_DTL_PIXEL	标记像素（每个零值像素和最靠近它的非零像素都有自己的标签）。

相关函数

- F_VN_DistanceTransformationExp [▶ 1220]

6.1.2.4.39 ETcVnDistanceTransformationMask

提供不同大小的距离转换掩膜。请注意，某些距离类型不支持某些设置 sizes 的数值大小。

语法

定义：

```
TYPE ETcVnDistanceTransformationMask :
(
    TCVN_DTM_PRECISE := 0,
    TCVN_DTM_3       := 3,
    TCVN_DTM_5       := 5
)DINT;
END_TYPE
```

值

名称	描述
TCVN_DTM_PRECISE	使用精确掩膜。
TCVN_DTM_3	使用 3x3 掩膜。
TCVN_DTM_5	使用 5x5 掩膜。

相关函数

- F_VN_DistanceTransformation(Exp) [▶ 1218]

6.1.2.4.40 ETcVnDistanceType

提供距离类型。

语法

定义：

```
TYPE ETcVnDistanceType :
(
    TCVN_DT_USER   := -1,
    TCVN_DT_L1     := 1,
    TCVN_DT_L2     := 2,
    TCVN_DT_C      := 3,
    TCVN_DT_L12    := 4,
    TCVN_DT_FAIR   := 5,
    TCVN_DT_WELSCH := 6,
    TCVN_DT_HUBER  := 7
)DINT;
END_TYPE
```

值

名称	描述
TCVN_DT_USER	用户自定义距离
TCVN_DT_L1	|x1 - x2| + |y1 - y2|
TCVN_DT_L2	欧几里得（Euclidean）距离
TCVN_DT_C	（|x1 - x2|, |y1 - y2|）最大值
TCVN_DT_L12	L1 - L2 : 2*(sqrt(1 + x*x/2) - 1))
TCVN_DT_FAIR	c^2(|x|/c-log(1+|x|/c))，c = 1.3998
TCVN_DT_WELSCH	c^2/2(1-exp(-(x/c)^2))，c = 2.9846
TCVN_DT_HUBER	|x|<c ? x^2/2 : c(|x|-c/2)，c=1.345

相关函数

- F_VN_DistanceTransformation(Exp) [▶ 1218]
- F_VN_FitLineExp [▶ 985]

6.1.2.4.41 ETcVnDotCodeOptions

提供 DotCode （点码）检测选项。

语法

定义：

```
TYPE ETcVnDotCodeOptions :
(
    TCVN_DCO_DEFAULT         := 0,
    TCVN_DCO_BINARY_INPUT    := 1,
    TCVN_DCO_DETECTBLOBS     := 2,
    TCVN_DCO_FILTERCONTOURS  := 4,
    TCVN_DCO_REGULAR_COLOR   := 16,
```

```
    TCVN_DCO_INVERTED_COLOR := 32
)ULINT;
END_TYPE
```

值

名称	描述
TCVN_DCO_DEFAULT	以灰度图像作为输入，并应用默认算法（类似于参考解码算法）
TCVN_DCO_BINARY_INPUT	预期将二进制图像作为输入（所有像素必须包含 0 或 255，禁用内部预处理）
TCVN_DCO_DETECTBLOBS	通过使用Blob检测来查找点，从而修改默认算法（无法与 BINARY_INPUT 或 FILTERCONTOURS 组合使用）
TCVN_DCO_FILTERCONTOURS	作为预处理步骤，移除与图像边界相连以及尺寸明显错误（相对于所提供的 fModuleWidth 而言）的轮廓
TCVN_DCO_REGULAR_COLOR	仅搜索常规颜色（浅色背景上的深色编码）
TCVN_DCO_INVERTED_COLOR	仅搜索反色（深色背景上的浅色编码）

6.1.2.4.42 ETcVnDrawMatchesFlags

提供多种标志组合，支持重绘制现有目标图像和/或跳过单个（即不匹配）关键点和/或绘制附加（丰富）关键点信息（大小和方向）。由 F_VN_DrawKeypointsExp 使用。

语法

定义：

```
TYPE ETcVnDrawMatchesFlags :
(
    TCVN_DMF_DEFAULT                          := 0,
    TCVN_DMF_OVERDRAW                         := 1,
    TCVN_DMF_SKIPSINGLE                       := 2,
    TCVN_DMF_OVERDRAW_SKIPSINGLE              := 3,
    TCVN_DMF_RICHKEYPOINT                     := 4,
    TCVN_DMF_OVERDRAW_RICHKEYPOINT            := 5,
    TCVN_DMF_SKIPSINGLE_RICHKEYPOINT          := 6,
    TCVN_DMF_OVERDRAW_SKIPSINGLE_RICHKEYPOINT := 7
)DINT;
END_TYPE
```

值

名称	描述
TCVN_DMF_DEFAULT	将所有关键点绘制到新图像中。
TCVN_DMF_OVERDRAW	将所有关键点绘制到现有的目标图像中。
TCVN_DMF_SKIPSINGLE	将关键点绘制到新图像中，但跳过单个关键点。
TCVN_DMF_OVERDRAW_SKIPSINGLE	将关键点绘制到现有目标图像中，但跳过单个关键点。
TCVN_DMF_RICHKEYPOINT	将包含丰富信息的所有关键点绘制到新图像中。
TCVN_DMF_OVERDRAW_RICHKEYPOINT	将包含丰富信息的所有关键点绘制到现有的目标图像中。
TCVN_DMF_SKIPSINGLE_RICHKEYPOINT	将包含丰富信息的关键点绘制到新图像中，但跳过单个关键点。
TCVN_DMF_OVERDRAW_SKIPSINGLE_RICHKEYPOINT	将包含丰富信息的关键点绘制到现有目标图像中，但跳过单个关键点。

相关函数

- F_VN_DrawKeypointsExp [▶ 1018]
- F_VN_DrawMatchesExp [▶ 1029]

6.1.2.4.43 ETcVnDrawShape

提供要绘制的形状。

语法

定义：

```
TYPE ETcVnDrawShape :
(
    TCVN_DS_RANDOM  := -1,
    TCVN_DS_CIRCLE  := 0,
    TCVN_DS_SQUARE  := 1,
    TCVN_DS_PLUS    := 2,
    TCVN_DS_X       := 3,
    TCVN_DS_DIAMOND := 4,
    TCVN_DS_MAX     := 5
)DINT;
END_TYPE
```

值

名称	描述
TCVN_DS_RANDOM	随机选择圆形、正方形、加号、X 或菱形。
TCVN_DS_CIRCLE	圆形
TCVN_DS_SQUARE	正方形
TCVN_DS_PLUS	+
TCVN_DS_X	x
TCVN_DS_DIAMOND	菱形
TCVN_DS_MAX	仅供内部使用，在添加新值时进行调整。

相关函数

- F_VN_DrawPoint(Exp) [▶ 1033]
- F_VN_DrawPoints(Exp) [▶ 1035]

6.1.2.4.44 ETcVnEdgeDetectionAlgorithm

提供边缘检测算法。

语法

定义：

```
TYPE ETcVnEdgeDetectionAlgorithm :
(
    TCVN_EDA_INTERPOLATION   := 0,
    TCVN_EDA_APPROX_ERF      := 1,
    TCVN_EDA_APPROX_GAUSSIAN := 2
)DINT;
END_TYPE
```

值

名称	描述
TCVN_EDA_INTERPOLATION	内插像素（双线性），然后找出最大梯度。这种方法快速、稳定，但通常不如函数近似算法精确。
TCVN_EDA_APPROX_ERF	通过 erf 函数近似边缘。这种方法比插值法慢，但通常更精确。但如果边缘不适合 ERF 模型，它就可能不准确。
TCVN_EDA_APPROX_GAUSSIAN	通过高斯函数近似边缘。这种方法主要用于找到相对较细线条的中心点，因此对于其它边缘可能不太准确。

相关函数

- 测量
  - F_VN_LocateCircularArcExp [▶ 1465]
  - F_VN_LocateEdgeExp [▶ 1471]
  - F_VN_LocateEdgesExp [▶ 1479]
  - F_VN_LocateEllipseExp [▶ 1486]
  - F_VN_MeasureAngleBetweenEdgesExp [▶ 1492]
  - F_VN_MeasureEdgeDistanceExp [▶ 1499]
  - F_VN_MeasureMinEdgeDistanceExp [▶ 1508]

6.1.2.4.45 ETcVnEdgeDirection

提供边缘的参考搜索方向。

语法

定义：

```
TYPE ETcVnEdgeDirection :
(
    TCVN_ED_DARK_TO_LIGHT := 0,
    TCVN_ED_LIGHT_TO_DARK := 1
)DINT;
END_TYPE
```

值

名称	描述
TCVN_ED_DARK_TO_LIGHT	从深到浅
TCVN_ED_LIGHT_TO_DARK	从浅到深

相关函数

- 测量

6.1.2.4.46 ETcVnElementType

提供元素类型。

语法

定义：

```
TYPE ETcVnElementType :
(
    TCVN_ET_SAME_AS_SOURCE := -1,
    TCVN_ET_USINT          := 0,
    TCVN_ET_SINT           := 1,
    TCVN_ET_UINT           := 2,
    TCVN_ET_INT            := 3,
    TCVN_ET_DINT           := 4,
    TCVN_ET_REAL           := 5,
    TCVN_ET_LREAL          := 6
)DINT;
END_TYPE
```

值

名称	描述
TCVN_ET_SAME_AS_SOURCE	将目标图像的元素类型设置为源图像的元素类型。
TCVN_ET_USINT	USINT（位深：8 位，0...255）
TCVN_ET_SINT	SINT（位深：8 位，-128...127）

名称	描述
TCVN_ET_UINT	UINT（位深：16 位，0...65535）
TCVN_ET_INT	INT（位深：16 位，-32768...32767）
TCVN_ET_DINT	DINT（位深：32 位，-2147483648...2147483647）
TCVN_ET_REAL	REAL（位深：32 位， ~ -3.402823E 10^38.. ~ 3.402823E 38)
TCVN_ET_LREAL	LREAL（深度：64 位， ~ -1.79769313486231E 308 .. ~ 1.79769313486232E 308)

更多信息

枚举`ETcVnElementType`主要用于指定图像的像素类型或位深。此外，它还包含在结构TcVnMatrix [▶ 226]中。

相关函数

- F_VN_ConvertElementType(Exp) [▶ 775]
- F_VN_CreateImage [▶ 784]
- F_VN_CreateImageAndSetPixels [▶ 785]
- F_VN_CreateImageFromArray [▶ 786]
- F_VN_CustomFilter(Exp) [▶ 1287]
- F_VN_InitMatrixStruct [▶ 1524]
- F_VN_LaplacianFilter(Exp) [▶ 1297]
- F_VN_NormalizeImageExp [▶ 1258]
- F_VN_ScharrFilter(Exp) [▶ 1306]
- F_VN_SeparableCustomFilter(Exp) [▶ 1309]
- F_VN_SobelFilter(Exp) [▶ 1312]

6.1.2.4.47 ETcVnEstimationAlgorithm

提供匹配点集的估计算法。

语法

定义：

```
TYPE ETcVnEstimationAlgorithm :
(
    TCVN_EA_DEFAULT := 0,
    TCVN_EA_LMEDS   := 4,
    TCVN_EA_RANSAC  := 8,
    TCVN_EA_RHO     := 16
)DINT;
END_TYPE
```

值

名称	描述
TCVN_EA_DEFAULT	使用所有点。
TCVN_EA_LMEDS	最小平方中值
TCVN_EA_RANSAC	随机抽样一致
TCVN_EA_RHO	渐进采样一致

相关函数

- F_VN_FindReferenceKeyPointsInImageORBExp [▶ 1336]
- F_VN_HomographyExp [▶ 1141]

6.1.2.4.48 ETcVnExtremePointDirection

提供极值点的搜索方向。

语法

定义：

```
TYPE ETcVnExtremePointDirection :
(
    TCVN_EPD_TOP_LEFT      := 0,
    TCVN_EPD_TOP_MEDIAN    := 1,
    TCVN_EPD_TOP_RIGHT     := 2,
    TCVN_EPD_BOTTOM_LEFT   := 3,
    TCVN_EPD_BOTTOM_MEDIAN := 4,
    TCVN_EPD_BOTTOM_RIGHT  := 5,
    TCVN_EPD_LEFT_TOP      := 6,
    TCVN_EPD_LEFT_MEDIAN   := 7,
    TCVN_EPD_LEFT_BOTTOM   := 8,
    TCVN_EPD_RIGHT_TOP     := 9,
    TCVN_EPD_RIGHT_MEDIAN  := 10,
    TCVN_EPD_RIGHT_BOTTOM  := 11
)DINT;
END_TYPE
```

值

名称	描述
TCVN_EPD_TOP_LEFT	找到最高点（最小 y，如果不止 1 个，则取左边的点）。
TCVN_EPD_TOP_MEDIAN	找到最高点（最小 y，如果不止 1 个，则取中值）。
TCVN_EPD_TOP_RIGHT	找到最高点（最小 y，如果不止 1 个，则取右边的点）。
TCVN_EPD_BOTTOM_LEFT	找到最低点（最大 y，如果不止 1 个，则取左边的点）。
TCVN_EPD_BOTTOM_MEDIAN	找出最低点（最大 y，如果不止 1 个，则取中值）。
TCVN_EPD_BOTTOM_RIGHT	找出最低点（最大 y，如果不止 1 个，则取右边的点）。
TCVN_EPD_LEFT_TOP	找出最左边的点（最小 x，如果不止 1 个，则取最高点）。
TCVN_EPD_LEFT_MEDIAN	找出最左边的点（最小 x，如果不止 1 个，则取中值）。
TCVN_EPD_LEFT_BOTTOM	找出最左边的点（最小 x，如果不止 1 个，则取最低点）。
TCVN_EPD_RIGHT_TOP	找到最右边的点（最大 x，如果不止 1 个，则取最高点）。
TCVN_EPD_RIGHT_MEDIAN	找出最右边的点（最大 x，如果不止 1 个，则取中值）。
TCVN_EPD_RIGHT_BOTTOM	找出最右边的点（最大 x，如果不止 1 个，则取最低点）。

相关函数

- F_VN_ContourExtremePoint [▶ 970]

6.1.2.4.49 ETcVnFeatureDescriptorTypeAKAZE

为 AKAZE 算法提供描述符类型。

语法

定义：

```
TYPE ETcVnFeatureDescriptorTypeAKAZE :
(
    TCVN_FDT_AKAZE_KAZE_UPRIGHT := 2,
    TCVN_FDT_AKAZE_KAZE         := 3,
    TCVN_FDT_AKAZE_MLDB_UPRIGHT := 4,
    TCVN_FDT_AKAZE_MLDB         := 5
)DINT;
END_TYPE
```

值

名称	描述
TCVN_FDT_AKAZE_KAZE_UPRIGHT	KAZE 描述符（upright，即旋转依赖性）。
TCVN_FDT_AKAZE_KAZE	KAZE 描述符（旋转不变性）。
TCVN_FDT_AKAZE_MLDB_UPRIGHT	MLDB 描述符（upright，即旋转依赖性）。
TCVN_FDT_AKAZE_MLDB	MLDB 描述符（旋转不变性）。

更多信息

枚举`ETcVnFeatureDescriptorTypAKAZE`用于结构TcVnParamsAKAZE [▶ 228]。

6.1.2.4.50 ETcVnFeatureScalingType

提供特征缩放类型

语法

定义：

```
TYPE ETcVnFeatureScalingType :
(
    TCVN_FST1_MAXABS          := 0,
    TCVN_FST1_MINMAX          := 1,
    TCVN_FST1_STANDARDIZATION := 2
)DINT;
END_TYPE
```

值

名称	描述
TCVN_FST1_MAXABS	使用特征向量绝对值的最大值。如果所有原始值都是正值，则归一化值的范围为 [-1 至 1] 或 [0 至 1]。
TCVN_FST1_MINMAX	使用特征向量的最小值和最大值。归一化值范围在 [0 至 1] 之间
TCVN_FST1_STANDARDIZATION	使用特征向量的平均值和标准偏差。归一化向量的平均值等于零，标准偏差等于一

6.1.2.4.51 ETcVnFeatureScoreTypeORB

提供用于对特征进行排序的算法。

语法

定义：

```
TYPE ETcVnFeatureScoreTypeORB :
(
    TCVN_FST_ORB_HARRIS := 0,
    TCVN_FST_ORB_FAST   := 1
)DINT;
END_TYPE
```

值

名称	描述
TCVN_FST_ORB_HARRIS	Harris 算法（产生的关键点更稳定，但计算时间更长）。
TCVN_FST_ORB_FAST	FAST 算法（产生的关键点稳定性稍差，但计算速度更快）。

更多信息

枚举`ETcVnFeatureScoreTypeORB`用于结构TcVnParamsORB [▶ 233]。

6.1.2.4.52 ETcVnFilterDirection

提供应用滤波的方向。

语法

定义：

```
TYPE ETcVnFilterDirection :
(
    TCVN_FD_X := 0,
    TCVN_FD_Y := 1
)INT;
END_TYPE
```

值

名称	描述
TCVN_FD_X	X-方向
TCVN_FD_Y	Y-方向

相关函数

- F_VN_ScharrFilter(Exp) [▶ 1306]

6.1.2.4.53 ETcVnFindTemplateWalshOptions

提供标志，用于自定义基于Walsh变换的模板搜索。

语法

定义：

```
TYPE ETcVnFindTemplateWalshOptions :
(
    TCVN_FTWO_NONE                 := 0,
    TCVN_FTWO_SKIP_DC              := 2,
    TCVN_FTWO_WD_STOP_CORRECTION   := 256,
    TCVN_FTWO_FUSE_MATCHES         := 512,
    TCVN_FTWO_ROTATED_RECTANGLE    := 1024,
    TCVN_FTWO_MATCH_OPPOSITE_RANGE := 65536
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_FTWO_NONE	不使用任何特殊选项。
TCVN_FTWO_SKIP_DC	在相似度计算期间跳过 DC 部分。DC 或“直流”分量是Walshd第一个系数（全为+1的方波序列）。与其他分量相比，将其用于投影会捕获总像素强度，而非空间差异。因此，跳过它可提高对光源变化的适应性。

名称	描述
TCVN_FTWO_WD_STOP_CORRECTION	使用试错式方法，在 Watchdog（看门狗）停止时调整匹配阈值
TCVN_FTWO_FUSE_MATCHES	融合不同旋转角度且位置相近的匹配项。
TCVN_FTWO_ROTATED_RECTANGLE	以 ContainerType_Vector_TcVnRotatedRectangle 类型的容器返回结果
TCVN_FTWO_MATCH_OPPOSITE_RANGE	使用给定的旋转角度以及与之相反的旋转角度进行匹配（仅可与基于范围的函数配合使用）

6.1.2.4.54 ETcVnFlipAxis

定义图像翻转（镜像）的轴线。

语法

定义：

```
TYPE ETcVnFlipAxis :
(
    TCVN_FA_XY := -1,
    TCVN_FA_X  := 0,
    TCVN_FA_Y  := 1
)INT;
END_TYPE
```

值

名称	描述
TCVN_FA_XY	绕 XY 两个轴翻转图像
TCVN_FA_X	绕 X 轴翻转图像
TCVN_FA_Y	绕 Y 轴翻转图像

相关函数

- F_VN_FlipImage [▶ 1135]

6.1.2.4.55 ETcVnFontType

提供字体类型。

语法

定义：

```
TYPE ETcVnFontType :
(
    TCVN_FT_HERSHEY_SIMPLEX              := 0,
    TCVN_FT_HERSHEY_PLAIN                := 1,
    TCVN_FT_HERSHEY_DUPLEX               := 2,
    TCVN_FT_HERSHEY_COMPLEX              := 3,
    TCVN_FT_HERSHEY_TRIPLEX              := 4,
    TCVN_FT_HERSHEY_COMPLEX_SMALL        := 5,
    TCVN_FT_HERSHEY_SCRIPT_SIMPLEX       := 6,
    TCVN_FT_HERSHEY_SCRIPT_COMPLEX       := 7,
    TCVN_FT_HERSHEY_PLAIN_ITALIC         := 17,
    TCVN_FT_HERSHEY_COMPLEX_ITALIC       := 19,
    TCVN_FT_HERSHEY_TRIPLEX_ITALIC       := 20,
    TCVN_FT_HERSHEY_COMPLEX_SMALL_ITALIC := 21
)DINT;
END_TYPE
```

值

名称	描述
TCVN_FT_HERSHEY_SIMPLEX	正常大小的无衬线字体。
TCVN_FT_HERSHEY_PLAIN	小号无衬线字体。

名称	描述
TCVN_FT_HERSHEY_DUPLEX	更复杂的正常大小无衬线字体。
TCVN_FT_HERSHEY_COMPLEX	正常大小的衬线字体。
TCVN_FT_HERSHEY_TRIPLEX	更复杂的正常大小衬线字体。
TCVN_FT_HERSHEY_COMPLEX_SMALL	小号衬线字体。
TCVN_FT_HERSHEY_SCRIPT_SIMPLEX	手写字体。
TCVN_FT_HERSHEY_SCRIPT_COMPLEX	更复杂的手写字体。
TCVN_FT_HERSHEY_PLAIN_ITALIC	小号无衬线字体（斜体）。
TCVN_FT_HERSHEY_COMPLEX_ITALIC	正常大小的衬线字体（斜体）。
TCVN_FT_HERSHEY_TRIPLEX_ITALIC	更复杂的正常大小衬线字体（斜体）。
TCVN_FT_HERSHEY_COMPLEX_SMALL_ITALIC	小号衬线字体（斜体）。

更多信息

以下字体可用：



相关函数

- F_VN_PutLabelExp [▶ 1054]
- F_VN_PutText(Exp) [▶ 1055]

6.1.2.4.56 ETcVnHoughMethod

提供霍夫 Hough 方法。

语法

定义：

```
TYPE ETcVnHoughMethod :
(
    TCVN_HM_STANDARD      := 0,
    TCVN_HM_PROBABILISTIC := 1,
    TCVN_HM_MULTI_SCALE   := 2,
    TCVN_HM_GRADIENT      := 3,
```

```
    TCVN_HM_GRADIENT_ALT  := 4
)DINT;
END_TYPE
```

值

名称	描述
TCVN_HM_STANDARD	经典或标准霍夫（Hough）变换。每条直线由两个浮点数（f1，f2）表示，其中 f1 是原点（0，0）与直线之间的距离，f2 是 x 轴与直线法线之间的夹角。
TCVN_HM_PROBABILISTIC	概率霍夫（Hough）变换。对于包含长线段的图片更有效。返回线段，而不是整条直线，每个线段以起点和终点表示。
TCVN_HM_MULTI_SCALE	经典霍夫（Hough）变换的多尺度变体。
TCVN_HM_GRADIENT	设计用于找到圆形的方法。参见论文：HK Yuen、John Princen、John Illingworth 和 Josef Kittler。Comparative study of hough transform methods for circle finding。Image and Vision Computing，8（1）：71-77，1990 年
TCVN_HM_GRADIENT_ALT	梯度算法的变化。

相关函数

- F_VN_HoughCirclesExp [▶ 1195]

6.1.2.4.57 ETcVnInitializableFunction

提供可初始化的函数。

语法

定义：

```
TYPE ETcVnInitializableFunction :
(
    TCVN_IF_OCR           := 0,
    TCVN_IF_CODEDETECTION := 1
)DINT;
END_TYPE
```

值

名称	描述
TCVN_IF_OCR	F_VN_OCR
TCVN_IF_CODEDETECTION	F_VN_DetectCodesNeuralNetwork

6.1.2.4.58 ETcVnInterpolationType

提供插值类型。

语法

定义：

```
TYPE ETcVnInterpolationType :
(
    TCVN_IT_NEAREST_NEIGHBOR       := 0,
    TCVN_IT_BILINEAR               := 1,
    TCVN_IT_BICUBIC                := 2,
    TCVN_IT_AREA_BASED             := 3,
    TCVN_IT_LANCZOS4               := 4,
    TCVN_IT_BILINEAR_EXACT         := 5,
    TCVN_IT_NEAREST_NEIGHBOR_EXACT := 6
)DINT;
END_TYPE
```

值

名称	描述
TCVN_IT_NEAREST_NEIGHBOR	最近邻插值
TCVN_IT_BILINEAR	双线性插值
TCVN_IT_BICUBIC	双三次插值
TCVN_IT_AREA_BASED	基于面积的插值
TCVN_IT_LANCZOS4	Lanczos4 插值
TCVN_IT_BILINEAR_EXACT	位精确双线性插值
TCVN_IT_NEAREST_NEIGHBOR_EXACT	位精确最近邻插值

相关函数

- F_VN_MatchTemplateAndEvaluateExp [▶ 1204]
- F_VN_RemapImageToLogPolarSpaceExp [▶ 1148]
- F_VN_RemapImageToPolarSpaceExp [▶ 1151]
- F_VN_ResizeImage [▶ 1153]
- F_VN_WarpAffineExp [▶ 1163]
- F_VN_WarpPerspectiveExp [▶ 1168]

6.1.2.4.59 ETcVnKeypointDetectionTypeAGAST

为 AGAST 算法提供不同的邻域类型（详见论文：E. Mair 等：Adaptive and Generic Corner Detection Based on the Accelerated Segment Test，2010 年）。

语法

定义：

```
TYPE ETcVnKeypointDetectionTypeAGAST :
(
    TCVN_KDT_AGAST_5_8   := 0,
    TCVN_KDT_AGAST_7_12d := 1,
    TCVN_KDT_AGAST_7_12s := 2,
    TCVN_KDT_AGAST_9_16  := 3
)DINT;
END_TYPE
```

值

名称	描述
TCVN_KDT_AGAST_5_8	8 选 5
TCVN_KDT_AGAST_7_12d	12 选 7（菱形）
TCVN_KDT_AGAST_7_12s	12 选 7，正方形
TCVN_KDT_AGAST_9_16	16 选 9

更多信息

枚举`ETcVnKeypointDetectionTypeAGAST`用于结构TcVnParamsAGAST [▶ 228]。

6.1.2.4.60 ETcVnKeypointDetectionTypeFAST

为 FAST 算法提供不同的邻域类型（详见论文：E. Rosten：Machine Learning for High-speed Corner Detection，2006 年）。

语法

定义：

```
TYPE ETcVnKeypointDetectionTypeFAST :
(
    TCVN_KDT_FAST_5_8  := 0,
    TCVN_KDT_FAST_7_12 := 1,
    TCVN_KDT_FAST_9_16 := 2
)DINT;
END_TYPE
```

值

名称	描述
TCVN_KDT_FAST_5_8	8 选 5
TCVN_KDT_FAST_7_12	12 选 7
TCVN_KDT_FAST_9_16	16 选 9

更多信息

枚举`ETcVnKeypointDetectionTypeFAST`用于结构TcVnParamsFAST [▶ 231]。

6.1.2.4.61 ETcVnKnn

提供 kNN 模型类型。

语法

定义：

```
TYPE ETcVnKnn :
(
    TCVN_KNN_CLASSIFIER       := 1,
    TCVN_KNN_NOVELTY_DETECTOR := 2,
    TCVN_KNN_REGRESSOR        := 4
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_KNN_CLASSIFIER	分类器
TCVN_KNN_NOVELTY_DETECTOR	异常检测器
TCVN_KNN_REGRESSOR	回归因子

6.1.2.4.62 ETcVnLayerNameType

Offer different types of neural network model layer names

Syntax

Definition:

```
TYPE ETcVnLayerNameType :
(
    TCVN_LNT_ALL     := 0,
    TCVN_LNT_OUTPUTS := 1,
    TCVN_LNT_INPUTS  := 2
)DINT;
END_TYPE
```

Values

Name	Description
TCVN_LNT_ALL	Return the name of all layers
TCVN_LNT_OUTPUTS	Return the names of unconnected output layers
TCVN_LNT_INPUTS	Return the names of model inputs

6.1.2.4.63 ETcVnLayerNameType

提供不同类型的神经网络模型层名称

语法

定义：

```
TYPE ETcVnLayerNameType :
(
    TCVN_LNT_ALL     := 0,
    TCVN_LNT_OUTPUTS := 1,
    TCVN_LNT_INPUTS  := 2
)DINT;
END_TYPE
```

值

名称	描述
TCVN_LNT_ALL	返回所有层的名称
TCVN_LNT_OUTPUTS	返回未连接的输出层的名称
TCVN_LNT_INPUTS	返回模型输入的名称

6.1.2.4.64 ETcVnLineType

提供线条类型。

语法

定义：

```
TYPE ETcVnLineType :
(
    TCVN_LT_4_CONNECTED := 4,
    TCVN_LT_8_CONNECTED := 8,
    TCVN_LT_ANTIALIASED := 16
)DINT;
END_TYPE
```

值

名称	描述
TCVN_LT_4_CONNECTED	4 连通（像素在水平和垂直方向连通）。
TCVN_LT_8_CONNECTED	8 连通（像素在水平、垂直和对角线方向连通）。
TCVN_LT_ANTIALIASED	抗锯齿（使用高斯滤波绘制，仅适用于 8 位图像）。

相关函数

- F_VN_DrawCircleExp [▶ 1005]
- F_VN_DrawCirclesExp [▶ 1007]
- F_VN_DrawCircularArcExp [▶ 1009]
- F_VN_DrawContoursExp [▶ 1014]
- F_VN_DrawEllipseExp [▶ 1016]
- F_VN_DrawLineExp [▶ 1022]
- F_VN_DrawLineExp_TcVnVector4_DINT [▶ 1024]
- F_VN_DrawLineExp_TcVnVector4_LREAL [▶ 1025]
- F_VN_DrawLinesExp [▶ 1027]
- F_VN_DrawPointExp [▶ 1034]
- F_VN_DrawPointsExp [▶ 1036]
- F_VN_DrawRotatedRectangleExp [▶ 1043]

- F_VN_PutLabelExp [▶ 1054]
- F_VN_PutTextExp [▶ 1056]

6.1.2.4.65 ETcVnMatchTemplateWalshOptions

提供标志，用于自定义基于Walsh变换的模板匹配。

语法

定义：

```
TYPE ETcVnMatchTemplateWalshOptions :
(
    TCVN_MTWO_NONE    := 0,
    TCVN_MTWO_NORMED  := 1,
    TCVN_MTWO_SKIP_DC := 2
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_MTWO_NONE	不使用任何特殊选项。
TCVN_MTWO_NORMED	将输出图像的元素归一化到 [0，1.0] 区间内。
TCVN_MTWO_SKIP_DC	在相似度计算期间跳过 DC 部分。DC 或“直流”分量是Walshd第一个系数（全为+1的方波序列）。与其他分量相比，将其用于投影会捕获总像素强度，而非空间差异。因此，跳过它可提高对光源变化的适应性。

6.1.2.4.66 ETcVnMorphologicalOperator

提供形态学算子。

语法

定义：

```
TYPE ETcVnMorphologicalOperator :
(
    TCVN_MO_EROSION                          := 0x0,
    TCVN_MO_DILATION                         := 0x1,
    TCVN_MO_OPENING                          := 0x2,
    TCVN_MO_CLOSING                          := 0x3,
    TCVN_MO_GRADIENT                         := 0x4,
    TCVN_MO_WHITE_TOPHAT                     := 0x5,
    TCVN_MO_BLACK_TOPHAT                     := 0x6,
    TCVN_MO_OPENING_BY_RECONSTRUCTION        := 0x40000002,
    TCVN_MO_CLOSING_BY_RECONSTRUCTION        := 0x40000003,
    TCVN_MO_WHITE_TOPHAT_BY_RECONSTRUCTION := 0x40000005,
    TCVN_MO_BLACK_TOPHAT_BY_RECONSTRUCTION := 0x40000006
)DINT;
END_TYPE
```

值

名称	描述
TCVN_MO_EROSION	收缩对象（前景（即白色）像素区域），完全删除小于结构元素的区域。
TCVN_MO_DILATION	扩张对象，从而封闭对象内部的小孔。
TCVN_MO_OPENING	先腐蚀，再膨胀。小于结构元素的对象被消除，而外部形状基本保持不变。
TCVN_MO_CLOSING	先膨胀，再腐蚀。适合结构元素的对象内部的孔完全被封闭，而外部形状则基本保持不变。
TCVN_MO_GRADIENT	图像膨胀和图像腐蚀之间的区别。
TCVN_MO_WHITE_TOPHAT	输入图像与其开运算结果图像之差。

名称	描述
TCVN_MO_BLACK_TOPHAT	输入图像与其闭运算结果图像之差。
TCVN_MO_OPENING_BY_RECONSTRUCTION	开运算，随后对开运算未消除的对象进行重建。
TCVN_MO_CLOSING_BY_RECONSTRUCTION	闭运算，随后对闭运算未消除的对象进行重建。
TCVN_MO_WHITE_TOPHAT_BY_RECONSTRUCTION	白帽，随后对白帽未消除的对象进行重建。
TCVN_MO_BLACK_TOPHAT_BY_RECONSTRUCTION	黑帽，随后对黑帽未消除的对象进行重建。

相关函数

- F_VN_MorphologicalOperator [▶ 1303]

6.1.2.4.67 ETcVnNbc

提供普通贝叶斯（Bayes）分类器类型。

语法

定义：

```
TYPE ETcVnNbc :
(
    TCVN_NBC_CLASSIFIER       := 1,
    TCVN_NBC_NOVELTY_DETECTOR := 2
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_NBC_CLASSIFIER	分类器
TCVN_NBC_NOVELTY_DETECTOR	异常检测器

6.1.2.4.68 ETcVnNormalizationType

提供归一化类型。

语法

定义：

```
TYPE ETcVnNormalizationType :
(
    TCVN_NT_INF               := 1,
    TCVN_NT_L1                := 2,
    TCVN_NT_L2                := 4,
    TCVN_NT_L2SQR             := 5,
    TCVN_NT_HAMMING           := 6,
    TCVN_NT_HAMMING2          := 7,
    TCVN_NT_RELATIVE_INF      := 9,
    TCVN_NT_RELATIVE_L1       := 10,
    TCVN_NT_RELATIVE_L2       := 12,
    TCVN_NT_RELATIVE_L2SQR    := 13,
    TCVN_NT_RELATIVE_HAMMING  := 14,
    TCVN_NT_RELATIVE_HAMMING2 := 15,
    TCVN_NT_MINMAX            := 32
)DINT;
END_TYPE
```

值

名称	描述
TCVN_NT_INF	无穷大范数
TCVN_NT_L1	L1 范数

名称	描述
TCVN_NT_L2	L2（欧几里得）范数
TCVN_NT_L2SQR	平方 L2 范数
TCVN_NT_HAMMING	汉明（Hamming）距离（按位表示）
TCVN_NT_HAMMING2	汉明（Hamming）距离（每 2 个比特位组合成 1 个位）
TCVN_NT_RELATIVE_INF	无穷大范数（比较 2 个数组时，用其差值的范数除以第二个数组的范数）
TCVN_NT_RELATIVE_L1	L1 范数（比较 2 个数组时，用其差值的范数除以第二个数组的范数）
TCVN_NT_RELATIVE_L2	L2（欧几里得）范数（比较 2 个数组时，用其差值的范数除以第二个数组的范数）
TCVN_NT_RELATIVE_L2SQR	平方 L2 范数（比较 2 个数组时，用其差值的范数除以第二个数组的范数）
TCVN_NT_RELATIVE_HAMMING	汉明距离（按位表示；比较 2 个数组时，用其差值的范数除以第二个数组的范数）
TCVN_NT_RELATIVE_HAMMING2	汉明距离（每 2 个比特位组合成 1 个位；比较 2 个数组时，用其差值的范数除以第二个数组的范数）
TCVN_NT_MINMAX	将数值归一化为由最小值和最大值给出的范围。

相关函数

- F_VN_FindReferenceKeyPointsInImageORBExp [▶ 1337]
- F_VN_MatchDescriptorsBFExp [▶ 1358]
- F_VN_MatchDescriptorsKnnBFExp [▶ 1362]
- F_VN_NormalizeImageExp [▶ 1258]

6.1.2.4.69 ETcVnOcrModelType

提供不同的 OCR 模型类型

语法

定义：

```
TYPE ETcVnOcrModelType :
(
    TCVN_OMT_NUMBERS              := 580542139465730,
    TCVN_OMT_NUMBERS_SC           := 580542139465732,
    TCVN_OMT_UCLETTERS            := 580542139465736,
    TCVN_OMT_NUMBERS_SC_UCLETTERS := 580542139465744
)ULINT;
END_TYPE
```

值

名称	描述
TCVN_OMT_NUMBERS	数字分类
TCVN_OMT_NUMBERS_SC	数字和特殊字符的分类
TCVN_OMT_UCLETTERS	大写字母分类
TCVN_OMT_NUMBERS_SC_UCLETTERS	对数字、特殊字符和大写字母分类

6.1.2.4.70 ETcVnOcrOptions

提供不同选项，说明是否必须执行某些操作才能实现预期结果。

语法

定义：

```
TYPE ETcVnOcrOptions :
(
    TCVN_OO_NONE       := 0,
    TCVN_OO_WITHBLANKS := 1
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_OO_NONE	无选项
TCVN_OO_WITHBLANKS	表示如果发现空白，则应将其包含在结果中。默认情况下，结果中省略所有空白。

6.1.2.4.71 ETcVnOrientationMethod

提供一组点的方向的计算方法。

语法

定义：

```
TYPE ETcVnOrientationMethod :
(
    TCVN_OM_PCA           := 0,
    TCVN_OM_FITELLIPSE    := 1,
    TCVN_OM_MOMENTS       := 2,
    TCVN_OM_ENCLOSINGRECT := 3
)DINT;
END_TYPE
```

值

名称	描述
TCVN_OM_PCA	将 PCA 应用于各点，并计算作为中心点的均值点和主轴的旋转角度
TCVN_OM_FITELLIPSE	在各点周围拟合一个椭圆，并计算中心点和主轴的旋转角度
TCVN_OM_MOMENTS	计算各点的力矩，估算质心和旋转角度
TCVN_OM_ENCLOSINGRECT	计算包括各点的最小面积矩形，并计算该矩形的中心点和旋转角度

6.1.2.4.72 ETcVnPaddingMode

提供填充模式。

语法

定义：

```
TYPE ETcVnPaddingMode :
(
    TCVN_PM_NONE        := 0,
    TCVN_PM_CROP_CENTER := 1,
    TCVN_PM_LETTERBOX   := 2
)DINT;
END_TYPE
```

值

名称	描述
TCVN_PM_NONE	调整图像大小至所需尺寸，无需额外处理

名称	描述
TCVN_PM_CROP_CENTER	调整图像大小并裁切
TCVN_PM_LETTERBOX	调整图像大小至所需尺寸，同时保留原始图像的长宽比

6.1.2.4.73 ETcVnPixelConnectivity

提供像素连通。

语法

定义：

```
TYPE ETcVnPixelConnectivity :
(
    TCVN_PC_4 := 4,
    TCVN_PC_8 := 8
)DINT;
END_TYPE
```

值

名称	描述
TCVN_PC_4	4 连通
TCVN_PC_8	8 连通

6.1.2.4.74 ETcVnPixelEncoding

提供像素编码。

语法

定义：

```
TYPE ETcVnPixelEncoding :
(
    TCVN_PE_NONE               := 0,
    TCVN_PE_BAYER_GR           := 1,
    TCVN_PE_BAYER_RG           := 2,
    TCVN_PE_BAYER_GB           := 3,
    TCVN_PE_BAYER_BG           := 4,
    TCVN_PE_YUV_411_UYYVYY     := 5,
    TCVN_PE_YUV_422_UYVY       := 6,
    TCVN_PE_YUV_422_YUYV       := 7,
    TCVN_PE_YCBCR_411_CBYYCRYY := 8,
    TCVN_PE_YCBCR_422_CBYCRY   := 9,
    TCVN_PE_YCBCR_422_YCBYCR   := 10
)BYTE;
END_TYPE
```

值

名称	描述
TCVN_PE_NONE	无编码，即每个像素值都独立于其它像素。
TCVN_PE_BAYER_GR	像素被编码为 BayerGR 图案。
TCVN_PE_BAYER_RG	像素被编码为 BayerRG 图案。
TCVN_PE_BAYER_GB	像素被编码为 BayerGB 图案。
TCVN_PE_BAYER_BG	像素被编码为 BayerBG 图案。
TCVN_PE_YUV_411_UYYVYY	像素被编码为 YUV411（UYYVYY）。
TCVN_PE_YUV_422_UYVY	像素被编码为 YUV422（UYVY）。
TCVN_PE_YUV_422_YUYV	像素被编码为 YUV422（YUYV）。
TCVN_PE_YCBCR_411_CBYYCRYY	像素被编码为 YCbCr411（CbYYCrYY）。
TCVN_PE_YCBCR_422_CBYCRY	像素被编码为 YCbCr422（CbYCrY）。

名称	描述
TCVN_PE_YCBCR_422_YCBYCR	像素被编码为 YCbCr422（YCbYCr）。

更多信息

枚举ETcVnPixelEncoding用于结构TcVnPixelFormat [▶ 235]。关于像素编码的信息，可以通过查询图像的像素格式信息获得。常见使用案例是查询 Bayer 模式，以了解一张图片是否还需要转换为 RGB 格式等。

6.1.2.4.75 ETcVnPixelPackMode

提供像素包装模式。

语法

定义：

```
TYPE ETcVnPixelPackMode :
(
    TCVN_PPM_NONE           := 0,
    TCVN_PPM_MONO1P         := 1,
    TCVN_PPM_MONO2P         := 2,
    TCVN_PPM_MONO4P         := 3,
    TCVN_PPM_MONO10PACKED   := 4,
    TCVN_PPM_MONO12PACKED   := 5,
    TCVN_PPM_RGB10V1PACKED := 6,
    TCVN_PPM_RGB10P32       := 7,
    TCVN_PPM_RGB12V1PACKED := 8,
    TCVN_PPM_RGB565P        := 9,
    TCVN_PPM_BGR565P        := 10,
    TCVN_PPM_MONO10P        := 11,
    TCVN_PPM_MONO12P        := 12,
    TCVN_PPM_MONO14P        := 13
)BYTE;
END_TYPE
```

值

名称	描述
TCVN_PPM_NONE	不包装
TCVN_PPM_MONO1P	Mono1p
TCVN_PPM_MONO2P	Mono2p
TCVN_PPM_MONO4P	Mono4p
TCVN_PPM_MONO10PACKED	Mono10Packed 或 BayerXX10Packed
TCVN_PPM_MONO12PACKED	Mono12Packed 或 BayerXX12Packed
TCVN_PPM_RGB10V1PACKED	RGB10V1Packed
TCVN_PPM_RGB10P32	RGB10V2Packed（RGB10p32）
TCVN_PPM_RGB12V1PACKED	RGB12V1Packed
TCVN_PPM_RGB565P	RGB565p
TCVN_PPM_BGR565P	BGR565p
TCVN_PPM_MONO10P	Mono10p 或 BayerXX10p
TCVN_PPM_MONO12P	Mono12p 或 BayerXX12p
TCVN_PPM_MONO14P	Mono14p 或 BayerXX14p

更多信息

枚举ETcVnPixelPackMode用于结构TcVnPixelFormat [▶ 235]。

6.1.2.4.76 ETcVnPrototypeClusterer

提供原型聚类器模型类型。

语法

定义：

```
TYPE ETcVnPrototypeClusterer :
(
    TCVN_PC_NOVELTY_DETECTOR := 2,
    TCVN_PC_CLUSTERER        := 8
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_PC_NOVELTY_DETECTOR	异常检测器
TCVN_PC_CLUSTERER	聚类器

6.1.2.4.77 ETcVnRectangleIntersection

提供矩形交集类型。

语法

定义：

```
TYPE ETcVnRectangleIntersection :
(
    TCVN_RI_NONE    := 0,
    TCVN_RI_PARTIAL := 1,
    TCVN_RI_FULL    := 2
)DINT;
END_TYPE
```

值

名称	描述
TCVN_RI_NONE	无交集
TCVN_RI_PARTIAL	部分相交
TCVN_RI_FULL	完全相交

6.1.2.4.78 ETcVnReduceType

提供减少类型。

语法

定义：

```
TYPE ETcVnReduceType :
(
    TCVN_RT_REDUCE_SUM := 0,
    TCVN_RT_REDUCE_AVG := 1,
    TCVN_RT_REDUCE_MAX := 2,
    TCVN_RT_REDUCE_MIN := 3
)DINT;
END_TYPE
```

值

名称	描述
TCVN_RT_REDUCE_SUM	所有行/列的总和（逐列/行）
TCVN_RT_REDUCE_AVG	所有行/列的平均值（逐列/行）
TCVN_RT_REDUCE_MAX	所有行/列的最大值（逐列/行）
TCVN_RT_REDUCE_MIN	所有行/列的最小值（逐列/行）

6.1.2.4.79 ETcVnRotationAngle

提供旋转角度。

语法

定义：

```
TYPE ETcVnRotationAngle :
(
    TCVN_RA_90_DEG  := 0,
    TCVN_RA_180_DEG := 1,
    TCVN_RA_270_DEG := 2
)DINT;
END_TYPE
```

值

名称	描述
TCVN_RA_90_DEG	90 度
TCVN_RA_180_DEG	180 度
TCVN_RA_270_DEG	270 度

相关函数

- F_VN_RotateImage [▶ 1156]

6.1.2.4.80 ETcVnRTrees

提供 RTrees 模型类型。

语法

定义：

```
TYPE ETcVnRTrees :
(
    TCVN_RT_CLASSIFIER := 1,
    TCVN_RT_REGRESSOR  := 4
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_RT_CLASSIFIER	分类器
TCVN_RT_REGRESSOR	回归因子

6.1.2.4.81 ETcVnSignedNormalization

提供对带符号数据类型的图像进行归一化处理的选项。

语法

定义：

```
TYPE ETcVnSignedNormalization :
(
    TCVN_SN_FIX_ZERO   := 0,
    TCVN_SN_FULL_SCALE := 1
)INT;
END_TYPE
```

值

名称	描述
TCVN_SN_FIX_ZERO	0 值是固定的，也就是说，归一化图像只能达到最小值或最大值。
TCVN_SN_FULL_SCALE	图像被归一化为全值范围，也就是说，先验值 0 在归一化后可能是 != 0。

相关函数

- F_VN_NormalizeImageForDisplay [▶ 1259]

6.1.2.4.82 ETcVnSolvePnPMethod

提供 SolvePnP 方法。

语法

定义：

```
TYPE ETcVnSolvePnPMethod :
(
    TCVN_SPM_ITERATIVE   := 0,
    TCVN_SPM_EPNP        := 1,
    TCVN_SPM_P3P         := 2,
    TCVN_SPM_AP3P        := 5,
    TCVN_SPM_IPPE        := 6,
    TCVN_SPM_IPPE_SQUARE := 7,
    TCVN_SPM_SQPNP       := 8
)DINT;
END_TYPE
```

值

名称	描述
TCVN_SPM_ITERATIVE	Levenberg-Marquardt 优化。需要至少 4 个平面参考点或 6 个非平面参考点。
TCVN_SPM_EPNP	有效 PnP 相机位姿估计。需要至少 4 个参考点。
TCVN_SPM_P3P	P3P 问题的完整解分类。需要 4 个参考点才能找到唯一解。
TCVN_SPM_AP3P	P3P 问题的有效代数解。需要 4 个参考点才能找到唯一解。
TCVN_SPM_IPPE	基于无限小平面的位姿估计。需要至少 4 个共面参考点。
TCVN_SPM_IPPE_SQUARE	方形 IPPE 算法。需要 4 个参考点，顺序如下（hsl：方形半边长）：（-hsl，hsl，0），（hsl，hsl，0），（hsl，-hsl，0），（-hsl，-hsl，0）
TCVN_SPM_SQPNP	对 PnP 问题持续快速做出全局最优解。需要至少 4 个参考点。

6.1.2.4.83 ETcVnSta

提供简化 TopoART 模型类型。

语法

定义：

```
TYPE ETcVnSta :
(
    TCVN_STA_CLASSIFIER       := 1,
    TCVN_STA_NOVELTY_DETECTOR := 2,
    TCVN_STA_REGRESSOR        := 4,
```

```
    TCVN_STA_CLUSTERER        := 8
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_STA_CLASSIFIER	分类器
TCVN_STA_NOVELTY_DETECTOR	异常检测器
TCVN_STA_REGRESSOR	回归因子
TCVN_STA_CLUSTERER	聚类器

6.1.2.4.84 ETcVnStructuringElementShape

提供结构元素的形状。

语法

定义：

```
TYPE ETcVnStructuringElementShape :
(
    TCVN_SES_RECTANGLE := 0,
    TCVN_SES_CROSS     := 1,
    TCVN_SES_ELLIPSE   := 2
)DINT;
END_TYPE
```

值

名称	描述
TCVN_SES_RECTANGLE	矩形
TCVN_SES_CROSS	十字形
TCVN_SES_ELLIPSE	椭圆

相关函数

- F_VN_CreateStructuringElement [▶ 1286]

6.1.2.4.85 ETcVnSvm

提供 SVM 模型类型。

语法

定义：

```
TYPE ETcVnSvm :
(
    TCVN_SVM_C_CLASSIFIER     := 1,
    TCVN_SVM_NU_CLASSIFIER    := 65537,
    TCVN_SVM_NOVELTY_DETECTOR := 2,
    TCVN_SVM_EPS_REGRESSOR    := 4,
    TCVN_SVM_NU_REGRESSOR     := 65540
)UDINT;
END_TYPE
```

值

名称	描述
TCVN_SVM_C_CLASSIFIER	分类器
TCVN_SVM_NU_CLASSIFIER	分类器
TCVN_SVM_NOVELTY_DETECTOR	异常检测器
TCVN_SVM_EPS_REGRESSOR	回归因子

名称	描述
TCVN_SVM_NU_REGRESSOR	回归因子

6.1.2.4.86 ETcVnSvmKernelType

提供 SVM 核类型。

语法

定义：

```
TYPE ETcVnSvmKernelType :
(
    TCVN_SKT_LINEAR  := 0,
    TCVN_SKT_POLY    := 1,
    TCVN_SKT_RBF     := 2,
    TCVN_SKT_SIGMOID := 3,
    TCVN_SKT_CHI2    := 4,
    TCVN_SKT_INTER   := 5
)DINT;
END_TYPE
```

值

名称	描述
TCVN_SKT_LINEAR	线性核（快速）
TCVN_SKT_POLY	多项式核
TCVN_SKT_RBF	径向基函数（RBF）核（良好默认值）
TCVN_SKT_SIGMOID	S 形（Sigmoid）核
TCVN_SKT_CHI2	卡方核
TCVN_SKT_INTER	直方图交叉核

6.1.2.4.87 ETcVnSvmSgdClassifierMarginType

为 SVMSGD 分类器提供不同的边际类型。

语法

定义：

```
TYPE ETcVnSvmSgdClassifierMarginType :
(
    TCVN_SSCMT_SOFT_MARGIN := 0,
    TCVN_SSCMT_HARD_MARGIN := 1
)DINT;
END_TYPE
```

值

名称	描述
TCVN_SSCMT_SOFT_MARGIN	允许异常值的软边际。（适用于无法线性分离的类别）
TCVN_SSCMT_HARD_MARGIN	硬边际。（最适合可线性分离的类别）

6.1.2.4.88 ETcVnSvmSgdClassifierType

提供不同类型的 SVMSGD 分类器。

语法

定义：

```
TYPE ETcVnSvmSgdClassifierType :
(
    TCVN_SSCT_SGD  := 0,
    TCVN_SSCT_ASGD := 1
)DINT;
END_TYPE
```

值

名称	描述
TCVN_SSCT_SGD	随机梯度下降
TCVN_SSCT_ASGD	平均随机梯度下降

6.1.2.4.89 ETcVnTemplateMatchMethod

提供模板匹配方法。

语法

定义：

```
TYPE ETcVnTemplateMatchMethod :
(
    TCVN_TMM_SQDIFF        := 0,
    TCVN_TMM_SQDIFF_NORMED := 1,
    TCVN_TMM_CCORR         := 2,
    TCVN_TMM_CCORR_NORMED  := 3,
    TCVN_TMM_CCOEFF        := 4,
    TCVN_TMM_CCOEFF_NORMED := 5
)DINT;
END_TYPE
```

值

名称	描述
TCVN_TMM_SQDIFF	平方差（支持模板掩膜）
TCVN_TMM_SQDIFF_NORMED	归一化平方差
TCVN_TMM_CCORR	互相关
TCVN_TMM_CCORR_NORMED	归一化互相关（支持模板掩膜）
TCVN_TMM_CCOEFF	相关系数
TCVN_TMM_CCOEFF_NORMED	归一化相关系数

相关函数

- F_VN_MatchTemplateAndEvaluateExp [▶ 1204]
- F_VN_MatchTemplateExp [▶ 1202]

6.1.2.4.90 ETcVnThresholdType

提供阈值类型。

语法

定义：

```
TYPE ETcVnThresholdType :
(
    TCVN_TT_BINARY              := 0,
    TCVN_TT_BINARY_INV          := 1,
    TCVN_TT_TRUNC               := 2,
    TCVN_TT_TOZERO              := 3,
    TCVN_TT_TOZERO_INV          := 4,
    TCVN_TT_OTSU_BINARY         := 8,
    TCVN_TT_OTSU_BINARY_INV     := 9,
    TCVN_TT_OTSU_TRUNC          := 10,
    TCVN_TT_OTSU_TOZERO         := 11,
    TCVN_TT_OTSU_TOZERO_INV     := 12,
    TCVN_TT_TRIANGLE_BINARY     := 16,
```

```
    TCVN_TT_TRIANGLE_BINARY_INV := 17,
    TCVN_TT_TRIANGLE_TRUNC      := 18,
    TCVN_TT_TRIANGLE_TOZERO     := 19,
    TCVN_TT_TRIANGLE_TOZERO_INV := 20
)DINT;
END_TYPE
```

值

名称	描述
TCVN_TT_BINARY	二值化阈值
TCVN_TT_BINARY_INV	反二值化阈值
TCVN_TT_TRUNC	截断阈值（像素 > 阈值时设置为阈值，其它情况下值不变）
TCVN_TT_TOZERO	阈值取零（像素 < 阈值时设置为零，其它情况下值不变）
TCVN_TT_TOZERO_INV	阈值反取零（像素 > 阈值设置为零，其它情况下值不变）
TCVN_TT_OTSU_BINARY	二值化阈值，根据大津（Otsu）算法选择阈值
TCVN_TT_OTSU_BINARY_INV	反二值化阈值，根据大津（Otsu）算法选择阈值
TCVN_TT_OTSU_TRUNC	截断阈值，根据大津（Otsu）算法选择阈值
TCVN_TT_OTSU_TOZERO	阈值取零，根据大津（Otsu）算法选择阈值
TCVN_TT_OTSU_TOZERO_INV	阈值反取零，根据大津（Otsu）算法选择阈值
TCVN_TT_TRIANGLE_BINARY	二值化阈值，根据三角形算法选择阈值
TCVN_TT_TRIANGLE_BINARY_INV	反二值化阈值，根据三角形算法选择阈值
TCVN_TT_TRIANGLE_TRUNC	截断阈值，根据三角形算法选择阈值
TCVN_TT_TRIANGLE_TOZERO	阈值取零，根据三角算法选择阈值
TCVN_TT_TRIANGLE_TOZERO_INV	阈值反取零，根据三角形算法选择阈值

更多信息

枚举`ETcVnThresholdType`提供了计算阈值所实施的方法的选择。它也被用于结构 TcVnParamsBlobDetection [▶ 229]。

相关函数

- F_VN_AdaptiveThresholdExp [▶ 1317]
- F_VN_Threshold [▶ 1322]

6.1.2.4.91 ETcVnTimestamp

提供图像采集时间戳。

语法

定义：

```
TYPE ETcVnTimestamp :
(
    TCVN_TS_IMAGE_COMPLETED    := 0x1,
    TCVN_TS_RECEPTION_STARTED  := 0x10,
    TCVN_TS_RECEPTION_FINISHED := 0x20,
    TCVN_TS_GETCURRENTIMAGE    := 0x100,
    TCVN_TS_CAMERA             := 0x10000,
    TCVN_TS_CUSTOM             := 0x1000000,
    TCVN_TS_LAST_CHANGE        := 0x1000000000000000
)ULINT;
END_TYPE
```

值

名称	描述
TCVN_TS_IMAGE_COMPLETED	完成图像时的时间戳，即 GigEVision 驱动程序收到所有数据包并将其组合成完整图像的时间。
TCVN_TS_RECEPTION_STARTED	图像接收开始时的时间戳，即 GigEVision 驱动程序收到 GVSP 引导数据包的时间。
TCVN_TS_RECEPTION_FINISHED	图像接收完成时的时间戳，即 GigEVision 驱动程序收到 GVSP 结束数据包的时间。
TCVN_TS_GETCURRENTIMAGE	通过调用 CameraControl FB 的一种 GetCurrentImage 方法在 PLC 中获取图像时的时间戳。
TCVN_TS_CAMERA	由相机创建并作为 GVSP 引导数据包的一部分传输的时间戳。该时间戳是相机特定的，默认情况下不与 TwinCAT 同步。
TCVN_TS_CUSTOM	自定义时间戳，可在创建可显示图像时选择设置（Copy/TransformIntoDisplayableImage 函数的 nTimestamp 参数）。
TCVN_TS_LAST_CHANGE	最近一次通过 TwinCAT Vision 函数改变图像内容的时间戳。

6.1.2.4.92 ETcVnVectorCompareMethod

提供向量比较的方法。

语法

定义：

```
TYPE ETcVnVectorCompareMethod :
(
    TCVN_VCM_EUCLIDEAN    := 0,
    TCVN_VCM_ELEMENTWISE := 1
)DINT;
END_TYPE
```

值

名称	描述
TCVN_VCM_EUCLIDEAN	欧式
TCVN_VCM_ELEMENTWISE	逐元素

相关函数

- F_VN_MaxPixelValueExp [▶ 1232]
- F_VN_MinPixelValueExp [▶ 1234]

6.1.2.4.93 ETcVnWorldCoordinateSystem

提供世界坐标系方向。

语法

定义：

```
TYPE ETcVnWorldCoordinateSystem :
(
    TCVN_WCS_X_RIGHT_Y_DOWN := 0,
    TCVN_WCS_X_DOWN_Y_RIGHT := 1,
    TCVN_WCS_X_LEFT_Y_DOWN  := 2,
    TCVN_WCS_X_UP_Y_RIGHT   := 3,
    TCVN_WCS_X_RIGHT_Y_UP   := 4,
```

```
    TCVN_WCS_X_DOWN_Y_LEFT  := 5,
    TCVN_WCS_X_LEFT_Y_UP    := 6,
    TCVN_WCS_X_UP_Y_LEFT    := 7
)DINT;
END_TYPE
```

值

名称	描述
TCVN_WCS_X_RIGHT_Y_DOWN	X 轴向右，Y 轴向下（与图像坐标系方向相同）
TCVN_WCS_X_DOWN_Y_RIGHT	X 轴向下，Y 轴向右
TCVN_WCS_X_LEFT_Y_DOWN	X 轴向左，Y 轴向下
TCVN_WCS_X_UP_Y_RIGHT	X 轴向上，Y 轴向右
TCVN_WCS_X_RIGHT_Y_UP	X 轴向右，Y 轴向上
TCVN_WCS_X_DOWN_Y_LEFT	X 轴向下，Y 轴向左
TCVN_WCS_X_LEFT_Y_UP	X 轴向左，Y 轴向上
TCVN_WCS_X_UP_Y_LEFT	X 轴向上，Y 轴向左

6.1.2.4.94 ETcWatchdogAccumulationType

提供用于计算已处理分数的看门狗累计类型。

语法

定义：

```
TYPE ETcWatchdogAccumulationType :
(
    WD_ACC_TYPE_MEAN    := 1,
    WD_ACC_TYPE_PRODUCT := 2
)UINT;
END_TYPE
```

值

名称	描述
WD_ACC_TYPE_MEAN	计算所有被监控函数中已处理的各个分数的平均值（建议用于独立函数）。
WD_ACC_TYPE_PRODUCT	计算所有被监控函数中已处理的各个分数的乘积（建议用于从属函数）。

更多信息

通过这个枚举，在设置当几个函数串联时，可以设置看门狗 [▶ 140]如何计算已执行函数的分数（nFractionProcessed）。

作为说明操作模式的一个示例，假设有三个函数受到监测。第一次是 100% 执行，第二次是 50%，第三次是 0%。

WD_ACC_TYPE_MEAN	所有函数的执行分数的平均值将作为结果返回。在例子中，结果是：**（100% + 50% + 0%）/ 3 = 50%**。
WD_ACC_TYPE_PRODUCT	所有函数的执行分数的乘积将作为结果返回。在这个例子中，结果将是：**100% * 50% * 0% = 0%**。

根据程序结构和机器的安全状态，使用情况有所不同。

相关函数

- F_VN_StartAbsWatchdogExp [▶ 995]
- F_VN_StartRelWatchdogExp [▶ 997]

6.1.2.4.95 GEV_CAMERA_STATE

描述 GigEVision 相机的连接状态。

语法

定义：

```
TYPE GEV_CAMERA_STATE :
(
    GEV_CAMERA_IDLE               := 0,
    GEV_CONTROL_CHANNEL_OPEN_MASK := 1,
    GEV_STREAM_CHANNELS_OPEN_MASK := 2,
    GEV_CONTROL_CHANNEL_OPEN      := 1,
    GEV_STREAM_CHANNELS_OPEN      := 3
)BYTE;
END_TYPE
```

值

名称	描述
GEV_CAMERA_IDLE	相机处于空闲状态（未打开控制或数据流通道）。
GEV_CONTROL_CHANNEL_OPEN_MASK	控制通道已打开。
GEV_STREAM_CHANNELS_OPEN_MASK	至少有 1 条数据流通道打开。
GEV_CONTROL_CHANNEL_OPEN	控制通道已打开，但没有数据流通道。
GEV_STREAM_CHANNELS_OPEN	至少有 1 个数据流通道和控制通道打开。

6.1.2.5 Structs

6.1.2.5.1 TcVnParamsFindRois

Parameters for FindRois.

Syntax

Definition:

```
TYPE TcVnParamsFindRois :
STRUCT
    stRoiSize             : TcVnSize2_DINT;
    fMinDensity           : LREAL;
    fDetectionScaleFactor : LREAL;
    fMaxOversizeFraction  : LREAL;
END_STRUCT
END_TYPE
```

Parameters

Name	Type	Default	Description
stRoiSize	TcVnSize2_DINT [▶ 219]	(.nWidth := 416, .nHeight := 416)	Desired ROI dimensions in original-image pixels.
fMinDensity	LREAL	0.2	Minimum feature density required to keep an ROI, given as a factor of the image's average density. Higher values keep fewer ROIs; suggested range [0.0, 2.0]. 0.0 disables the filter.

Name	Type	Default	Description
fDetectionScaleFactor	LREAL	0.25	Scale factor that controls the resolution of the ROI detection (Lower = faster density evaluation; valid range [0.1, 1.0]).
fMaxOversizeFraction	LREAL	1.0	Fraction by which ROIs may exceed stRoiSize (required only when eFindRoisMethod = TCVN_FRM_CONNECTED COMPONENTS).

6.1.2.5.2 TcVnParamsWalshBarcodeDetection

Parameters for bar code detection based on the Walsh transform.

Syntax

Definition:

```
TYPE TcVnParamsWalshBarcodeDetection :
STRUCT
    nMaxBarDistance             : UDINT;
    fMinAspectRatio             : REAL;
    fMaxAspectRatio             : REAL;
    nIntensityDifference        : UDINT;
    nMinArea                    : DINT;
    nChannelIndex               : DINT;
    nWalshFilterNum             : UDINT;
    fOutlierReductionFactor     : REAL;
    fBarConnectionFactor        : REAL;
    fRegionFilterFactor         : REAL;
    fPostProcessingResizeFactor : REAL;
    eInterpolationType          : ETcVnInterpolationType;
END_STRUCT
END_TYPE
```

Parameters

Name	Type	Default	Description
nMaxBarDistance	UDINT	8	Maximum distance between two consecutive dark or bright bars in pixels
fMinAspectRatio	REAL	1.0	Minimum aspect ratio (width / height) of valid code regions
fMaxAspectRatio	REAL	5.0	Maximum aspect ratio (width / height) of valid code regions
nIntensityDifference	UDINT	100	Approximate mean difference of dark and bright code pixels
nMinArea	DINT	-1	Expected minimum code area in pixels (-1: use an automatic guess based on nMaxBarDistance)
nChannelIndex	DINT	-1	Choose the index of a specific color channel (e.g., 0 = red, 1 = green, 2 = blue) or automatic graylevel conversion (-1).

Name	Type	Default	Description
			Using automatic graylevel conversion, 3-channel images and 4-channel images are considered as RGB and RGBA images, respectively, and converted to graylevel images before processing. For images with another number of channels the first one (index 0) is selected.
nWalshFilterNum	UDINT	5	Number of Walsh filters to be used (higher values may result in improved results but require more time to compute)
fOutlierReductionFactor	REAL	1.0	Factor to control outlier reduction (Higher values reduce the number of outlier pixels but may disturb the barcodes to be localized.)
fBarConnectionFactor	REAL	1.0	Factor to control bar linking (Higher values reduce the number of outlier pixels but may disturb the barcodes to be localized.)
fRegionFilterFactor	REAL	1.0	Factor to control the rejection of small regions and region parts (Higher values reduce the number of found code regions, adjust their shape and increase the computation time.)
fPostProcessingResizeFactor	REAL	0.5	Factor to control resizing before post-processiong (Smaller values reduce the required computation time but may deteriorate the results.)
eInterpolationType	ETcVnInterpolationType [▶ 198]	TCVN_IT_BILINEAR	Image rotation interpolation type (TCVN_IT_BILINEAR recommended for most cases)

6.1.2.5.3 TcVnSize2_DINT

Contains width and height.

Syntax

Definition:

```
TYPE TcVnSize2_DINT :
STRUCT
    nWidth  : DINT;
    nHeight : DINT;
END_STRUCT
END_TYPE
```

Parameters

Name	Type	Description
nWidth	DINT	Width
nHeight	DINT	Height

6.1.2.5.4 GVCP_ADDRESS_VALUE_PAIR

描述一对 GVCP 寄存器地址与值。

语法

定义：

```
TYPE GVCP_ADDRESS_VALUE_PAIR :
STRUCT
    nAddress : GVCP_REGISTER_ADDRESS;
    nValue   : GVCP_REGISTER_VALUE;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 152]	寄存器地址
nValue	GVCP_REGISTER_VALUE [▶ 152]	寄存器值

6.1.2.5.5 GVSP_IMAGE_INFO

显示 GVSP（千兆以太网流协议）元信息。

语法

定义：

```
TYPE GVSP_IMAGE_INFO :
STRUCT
    CameraIpAddress : UDINT;
    LocalIpAddress  : UDINT;
    CameraUdpPort   : UINT;
    LocalUdpPort    : UINT;
    GvspChannelId   : UINT;
    GevStatus       : UINT;
    BlockId         : ULINT;
    LeaderInfo      : GVSP_LEADER_PAYLOAD_IMAGE;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
CameraIpAddress	UDINT	相机 IP 地址
LocalIpAddress	UDINT	本地 IP 地址
CameraUdpPort	UINT	相机 UDP 端口
LocalUdpPort	UINT	本地 UDP 端口

名称	类型	描述
GvspChannelId	UINT	GVSP 通道 ID
GevStatus	UINT	GigE Vision 状态代码
BlockId	ULINT	块 ID（每采集一幅图像递增一次，溢出时重置为 0）
LeaderInfo	GVSP_LEADER_PAYLOAD_IMAGE [▶ 221]	包含有关时间戳、像素格式和大小的信息。

6.1.2.5.6 GVSP_LEADER_PAYLOAD_IMAGE

显示 GVSP（千兆以太网流协议）前导有效载荷图像的信息。

语法

定义：

```
TYPE GVSP_LEADER_PAYLOAD_IMAGE :
STRUCT
    Timestamp   : ULINT;
    PixelFormat : GVSP_PIXEL_FORMAT;
    SizeX       : UDINT;
    SizeY       : UDINT;
    OffsetX     : UDINT;
    OffsetY     : UDINT;
    PaddingX    : UINT;
    PaddingY    : UINT;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
Timestamp	ULINT	图像时间戳
PixelFormat	GVSP_PIXEL_FORMAT [▶ 221]	图像像素格式
SizeX	UDINT	x 方向的图像大小
SizeY	UDINT	y 方向的图像大小
OffsetX	UDINT	图像与（0，0）原点的 x 偏移量
OffsetY	UDINT	图像与（0，0）原点的 y 偏移量
PaddingX	UINT	x 方向的图像填充
PaddingY	UINT	Y 方向的图像填充

6.1.2.5.7 GVSP_PIXEL_FORMAT

显示 GVSP（千兆以太网流协议）像素格式。

语法

定义：

```
TYPE GVSP_PIXEL_FORMAT :
STRUCT
    Color              : BYTE;
    EffectivePixelSize : BYTE;
    Id                 : UINT;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
颜色	BYTE	表示像素格式是单色还是彩色。
EffectivePixelSize	BYTE	有效像素大小（单位：位）

名称	类型	描述
Id	UINT	GVSP 像素格式 ID

6.1.2.5.8 TcVnCameraCalibrationOptions

提供相机标定选项。

语法

定义：

```
TYPE TcVnCameraCalibrationOptions :
STRUCT
    bUseIntrinsicGuess : BIT;
    bFixAspectRatio    : BIT;
    bFixPrincipalPoint : BIT;
    bZeroTangentDist   : BIT;
    bFixFocalLength    : BIT;
    bFixK1             : BIT;
    bFixK2             : BIT;
    bFixK3             : BIT;
    bFixK4             : BIT;
    bFixK5             : BIT;
    bFixK6             : BIT;
    bRationalModel     : BIT;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
bUseIntrinsicGuess	BIT	相机矩阵和畸变系数已经包含了一个进一步优化的有效初始猜测。
bFixAspectRatio	BIT	fx/fy 的比例与输入相机矩阵中的比例相同。
bFixPrincipalPoint	BIT	主点固定在图像中心（如果使用 bUseIntrinsicGuess，则提供 cx、cy）。
bZeroTangentDist	BIT	切向畸变系数（p1，p2）强制为 0。
bFixFocalLength	BIT	参数 fx 和 fy 与输入相机矩阵中的参数相同。
bFixK1	BIT	径向畸变系数 k1 固定为 0（如果使用 bUseIntrinsicGuess，则为输入值）
bFixK2	BIT	径向畸变系数 k2 固定为 0（如果使用 bUseIntrinsicGuess，则为输入值）
bFixK3	BIT	径向畸变系数 k3 固定为 0（如果使用 bUseIntrinsicGuess，则为输入值）
bFixK4	BIT	径向畸变系数 k4 固定为 0（如果使用 bUseIntrinsicGuess，则为输入值）
bFixK5	BIT	径向畸变系数 k5 固定为 0（如果使用 bUseIntrinsicGuess，则为输入值）
bFixK6	BIT	径向畸变系数 k6 固定为 0（如果使用 bUseIntrinsicGuess，则为输入值）
bRationalModel	BIT	启用径向畸变系数 k4、k5 和 k6。

6.1.2.5.9 TcVnCircularArc

描述圆弧。

语法

定义：

```
TYPE TcVnCircularArc :
STRUCT
    aCenter     : TcVnPoint2_REAL;
    fRadius     : REAL;
    fStartAngle : REAL;
    fEndAngle   : REAL;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
aCenter	TcVnPoint2_REAL [▶ 152]	圆弧中心
fRadius	REAL	圆弧半径
fStartAngle	REAL	圆弧起点角度
fEndAngle	REAL	圆弧的终点角度

6.1.2.5.10 TcVnCodeGrades1D

根据 ISO/IEC 15416:2016 标准，描述 1D 条形码的编码质量等级。等级范围从 0（非常差）至 4（非常好），是几条扫描线的平均值。

语法

定义：

```
TYPE TcVnCodeGrades1D :
STRUCT
    fDecode          : REAL;
    fSymbolContrast  : REAL;
    fMinReflectance  : REAL;
    fMinEdgeContrast : REAL;
    fModulation      : REAL;
    fDefects         : REAL;
    fDecodability    : REAL;
    fOverall         : REAL;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
fDecode	REAL	解码等级
fSymbolContrast	REAL	符号对比度等级
fMinReflectance	REAL	最小反射率等级
fMinEdgeContrast	REAL	最小边缘对比度等级
fModulation	REAL	调制等级
fDefects	REAL	缺陷等级
fDecodability	REAL	可解码性等级
fOverall	REAL	整体编码等级，即所有扫描线平均达到的最低单个等级。如果不同的扫描线产生不同的解码数据，则整体等级为 0。

6.1.2.5.11 TcVnCodeGradesDM

根据 ISO/IEC 15415:2011 标准，描述数据矩阵编码的编码质量等级。等级范围从 0（非常差）至 4（非常好）。

语法

定义：

```
TYPE TcVnCodeGradesDM :
STRUCT
    nDecode                : USINT;
    nSymbolContrast        : USINT;
    nModulation            : USINT;
    nReflectanceMargin     : USINT;
    nFixedPatternDamage    : USINT;
    nAxialNonuniformity    : USINT;
    nGridNonuniformity     : USINT;
    nUnusedErrorCorrection : USINT;
    nOverall               : USINT;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
nDecode	USINT	解码等级
nSymbolContrast	USINT	符号对比度等级
nModulation	USINT	调制等级
nReflectanceMargin	USINT	反射率裕度等级
nFixedPatternDamage	USINT	固定图案损坏等级
nAxialNonuniformity	USINT	轴向不均匀性等级
nGridNonuniformity	USINT	网格不均匀性等级
nUnusedErrorCorrection	USINT	未使用的错误校正等级
nOverall	USINT	整体编码等级，即达到的最低单个等级

6.1.2.5.12 TcVnCodeGradesQR

根据 ISO/IEC 15415:2011 标准，描述二维码的编码质量等级。等级范围从 0（非常差）至 4（非常好）。

语法

定义：

```
TYPE TcVnCodeGradesQR :
STRUCT
    nDecode                : USINT;
    nSymbolContrast        : USINT;
    nModulation            : USINT;
    nReflectanceMargin     : USINT;
    nFixedPatternDamage    : USINT;
    nAxialNonuniformity    : USINT;
    nGridNonuniformity     : USINT;
    nUnusedErrorCorrection : USINT;
    nFormatInfo            : USINT;
    nVersionInfo           : USINT;
    nOverall               : USINT;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
nDecode	USINT	解码等级
nSymbolContrast	USINT	符号对比度等级
nModulation	USINT	调制等级

名称	类型	描述
nReflectanceMargin	USINT	反射率裕度等级
nFixedPatternDamage	USINT	固定图案损坏等级
nAxialNonuniformity	USINT	轴向不均匀性等级
nGridNonuniformity	USINT	网格不均匀性等级
nUnusedErrorCorrection	USINT	未使用的错误校正等级
nFormatInfo	USINT	格式信息等级
nVersionInfo	USINT	版本信息等级
nOverall	USINT	整体编码等级，即达到的最低单个等级

6.1.2.5.13 TcVnDMatch

对描述符匹配器进行描述

语法

定义：

```
TYPE TcVnDMatch :
STRUCT
    nQueryIdx : DINT;
    nTrainIdx : DINT;
    nImageIdx : DINT;
    fDistance : REAL;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
nQueryIdx	DINT	查询描述符索引
nTrainIdx	DINT	训练描述符索引
nImageIdx	DINT	训练图像索引
fDistance	REAL	描述符之间的距离（距离越小，匹配度越高）

6.1.2.5.14 TcVnImageInfo

显示图像信息。

语法

定义：

```
TYPE TcVnImageInfo :
STRUCT
    nImageSize    : ULINT;
    nWidth        : UDINT;
    nHeight       : UDINT;
    nXPadding     : UINT;
    nYPadding     : UINT;
    stPixelFormat : TcVnPixelFormat;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
nImageSize	ULINT	图像大小（像素数）
nWidth	UDINT	图像宽度
nHeight	UDINT	图像高度

名称	类型	描述
nXPadding	UINT	图像 x-填充
nYPadding	UINT	图像 y-填充
stPixelFormat	TcVnPixelFormat [▶ 235]	像素格式

6.1.2.5.15 TcVnKeyPoint

描述一个关键点。

语法

定义：

```
TYPE TcVnKeyPoint :
STRUCT
    aPoint    : TcVnPoint2_REAL;
    fDiameter : REAL;
    fAngle    : REAL;
    fResponse : REAL;
    nOctave   : DINT;
    nClassId  : DINT;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
aPoint	TcVnPoint2_REAL [▶ 152]	位置
fDiameter	REAL	直径
fAngle	REAL	角度
fResponse	REAL	响应
nOctave	DINT	尺度空间（例如：高斯金字塔数）
nClassId	DINT	类别 ID

6.1.2.5.16 TcVnMatrix

提供用户定义的矩阵，行、列和元素类型可变。

语法

定义：

```
TYPE TcVnMatrix :
STRUCT
    nRows : UDINT;
    nCols : UDINT;
    eType : ETcVnElementType;
    pData : PVOID;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
nRows	UDINT	行数
nCols	UDINT	列数
eType	ETcVnElementType [▶ 191]	元素类型
pData	PVOID	数据指针

6.1.2.5.17 TcVnMoments

提供图像矩或轮廓矩。

语法

定义：

```
TYPE TcVnMoments :
STRUCT
    fM00  : LREAL;
    fM10  : LREAL;
    fM01  : LREAL;
    fM20  : LREAL;
    fM11  : LREAL;
    fM02  : LREAL;
    fM30  : LREAL;
    fM21  : LREAL;
    fM12  : LREAL;
    fM03  : LREAL;
    fMu20 : LREAL;
    fMu11 : LREAL;
    fMu02 : LREAL;
    fMu30 : LREAL;
    fMu21 : LREAL;
    fMu12 : LREAL;
    fMu03 : LREAL;
    fNu20 : LREAL;
    fNu11 : LREAL;
    fNu02 : LREAL;
    fNu30 : LREAL;
    fNu21 : LREAL;
    fNu12 : LREAL;
    fNu03 : LREAL;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
fM00	LREAL	空间矩 00
fM10	LREAL	空间矩 10
fM01	LREAL	空间矩 01
fM20	LREAL	空间矩 20
fM11	LREAL	空间矩 11
fM02	LREAL	空间矩 02
fM30	LREAL	空间矩 30
fM21	LREAL	空间矩 21
fM12	LREAL	空间矩 12
fM03	LREAL	空间矩 03
fMu20	LREAL	中心矩 20
fMu11	LREAL	中心矩 11
fMu02	LREAL	中心矩 02
fMu30	LREAL	中心矩 30
fMu21	LREAL	中心矩 21
fMu12	LREAL	中心矩 12
fMu03	LREAL	中心矩 03
fNu20	LREAL	归一化中心矩 20
fNu11	LREAL	归一化中心矩 11
fNu02	LREAL	归一化中心矩 02
fNu30	LREAL	归一化中心矩 30
fNu21	LREAL	归一化中心矩 21
fNu12	LREAL	归一化中心矩 12
fNu03	LREAL	归一化中心矩 03

6.1.2.5.18 TcVnParamsAGAST

提供 AGAST 算法的参数。

语法

定义：

```
TYPE TcVnParamsAGAST :
STRUCT
    nThreshold         : DINT;
    bNonMaxSuppression : BOOL;
    eType              : ETcVnKeypointDetectionTypeAGAST;
END_STRUCT
END_TYPE
```

参数

名称	类型	默认值	描述
nThreshold	DINT	10	中心像素及其外接圆的像素值差阈值
bNonMaxSuppression	BOOL	TRUE	如果为 true，则应用非极大值抑制。
eType	ETcVnKeypointDetectionTypeAGAST [▶ 199]	TCVN_KDT_AGAST_9_16	邻域类型

6.1.2.5.19 TcVnParamsAKAZE

提供 AKAZE 算法的参数。

语法

定义：

```
TYPE TcVnParamsAKAZE :
STRUCT
    eDescrType     : ETcVnFeatureDescriptorTypeAKAZE;
    nDescrSize     : UDINT;
    nDescrChannels : UDINT;
    fThreshold     : REAL;
    nOctaves       : UDINT;
    nOctaveLayers  : UDINT;
    eDiffusivity   : ETcVnDiffusivityTypeKAZE;
END_STRUCT
END_TYPE
```

参数

名称	类型	默认值	描述
eDescrType	ETcVnFeatureDescriptorTypeAKAZE [▶ 193]	TCVN_FDT_AKAZE_MLDB	描述符类型
nDescrSize	UDINT	0	描述符大小（单位：位）（仅适用于 MLDB；0 = 完整大小）
nDescrChannels	UDINT	3	描述符通道数（目前 MLDB 仅支持 3 个通道）
fThreshold	REAL	0.001	检测器响应阈值
尺度空间数	UDINT	2	最大尺度空间数
nOctaveLayers	UDINT	1	每个尺度等级的分等级数（如：某个尺度空间的金字塔层的级数）

名称	类型	默认值	描述
eDiffusivity	ETcVnDiffusivityTypeKAZE [▶ 186]	TCVN_DT1_KAZE_PM_G2	扩散类型

6.1.2.5.20 TcVnParamsBlobDetection

提供斑点检测参数

语法

定义：

```
TYPE TcVnParamsBlobDetection :
STRUCT
    bFilterByArea         : BOOL;
    bFilterByCircularity  : BOOL;
    bFilterByConvexity    : BOOL;
    bFilterByEccentricity : BOOL;
    bFilterByInertiaRatio : BOOL;
    fMinArea              : REAL;
    fMaxArea              : REAL;
    fMinCircularity       : REAL;
    fMaxCircularity       : REAL;
    fMinConvexity         : REAL;
    fMaxConvexity         : REAL;
    fMinEccentricity      : REAL;
    fMaxEccentricity      : REAL;
    fMinInertiaRatio      : REAL;
    fMaxInertiaRatio      : REAL;
    eThresholdType        : ETcVnThresholdType;
    fMinThreshold         : REAL;
    fMaxThreshold         : REAL;
    fThresholdStep        : REAL;
    fMinBlobDistance      : REAL;
    nMinRepeatability     : UDINT;
    eBlobCombination      : ETcVnBlobCombination;
    bExcludeBorderBlobs   : BOOL;
END_STRUCT
END_TYPE
```

参数

名称	类型	默认值	描述
bFilterByArea	BOOL	TRUE	启用面积筛选（fMinArea <= 面积 <= fMaxArea）；强烈建议激活使用 fMinArea 进行噪声区域滤除。
bFilterByCircularity	BOOL	FALSE	启用圆度筛选（fMinCircularity <= 圆度（4pi * 面积 / 周长^2）<= fMaxCircularity）。
bFilterByConvexity	BOOL	FALSE	启用凸度筛选（fMinConvexity <= 凸度（面积 / 凸包面积）<= fMaxConvexity）。
bFilterByEccentricity	BOOL	FALSE	启用偏心率筛选（fMinEccentricity <= 偏心率 <= fMaxEccentricity）。
bFilterByInertiaRatio	BOOL	FALSE	启用惯性比筛选（fMinInertiaRatio <= 惯性比 <= fMaxInertiaRatio）。
fMinArea	REAL	10	最小估计斑点面积，以像素为单位

名称	类型	默认值	描述
fMaxArea	REAL	100000000	最大估计斑点面积，以像素为单位
fMinCircularity	REAL	0	最小圆度（1.0：理想圆，< 1：不太圆，0：完全不圆）
fMaxCircularity	REAL	1	最大圆度（1.0：理想圆，< 1：不太圆，0：完全不圆）
fMinConvexity	REAL	0	最小凸度（1.0：斑点完全凸包络线，< 1：凸度较小）
fMaxConvexity	REAL	1	最大凸度（1.0：斑点完全凸包络线，< 1：凸度较小）
fMinEccentricity	REAL	0	最小偏心率（0.0：圆，1.0：直线）
fMaxEccentricity	REAL	1	最大偏心率（0.0：圆，1.0：直线）
fMinInertiaRatio	REAL	0	最小惯性比（1.0：等宽等高，0.0：直线）
fMaxInertiaRatio	REAL	1	最大惯性比（1.0：等宽等高，0.0：直线）
eThresholdType	ETcVnThresholdType [▶ 213]	TCVN_TT_BINARY	内部应用阈值法的阈值类型（仅 USINT 图像支持 OTSU_XXX）。
fMinThreshold	REAL	30	起始阈值（如果 fThresholdStep > 0，否则只能使用该阈值）。
fMaxThreshold	REAL	225	终止阈值（如果 fThresholdStep > 0，否则不使用此值）。
fThresholdStep	REAL	0	如果只使用 1 个阈值，则设置为 0（比使用多个阈值并合并结果快得多）。
fMinBlobDistance	REAL	5	两个不同斑点中心点之间的最小距离（仅在 fThresholdStep > 0 时使用；如果距离 < fMinBlobDistance，则视为两个斑点相同）。
nMinRepeatability	UDINT	2	必须检测到相同轮廓的最小阈值步数（仅在 fThresholdStep > 0 时使用；相同意味着中心点距离 < fMinBlobDistance）。
eBlobCombination	ETcVnBlobCombination [▶ 159]	TCVN_BC_MEDIAN_THRESHOLD	选择应返回哪个多阈值斑点轮廓。
bExcludeBorderBlobs	BOOL	FALSE	如果为 true，则会从结果中排除接触图像边界的斑点。

6.1.2.5.21 TcVnParamsBRISK

提供 BRISK 算法的参数。

语法

定义：

```
TYPE TcVnParamsBRISK :
STRUCT
    nThreshold    : DINT;
    nOctaves      : UDINT;
    fPatternScale : REAL;
END_STRUCT
END_TYPE
```

参数

名称	类型	默认值	描述
nThreshold	DINT	30	检测阈值
nOctaves	UDINT	3	检测尺度空间数（0 表示单尺度）
fPatternScale	REAL	1	邻域模式的比例系数

6.1.2.5.22 TcVnParamsFAST

提供 FAST 算法的参数。

语法

定义：

```
TYPE TcVnParamsFAST :
STRUCT
    nThreshold         : DINT;
    bNonMaxSuppression : BOOL;
    eType              : ETcVnKeypointDetectionTypeFAST;
END_STRUCT
END_TYPE
```

参数

名称	类型	默认值	描述
nThreshold	DINT	10	中心像素及其外接圆的像素值差阈值
bNonMaxSuppression	BOOL	TRUE	如果为 true，则应用非极大值抑制。
eType	ETcVnKeypointDetectionTypeFAST [▶ 199]	TCVN_KDT_FAST_9_16	邻域类型

6.1.2.5.23 TcVnParamsGFTT

提供 GFTT 算法的参数。

语法

定义：

```
TYPE TcVnParamsGFTT :
STRUCT
    nMaxCorners        : UDINT;
    fQualityLevel      : LREAL;
    fMinDistance       : LREAL;
    nBlockSize         : UDINT;
    bUseHarrisDetector : BOOL;
```

```
    fHarrisK           : LREAL;
END_STRUCT
END_TYPE
```

参数

名称	类型	默认值	描述
nMaxCorners	UDINT	1000	返回角点的最大数量（最强角点）
fQualityLevel	LREAL	0.01	相对于最强角点可接受的最低角点质量
fMinDistance	LREAL	1	返回关键点之间的最小欧几里得距离
nBlockSize	UDINT	3	邻域块的大小
bUseHarrisDetector	BOOL	FALSE	如果为 true，则使用哈里斯（Harris）检测器代替默认方法。
fHarrisK	LREAL	0.04	哈里斯（Harris）探测器的自由参数

6.1.2.5.24 TcVnParamsKAZE

提供 KAZE 算法的参数

语法

定义：

```
TYPE TcVnParamsKAZE :
STRUCT
    bExtended     : BOOL;
    bUpright      : BOOL;
    fThreshold    : REAL;
    nOctaves      : UDINT;
    nOctaveLayers : UDINT;
    eDiffusivity  : ETcVnDiffusivityTypeKAZE;
END_STRUCT
END_TYPE
```

参数

名称	类型	默认值	描述
bExtended	BOOL	FALSE	如果为 true，描述符大小将从默认的 64 字节扩展到 128 字节。
bUpright	BOOL	FALSE	如果为true，upright描述符生成基于原始图像具有（旋转依赖性）。
fThreshold	REAL	0.001	检测器响应阈值
nOctaves	UDINT	4	图像的最大尺度空间层数
nOctaveLayers	UDINT	2	每个尺度等级的分等级数（如：某个尺度空间的金字塔层的级数）
eDiffusivity	ETcVnDiffusivityTypeKAZE [▶ 186]	TCVN_DT1_KAZE_PM_G2	扩散类型

6.1.2.5.25 TcVnParamsMSER

提供 MSER 算法的参数

语法

定义：

```
TYPE TcVnParamsMSER :
STRUCT
    nDelta         : DINT;
    nMinArea       : UDINT;
    nMaxArea       : UDINT;
    fMaxVariation  : LREAL;
    fMinDiversity  : LREAL;
    nMaxEvolution  : UDINT;
    fAreaThreshold : LREAL;
    fMinMargin     : LREAL;
    nEdgeBlurSize  : UDINT;
END_STRUCT
END_TYPE
```

参数

名称	类型	默认值	描述
nDelta	DINT	5	用于尺寸比较的 delta
nMinArea	UDINT	60	最小斑点尺寸
nMaxArea	UDINT	14400	最大斑点尺寸
fMaxVariation	LREAL	0.25	最大斑点尺寸变化
fMinDiversity	LREAL	0.2	最小 MSER 多样性
nMaxEvolution	UDINT	200	最大进化步长（仅用于彩色图像）
fAreaThreshold	LREAL	1.01	导致重新初始化的面积阈值（仅用于彩色图像）
fMinMargin	LREAL	0.003	最小边际（仅用于彩色图像）
nEdgeBlurSize	UDINT	5	边缘模糊的孔径大小（仅用于彩色图像）

6.1.2.5.26 TcVnParamsORB

提供 ORB 算法的参数。

语法

定义：

```
TYPE TcVnParamsORB :
STRUCT
    nMaxPoints     : UDINT;
    fPyramidScale  : REAL;
    nPyramidLevels : UDINT;
    nEdgeThreshold : UDINT;
    nFirstLevel    : UDINT;
    nBriefPoints   : UDINT;
    eScoreType     : ETcVnFeatureScoreTypeORB;
    nPatchSize     : UDINT;
    nFastThreshold : DINT;
END_STRUCT
END_TYPE
```

参数

名称	类型	默认值	描述
nMaxPoints	UDINT	500	返回关键点的最大数量
fPyramidScale	REAL	1.2	金字塔抽取比率（必须大于 1，应小于 2）
nPyramidLevels	UDINT	8	金字塔层数
nEdgeThreshold	UDINT	31	未检测到特征的边界尺寸（应与 nPatchSize 匹配）

名称	类型	默认值	描述
nFirstLevel	UDINT	0	金字塔第一层（目前仅支持 0）
nBriefPoints	UDINT	2	生成每个 BRIEF 描述元素的点数（2、3、4）
eScoreType	ETcVnFeatureScoreTypeORB [▶ 194]	TCVN_FST_ORB_HARRIS	分数类型（HARRIS 更稳定，但比 FAST 稍慢）
nPatchSize	UDINT	31	BRIEF 描述符的补丁大小
nFastThreshold	DINT	20	FAST 关键点检测阈值

6.1.2.5.27 TcVnParamsSB

提供 SB 算法（一种具有多个阈值的简单斑点检测器）的参数。

语法

定义：

```
TYPE TcVnParamsSB :
STRUCT
    bFilterByArea         : BOOL;
    bFilterByCircularity  : BOOL;
    bFilterByColor        : BOOL;
    bFilterByConvexity    : BOOL;
    bFilterByInertia      : BOOL;
    fMinArea              : REAL;
    fMaxArea              : REAL;
    fMinCircularity       : REAL;
    fMaxCircularity       : REAL;
    nBlobColor            : USINT;
    fMinConvexity         : REAL;
    fMaxConvexity         : REAL;
    fMinInertiaRatio      : REAL;
    fMaxInertiaRatio      : REAL;
    fMinBlobDist          : REAL;
    nMinRepeatability     : UDINT;
    fMinThreshold         : REAL;
    fMaxThreshold         : REAL;
    fThresholdStep        : REAL;
END_STRUCT
END_TYPE
```

参数

名称	类型	默认值	描述
bFilterByArea	BOOL	TRUE	启用面积筛选（fMinArea <= 面积 < fMaxArea）。
bFilterByCircularity	BOOL	FALSE	启用圆度筛选（fMinCircularity <= 圆度（4pi * 面积 / 周长^2）< fMaxCircularity）。
bFilterByColor	BOOL	FALSE	启用颜色筛选（thresholdedColor（0 或 255）= nBlobColor）。
bFilterByConvexity	BOOL	FALSE	启用凸度筛选（fMinConvexity <= 凸度（面积 / 凸包面积）< fMaxConvexity）。
bFilterByInertia	BOOL	FALSE	启用惯性比筛选（fMinInertiaRatio <= 惯性比 < fMaxInertiaRatio）。

名称	类型	默认值	描述
fMinArea	REAL	25	最小估计斑点面积，以像素为单位
fMaxArea	REAL	15000	最大估计斑点面积，以像素为单位
fMinCircularity	REAL	0	最小圆度（1.0：理想圆，< 1：不太圆，0：完全不圆）
fMaxCircularity	REAL	1	最大圆度（1.0：理想圆，< 1：不太圆，0：完全不圆）
nBlobColor	USINT	255	0 或 255
fMinConvexity	REAL	0	最小凸度（1.0：斑点完全凸包络线，< 1：凸度较小）
fMaxConvexity	REAL	1	最大凸度（1.0：斑点完全凸包络线，< 1：凸度较小）
fMinInertiaRatio	REAL	0	最小惯性比（0.0... 1.0）
fMaxInertiaRatio	REAL	1	最大惯性比（0.0... 1.0）
fMinBlobDist	REAL	5	不同斑点之间的最小距离
nMinRepeatability	UDINT	2	不同阈值下检测到的相同斑点的最小数量（距离 < fMinBlobDist）
fMinThreshold	REAL	30	最小阈值（开始）
fMaxThreshold	REAL	225	最大阈值（停止）
fThresholdStep	REAL	10	最小阈值和最大阈值之间的阈值步长

6.1.2.5.28 TcVnPixelFormat

包含有关像素格式的详细信息。

语法

定义：

```
TYPE TcVnPixelFormat :
STRUCT
    bSupported     : BIT;
    bSigned        : BIT;
    bPlanar        : BIT;
    bFloat         : BIT;
    nChannels      : BYTE;
    ePixelEncoding : ETcVnPixelEncoding;
    ePixelPackMode : ETcVnPixelPackMode;
    nElementSize   : UINT;
    nTotalSize     : UINT;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
bSupported	BIT	如果为 false，则不支持像素格式。
bSigned	BIT	如果为 true，像素灰度值为带符号值。

名称	类型	描述
bPlanar	BIT	如果为 true，图像通道将采用平面存储，而不是交叉存储（例如，以 RRRRRGGGGGBBBBB 代替 RGBRGBRGBRGBRGB）。
bFloat	BIT	如果为 true，则像素格式为浮点。
nChannels	BYTE	通道数量
ePixelEncoding	ETcVnPixelEncoding [▶ 206]	像素编码
ePixelPackMode	ETcVnPixelPackMode [▶ 207]	像素包模式
nElementSize	UINT	单像素通道的大小（位）
nTotalSize	UINT	所有像素通道的大小（位）

6.1.2.5.29 TcVnRectangle_DINT

包含矩形的原点和大小。

语法

定义：

```
TYPE TcVnRectangle_DINT :
STRUCT
    nX      : DINT;
    nY      : DINT;
    nWidth  : DINT;
    nHeight : DINT;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
nX	DINT	左上角的 X 坐标
nY	DINT	左上角的 Y 坐标
nWidth	DINT	Width（宽度）
nHeight	DINT	高度

6.1.2.5.30 TcVnRectangle_UDINT

包含矩形的原点和大小。

语法

定义：

```
TYPE TcVnRectangle_UDINT :
STRUCT
    nX      : UDINT;
    nY      : UDINT;
    nWidth  : UDINT;
    nHeight : UDINT;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
nX	UDINT	左上角的 X 坐标
nY	UDINT	左上角的 Y 坐标
nWidth	UDINT	Width（宽度）
nHeight	UDINT	高度

6.1.2.5.31 TcVnRotatedRectangle

包含旋转矩形的中心、大小和角度。

语法

定义：

```
TYPE TcVnRotatedRectangle :
STRUCT
    aCenter : TcVnPoint2_REAL;
    stSize  : TcVnSize2_REAL;
    fAngle  : REAL;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
aCenter	TcVnPoint2_REAL [▶ 152]	中心点
stSize	TcVnSize2_REAL [▶ 237]	由 fWidth 和 fHeight 组成的大小
fAngle	REAL	角度（单位：度）

6.1.2.5.32 TcVnSize2_REAL

包含宽度和高度。

语法

定义：

```
TYPE TcVnSize2_REAL :
STRUCT
    fWidth  : REAL;
    fHeight : REAL;
END_STRUCT
END_TYPE
```

参数

名称	类型	描述
fWidth	REAL	Width（宽度）
fHeight	REAL	高度

6.1.3 接口

接口不需要许可证。

接口分组

TwinCAT Vision 库的所有接口按主题分类为以下几组：

- 图像 [▶ 370] ()
- 容器 [▶ 237] ()
- 机器学习 [▶ 415] ()
- 其他 [▶ 418] ()

6.1.3.1 容器

该组包含用于处理容器 [▶ 146]的接口。

常规信息

容器基本上是几个元素组成的矢量。元素的数量可以动态地改变。路由器内存 [▶ 54]中的相应内存为此自动分配和释放。

容器类型

容器的所有元素都为相同的类型。根据容器包含的元素类型，容器的特点为独特的 GUID。为了便于使用，这些 GUID 由常量 [▶ 152]定义。容器的所有类型名称以“ContainerType_Vector_”开始（“ContainerType_String_SINT”除外）。对于所有使用容器的 API 元素，类型要求都有记录。

容器类型可按以下方式确定：

```
ipContainer.GetElementTypeGuid(nTypeGuid);
```

6.1.3.1.1 功能块

该组包含用于实现某些功能块的接口。

6.1.3.1.1.1 ITcVnCustomContainerOperation_ITcVnContainer

为多达 3 个不同容器的自定义容器运算（容器的容器）提供接口。

继承层次

ITcVnCustomContainerOperation_ITcVnContainer

方法

名称	起源	描述
Execute [▶ 238]	ITcVnCustomContainerOperation_ITcVnContainer	对容器元素执行自定义操作。

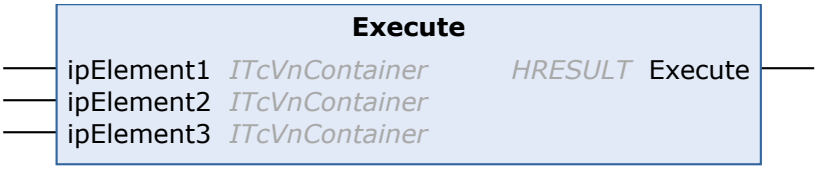
相关函数

- F_VN_CustomElementWiseContainerOperation_ITcVnContainer [▶ 979]

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.1.1.1 *执行*



对容器元素执行自定义操作。

语法

定义：

```
METHOD Execute : HRESULT
VAR_INPUT
    ipElement1 : ITcVnContainer;
```

```
    ipElement2 : ITcVnContainer;
    ipElement3 : ITcVnContainer;
END_VAR
```

输入

名称	类型	描述
ipElement1	ITcVnContainer [▶ 365]	第 1 个容器的当前元素。
ipElement2	ITcVnContainer [▶ 365]	第 2 个容器的当前元素。
ipElement3	ITcVnContainer [▶ 365]	第 3 个容器的当前元素。

返回值

HRESULT [▶ 135]

6.1.3.1.1.2 ITcVnCustomContainerOperation_ITcVnForwardIterator

为多达 3 个不同容器的自定义逐元素容器运算提供接口。

继承层次

ITcVnCustomContainerOperation_ITcVnForwardIterator

方法

名称	起源	描述
Execute [▶ 239]	ITcVnCustomContainerOperation_ITcVnForwardIterator	对容器元素执行自定义操作。

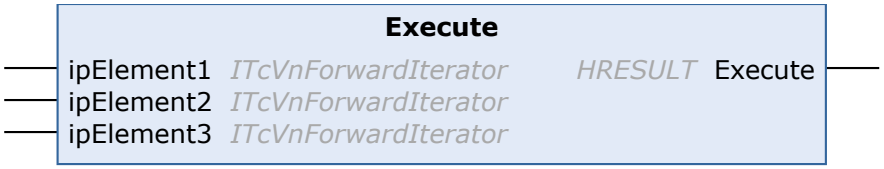
相关函数

- F_VN_CustomElementWiseContainerOperation_ITcVnForwardIterator [▶ 980]

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.1.2.1 *执行*



对容器元素执行自定义操作。

语法

定义：

```
METHOD Execute : HRESULT
VAR_INPUT
    ipElement1 : ITcVnForwardIterator;
    ipElement2 : ITcVnForwardIterator;
    ipElement3 : ITcVnForwardIterator;
END_VAR
```

输入

名称	类型	描述
ipElement1	ITcVnForwardIterator [▶ 355]	第 1 个容器的当前元素。
ipElement2	ITcVnForwardIterator [▶ 355]	第 2 个容器的当前元素。
ipElement3	ITcVnForwardIterator [▶ 355]	第 3 个容器的当前元素。

返回值

HRESULT [▶ 135]

6.1.3.1.1.3 ITcVnCustomElementCondition_ITcVnContainer

提供一个接口，用于为容器计算的自定义条件。

继承层次

ITcVnCustomElementCondition_ITcVnContainer

方法

名称	起源	描述
Condition [▶ 240]	ITcVnCustomElementCondition_ITcVnContainer	评估容器的条件。

相关函数

- F_VN_CopyContainerElementsConditional_ITcVnContainer [▶ 749]

样本

- 容器元素选择 [▶ 2723]

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.1.3.1 *条件*



评估容器的条件。

语法

定义：

```
METHOD Condition : BOOL
VAR_INPUT
    ipElement : ITcVnContainer;
END_VAR
```

输入

名称	类型	描述
ipElement	ITcVnContainer [▶ 365]	为其评估条件的容器。

返回值

BOOL

6.1.3.1.1.4 ITcVnCustomElementCondition_ITcVnForwardIterator

提供一个接口，用于为 ITcVnForwardIterator 表示的元素计算自定义条件。

继承层次

ITcVnCustomElementCondition_ITcVnForwardIterator

方法

名称	起源	描述
Condition [▶ 241]	ITcVnCustomElementCondition_ITcVnForwardIterator	评估所提供元素的条件。

相关函数

- F_VN_CopyContainerElementsConditional_ITcVnForwardIterator [▶ 750]

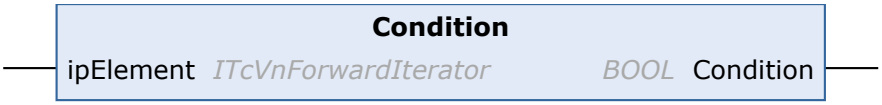
样本

- 容器元素选择 [▶ 2723]

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.1.4.1 *条件*



评估所提供元素的条件。

语法

定义：

```
METHOD Condition : BOOL
VAR_INPUT
    ipElement : ITcVnForwardIterator;
END_VAR
```

输入

名称	类型	描述
ipElement	ITcVnForwardIterator [▶ 355]	为其评估条件的元素。

返回值

BOOL

6.1.3.1.2 ITcVnAccess

接口ITcVnAccess用于访问容器元素 [▶ 146]，并由迭代器 [▶ 354]提供。与ITcVnRandomAccess [▶ 295]不同，只有相关迭代器在那一刻指向的元素可以被寻址。

6.1.3.1.2.1 ITcVnAccess_DINT

Offers an access interface for DINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_DINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 242]	ITcVnAccess_DINT	获取值
Set [▶ 243]	ITcVnAccess_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.1.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    nValue : Reference To DINT;
END_VAR
```

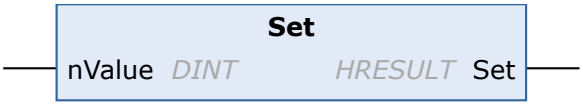
输入

名称	类型	描述
nValue	引用 DINT	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.1.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    nValue : DINT;
END_VAR
```

输入

名称	类型	描述
nValue	DINT	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.2 ITcVnAccess_INT

Offers an access interface for INT values.

继承层次

ITcUnknown [▶ 424]
ITcVnAccess_INT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 244]	ITcVnAccess_INT	获取值
Set [▶ 244]	ITcVnAccess_INT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.2.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    nValue : Reference To INT;
END_VAR
```

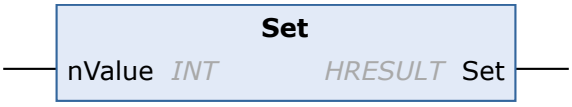
输入

名称	类型	描述
nValue	引用 INT	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.2.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    nValue : INT;
END_VAR
```

输入

名称	类型	描述
nValue	INT	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.3 ITcVnAccess_ITcVnImage

Offers an access interface for images.

继承层次

ITcUnknown [▶ 424]
ITcVnAccess_ITcVnImage

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 245]	ITcVnAccess_ITcVnImage	获取图像。
Set [▶ 245]	ITcVnAccess_ITcVnImage	设置图像。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.3.1 *获取*



获取图像。

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    ipImage : Reference To ITcVnImage;
END_VAR
```

输入

名称	类型	描述
ipImage	引用 ITcVnImage [▶ 407]	返回图像。

返回值

HRESULT [▶ 135]

6.1.3.1.2.3.2 *设置*



设置图像。

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    ipImage : ITcVnImage;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	要设置的图像。

返回值

HRESULT [▶ 135]

6.1.3.1.2.4 ITcVnAccess_LREAL

Offers an access interface for LREAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_LREAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 246]	ITcVnAccess_LREAL	获取值
Set [▶ 247]	ITcVnAccess_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.4.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    fValue : Reference To LREAL;
END_VAR
```

输入

名称	类型	描述
fValue	引用 LREAL	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.4.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    fValue : LREAL;
END_VAR
```

输入

名称	类型	描述
fValue	LREAL	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.5 ITcVnAccess_REAL

Offers an access interface for REAL values.

继承层次

ITcUnknown [▶ 424]
ITcVnAccess_REAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 248]	ITcVnAccess_REAL	获取值
Set [▶ 248]	ITcVnAccess_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.5.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    fValue : Reference To REAL;
END_VAR
```

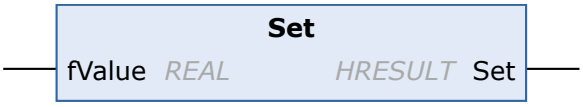
输入

名称	类型	描述
fValue	引用 REAL	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.5.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    fValue : REAL;
END_VAR
```

输入

名称	类型	描述
fValue	REAL	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.6 ITcVnAccess_SINT

Offers an access interface for SINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_SINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 249]	ITcVnAccess_SINT	获取值
Set [▶ 249]	ITcVnAccess_SINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.6.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    nValue : Reference To SINT;
END_VAR
```

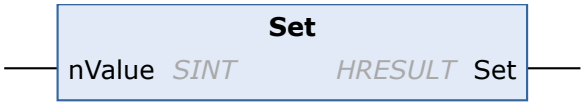
输入

名称	类型	描述
nValue	引用 SINT	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.6.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    nValue : SINT;
END_VAR
```

输入

名称	类型	描述
nValue	SINT	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.7 ITcVnAccess_TcVnDMatch

Offers an access interface for TcVnDMatch values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnDMatch

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 250]	ITcVnAccess_TcVnDMatch	获取值
Set [▶ 251]	ITcVnAccess_TcVnDMatch	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.7.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    stDMatch : Reference To TcVnDMatch;
END_VAR
```

输入

名称	类型	描述
stDMatch	引用 TcVnDMatch [▶ 225]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.7.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    stDMatch : Reference To TcVnDMatch;
END_VAR
```

输入

名称	类型	描述
stDMatch	引用 TcVnDMatch [▶ 225]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.8 ITcVnAccess_TcVnKeyPoint

Offers an access interface for TcVnKeyPoint values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnKeyPoint

方法

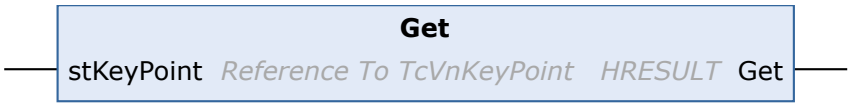
名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 252]	ITcVnAccess_TcVnKeyPoint	获取值。

名称	起源	描述
Set [▶ 252]	ITcVnAccess_TcVnKeyPoint	设置值。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.8.1 *获取*



获取值。

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    stKeyPoint : Reference To TcVnKeyPoint;
END_VAR
```

输入

名称	类型	描述
stKeyPoint	引用 TcVnKeyPoint [▶ 226]	返回值。

返回值

HRESULT [▶ 135]

6.1.3.1.2.8.2 *设置*



设置值。

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    stKeyPoint : Reference To TcVnKeyPoint;
END_VAR
```

输入

名称	类型	描述
stKeyPoint	引用 TcVnKeyPoint [▶ 226]	要设置的值。

返回值

HRESULT [▶ 135]

6.1.3.1.2.9 ITcVnAccess_TcVnPoint2_DINT

Offers an access interface for TcVnPoint2_DINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnPoint2_DINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 253]	ITcVnAccess_TcVnPoint2_DINT	获取值。
Set [▶ 254]	ITcVnAccess_TcVnPoint2_DINT	设置值。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.9.1 *获取*



获取值。

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aPoint : Reference To TcVnPoint2_DINT;
END_VAR
```

输入

名称	类型	描述
aPoint	引用 TcVnPoint2_DINT [▶ 152]	返回值。

返回值

HRESULT [▶ 135]

6.1.3.1.2.9.2 *设置*



设置值。

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aPoint : Reference To TcVnPoint2_DINT;
END_VAR
```

输入

名称	类型	描述
aPoint	引用 TcVnPoint2_DINT [▶ 152]	要设置的值。

返回值

HRESULT [▶ 135]

6.1.3.1.2.10 ITcVnAccess_TcVnPoint2_LREAL

Offers an access interface for TcVnPoint2_LREAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnPoint2_LREAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 255]	ITcVnAccess_TcVnPoint2_LREAL	获取值
Set [▶ 255]	ITcVnAccess_TcVnPoint2_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.10.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aPoint : Reference To TcVnPoint2_LREAL;
END_VAR
```

输入

名称	类型	描述
aPoint	引用 TcVnPoint2_LREAL [▶ 152]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.10.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aPoint : Reference To TcVnPoint2_LREAL;
END_VAR
```

输入

名称	类型	描述
aPoint	引用 TcVnPoint2_LREAL [▶ 152]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.11 ITcVnAccess_TcVnPoint2_REAL

Offers an access interface for TcVnPoint2_REAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnPoint2_REAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 256]	ITcVnAccess_TcVnPoint2_REAL	获取值
Set [▶ 256]	ITcVnAccess_TcVnPoint2_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.11.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aPoint : Reference To TcVnPoint2_REAL;
END_VAR
```

输入

名称	类型	描述
aPoint	引用 TcVnPoint2_REAL [▶ 152]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.11.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aPoint : Reference To TcVnPoint2_REAL;
END_VAR
```

输入

名称	类型	描述
aPoint	引用 TcVnPoint2_REAL [▶ 152]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.12 ITcVnAccess_TcVnPoint3_LREAL

Offers an access interface for TcVnPoint3_LREAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnPoint3_LREAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 257]	ITcVnAccess_TcVnPoint3_LREAL	获取值
Set [▶ 258]	ITcVnAccess_TcVnPoint3_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.12.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aPoint : Reference To TcVnPoint3_LREAL;
END_VAR
```

输入

名称	类型	描述
aPoint	引用 TcVnPoint3_LREAL [▶ 152]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.12.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aPoint : Reference To TcVnPoint3_LREAL;
END_VAR
```

输入

名称	类型	描述
aPoint	引用 TcVnPoint3_LREAL [▶ 152]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.13 ITcVnAccess_TcVnPoint3_REAL

Offers an access interface for TcVnPoint3_REAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnPoint3_REAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。

名称	起源	描述
Get [▶ 259]	ITcVnAccess_TcVnPoint3_REAL	获取值
Set [▶ 259]	ITcVnAccess_TcVnPoint3_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.13.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aPoint : Reference To TcVnPoint3_REAL;
END_VAR
```

输入

名称	类型	描述
aPoint	引用 TcVnPoint3_REAL [▶ 152]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.13.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aPoint : Reference To TcVnPoint3_REAL;
END_VAR
```

输入

名称	类型	描述
aPoint	引用 TcVnPoint3_REAL [▶ 152]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.14 ITcVnAccess_TcVnRectangle_DINT

Offers an access interface for TcVnRectangle_DINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnRectangle_DINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 260]	ITcVnAccess_TcVnRectangle_DINT	获取值
Set [▶ 261]	ITcVnAccess_TcVnRectangle_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.14.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    stRectangle : Reference To TcVnRectangle_DINT;
END_VAR
```

输入

名称	类型	描述
stRectangle	引用 TcVnRectangle_DINT [▶ 236]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.14.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    stRectangle : Reference To TcVnRectangle_DINT;
END_VAR
```

输入

名称	类型	描述
stRectangle	引用 TcVnRectangle_DINT [▶ 236]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.15 ITcVnAccess_TcVnVector2_DINT

Offers an access interface for TcVnVector2_DINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector2_DINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 262]	ITcVnAccess_TcVnVector2_DINT	获取值

名称	起源	描述
Set [▶ 262]	ITcVnAccess_TcVnVector2_DINT	设置值。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.15.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector2_DINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector2_DINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.15.2 *设置*



设置值。

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector2_DINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector2_DINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.16 ITcVnAccess_TcVnVector2_INT

Offers an access interface for TcVnVector2_INT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector2_INT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 263]	ITcVnAccess_TcVnVector2_INT	获取值
Set [▶ 264]	ITcVnAccess_TcVnVector2_INT	设置值。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.16.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector2_INT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector2_INT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.16.2 *设置*



设置值。

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector2_INT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector2_INT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.17 ITcVnAccess_TcVnVector2_REAL

Offers an access interface for TcVnVector2_REAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector2_REAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 265]	ITcVnAccess_TcVnVector2_REAL	获取值
Set [▶ 265]	ITcVnAccess_TcVnVector2_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.17.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector2_REAL;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector2_REAL [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.17.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector2_REAL;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector2_REAL [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.18 ITcVnAccess_TcVnVector2_SINT

Offers an access interface for TcVnVector2_SINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector2_SINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 266]	ITcVnAccess_TcVnVector2_SINT	获取值
Set [▶ 266]	ITcVnAccess_TcVnVector2_SINT	设置值。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.18.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector2_SINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector2_SINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.18.2 *设置*



设置值。

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector2_SINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector2_SINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.19 ITcVnAccess_TcVnVector2_UINT

Offers an access interface for TcVnVector2_UINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector2_UINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 267]	ITcVnAccess_TcVnVector2_UINT	获取值
Set [▶ 268]	ITcVnAccess_TcVnVector2_UINT	设置值。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.19.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector2_UINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector2_UINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.19.2 *设置*



设置值。

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector2_UINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector2_UINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.20 ITcVnAccess_TcVnVector2_USINT

Offers an access interface for TcVnVector2_USINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector2_USINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。

名称	起源	描述
Get [▶ 269]	ITcVnAccess_TcVnVector2_USINT	获取值
Set [▶ 269]	ITcVnAccess_TcVnVector2_USINT	设置值。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.20.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector2_USINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector2_USINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.20.2 *设置*



设置值。

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector2_USINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector2_USINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.21 ITcVnAccess_TcVnVector3_INT

Offers an access interface for TcVnVector3_INT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector3_INT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 270]	ITcVnAccess_TcVnVector3_INT	获取值
Set [▶ 271]	ITcVnAccess_TcVnVector3_INT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.21.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector3_INT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector3_INT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.21.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector3_INT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector3_INT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.22 ITcVnAccess_TcVnVector3_REAL

Offers an access interface for TcVnVector3_REAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector3_REAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 272]	ITcVnAccess_TcVnVector3_REAL	获取值
Set [▶ 272]	ITcVnAccess_TcVnVector3_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.22.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector3_REAL;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector3_REAL [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.22.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector3_REAL;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector3_REAL [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.23 ITcVnAccess_TcVnVector3_SINT

Offers an access interface for TcVnVector3_SINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector3_SINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 273]	ITcVnAccess_TcVnVector3_SINT	获取值
Set [▶ 274]	ITcVnAccess_TcVnVector3_SINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.23.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector3_SINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector3_SINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.23.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector3_SINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector3_SINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.24 ITcVnAccess_TcVnVector3_UINT

Offers an access interface for TcVnVector3_UINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector3_UINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 275]	ITcVnAccess_TcVnVector3_UINT	获取值
Set [▶ 275]	ITcVnAccess_TcVnVector3_UINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.24.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector3_UINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector3_UINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.24.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector3_UINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector3_UINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.25 ITcVnAccess_TcVnVector3_USINT

Offers an access interface for TcVnVector3_USINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector3_USINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 276]	ITcVnAccess_TcVnVector3_USINT	获取值
Set [▶ 276]	ITcVnAccess_TcVnVector3_USINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.25.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector3_USINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector3_USINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.25.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector3_USINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector3_USINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.26 ITcVnAccess_TcVnVector4_DINT

Offers an access interface for TcVnVector4_DINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector4_DINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 277]	ITcVnAccess_TcVnVector4_DINT	获取值
Set [▶ 278]	ITcVnAccess_TcVnVector4_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.26.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector4_DINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector4_DINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.26.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector4_DINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector4_DINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.27 ITcVnAccess_TcVnVector4_INT

Offers an access interface for TcVnVector4_INT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector4_INT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。

名称	起源	描述
Get [▶ 279]	ITcVnAccess_TcVnVector4_INT	获取值
Set [▶ 279]	ITcVnAccess_TcVnVector4_INT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.27.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector4_INT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector4_INT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.27.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector4_INT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector4_INT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.28 ITcVnAccess_TcVnVector4_LREAL

Offers an access interface for TcVnVector4_LREAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector4_LREAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 280]	ITcVnAccess_TcVnVector4_LREAL	获取值
Set [▶ 281]	ITcVnAccess_TcVnVector4_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.28.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector4_LREAL;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector4_LREAL [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.28.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector4_LREAL;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector4_LREAL [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.29 ITcVnAccess_TcVnVector4_SINT

Offers an access interface for TcVnVector4_SINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector4_SINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 282]	ITcVnAccess_TcVnVector4_SINT	获取值

名称	起源	描述
Set [▶ 282]	ITcVnAccess_TcVnVector4_SINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.29.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector4_SINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector4_SINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.29.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector4_SINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector4_SINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.30 ITcVnAccess_TcVnVector4_UINT

Offers an access interface for TcVnVector4_UINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector4_UINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 283]	ITcVnAccess_TcVnVector4_UINT	获取值
Set [▶ 284]	ITcVnAccess_TcVnVector4_UINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.30.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector4_UINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector4_UINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.30.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector4_UINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector4_UINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.31 ITcVnAccess_TcVnVector4_USINT

Offers an access interface for TcVnVector4_USINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector4_USINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 285]	ITcVnAccess_TcVnVector4_USINT	获取值
Set [▶ 285]	ITcVnAccess_TcVnVector4_USINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.31.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector4_USINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector4_USINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.31.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector4_USINT;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector4_USINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.32 ITcVnAccess_UDINT

Offers an access interface for UDINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_UDINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 286]	ITcVnAccess_UDINT	获取值
Set [▶ 286]	ITcVnAccess_UDINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.32.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    nValue : Reference To UDINT;
END_VAR
```

输入

名称	类型	描述
nValue	引用 UDINT	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.32.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    nValue : UDINT;
END_VAR
```

输入

名称	类型	描述
nValue	UDINT	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.33 ITcVnAccess_UINT

Offers an access interface for UINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_UINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 287]	ITcVnAccess_UINT	获取值
Set [▶ 288]	ITcVnAccess_UINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.33.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    nValue : Reference To UINT;
END_VAR
```

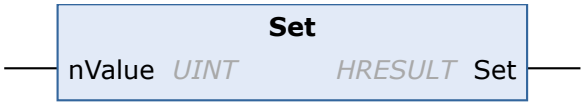
输入

名称	类型	描述
nValue	引用 UINT	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.33.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    nValue : UINT;
END_VAR
```

输入

名称	类型	描述
nValue	UINT	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.34 ITcVnAccess_ULINT

Offers an access interface for ULINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_ULINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 289]	ITcVnAccess_ULINT	获取值
Set [▶ 289]	ITcVnAccess_ULINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.34.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    nValue : Reference To ULINT;
END_VAR
```

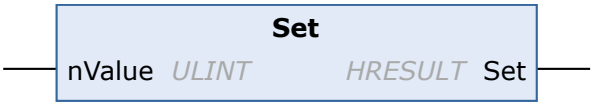
输入

名称	类型	描述
nValue	引用 ULINT	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.34.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    nValue : ULINT;
END_VAR
```

输入

名称	类型	描述
nValue	ULINT	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.35 ITcVnAccess_USINT

Offers an access interface for USINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_USINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 290]	ITcVnAccess_USINT	获取值
Set [▶ 290]	ITcVnAccess_USINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.35.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    nValue : Reference To USINT;
END_VAR
```

输入

名称	类型	描述
nValue	引用 USINT	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.35.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    nValue : USINT;
END_VAR
```

输入

名称	类型	描述
nValue	USINT	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.36 ITcVnAccess_TcVnVector4_REAL

Offers an access interface for TcVnVector4_REAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector4_REAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 291]	ITcVnAccess_TcVnVector4_REAL	获取值
Set [▶ 292]	ITcVnAccess_TcVnVector4_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.36.1 *获取*



获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector4_REAL;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector4_REAL [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.36.2 *设置*



设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    aVector : Reference To TcVnVector4_REAL;
END_VAR
```

输入

名称	类型	描述
aVector	引用 TcVnVector4_REAL [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.2.37 ITcVnAccess_TcVnRotatedRectangle

Offers an access interface for TcVnRotatedRectangle values.

Inheritance Hierarchy

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnRotatedRectangle

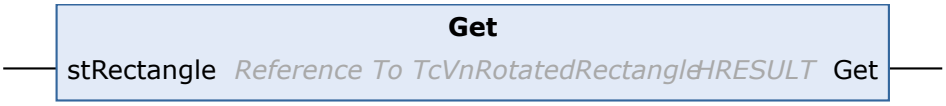
Methods

Name	Origin	Description
TcAddRef [▶ 424]	ITcUnknown	Increments the reference counter.
TcQueryInterface [▶ 425]	ITcUnknown	Get a reference to an implemented interface.
TcRelease [▶ 426]	ITcUnknown	Decrements the reference counter.
Get [▶ 293]	ITcVnAccess_TcVnRotatedRectangle	Gets the value
Set [▶ 293]	ITcVnAccess_TcVnRotatedRectangle	Sets the value

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.37.1 *Get*



Gets the value

Syntax

Definition:

```
METHOD Get : HRESULT
VAR_INPUT
    stRectangle : Reference To TcVnRotatedRectangle;
END_VAR
```

Inputs

Name	Type	Description
stRectangle	Reference To TcVnRotatedRectangle [▶ 237]	Returns the value

Return value

HRESULT [▶ 135]

6.1.3.1.2.37.2 *Set*



Sets the value

Syntax

Definition:

```
METHOD Set : HRESULT
VAR_INPUT
    stRectangle : Reference To TcVnRotatedRectangle;
END_VAR
```

Inputs

Name	Type	Description
stRectangle	Reference To TcVnRotatedRectangle [▶ 237]	The value to set

Return value

HRESULT [▶ 135]

6.1.3.1.2.38 ITcVnAccess_TcVnRotatedRectangle

提供 TcVnRotatedRectangle 值的访问接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnRotatedRectangle

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 294]	ITcVnAccess_TcVnRotatedRectangle	获取值
Set [▶ 294]	ITcVnAccess_TcVnRotatedRectangle	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.2.38.1 *获取*

获取值

语法

定义：

```
METHOD Get : HRESULT
VAR_INPUT
    stRectangle : Reference To TcVnRotatedRectangle;
END_VAR
```

输入

名称	类型	描述
stRectangle	引用 TcVnRotatedRectangle [▶ 237]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.2.38.2 *设置*

设置值

语法

定义：

```
METHOD Set : HRESULT
VAR_INPUT
    stRectangle : Reference To TcVnRotatedRectangle;
END_VAR
```

输入

名称	类型	描述
stRectangle	引用 TcVnRotatedRectangle [▶ 237]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3 ITcVnRandomAccess

接口ITcVnRandomAccess用于访问容器的任何元素 [▶ 146]，并由ITcVnRandomAccessIterator [▶ 361]提供。与ITcVnAccess [▶ 242]不同，容器中的任何元素都可以被寻址。

6.1.3.1.3.1 ITcVnRandomAccess_DINT

Offers a random access interface for DINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_DINT [▶ 242]
    ITcVnRandomAccess_DINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 242]	ITcVnAccess_DINT	获取值
Set [▶ 243]	ITcVnAccess_DINT	设置值
GetAt [▶ 295]	ITcVnRandomAccess_DINT	获取值
SetAt [▶ 296]	ITcVnRandomAccess_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.1.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    nValue  : Reference To DINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
nValue	引用 DINT	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.1.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    nValue  : DINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
nValue	DINT	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.2 ITcVnRandomAccess_INT

Offers a random access interface for INT values

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_INT [▶ 243]
    ITcVnRandomAccess_INT

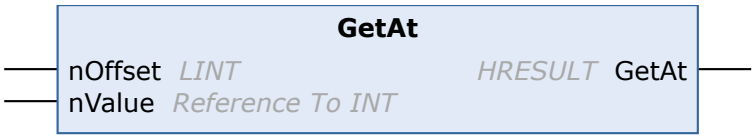
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 244]	ITcVnAccess_INT	获取值
Set [▶ 244]	ITcVnAccess_INT	设置值
GetAt [▶ 297]	ITcVnRandomAccess_INT	获取值
SetAt [▶ 297]	ITcVnRandomAccess_INT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.2.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    nValue  : Reference To INT;
END_VAR
```

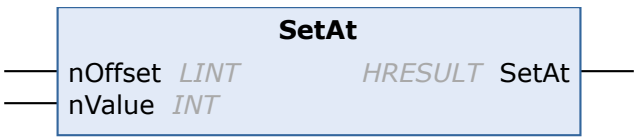
输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
nValue	引用 INT	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.2.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    nValue  : INT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
nValue	INT	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.3 ITcVnRandomAccess_ITcVnImage

Offers an random access interface for images.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_ITcVnImage [▶ 245]
    ITcVnRandomAccess_ITcVnImage

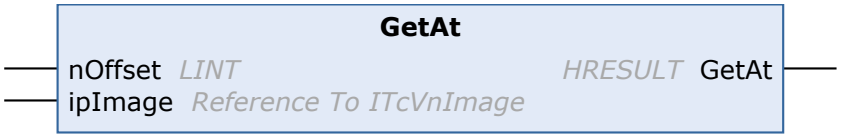
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 245]	ITcVnAccess_ITcVnImage	获取图像。
Set [▶ 245]	ITcVnAccess_ITcVnImage	设置图像。
GetAt [▶ 298]	ITcVnRandomAccess_ITcVnImage	获取图像。
SetAt [▶ 299]	ITcVnRandomAccess_ITcVnImage	设置图像。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.3.1 *GetAt*



获取图像。

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    ipImage : Reference To ITcVnImage;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量。
ipImage	引用 ITcVnImage [▶ 407]	返回图像。

返回值

HRESULT [▶ 135]

6.1.3.1.3.3.2 *SetAt*



设置图像。

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    ipImage : ITcVnImage;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
ipImage	ITcVnImage [▶ 407]	要设置的图像。

返回值

HRESULT [▶ 135]

6.1.3.1.3.4 ITcVnRandomAccess_LREAL

Offers a random access interface for LREAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_LREAL [▶ 246]
    ITcVnRandomAccess_LREAL

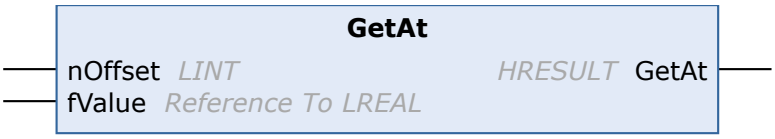
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 246]	ITcVnAccess_LREAL	获取值
Set [▶ 247]	ITcVnAccess_LREAL	设置值
GetAt [▶ 300]	ITcVnRandomAccess_LREAL	获取值
SetAt [▶ 300]	ITcVnRandomAccess_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.4.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    fValue  : Reference To LREAL;
END_VAR
```

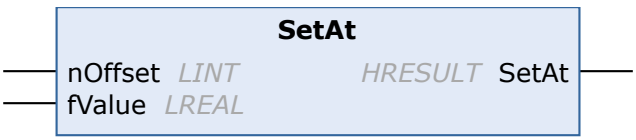
输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
fValue	引用 LREAL	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.4.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    fValue  : LREAL;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量。
fValue	LREAL	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.5 ITcVnRandomAccess_REAL

Offers a random access interface for REAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_REAL [▶ 247]
    ITcVnRandomAccess_REAL

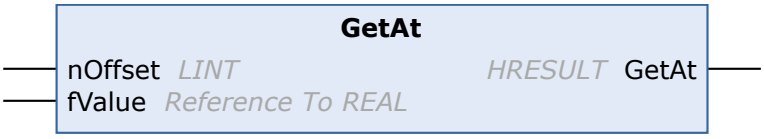
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 248]	ITcVnAccess_REAL	获取值
Set [▶ 248]	ITcVnAccess_REAL	设置值
GetAt [▶ 301]	ITcVnRandomAccess_REAL	获取值
SetAt [▶ 302]	ITcVnRandomAccess_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.5.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    fValue  : Reference To REAL;
END_VAR
```

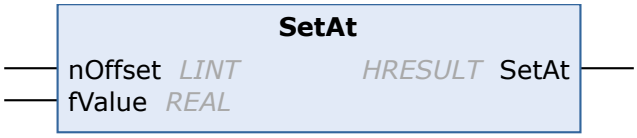
输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
fValue	引用 REAL	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.5.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    fValue  : REAL;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
fValue	REAL	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.6 ITcVnRandomAccess_SINT

Offers a random access interface for SINT values

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_SINT [▶ 249]
    ITcVnRandomAccess_SINT

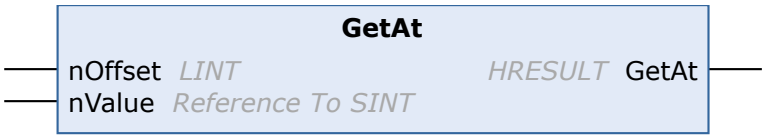
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 249]	ITcVnAccess_SINT	获取值
Set [▶ 249]	ITcVnAccess_SINT	设置值
GetAt [▶ 303]	ITcVnRandomAccess_SINT	获取值
SetAt [▶ 303]	ITcVnRandomAccess_SINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.6.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    nValue  : Reference To SINT;
END_VAR
```

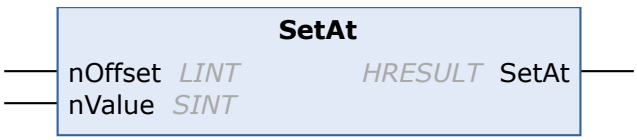
输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
nValue	引用 SINT	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.6.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    nValue  : SINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
nValue	SINT	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.7 ITcVnRandomAccess_TcVnDMatch

Offers an random access interface for TcVnDMatch values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnDMatch [▶ 250]
    ITcVnRandomAccess_TcVnDMatch

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 250]	ITcVnAccess_TcVnDMatch	获取值
Set [▶ 251]	ITcVnAccess_TcVnDMatch	设置值
GetAt [▶ 304]	ITcVnRandomAccess_TcVnDMatch	获取值
SetAt [▶ 305]	ITcVnRandomAccess_TcVnDMatch	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.7.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset  : LINT;
    stDMatch : Reference To TcVnDMatch;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量。
stDMatch	引用 TcVnDMatch [▶ 225]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.7.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset  : LINT;
    stDMatch : Reference To TcVnDMatch;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
stDMatch	引用 TcVnDMatch [▶ 225]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.8 ITcVnRandomAccess_TcVnKeyPoint

Offers a random access interface for TcVnKeyPoint values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnKeyPoint [▶ 251]
    ITcVnRandomAccess_TcVnKeyPoint

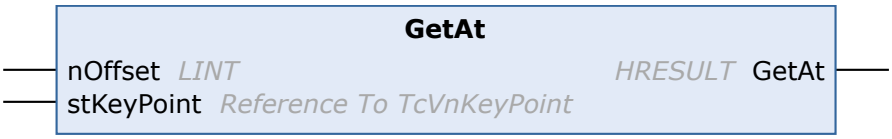
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 252]	ITcVnAccess_TcVnKeyPoint	获取值。
Set [▶ 252]	ITcVnAccess_TcVnKeyPoint	设置值。
GetAt [▶ 306]	ITcVnRandomAccess_TcVnKeyPoint	获取值
SetAt [▶ 306]	ITcVnRandomAccess_TcVnKeyPoint	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.8.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset    : LINT;
    stKeyPoint : Reference To TcVnKeyPoint;
END_VAR
```

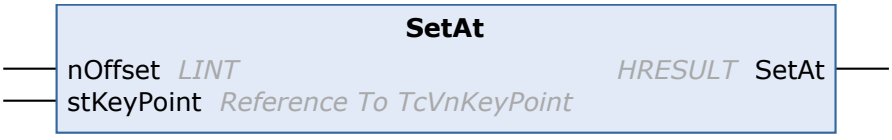
输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
stKeyPoint	引用 TcVnKeyPoint [▶ 226]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.8.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset    : LINT;
    stKeyPoint : Reference To TcVnKeyPoint;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
stKeyPoint	引用 TcVnKeyPoint [▶ 226]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.9 ITcVnRandomAccess_TcVnPoint2_DINT

Offers a random access interface for TcVnPoint2_DINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnPoint2_DINT [▶ 253]
    ITcVnRandomAccess_TcVnPoint2_DINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 253]	ITcVnAccess_TcVnPoint2_DINT	获取值。
Set [▶ 254]	ITcVnAccess_TcVnPoint2_DINT	设置值。
GetAt [▶ 308]	ITcVnRandomAccess_TcVnPoint2_DINT	获取值
SetAt [▶ 308]	ITcVnRandomAccess_TcVnPoint2_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.9.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aPoint  : Reference To TcVnPoint2_DINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aPoint	引用 TcVnPoint2_DINT [▶ 152]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.9.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aPoint  : Reference To TcVnPoint2_DINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aPoint	引用 TcVnPoint2_DINT [▶ 152]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.10 ITcVnRandomAccess_TcVnPoint2_LREAL

Offers a random access interface for TcVnPoint2_LREAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnPoint2_LREAL [▶ 254]
    ITcVnRandomAccess_TcVnPoint2_LREAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 255]	ITcVnAccess_TcVnPoint2_LREAL	获取值
Set [▶ 255]	ITcVnAccess_TcVnPoint2_LREAL	设置值
GetAt [▶ 309]	ITcVnRandomAccess_TcVnPoint2_LREAL	获取值
SetAt [▶ 310]	ITcVnRandomAccess_TcVnPoint2_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.10.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aPoint  : Reference To TcVnPoint2_LREAL;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aPoint	引用 TcVnPoint2_LREAL [▶ 152]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.10.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aPoint  : Reference To TcVnPoint2_LREAL;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aPoint	引用 TcVnPoint2_LREAL [▶ 152]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.11 ITcVnRandomAccess_TcVnPoint2_REAL

Offers a random access interface for TcVnPoint2_REAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnPoint2_REAL [▶ 255]
    ITcVnRandomAccess_TcVnPoint2_REAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 256]	ITcVnAccess_TcVnPoint2_REAL	获取值
Set [▶ 256]	ITcVnAccess_TcVnPoint2_REAL	设置值
GetAt [▶ 311]	ITcVnRandomAccess_TcVnPoint2_REAL	获取值

名称	起源	描述
SetAt [▶ 311]	ITcVnRandomAccess_TcVnPoint2_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.11.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aPoint  : Reference To TcVnPoint2_REAL;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aPoint	引用 TcVnPoint2_REAL [▶ 152]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.11.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aPoint  : Reference To TcVnPoint2_REAL;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aPoint	引用 TcVnPoint2_REAL [▶ 152]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.12 ITcVnRandomAccess_TcVnPoint3_LREAL

Offers a random access interface for TcVnPoint3_LREAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnPoint3_LREAL [▶ 257]
    ITcVnRandomAccess_TcVnPoint3_LREAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 257]	ITcVnAccess_TcVnPoint3_LREAL	获取值
Set [▶ 258]	ITcVnAccess_TcVnPoint3_LREAL	设置值
GetAt [▶ 312]	ITcVnRandomAccess_TcVnPoint3_LREAL	获取值
SetAt [▶ 313]	ITcVnRandomAccess_TcVnPoint3_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.12.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aPoint  : Reference To TcVnPoint3_LREAL;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aPoint	引用 TcVnPoint3_LREAL [▶ 152]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.12.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aPoint  : Reference To TcVnPoint3_LREAL;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aPoint	引用 TcVnPoint3_LREAL [▶ 152]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.13 ITcVnRandomAccess_TcVnPoint3_REAL

Offers a random access interface for TcVnPoint3_REAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnPoint3_REAL [▶ 258]
    ITcVnRandomAccess_TcVnPoint3_REAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 259]	ITcVnAccess_TcVnPoint3_REAL	获取值
Set [▶ 259]	ITcVnAccess_TcVnPoint3_REAL	设置值
GetAt [▶ 314]	ITcVnRandomAccess_TcVnPoint3_REAL	获取值
SetAt [▶ 314]	ITcVnRandomAccess_TcVnPoint3_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.13.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aPoint  : Reference To TcVnPoint3_REAL;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aPoint	引用 TcVnPoint3_REAL [▶ 152]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.13.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aPoint  : Reference To TcVnPoint3_REAL;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aPoint	引用 TcVnPoint3_REAL [▶ 152]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.14 ITcVnRandomAccess_TcVnRectangle_DINT

Offers a random access interface for TcVnRectangle_DINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnRectangle_DINT [▶ 260]
    ITcVnRandomAccess_TcVnRectangle_DINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 260]	ITcVnAccess_TcVnRectangle_DINT	获取值
Set [▶ 261]	ITcVnAccess_TcVnRectangle_DINT	设置值
GetAt [▶ 316]	ITcVnRandomAccess_TcVnRectangle_DINT	获取值
SetAt [▶ 316]	ITcVnRandomAccess_TcVnRectangle_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.14.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset     : LINT;
    stRectangle : Reference To TcVnRectangle_DINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
stRectangle	引用 TcVnRectangle_DINT [▶ 236]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.14.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset     : LINT;
    stRectangle : Reference To TcVnRectangle_DINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
stRectangle	引用 TcVnRectangle_DINT [▶ 236]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.15 ITcVnRandomAccess_TcVnVector2_DINT

Offers a random access interface for TcVnVector2_DINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector2_DINT [▶ 261]
    ITcVnRandomAccess_TcVnVector2_DINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 262]	ITcVnAccess_TcVnVector2_DINT	获取值
Set [▶ 262]	ITcVnAccess_TcVnVector2_DINT	设置值。
GetAt [▶ 317]	ITcVnRandomAccess_TcVnVector2_DINT	获取值
SetAt [▶ 318]	ITcVnRandomAccess_TcVnVector2_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.15.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector2_DINT;
END_VAR
```

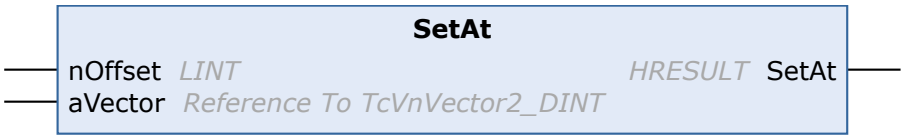
输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector2_DINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.15.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector2_DINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector2_DINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.16 ITcVnRandomAccess_TcVnVector2_INT

Offers a random access interface for TcVnVector2_INT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector2_INT [▶ 263]
    ITcVnRandomAccess_TcVnVector2_INT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 263]	ITcVnAccess_TcVnVector2_INT	获取值
Set [▶ 264]	ITcVnAccess_TcVnVector2_INT	设置值。
GetAt [▶ 319]	ITcVnRandomAccess_TcVnVector2_INT	获取值

名称	起源	描述
SetAt [▶ 319]	ITcVnRandomAccess_TcVnVector2_INT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.16.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector2_INT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector2_INT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.16.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector2_INT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector2_INT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.17 ITcVnRandomAccess_TcVnVector2_REAL

Offers a random access interface for TcVnVector2_REAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector2_REAL [▶ 264]
    ITcVnRandomAccess_TcVnVector2_REAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 265]	ITcVnAccess_TcVnVector2_REAL	获取值
Set [▶ 265]	ITcVnAccess_TcVnVector2_REAL	设置值
GetAt [▶ 320]	ITcVnRandomAccess_TcVnVector2_REAL	获取值
SetAt [▶ 321]	ITcVnRandomAccess_TcVnVector2_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.17.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector2_REAL;
END_VAR
```

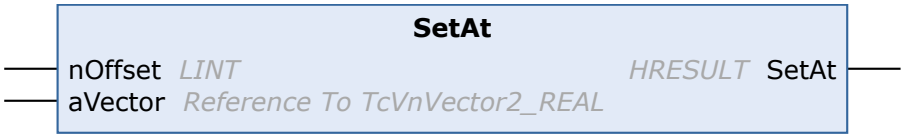
输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector2_REAL [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.17.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector2_REAL;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector2_REAL [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.18 ITcVnRandomAccess_TcVnVector2_SINT

Offers a random access interface for TcVnVector2_SINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector2_SINT [▶ 265]
    ITcVnRandomAccess_TcVnVector2_SINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 266]	ITcVnAccess_TcVnVector2_SINT	获取值
Set [▶ 266]	ITcVnAccess_TcVnVector2_SINT	设置值。
GetAt [▶ 322]	ITcVnRandomAccess_TcVnVector2_SINT	获取值
SetAt [▶ 322]	ITcVnRandomAccess_TcVnVector2_SINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.18.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector2_SINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector2_SINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.18.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector2_SINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector2_SINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.19 ITcVnRandomAccess_TcVnVector2_UINT

Offers a random access interface for TcVnVector2_UINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector2_UINT [▶ 267]
    ITcVnRandomAccess_TcVnVector2_UINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 267]	ITcVnAccess_TcVnVector2_UINT	获取值
Set [▶ 268]	ITcVnAccess_TcVnVector2_UINT	设置值。
GetAt [▶ 324]	ITcVnRandomAccess_TcVnVector2_UINT	获取值
SetAt [▶ 324]	ITcVnRandomAccess_TcVnVector2_UINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.19.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector2_UINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector2_UINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.19.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector2_UINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector2_UINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.20 ITcVnRandomAccess_TcVnVector2_USINT

Offers a random access interface for TcVnVector2_USINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector2_USINT [▶ 268]
    ITcVnRandomAccess_TcVnVector2_USINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 269]	ITcVnAccess_TcVnVector2_USINT	获取值
Set [▶ 269]	ITcVnAccess_TcVnVector2_USINT	设置值。
GetAt [▶ 325]	ITcVnRandomAccess_TcVnVector2_USINT	获取值
SetAt [▶ 326]	ITcVnRandomAccess_TcVnVector2_USINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.20.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector2_USINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector2_USINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.20.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector2_USINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector2_USINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.21 ITcVnRandomAccess_TcVnVector3_INT

Offers a random access interface for TcVnVector3_INT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector3_INT [▶ 270]
    ITcVnRandomAccess_TcVnVector3_INT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 270]	ITcVnAccess_TcVnVector3_INT	获取值
Set [▶ 271]	ITcVnAccess_TcVnVector3_INT	设置值
GetAt [▶ 327]	ITcVnRandomAccess_TcVnVector3_INT	获取值

名称	起源	描述
SetAt [▶ 327]	ITcVnRandomAccess_TcVnVector3_INT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.21.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector3_INT;
END_VAR
```

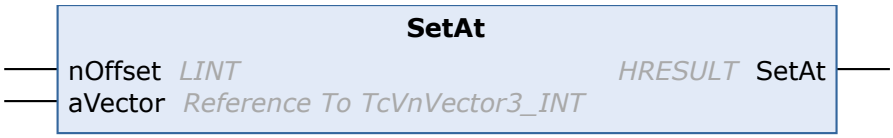
输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector3_INT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.21.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector3_INT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector3_INT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.22 ITcVnRandomAccess_TcVnVector3_REAL

Offers a random access interface for TcVnVector3_REAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector3_REAL [▶ 271]
    ITcVnRandomAccess_TcVnVector3_REAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 272]	ITcVnAccess_TcVnVector3_REAL	获取值
Set [▶ 272]	ITcVnAccess_TcVnVector3_REAL	设置值
GetAt [▶ 328]	ITcVnRandomAccess_TcVnVector3_REAL	获取值
SetAt [▶ 329]	ITcVnRandomAccess_TcVnVector3_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.22.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector3_REAL;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector3_REAL [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.22.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector3_REAL;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector3_REAL [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.23 ITcVnRandomAccess_TcVnVector3_SINT

Offers a random access interface for TcVnVector3_SINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector3_SINT [▶ 273]
    ITcVnRandomAccess_TcVnVector3_SINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 273]	ITcVnAccess_TcVnVector3_SINT	获取值
Set [▶ 274]	ITcVnAccess_TcVnVector3_SINT	设置值
GetAt [▶ 330]	ITcVnRandomAccess_TcVnVector3_SINT	获取值
SetAt [▶ 330]	ITcVnRandomAccess_TcVnVector3_SINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.23.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector3_SINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector3_SINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.23.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector3_SINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector3_SINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.24 ITcVnRandomAccess_TcVnVector3_UINT

Offers a random access interface for TcVnVector3_UINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector3_UINT [▶ 274]
    ITcVnRandomAccess_TcVnVector3_UINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 275]	ITcVnAccess_TcVnVector3_UINT	获取值
Set [▶ 275]	ITcVnAccess_TcVnVector3_UINT	设置值
GetAt [▶ 332]	ITcVnRandomAccess_TcVnVector3_UINT	获取值
SetAt [▶ 332]	ITcVnRandomAccess_TcVnVector3_UINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.24.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector3_UINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector3_UINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.24.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector3_UINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector3_UINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.25 ITcVnRandomAccess_TcVnVector3_USINT

Offers a random access interface for TcVnVector3_USINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector3_USINT [▶ 275]
    ITcVnRandomAccess_TcVnVector3_USINT

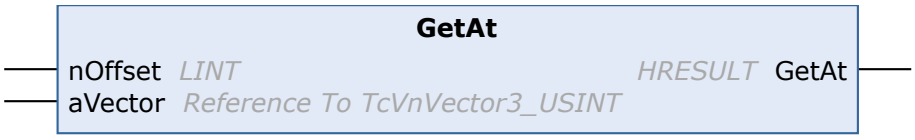
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 276]	ITcVnAccess_TcVnVector3_USINT	获取值
Set [▶ 276]	ITcVnAccess_TcVnVector3_USINT	设置值
GetAt [▶ 333]	ITcVnRandomAccess_TcVnVector3_USINT	获取值
SetAt [▶ 334]	ITcVnRandomAccess_TcVnVector3_USINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.25.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector3_USINT;
END_VAR
```

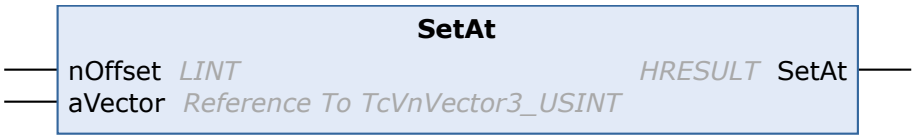
输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector3_USINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.25.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector3_USINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector3_USINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.26 ITcVnRandomAccess_TcVnVector4_DINT

Offers a random access interface for TcVnVector4_DINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector4_DINT [▶ 277]
    ITcVnRandomAccess_TcVnVector4_DINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 277]	ITcVnAccess_TcVnVector4_DINT	获取值
Set [▶ 278]	ITcVnAccess_TcVnVector4_DINT	设置值
GetAt [▶ 335]	ITcVnRandomAccess_TcVnVector4_DINT	获取值

名称	起源	描述
SetAt [▶ 335]	ITcVnRandomAccess_TcVnVector4_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.26.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector4_DINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector4_DINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.26.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector4_DINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector4_DINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.27 ITcVnRandomAccess_TcVnVector4_INT

Offers a random access interface for TcVnVector4_INT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector4_INT [▶ 278]
    ITcVnRandomAccess_TcVnVector4_INT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 279]	ITcVnAccess_TcVnVector4_INT	获取值
Set [▶ 279]	ITcVnAccess_TcVnVector4_INT	设置值
GetAt [▶ 336]	ITcVnRandomAccess_TcVnVector4_INT	获取值
SetAt [▶ 337]	ITcVnRandomAccess_TcVnVector4_INT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.27.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector4_INT;
END_VAR
```

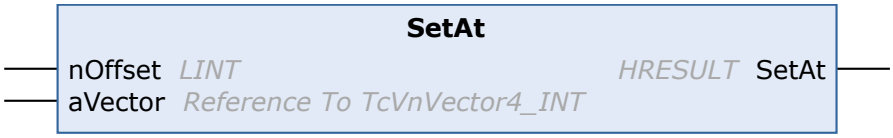
输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector4_INT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.27.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector4_INT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector4_INT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.28 ITcVnRandomAccess_TcVnVector4_LREAL

Offers a random access interface for TcVnVector4_LREAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector4_LREAL [▶ 280]
    ITcVnRandomAccess_TcVnVector4_LREAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 280]	ITcVnAccess_TcVnVector4_LREAL	获取值
Set [▶ 281]	ITcVnAccess_TcVnVector4_LREAL	设置值
GetAt [▶ 338]	ITcVnRandomAccess_TcVnVector4_LREAL	获取值
SetAt [▶ 339]	ITcVnRandomAccess_TcVnVector4_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.28.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector4_LREAL;
END_VAR
```

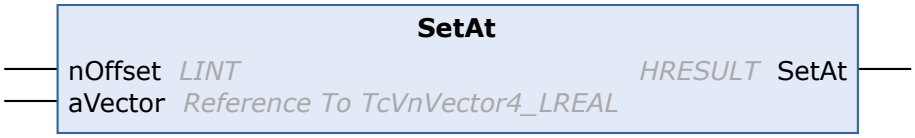
输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector4_LREAL [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.28.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector4_LREAL;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector4_LREAL [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.29 ITcVnRandomAccess_TcVnVector4_SINT

Offers a random access interface for TcVnVector4_SINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector4_SINT [▶ 281]
    ITcVnRandomAccess_TcVnVector4_SINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 282]	ITcVnAccess_TcVnVector4_SINT	获取值
Set [▶ 282]	ITcVnAccess_TcVnVector4_SINT	设置值
GetAt [▶ 340]	ITcVnRandomAccess_TcVnVector4_SINT	获取值
SetAt [▶ 340]	ITcVnRandomAccess_TcVnVector4_SINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.29.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector4_SINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector4_SINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.29.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector4_SINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector4_SINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.30 ITcVnRandomAccess_TcVnVector4_UINT

Offers a random access interface for TcVnVector4_UINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector4_UINT [▶ 283]
    ITcVnRandomAccess_TcVnVector4_UINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 283]	ITcVnAccess_TcVnVector4_UINT	获取值
Set [▶ 284]	ITcVnAccess_TcVnVector4_UINT	设置值
GetAt [▶ 341]	ITcVnRandomAccess_TcVnVector4_UINT	获取值
SetAt [▶ 342]	ITcVnRandomAccess_TcVnVector4_UINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.30.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector4_UINT;
END_VAR
```

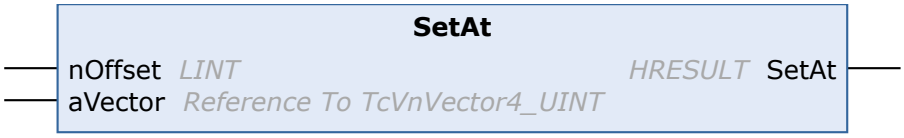
输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector4_UINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.30.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector4_UINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector4_UINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.31 ITcVnRandomAccess_TcVnVector4_USINT

Offers a random access interface for TcVnVector4_USINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector4_USINT [▶ 284]
    ITcVnRandomAccess_TcVnVector4_USINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。

名称	起源	描述
Get [▶ 285]	ITcVnAccess_TcVnVector4_USINT	获取值
Set [▶ 285]	ITcVnAccess_TcVnVector4_USINT	设置值
GetAt [▶ 343]	ITcVnRandomAccess_TcVnVector4_USINT	获取值
SetAt [▶ 343]	ITcVnRandomAccess_TcVnVector4_USINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.31.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector4_USINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector4_USINT [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.31.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector4_USINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector4_USINT [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.32 ITcVnRandomAccess_UDINT

Offers a random access interface for UDINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_UDINT [▶ 285]
    ITcVnRandomAccess_UDINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 286]	ITcVnAccess_UDINT	获取值
Set [▶ 286]	ITcVnAccess_UDINT	设置值
GetAt [▶ 344]	ITcVnRandomAccess_UDINT	获取值
SetAt [▶ 345]	ITcVnRandomAccess_UDINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.32.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    nValue  : Reference To UDINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
nValue	引用 UDINT	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.32.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    nValue  : UDINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
nValue	UDINT	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.33 ITcVnRandomAccess_UINT

Offers a random access interface for UINT values

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_UINT [▶ 287]
    ITcVnRandomAccess_UINT

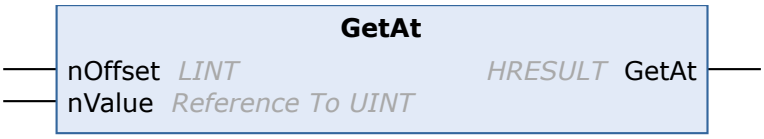
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 287]	ITcVnAccess_UINT	获取值
Set [▶ 288]	ITcVnAccess_UINT	设置值
GetAt [▶ 346]	ITcVnRandomAccess_UINT	获取值
SetAt [▶ 346]	ITcVnRandomAccess_UINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.33.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    nValue  : Reference To UINT;
END_VAR
```

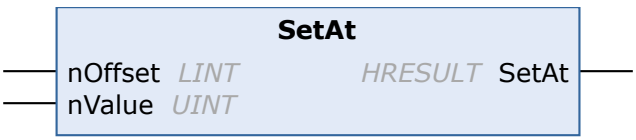
输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
nValue	引用 UINT	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.33.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    nValue  : UINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
nValue	UINT	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.34 ITcVnRandomAccess_ULINT

Offers a randowm access interface for ULINT values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_ULINT [▶ 288]
    ITcVnRandomAccess_ULINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 289]	ITcVnAccess_ULINT	获取值
Set [▶ 289]	ITcVnAccess_ULINT	设置值
GetAt [▶ 347]	ITcVnRandomAccess_ULINT	获取值
SetAt [▶ 348]	ITcVnRandomAccess_ULINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.34.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    nValue  : Reference To ULINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
nValue	引用 ULINT	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.34.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    nValue  : ULINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
nValue	ULINT	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.35 ITcVnRandomAccess_USINT

Offers a random access interface for USINT values

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_USINT [▶ 290]
    ITcVnRandomAccess_USINT

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 290]	ITcVnAccess_USINT	获取值
Set [▶ 290]	ITcVnAccess_USINT	设置值
GetAt [▶ 349]	ITcVnRandomAccess_USINT	获取值
SetAt [▶ 349]	ITcVnRandomAccess_USINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.35.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    nValue  : Reference To USINT;
END_VAR
```

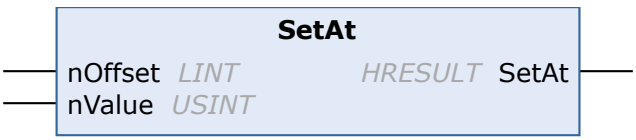
输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
nValue	引用 USINT	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.35.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    nValue  : USINT;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
nValue	USINT	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.36 ITcVnRandomAccess_TcVnVector4_REAL

Offers a random access interface for TcVnVector4_REAL values.

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnVector4_REAL [▶ 291]
    ITcVnRandomAccess_TcVnVector4_REAL

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 291]	ITcVnAccess_TcVnVector4_REAL	获取值
Set [▶ 292]	ITcVnAccess_TcVnVector4_REAL	设置值
GetAt [▶ 350]	ITcVnRandomAccess_TcVnVector4_REAL	获取值
SetAt [▶ 351]	ITcVnRandomAccess_TcVnVector4_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.36.1 *GetAt*



获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector4_REAL;
END_VAR
```

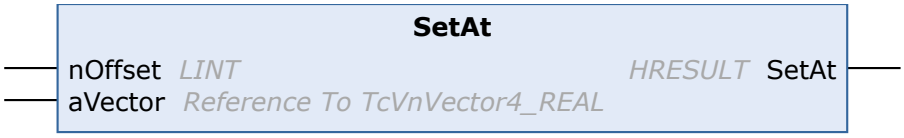
输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector4_REAL [▶ 153]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.36.2 *SetAt*



设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset : LINT;
    aVector : Reference To TcVnVector4_REAL;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
aVector	引用 TcVnVector4_REAL [▶ 153]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.3.37 ITcVnRandomAccess_TcVnRotatedRectangle

Offers a random access interface for TcVnRotatedRectangle values.

Inheritance Hierarchy

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnRotatedRectangle [▶ 292]
    ITcVnRandomAccess_TcVnRotatedRectangle

Methods

Name	Origin	Description
TcAddRef [▶ 424]	ITcUnknown	Increments the reference counter.
TcQueryInterface [▶ 425]	ITcUnknown	Get a reference to an implemented interface.
TcRelease [▶ 426]	ITcUnknown	Decrements the reference counter.
Get [▶ 293]	ITcVnAccess_TcVnRotatedRectangle	Gets the value
Set [▶ 293]	ITcVnAccess_TcVnRotatedRectangle	Sets the value
GetAt [▶ 352]	ITcVnRandomAccess_TcVnRotatedRectangle	Gets the value
SetAt [▶ 352]	ITcVnRandomAccess_TcVnRotatedRectangle	Sets the value

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.37.1 *GetAt*



Gets the value

Syntax

Definition:

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset     : LINT;
    stRectangle : Reference To TcVnRotatedRectangle;
END_VAR
```

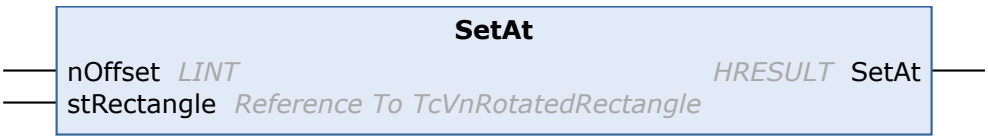
Inputs

Name	Type	Description
nOffset	LINT	Offset to the current position
stRectangle	Reference To TcVnRotatedRectangle [▶ 237]	Returns the value

Return value

HRESULT [▶ 135]

6.1.3.1.3.37.2 *SetAt*



Sets the value

Syntax

Definition:

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset     : LINT;
    stRectangle : Reference To TcVnRotatedRectangle;
END_VAR
```

Inputs

Name	Type	Description
nOffset	LINT	Offset to the current position
stRectangle	Reference To TcVnRotatedRectangle [▶ 237]	The value to set

Return value

HRESULT [▶ 135]

6.1.3.1.3.38 ITcVnRandomAccess_TcVnRotatedRectangle

提供 TcVnRotatedRectangle 值的随机访问接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnAccess_TcVnRotatedRectangle [▶ 294]
    ITcVnRandomAccess_TcVnRotatedRectangle

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Get [▶ 294]	ITcVnAccess_TcVnRotatedRectangle	获取值
Set [▶ 294]	ITcVnAccess_TcVnRotatedRectangle	设置值
GetAt [▶ 353]	ITcVnRandomAccess_TcVnRotatedRectangle	获取值
SetAt [▶ 354]	ITcVnRandomAccess_TcVnRotatedRectangle	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.3.38.1 *GetAt*

获取值

语法

定义：

```
METHOD GetAt : HRESULT
VAR_INPUT
    nOffset     : LINT;
    stRectangle : Reference To TcVnRotatedRectangle;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
stRectangle	引用 TcVnRotatedRectangle [▶ 237]	返回值

返回值

HRESULT [▶ 135]

6.1.3.1.3.38.2 *SetAt*

设置值

语法

定义：

```
METHOD SetAt : HRESULT
VAR_INPUT
    nOffset     : LINT;
    stRectangle : Reference To TcVnRotatedRectangle;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前位置的偏移量
stRectangle	引用 TcVnRotatedRectangle [▶ 237]	要设置的值

返回值

HRESULT [▶ 135]

6.1.3.1.4 迭代器

该组包含迭代器处理的接口，以便访问容器 [▶ 146]的元素。

6.1.3.1.4.1 ITcVnBidirectionalIterator

Offers an interface for a bidirectional iterator.

继承层次

ITcUnknown [▶ 424]
  ITcVnIteratorBase [▶ 358]
    ITcVnForwardIterator [▶ 355]
      ITcVnBidirectionalIterator

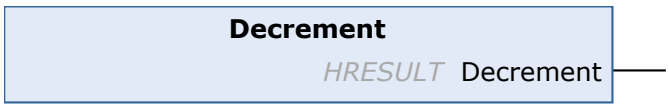
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
CheckIfBegin [▶ 359]	ITcVnIteratorBase	检查迭代器是否指向第一个元素。
CheckIfEnd [▶ 359]	ITcVnIteratorBase	检查迭代器是否指向最后一个元素的后一个位置（或者使用 F_VN_CheckIfIteratorIsAtEnd）。
GetValueSize [▶ 359]	ITcVnIteratorBase	获取迭代器指向的值所需的内存大小（以字节为单位）。
GetValueTypeGuid [▶ 360]	ITcVnIteratorBase	获取迭代器指向的值的 GUID 类型。
SetToBegin [▶ 360]	ITcVnIteratorBase	将迭代器设置为第一个元素（或者使用 F_VN_SetIteratorToBegin）。
SetToEnd [▶ 360]	ITcVnIteratorBase	将迭代器设置为最后一个元素的后一个位置。
CheckIfEqualTo [▶ 356]	ITcVnForwardIterator	检查迭代器是否等于另一个迭代器。
GetContainer [▶ 357]	ITcVnForwardIterator	获取转换为 ITcVnContainer 接口的当前元素的指针，并递增其引用计数器（仅适用于容器类型）。（或者使用 F_VN_GetContainer。）
Increment [▶ 357]	ITcVnForwardIterator	递增迭代器。（或者使用 F_VN_IncrementIterator。）
SetContainer [▶ 358]	ITcVnForwardIterator	使用 ITcVnContainer 接口设置当前元素（仅适用于容器类型）。（或者使用 F_VN_SetContainer。）
Decrement [▶ 355]	ITcVnBidirectionalIterator	递减迭代器。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.4.1.1 *递减*



递减迭代器。

语法

定义：

```
METHOD Decrement : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.1.4.2 ITcVnForwardIterator

Offers an interface for a forward iterator.

继承层次

ITcUnknown [▶ 424]
  ITcVnIteratorBase [▶ 358]
    ITcVnForwardIterator

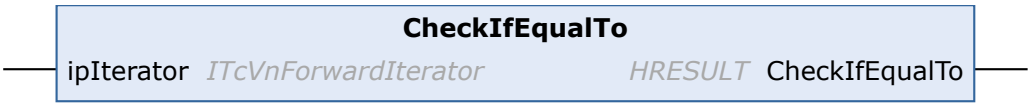
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
CheckIfBegin [▶ 359]	ITcVnIteratorBase	检查迭代器是否指向第一个元素。
CheckIfEnd [▶ 359]	ITcVnIteratorBase	检查迭代器是否指向最后一个元素的后一个位置（或者使用 F_VN_CheckIfIteratorIsAtEnd）。
GetValueSize [▶ 359]	ITcVnIteratorBase	获取迭代器指向的值所需的内存大小（以字节为单位）。
GetValueTypeGuid [▶ 360]	ITcVnIteratorBase	获取迭代器指向的值的 GUID 类型。
SetToBegin [▶ 360]	ITcVnIteratorBase	将迭代器设置为第一个元素（或者使用 F_VN_SetIteratorToBegin）。
SetToEnd [▶ 360]	ITcVnIteratorBase	将迭代器设置为最后一个元素的后一个位置。
CheckIfEqualTo [▶ 356]	ITcVnForwardIterator	检查迭代器是否等于另一个迭代器。
GetContainer [▶ 357]	ITcVnForwardIterator	获取转换为 ITcVnContainer 接口的当前元素的指针，并递增其引用计数器（仅适用于容器类型）。（或者使用 F_VN_GetContainer。）
Increment [▶ 357]	ITcVnForwardIterator	递增迭代器。（或者使用 F_VN_IncrementIterator。）
SetContainer [▶ 358]	ITcVnForwardIterator	使用 ITcVnContainer 接口设置当前元素（仅适用于容器类型）。（或者使用 F_VN_SetContainer。）

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.4.2.1 *CheckIfEqualTo*



检查迭代器是否等于另一个迭代器。

语法

定义：

```
METHOD CheckIfEqualTo : HRESULT
VAR_INPUT
    ipIterator : ITcVnForwardIterator;
END_VAR
```

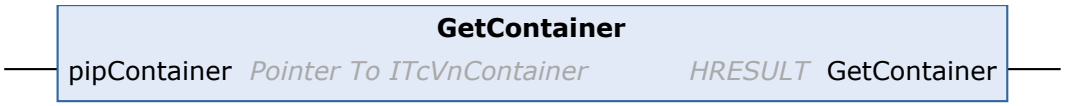
输入

名称	类型	描述
ipIterator	ITcVnForwardIterator [▶ 355]	进行比较的迭代器接口

返回值

HRESULT [▶ 135]

6.1.3.1.4.2.2 *GetContainer*



获取转换为 ITcVnContainer 接口的当前元素的指针，并递增其引用计数器（仅适用于容器类型）。（或者使用 F_VN_GetContainer。）

语法

定义：

```
METHOD GetContainer : HRESULT
VAR_INPUT
    pipContainer : Pointer To ITcVnContainer;
END_VAR
```

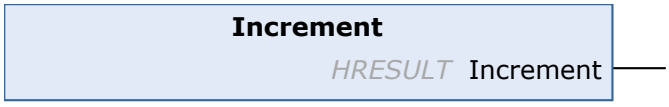
输入

名称	类型	描述
pipContainer	指针指向 ITcVnContainer [▶ 365]	返回容器接口。

返回值

HRESULT [▶ 135]

6.1.3.1.4.2.3 *增量*



递增迭代器。（或者使用 F_VN_IncrementIterator。）

语法

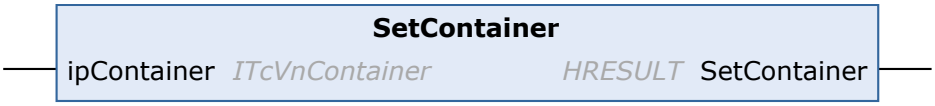
定义：

```
METHOD Increment : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.1.4.2.4 SetContainer



使用 ITcVnContainer 接口设置当前元素（仅适用于容器类型）。（或者使用 F_VN_SetContainer。）

语法

定义：

```
METHOD SetContainer : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将其内容分配给当前元素的容器接口。

返回值

HRESULT [▶ 135]

6.1.3.1.4.3 ITcVnIteratorBase

Offers a base interface for iterators

继承层次

ITcUnknown [▶ 424]
  ITcVnIteratorBase

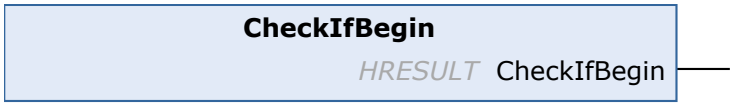
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
CheckIfBegin [▶ 359]	ITcVnIteratorBase	检查迭代器是否指向第一个元素。
CheckIfEnd [▶ 359]	ITcVnIteratorBase	检查迭代器是否指向最后一个元素的后一个位置（或者使用 F_VN_CheckIfIteratorIsAtEnd）。
GetValueSize [▶ 359]	ITcVnIteratorBase	获取迭代器指向的值所需的内存大小（以字节为单位）。
GetValueTypeGuid [▶ 360]	ITcVnIteratorBase	获取迭代器指向的值的 GUID 类型。
SetToBegin [▶ 360]	ITcVnIteratorBase	将迭代器设置为第一个元素（或者使用 F_VN_SetIteratorToBegin）。
SetToEnd [▶ 360]	ITcVnIteratorBase	将迭代器设置为最后一个元素的后一个位置。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.4.3.1 *CheckIfBegin*



检查迭代器是否指向第一个元素。

语法

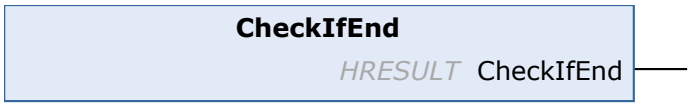
定义：

```
METHOD CheckIfBegin : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.1.4.3.2 *CheckIfEnd*



检查迭代器是否指向最后一个元素的后一个位置（或者使用 F_VN_CheckIfIteratorIsAtEnd）。

语法

定义：

```
METHOD CheckIfEnd : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.1.4.3.3 *GetValueSize*



获取迭代器指向的值所需的内存大小（以字节为单位）。

语法

定义：

```
METHOD GetValueSize : HRESULT
VAR_INPUT
    nSize : Reference To ULINT;
END_VAR
```

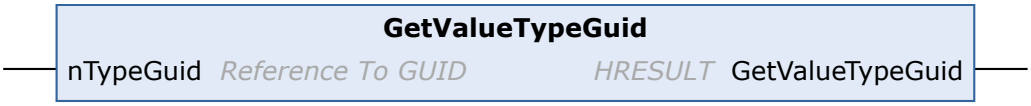
输入

名称	类型	描述
nSize	引用 ULINT	返回以字节为单位的元素大小。

返回值

HRESULT [▶ 135]

6.1.3.1.4.3.4 *GetValueTypeGuid*



获取迭代器指向的值的 GUID 类型。

语法

定义：

```
METHOD GetValueTypeGuid : HRESULT
VAR_INPUT
    nTypeGuid : Reference To GUID;
END_VAR
```

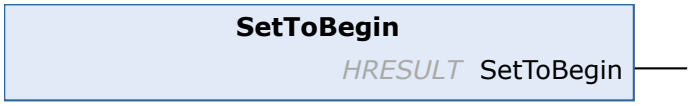
输入

名称	类型	描述
nTypeGuid	引用 GUID	返回 GUID 类型。

返回值

HRESULT [▶ 135]

6.1.3.1.4.3.5 *SetToBegin*



将迭代器设置为第一个元素（或者使用 F_VN_SetIteratorToBegin）。

语法

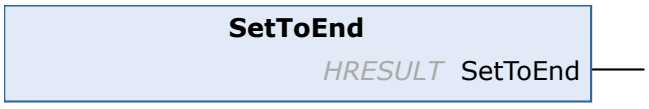
定义：

```
METHOD SetToBegin : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.1.4.3.6 *SetToEnd*



将迭代器设置为最后一个元素的后一个位置。

语法

定义：

```
METHOD SetToEnd : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.1.4.4 ITcVnIteratorCopyCreator

Offers an interface providing a method for creating a new iterator to the same position.

继承层次

ITcUnknown [▶ 424]
  ITcVnIteratorCopyCreator

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
Create [▶ 361]	ITcVnIteratorCopyCreator	创建一个新迭代器，指向调用迭代器的位置。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.4.4.1 *创建*



创建一个新迭代器，指向调用迭代器的位置。

语法

定义：

```
METHOD Create : HRESULT
VAR_INPUT
    pipIterator : Pointer To ITcVnForwardIterator;
END_VAR
```

输入

名称	类型	描述
pipIterator	指针指向 ITcVnForwardIterator [▶ 355]	返回创建的迭代器。

返回值

HRESULT [▶ 135]

6.1.3.1.4.5 ITcVnRandomAccessIterator

Offers an interface for a random access iterator.

继承层次

ITcUnknown [▶ 424]
  ITcVnIteratorBase [▶ 358]
    ITcVnForwardIterator [▶ 355]
      ITcVnBidirectionalIterator [▶ 354]
        ITcVnRandomAccessIterator

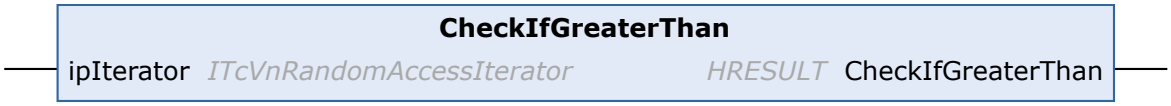
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
CheckIfBegin [▶ 359]	ITcVnIteratorBase	检查迭代器是否指向第一个元素。
CheckIfEnd [▶ 359]	ITcVnIteratorBase	检查迭代器是否指向最后一个元素的后一个位置（或者使用 F_VN_CheckIfIteratorIsAtEnd）。
GetValueSize [▶ 359]	ITcVnIteratorBase	获取迭代器指向的值所需的内存大小（以字节为单位）。
GetValueTypeGuid [▶ 360]	ITcVnIteratorBase	获取迭代器指向的值的 GUID 类型。
SetToBegin [▶ 360]	ITcVnIteratorBase	将迭代器设置为第一个元素（或者使用 F_VN_SetIteratorToBegin）。
SetToEnd [▶ 360]	ITcVnIteratorBase	将迭代器设置为最后一个元素的后一个位置。
CheckIfEqualTo [▶ 356]	ITcVnForwardIterator	检查迭代器是否等于另一个迭代器。
GetContainer [▶ 357]	ITcVnForwardIterator	获取转换为 ITcVnContainer 接口的当前元素的指针，并递增其引用计数器（仅适用于容器类型）。（或者使用 F_VN_GetContainer。）
Increment [▶ 357]	ITcVnForwardIterator	递增迭代器。（或者使用 F_VN_IncrementIterator。）
SetContainer [▶ 358]	ITcVnForwardIterator	使用 ITcVnContainer 接口设置当前元素（仅适用于容器类型）。（或者使用 F_VN_SetContainer。）
Decrement [▶ 355]	ITcVnBidirectionalIterator	递减迭代器。
CheckIfGreaterThan [▶ 363]	ITcVnRandomAccessIterator	检查该迭代器是否大于另一个迭代器。
CheckIfLessThan [▶ 363]	ITcVnRandomAccessIterator	检查该迭代器是否小于另一个迭代器。
GetContainerAt [▶ 364]	ITcVnRandomAccessIterator	获取一个转换为 ITcVnContainer 接口的指针，该指针指向在当前元素特定偏移位置的元素，并递增其引用计数器（仅适用于容器类型）。（或者使用 F_VN_GetAt_ITcVnContainer。）
SetContainerAt [▶ 364]	ITcVnRandomAccessIterator	使用 ITcVnContainer 接口（仅适用于容器类型）在当前元素的特定偏移位置设置元素。（或者使用 F_VN_SetAt_ITcVnContainer。）

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.4.5.1 *CheckIfGreaterThan*



检查该迭代器是否大于另一个迭代器。

语法

定义：

```
METHOD CheckIfGreaterThan : HRESULT
VAR_INPUT
    ipIterator : ITcVnRandomAccessIterator;
END_VAR
```

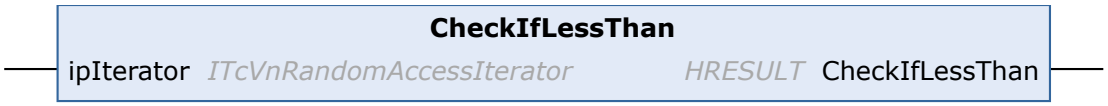
输入

名称	类型	描述
ipIterator	ITcVnRandomAccessIterator [▶ 361]	进行比较的迭代器接口。

返回值

HRESULT [▶ 135]

6.1.3.1.4.5.2 *CheckIfLessThan*



检查该迭代器是否小于另一个迭代器。

语法

定义：

```
METHOD CheckIfLessThan : HRESULT
VAR_INPUT
    ipIterator : ITcVnRandomAccessIterator;
END_VAR
```

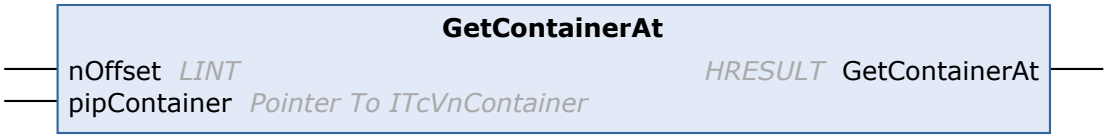
输入

名称	类型	描述
ipIterator	ITcVnRandomAccessIterator [▶ 361]	进行比较的迭代器接口。

返回值

HRESULT [▶ 135]

6.1.3.1.4.5.3 GetContainerAt



获取一个转换为 ITcVnContainer 接口的指针，该指针指向在当前元素特定偏移位置的元素，并递增其引用计数器（仅适用于容器类型）。（或者使用 F_VN_GetAt_ITcVnContainer。）

语法

定义：

```
METHOD GetContainerAt : HRESULT
VAR_INPUT
    nOffset      : LINT;
    pipContainer : Pointer To ITcVnContainer;
END_VAR
```

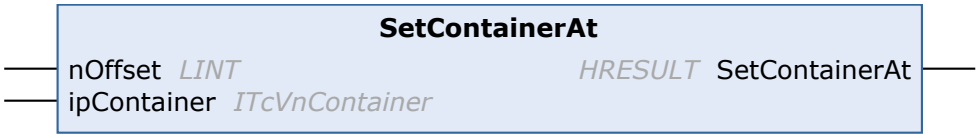
输入

名称	类型	描述
nOffset	LINT	与当前元素的偏移量。
pipContainer	指针指向 ITcVnContainer [▶ 365]	返回指向容器接口的指针。

返回值

HRESULT [▶ 135]

6.1.3.1.4.5.4 SetContainerAt



使用 ITcVnContainer 接口（仅适用于容器类型）在当前元素的特定偏移位置设置元素。（或者使用 F_VN_SetAt_ITcVnContainer。）

语法

定义：

```
METHOD SetContainerAt : HRESULT
VAR_INPUT
    nOffset     : LINT;
    ipContainer : ITcVnContainer;
END_VAR
```

输入

名称	类型	描述
nOffset	LINT	与当前元素的偏移量。
ipContainer	ITcVnContainer [▶ 365]	将其内容分配给当前元素的容器接口。

返回值

HRESULT [▶ 135]

6.1.3.1.5 ITcVnContainer

提供用于对象容器的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnContainer

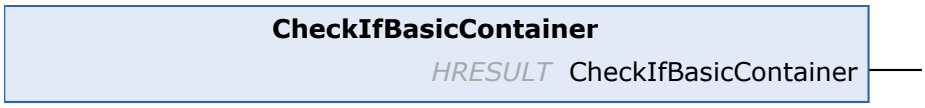
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
CheckIfBasicContainer [▶ 365]	ITcVnContainer	检查容器是否只包含基本（非容器）元素。
CheckIfEmpty [▶ 366]	ITcVnContainer	检查容器是否为空。（或者使用 F_VN_CheckIfEmpty。）
GetBidirectionalIterator [▶ 366]	ITcVnContainer	获取双向迭代器的接口指针（如果容器支持这种迭代器类型），并递增其引用计数器。
GetElementNum [▶ 366]	ITcVnContainer	获取容器的大小（元素数量）。（或者使用 F_VN_GetNumberOfElements。）
GetElementSize [▶ 367]	ITcVnContainer	获取容器中每个元素的大小（以字节为单位）。
GetElementTypeGuid [▶ 367]	ITcVnContainer	获取容器元素的 GUID。
GetExportSize [▶ 368]	ITcVnContainer	获取容器中所有元素的合并大小（以字节为单位）。
GetForwardIterator [▶ 368]	ITcVnContainer	获取指向前向迭代器的接口指针（如果容器支持该迭代器类型），并递增其引用计数器。（或者使用 F_VN_GetForwardIterator。）
GetRandomAccessIterator [▶ 369]	ITcVnContainer	获取指向随机访问迭代器的接口指针（如果容器支持该迭代器类型），并递增其引用计数器。（或者使用 F_VN_GetRandomAccessIterator。）
GetTypeGuid [▶ 369]	ITcVnContainer	获取容器的 GUID。
GetTypeName [▶ 370]	ITcVnContainer	获取字符串形式的容器类型名称。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.1.5.1 CheckIfBasicContainer



检查容器是否只包含基本（非容器）元素。

语法

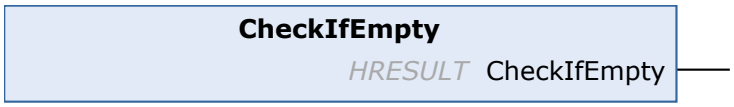
定义：

```
METHOD CheckIfBasicContainer : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.1.5.2 CheckIfEmpty



检查容器是否为空。（或者使用 F_VN_CheckIfEmpty。）

语法

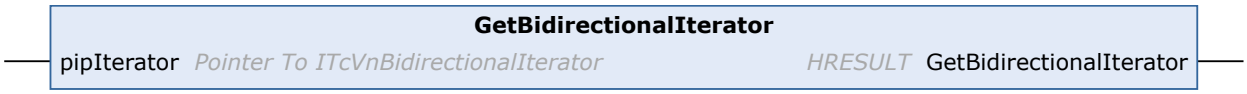
定义：

```
METHOD CheckIfEmpty : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.1.5.3 GetBidirectionalIterator



获取双向迭代器的接口指针（如果容器支持这种迭代器类型），并递增其引用计数器。

语法

定义：

```
METHOD GetBidirectionalIterator : HRESULT
VAR_INPUT
    pipIterator : Pointer To ITcVnBidirectionalIterator;
END_VAR
```

输入

名称	类型	描述
pipIterator	指针指向 ITcVnBidirectionalIterator [▶ 354]	返回迭代器接口。

返回值

HRESULT [▶ 135]

6.1.3.1.5.4 GetElementNum



获取容器的大小（元素数量）。（或者使用 F_VN_GetNumberOfElements。）

语法

定义：

```
METHOD GetElementNum : HRESULT
VAR_INPUT
    nElementNum : Reference To ULINT;
END_VAR
```

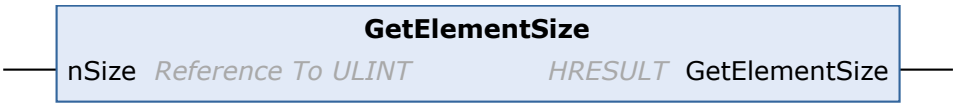
输入

名称	类型	描述
nElementNum	引用 ULINT	返回容器中元素的个数。

返回值

HRESULT [▶ 135]

6.1.3.1.5.5 GetElementSize



获取容器中每个元素的大小（以字节为单位）。

语法

定义：

```
METHOD GetElementSize : HRESULT
VAR_INPUT
    nSize : Reference To ULINT;
END_VAR
```

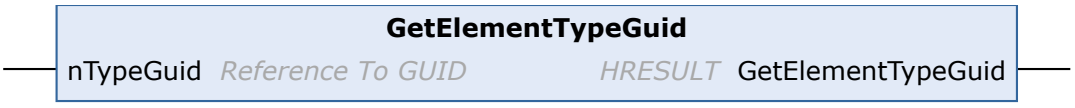
输入

名称	类型	描述
nSize	引用 ULINT	返回容器中单个元素的大小（以字节为单位）。

返回值

HRESULT [▶ 135]

6.1.3.1.5.6 GetElementTypeGuid



获取容器元素的 GUID。

语法

定义：

```
METHOD GetElementTypeGuid : HRESULT
VAR_INPUT
    nTypeGuid : Reference To GUID;
END_VAR
```

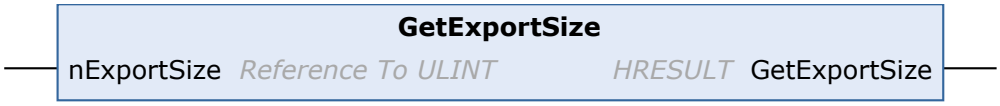
输入

名称	类型	描述
nTypeGuid	引用 GUID	返回容器元素的 GUID。

返回值

HRESULT [▶ 135]

6.1.3.1.5.7 GetExportSize



获取容器中所有元素的合并大小（以字节为单位）。

语法

定义：

```
METHOD GetExportSize : HRESULT
VAR_INPUT
    nExportSize : Reference To ULINT;
END_VAR
```

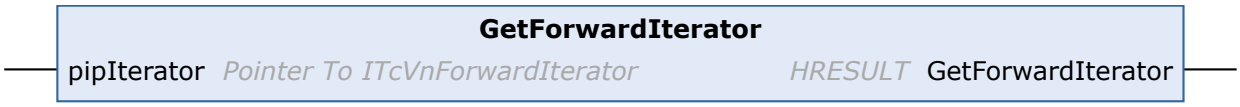
输入

名称	类型	描述
nExportSize	引用 ULINT	返回容器中所有元素的总大小（以字节为单位）。

返回值

HRESULT [▶ 135]

6.1.3.1.5.8 GetForwardIterator



获取指向前向迭代器的接口指针（如果容器支持该迭代器类型），并递增其引用计数器。（或者使用 F_VN_GetForwardIterator。）

语法

定义：

```
METHOD GetForwardIterator : HRESULT
VAR_INPUT
    pipIterator : Pointer To ITcVnForwardIterator;
END_VAR
```

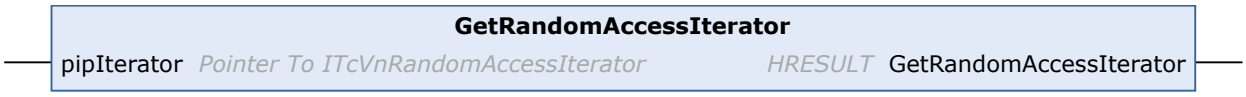
输入

名称	类型	描述
pipIterator	指针指向 ITcVnForwardIterator [▶ 355]	返回迭代器接口。

返回值

HRESULT [▶ 135]

6.1.3.1.5.9 GetRandomAccessIterator



获取指向随机访问迭代器的接口指针（如果容器支持该迭代器类型），并递增其引用计数器。（或者使用 F_VN_GetRandomAccessIterator。）

语法

定义：

```
METHOD GetRandomAccessIterator : HRESULT
VAR_INPUT
    pipIterator : Pointer To ITcVnRandomAccessIterator;
END_VAR
```

输入

名称	类型	描述
pipIterator	指针指向 ITcVnRandomAccessIterator [▶ 361]	返回迭代器接口。

返回值

HRESULT [▶ 135]

6.1.3.1.5.10 GetTypeGuid



获取容器的 GUID。

语法

定义：

```
METHOD GetTypeGuid : HRESULT
VAR_INPUT
    nTypeGuid : Reference To GUID;
END_VAR
```

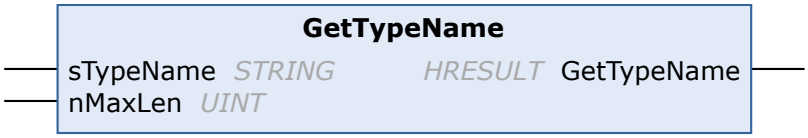
输入

名称	类型	描述
nTypeGuid	引用 GUID	返回容器的 GUID。

返回值

HRESULT [▶ 135]

6.1.3.1.5.11 GetTypeName



获取字符串形式的容器类型名称。

语法

定义：

```
METHOD GetTypeName : HRESULT
VAR_INPUT
    sTypeName : STRING;
    nMaxLen   : UINT;
END_VAR
```

输入

名称	类型	描述
sTypeName	STRING	以字符串形式返回容器类型名称。
nMaxLen	UINT	允许写入 sTypeName 的最大字符串长度

返回值

HRESULT [▶ 135]

6.1.3.2 图像

该组包含用于记录和处理图像 [▶ 143]的接口。

6.1.3.2.1 采集

该组包含用于处理图像采集的接口。

6.1.3.2.1.1 ITcIoFileImageAcquisition

文件图像采集接口。

继承层次

ITcUnknown [▶ 424]
  ITcIoImageAcquisition [▶ 375]
    ITcIoFileImageAcquisition

方法

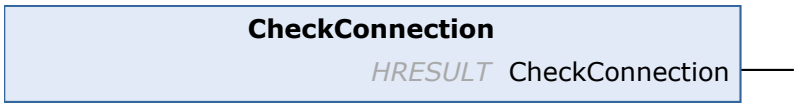
名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
RegisterReceiver [▶ 375]	ITcIoImageAcquisition	注册回调函数接口。
UnregisterReceiver [▶ 376]	ITcIoImageAcquisition	取消注册回调函数接口。

名称	起源	描述
OpenCamera [▶ 376]	ITcIoImageAcquisition	打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。
CloseCamera [▶ 376]	ITcIoImageAcquisition	关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。
StartAcquisition [▶ 377]	ITcIoImageAcquisition	开始图像采集。
StopAcquisition [▶ 377]	ITcIoImageAcquisition	停止图像采集。
SoftwareTrigger [▶ 377]	ITcIoImageAcquisition	初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。
InitializeCamera [▶ 378]	ITcIoImageAcquisition	向相机发送初始化命令。
CheckConnection [▶ 371]	ITcIoFileImageAcquisition	检查连接
TriggerImage [▶ 371]	ITcIoFileImageAcquisition	初始化软件触发器并触发单个图像。跳过 nSkipImages 并触发采集图像 nSkipImages+1。
TriggerImageByName [▶ 372]	ITcIoFileImageAcquisition	初始化软件触发器，并触发客户端助手中按名称指定的单个图像。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.1.1.1 *CheckConnection*



检查连接

语法

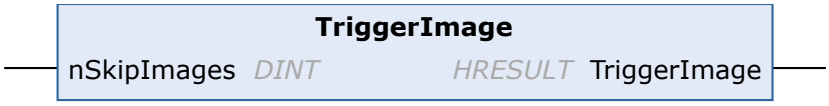
定义：

```
METHOD CheckConnection : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.1.1.2 *TriggerImage*



初始化软件触发器并触发单个图像。跳过 nSkipImages 并触发采集图像 nSkipImages+1。

语法

定义：

```
METHOD TriggerImage : HRESULT
VAR_INPUT
    nSkipImages : DINT;
END_VAR
```

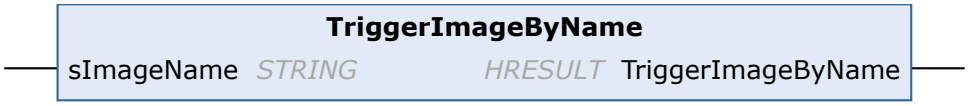
输入

名称	类型	描述
nSkipImages	DINT	要跳过的图像数量

返回值

HRESULT [▶ 135]

6.1.3.2.1.1.3 *TriggerImageByName*



初始化软件触发器，并触发客户端助手中按名称指定的单个图像。

语法

定义：

```
METHOD TriggerImageByName : HRESULT
VAR_INPUT
    sImageName : STRING;
END_VAR
```

输入

名称	类型	描述
sImageName	STRING	要触发的图像名称

返回值

HRESULT [▶ 135]

6.1.3.2.1.2 ITcIoFileImageRecv

图像接收器的接口。

继承层次

ITcUnknown [▶ 424]
  ITcIoImageRecv [▶ 378]
    ITcIoFileImageRecv

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
ReceiveImage [▶ 379]	ITcIoImageRecv	从 ITcIoGevImageAcquisition 实例接收图像。
ReceiveOpResult [▶ 379]	ITcIoImageRecv	从 ITcIoGevImageAcquisition 实例接收操作结果。
ReceiveImage [▶ 373]	ITcIoFileImageRecv	从 ITcIoFileImageAcquisition 的实例中接收图像。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.1.2.1 *ReceiveImage*



从 ITcIoFileImageAcquisition 的实例中接收图像。

语法

定义：

```
METHOD ReceiveImage : HRESULT
VAR_INPUT
    ipImage             : ITcVnImageBase;
    sFileName           : STRING;
    hrAcquisitionResult : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImageBase [▶ 410]	图像的接口指针。
sFileName	STRING	返回图像的文件名。
hrAcquisitionResult	HRESULT [▶ 135]	指示图像采集状态的 HRESULT。

返回值

HRESULT [▶ 135]

6.1.3.2.1.3 ITcIoGevImageAcquisition

GigE Vision 图像采集接口。

继承层次

ITcUnknown [▶ 424]
  ITcIoImageAcquisition [▶ 375]
    ITcIoGevImageAcquisition

方法

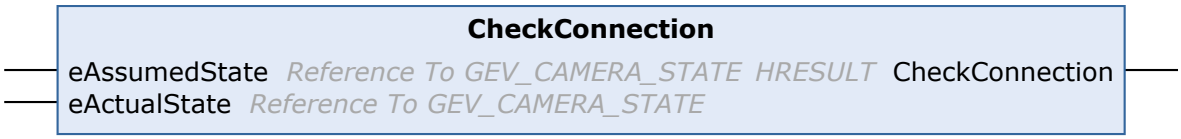
名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
RegisterReceiver [▶ 375]	ITcIoImageAcquisition	注册回调函数接口。

名称	起源	描述
UnregisterReceiver [▶ 376]	ITcIoImageAcquisition	取消注册回调函数接口。
OpenCamera [▶ 376]	ITcIoImageAcquisition	打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。
CloseCamera [▶ 376]	ITcIoImageAcquisition	关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。
StartAcquisition [▶ 377]	ITcIoImageAcquisition	开始图像采集。
StopAcquisition [▶ 377]	ITcIoImageAcquisition	停止图像采集。
SoftwareTrigger [▶ 377]	ITcIoImageAcquisition	初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。
InitializeCamera [▶ 378]	ITcIoImageAcquisition	向相机发送初始化命令。
CheckConnection [▶ 374]	ITcIoGevImageAcquisition	检查相机连接

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.1.3.1 *CheckConnection*



检查相机连接

语法

定义：

```
METHOD CheckConnection : HRESULT
VAR_INPUT
    eAssumedState : Reference To GEV_CAMERA_STATE;
    eActualState  : Reference To GEV_CAMERA_STATE;
END_VAR
```

输入

名称	类型	描述
eAssumedState	引用 GEV_CAMERA_STATE [▶ 217]	相机内部假定的状态。
eActualState	引用 GEV_CAMERA_STATE [▶ 217]	相机的实际可观测状态。

返回值

HRESULT [▶ 135]

6.1.3.2.1.4 ITcIoImageAcquisition

图像采集接口。

继承层次

ITcUnknown [▶ 424]
  ITcIoImageAcquisition

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
RegisterReceiver [▶ 375]	ITcIoImageAcquisition	注册回调函数接口。
UnregisterReceiver [▶ 376]	ITcIoImageAcquisition	取消注册回调函数接口。
OpenCamera [▶ 376]	ITcIoImageAcquisition	打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。
CloseCamera [▶ 376]	ITcIoImageAcquisition	关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。
StartAcquisition [▶ 377]	ITcIoImageAcquisition	开始图像采集。
StopAcquisition [▶ 377]	ITcIoImageAcquisition	停止图像采集。
SoftwareTrigger [▶ 377]	ITcIoImageAcquisition	初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。
InitializeCamera [▶ 378]	ITcIoImageAcquisition	向相机发送初始化命令。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.1.4.1 *RegisterReceiver*



注册回调函数接口。

语法

定义：

```
METHOD RegisterReceiver : HRESULT
VAR_INPUT
    ipRecv : ITcIoImageRecv;
END_VAR
```

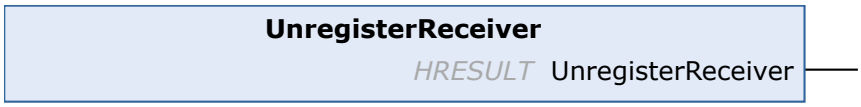
输入

名称	类型	描述
ipRecv	ITcIoImageRecv [▶ 378]	指向包含回调函数的接口的指针。

返回值

HRESULT [▶ 135]

6.1.3.2.1.4.2 *UnregisterReceiver*



取消注册回调函数接口。

语法

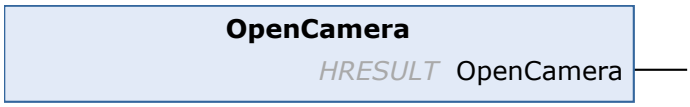
定义：

```
METHOD UnregisterReceiver : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.1.4.3 *OpenCamera*



打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。

语法

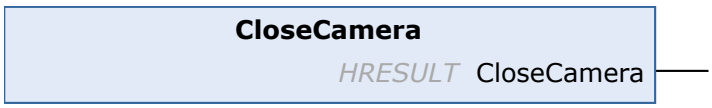
定义：

```
METHOD OpenCamera : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.1.4.4 *CloseCamera*



关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。

语法

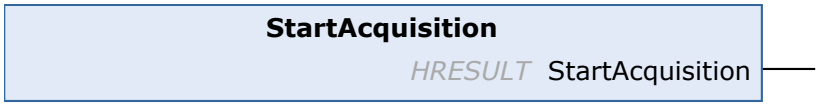
定义：

```
METHOD CloseCamera : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.1.4.5 *StartAcquisition*



开始图像采集。

语法

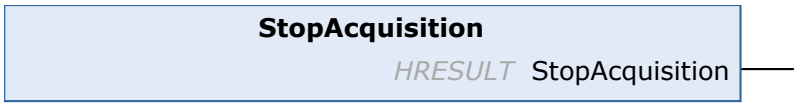
定义：

```
METHOD StartAcquisition : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.1.4.6 *StopAcquisition*



停止图像采集。

语法

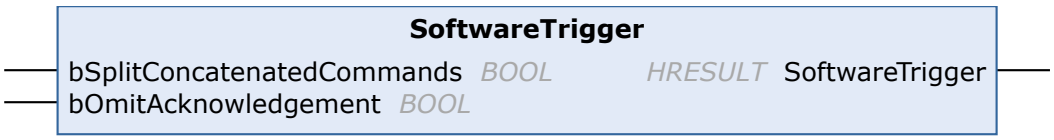
定义：

```
METHOD StopAcquisition : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.1.4.7 *SoftwareTrigger*



初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。

语法

定义：

```
METHOD SoftwareTrigger : HRESULT
VAR_INPUT
    bSplitConcatenatedCommands : BOOL;
    bOmitAcknowledgement       : BOOL;
END_VAR
```

输入

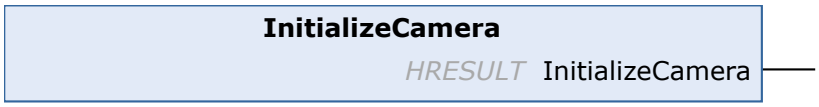
名称	类型	描述
bSplitConcatenatedCommands	BOOL	如果为 true，多读和多写命令将分别被拆分成单读和单写命令序列。

名称	类型	描述
bOmitAcknowledgement	BOOL	表示不请求确认数据包。

返回值

HRESULT [▶ 135]

6.1.3.2.1.4.8 *InitializeCamera*



向相机发送初始化命令。

语法

定义：

```
METHOD InitializeCamera : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.1.5 ITcIoImageRecv

图像接收器的接口。

继承层次

ITcUnknown [▶ 424]
  ITcIoImageRecv

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
ReceiveImage [▶ 379]	ITcIoImageRecv	从 ITcIoGevImageAcquisition 实例接收图像。
ReceiveOpResult [▶ 379]	ITcIoImageRecv	从 ITcIoGevImageAcquisition 实例接收操作结果。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.1.5.1 *ReceiveImage*



从 ITcIoGevImageAcquisition 实例接收图像。

语法

定义：

```
METHOD ReceiveImage : HRESULT
VAR_INPUT
    ipImage             : ITcVnImageBase;
    hrAcquisitionResult : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImageBase [▶ 410]	图像的接口指针。
hrAcquisitionResult	HRESULT [▶ 135]	指示图像采集状态的 HRESULT。

返回值

HRESULT [▶ 135]

6.1.3.2.1.5.2 *ReceiveOpResult*



从 ITcIoGevImageAcquisition 实例接收操作结果。

语法

定义：

```
METHOD ReceiveOpResult : HRESULT
VAR_INPUT
    hrOperationResult : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrOperationResult	HRESULT [▶ 135]	指示已执行操作状态的 HRESULT。

返回值

HRESULT [▶ 135]

6.1.3.2.1.6 ITcVnFileImageProvider

用于外部实时环境图像来源的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnImageProvider [▶ 383]
    ITcVnFileImageProvider

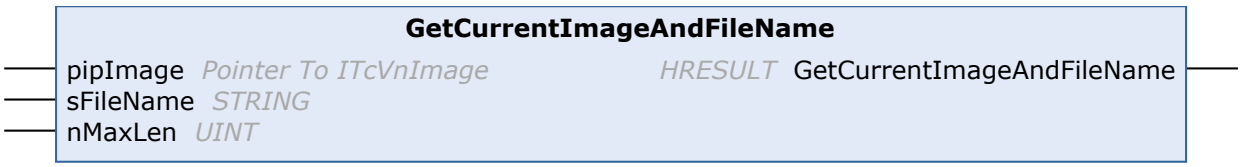
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
CheckIfBusy [▶ 384]	ITcVnImageProvider	检查模块是否繁忙，即是否正在进行操作。
CheckIfCameraConnected [▶ 385]	ITcVnImageProvider	检查相机是否正确连接（如果已连接，返回 S_OK；未连接则返回 S_FALSE）。只能检查软件连接，即如果返回 S_FALSE，检查无法确定电缆是否连接）。
GetCurrentImage [▶ 385]	ITcVnImageProvider	通过脱离内部引用获取当前图像。
GetError [▶ 385]	ITcVnImageProvider	获取最后一次执行的操作的结果。
OpenCamera [▶ 386]	ITcVnImageProvider	打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。
CloseCamera [▶ 386]	ITcVnImageProvider	关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。
StartAcquisition [▶ 386]	ITcVnImageProvider	开始图像采集。
StopAcquisition [▶ 387]	ITcVnImageProvider	停止图像采集。
SoftwareTrigger [▶ 387]	ITcVnImageProvider	初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。
InitializeCamera [▶ 387]	ITcVnImageProvider	将相机设置为初始状态。
GetCurrentImageAndFileName [▶ 380]	ITcVnFileImageProvider	通过脱离内部引用获取当前图像。
TriggerImage [▶ 381]	ITcVnFileImageProvider	初始化软件触发器并触发单个图像。跳过 nSkipImages 并触发采集图像 nSkipImages+1。
TriggerImageByName [▶ 381]	ITcVnFileImageProvider	初始化软件触发器，并触发客户端助手中按名称指定的单个图像。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.1.6.1 *GetCurrentImageAndFileName*



通过脱离内部引用获取当前图像。

语法

定义：

```
METHOD GetCurrentImageAndFileName : HRESULT
VAR_INPUT
    pipImage  : Pointer To ITcVnImage;
    sFileName : STRING;
    nMaxLen   : UINT;
END_VAR
```

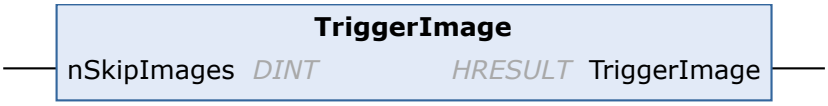
输入

名称	类型	描述
pipImage	指针指向 ITcVnImage [▶ 407]	应返回的接口指针的指针，如果图像采集失败，可能为 0
sFileName	STRING	以字符串形式返回文件名
nMaxLen	UINT	允许写入 sFileName 的最大字符串长度 、

返回值

HRESULT [▶ 135]

6.1.3.2.1.6.2 *TriggerImage*



初始化软件触发器并触发单个图像。跳过 nSkipImages 并触发采集图像 nSkipImages+1。

语法

定义：

```
METHOD TriggerImage : HRESULT
VAR_INPUT
    nSkipImages : DINT;
END_VAR
```

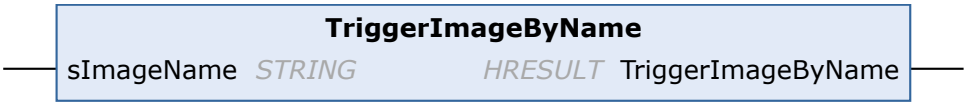
输入

名称	类型	描述
nSkipImages	DINT	要跳过的图像数量

返回值

HRESULT [▶ 135]

6.1.3.2.1.6.3 *TriggerImageByName*



初始化软件触发器，并触发客户端助手中按名称指定的单个图像。

语法

定义：

```
METHOD TriggerImageByName : HRESULT
VAR_INPUT
    sImageName : STRING;
END_VAR
```

输入

名称	类型	描述
sImageName	STRING	要触发的图像名称

返回值

HRESULT [▶ 135]

6.1.3.2.1.7 ITcVnGevImageProvider

GigE Vision 图像来源的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnImageProvider [▶ 383]
    ITcVnGevImageProvider

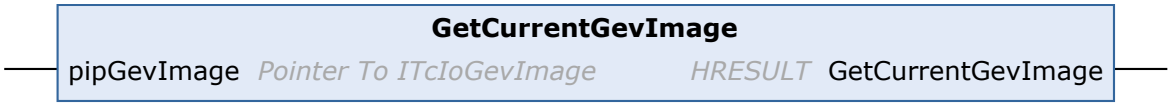
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
CheckIfBusy [▶ 384]	ITcVnImageProvider	检查模块是否繁忙，即是否正在进行操作。
CheckIfCameraConnected [▶ 385]	ITcVnImageProvider	检查相机是否正确连接（如果已连接，返回 S_OK；未连接则返回 S_FALSE）。只能检查软件连接，即如果返回 S_FALSE，检查无法确定电缆是否连接）。
GetCurrentImage [▶ 385]	ITcVnImageProvider	通过脱离内部引用获取当前图像。
GetError [▶ 385]	ITcVnImageProvider	获取最后一次执行的操作的结果。
OpenCamera [▶ 386]	ITcVnImageProvider	打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。
CloseCamera [▶ 386]	ITcVnImageProvider	关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。
StartAcquisition [▶ 386]	ITcVnImageProvider	开始图像采集。
StopAcquisition [▶ 387]	ITcVnImageProvider	停止图像采集。
SoftwareTrigger [▶ 387]	ITcVnImageProvider	初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。
InitializeCamera [▶ 387]	ITcVnImageProvider	将相机设置为初始状态。
GetCurrentGevImage [▶ 383]	ITcVnGevImageProvider	通过脱离 GigE Vision 图像的内部引用，获取当前 GigE Vision 图像。
GetCurrentImageWithGvspInfo [▶ 383]	ITcVnGevImageProvider	通过脱离内部引用获取当前图像，并提供 GVSP 信息。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.1.7.1 GetCurrentGevImage



通过脱离 GigE Vision 图像的内部引用，获取当前 GigE Vision 图像。

语法

定义：

```
METHOD GetCurrentGevImage : HRESULT
VAR_INPUT
    pipGevImage : Pointer To ITcIoGevImage;
END_VAR
```

输入

名称	类型	描述
pipGevImage	指针指向 ITcIoGevImage [▶ 403]	应返回的接口指针的指针，如果图像采集失败，可能为 0。

返回值

HRESULT [▶ 135]

6.1.3.2.1.7.2 GetCurrentImageWithGvspInfo



通过脱离内部引用获取当前图像，并提供 GVSP 信息。

语法

定义：

```
METHOD GetCurrentImageWithGvspInfo : HRESULT
VAR_INPUT
    pipImage   : Pointer To ITcVnImage;
    stGvspInfo : Reference To GVSP_IMAGE_INFO;
END_VAR
```

输入

名称	类型	描述
pipImage	指针指向 ITcVnImage [▶ 407]	应返回的接口指针的指针，如果图像采集失败，可能为 0。
stGvspInfo	引用 GVSP_IMAGE_INFO [▶ 220]	包含有用的元信息，如图像 ID、时间戳等。

返回值

HRESULT [▶ 135]

6.1.3.2.1.8 ITcVnImageProvider

图像来源的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnImageProvider

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
CheckIfBusy [▶ 384]	ITcVnImageProvider	检查模块是否繁忙，即是否正在进行操作。
CheckIfCameraConnected [▶ 385]	ITcVnImageProvider	检查相机是否正确连接（如果已连接，返回 S_OK；未连接则返回 S_FALSE）。只能检查软件连接，即如果返回 S_FALSE，检查无法确定电缆是否连接）。
GetCurrentImage [▶ 385]	ITcVnImageProvider	通过脱离内部引用获取当前图像。
GetError [▶ 385]	ITcVnImageProvider	获取最后一次执行的操作的结果。
OpenCamera [▶ 386]	ITcVnImageProvider	打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。
CloseCamera [▶ 386]	ITcVnImageProvider	关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。
StartAcquisition [▶ 386]	ITcVnImageProvider	开始图像采集。
StopAcquisition [▶ 387]	ITcVnImageProvider	停止图像采集。
SoftwareTrigger [▶ 387]	ITcVnImageProvider	初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。
InitializeCamera [▶ 387]	ITcVnImageProvider	将相机设置为初始状态。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.1.8.1 *CheckIfBusy*



检查模块是否繁忙，即是否正在进行操作。

语法

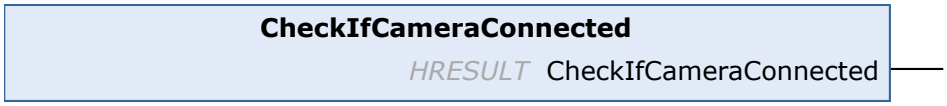
定义：

```
METHOD CheckIfBusy : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.1.8.2 *CheckIfCameraConnected*



检查相机是否正确连接（如果已连接，返回 S_OK；未连接则返回 S_FALSE）。只能检查软件连接，即如果返回 S_FALSE，检查无法确定电缆是否连接）。

语法

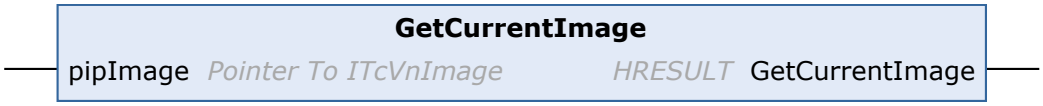
定义：

```
METHOD CheckIfCameraConnected : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.1.8.3 *GetCurrentImage*



通过脱离内部引用获取当前图像。

语法

定义：

```
METHOD GetCurrentImage : HRESULT
VAR_INPUT
    pipImage : Pointer To ITcVnImage;
END_VAR
```

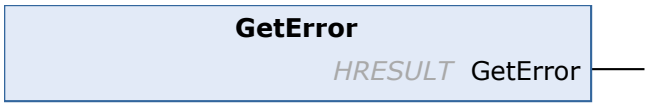
输入

名称	类型	描述
pipImage	指针指向 ITcVnImage [▶ 407]	应返回的接口指针的指针，如果图像采集失败，可能为 0。

返回值

HRESULT [▶ 135]

6.1.3.2.1.8.4 *GetError*



获取最后一次执行的操作的结果。

语法

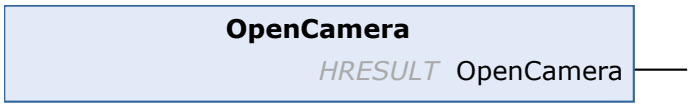
定义：

```
METHOD GetError : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.1.8.5 *OpenCamera*



打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。

语法

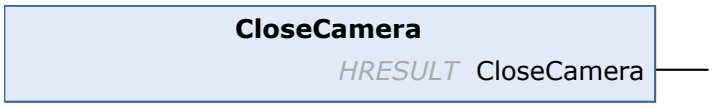
定义：

```
METHOD OpenCamera : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.1.8.6 *CloseCamera*



关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。

语法

定义：

```
METHOD CloseCamera : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.1.8.7 *StartAcquisition*



开始图像采集。

语法

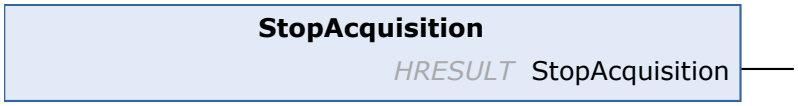
定义：

```
METHOD StartAcquisition : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.1.8.8 *StopAcquisition*



停止图像采集。

语法

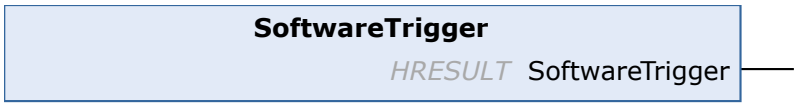
定义：

```
METHOD StopAcquisition : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.1.8.9 *SoftwareTrigger*



初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。

语法

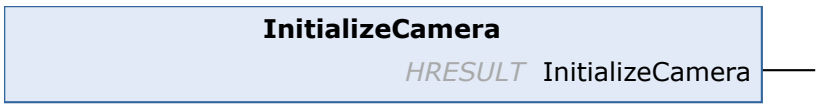
定义：

```
METHOD SoftwareTrigger : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.1.8.10 *InitializeCamera*



将相机设置为初始状态。

语法

定义：

```
METHOD InitializeCamera : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.2 导出

该组包含处理与图像有关的数据输出的接口。

6.1.3.2.2.1 ITcVnBitmapExport

将图像导出为 Windows 位图（BMP）的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnBitmapExport

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
GetBitmapSize [▶ 388]	ITcVnBitmapExport	如果图像转换为 Windows 位图（BMP），则获取图像的大小。
GetBitmapImage [▶ 389]	ITcVnBitmapExport	将图像以 Windows 位图（BMP）形式导出到给定的缓冲区中。

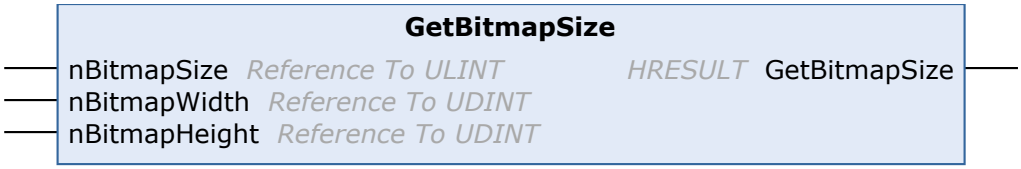
更多信息

根据定义，一个位图只能有 8 位（对于单通道图像）或 24 位（对于 3 通道和 4 通道图像）。因此，16 位图像以及元素类型为 REAL 或 LREAL 的图像被转换为位图输出。在 16 位图像的情况下，数值范围被简单地缩减为 8 位（0 对应于 0，而 255 对应于 65,280）。在 (L)REAL 图像的情况下，值范围 [-1, +1] 被缩放为 [-127, +127] 并作为有符号整数传输。在这些限制之外的值将分别被解释为 -1 或 +1。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.2.1.1 *GetBitmapSize*



如果图像转换为 Windows 位图（BMP），则获取图像的大小。

语法

定义：

```
METHOD GetBitmapSize : HRESULT
VAR_INPUT
    nBitmapSize   : Reference To ULINT;
    nBitmapWidth  : Reference To UDINT;
    nBitmapHeight : Reference To UDINT;
END_VAR
```

输入

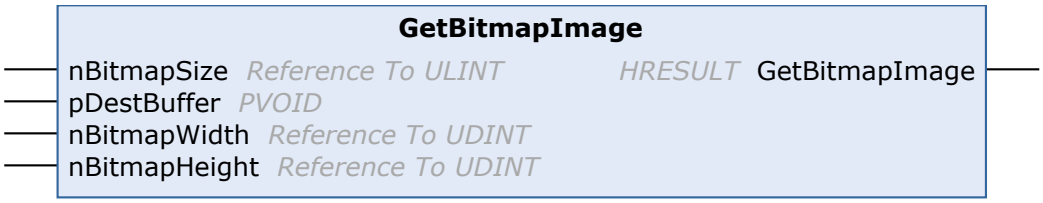
名称	类型	描述
nBitmapSize	引用 ULINT	输出参数，包含所需的缓冲区大小。

名称	类型	描述
nBitmapWidth	引用 UDINT	所需的宽度，或使用 0 来保持原始宽度（输入）和实际宽度（输出）。
nBitmapHeight	引用 UDINT	所需的高度或 0，以保持原始高度（输入）和实际高度（输出）。

返回值

HRESULT [▶ 135]

6.1.3.2.2.1.2 *GetBitmapImage*



将图像以 Windows 位图（BMP）形式导出到给定的缓冲区中。

语法

定义：

```
METHOD GetBitmapImage : HRESULT
VAR_INPUT
    nBitmapSize   : Reference To ULINT;
    pDestBuffer   : PVOID;
    nBitmapWidth  : Reference To UDINT;
    nBitmapHeight : Reference To UDINT;
END_VAR
```

输入

名称	类型	描述
nBitmapSize	引用 ULINT	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。
nBitmapWidth	引用 UDINT	所需的宽度，或使用 0 来保持原始宽度（输入）和实际宽度（输出）。
nBitmapHeight	引用 UDINT	所需的高度或 0，以保持原始高度（输入）和实际高度（输出）。

返回值

HRESULT [▶ 135]

6.1.3.2.2.2 ITcVnBitmapExportNotification

将可显示图像作为 ADS 通知发送所需的接口

继承层次

ITcUnknown [▶ 424]
  ITcVnBitmapExportNotification

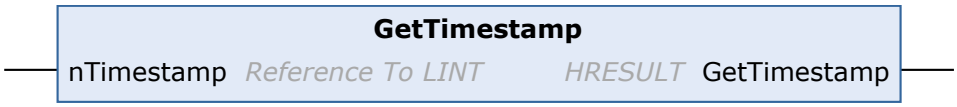
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
GetTimestamp [▶ 390]	ITcVnBitmapExportNotification	获取最新图像更改的时间戳。
GetBitmapImageRpcUnlocked [▶ 390]	ITcVnBitmapExportNotification	通过未锁定的远程过程调用，将图像以 Windows 位图（BMP）的形式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.2.2.1 *GetTimestamp*



获取最新图像更改的时间戳。

语法

定义：

```
METHOD GetTimestamp : HRESULT
VAR_INPUT
    nTimestamp : Reference To LINT;
END_VAR
```

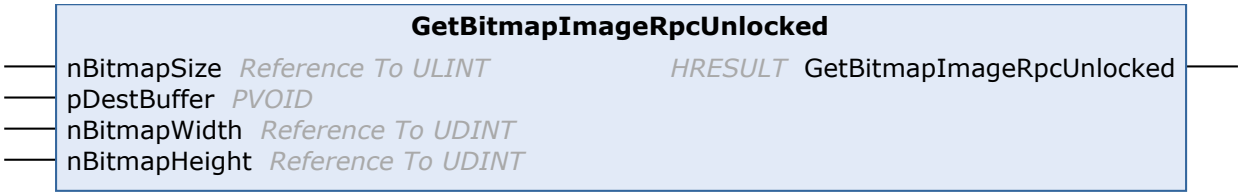
输入

名称	类型	描述
nTimestamp	引用 LINT	返回时间戳。

返回值

HRESULT [▶ 135]

6.1.3.2.2.2.2 *GetBitmapImageRpcUnlocked*



通过未锁定的远程过程调用，将图像以 Windows 位图（BMP）的形式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

语法

定义：

```
METHOD GetBitmapImageRpcUnlocked : HRESULT
VAR_INPUT
    nBitmapSize   : Reference To ULINT;
    pDestBuffer   : PVOID;
    nBitmapWidth  : Reference To UDINT;
    nBitmapHeight : Reference To UDINT;
END_VAR
```

输入

名称	类型	描述
nBitmapSize	引用 ULINT	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。
nBitmapWidth	引用 UDINT	所需的宽度，或使用 0 来保持原始宽度（输入）和实际宽度（输出）。
nBitmapHeight	引用 UDINT	所需的高度或 0，以保持原始高度（输入）和实际高度（输出）。

返回值

HRESULT [▶ 135]

6.1.3.2.2.3 ITcVnBitmapExportRpcUnlocked

通过未锁定的远程过程调用，将图像以 Windows 位图（BMP）形式导出到给定缓冲区的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnBitmapExport [▶ 388]
    ITcVnBitmapExportRpcUnlocked

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
GetBitmapSize [▶ 388]	ITcVnBitmapExport	如果图像转换为 Windows 位图（BMP），则获取图像的大小。
GetBitmapImage [▶ 389]	ITcVnBitmapExport	将图像以 Windows 位图（BMP）形式导出到给定的缓冲区中。
GetBitmapImageRpcUnlocked [▶ 392]	ITcVnBitmapExportRpcUnlocked	通过未锁定的远程过程调用，将图像以 Windows 位图（BMP）的形式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

更多信息

根据定义，一个位图只能有 8 位（对于单通道图像）或 24 位（对于 3 通道和 4 通道图像）。因此，16 位图像以及元素类型为 REAL 或 LREAL 的图像被转换为位图输出。在 16 位图像的情况下，数值范围被简单地缩减为 8 位（0 对应于 0，而 255 对应于 65,280）。在 (L)REAL 图像的情况下，值范围 [-1, +1] 被缩放为 [-127, +127] 并作为有符号整数传输。在这些限制之外的值将分别被解释为 -1 或 +1。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.2.3.1 *GetBitmapImageRpcUnlocked*



通过未锁定的远程过程调用，将图像以 Windows 位图（BMP）的形式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

语法

定义：

```
METHOD GetBitmapImageRpcUnlocked : HRESULT
VAR_INPUT
    nBitmapSize   : Reference To ULINT;
    pDestBuffer   : PVOID;
    nBitmapWidth  : Reference To UDINT;
    nBitmapHeight : Reference To UDINT;
END_VAR
```

输入

名称	类型	描述
nBitmapSize	引用 ULINT	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。
nBitmapWidth	引用 UDINT	所需的宽度，或使用 0 来保持原始宽度（输入）和实际宽度（输出）。
nBitmapHeight	引用 UDINT	所需的高度或 0，以保持原始高度（输入）和实际高度（输出）。

返回值

HRESULT [▶ 135]

6.1.3.2.2.4 ITcVnHistogramExport

以数组形式导出图像直方图的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnHistogramExport

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。

名称	起源	描述
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
GetHistogramSize [▶ 393]	ITcVnHistogramExport	获取直方图数组的大小。
GetHistogramArray [▶ 393]	ITcVnHistogramExport	获取直方图数组。
GetCustomHistogramSize [▶ 394]	ITcVnHistogramExport	获取自定义直方图数组的大小。
GetCustomHistogramArray [▶ 394]	ITcVnHistogramExport	获取自定义直方图数组。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.2.4.1 *GetHistogramSize*



获取直方图数组的大小。

语法

定义：

```
METHOD GetHistogramSize : HRESULT
VAR_INPUT
    nHistogramSize : Reference To ULINT;
END_VAR
```

输入

名称	类型	描述
nHistogramSize	引用 ULINT	输出参数，包含所需的缓冲区大小。

返回值

HRESULT [▶ 135]

6.1.3.2.2.4.2 *GetHistogramArray*



获取直方图数组。

语法

定义：

```
METHOD GetHistogramArray : HRESULT
VAR_INPUT
    nHistogramSize : Reference To ULINT;
    pDestBuffer    : PVOID;
END_VAR
```

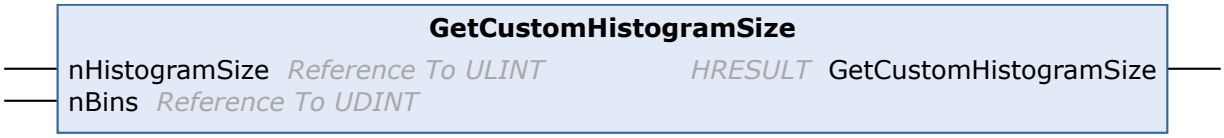
输入

名称	类型	描述
nHistogramSize	引用 ULINT	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。

返回值

HRESULT [▶ 135]

6.1.3.2.2.4.3 *GetCustomHistogramSize*



获取自定义直方图数组的大小。

语法

定义：

```
METHOD GetCustomHistogramSize : HRESULT
VAR_INPUT
    nHistogramSize : Reference To ULINT;
    nBins          : Reference To UDINT;
END_VAR
```

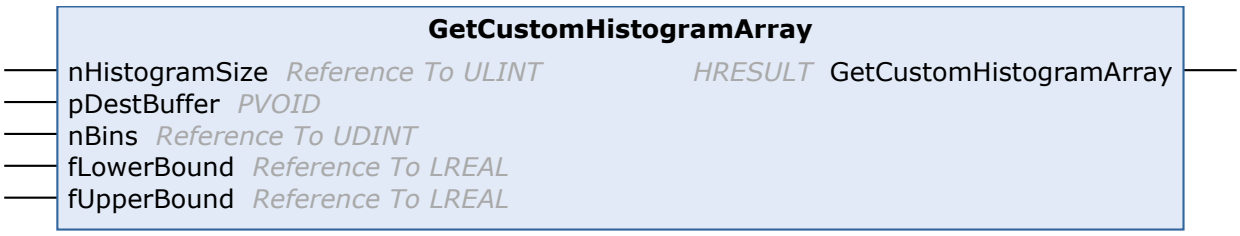
输入

名称	类型	描述
nHistogramSize	引用 ULINT	输出参数，包含所需的缓冲区大小。
nBins	引用 UDINT	所需的分组数量，或用 0 保持相应图像格式的默认值（输入）和默认分组数量（输出）。

返回值

HRESULT [▶ 135]

6.1.3.2.2.4.4 *GetCustomHistogramArray*



获取自定义直方图数组。

语法

定义：

```
METHOD GetCustomHistogramArray : HRESULT
VAR_INPUT
    nHistogramSize : Reference To ULINT;
    pDestBuffer    : PVOID;
    nBins          : Reference To UDINT;
    fLowerBound    : Reference To LREAL;
    fUpperBound    : Reference To LREAL;
END_VAR
```

输入

名称	类型	描述
nHistogramSize	引用 ULINT	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。
nBins	引用 UDINT	所需的分组数量，或用 0 保持相应图像格式的默认值（输入）和默认分组数量（输出）。
fLowerBound	引用 LREAL	第 0 个直方图分组的下边界（包含）（输入），如果 fLowerBound 与 fUpperBound 都设置为 0，则接收默认值（输出）。
fUpperBound	引用 LREAL	最后一个直方图分组 nBins-1 的上边界（不包含）（输入），如果 fLowerBound 与 fUpperBound 设置为 0，则接收默认值（输出）。

返回值

HRESULT [▶ 135]

6.1.3.2.2.5 ITcVnTiffExport

将图像导出为 tiff 格式的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnTiffExport

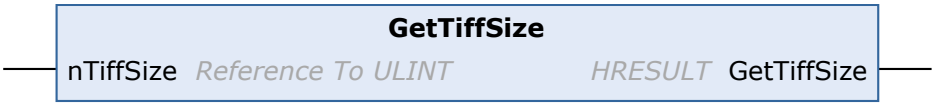
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
GetTiffSize [▶ 396]	ITcVnTiffExport	如果图像转换为 tiff 格式，则获取图像的大小。
GetTiffImage [▶ 396]	ITcVnTiffExport	将图像以 tiff 格式导出到给定的缓冲区中。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.2.5.1 *GetTiffSize*



如果图像转换为 tiff 格式，则获取图像的大小。

语法

定义：

```
METHOD GetTiffSize : HRESULT
VAR_INPUT
    nTiffSize : Reference To ULINT;
END_VAR
```

输入

名称	类型	描述
nTiffSize	引用 ULINT	输出参数，包含所需的缓冲区大小。

返回值

HRESULT [▶ 135]

6.1.3.2.2.5.2 *GetTiffImage*



将图像以 tiff 格式导出到给定的缓冲区中。

语法

定义：

```
METHOD GetTiffImage : HRESULT
VAR_INPUT
    nTiffSize   : Reference To ULINT;
    pDestBuffer : PVOID;
END_VAR
```

输入

名称	类型	描述
nTiffSize	引用 ULINT	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。

返回值

HRESULT [▶ 135]

6.1.3.2.2.6 ITcVnTiffExportNotification

用于将可显示图像以 tiff 格式通过 ADS 通知发送的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnTiffExportNotification

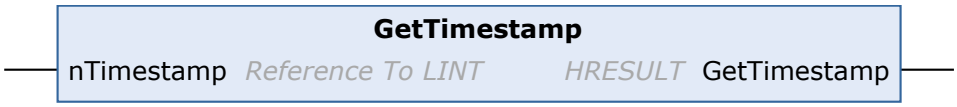
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
GetTimestamp [▶ 397]	ITcVnTiffExportNotification	获取最新图像更改的时间戳。
GetTiffImageRpcUnlocked [▶ 397]	ITcVnTiffExportNotification	通过未锁定的远程过程调用，将图像以 tiff 格式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.2.6.1 *GetTimestamp*



获取最新图像更改的时间戳。

语法

定义：

```
METHOD GetTimestamp : HRESULT
VAR_INPUT
    nTimestamp : Reference To LINT;
END_VAR
```

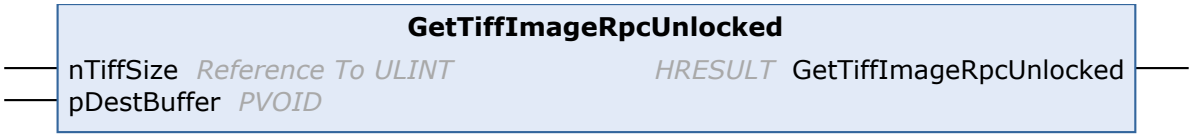
输入

名称	类型	描述
nTimestamp	引用 LINT	返回时间戳。

返回值

HRESULT [▶ 135]

6.1.3.2.2.6.2 *GetTiffImageRpcUnlocked*



通过未锁定的远程过程调用，将图像以 tiff 格式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

语法

定义：

```
METHOD GetTiffImageRpcUnlocked : HRESULT
VAR_INPUT
    nTiffSize   : Reference To ULINT;
    pDestBuffer : PVOID;
END_VAR
```

输入

名称	类型	描述
nTiffSize	引用 ULINT	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。

返回值

HRESULT [▶ 135]

6.1.3.2.2.7 ITcVnTiffExportRpcUnlocked

通过未锁定的远程过程调用，将图像导出为 tiff 格式的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnTiffExport [▶ 395]
    ITcVnTiffExportRpcUnlocked

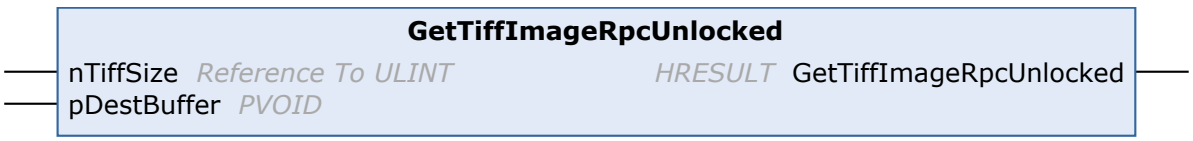
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
GetTiffSize [▶ 396]	ITcVnTiffExport	如果图像转换为 tiff 格式，则获取图像的大小。
GetTiffImage [▶ 396]	ITcVnTiffExport	将图像以 tiff 格式导出到给定的缓冲区中。
GetTiffImageRpcUnlocked [▶ 398]	ITcVnTiffExportRpcUnlocked	通过未锁定的远程过程调用，将图像以 tiff 格式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.2.7.1 *GetTiffImageRpcUnlocked*



通过未锁定的远程过程调用，将图像以 tiff 格式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

语法

定义：

```
METHOD GetTiffImageRpcUnlocked : HRESULT
VAR_INPUT
    nTiffSize   : Reference To ULINT;
    pDestBuffer : PVOID;
END_VAR
```

输入

名称	类型	描述
nTiffSize	引用 ULINT	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。

返回值

HRESULT [▶ 135]

6.1.3.2.2.8 ITcVnDataExport

Interface for exporting binary data.

Inheritance Hierarchy

ITcUnknown [▶ 424]
  ITcVnDataExport

Methods

Name	Origin	Description
TcAddRef [▶ 424]	ITcUnknown	Increments the reference counter.
TcQueryInterface [▶ 425]	ITcUnknown	Get a reference to an implemented interface.
TcRelease [▶ 426]	ITcUnknown	Decrements the reference counter.
GetSize [▶ 399]	ITcVnDataExport	Get the size of the data.
GetData [▶ 400]	ITcVnDataExport	Export the data into a given buffer.
GetDataIfNew [▶ 400]	ITcVnDataExport	Export the data into a given buffer, if the data is new (i.e. not yet exported).

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.2.8.1 *GetSize*



Get the size of the data.

Syntax

Definition:

```
METHOD GetSize : HRESULT
VAR_INPUT
    nSize : Reference To ULINT;
END_VAR
```

Inputs

Name	Type	Description
nSize	Reference To ULINT	Output parameter containing the data size.

6.1.3.2.2.8.2 *GetData*



Export the data into a given buffer.

Syntax

Definition:

```
METHOD GetData : HRESULT
VAR_INPUT
    nSize       : Reference To ULINT;
    pDestBuffer : PVOID;
END_VAR
```

Inputs

Name	Type	Description
nSize	Reference To ULINT	Provided buffer size (in) and actual data size (out).
pDestBuffer	PVOID	Pointer to the destination buffer.

Return value

HRESULT [▶ 135]

6.1.3.2.2.8.3 *GetDataIfNew*



Export the data into a given buffer, if the data is new (i.e. not yet exported).

Syntax

Definition:

```
METHOD GetDataIfNew : HRESULT
VAR_INPUT
    nSize       : Reference To ULINT;
    pDestBuffer : PVOID;
END_VAR
```

Inputs

Name	Type	Description
nSize	Reference To ULINT	Provided buffer size (in) and actual data size (out).
pDestBuffer	PVOID	Pointer to the destination buffer.

Return value

HRESULT [▶ 135]

6.1.3.2.2.9 ITcVnDataExportRpcUnlocked

Interface for exporting binary data by means of an unlocked remote procedure call.

Inheritance Hierarchy

ITcUnknown [▶ 424]
  ITcVnDataExport [▶ 399]
    ITcVnDataExportRpcUnlocked

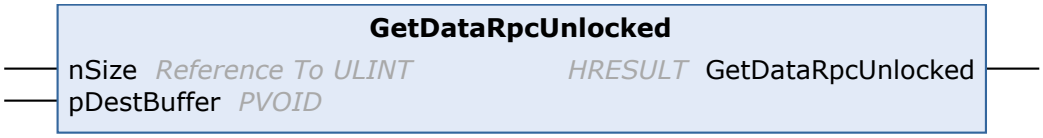
Methods

Name	Origin	Description
TcAddRef [▶ 424]	ITcUnknown	Increments the reference counter.
TcQueryInterface [▶ 425]	ITcUnknown	Get a reference to an implemented interface.
TcRelease [▶ 426]	ITcUnknown	Decrements the reference counter.
GetSize [▶ 399]	ITcVnDataExport	Get the size of the data.
GetData [▶ 400]	ITcVnDataExport	Export the data into a given buffer.
GetDataIfNew [▶ 400]	ITcVnDataExport	Export the data into a given buffer, if the data is new (i.e. not yet exported).
GetDataRpcUnlocked [▶ 401]	ITcVnDataExportRpcUnlocked	Export the data into a given buffer by means of an unlocked remote procedure call. It lies within the responsibility of the user to ensure that no conflicting accesses can occur.
GetDataIfNewRpcUnlocked [▶ 402]	ITcVnDataExportRpcUnlocked	Export the data into a given buffer by means of an unlocked remote procedure call, if the data is new (i.e. not yet exported). It lies within the responsibility of the user to ensure that no conflicting accesses can occur.

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.2.9.1 *GetDataRpcUnlocked*



Export the data into a given buffer by means of an unlocked remote procedure call. It lies within the responsibility of the user to ensure that no conflicting accesses can occur.

Syntax

Definition:

```
METHOD GetDataRpcUnlocked : HRESULT
VAR_INPUT
    nSize       : Reference To ULINT;
    pDestBuffer : PVOID;
END_VAR
```

Inputs

Name	Type	Description
nSize	Reference To ULINT	Provided buffer size (in) and actual data size (out).
pDestBuffer	PVOID	Pointer to the destination buffer.

Return value

HRESULT [▶ 135]

6.1.3.2.2.9.2 *GetDataIfNewRpcUnlocked*



Export the data into a given buffer by means of an unlocked remote procedure call, if the data is new (i.e. not yet exported). It lies within the responsibility of the user to ensure that no conflicting accesses can occur.

Syntax

Definition:

```
METHOD GetDataIfNewRpcUnlocked : HRESULT
VAR_INPUT
    nSize       : Reference To ULINT;
    pDestBuffer : PVOID;
END_VAR
```

Inputs

Name	Type	Description
nSize	Reference To ULINT	Provided buffer size (in) and actual data size (out).
pDestBuffer	PVOID	Pointer to the destination buffer.

Return value

HRESULT [▶ 135]

6.1.3.2.3 Import

该组包含用于处理与图像相关的数据导入的接口。

6.1.3.2.3.1 ITcVnTiffImport

导入 tiff 图像的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnTiffImport

方法

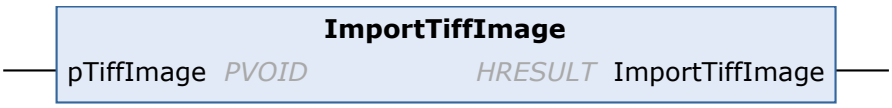
名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。

名称	起源	描述
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
ImportTiffImage [▶ 403]	ITcVnTiffImport	导入一张 tiff 图像。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.3.1.1 *ImportTiffImage*



导入一张 tiff 图像。

语法

定义：

```
METHOD ImportTiffImage : HRESULT
VAR_INPUT
    pTiffImage : PVOID;
END_VAR
```

输入

名称	类型	描述
pTiffImage	PVOID	指向 tiff 文件的指针。

返回值

HRESULT [▶ 135]

6.1.3.2.4 ITcIoGevImage

Interface for GigE Vision images.

继承层次

ITcUnknown [▶ 424]
  ITcVnImageBase [▶ 410]
    ITcIoGevImage

方法

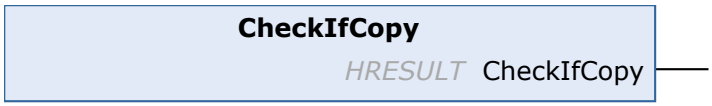
名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
GetImageData [▶ 410]	ITcVnImageBase	获取数据接口（不要在其中明确释放指针！）。
GetImageSize [▶ 411]	ITcVnImageBase	获取图像大小。
GetWidth [▶ 411]	ITcVnImageBase	获取图像宽度（或者使用 F_VN_GetImageWidth）。

名称	起源	描述
GetHeight [▶ 412]	ITcVnImageBase	获取图像高度（或者使用 F_VN_GetImageHeight）。
GetXPadding [▶ 412]	ITcVnImageBase	获取水平填充。
GetYPadding [▶ 413]	ITcVnImageBase	获取垂直填充。
GetPixelFormat [▶ 413]	ITcVnImageBase	获取像素格式（或者使用 F_VN_GetPixelFormat）。
GetImageInfo [▶ 414]	ITcVnImageBase	获取包含图像所有常见元信息的结构体。这基本上包含了可通过该接口访问的所有元信息。（或者使用 F_VN_GetImageInfo。）
CheckIfCopy [▶ 404]	ITcIoGevImage	检查图像数据是否为副本。
DecreaseHeight [▶ 404]	ITcIoGevImage	如果图像高度小于当前值，则将图像高度设置为 newHeight。
GetBlockId [▶ 405]	ITcIoGevImage	获取 GVSP 块 ID。
GetGevStatus [▶ 405]	ITcIoGevImage	获取块状态。
GetGvspImageInfo [▶ 406]	ITcIoGevImage	获取指向 GVSP 前导有效载荷的指针。
SetGevStatus [▶ 406]	ITcIoGevImage	设置块状态。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.4.1 CheckIfCopy



检查图像数据是否为副本。

语法

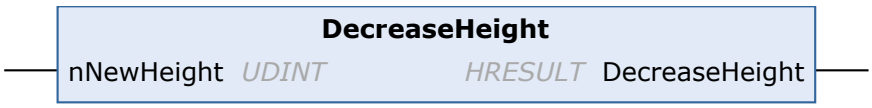
定义：

```
METHOD CheckIfCopy : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.2.4.2 DecreaseHeight



如果图像高度小于当前值，则将图像高度设置为 newHeight。

语法

定义：

```
METHOD DecreaseHeight : HRESULT
VAR_INPUT
    nNewHeight : UDINT;
END_VAR
```

输入

名称	类型	描述
nNewHeight	UDINT	要设置的高度值。

返回值

HRESULT [▶ 135]

6.1.3.2.4.3 GetBlockId



获取 GVSP 块 ID。

语法

定义：

```
METHOD GetBlockId : HRESULT
VAR_INPUT
    nBlockId : Reference To ULINT;
END_VAR
```

输入

名称	类型	描述
nBlockId	引用 ULINT	返回块 ID。

返回值

HRESULT [▶ 135]

6.1.3.2.4.4 GetGevStatus



获取块状态。

语法

定义：

```
METHOD GetGevStatus : HRESULT
VAR_INPUT
    nGevStatus : Reference To UINT;
END_VAR
```

输入

名称	类型	描述
nGevStatus	引用 UINT	返回 GigE Vision 状态代码。

返回值

HRESULT [▶ 135]

6.1.3.2.4.5 GetGvspImageInfo



获取指向 GVSP 前导有效载荷的指针。

语法

定义：

```
METHOD GetGvspImageInfo : HRESULT
VAR_INPUT
    ppGvspImageInfo : Pointer To Pointer To GVSP_IMAGE_INFO;
END_VAR
```

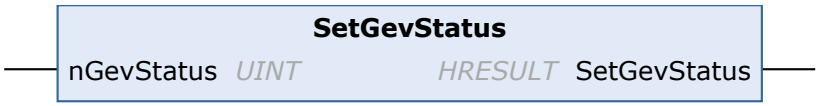
输入

名称	类型	描述
ppGvspImageInfo	指针指向 GVSP_IMAGE_INFO [▶ 220]	指向 GVSP 元信息的指针。

返回值

HRESULT [▶ 135]

6.1.3.2.4.6 SetGevStatus



设置块状态。

语法

定义：

```
METHOD SetGevStatus : HRESULT
VAR_INPUT
    nGevStatus : UINT;
END_VAR
```

输入

名称	类型	描述
nGevStatus	UINT	GigE Vision 状态代码。

返回值

HRESULT [▶ 135]

6.1.3.2.5 ITcVnDisplayableImage

可显示图像的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnDisplayableImage

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。

更多信息

该接口用于通过 ADS 显示图像（例如使用 ADS 图像查看）。它不适合用于处理图像。处理图像需要 ITcVnImage [▶ 407]接口。关于更多信息，请参见可处理和可显示图像之间的区别 [▶ 143]。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.6 ITcVnImage

图像的基本接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnImageBase [▶ 410]
    ITcVnImage

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
GetImageData [▶ 410]	ITcVnImageBase	获取数据接口（不要在其中明确释放指针！）。
GetImageSize [▶ 411]	ITcVnImageBase	获取图像大小。
GetWidth [▶ 411]	ITcVnImageBase	获取图像宽度（或者使用 F_VN_GetImageWidth）。
GetHeight [▶ 412]	ITcVnImageBase	获取图像高度（或者使用 F_VN_GetImageHeight）。
GetXPadding [▶ 412]	ITcVnImageBase	获取水平填充。
GetYPadding [▶ 413]	ITcVnImageBase	获取垂直填充。
GetPixelFormat [▶ 413]	ITcVnImageBase	获取像素格式（或者使用 F_VN_GetPixelFormat）。
GetImageInfo [▶ 414]	ITcVnImageBase	获取包含图像所有常见元信息的结构体。这基本上包含了可通过该接口访问的所有元信息。（或者使用 F_VN_GetImageInfo。）

名称	起源	描述
Init [▶ 408]	ITcVnImage	使用 ITcVnImageBase 接口初始化图像。
GetRowPointer [▶ 409]	ITcVnImage	获取指向图像特定行的指针。
ReleaseRowPointer [▶ 409]	ITcVnImage	释放指向图像特定行的指针。

更多信息

接口ITcVnImage用于处理图像 [▶ 143]。它提供了获取元数据（如图像尺寸、通道数、像素类型等）的方法以及访问原始图像数据的方法。

这个接口不适合通过 ADS 显示图像（例如通过ADS 图像查看 [▶ 128]）。通过 ADS 显示图像需要ITcVnDisplayableImage [▶ 407]接口。可以通过以下功能进行转换：

- F_VN_CopyIntoDisplayableImage [▶ 781]
- F_VN_TransformIntoDisplaybleImage [▶ 818]

相关函数

- 基本图像操作
- 代数图像操作

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.6.1 Init



使用 ITcVnImageBase 接口初始化图像。

语法

定义：

```
METHOD Init : HRESULT
VAR_INPUT
    ipImageBase : ITcVnImageBase;
END_VAR
```

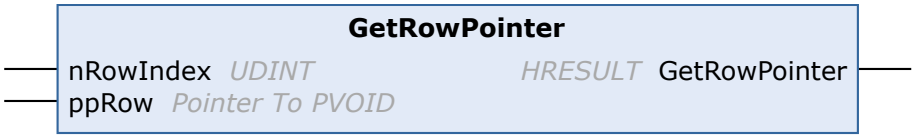
输入

名称	类型	描述
ipImageBase	ITcVnImageBase [▶ 410]	从中获取数据的图像。

返回值

HRESULT [▶ 135]

6.1.3.2.6.2 GetRowPointer



获取指向图像特定行的指针。

语法

定义：

```
METHOD GetRowPointer : HRESULT
VAR_INPUT
    nRowIndex : UDINT;
    ppRow     : Pointer To PVOID;
END_VAR
```

输入

名称	类型	描述
nRowIndex	UDINT	行索引。
ppRow	指向 PVOID 的指针	返回指向所请求图像行的指针。

返回值

HRESULT [▶ 135]

更多信息

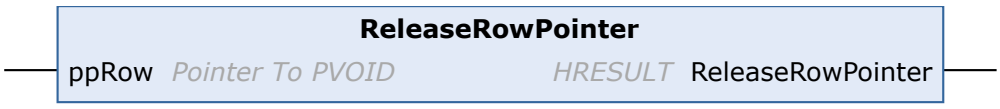
确保描述指针的基本类型与图像的像素类型相同。

注意

启用行指针

如果使用 GetRowPointer 创建了一个指针，你也必须用 ReleaseRowPointer 方法再次释放它！

6.1.3.2.6.3 ReleaseRowPointer



释放指向图像特定行的指针。

语法

定义：

```
METHOD ReleaseRowPointer : HRESULT
VAR_INPUT
    ppRow : Pointer To PVOID;
END_VAR
```

输入

名称	类型	描述
ppRow	指向 PVOID 的指针	指向要释放的行指针的指针。

返回值

HRESULT [▶ 135]

6.1.3.2.7 ITcVnImageBase

所有图像类型的基本接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnImageBase

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
GetImageData [▶ 410]	ITcVnImageBase	获取数据接口（不要在其中明确释放指针！）。
GetImageSize [▶ 411]	ITcVnImageBase	获取图像大小。
GetWidth [▶ 411]	ITcVnImageBase	获取图像宽度（或者使用 F_VN_GetImageWidth）。
GetHeight [▶ 412]	ITcVnImageBase	获取图像高度（或者使用 F_VN_GetImageHeight）。
GetXPadding [▶ 412]	ITcVnImageBase	获取水平填充。
GetYPadding [▶ 413]	ITcVnImageBase	获取垂直填充。
GetPixelFormat [▶ 413]	ITcVnImageBase	获取像素格式（或者使用 F_VN_GetPixelFormat）。
GetImageInfo [▶ 414]	ITcVnImageBase	获取包含图像所有常见元信息的结构体。这基本上包含了可通过该接口访问的所有元信息。（或者使用 F_VN_GetImageInfo。）

更多信息

接口`ITcVnImageBase`用于内部用途，而用户通常不需要。

对于处理图像 [▶ 143]，请使用ITcVnImage [▶ 407]接口代替。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.2.7.1 GetImageData



获取数据接口（不要在其中明确释放指针！）。

语法

定义：

```
METHOD GetImageData : HRESULT
VAR_INPUT
    pipImageData : Pointer To ITcVnData;
END_VAR
```

输入

名称	类型	描述
pipImageData	指针指向 ITcVnData [▶ 420]	返回指向图像数据接口的指针。

返回值

HRESULT [▶ 135]

6.1.3.2.7.2 GetImageSize



获取图像大小。

语法

定义：

```
METHOD GetImageSize : HRESULT
VAR_INPUT
    nSize : Reference To ULINT;
END_VAR
```

输入

名称	类型	描述
nSize	引用 ULINT	返回以字节为单位的图像大小。

返回值

HRESULT [▶ 135]

6.1.3.2.7.3 GetWidth



获取图像宽度（或者使用 F_VN_GetImageWidth）。

语法

定义：

```
METHOD GetWidth : HRESULT
VAR_INPUT
    nWidth : Reference To UDINT;
END_VAR
```

输入

名称	类型	描述
nWidth	引用 UDINT	返回以像素为单位的图像宽度。

返回值

HRESULT [▶ 135]

6.1.3.2.7.4 GetHeight



获取图像高度（或者使用 F_VN_GetImageHeight）。

语法

定义：

```
METHOD GetHeight : HRESULT
VAR_INPUT
    nHeight : Reference To UDINT;
END_VAR
```

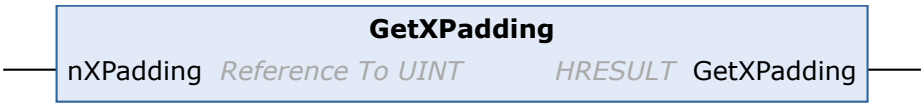
输入

名称	类型	描述
nHeight	引用 UDINT	返回以像素为单位的图像高度。

返回值

HRESULT [▶ 135]

6.1.3.2.7.5 GetXPadding



获取水平填充。

语法

定义：

```
METHOD GetXPadding : HRESULT
VAR_INPUT
    nXPadding : Reference To UINT;
END_VAR
```

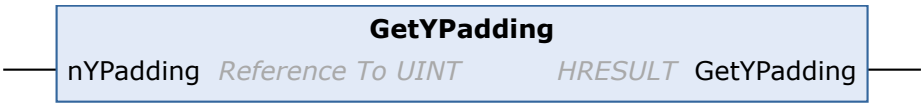
输入

名称	类型	描述
nXPadding	引用 UINT	返回以字节为单位的水平填充。

返回值

HRESULT [▶ 135]

6.1.3.2.7.6 GetYPadding



获取垂直填充。

语法

定义：

```
METHOD GetYPadding : HRESULT
VAR_INPUT
    nYPadding : Reference To UINT;
END_VAR
```

输入

名称	类型	描述
nYPadding	引用 UINT	返回以字节为单位的垂直填充。

返回值

HRESULT [▶ 135]

6.1.3.2.7.7 GetPixelFormat



获取像素格式（或者使用 F_VN_GetPixelFormat）。

语法

定义：

```
METHOD GetPixelFormat : HRESULT
VAR_INPUT
    stPixelFormat : Reference To TcVnPixelFormat;
END_VAR
```

输入

名称	类型	描述
stPixelFormat	引用 TcVnPixelFormat [▶ 235]	返回描述像素格式的结构体。

返回值

HRESULT [▶ 135]

6.1.3.2.7.8 GetImageInfo



获取包含图像所有常见元信息的结构体。这基本上包含了可通过该接口访问的所有元信息。（或者使用 F_VN_GetImageInfo。）

语法

定义：

```
METHOD GetImageInfo : HRESULT
VAR_INPUT
    stImageInfo : Reference To TcVnImageInfo;
END_VAR
```

输入

名称	类型	描述
stImageInfo	引用 TcVnImageInfo [▶ 225]	返回描述图像的结构体。

返回值

HRESULT [▶ 135]

6.1.3.2.8 ITcVnImageTimestamps

用于读取图像时间戳的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnImageTimestamps

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
GetTimestamps [▶ 414]	ITcVnImageTimestamps	获取选定的图像时间戳。

6.1.3.2.8.1 GetTimestamps



获取选定的图像时间戳。

语法

定义：

```
METHOD GetTimestamps : HRESULT
VAR_INPUT
    nTimestamps : ULINT;
    nArraySize  : UDINT;
    pTimestamps : Pointer To LINT;
END_VAR
```

输入

名称	类型	描述
nTimestamps	ULINT	选择要返回的时间戳（使用按位 OR 组合多个 ETcVnTimestamp 值）
nArraySize	UDINT	pTimestamps 数组大小（元素的数量）
pTimestamps	指针指向 LINT	返回请求的时间戳，按其 ETcVnTimestamp 值升序排列（确保提供足够大小的数组）。

返回值

HRESULT [▶ 135]

6.1.3.3 机器学习

该组包含机器学习的接口。

6.1.3.3.1 ITcVnColorModel

对图像颜色进行训练和分类的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnMlModel [▶ 417]
    ITcVnColorModel

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
GetTypeGuid [▶ 417]	ITcVnMlModel	获取模型的 GUID。
GetTypeName [▶ 418]	ITcVnMlModel	获取字符串形式的模型类型名称。
TrainImageColor [▶ 416]	ITcVnColorModel	训练所提供图像的颜色。
ClassifyImageColor [▶ 416]	ITcVnColorModel	计算每个图像像素与所训练参考色的相似度。

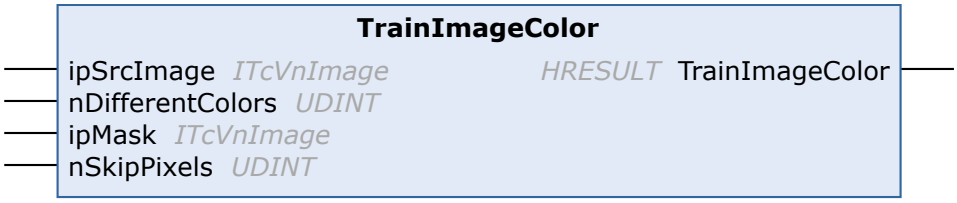
更多信息

接口`ITcVnColorModel`用于描述一个颜色区域。模型可以通过函数F_VN_TrainImageColor [▶ 1265]进行创建或调整，并使用函数F_VN_ReferenceColorSimilarity_ITcVnColorModel [▶ 1260]与图像的颜色进行比较。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.3.1.1 TrainImageColor



训练所提供图像的颜色。

语法

定义：

```
METHOD TrainImageColor : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
    nDifferentColors : UDINT;
    ipMask           : ITcVnImage;
    nSkipPixels      : UDINT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像。
nDifferentColors	UDINT	要区分的不同颜色的数量。
ipMask	ITcVnImage [▶ 407]	可选图像掩膜。
nSkipPixels	UDINT	在每个评估的颜色样本之间跳过的像素数量（以获得更好的性能）。0 将每个像素都考虑在内，往往更准确。

返回值

HRESULT [▶ 135]

6.1.3.3.1.2 ClassifyImageColor



计算每个图像像素与所训练参考色的相似度。

语法

定义：

```
METHOD ClassifyImageColor : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
    pipDestImage     : Pointer To ITcVnImage;
    fVariance        : REAL;
    fLuminanceWeight : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像。
pipDestImage	指针指向 ITcVnImage [▶ 407]	返回颜色相似度。
fVariance	REAL	允许的颜色方差。
fLuminanceWeight	REAL	权衡亮度的影响。

返回值

HRESULT [▶ 135]

6.1.3.3.2 ITcVnMlModel

机器学习模型的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnMlModel

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
GetTypeGuid [▶ 417]	ITcVnMlModel	获取模型的 GUID。
GetTypeName [▶ 418]	ITcVnMlModel	获取字符串形式的模型类型名称。

更多信息

接口`ITcVnMlModel`用于处理机器学习模型。目前，这只包括使用接口ITcVnColorModel [▶ 415]。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.3.2.1 GetTypeGuid



获取模型的 GUID。

语法

定义：

```
METHOD GetTypeGuid : HRESULT
VAR_INPUT
    nTypeGuid : Reference To GUID;
END_VAR
```

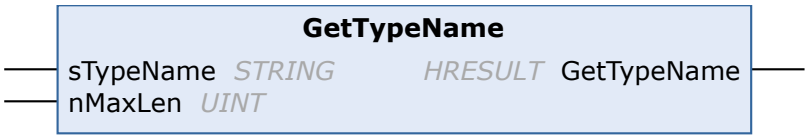
输入

名称	类型	描述
nTypeGuid	引用 GUID	返回模型的 GUID。

返回值

HRESULT [▶ 135]

6.1.3.3.2.2 GetTypeName



获取字符串形式的模型类型名称。

语法

定义：

```
METHOD GetTypeName : HRESULT
VAR_INPUT
    sTypeName : STRING;
    nMaxLen   : UINT;
END_VAR
```

输入

名称	类型	描述
sTypeName	STRING	以字符串形式返回模型类型名称。
nMaxLen	UINT	允许写入 sTypeName 的最大字符串长度。

返回值

HRESULT [▶ 135]

6.1.3.4 其它

该组包含其他接口。

6.1.3.4.1 ITcSerializableNotification

Interface required for sending serialized objects as ADS notifications.

继承层次

ITcUnknown [▶ 424]
  ITcSerializableNotification

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
GetComparisonValue [▶ 419]	ITcSerializableNotification	获取反映底层对象变化的值（例如对象最新变化的时间戳）。
SerializeRpcUnlocked [▶ 419]	ITcSerializableNotification	将对象序列化到给定的缓冲区中。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.4.1.1 GetComparisonValue



获取反映底层对象变化的值（例如对象最新变化的时间戳）。

语法

定义：

```
METHOD GetComparisonValue : HRESULT
VAR_INPUT
    nComparisonValue : Reference To LINT;
END_VAR
```

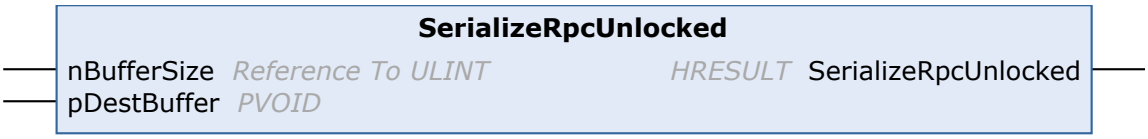
输入

名称	类型	描述
nComparisonValue	引用 LINT	返回比较值。

返回值

HRESULT [▶ 135]

6.1.3.4.1.2 SerializeRpcUnlocked



将对象序列化到给定的缓冲区中。

语法

定义：

```
METHOD SerializeRpcUnlocked : HRESULT
VAR_INPUT
    nBufferSize : Reference To ULINT;
    pDestBuffer : PVOID;
END_VAR
```

输入

名称	类型	描述
nBufferSize	引用 ULINT	最大缓冲区大小（输入）和实际使用的缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。

返回值

HRESULT [▶ 135]

6.1.3.4.2 ITcVnData

用于访问数据的接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnData

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
CheckIfCopy [▶ 421]	ITcVnData	检查数据是否为副本（副本返回 S_OK，否则返回 S_FALSE）。
CheckIfImage [▶ 421]	ITcVnData	检查数据是否属于图像（属于返回 S_OK，不属于返回 S_FALSE）。
GetSize [▶ 421]	ITcVnData	获取数据的大小。
CheckDataPointer [▶ 422]	ITcVnData	检查数据指针是否非 0（不为 0 时返回 S_OK，否则返回 S_FALSE）。
GetDataPointer [▶ 422]	ITcVnData	获取数据指针（必须通过调用 ReleaseDataPointer 释放获取的数据指针）。
ReleaseDataPointer [▶ 422]	ITcVnData	释放数据指针。

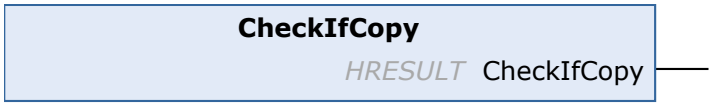
更多信息

接口`ITcVnData`用于内部用途，而用户通常不需要。对于访问原始图像数据，请使用接口ITcVnImage [▶ 407]的函数`GetRowPointer`代替。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.4.2.1 CheckIfCopy



检查数据是否为副本（副本返回 S_OK，否则返回 S_FALSE）。

语法

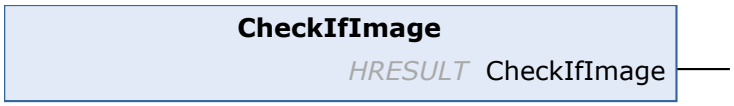
定义：

```
METHOD CheckIfCopy : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.4.2.2 CheckIfImage



检查数据是否属于图像（属于返回 S_OK，不属于返回 S_FALSE）。

语法

定义：

```
METHOD CheckIfImage : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.4.2.3 GetSize



获取数据的大小。

语法

定义：

```
METHOD GetSize : HRESULT
VAR_INPUT
    nSize : Reference To ULINT;
END_VAR
```

输入

名称	类型	描述
nSize	引用 ULINT	存储数据的大小（以字节为单位）。

返回值

HRESULT [▶ 135]

6.1.3.4.2.4 CheckDataPointer



检查数据指针是否非 0（不为 0 时返回 S_OK，否则返回 S_FALSE）。

语法

定义：

```
METHOD CheckDataPointer : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.4.2.5 GetDataPointer



获取数据指针（必须通过调用 ReleaseDataPointer 释放获取的数据指针）。

语法

定义：

```
METHOD GetDataPointer : HRESULT
VAR_INPUT
    ppData : Pointer To PVOID;
END_VAR
```

输入

名称	类型	描述
ppData	指向 PVOID 的指针	返回数据指针。

返回值

HRESULT [▶ 135]

6.1.3.4.2.6 ReleaseDataPointer



释放数据指针。

语法

定义：

```
METHOD ReleaseDataPointer : HRESULT
VAR_INPUT
    ppData : Pointer To PVOID;
END_VAR
```

输入

名称	类型	描述
ppData	指向 PVOID 的指针	指向要释放的数据指针的指针。

返回值

HRESULT [▶ 135]

6.1.3.4.3 ITcVnTimestamp

Offers an interface for timestamps.

继承层次

ITcUnknown [▶ 424]
  ITcVnTimestamp

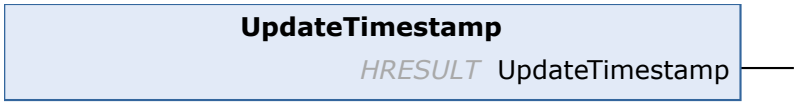
方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
UpdateTimestamp [▶ 423]	ITcVnTimestamp	将时间戳更新为当前时间
GetTimestamp [▶ 424]	ITcVnTimestamp	获取时间戳

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.4.3.1 UpdateTimestamp



将时间戳更新为当前时间

语法

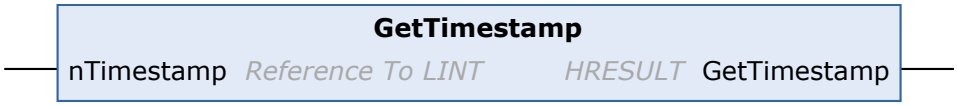
定义：

```
METHOD UpdateTimestamp : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.3.4.3.2 GetTimestamp



获取时间戳

语法

定义：

```
METHOD GetTimestamp : HRESULT
VAR_INPUT
    nTimestamp : Reference To LINT;
END_VAR
```

输入

名称	类型	描述
nTimestamp	引用 LINT	返回时间戳

返回值

HRESULT [▶ 135]

6.1.3.4.4 接口 ITcUnknown

ITcUnknown 定义引用计数以及对更具体接口的引用查询。

语法

```
TCOM_DECL_INTERFACE("00000001-0000-0000-e000-000000000064", ITcUnknown)
```

声明形式：TcInterfaces.h

要求包括：

方法

名称	描述
TcAddRef [▶ 424]	增加引用计数器。
TcQueryInterface [▶ 425]	通过 IID 查询已实施接口的引用。
TcRelease [▶ 426]	减少引用计数器。

每个 TcCOM 界面都直接或间接从 ITcUnknown 派生 因此，每个 TcCOM 模块类均执行 ITcUnknown，因为它是从 ITComObject 派生。

ITcUnknown 的标准执行可确保在释放最后一个引用后删除对象。因此，在调用 TcRelease() 后，不得取消引用接口指针。

6.1.3.4.4.1 方法 ITcUnknown:TcAddRef

增加引用计数器并返回新值。

语法

```
ULONG TcAddRef( )
```

返回值

产生引用计数值。

6.1.3.4.4.2 方法 ITcUnknown:TcQueryInterface

根据接口 ID (IID) 给出的接口查询接口指针。

语法

```
HRESULT TcQueryInterface(RITCID iid, PPVOID pipItf )
```

返回值

如果成功，将返回 S_OK（“0”）或其他正值，请参见返回值。关于扩展信息，请特别参见 ADS 返回代码 [▶ 2857]中的 HRESULT 栏。

如果所要求的接口不可用，该方法将返回 ADSERR_DEVICE_NOINTERFACE。

参数

名称	类型	描述
iid	RITCID	接口 IID。
pipItf	PPVOID	指向接口指针。当所请求的接口类型在相应实例中可用时设置。

描述

通过 IID 查询已实施接口的引用。建议使用智能指针来初始化和保存接口指针。

变量 1：

```
HRESULT GetTraceLevel(ITcUnkown* ip, TcTraceLevel& tl)
{
HRESULT hr = S_OK;
if (ip != NULL)
{
ITComObjectPtr spObj;
hr = ip->TcQueryInterface(spObj.GetIID(), &spObj);
if (SUCCEEDED(hr))
{
hr = spObj->TcGetObjPara(PID_TcTraceLevel, &tl, sizeof(tl));
}
return hr;
}
}
```

与智能指针相关的接口 ID 可用作 TcQueryInterface 中的参数。“&”操作符将返回指向智能指针内部接口指针成员变量的指针。变量 1 假定当 TcQueryInterface 指示成功时，接口指针被初始化。如果范围仍然存在，则智能指针 spObj 的析构函数会释放引用。

变量 2：

```
HRESULT GetTraceLevel(ITcUnkown* ip, TcTraceLevel& tl)
{
HRESULT hr = S_OK;
ITComObjectPtr spObj = ip;
if (spObj != NULL)
{
spObj->TcGetObjParam(PID_TcTraceLevel, &tl);
}
else
{
hr = ADS_E_NOINTERFACE;
}
return hr;
}
```

如果接口指针 ip 被分配给智能指针 spObj，则将在 ip 引用的实例上使用 IID_ITComObject 隐式调用 TcQueryInterface 方法。这样会缩短代码，但会丢失 TcQueryInterface 的原始返回代码。

6.1.3.4.4.3 方法 ITcUnknown:TcRelease

该方法减少引用计数器。

语法

```
ULONG TcRelease( )
```

返回值

产生引用计数值。

描述

减少引用计数器并返回新值。

如果引用计数器为 0，则对象会自行删除。

6.1.3.5 神经网络

6.1.3.5.1 ITcVnNeuralNetwork

神经网络接口。

继承层次

ITcUnknown [▶ 424]
  ITcVnMlModel [▶ 417]
    ITcVnNeuralNetwork

方法

名称	起源	描述
TcAddRef [▶ 424]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 425]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 426]	ITcUnknown	减少引用计数器。
GetTypeGuid [▶ 417]	ITcVnMlModel	获取模型的 GUID。
GetTypeName [▶ 418]	ITcVnMlModel	获取字符串形式的模型类型名称。
Execute [▶ 427]	ITcVnNeuralNetwork	使用提供的图像作为输入来执行神经网络，并提供一个输出层的输出。该方法不执行任何预处理或转换。
Execute_SingleLayerOutput [▶ 427]	ITcVnNeuralNetwork	使用提供的图像作为输入来执行神经网络，并提供指定输出层的输出。该方法不执行任何预处理或转换。
Execute_MultiLayerOutput [▶ 428]	ITcVnNeuralNetwork	使用提供的图像作为输入来执行神经网络，并提供指定输出层的输出。如果 sLayerNames 为空，该方法将返回未连接输出层的输出。该方法不执行任何预处理或转换。
GetLayerNames [▶ 428]	ITcVnNeuralNetwork	获取模型中所有图层的名称。如果 bOnlyOuput 标志被设置为 true，则仅返回未连接输出层的名称。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.3.5.1.1 执行



使用提供的图像作为输入来执行神经网络，并提供一个输出层的输出。该方法不执行任何预处理或转换。

语法

定义：

```
METHOD Execute : HRESULT
VAR_INPUT
    ipSrcImage   : ITcVnImage;
    pipDestImage : Pointer To ITcVnImage;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）
pipDestImage	指针指向 ITcVnImage [▶ 407]	目标图像（根据输出层创建适当的目标图像。要处理目标图像，必要时可使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数。

返回值

HRESULT [▶ 135]

6.1.3.5.1.2 Execute_SingleLayerOutput



使用提供的图像作为输入来执行神经网络，并提供指定输出层的输出。该方法不执行任何预处理或转换。

语法

定义：

```
METHOD Execute_SingleLayerOutput : HRESULT
VAR_INPUT
    ipSrcImage   : ITcVnImage;
    pipDestImage : Pointer To ITcVnImage;
    sLayerName   : STRING;
END_VAR
```

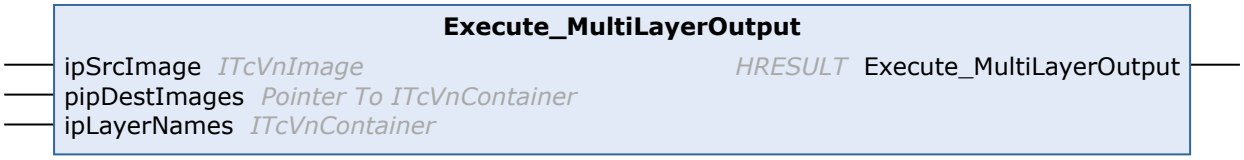
输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）
pipDestImage	指针指向 ITcVnImage [▶ 407]	目标图像（根据输出层创建适当的目标图像。要处理目标图像，必要时可使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数。
sLayerName	STRING	指定需要输出的层的名称。如果为空，则提供一个输出层的输出。

返回值

HRESULT [▶ 135]

6.1.3.5.1.3 Execute_MultiLayerOutput



使用提供的图像作为输入来执行神经网络，并提供指定输出层的输出。如果 sLayerNames 为空，该方法将返回未连接输出层的输出。该方法不执行任何预处理或转换。

语法

定义：

```
METHOD Execute_MultiLayerOutput : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    pipDestImages : Pointer To ITcVnContainer;
    ipLayerNames  : ITcVnContainer;
END_VAR
```

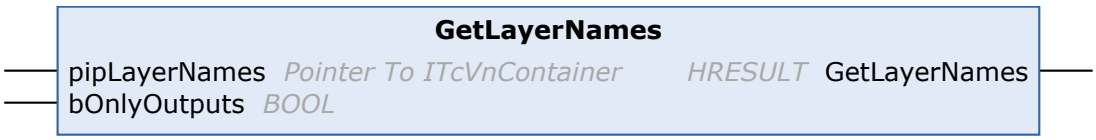
输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）
pipDestImages	指针指向 ITcVnContainer [▶ 365]	包含指定层的输出的图像的容器
ipLayerNames	ITcVnContainer [▶ 365]	要返回输出的层名称的容器（ContainerType_Vector_String_SINT）。如果设置为零，则容器 ipDestImages 将保留所有未连接输出层的输出。

返回值

HRESULT [▶ 135]

6.1.3.5.1.4 GetLayerNames



获取模型中所有图层的名称。如果 bOnlyOuput 标志被设置为 true，则仅返回未连接输出层的名称。

语法

定义：

```
METHOD GetLayerNames : HRESULT
VAR_INPUT
    pipLayerNames : Pointer To ITcVnContainer;
    bOnlyOutputs  : BOOL;
END_VAR
```

输入

名称	类型	描述
pipLayerNames	指针指向 ITcVnContainer [▶ 365]	带有层名称的容器（ContainerType_Vector_String_SINT）
bOnlyOutputs	BOOL	如果为 true，则仅返回未连接输出层的名称

返回值

HRESULT [▶ 135]

6.1.4 功能

6.1.4.1 偏振图像

该组包含用于处理和分析符合特定偏振图像格式的偏振图像的函数。

函数

- F_VN_DemosaicPolarizedImage [▶ 431]：根据偏振角度（0°、45°、90° 和 135°）从原始传感器图像中重建四幅独立图像。
- F_VN_AngleAndDegreeOfLinearPolarization [▶ 429]：基于四幅偏振角度图像计算线性偏振角（AoLP）和线性偏振度（DoLP），适用于复杂检测任务，例如检测材料应力或细微划痕。
- F_VN_PolarizedImageGlareReduction [▶ 433]：基于原始数据减少眩光和反射，例如用于揭示光亮或透明表面上的隐藏细节。

特殊的偏振图像格式

这些函数需要特定的偏振格式，此类格式可通过搭载片上偏振滤光片的图像传感器采集，覆盖 0°、45°、90° 和 135° 四个偏振方向，例如倍福 VCS2002 或 VCS2003 相机。

6.1.4.1.1 F_VN_AngleAndDegreeOfLinearPolarization



根据四幅偏振角度图像（DemosaicPolarizedImage 的结果）计算线性偏振角与线性偏振度（AoLP、DoLP）。
可使用创建的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_AngleAndDegreeOfLinearPolarization : HRESULT
VAR_INPUT
    ip0Deg    : ITcVnImage;
    ip45Deg   : ITcVnImage;
    ip90Deg   : ITcVnImage;
    ip135Deg  : ITcVnImage;
    ipAoLP    : Reference To ITcVnImage;
    ipDoLP    : Reference To ITcVnImage;
    eDestType : ETcVnElementType;
    hrPrev    : HRESULT;
END_VAR
```

INPUTS

名称	类型	描述
ip0Deg	ITcVnImage [▶ 407]	偏振图像的 0° 分量（USINT 或 UINT，1 或 3 通道）
ip45Deg	ITcVnImage [▶ 407]	偏振图像的 45° 分量（USINT 或 UINT，1 或 3 通道）
ip90Deg	ITcVnImage [▶ 407]	偏振图像的 90° 分量（USINT 或 UINT，1 或 3 通道）
ip135Deg	ITcVnImage [▶ 407]	偏振图像的 135° 分量（USINT 或 UINT，1 或 3 通道）
ipAoLP	Reference To ITcVnImage [▶ 407]	返回线性偏振角（AoLP）图像（可选项，如果不需要，则设为 0。必须与源图像不同！如有需要，将创建适当的目标图像。）
ipDoLP	Reference To ITcVnImage [▶ 407]	返回线性偏振度（DoLP）图像（可选项，如果不需要，则设为 0。必须与源图像不同！如有需要，将创建适当的目标图像。）
eDestType	ETcVnElementType [▶ 191]	输出图像的元素类型（ET_REAL 或 ET_USINT）。对于 ET_REAL，AoLP 的范围为 [-pi/2，pi/2]，DoLP 的范围为 [0，1]。对于 ET_USINT，两者均缩放至 [0，255] 范围。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

`F_VN_AngleAndDegreeOfLinearPolarization` 函数根据四幅偏振角度图像计算线性偏振角（AoLP）和线性偏振度（DoLP）。这四幅图像是 F_VN_DemosaicPolarizedImage [▶ 431] 函数的结果。

AoLP 描述偏振面的取向，而 DoLP 表示某个像素处光的偏振程度。例如，这两个量可用于可视化材料属性、表面取向或应力，而这些在传统强度图像中往往无法辨识。

参数

偏振角度的输入图像

四幅输入图像 `ip0Deg`、`ip45Deg`、`ip90Deg` 和 `ip135Deg` 分别代表分离后的偏振角度（0°、45°、90° 和 135°）。每幅图像的类型必须为 `USINT`（8 位）或 `UINT`（16 位），且具有 1 或 3 个通道。

结果图像（AoLP 和 DoLP）

线性偏振角（AoLP）以图像形式输出 `ipAoLP`，线性偏振度（DoLP）以图像形式输出 `ipDoLP`。如果不需要相应的输出，这两个参数均可设为 0。

输出元素类型

类型为 ETcVnElementType [▶ 191] 的参数 `eDestType` 指定了输出图像的元素类型：

- `TCVN_ET_REAL`：AoLP 以 `[-π/2, π/2]` 范围输出，DoLP 以 `[0, 1]` 范围输出。
- `TCVN_ET_USINT`：两个值均缩放至 `[0, 255]` 范围。

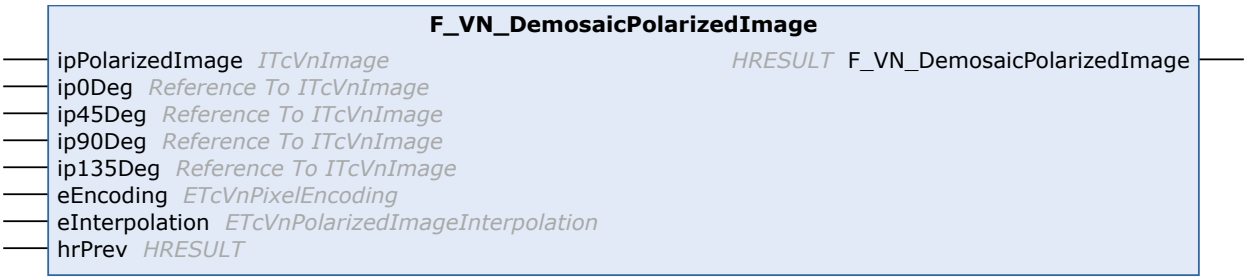
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.1.2 F_VN_DemosaicPolarizedImage



对偏振图像（由搭载特殊偏振传感器的相机拍摄）进行去马赛克处理，将其分离为四个偏振角度（预期传感器布局：左上 90°、右上 45°、左下 135°、右下 0°）。

语法

定义：

```
FUNCTION F_VN_DemosaicPolarizedImage : HRESULT
VAR_INPUT
    ipPolarizedImage : ITcVnImage;
    ip0Deg           : Reference To ITcVnImage;
    ip45Deg          : Reference To ITcVnImage;
    ip90Deg          : Reference To ITcVnImage;
    ip135Deg         : Reference To ITcVnImage;
    eEncoding        : ETcVnPixelEncoding;
    eInterpolation   : ETcVnPolarizedImageInterpolation;
    hrPrev           : HRESULT;
END_VAR
```

INPUTS

名称	类型	描述
ipPolarizedImage	ITcVnImage [▶ 407]	偏振图像（USINT 或 UINT）
ip0Deg	Reference To ITcVnImage [▶ 407]	返回偏振图像的 0° 分量（可选项，如果不需要，则设为 0。必须与 ipPolarizedImage 不同！如有需要，将创建适当的目标图像。）
ip45Deg	Reference To ITcVnImage [▶ 407]	返回偏振图像的 45° 分量（可选项，如果不需要，则设为 0。必须与 ipPolarizedImage 不同！如有需要，将创建适当的目标图像。）
ip90Deg	Reference To ITcVnImage [▶ 407]	返回偏振图像的 90° 分量（可选项，如果不需要，则设为 0。必须与 ipPolarizedImage 不同！如有需要，将创建适当的目标图像。）
ip135Deg	Reference To ITcVnImage [▶ 407]	返回偏振图像的 135° 分量（可选项，如果不需要，则设为 0。必须与 ipPolarizedImage 不同！如有需要，将创建适当的目标图像。）
eEncoding	ETcVnPixelEncoding [▶ 206]	对于单色传感器，选择 PE_NONE；对于 Bayer 传感器，选择 PE_BAYER_XX

名称	类型	描述
eInterpolation	ETcVnPolarizedImageInterpolation [▶ 156]	选择 PII_NONE 可获取未插值数据，即 ipXDeg 结果图像的宽和高仅为 ipPolarizedImage 的一半。其他选项则会对数据进行插值处理，使其分辨率提升至与 ipPolarizedImage 一致的全分辨率。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_DemosaicPolarizedImage` 可从符合特定偏振图像格式的原始传感器图像中重建与偏振角度（0°、45°、90° 和 135°）相对应的四幅独立图像。

如果原始图像包含颜色信息，也可以选择去马赛克处理。该函数要求四个偏振方向采用以下滤光片布局：左上 90°、右上 45°、左下 135°、右下 0°。

这四幅独立的角度图像构成了进一步分析的基础，例如使用 F_VN_AngleAndDegreeOfLinearPolarization [▶ 429] 计算偏振角度和线性偏振度。

参数

输入图像

输入图像 `ipPolarizedImage` 必须是类型为 `USINT`（8 位）或 `UINT`（16 位）的偏振图像。

偏振角度的结果图像

四幅输出图像 `ip0Deg`、`ip45Deg`、`ip90Deg` 和 `ip135Deg` 分别返回对应偏振角度（0°、45°、90° 和 135°）的分离图像分量。如果不需要某个特定角度，可将相应参数设为 0。目标图像必须与输入图像 `ipPolarizedImage` 不同。

像素编码

类型为 ETcVnPixelEncoding [▶ 206] 的参数 `eEncoding` 指定了传感器的像素编码。对于单色传感器，选择 `TCVN_PE_NONE`。对于彩色偏振传感器，必须将相应的 Bayer 模式作为 `TCVN_PE_BAYER_XX` 传入。

插值类型

类型为 ETcVnPolarizedImageInterpolation [▶ 156] 的参数 `eInterpolation` 确定了分离后的角度图像如何进行插值或解析。由于每个偏振角度在水平和垂直方向上均仅占据每隔一个像素，因此未插值的结果图像的宽和高仅为输入图像的一半。
提供以下选项：

- `TCVN_PII_NONE`：不进行插值；结果图像的宽和高均为 `ipPolarizedImage` 的一半。
- `TCVN_PII_NEAREST_NEIGHBOR`、`TCVN_PII_BILINEAR`、`TCVN_PII_BICUBIC`、`TCVN_PII_LANCZOS4`：使用所选方法插值至完整传感器分辨率。

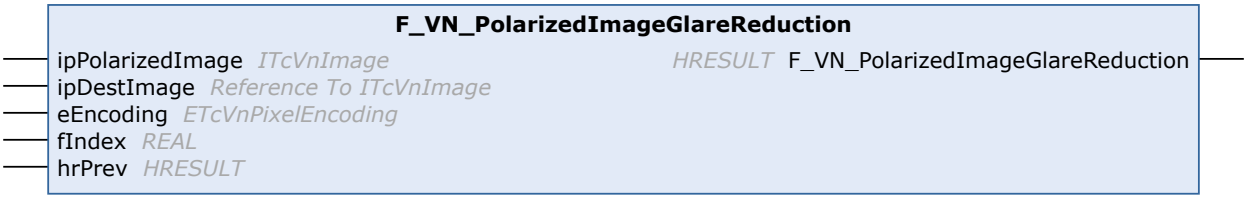
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.1.3 F_VN_PolarizedImageGlareReduction



通过为每个像素在四个偏振角度中选取一个较小的强度值（请参见 fIndex 参数），减少偏振图像（由搭载特殊偏振传感器的相机拍摄）中的眩光。如需进行更高级的评估，您可以使用 DemosaicPolarizedImage 访问所有单独的偏振角度。
可使用创建的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_PolarizedImageGlareReduction : HRESULT
VAR_INPUT
    ipPolarizedImage : ITcVnImage;
    ipDestImage      : Reference To ITcVnImage;
    eEncoding        : ETcVnPixelEncoding;
    fIndex           : REAL;
    hrPrev           : HRESULT;
END_VAR
```

INPUTS

名称	类型	描述
ipPolarizedImage	ITcVnImage [▶ 407]	偏振图像（USINT 或 UINT）
ipDestImage	Reference To ITcVnImage [▶ 407]	返回眩光减弱后的图像（如果需要，将创建适当的目标图像。）
eEncoding	ETcVnPixelEncoding [▶ 206]	对于单色传感器，选择 PE_NONE；对于 Bayer 传感器，选择 PE_BAYER_XX
fIndex	REAL	每个像素的四个偏振角度按强度排序。选择要用于目标图像的索引（范围 [0，3]）。小数部分表示对 2 个相邻索引取加权平均值。建议先尝试 0.0，因为这样可以实现最大程度的眩光减弱效果（但可能会丢失一些细节）。大于 1 的值虽然有效，但通常会增强眩光，而非减弱眩光。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

`F_VN_PolarizedImageGlareReduction` 函数可减少符合特定偏振图像格式的偏振图像中的眩光和反射。

对于每个像素，先对四个偏振角度的强度进行排序，然后使用 `fIndex` 确定结果图像的强度值。只有当强度在很大程度上取决于偏振角度时，也就是强度值存在显著差异时，才会对该像素进行调暗处理。非偏振图像区域在四个角度上的强度几乎相同，因此基本保持不变。

如需更详细地分析各个偏振角度，您可以使用 F_VN_DemosaicPolarizedImage [▶ 431] 函数。

参数

输入图像

输入图像 `ipPolarizedImage` 必须是类型为 `USINT`（8 位）或 `UINT`（16 位）的偏振图像。

结果图像

亮度降低后的图像通过 `ipDestImage` 返回。

eEncoding（编码方式）

类型为 ETcVnPixelEncoding [▶ 206] 的参数 `eEncoding` 指定了传感器的像素编码。对于单色传感器，选择 `TCVN_PE_NONE`。对于彩色偏振传感器，必须将相应的 Bayer 模式作为 `TCVN_PE_BAYER_XX` 传入。

选择索引

每个像素的四个偏振角度按强度排序。范围为 `0.0-3.0` 的参数 `fIndex` 决定了用于输出图像的强度值。如果指定了十进制值，则会对两个相邻索引或排序后的强度值计算加权平均值。建议先使用设置 `0.0`，因为这样可以实现最大程度的眩光减弱效果（但可能会丢失一些细节）。例如，`0.5` 会计算最小与次小强度值的平均值。不过，大于 `1.0` 的值通常会增强眩光，而非减弱眩光。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.2 许可证概述

以下列表显示了 TwinCAT Vision 功能与可用许可证的分配情况（见授权模式 [▶ 12]）。它显示了应用某些功能需要哪些许可证。

TF7100

始终需要基本许可证 TC3 Vision Base（TF7100）。

TC3 Vision Base

F_VN_AdaptiveThreshold [▶ 1316]

F_VN_AdaptiveThresholdExp [▶ 1317]

F_VN_AddContainers [▶ 493]

F_VN_AddImages [▶ 506]

F_VN_AddImagesWeighted [▶ 507]

F_VN_AddImagesWeightedExp [▶ 508]

F_VN_AddScalarToImage [▶ 509]

F_VN_AddToContainerElements_DINT [▶ 465]

F_VN_AddToContainerElements_INT [▶ 465]

F_VN_AddToContainerElements_LREAL [▶ 466]

F_VN_AddToContainerElements_REAL [▶ 467]

F_VN_AddToContainerElements_SINT [▶ 467]

F_VN_AddToContainerElements_TcVnKeyPoint [▶ 468]

F_VN_AddToContainerElements_TcVnPoint2_DINT [▶ 469]

F_VN_AddToContainerElements_TcVnPoint2_LREAL [▶ 470]
F_VN_AddToContainerElements_TcVnPoint2_REAL [▶ 471]
F_VN_AddToContainerElements_TcVnPoint3_LREAL [▶ 471]
F_VN_AddToContainerElements_TcVnPoint3_REAL [▶ 472]
F_VN_AddToContainerElements_TcVnRectangle_DINT [▶ 473]
F_VN_AddToContainerElements_TcVnRotatedRectangle [▶ 492]
F_VN_AddToContainerElements_TcVnVector2_DINT [▶ 474]
F_VN_AddToContainerElements_TcVnVector2_INT [▶ 475]
F_VN_AddToContainerElements_TcVnVector2_REAL [▶ 476]
F_VN_AddToContainerElements_TcVnVector2_SINT [▶ 476]
F_VN_AddToContainerElements_TcVnVector2_UINT [▶ 477]
F_VN_AddToContainerElements_TcVnVector2_USINT [▶ 478]
F_VN_AddToContainerElements_TcVnVector3_INT [▶ 479]
F_VN_AddToContainerElements_TcVnVector3_REAL [▶ 480]
F_VN_AddToContainerElements_TcVnVector3_SINT [▶ 481]
F_VN_AddToContainerElements_TcVnVector3_UINT [▶ 481]
F_VN_AddToContainerElements_TcVnVector3_USINT [▶ 482]
F_VN_AddToContainerElements_TcVnVector4_DINT [▶ 483]
F_VN_AddToContainerElements_TcVnVector4_INT [▶ 484]
F_VN_AddToContainerElements_TcVnVector4_LREAL [▶ 485]
F_VN_AddToContainerElements_TcVnVector4_REAL [▶ 491]
F_VN_AddToContainerElements_TcVnVector4_SINT [▶ 486]
F_VN_AddToContainerElements_TcVnVector4_UINT [▶ 486]
F_VN_AddToContainerElements_TcVnVector4_USINT [▶ 487]
F_VN_AddToContainerElements_UDINT [▶ 488]
F_VN_AddToContainerElements_UINT [▶ 489]
F_VN_AddToContainerElements_ULINT [▶ 489]
F_VN_AddToContainerElements_USINT [▶ 490]
F_VN_AddVectorToImage [▶ 509]
F_VN_AdjustActiveContour [▶ 1170]
F_VN_AdvanceIterator [▶ 744]
F_VN_AlignRotatedImageRegion [▶ 1109]
F_VN_AlignRotatedImageRegionExp [▶ 1110]
F_VN_AppendToContainer_DINT [▶ 546]
F_VN_AppendToContainer_INT [▶ 547]

F_VN_AppendToContainer_ITcVnContainer [▶ 547]
F_VN_AppendToContainer_ITcVnForwardIterator [▶ 548]
F_VN_AppendToContainer_ITcVnImage [▶ 549]
F_VN_AppendToContainer_LREAL [▶ 550]
F_VN_AppendToContainer_REAL [▶ 550]
F_VN_AppendToContainer_SINT [▶ 551]
F_VN_AppendToContainer_TcVnDMatch [▶ 552]
F_VN_AppendToContainer_TcVnKeyPoint [▶ 553]
F_VN_AppendToContainer_TcVnPoint2_DINT [▶ 554]
F_VN_AppendToContainer_TcVnPoint2_LREAL [▶ 554]
F_VN_AppendToContainer_TcVnPoint2_REAL [▶ 555]
F_VN_AppendToContainer_TcVnPoint3_LREAL [▶ 556]
F_VN_AppendToContainer_TcVnPoint3_REAL [▶ 557]
F_VN_AppendToContainer_TcVnRectangle_DINT [▶ 558]
F_VN_AppendToContainer_TcVnRotatedRectangle [▶ 545]
F_VN_AppendToContainer_TcVnVector2_DINT [▶ 559]
F_VN_AppendToContainer_TcVnVector2_INT [▶ 559]
F_VN_AppendToContainer_TcVnVector2_REAL [▶ 560]
F_VN_AppendToContainer_TcVnVector2_SINT [▶ 561]
F_VN_AppendToContainer_TcVnVector2_UINT [▶ 562]
F_VN_AppendToContainer_TcVnVector2_USINT [▶ 563]
F_VN_AppendToContainer_TcVnVector3_INT [▶ 564]
F_VN_AppendToContainer_TcVnVector3_REAL [▶ 564]
F_VN_AppendToContainer_TcVnVector3_SINT [▶ 565]
F_VN_AppendToContainer_TcVnVector3_UINT [▶ 566]
F_VN_AppendToContainer_TcVnVector3_USINT [▶ 567]
F_VN_AppendToContainer_TcVnVector4_DINT [▶ 568]
F_VN_AppendToContainer_TcVnVector4_INT [▶ 569]
F_VN_AppendToContainer_TcVnVector4_LREAL [▶ 569]
F_VN_AppendToContainer_TcVnVector4_REAL [▶ 570]
F_VN_AppendToContainer_TcVnVector4_SINT [▶ 571]
F_VN_AppendToContainer_TcVnVector4_UINT [▶ 572]
F_VN_AppendToContainer_TcVnVector4_USINT [▶ 573]
F_VN_AppendToContainer_UDINT [▶ 574]
F_VN_AppendToContainer_UINT [▶ 574]

F_VN_AppendToContainer_ULINT [▶ 575]

F_VN_AppendToContainer_USINT [▶ 576]

F_VN_ApplyColorMap [▶ 1244]

F_VN_ApplyLut [▶ 1245]

F_VN_ApplyRotationToAffineTransformation [▶ 1111]

F_VN_ApplyRotationToAffineTransformationExp [▶ 1112]

F_VN_ApplyScalingToAffineTransformation [▶ 1113]

F_VN_ApplyScalingToAffineTransformationExp [▶ 1114]

F_VN_ApplyTranslationToAffineTransformation [▶ 1114]

F_VN_ApplyTranslationToAffineTransformationExp [▶ 1115]

F_VN_ApplyYAxisInversionToAffineTransformation [▶ 1116]

F_VN_ApplyYAxisInversionToAffineTransformationExp [▶ 1117]

F_VN_ApproximatePolygon [▶ 963]

F_VN_BilateralFilter [▶ 1278]

F_VN_BilateralFilterExp [▶ 1280]

F_VN_BitwiseAndContainers [▶ 493]

F_VN_BitwiseAndImages [▶ 510]

F_VN_BitwiseAndImagesExp [▶ 511]

F_VN_BitwiseAndScalarWithImage [▶ 512]

F_VN_BitwiseAndVectorWithImage [▶ 513]

F_VN_BitwiseNotContainer [▶ 494]

F_VN_BitwiseNotImage [▶ 513]

F_VN_BitwiseNotImageExp [▶ 514]

F_VN_BitwiseOrContainers [▶ 495]

F_VN_BitwiseOrImages [▶ 515]

F_VN_BitwiseOrImagesExp [▶ 516]

F_VN_BitwiseOrScalarWithImage [▶ 517]

F_VN_BitwiseOrVectorWithImage [▶ 517]

F_VN_BitwiseXorContainers [▶ 496]

F_VN_BitwiseXorImages [▶ 518]

F_VN_BitwiseXorImagesExp [▶ 519]

F_VN_BitwiseXorScalarWithImage [▶ 520]

F_VN_BitwiseXorVectorWithImage [▶ 521]

F_VN_BlendImages [▶ 522]

F_VN_BoxFilter [▶ 1282]

F_VN_BoxFilterExp [▶ 1283]
F_VN_BrightBorderObjects [▶ 1284]
F_VN_CalibrateCameraPlanar [▶ 1083]
F_VN_CalibrateCameraPlanarExp [▶ 1084]
F_VN_CannyEdgeDetection [▶ 1209]
F_VN_CannyEdgeDetectionExp [▶ 1211]
F_VN_CheckColorRange [▶ 1319]
F_VN_CheckFunctionInitialization [▶ 1514]
F_VN_CheckIfEmpty [▶ 745]
F_VN_CheckIfIteratorIsAtEnd [▶ 746]
F_VN_CheckIfPointIsInsideContour [▶ 964]
F_VN_Clahe [▶ 1246]
F_VN_ClaheExp [▶ 1247]
F_VN_ClipLineToBoundary [▶ 1515]
F_VN_ClipLineToBoundary_ITcVnImage [▶ 1516]
F_VN_ClipLineToBoundary_TcVnRectangle_DINT [▶ 1517]
F_VN_CombineImageChannels [▶ 775]
F_VN_ConnectedComponents [▶ 1213]
F_VN_ConnectedComponentsExp [▶ 1214]
F_VN_ConnectedComponentsWithStats [▶ 1215]
F_VN_ConnectedComponentsWithStatsExp [▶ 1216]
F_VN_ContainerAverage [▶ 950]
F_VN_ContainerAverageElementwise2 [▶ 951]
F_VN_ContainerAverageElementwise3 [▶ 953]
F_VN_ContainerAverageElementwise4 [▶ 954]
F_VN_ContainerAverageVariance [▶ 955]
F_VN_ContainerAverageVarianceElementwise2 [▶ 956]
F_VN_ContainerAverageVarianceElementwise3 [▶ 958]
F_VN_ContainerAverageVarianceElementwise4 [▶ 959]
F_VN_ContourArea [▶ 965]
F_VN_ContourCenterOfMass [▶ 966]
F_VN_ContourCircularity [▶ 967]
F_VN_ContourCircularityExp [▶ 993]
F_VN_ContourConvexity [▶ 967]
F_VN_ContourEccentricity [▶ 968]

F_VN_ContourElongation [▶ 969]

F_VN_ContourExtremePoint [▶ 970]

F_VN_ContourInertiaRatio [▶ 971]

F_VN_ContourMoments [▶ 972]

F_VN_ContourOrientation [▶ 973]

F_VN_ContourOrientationExp [▶ 974]

F_VN_ContourPerimeter [▶ 975]

F_VN_ContourRoundness [▶ 976]

F_VN_ConvertCartesianToPolarAngleImage [▶ 1118]

F_VN_ConvertCartesianToPolarAngleImageExp [▶ 1118]

F_VN_ConvertCartesianToPolarAngles [▶ 1119]

F_VN_ConvertCartesianToPolarAnglesExp [▶ 1120]

F_VN_ConvertCartesianToPolarImages [▶ 1121]

F_VN_ConvertCartesianToPolarImagesExp [▶ 1122]

F_VN_ConvertCartesianToPolarMagnitudeImage [▶ 1123]

F_VN_ConvertCartesianToPolarMagnitudes [▶ 1123]

F_VN_ConvertCartesianToPolarPoints [▶ 1124]

F_VN_ConvertCartesianToPolarPointsExp [▶ 1125]

F_VN_ConvertColorSpace [▶ 1248]

F_VN_ConvertContainerType [▶ 747]

F_VN_ConvertDataLayout [▶ 1549]

F_VN_ConvertElementType [▶ 775]

F_VN_ConvertElementTypeExp [▶ 776]

F_VN_ConvertITcUnknownToITcVnBitmapExport [▶ 1519]

F_VN_ConvertITcUnknownToITcVnContainer [▶ 1519]

F_VN_ConvertITcUnknownToITcVnImage [▶ 1520]

F_VN_ConvertITcUnknownToITcVnMlModel [▶ 1521]

F_VN_ConvertMaps [▶ 1126]

F_VN_ConvertPolarToCartesianImages [▶ 1127]

F_VN_ConvertPolarToCartesianImagesExp [▶ 1128]

F_VN_ConvertPolarToCartesianPoints [▶ 1129]

F_VN_ConvertPolarToCartesianPointsExp [▶ 1129]

F_VN_ConvexHullPoints [▶ 977]

F_VN_ConvexHullPointsExp [▶ 977]

F_VN_ConvexityDefects [▶ 978]

F_VN_CopyContainer [▶ 748]
F_VN_CopyContainerElementsConditional_ITcVnContainer [▶ 749]
F_VN_CopyContainerElementsConditional_ITcVnForwardIterator [▶ 750]
F_VN_CopyImage [▶ 777]
F_VN_CopyImageRegion [▶ 778]
F_VN_CopyImageRegionToRegion [▶ 780]
F_VN_CopyIntoDisplayableImage [▶ 781]
F_VN_CopyIntoDisplayableImageExp [▶ 782]
F_VN_CountNonZeroPixels [▶ 1217]
F_VN_CreateAssociatedImage [▶ 783]
F_VN_CreateBandpassButterworthFilter [▶ 1057]
F_VN_CreateBandpassGaussianFilter [▶ 1058]
F_VN_CreateBandrejectButterworthFilter [▶ 1059]
F_VN_CreateBandrejectGaussianFilter [▶ 1061]
F_VN_CreateContainer [▶ 752]
F_VN_CreateContainerFromArray [▶ 752]
F_VN_CreateEmptyImage [▶ 783]
F_VN_CreateHighpassButterworthFilter [▶ 1062]
F_VN_CreateHighpassGaussianFilter [▶ 1063]
F_VN_CreateImage [▶ 784]
F_VN_CreateImageAndSetPixels [▶ 785]
F_VN_CreateImageFromArray [▶ 786]
F_VN_CreateLowpassButterworthFilter [▶ 1064]
F_VN_CreateLowpassGaussianFilter [▶ 1065]
F_VN_CreateNDimensionalImage [▶ 787]
F_VN_CreateNDimensionalImageAndSetPixels [▶ 788]
F_VN_CreateNDimensionalImageFromArray [▶ 789]
F_VN_CreateStructuringElement [▶ 1286]
F_VN_CustomElementWiseContainerOperation_ITcVnContainer [▶ 979]
F_VN_CustomElementWiseContainerOperation_ITcVnForwardIterator [▶ 980]
F_VN_CustomFilter [▶ 1287]
F_VN_CustomFilterExp [▶ 1290]
F_VN_DarkBorderObjects [▶ 1292]
F_VN_DecomposeAffineTransformation [▶ 1130]
F_VN_DecomposeAffineTransformationExp [▶ 1131]

F_VN_DecomposeHomography [▶ 1132]

F_VN_DecomposeHomographyExp [▶ 1133]

F_VN_DeinitializeFunction [▶ 1522]

F_VN_DetectBlobs [▶ 1171]

F_VN_DetectBlobsExp [▶ 1174]

F_VN_Dft [▶ 1066]

F_VN_DistanceTransformation [▶ 1218]

F_VN_DistanceTransformationExp [▶ 1220]

F_VN_DivideContainers [▶ 497]

F_VN_DivideContainersExp [▶ 498]

F_VN_DivideImageByScalar [▶ 523]

F_VN_DivideImageByVector [▶ 523]

F_VN_DivideImages [▶ 524]

F_VN_DivideScalarByImage [▶ 525]

F_VN_DivideVectorByImage [▶ 526]

F_VN_DoubleThreshold [▶ 1320]

F_VN_DrawArrow [▶ 999]

F_VN_DrawArrow_TcVnVector4_DINT [▶ 1000]

F_VN_DrawArrowExp [▶ 1001]

F_VN_DrawArrowExp_TcVnVector4_DINT [▶ 1002]

F_VN_DrawCircle [▶ 1003]

F_VN_DrawCircleExp [▶ 1005]

F_VN_DrawCircles [▶ 1006]

F_VN_DrawCirclesExp [▶ 1007]

F_VN_DrawCircularArc [▶ 1008]

F_VN_DrawCircularArcExp [▶ 1009]

F_VN_DrawComponents [▶ 1010]

F_VN_DrawComponentsExp [▶ 1011]

F_VN_DrawContours [▶ 1012]

F_VN_DrawContoursExp [▶ 1014]

F_VN_DrawEllipse [▶ 1015]

F_VN_DrawEllipseExp [▶ 1016]

F_VN_DrawLine [▶ 1019]

F_VN_DrawLine_TcVnVector4_DINT [▶ 1020]

F_VN_DrawLine_TcVnVector4_LREAL [▶ 1021]

F_VN_DrawLineExp [▶ 1022]
F_VN_DrawLineExp_TcVnVector4_DINT [▶ 1024]
F_VN_DrawLineExp_TcVnVector4_LREAL [▶ 1025]
F_VN_DrawLines [▶ 1026]
F_VN_DrawLinesExp [▶ 1027]
F_VN_DrawOrientation [▶ 1031]
F_VN_DrawOrientationExp [▶ 1032]
F_VN_DrawPoint [▶ 1033]
F_VN_DrawPointExp [▶ 1034]
F_VN_DrawPoints [▶ 1035]
F_VN_DrawPointsExp [▶ 1036]
F_VN_DrawPolygon [▶ 1037]
F_VN_DrawPolygonExp [▶ 1038]
F_VN_DrawRectangle [▶ 1039]
F_VN_DrawRectangle_TcVnRectangle_DINT [▶ 1040]
F_VN_DrawRectangle_TcVnRectangle_UDINT [▶ 1041]
F_VN_DrawRotatedRectangle [▶ 1042]
F_VN_DrawRotatedRectangleExp [▶ 1043]
F_VN_ElementwiseExp [▶ 527]
F_VN_ElementwiseLog [▶ 528]
F_VN_EnclosingCircle [▶ 981]
F_VN_EnclosingRectangle [▶ 982]
F_VN_EnclosingTriangle [▶ 983]
F_VN_EraseFromContainer [▶ 753]
F_VN_ExportContainer [▶ 754]
F_VN_ExportContainer_String [▶ 756]
F_VN_ExportContainerSize [▶ 757]
F_VN_ExportImage [▶ 790]
F_VN_ExportImageAsBmp [▶ 791]
F_VN_ExportImageAsBmpExp [▶ 792]
F_VN_ExportImageAsBmpSize [▶ 793]
F_VN_ExportImageAsBmpSizeExp [▶ 794]
F_VN_ExportImageSize [▶ 795]
F_VN_ExportSubContainer [▶ 758]
F_VN_ExportSubContainer_String [▶ 759]

F_VN_ExportSubContainerSize [▶ 760]
F_VN_ExtractContainerRange [▶ 761]
F_VN_FillCircle [▶ 1044]
F_VN_FillContainer_DINT [▶ 577]
F_VN_FillContainer_INT [▶ 578]
F_VN_FillContainer_ITcVnImage [▶ 579]
F_VN_FillContainer_LREAL [▶ 579]
F_VN_FillContainer_REAL [▶ 580]
F_VN_FillContainer_SINT [▶ 581]
F_VN_FillContainer_TcVnDMatch [▶ 582]
F_VN_FillContainer_TcVnKeyPoint [▶ 582]
F_VN_FillContainer_TcVnPoint2_DINT [▶ 583]
F_VN_FillContainer_TcVnPoint2_LREAL [▶ 584]
F_VN_FillContainer_TcVnPoint2_REAL [▶ 585]
F_VN_FillContainer_TcVnPoint3_LREAL [▶ 586]
F_VN_FillContainer_TcVnPoint3_REAL [▶ 587]
F_VN_FillContainer_TcVnRectangle_DINT [▶ 587]
F_VN_FillContainer_TcVnRotatedRectangle [▶ 576]
F_VN_FillContainer_TcVnVector2_DINT [▶ 588]
F_VN_FillContainer_TcVnVector2_INT [▶ 589]
F_VN_FillContainer_TcVnVector2_REAL [▶ 590]
F_VN_FillContainer_TcVnVector2_SINT [▶ 591]
F_VN_FillContainer_TcVnVector2_UINT [▶ 592]
F_VN_FillContainer_TcVnVector2_USINT [▶ 592]
F_VN_FillContainer_TcVnVector3_INT [▶ 593]
F_VN_FillContainer_TcVnVector3_REAL [▶ 594]
F_VN_FillContainer_TcVnVector3_SINT [▶ 595]
F_VN_FillContainer_TcVnVector3_UINT [▶ 596]
F_VN_FillContainer_TcVnVector3_USINT [▶ 597]
F_VN_FillContainer_TcVnVector4_DINT [▶ 598]
F_VN_FillContainer_TcVnVector4_INT [▶ 598]
F_VN_FillContainer_TcVnVector4_LREAL [▶ 599]
F_VN_FillContainer_TcVnVector4_REAL [▶ 600]
F_VN_FillContainer_TcVnVector4_SINT [▶ 601]
F_VN_FillContainer_TcVnVector4_UINT [▶ 602]

F_VN_FillContainer_TcVnVector4_USINT [▶ 603]
F_VN_FillContainer_UDINT [▶ 603]
F_VN_FillContainer_UINT [▶ 604]
F_VN_FillContainer_ULINT [▶ 605]
F_VN_FillContainer_USINT [▶ 606]
F_VN_FillContainerExp_DINT [▶ 607]
F_VN_FillContainerExp_INT [▶ 608]
F_VN_FillContainerExp_ITcVnImage [▶ 609]
F_VN_FillContainerExp_LREAL [▶ 610]
F_VN_FillContainerExp_REAL [▶ 611]
F_VN_FillContainerExp_SINT [▶ 611]
F_VN_FillContainerExp_TcVnDMatch [▶ 612]
F_VN_FillContainerExp_TcVnKeyPoint [▶ 613]
F_VN_FillContainerExp_TcVnPoint2_DINT [▶ 614]
F_VN_FillContainerExp_TcVnPoint2_LREAL [▶ 615]
F_VN_FillContainerExp_TcVnPoint2_REAL [▶ 616]
F_VN_FillContainerExp_TcVnPoint3_LREAL [▶ 617]
F_VN_FillContainerExp_TcVnPoint3_REAL [▶ 618]
F_VN_FillContainerExp_TcVnRectangle_DINT [▶ 619]
F_VN_FillContainerExp_TcVnRotatedRectangle [▶ 606]
F_VN_FillContainerExp_TcVnVector2_DINT [▶ 620]
F_VN_FillContainerExp_TcVnVector2_INT [▶ 621]
F_VN_FillContainerExp_TcVnVector2_REAL [▶ 622]
F_VN_FillContainerExp_TcVnVector2_SINT [▶ 623]
F_VN_FillContainerExp_TcVnVector2_UINT [▶ 624]
F_VN_FillContainerExp_TcVnVector2_USINT [▶ 625]
F_VN_FillContainerExp_TcVnVector3_INT [▶ 626]
F_VN_FillContainerExp_TcVnVector3_REAL [▶ 627]
F_VN_FillContainerExp_TcVnVector3_SINT [▶ 628]
F_VN_FillContainerExp_TcVnVector3_UINT [▶ 629]
F_VN_FillContainerExp_TcVnVector3_USINT [▶ 630]
F_VN_FillContainerExp_TcVnVector4_DINT [▶ 631]
F_VN_FillContainerExp_TcVnVector4_INT [▶ 632]
F_VN_FillContainerExp_TcVnVector4_LREAL [▶ 633]
F_VN_FillContainerExp_TcVnVector4_REAL [▶ 634]

F_VN_FillContainerExp_TcVnVector4_SINT [▶ 635]
F_VN_FillContainerExp_TcVnVector4_UINT [▶ 636]
F_VN_FillContainerExp_TcVnVector4_USINT [▶ 637]
F_VN_FillContainerExp_UDINT [▶ 638]
F_VN_FillContainerExp_UINT [▶ 639]
F_VN_FillContainerExp_ULINT [▶ 639]
F_VN_FillContainerExp_USINT [▶ 640]
F_VN_FillContours [▶ 1045]
F_VN_FillEllipse [▶ 1046]
F_VN_FillHoles [▶ 1293]
F_VN_FillPolygon [▶ 1047]
F_VN_FillRectangle [▶ 1048]
F_VN_FillRotatedRectangle [▶ 1049]
F_VN_FindContourHierarchyExp [▶ 1175]
F_VN_FindContours [▶ 1179]
F_VN_FindContoursExp [▶ 1181]
F_VN_FitEllipse [▶ 984]
F_VN_FitLine [▶ 984]
F_VN_FitLineExp [▶ 985]
F_VN_FlipImage [▶ 1135]
F_VN_FourierDescriptors [▶ 987]
F_VN_FuseImages [▶ 795]
F_VN_FuseImagesArray [▶ 797]
F_VN_GaussianFilter [▶ 1295]
F_VN_GaussianFilterExp [▶ 1296]
F_VN_GenerateAffineTransformationUnitMatrix2D [▶ 1135]
F_VN_GenerateColorMap [▶ 1249]
F_VN_GenerateCustomColorMap [▶ 1250]
F_VN_GetAffineTransformation [▶ 1136]
F_VN_GetAffineTransformation2D [▶ 1137]
F_VN_GetAffineTransformation2DExp [▶ 1138]
F_VN_GetAt_DINT [▶ 642]
F_VN_GetAt_INT [▶ 643]
F_VN_GetAt_ITcVnContainer [▶ 644]
F_VN_GetAt_ITcVnImage [▶ 645]

F_VN_GetAt_LREAL [▶ 645]
F_VN_GetAt_REAL [▶ 646]
F_VN_GetAt_SINT [▶ 647]
F_VN_GetAt_TcVnDMatch [▶ 648]
F_VN_GetAt_TcVnKeyPoint [▶ 649]
F_VN_GetAt_TcVnPoint2_DINT [▶ 650]
F_VN_GetAt_TcVnPoint2_LREAL [▶ 651]
F_VN_GetAt_TcVnPoint2_REAL [▶ 652]
F_VN_GetAt_TcVnPoint3_LREAL [▶ 653]
F_VN_GetAt_TcVnPoint3_REAL [▶ 654]
F_VN_GetAt_TcVnRectangle_DINT [▶ 655]
F_VN_GetAt_TcVnRotatedRectangle [▶ 641]
F_VN_GetAt_TcVnVector2_DINT [▶ 656]
F_VN_GetAt_TcVnVector2_INT [▶ 657]
F_VN_GetAt_TcVnVector2_REAL [▶ 658]
F_VN_GetAt_TcVnVector2_SINT [▶ 659]
F_VN_GetAt_TcVnVector2_UINT [▶ 660]
F_VN_GetAt_TcVnVector2_USINT [▶ 661]
F_VN_GetAt_TcVnVector3_INT [▶ 662]
F_VN_GetAt_TcVnVector3_REAL [▶ 663]
F_VN_GetAt_TcVnVector3_SINT [▶ 664]
F_VN_GetAt_TcVnVector3_UINT [▶ 665]
F_VN_GetAt_TcVnVector3_USINT [▶ 666]
F_VN_GetAt_TcVnVector4_DINT [▶ 667]
F_VN_GetAt_TcVnVector4_INT [▶ 668]
F_VN_GetAt_TcVnVector4_LREAL [▶ 669]
F_VN_GetAt_TcVnVector4_REAL [▶ 670]
F_VN_GetAt_TcVnVector4_SINT [▶ 670]
F_VN_GetAt_TcVnVector4_UINT [▶ 671]
F_VN_GetAt_TcVnVector4_USINT [▶ 672]
F_VN_GetAt_UDINT [▶ 673]
F_VN_GetAt_UINT [▶ 674]
F_VN_GetAt_ULINT [▶ 675]
F_VN_GetAt_USINT [▶ 676]
F_VN_GetConnectedComponent [▶ 1221]

F_VN_GetConnectedComponentExp [▶ 1222]
F_VN_GetContainer [▶ 762]
F_VN_GetContainerExp [▶ 763]
F_VN_GetForwardIterator [▶ 764]
F_VN_GetImageChannel [▶ 799]
F_VN_GetImageHeight [▶ 799]
F_VN_GetImageInfo [▶ 800]
F_VN_GetImageWidth [▶ 801]
F_VN_GetLayoutInfo [▶ 802]
F_VN_GetNumberOfElements [▶ 764]
F_VN_GetPerspectiveTransformation [▶ 1139]
F_VN_GetPixel [▶ 803]
F_VN_GetPixelFormat [▶ 804]
F_VN_GetRandomAccessIterator [▶ 765]
F_VN_GetRoi [▶ 805]
F_VN_GetTimestamp [▶ 1522]
F_VN_Histogram [▶ 1251]
F_VN_HistogramEqualization [▶ 1252]
F_VN_HistogramEqualizationExp [▶ 1253]
F_VN_HistogramExp [▶ 1254]
F_VN_Homography [▶ 1140]
F_VN_HomographyExp [▶ 1141]
F_VN_HoughCircles [▶ 1194]
F_VN_HoughCirclesExp [▶ 1195]
F_VN_HoughLines [▶ 1196]
F_VN_HoughLinesExp [▶ 1197]
F_VN_HoughLinesP [▶ 1198]
F_VN_HoughLinesPExp [▶ 1199]
F_VN_HuMomentInvariants [▶ 1523]
F_VN_ImageAbsoluteAverage [▶ 1237]
F_VN_ImageAverage [▶ 1223]
F_VN_ImageAverageExp [▶ 1224]
F_VN_ImageAverageStdDev [▶ 1225]
F_VN_ImageAverageStdDevExp [▶ 1226]
F_VN_ImageCenterOfMass [▶ 1227]

F_VN_ImageCenterOfMassExp [▶ 1228]

F_VN_ImageMedian [▶ 1229]

F_VN_ImageMedianExp [▶ 1229]

F_VN_ImageMoments [▶ 1230]

F_VN_IncrementIterator [▶ 766]

F_VN_InitMatrixStruct [▶ 1524]

F_VN_InsertIntoContainer_DINT [▶ 677]

F_VN_InsertIntoContainer_INT [▶ 678]

F_VN_InsertIntoContainer_ITcVnContainer [▶ 678]

F_VN_InsertIntoContainer_ITcVnForwardIterator [▶ 679]

F_VN_InsertIntoContainer_ITcVnImage [▶ 680]

F_VN_InsertIntoContainer_LREAL [▶ 681]

F_VN_InsertIntoContainer_REAL [▶ 682]

F_VN_InsertIntoContainer_SINT [▶ 682]

F_VN_InsertIntoContainer_TcVnDMatch [▶ 683]

F_VN_InsertIntoContainer_TcVnKeyPoint [▶ 684]

F_VN_InsertIntoContainer_TcVnPoint2_DINT [▶ 685]

F_VN_InsertIntoContainer_TcVnPoint2_LREAL [▶ 686]

F_VN_InsertIntoContainer_TcVnPoint2_REAL [▶ 686]

F_VN_InsertIntoContainer_TcVnPoint3_LREAL [▶ 687]

F_VN_InsertIntoContainer_TcVnPoint3_REAL [▶ 688]

F_VN_InsertIntoContainer_TcVnRectangle_DINT [▶ 689]

F_VN_InsertIntoContainer_TcVnRotatedRectangle [▶ 709]

F_VN_InsertIntoContainer_TcVnVector2_DINT [▶ 690]

F_VN_InsertIntoContainer_TcVnVector2_INT [▶ 691]

F_VN_InsertIntoContainer_TcVnVector2_REAL [▶ 692]

F_VN_InsertIntoContainer_TcVnVector2_SINT [▶ 692]

F_VN_InsertIntoContainer_TcVnVector2_UINT [▶ 693]

F_VN_InsertIntoContainer_TcVnVector2_USINT [▶ 694]

F_VN_InsertIntoContainer_TcVnVector3_INT [▶ 695]

F_VN_InsertIntoContainer_TcVnVector3_REAL [▶ 696]

F_VN_InsertIntoContainer_TcVnVector3_SINT [▶ 697]

F_VN_InsertIntoContainer_TcVnVector3_UINT [▶ 698]

F_VN_InsertIntoContainer_TcVnVector3_USINT [▶ 699]

F_VN_InsertIntoContainer_TcVnVector4_DINT [▶ 699]

F_VN_InsertIntoContainer_TcVnVector4_INT [▶ 700]
F_VN_InsertIntoContainer_TcVnVector4_LREAL [▶ 701]
F_VN_InsertIntoContainer_TcVnVector4_REAL [▶ 708]
F_VN_InsertIntoContainer_TcVnVector4_SINT [▶ 702]
F_VN_InsertIntoContainer_TcVnVector4_UINT [▶ 703]
F_VN_InsertIntoContainer_TcVnVector4_USINT [▶ 704]
F_VN_InsertIntoContainer_UDINT [▶ 705]
F_VN_InsertIntoContainer_UINT [▶ 705]
F_VN_InsertIntoContainer_ULINT [▶ 706]
F_VN_InsertIntoContainer_USINT [▶ 707]
F_VN_InverseDft [▶ 1067]
F_VN_InvertAffineTransformation [▶ 1142]
F_VN_InvertImageColor [▶ 1256]
F_VN_InvertImageColorExp [▶ 1256]
F_VN_InvertMatrix3x3 [▶ 1526]
F_VN_IteratorDistance [▶ 767]
F_VN_LaplacianFilter [▶ 1297]
F_VN_LaplacianFilterExp [▶ 1298]
F_VN_LineIntersectionPoint [▶ 1527]
F_VN_LineIntersectionPointAndAngle [▶ 1528]
F_VN_LocalMaxima [▶ 1300]
F_VN_LocalMinima [▶ 1301]
F_VN_MaxContainer [▶ 499]
F_VN_MaxElement [▶ 960]
F_VN_MaxElementElementwise_DINT [▶ 870]
F_VN_MaxElementElementwise_INT [▶ 871]
F_VN_MaxElementElementwise_LREAL [▶ 872]
F_VN_MaxElementElementwise_REAL [▶ 872]
F_VN_MaxElementElementwise_SINT [▶ 873]
F_VN_MaxElementElementwise_TcVnPoint2_DINT [▶ 874]
F_VN_MaxElementElementwise_TcVnPoint2_LREAL [▶ 875]
F_VN_MaxElementElementwise_TcVnPoint2_REAL [▶ 876]
F_VN_MaxElementElementwise_TcVnPoint3_LREAL [▶ 877]
F_VN_MaxElementElementwise_TcVnPoint3_REAL [▶ 877]
F_VN_MaxElementElementwise_TcVnVector2_DINT [▶ 878]

F_VN_MaxElementElementwise_TcVnVector2_INT [▶ 879]
F_VN_MaxElementElementwise_TcVnVector2_REAL [▶ 880]
F_VN_MaxElementElementwise_TcVnVector2_SINT [▶ 881]
F_VN_MaxElementElementwise_TcVnVector2_UINT [▶ 882]
F_VN_MaxElementElementwise_TcVnVector2_USINT [▶ 882]
F_VN_MaxElementElementwise_TcVnVector3_INT [▶ 883]
F_VN_MaxElementElementwise_TcVnVector3_REAL [▶ 884]
F_VN_MaxElementElementwise_TcVnVector3_SINT [▶ 885]
F_VN_MaxElementElementwise_TcVnVector3_UINT [▶ 886]
F_VN_MaxElementElementwise_TcVnVector3_USINT [▶ 887]
F_VN_MaxElementElementwise_TcVnVector4_DINT [▶ 887]
F_VN_MaxElementElementwise_TcVnVector4_INT [▶ 888]
F_VN_MaxElementElementwise_TcVnVector4_LREAL [▶ 889]
F_VN_MaxElementElementwise_TcVnVector4_REAL [▶ 896]
F_VN_MaxElementElementwise_TcVnVector4_SINT [▶ 890]
F_VN_MaxElementElementwise_TcVnVector4_UINT [▶ 891]
F_VN_MaxElementElementwise_TcVnVector4_USINT [▶ 892]
F_VN_MaxElementElementwise_UDINT [▶ 892]
F_VN_MaxElementElementwise_UINT [▶ 893]
F_VN_MaxElementElementwise_ULINT [▶ 894]
F_VN_MaxElementElementwise_USINT [▶ 895]
F_VN_MaxImage [▶ 528]
F_VN_MaxImageExp [▶ 529]
F_VN_MaxImageWithScalar [▶ 530]
F_VN_MaxImageWithScalarExp [▶ 531]
F_VN_MaxImageWithVector [▶ 532]
F_VN_MaxImageWithVectorExp [▶ 533]
F_VN_MaxPixelValue [▶ 1231]
F_VN_MaxPixelValueExp [▶ 1232]
F_VN_MedianElement [▶ 961]
F_VN_MedianElementElementwise_DINT [▶ 897]
F_VN_MedianElementElementwise_INT [▶ 898]
F_VN_MedianElementElementwise_LREAL [▶ 898]
F_VN_MedianElementElementwise_REAL [▶ 899]
F_VN_MedianElementElementwise_SINT [▶ 900]

F_VN_MedianElementElementwise_TcVnPoint2_DINT [▶ 901]
F_VN_MedianElementElementwise_TcVnPoint2_LREAL [▶ 902]
F_VN_MedianElementElementwise_TcVnPoint2_REAL [▶ 903]
F_VN_MedianElementElementwise_TcVnPoint3_LREAL [▶ 903]
F_VN_MedianElementElementwise_TcVnPoint3_REAL [▶ 904]
F_VN_MedianElementElementwise_TcVnVector2_DINT [▶ 905]
F_VN_MedianElementElementwise_TcVnVector2_INT [▶ 906]
F_VN_MedianElementElementwise_TcVnVector2_REAL [▶ 907]
F_VN_MedianElementElementwise_TcVnVector2_SINT [▶ 908]
F_VN_MedianElementElementwise_TcVnVector2_UINT [▶ 908]
F_VN_MedianElementElementwise_TcVnVector2_USINT [▶ 909]
F_VN_MedianElementElementwise_TcVnVector3_INT [▶ 910]
F_VN_MedianElementElementwise_TcVnVector3_REAL [▶ 911]
F_VN_MedianElementElementwise_TcVnVector3_SINT [▶ 912]
F_VN_MedianElementElementwise_TcVnVector3_UINT [▶ 913]
F_VN_MedianElementElementwise_TcVnVector3_USINT [▶ 913]
F_VN_MedianElementElementwise_TcVnVector4_DINT [▶ 914]
F_VN_MedianElementElementwise_TcVnVector4_INT [▶ 915]
F_VN_MedianElementElementwise_TcVnVector4_LREAL [▶ 916]
F_VN_MedianElementElementwise_TcVnVector4_REAL [▶ 923]
F_VN_MedianElementElementwise_TcVnVector4_SINT [▶ 917]
F_VN_MedianElementElementwise_TcVnVector4_UINT [▶ 918]
F_VN_MedianElementElementwise_TcVnVector4_USINT [▶ 918]
F_VN_MedianElementElementwise_UDINT [▶ 919]
F_VN_MedianElementElementwise_UINT [▶ 920]
F_VN_MedianElementElementwise_ULINT [▶ 921]
F_VN_MedianElementElementwise_USINT [▶ 922]
F_VN_MedianFilter [▶ 1302]
F_VN_MinContainer [▶ 499]
F_VN_MinElement [▶ 962]
F_VN_MinElementElementwise_DINT [▶ 924]
F_VN_MinElementElementwise_INT [▶ 924]
F_VN_MinElementElementwise_LREAL [▶ 925]
F_VN_MinElementElementwise_REAL [▶ 926]
F_VN_MinElementElementwise_SINT [▶ 927]

F_VN_MinElementElementwise_TcVnPoint2_DINT [▶ 928]
F_VN_MinElementElementwise_TcVnPoint2_LREAL [▶ 929]
F_VN_MinElementElementwise_TcVnPoint2_REAL [▶ 929]
F_VN_MinElementElementwise_TcVnPoint3_LREAL [▶ 930]
F_VN_MinElementElementwise_TcVnPoint3_REAL [▶ 931]
F_VN_MinElementElementwise_TcVnVector2_DINT [▶ 932]
F_VN_MinElementElementwise_TcVnVector2_INT [▶ 933]
F_VN_MinElementElementwise_TcVnVector2_REAL [▶ 934]
F_VN_MinElementElementwise_TcVnVector2_SINT [▶ 934]
F_VN_MinElementElementwise_TcVnVector2_UINT [▶ 935]
F_VN_MinElementElementwise_TcVnVector2_USINT [▶ 936]
F_VN_MinElementElementwise_TcVnVector3_INT [▶ 937]
F_VN_MinElementElementwise_TcVnVector3_REAL [▶ 938]
F_VN_MinElementElementwise_TcVnVector3_SINT [▶ 939]
F_VN_MinElementElementwise_TcVnVector3_UINT [▶ 939]
F_VN_MinElementElementwise_TcVnVector3_USINT [▶ 940]
F_VN_MinElementElementwise_TcVnVector4_DINT [▶ 941]
F_VN_MinElementElementwise_TcVnVector4_INT [▶ 942]
F_VN_MinElementElementwise_TcVnVector4_LREAL [▶ 943]
F_VN_MinElementElementwise_TcVnVector4_REAL [▶ 949]
F_VN_MinElementElementwise_TcVnVector4_SINT [▶ 944]
F_VN_MinElementElementwise_TcVnVector4_UINT [▶ 944]
F_VN_MinElementElementwise_TcVnVector4_USINT [▶ 945]
F_VN_MinElementElementwise_UDINT [▶ 946]
F_VN_MinElementElementwise_UINT [▶ 947]
F_VN_MinElementElementwise_ULINT [▶ 948]
F_VN_MinElementElementwise_USINT [▶ 949]
F_VN_MinImage [▶ 534]
F_VN_MinImageExp [▶ 534]
F_VN_MinImageWithScalar [▶ 535]
F_VN_MinImageWithScalarExp [▶ 536]
F_VN_MinImageWithVector [▶ 537]
F_VN_MinImageWithVectorExp [▶ 538]
F_VN_MinPixelValue [▶ 1233]
F_VN_MinPixelValueExp [▶ 1234]

F_VN_MixImageChannels [▶ 806]
F_VN_MorphologicalOperator [▶ 1303]
F_VN_MultiplyContainers [▶ 500]
F_VN_MultiplyImages [▶ 539]
F_VN_MultiplyImageWithScalar [▶ 539]
F_VN_MultiplyImageWithVector [▶ 540]
F_VN_MultiplyMatrices [▶ 1529]
F_VN_MultiplyWithContainerElements1 [▶ 501]
F_VN_MultiplyWithContainerElements2 [▶ 502]
F_VN_MultiplyWithContainerElements3 [▶ 503]
F_VN_MultiplyWithContainerElements4 [▶ 503]
F_VN_NegateContainer [▶ 504]
F_VN_NonMaxSuppression [▶ 1553]
F_VN_NonMaxSuppressionExp [▶ 1554]
F_VN_NormalizeImage [▶ 1257]
F_VN_NormalizeImageExp [▶ 1258]
F_VN_NormalizeImageForDisplay [▶ 1259]
F_VN_OptimalDftSize [▶ 1068]
F_VN_OrthogonalVector_TcVnVector2_DINT [▶ 1530]
F_VN_OrthogonalVector_TcVnVector2_LREAL [▶ 1530]
F_VN_OrthogonalVector_TcVnVector2_REAL [▶ 1531]
F_VN_PadImageBorder [▶ 807]
F_VN_PadImageBorderExp [▶ 808]
F_VN_PerspectiveTransformation [▶ 1143]
F_VN_PlotIntensityProfile [▶ 1050]
F_VN_PlotIntensityProfileExp [▶ 1051]
F_VN_PointToLineDistance_TcVnPoint2_DINT [▶ 1532]
F_VN_PointToLineDistance_TcVnPoint2_LREAL [▶ 1533]
F_VN_PointToLineDistance_TcVnPoint2_REAL [▶ 1533]
F_VN_PointToPointDirection_TcVnPoint2_DINT [▶ 1534]
F_VN_PointToPointDirection_TcVnPoint2_LREAL [▶ 1535]
F_VN_PointToPointDirection_TcVnPoint2_REAL [▶ 1536]
F_VN_PointToPointDistance_TcVnPoint2_DINT [▶ 1537]
F_VN_PointToPointDistance_TcVnPoint2_LREAL [▶ 1537]
F_VN_PointToPointDistance_TcVnPoint2_REAL [▶ 1538]

F_VN_PutLabel [▶ 1053]

F_VN_PutLabelExp [▶ 1054]

F_VN_PutText [▶ 1055]

F_VN_PutTextExp [▶ 1056]

F_VN_PyramidDown [▶ 1144]

F_VN_PyramidUp [▶ 1144]

F_VN_Reduce [▶ 1555]

F_VN_ReduceArg [▶ 1556]

F_VN_ReferenceColorSimilarity_ITcVnColorModel [▶ 1260]

F_VN_ReferenceColorSimilarity_ITcVnMlModel [▶ 1261]

F_VN_ReferenceColorSimilarity_TcVnVector3_LREAL [▶ 1262]

F_VN_ReferenceColorSimilarityExp_ITcVnColorModel [▶ 1262]

F_VN_ReferenceColorSimilarityExp_ITcVnMlModel [▶ 1263]

F_VN_ReferenceColorSimilarityExp_TcVnVector3_LREAL [▶ 1264]

F_VN_RegionOrientation [▶ 1235]

F_VN_RegionOrientationExp [▶ 1236]

F_VN_ReinterpretUnsupportedImage [▶ 809]

F_VN_RemapImage [▶ 1145]

F_VN_RemapImageExp [▶ 1146]

F_VN_RemapImageToLogPolarSpace [▶ 1147]

F_VN_RemapImageToLogPolarSpaceExp [▶ 1148]

F_VN_RemapImageToLogPolarSpaceExp2 [▶ 1149]

F_VN_RemapImageToPolarSpace [▶ 1150]

F_VN_RemapImageToPolarSpaceExp [▶ 1151]

F_VN_RemapImageToPolarSpaceExp2 [▶ 1152]

F_VN_RemoveLocalMaxima [▶ 1304]

F_VN_RemoveLocalMinima [▶ 1305]

F_VN_ReserveContainerMemory [▶ 768]

F_VN_ResetRoi [▶ 810]

F_VN_ReshapeImage [▶ 1557]

F_VN_ResizeImage [▶ 1153]

F_VN_ResizeImageExp [▶ 1154]

F_VN_ResizeRegion_TcVnRectangle_DINT [▶ 1155]

F_VN_ReverseContainer [▶ 769]

F_VN_RotatedRectangleCorners [▶ 1539]

F_VN_RotatedRectangleIntersection [▶ 1540]
F_VN_RotateImage [▶ 1156]
F_VN_RotateImageExp [▶ 1157]
F_VN_RotateImageExp2 [▶ 1158]
F_VN_SauvolaThreshold [▶ 1324]
F_VN_ScharrFilter [▶ 1306]
F_VN_ScharrFilterExp [▶ 1307]
F_VN_SeparableCustomFilter [▶ 1309]
F_VN_SeparableCustomFilterExp [▶ 1310]
F_VN_SetAt_DINT [▶ 710]
F_VN_SetAt_INT [▶ 711]
F_VN_SetAt_ITcVnContainer [▶ 712]
F_VN_SetAt_ITcVnImage [▶ 713]
F_VN_SetAt_LREAL [▶ 714]
F_VN_SetAt_REAL [▶ 714]
F_VN_SetAt_SINT [▶ 715]
F_VN_SetAt_TcVnDMatch [▶ 716]
F_VN_SetAt_TcVnKeyPoint [▶ 717]
F_VN_SetAt_TcVnPoint2_DINT [▶ 718]
F_VN_SetAt_TcVnPoint2_LREAL [▶ 719]
F_VN_SetAt_TcVnPoint2_REAL [▶ 720]
F_VN_SetAt_TcVnPoint3_LREAL [▶ 721]
F_VN_SetAt_TcVnPoint3_REAL [▶ 722]
F_VN_SetAt_TcVnRectangle_DINT [▶ 723]
F_VN_SetAt_TcVnRotatedRectangle [▶ 710]
F_VN_SetAt_TcVnVector2_DINT [▶ 724]
F_VN_SetAt_TcVnVector2_INT [▶ 725]
F_VN_SetAt_TcVnVector2_REAL [▶ 726]
F_VN_SetAt_TcVnVector2_SINT [▶ 727]
F_VN_SetAt_TcVnVector2_UINT [▶ 728]
F_VN_SetAt_TcVnVector2_USINT [▶ 729]
F_VN_SetAt_TcVnVector3_INT [▶ 730]
F_VN_SetAt_TcVnVector3_REAL [▶ 731]
F_VN_SetAt_TcVnVector3_SINT [▶ 732]
F_VN_SetAt_TcVnVector3_UINT [▶ 733]

F_VN_SetAt_TcVnVector3_USINT [▶ 734]
F_VN_SetAt_TcVnVector4_DINT [▶ 735]
F_VN_SetAt_TcVnVector4_INT [▶ 736]
F_VN_SetAt_TcVnVector4_LREAL [▶ 737]
F_VN_SetAt_TcVnVector4_REAL [▶ 738]
F_VN_SetAt_TcVnVector4_SINT [▶ 738]
F_VN_SetAt_TcVnVector4_UINT [▶ 739]
F_VN_SetAt_TcVnVector4_USINT [▶ 740]
F_VN_SetAt_UDINT [▶ 741]
F_VN_SetAt_UINT [▶ 742]
F_VN_SetAt_ULINT [▶ 743]
F_VN_SetAt_USINT [▶ 744]
F_VN_SetContainer [▶ 769]
F_VN_SetImageChannel [▶ 810]
F_VN_SetIteratorToBegin [▶ 770]
F_VN_SetPixel [▶ 811]
F_VN_SetPixels [▶ 812]
F_VN_SetPixelsExp [▶ 813]
F_VN_SetRngSeed [▶ 1541]
F_VN_SetRoi [▶ 814]
F_VN_SetRoi_TcVnRectangle_DINT [▶ 815]
F_VN_SetRoi_TcVnRectangle_UDINT [▶ 816]
F_VN_Sigmoid [▶ 1558]
F_VN_SigmoidExp [▶ 1559]
F_VN_SobelFilter [▶ 1312]
F_VN_SobelFilterExp [▶ 1312]
F_VN_SoftMax [▶ 1560]
F_VN_SplitImageChannels [▶ 817]
F_VN_StartAbsWatchdog [▶ 994]
F_VN_StartAbsWatchdogExp [▶ 995]
F_VN_StartRelWatchdog [▶ 996]
F_VN_StartRelWatchdogExp [▶ 997]
F_VN_StopWatchdog [▶ 998]
F_VN_SubtractContainers [▶ 505]
F_VN_SubtractImageFromScalar [▶ 541]

F_VN_SubtractImageFromVector [▶ 542]

F_VN_SubtractImages [▶ 543]

F_VN_SubtractScalarFromImage [▶ 544]

F_VN_SubtractVectorFromImage [▶ 544]

F_VN_Threshold [▶ 1322]

F_VN_TrainImageColor [▶ 1265]

F_VN_TrainImageColor_ITcVnMlModel [▶ 1267]

F_VN_TrainImageColorExp [▶ 1268]

F_VN_TrainImageColorExp_ITcVnMlModel [▶ 1270]

F_VN_TrainImageColorExp2 [▶ 1272]

F_VN_TrainImageColorExp2_ITcVnMlModel [▶ 1275]

F_VN_TransformCoordinatesPlanar_Container [▶ 1106]

F_VN_TransformCoordinatesPlanar_Point [▶ 1107]

F_VN_TransformIntoDisplayableImage [▶ 818]

F_VN_TransformIntoDisplayableImageExp [▶ 819]

F_VN_TransformIntoDisplayableImageExp2 [▶ 819]

F_VN_UpdateTimestamp [▶ 1542]

F_VN_UprightBoundingRectangle [▶ 992]

F_VN_UprightInnerRectangle [▶ 1238]

F_VN_UprightInnerRectangleExp [▶ 1239]

F_VN_VarianceFilter [▶ 1314]

F_VN_VarianceFilterExp [▶ 1315]

F_VN_WarpAffine [▶ 1159]

F_VN_WarpAffine_Container [▶ 1160]

F_VN_WarpAffine_Point [▶ 1161]

F_VN_WarpAffine_Rectangle [▶ 1162]

F_VN_WarpAffineExp [▶ 1163]

F_VN_WarpPerspective [▶ 1164]

F_VN_WarpPerspective_Container [▶ 1165]

F_VN_WarpPerspective_Point [▶ 1166]

F_VN_WarpPerspective_Rectangle [▶ 1167]

F_VN_WarpPerspectiveExp [▶ 1168]

F_VN_WatershedSegmentationExp [▶ 1323]

F_VN_WhiteBalance [▶ 1277]

TC3 Vision Code Reading

F_VN_DetectCodesNeuralNetwork [▶ 839]

F_VN_ReadBarcode [▶ 842]

F_VN_ReadBarcodeExp [▶ 842]

F_VN_ReadBarcodeExp2 [▶ 843]

F_VN_ReadBarcodeRoi [▶ 845]

F_VN_ReadBarcodeRoiExp [▶ 847]

F_VN_ReadDataMatrixCode [▶ 850]

F_VN_ReadDataMatrixCodeExp [▶ 852]

F_VN_ReadDataMatrixCodeExp2 [▶ 853]

F_VN_ReadDataMatrixCodeRoi [▶ 855]

F_VN_ReadDataMatrixCodeRoiExp [▶ 856]

F_VN_ReadDotCodeRoi [▶ 858]

F_VN_ReadDotCodeRoiExp [▶ 859]

F_VN_ReadPharmaCode [▶ 861]

F_VN_ReadPharmaCodeExp [▶ 861]

F_VN_ReadPharmaCodeExp2 [▶ 863]

F_VN_ReadQRCode [▶ 864]

F_VN_ReadQRCodeExp [▶ 866]

F_VN_ReadQRCodeExp2 [▶ 868]

TC3 Vision Code Quality

F_VN_GradeBarcode [▶ 821]

F_VN_GradeBarcodeExp [▶ 824]

F_VN_GradeDataMatrixCode [▶ 827]

F_VN_GradeDataMatrixCodeExp [▶ 829]

F_VN_GradeQRCode [▶ 832]

F_VN_GradeQRCodeExp [▶ 834]

TC3 Vision Metrology 2D

F_VN_AdjustSearchWindowOrientationToLinearEdge [▶ 1461]

F_VN_CalibrateCamera [▶ 1071]

F_VN_CalibrateCameraExp [▶ 1073]

F_VN_CalibrateCameraExp2 [▶ 1076]

F_VN_CalibrateCameraExp3 [▶ 1078]

F_VN_CalibrateCameraManually [▶ 1080]

F_VN_CalibrateCameraManuallyExp [▶ 1081]

F_VN_CalibrateLinescanCamera [▶ 1085]

F_VN_CalibrateLinescanCameraExp [▶ 1087]

F_VN_ClosestPointsBF [▶ 1462]

F_VN_CompensateLensDistortion [▶ 1089]

F_VN_CompensateLensDistortionExp1 [▶ 1089]

F_VN_CompensateLensDistortionExp2 [▶ 1090]

F_VN_CompensateLensDistortionForPoints [▶ 1091]

F_VN_CompensateLensDistortionForPointsExp1 [▶ 1092]

F_VN_CompensateLensDistortionForPointsExp2 [▶ 1093]

F_VN_DetectPatternPoints [▶ 1094]

F_VN_DetectPatternPoints2 [▶ 1095]

F_VN_DetectPatternPointsExp [▶ 1096]

F_VN_GenerateCalibrationPatternReferencePoints [▶ 1098]

F_VN_ImagePointsWorldDistance [▶ 1099]

F_VN_LocateCircularArc [▶ 1464]

F_VN_LocateCircularArcExp [▶ 1465]

F_VN_LocateCircularArcExp2 [▶ 1467]

F_VN_LocateEdge [▶ 1470]

F_VN_LocateEdgeExp [▶ 1471]

F_VN_LocateEdgeExp2 [▶ 1473]

F_VN_LocateEdges [▶ 1476]

F_VN_LocateEdgesExp [▶ 1479]

F_VN_LocateEdgesExp2 [▶ 1482]

F_VN_LocateEllipse [▶ 1485]

F_VN_LocateEllipseExp [▶ 1486]

F_VN_LocateEllipseExp2 [▶ 1488]

F_VN_MeasureAngleBetweenEdges [▶ 1490]

F_VN_MeasureAngleBetweenEdgesExp [▶ 1492]

F_VN_MeasureAngleBetweenEdgesExp2 [▶ 1494]

F_VN_MeasureEdgeDistance [▶ 1496]

F_VN_MeasureEdgeDistanceExp [▶ 1499]

F_VN_MeasureEdgeDistanceExp2 [▶ 1502]

F_VN_MeasureMinEdgeDistance [▶ 1505]

F_VN_MeasureMinEdgeDistanceExp [▶ 1508]

F_VN_MeasureMinEdgeDistanceExp2 [▶ 1511]

F_VN_SolvePnP [▶ 1100]

F_VN_SolvePnPExp [▶ 1101]

F_VN_SortDetectedPatternPoints [▶ 1103]

F_VN_TransformCoordinatesImageToWorld_Container [▶ 1103]

F_VN_TransformCoordinatesImageToWorld_Point [▶ 1105]

F_VN_TransformCoordinatesWorldToImage_Container [▶ 1107]

F_VN_TransformCoordinatesWorldToImage_Point [▶ 1108]

TC3 Vision Matching

F_VN_DrawKeypoints [▶ 1017]

F_VN_DrawKeypointsExp [▶ 1018]

F_VN_DrawMatches [▶ 1028]

F_VN_DrawMatchesExp [▶ 1029]

F_VN_FilterGoodMatches [▶ 1325]

F_VN_FindReferenceKeyPointsInImage [▶ 1326]

F_VN_FindReferenceKeyPointsInImageAKAZE [▶ 1327]

F_VN_FindReferenceKeyPointsInImageAKAZEExp [▶ 1329]

F_VN_FindReferenceKeyPointsInImageBRISK [▶ 1331]

F_VN_FindReferenceKeyPointsInImageBRISKExp [▶ 1332]

F_VN_FindReferenceKeyPointsInImageExp [▶ 1334]

F_VN_FindReferenceKeyPointsInImageORB [▶ 1336]

F_VN_FindReferenceKeyPointsInImageORBExp [▶ 1337]

F_VN_FindTemplateWalsh [▶ 1184]

F_VN_FindTemplateWalsh_AngleRange [▶ 1188]

F_VN_FindTemplateWalshExp [▶ 1185]

F_VN_FindTemplateWalshExp_AngleRange [▶ 1191]

F_VN_GetMatchCoordinates [▶ 1339]

F_VN_KeyPointsAGAST [▶ 1340]

F_VN_KeyPointsAGASTExp [▶ 1341]

F_VN_KeyPointsAndDescriptorsAKAZE [▶ 1342]

F_VN_KeyPointsAndDescriptorsAKAZEExp [▶ 1343]

F_VN_KeyPointsAndDescriptorsBRISK [▶ 1344]

F_VN_KeyPointsAndDescriptorsBRISKExp [▶ 1345]

F_VN_KeyPointsAndDescriptorsKAZE [▶ 1346]

F_VN_KeyPointsAndDescriptorsKAZEExp [▶ 1347]

F_VN_KeyPointsAndDescriptorsORB [▶ 1348]

F_VN_KeyPointsAndDescriptorsORBExp [▶ 1349]

F_VN_KeyPointsFAST [▶ 1350]

F_VN_KeyPointsFASTExp [▶ 1351]

F_VN_KeyPointsGFTT [▶ 1352]

F_VN_KeyPointsGFTTExp [▶ 1352]

F_VN_KeyPointsMSER [▶ 1353]

F_VN_KeyPointsMSERExp [▶ 1354]

F_VN_KeyPointsSB [▶ 1355]

F_VN_KeyPointsSBExp [▶ 1356]

F_VN_MatchContours [▶ 988]

F_VN_MatchContours1vsN [▶ 989]

F_VN_MatchContours1vsNExp [▶ 990]

F_VN_MatchContoursExp [▶ 991]

F_VN_MatchDescriptorsBF [▶ 1357]

F_VN_MatchDescriptorsBFExp [▶ 1358]

F_VN_MatchDescriptorsFlannLsh [▶ 1359]

F_VN_MatchDescriptorsFlannLshExp [▶ 1360]

F_VN_MatchDescriptorsKnnBF [▶ 1361]

F_VN_MatchDescriptorsKnnBFExp [▶ 1362]

F_VN_MatchDescriptorsKnnFlannLsh [▶ 1363]

F_VN_MatchDescriptorsKnnFlannLshExp [▶ 1364]

F_VN_MatchImageHuMoments [▶ 1200]

F_VN_MatchTemplate [▶ 1201]

F_VN_MatchTemplateAndEvaluate [▶ 1203]

F_VN_MatchTemplateAndEvaluateExp [▶ 1204]

F_VN_MatchTemplateExp [▶ 1202]

F_VN_MatchTemplateWalsh [▶ 1205]

F_VN_MatchTemplateWalshExp [▶ 1207]

F_VN_RegionsMSER [▶ 1365]

F_VN_RegionsMSERExp [▶ 1366]

TC3 Vision OCR

F_VN_OCR [▶ 1563]

F_VN_OCRExp [▶ 1564]

TC3 Machine Learning Realtime Inference

F_VN_CreateBoostClassifier [▶ 1367]

F_VN_CreateBoostClassifierExp [▶ 1368]

F_VN_CreateBoostClassifierExp2 [▶ 1370]

F_VN_CreateKmppModel [▶ 1372]

F_VN_CreateKmppModelExp [▶ 1374]

F_VN_CreateKnnModel [▶ 1375]

F_VN_CreateLbgModel [▶ 1376]

F_VN_CreateLbgModelExp [▶ 1378]

F_VN_CreateLdaTransform [▶ 1380]

F_VN_CreateLdaTransformViaComponentNum [▶ 1381]

F_VN_CreateNbcModel [▶ 1383]

F_VN_CreatePcaTransform [▶ 1384]

F_VN_CreatePcaTransformViaComponentNum [▶ 1385]

F_VN_CreatePcaTransformViaVariance [▶ 1386]

F_VN_CreateRTreesModel [▶ 1388]

F_VN_CreateRTreesModelExp [▶ 1389]

F_VN_CreateRTreesModelExp2 [▶ 1391]

F_VN_CreateStaModel [▶ 1393]

F_VN_CreateStaModelExp [▶ 1394]

F_VN_CreateStaModelExp2 [▶ 1396]

F_VN_CreateSvmModel [▶ 1398]

F_VN_CreateSvmModelExp [▶ 1400]

F_VN_CreateSvmModelExp2 [▶ 1402]

F_VN_CreateSvmSgdClassifier [▶ 1405]

F_VN_CreateSvmSgdClassifierExp [▶ 1406]

F_VN_FeatureScaling [▶ 1408]

F_VN_FeatureScalingExp [▶ 1410]

F_VN_FeatureTransform [▶ 1411]

F_VN_GetBatchClusters [▶ 1413]

F_VN_GetBatchClustersExp [▶ 1414]

F_VN_GetBatchNovelties [▶ 1415]

F_VN_GetClusterCenter [▶ 1417]

F_VN_GetClusterNum [▶ 1418]

F_VN_GetFeatureScales [▶ 1419]

F_VN_GetSampleCluster [▶ 1421]

F_VN_GetSampleClusterExp [▶ 1422]

F_VN_GetSampleNovelty [▶ 1424]

F_VN_Granulometry [▶ 1567]

F_VN_HaralickFeatures [▶ 1568]

F_VN_InverseFeatureScaling [▶ 1425]

F_VN_InverseFeatureScaling_REAL [▶ 1427]

F_VN_InverseFeatureScalingExp [▶ 1428]

F_VN_InverseFeatureScalingExp_REAL [▶ 1430]

F_VN_InverseFeatureTransform [▶ 1432]

F_VN_PredictBatch [▶ 1433]

F_VN_PredictBatchExp [▶ 1435]

F_VN_PredictSampleClass [▶ 1437]

F_VN_PredictSampleClassExp [▶ 1438]

F_VN_PredictSampleScalar [▶ 1440]

F_VN_PredictSampleScalarExp [▶ 1441]

F_VN_PredictSampleVector [▶ 1443]

F_VN_PredictSampleVectorExp [▶ 1444]

F_VN_TrainBatch [▶ 1446]

F_VN_TrainBatchClusters [▶ 1447]

F_VN_TrainSample [▶ 1449]

F_VN_TrainSampleClass [▶ 1450]

F_VN_TrainSampleCluster [▶ 1451]

F_VN_TrainSampleScalar [▶ 1453]

F_VN_TrainSampleVector [▶ 1454]

TC3 Neural Network Realtime Inference

F_VN_ExecuteNeuralNetwork [▶ 1550]

F_VN_ExecuteNeuralNetwork_MI [▶ 1543]

F_VN_ExecuteNeuralNetwork_MIMO_String [▶ 1544]

F_VN_ExecuteNeuralNetwork_MISO [▶ 1545]

F_VN_ExecuteNeuralNetwork_MO [▶ 1546]

F_VN_ExecuteNeuralNetwork_MultiLayerOutput [▶ 1550]

F_VN_ExecuteNeuralNetwork_SingleLayerOutput [▶ 1551]

F_VN_ExecuteNeuralNetwork_SO [▶ 1546]

F_VN_GetNeuralNetworkLayerNames [▶ 1552]

F_VN_GetNeuralNetworkLayerNamesExp [▶ 1547]

F_VN_GetNeuralNetworkLayerNamesExp_String [▶ 1548]

还请参阅有关此

- F_VN_AngleAndDegreeOfLinearPolarization [▶ 429]
- F_VN_DemosaicPolarizedImage [▶ 431]
- F_VN_ExportImageAsJpeg [▶ 772]
- F_VN_ExportImageAsJpegExp [▶ 773]
- F_VN_FindRois [▶ 1240]
- F_VN_InnerCircle [▶ 1241]
- F_VN_InnerCircleExp [▶ 1242]
- F_VN_MergeRegions [▶ 1069]
- F_VN_PolarizedImageGlareReduction [▶ 433]
- F_VN_SalientFeatures [▶ 1243]
- F_VN_SortContainer [▶ 771]
- F_VN_SortContainerExp [▶ 772]
- F_VN_AnalyzeBarcodeWalsh [▶ 837]
- F_VN_DetectBarcodesWalsh [▶ 838]
- F_VN_LocateAxisAlignedEdges [▶ 1457]
- F_VN_LocateAxisAlignedEdgesExp [▶ 1459]
- F_VN_SortAxisAlignedPatternPoints [▶ 1070]
- F_VN_GetTransformParameters [▶ 1456]

6.1.4.3 容器代数运算

该组包含代数算子以元素方式应用于容器 [▶ 146]的函数。

函数

代数

- F_VN_AddContainers [▶ 493]
- F_VN_AddToContainerElements [▶ 465]
- F_VN_DivideContainers(Exp) [▶ 497]
- F_VN_MultiplyContainers [▶ 500]
- F_VN_MultiplyWithContainerElements [▶ 501]
- F_VN_NegateContainer [▶ 504]
- F_VN_SubtractContainers [▶ 505]

位运算符

- F_VN_BitwiseAndContainers [▶ 493]
- F_VN_BitwiseNotContainer [▶ 494]
- F_VN_BitwiseOrContainers [▶ 495]
- F_VN_BitwiseXorContainers [▶ 496]

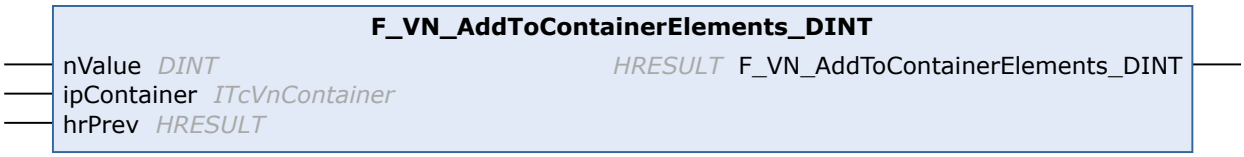
统计学功能

- F_VN_MaxContainer [▶ 499]
- F_VN_MinContainer [▶ 499]

6.1.4.3.1 F_VN_AddToContainerElements

本章包含根据数据类型向每个容器元素添加数值的函数。

6.1.4.3.1.1 F_VN_AddToContainerElements_DINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_DINT : HRESULT
VAR_INPUT
    nValue      : DINT;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nValue	DINT	值
ipContainer	ITcVnContainer [▶ 365]	包含 DINT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

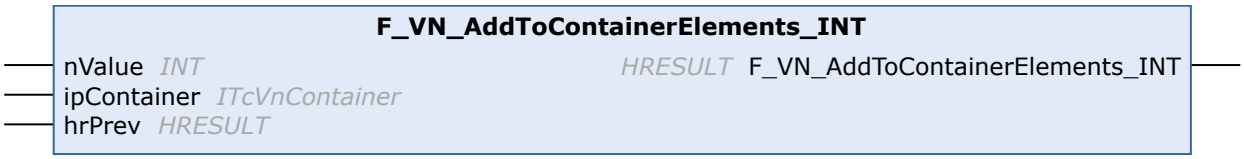
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.2 F_VN_AddToContainerElements_INT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_INT : HRESULT
VAR_INPUT
    nValue      : INT;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nValue	INT	值
ipContainer	ITcVnContainer [▶ 365]	包含 INT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

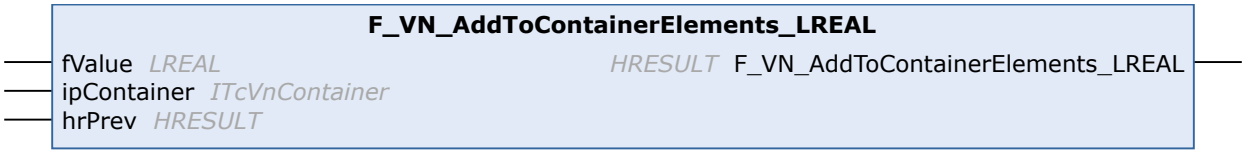
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.3 F_VN_AddToContainerElements_LREAL



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_LREAL : HRESULT
VAR_INPUT
    fValue      : LREAL;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fValue	LREAL	值
ipContainer	ITcVnContainer [▶ 365]	包含 LREAL 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

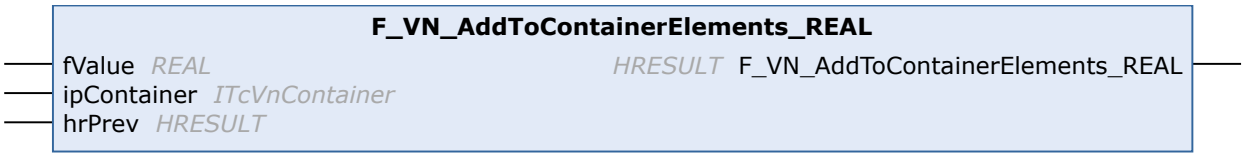
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.4 F_VN_AddToContainerElements_REAL



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_REAL : HRESULT
VAR_INPUT
    fValue      : REAL;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fValue	REAL	值
ipContainer	ITcVnContainer [▶ 365]	包含 REAL 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

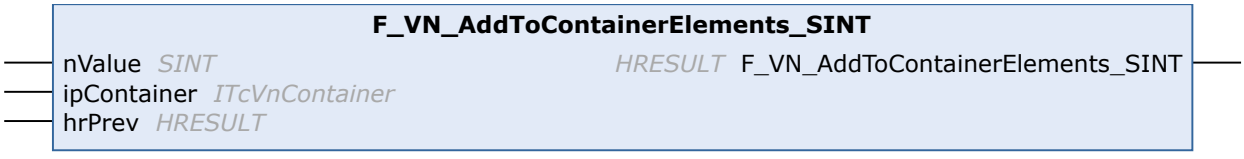
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.5 F_VN_AddToContainerElements_SINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_SINT : HRESULT
VAR_INPUT
    nValue      : SINT;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nValue	SINT	值
ipContainer	ITcVnContainer [▶ 365]	装有 SINT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

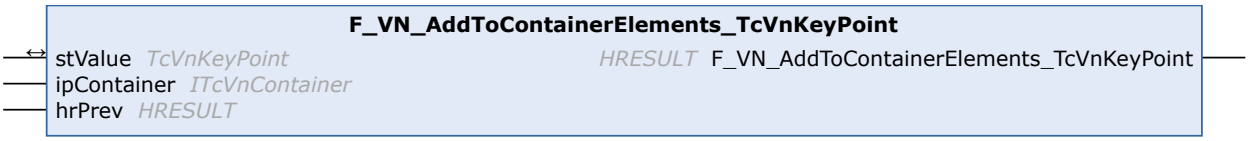
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.6 F_VN_AddToContainerElements_TcVnKeyPoint



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnKeyPoint : HRESULT
VAR_IN_OUT
    stValue     : TcVnKeyPoint;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnKeyPoint 元素的容器

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stValue	TcVnKeyPoint [▶ 226]	值

返回值

HRESULT [▶ 135]

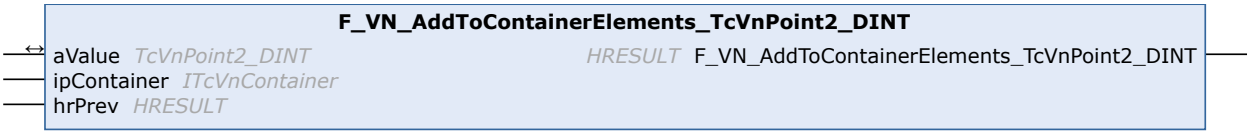
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.7 F_VN_AddToContainerElements_TcVnPoint2_DINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnPoint2_DINT : HRESULT
VAR_IN_OUT
    aValue       : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnPoint2_DINT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnPoint2_DINT [▶ 152]	值

返回值

HRESULT [▶ 135]

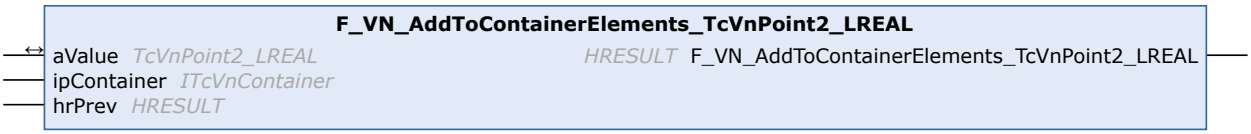
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.8 F_VN_AddToContainerElements_TcVnPoint2_LREAL



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnPoint2_LREAL : HRESULT
VAR_IN_OUT
    aValue       : TcVnPoint2_LREAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnPoint2_LREAL 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnPoint2_LREAL [▶ 152]	值

返回值

HRESULT [▶ 135]

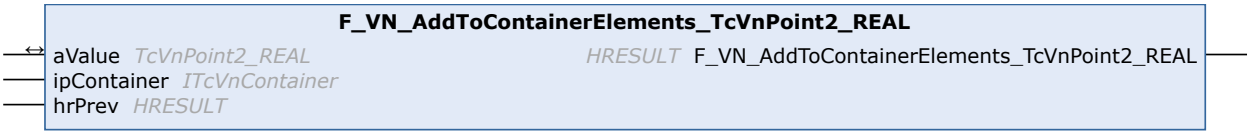
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.9 F_VN_AddToContainerElements_TcVnPoint2_REAL



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnPoint2_REAL : HRESULT
VAR_IN_OUT
    aValue      : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnPoint2_REAL 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnPoint2_REAL [▶ 152]	值

返回值

HRESULT [▶ 135]

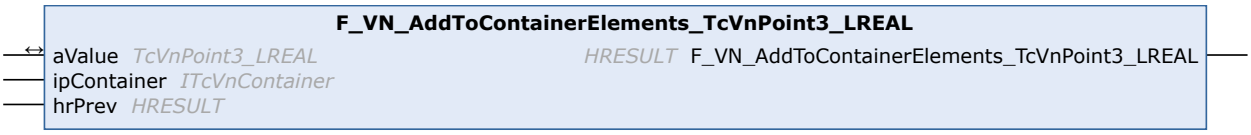
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.10 F_VN_AddToContainerElements_TcVnPoint3_LREAL



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnPoint3_LREAL : HRESULT
VAR_IN_OUT
    aValue      : TcVnPoint3_LREAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnPoint3_LREAL 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnPoint3_LREAL [▶ 152]	值

返回值

HRESULT [▶ 135]

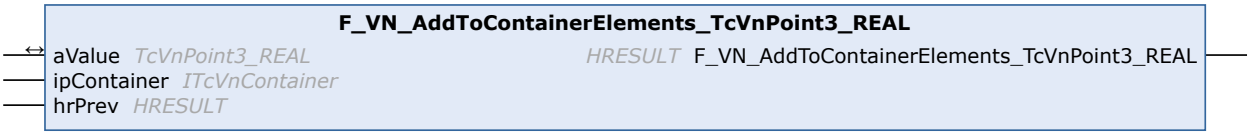
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.11 F_VN_AddToContainerElements_TcVnPoint3_REAL



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnPoint3_REAL : HRESULT
VAR_IN_OUT
    aValue      : TcVnPoint3_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnPoint3_REAL 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnPoint3_REAL [▶ 152]	值

返回值

HRESULT [▶ 135]

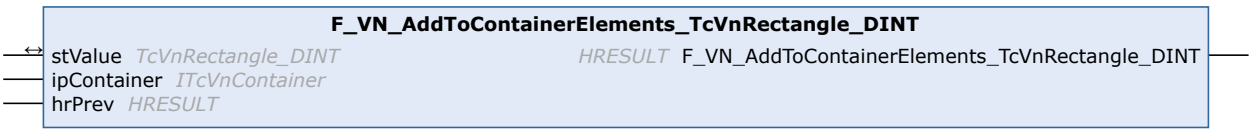
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.12 F_VN_AddToContainerElements_TcVnRectangle_DINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnRectangle_DINT : HRESULT
VAR_IN_OUT
    stValue     : TcVnRectangle_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnRectangle_DINT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stValue	TcVnRectangle_DINT [▶ 236]	值

返回值

HRESULT [▶ 135]

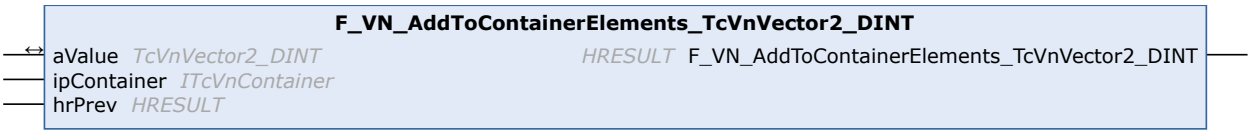
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.13 F_VN_AddToContainerElements_TcVnVector2_DINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector2_DINT : HRESULT
VAR_IN_OUT
    aValue      : TcVnVector2_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_DINT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_DINT [▶ 153]	值

返回值

HRESULT [▶ 135]

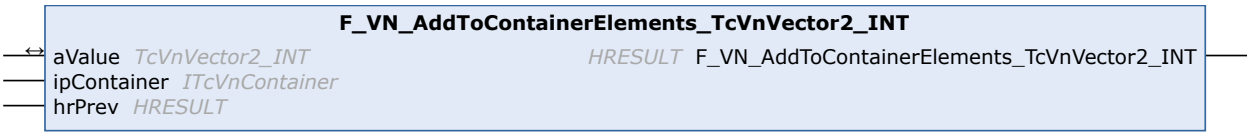
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.14 F_VN_AddToContainerElements_TcVnVector2_INT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector2_INT : HRESULT
VAR_IN_OUT
    aValue       : TcVnVector2_INT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_INT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_INT [▶ 153]	值

返回值

HRESULT [▶ 135]

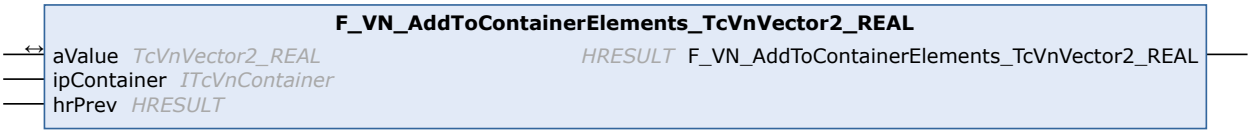
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.15 F_VN_AddToContainerElements_TcVnVector2_REAL



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector2_REAL : HRESULT
VAR_IN_OUT
    aValue      : TcVnVector2_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_REAL 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_REAL [▶ 153]	值

返回值

HRESULT [▶ 135]

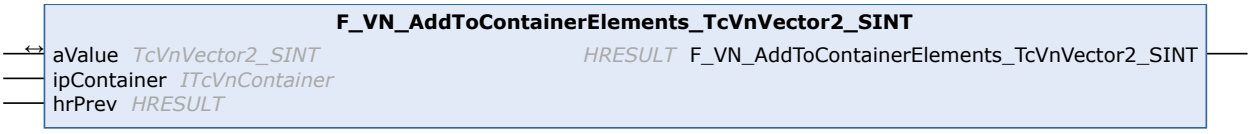
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.16 F_VN_AddToContainerElements_TcVnVector2_SINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector2_SINT : HRESULT
VAR_IN_OUT
    aValue       : TcVnVector2_SINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_SINT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_SINT [▶ 153]	值

返回值

HRESULT [▶ 135]

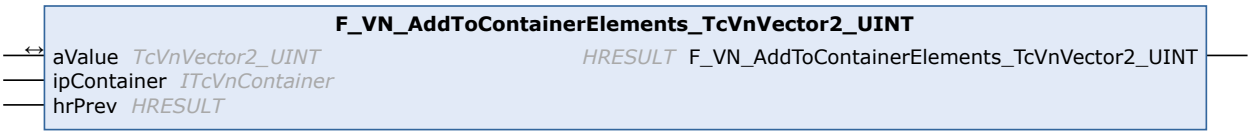
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.17 F_VN_AddToContainerElements_TcVnVector2_UINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector2_UINT : HRESULT
VAR_IN_OUT
    aValue       : TcVnVector2_UINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_UINT 元素的容器

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_UINT [▶ 153]	值

返回值

HRESULT [▶ 135]

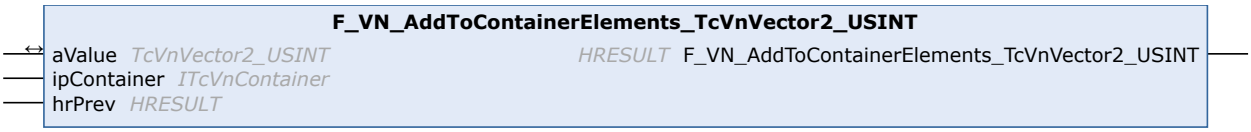
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.18 F_VN_AddToContainerElements_TcVnVector2_USINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector2_USINT : HRESULT
VAR_IN_OUT
    aValue      : TcVnVector2_USINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_USINT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_USINT [▶ 153]	值

返回值

HRESULT [▶ 135]

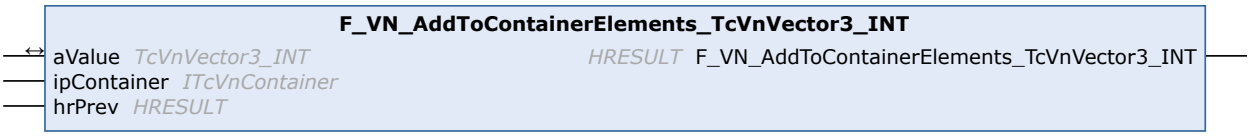
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.19 F_VN_AddToContainerElements_TcVnVector3_INT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector3_INT : HRESULT
VAR_IN_OUT
    aValue       : TcVnVector3_INT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector3_INT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_INT [▶ 153]	值

返回值

HRESULT [▶ 135]

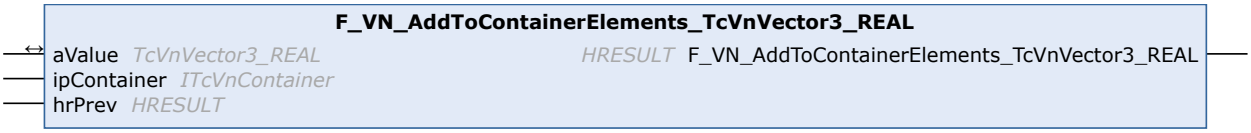
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.20 F_VN_AddToContainerElements_TcVnVector3_REAL



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector3_REAL : HRESULT
VAR_IN_OUT
    aValue      : TcVnVector3_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector3_REAL 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_REAL [▶ 153]	值

返回值

HRESULT [▶ 135]

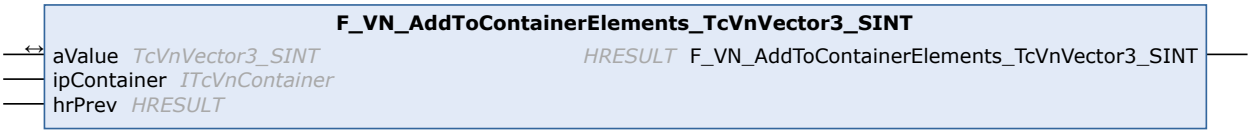
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.21 F_VN_AddToContainerElements_TcVnVector3_SINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector3_SINT : HRESULT
VAR_IN_OUT
    aValue      : TcVnVector3_SINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector3_SINT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_SINT [▶ 153]	值

返回值

HRESULT [▶ 135]

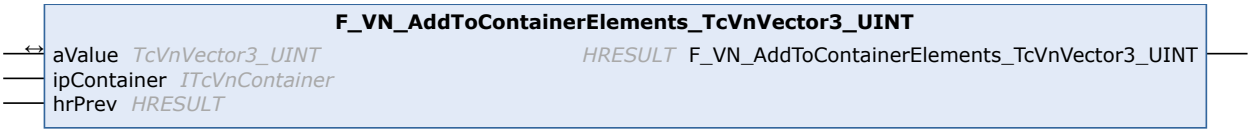
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.22 F_VN_AddToContainerElements_TcVnVector3_UINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector3_UINT : HRESULT
VAR_IN_OUT
    aValue      : TcVnVector3_UINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector3_UINT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_UINT [▶ 153]	值

返回值

HRESULT [▶ 135]

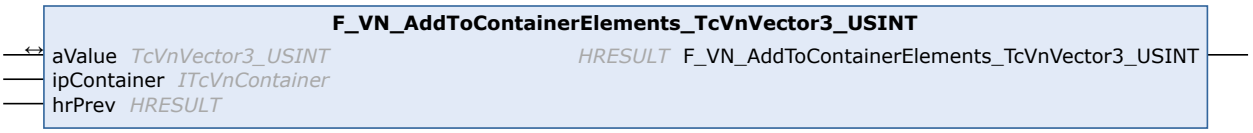
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.23 F_VN_AddToContainerElements_TcVnVector3_USINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector3_USINT : HRESULT
VAR_IN_OUT
    aValue      : TcVnVector3_USINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector3_USINT 元素的容器

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_USINT [▶ 153]	值

返回值

HRESULT [▶ 135]

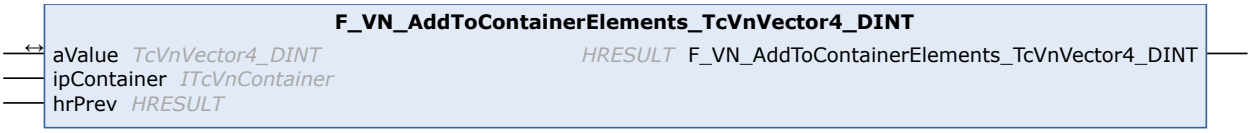
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.24 F_VN_AddToContainerElements_TcVnVector4_DINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector4_DINT : HRESULT
VAR_IN_OUT
    aValue      : TcVnVector4_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_DINT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_DINT [▶ 153]	值

返回值

HRESULT [▶ 135]

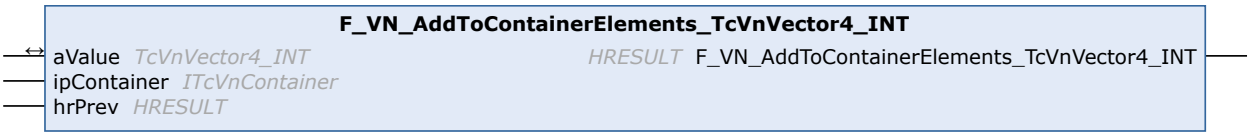
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.25 F_VN_AddToContainerElements_TcVnVector4_INT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector4_INT : HRESULT
VAR_IN_OUT
    aValue       : TcVnVector4_INT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_INT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_INT [▶ 153]	值

返回值

HRESULT [▶ 135]

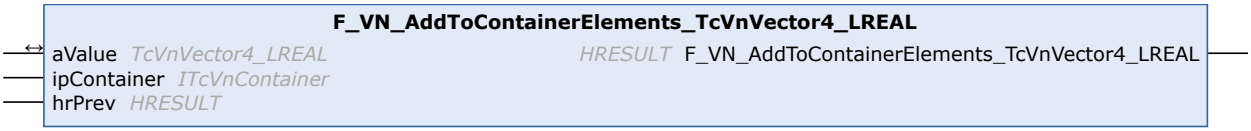
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.26 F_VN_AddToContainerElements_TcVnVector4_LREAL



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector4_LREAL : HRESULT
VAR_IN_OUT
    aValue      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_LREAL 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_LREAL [▶ 153]	值

返回值

HRESULT [▶ 135]

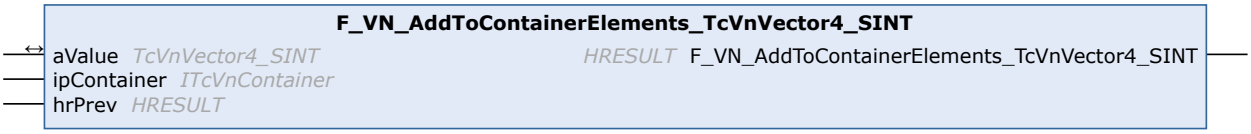
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.27 F_VN_AddToContainerElements_TcVnVector4_SINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector4_SINT : HRESULT
VAR_IN_OUT
    aValue      : TcVnVector4_SINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_SINT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_SINT [▶ 153]	值

返回值

HRESULT [▶ 135]

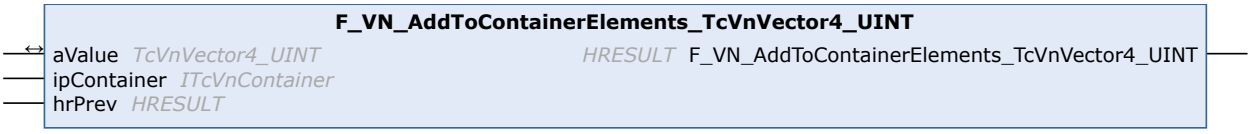
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.28 F_VN_AddToContainerElements_TcVnVector4_UINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector4_UINT : HRESULT
VAR_IN_OUT
    aValue      : TcVnVector4_UINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_UINT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_UINT [▶ 153]	值

返回值

HRESULT [▶ 135]

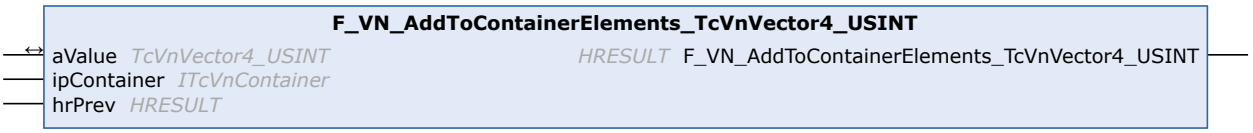
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.29 F_VN_AddToContainerElements_TcVnVector4_USINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector4_USINT : HRESULT
VAR_IN_OUT
    aValue      : TcVnVector4_USINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_USINT 元素的容器

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_USINT [▶ 153]	值

返回值

HRESULT [▶ 135]

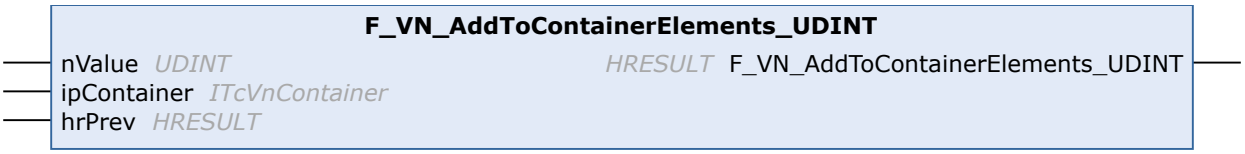
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.30 F_VN_AddToContainerElements_UDINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_UDINT : HRESULT
VAR_INPUT
    nValue      : UDINT;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nValue	UDINT	值
ipContainer	ITcVnContainer [▶ 365]	包含 UDINT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

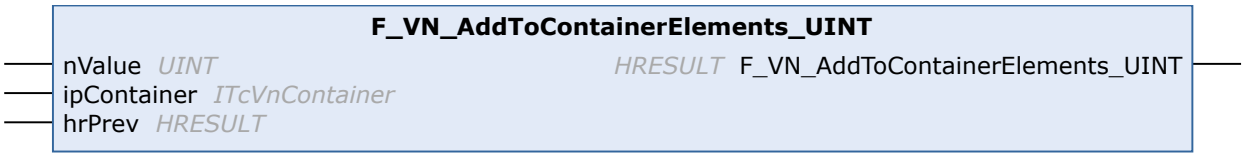
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.31 F_VN_AddToContainerElements_UINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_UINT : HRESULT
VAR_INPUT
    nValue       : UINT;
    ipContainer : ITcVnContainer;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
nValue	UINT	值
ipContainer	ITcVnContainer [▶ 365]	包含 UINT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

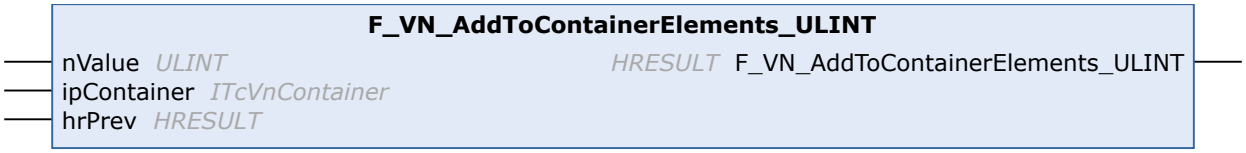
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.32 F_VN_AddToContainerElements_ULINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_ULINT : HRESULT
VAR_INPUT
    nValue      : ULINT;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nValue	ULINT	值
ipContainer	ITcVnContainer [▶ 365]	包含 ULINT 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

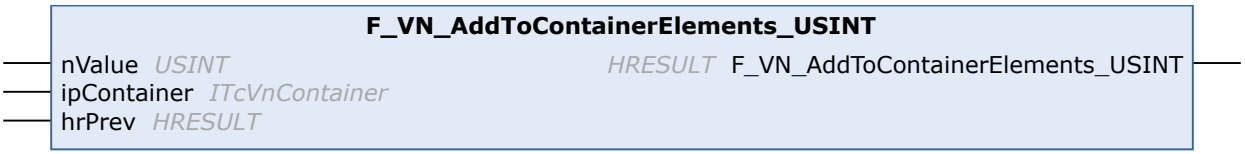
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.33 F_VN_AddToContainerElements_USINT



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_USINT : HRESULT
VAR_INPUT
    nValue      : USINT;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nValue	USINT	值
ipContainer	ITcVnContainer [▶ 365]	包含 USINT 元素的容器

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

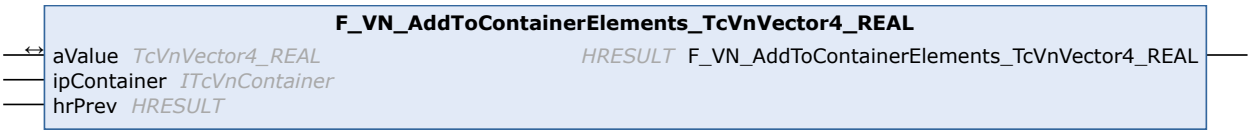
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.34 F_VN_AddToContainerElements_TcVnVector4_REAL



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnVector4_REAL : HRESULT
VAR_IN_OUT
    aValue       : TcVnVector4_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_REAL 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_REAL [▶ 153]	值

返回值

HRESULT [▶ 135]

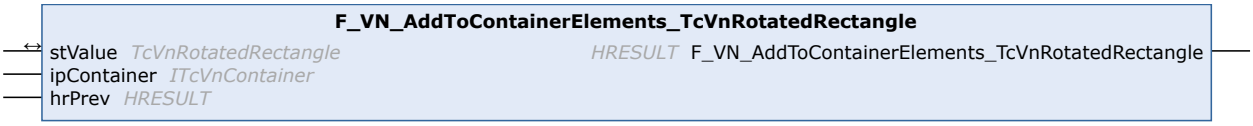
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.1.35 F_VN_AddToContainerElements_TcVnRotatedRectangle



将每个容器元素与一个值相加。

语法

定义：

```
FUNCTION F_VN_AddToContainerElements_TcVnRotatedRectangle : HRESULT
VAR_IN_OUT
    stValue     : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnRotatedRectangle 元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stValue	TcVnRotatedRectangle [▶ 237]	值

返回值

HRESULT [▶ 135]

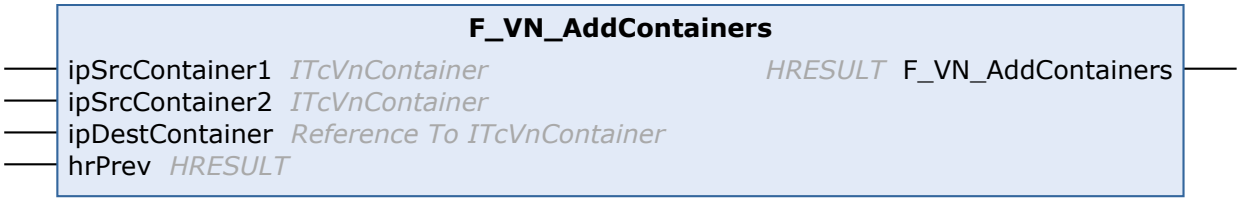
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.2 F_VN_AddContainers



Element-wise addition of two containers (same length and type).

语法

定义：

```
FUNCTION F_VN_AddContainers : HRESULT
VAR_INPUT
    ipSrcContainer1 : ITcVnContainer;
    ipSrcContainer2 : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer1	ITcVnContainer [▶ 365]	源容器 1
ipSrcContainer2	ITcVnContainer [▶ 365]	源容器 2
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回生成的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.3 F_VN_BitwiseAndContainers



Element-wise application of a bit-wise AND operator to two containers (same length and type, integer only).

语法

定义：

```
FUNCTION F_VN_BitwiseAndContainers : HRESULT
VAR_INPUT
    ipSrcContainer1 : ITcVnContainer;
    ipSrcContainer2 : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer1	ITcVnContainer [▶ 365]	源容器 1
ipSrcContainer2	ITcVnContainer [▶ 365]	源容器 2
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回生成的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

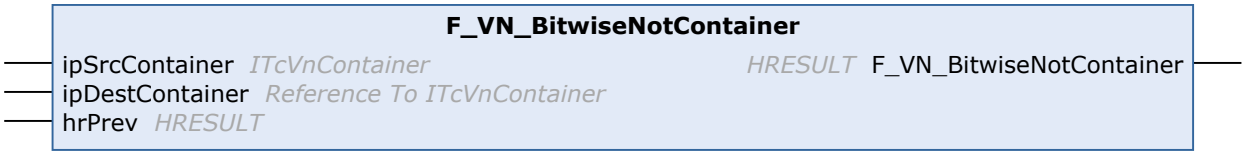
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.4 F_VN_BitwiseNotContainer



对一个容器（仅限整数）逐元素应用逐位 NOT 算子。

语法

定义：

```
FUNCTION F_VN_BitwiseNotContainer : HRESULT
VAR_INPUT
    ipSrcContainer  : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	源容器
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回生成的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.5 F_VN_BitwiseOrContainers



对两个容器（长度和类型相同，仅限整数）逐元素应用逐位 OR 算子。

语法

定义：

```
FUNCTION F_VN_BitwiseOrContainers : HRESULT
VAR_INPUT
    ipSrcContainer1 : ITcVnContainer;
    ipSrcContainer2 : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer1	ITcVnContainer [▶ 365]	源容器 1
ipSrcContainer2	ITcVnContainer [▶ 365]	源容器 2
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回生成的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

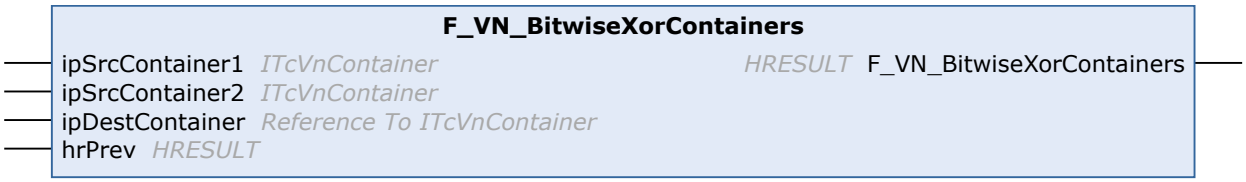
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.6 F_VN_BitwiseXorContainers



对两个容器（长度和类型相同，仅限整数）逐元素应用逐位 XOR 算子。

语法

定义：

```
FUNCTION F_VN_BitwiseXorContainers : HRESULT
VAR_INPUT
    ipSrcContainer1 : ITcVnContainer;
    ipSrcContainer2 : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer1	ITcVnContainer [▶ 365]	源容器 1
ipSrcContainer2	ITcVnContainer [▶ 365]	源容器 2
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回生成的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

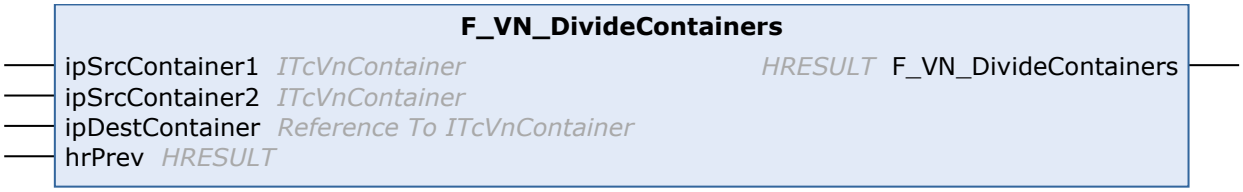
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.7 F_VN_DivideContainers



两个容器逐元素相除（长度和类型相同）。在整数除以 0 的情况下，结果元素将被设置为 0，并返回 S_DIVISION_BY_ZERO，而不是 S_OK（可在 F_VN_DivideContainersExp 中更改值）。

语法

定义：

```
FUNCTION F_VN_DivideContainers : HRESULT
VAR_INPUT
    ipSrcContainer1 : ITcVnContainer;
    ipSrcContainer2 : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer1	ITcVnContainer [▶ 365]	源容器 1
ipSrcContainer2	ITcVnContainer [▶ 365]	源容器 2
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回生成的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

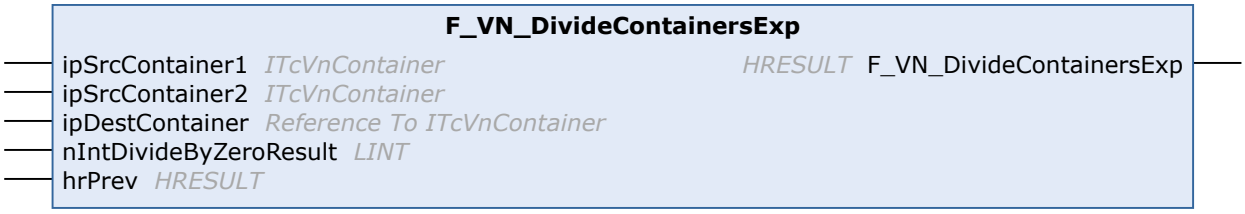
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.8 F_VN_DivideContainersExp



两个容器逐元素相除（长度和类型相同）。在整数除以 0 的情况下，结果元素将被设置为 nIntDivideByZeroResult，并返回 S_DIVISION_BY_ZERO，而不是 S_OK。（专家函数）

语法

定义：

```
FUNCTION F_VN_DivideContainersExp : HRESULT
VAR_INPUT
    ipSrcContainer1         : ITcVnContainer;
    ipSrcContainer2         : ITcVnContainer;
    ipDestContainer         : Reference To ITcVnContainer;
    nIntDivideByZeroResult : LINT;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer1	ITcVnContainer [▶ 365]	源容器 1
ipSrcContainer2	ITcVnContainer [▶ 365]	源容器 2
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回生成的容器
nIntDivideByZeroResult	LINT	整数除以零时设为结果的值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

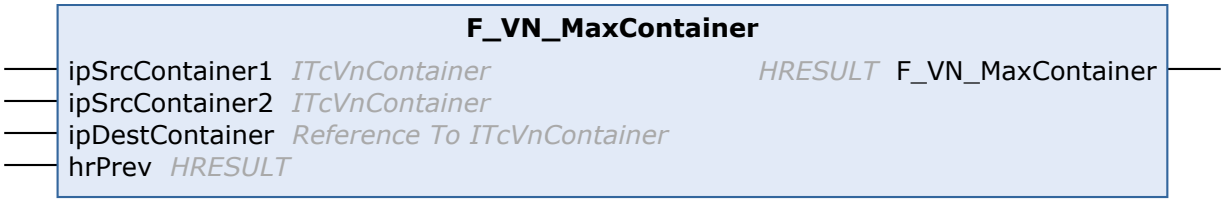
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.9 F_VN_MaxContainer



两个容器（长度和类型相同）的逐元素最大值。

语法

定义：

```
FUNCTION F_VN_MaxContainer : HRESULT
VAR_INPUT
    ipSrcContainer1 : ITcVnContainer;
    ipSrcContainer2 : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer1	ITcVnContainer [▶ 365]	源容器 1
ipSrcContainer2	ITcVnContainer [▶ 365]	源容器 2
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回生成的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.10 F_VN_MinContainer



两个容器（长度和类型相同）的逐元素最小值。

语法

定义：

```
FUNCTION F_VN_MinContainer : HRESULT
VAR_INPUT
    ipSrcContainer1 : ITcVnContainer;
    ipSrcContainer2 : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer1	ITcVnContainer [▶ 365]	源容器 1
ipSrcContainer2	ITcVnContainer [▶ 365]	源容器 2
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回生成的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.11 F_VN_MultiplyContainers



两个容器（长度和类型相同）逐元素相乘。

语法

定义：

```
FUNCTION F_VN_MultiplyContainers : HRESULT
VAR_INPUT
    ipSrcContainer1 : ITcVnContainer;
    ipSrcContainer2 : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer1	ITcVnContainer [▶ 365]	源容器 1
ipSrcContainer2	ITcVnContainer [▶ 365]	源容器 2
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回生成的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

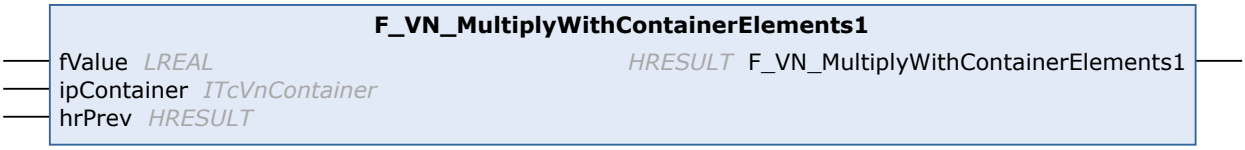
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.12 F_VN_MultiplyWithContainerElements1



将每个容器元素乘以一个值。

语法

定义：

```
FUNCTION F_VN_MultiplyWithContainerElements1 : HRESULT
VAR_INPUT
    fValue      : LREAL;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fValue	LREAL	值
ipContainer	ITcVnContainer [▶ 365]	包含一维元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

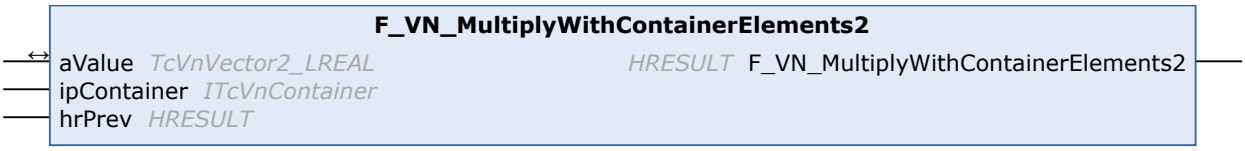
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.13 F_VN_MultiplyWithContainerElements2



将每个容器元素乘以一个值。

语法

定义：

```
FUNCTION F_VN_MultiplyWithContainerElements2 : HRESULT
VAR_IN_OUT
    aValue       : TcVnVector2_LREAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含二维元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_LREAL [▶ 153]	值

返回值

HRESULT [▶ 135]

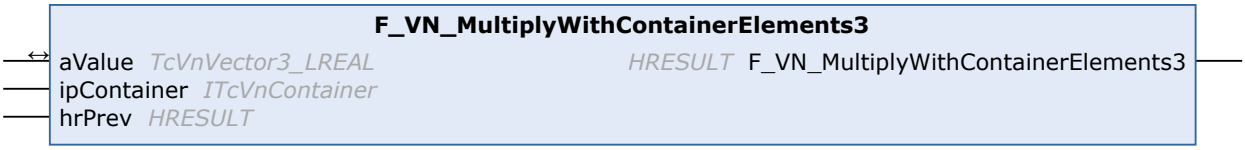
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.14 F_VN_MultiplyWithContainerElements3



将每个容器元素乘以一个值。

语法

定义：

```
FUNCTION F_VN_MultiplyWithContainerElements3 : HRESULT
VAR_IN_OUT
    aValue      : TcVnVector3_LREAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	带有三维元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_LREAL [▶ 153]	值

返回值

HRESULT [▶ 135]

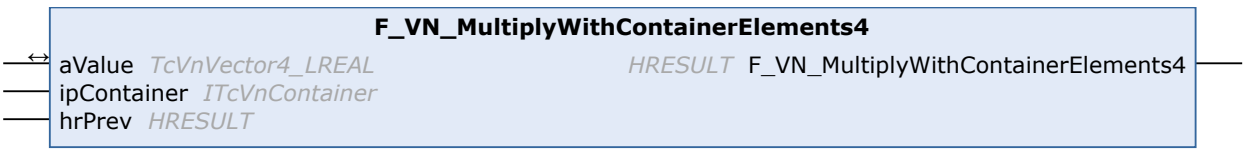
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.15 F_VN_MultiplyWithContainerElements4



将每个容器元素乘以一个值。

语法

定义：

```
FUNCTION F_VN_MultiplyWithContainerElements4 : HRESULT
VAR_IN_OUT
    aValue      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含四维元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_LREAL [▶ 153]	值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.16 F_VN_NegateContainer



对一个容器逐元素取反（二进制）。

语法

定义：

```
FUNCTION F_VN_NegateContainer : HRESULT
VAR_INPUT
    ipSrcContainer  : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	源容器
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回生成的容器（与 ipSrcContainer 类型相同）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

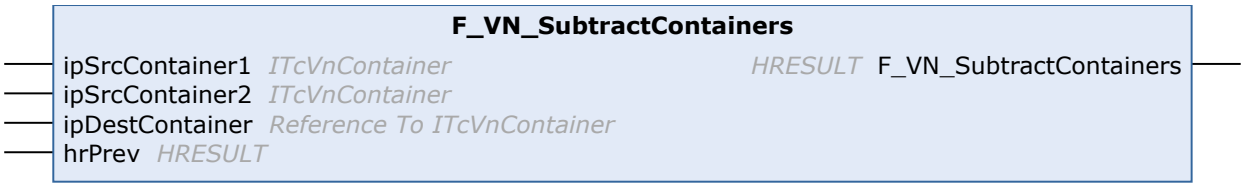
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.3.17 F_VN_SubtractContainers



两个容器（长度和类型相同）逐元素相减。

语法

定义：

```
FUNCTION F_VN_SubtractContainers : HRESULT
VAR_INPUT
    ipSrcContainer1 : ITcVnContainer;
    ipSrcContainer2 : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer1	ITcVnContainer [▶ 365]	源容器 1
ipSrcContainer2	ITcVnContainer [▶ 365]	源容器 2
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回生成的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4 代数图像运算

6.1.4.4.1 F_VN_AddImages



Element-wise addition of two images using saturation arithmetics.

语法

定义：

```
FUNCTION F_VN_AddImages : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

样本

- 复制图像区域 [▶ 2726]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.2 F_VN_AddImagesWeighted



Weighted, element-wise addition of two images using saturation arithmetics.

语法

定义：

```
FUNCTION F_VN_AddImagesWeighted : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    fWeight1    : LREAL;
    fWeight2    : LREAL;
    fDelta      : LREAL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
fWeight1	LREAL	应用于 ipSrcImage1 的权重系数
fWeight2	LREAL	应用于 ipSrcImage2 的权重系数
fDelta	LREAL	与两个图像的加权总和相加的值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

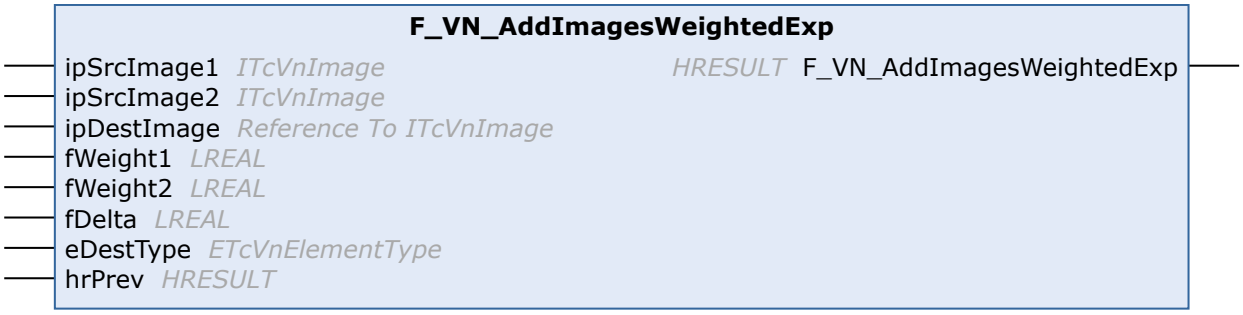
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.3 F_VN_AddImagesWeightedExp



Weighted, element-wise addition of two images using saturation arithmetics.

语法

定义：

```
FUNCTION F_VN_AddImagesWeightedExp : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    fWeight1    : LREAL;
    fWeight2    : LREAL;
    fDelta      : LREAL;
    eDestType   : ETcVnElementType;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
fWeight1	LREAL	应用于 ipSrcImage1 的权重系数
fWeight2	LREAL	应用于 ipSrcImage2 的权重系数
fDelta	LREAL	与两个图像的加权总和相加的值
eDestType	ETcVnElementType [▶ 191]	目标图像深度
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

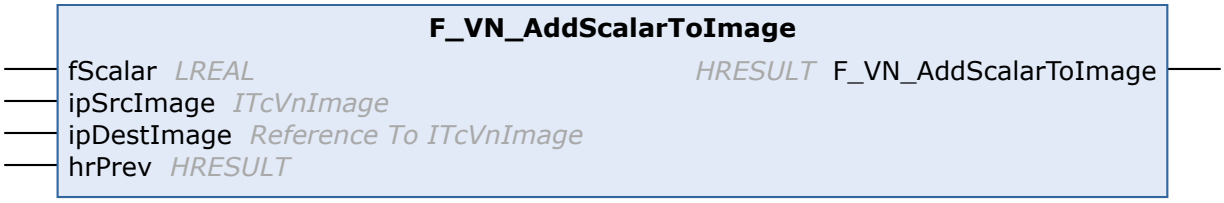
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.4 F_VN_AddScalarToImage



Add a scalar value to each image pixel using saturation arithmetics.

语法

定义：

```
FUNCTION F_VN_AddScalarToImage : HRESULT
VAR_INPUT
    fScalar     : LREAL;
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fScalar	LREAL	标量值
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

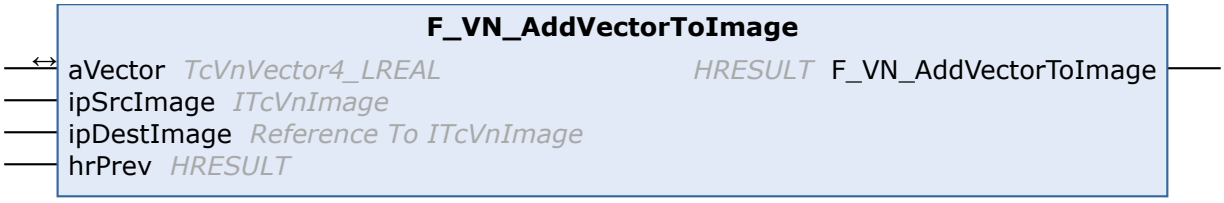
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.5 F_VN_AddVectorToImage



Add a vector (1 element for each image channel) to each image pixel using saturation arithmetics.

语法

定义：

```
FUNCTION F_VN_AddVectorToImage : HRESULT
VAR_IN_OUT
    aVector     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aVector	TcVnVector4_LREAL [▶ 153]	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。

返回值

HRESULT [▶ 135]

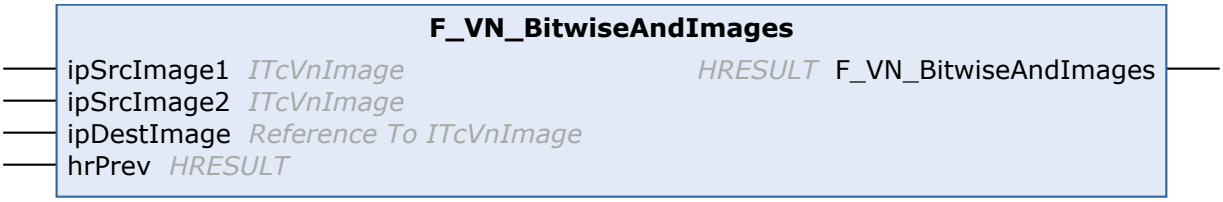
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.6 F_VN_BitwiseAndImages



Element-wise application of a bit-wise AND operator to two images.

语法

定义：

```
FUNCTION F_VN_BitwiseAndImages : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

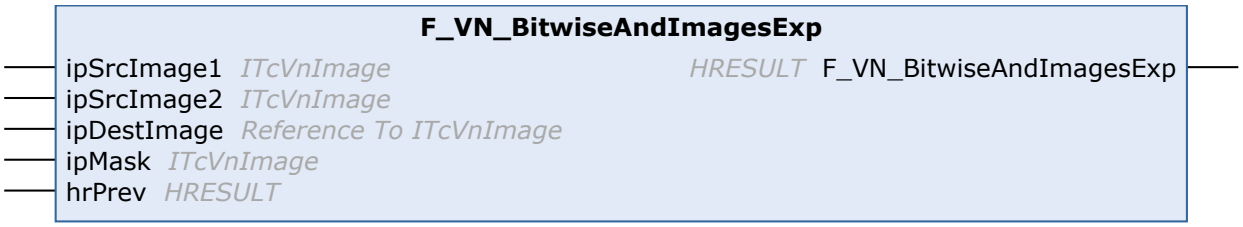
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.7 F_VN_BitwiseAndImagesExp



Element-wise application of a bit-wise AND operator to two images. (expert function)

语法

定义：

```
FUNCTION F_VN_BitwiseAndImagesExp : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    ipMask      : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage [▶ 407]	掩膜用于将运算限制在特定像素位置（掩膜值 0：跳过像素，掩码值 > 0：对像素执行运算）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

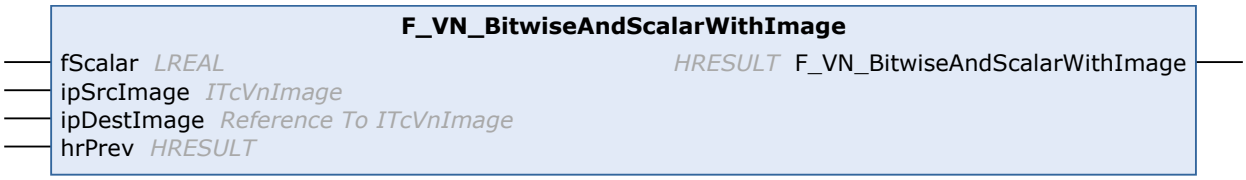
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.8 F_VN_BitwiseAndScalarWithImage



Bitwise and a scalar value with each image pixel.

语法

定义：

```
FUNCTION F_VN_BitwiseAndScalarWithImage : HRESULT
VAR_INPUT
    fScalar     : LREAL;
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fScalar	LREAL	标量值
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

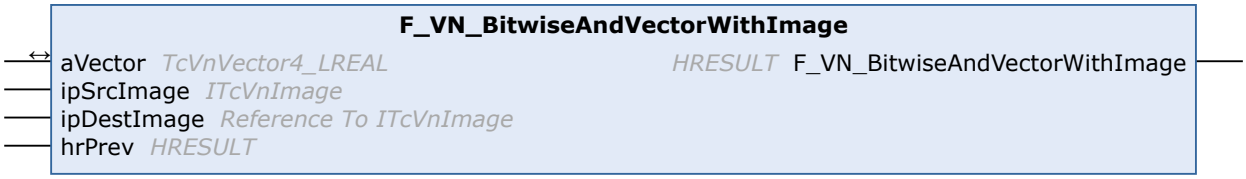
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.9 F_VN_BitwiseAndVectorWithImage



Bitwise and a vector (1 element for each image channel) with each image pixel.

语法

定义：

```
FUNCTION F_VN_BitwiseAndVectorWithImage : HRESULT
VAR_IN_OUT
    aVector     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aVector	TcVnVector4_LREAL [▶ 153]	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。

返回值

HRESULT [▶ 135]

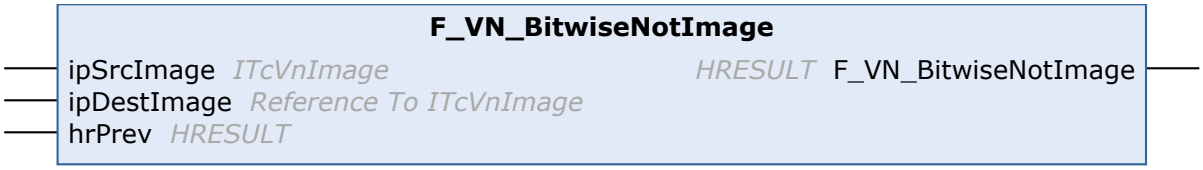
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.10 F_VN_BitwiseNotImage



Element-wise application of a bit-wise NOT operator to an image.

语法

定义：

```
FUNCTION F_VN_BitwiseNotImage : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

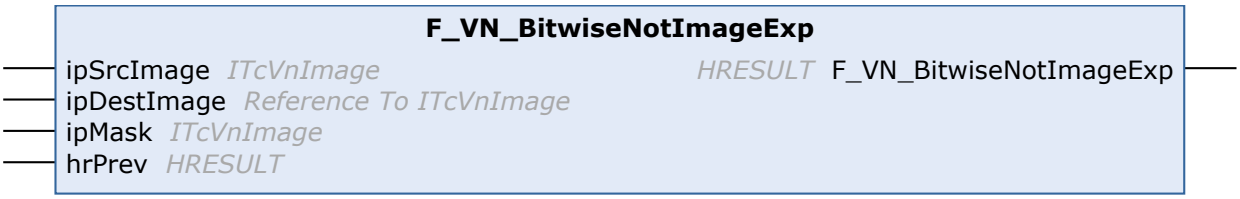
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.11 F_VN_BitwiseNotImageExp



Element-wise application of a bit-wise NOT operator to an image. (expert function)

语法

定义：

```
FUNCTION F_VN_BitwiseNotImageExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    ipMask      : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像

名称	类型	描述
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage [▶ 407]	掩膜用于将运算限制在特定像素位置（掩膜值 0：跳过像素，掩码值 > 0：对像素执行运算）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

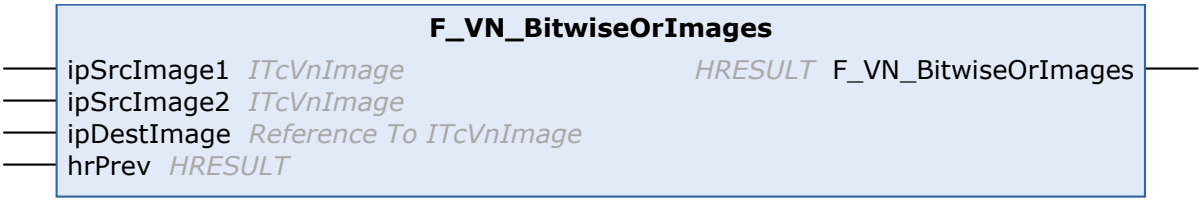
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.12 F_VN_BitwiseOrImages



Element-wise application of a bit-wise OR operator to two images.

语法

定义：

```
FUNCTION F_VN_BitwiseOrImages : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

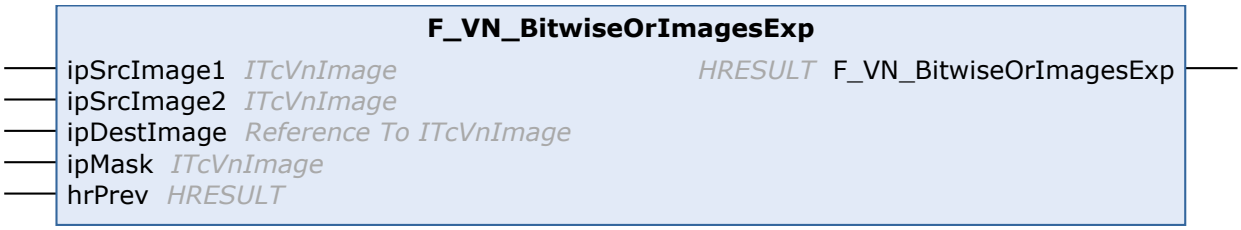
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.13 F_VN_BitwiseOrImagesExp



Element-wise application of a bit-wise OR operator to two images. (expert function)

语法

定义：

```
FUNCTION F_VN_BitwiseOrImagesExp : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    ipMask      : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage [▶ 407]	掩膜用于将运算限制在特定像素位置（掩膜值 0：跳过像素，掩码值 > 0：对像素执行运算）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

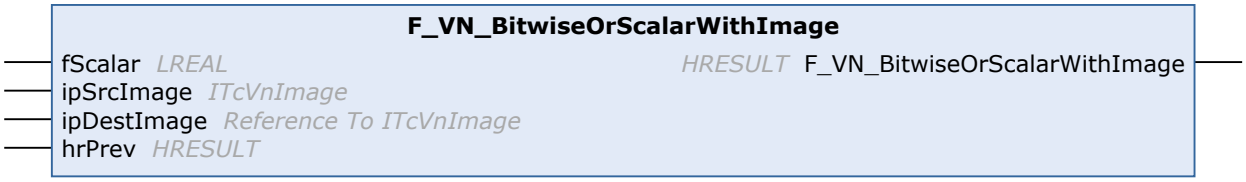
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.14 F_VN_BitwiseOrScalarWithImage



Bitwise or a scalar value with each image pixel.

语法

定义：

```
FUNCTION F_VN_BitwiseOrScalarWithImage : HRESULT
VAR_INPUT
    fScalar     : LREAL;
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fScalar	LREAL	标量值
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

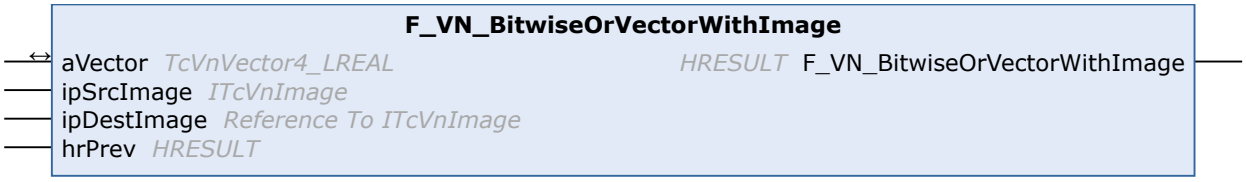
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.15 F_VN_BitwiseOrVectorWithImage



Bitwise or a vector (1 element for each image channel) with each image pixel.

语法

定义：

```
FUNCTION F_VN_BitwiseOrVectorWithImage : HRESULT
VAR_IN_OUT
    aVector     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aVector	TcVnVector4_LREAL [▶ 153]	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。

返回值

HRESULT [▶ 135]

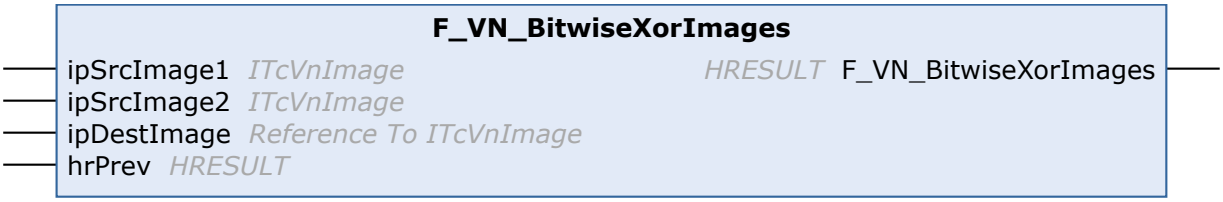
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.16 F_VN_BitwiseXorImages



Element-wise application of a bit-wise XOR operator to two images.

语法

定义：

```
FUNCTION F_VN_BitwiseXorImages : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

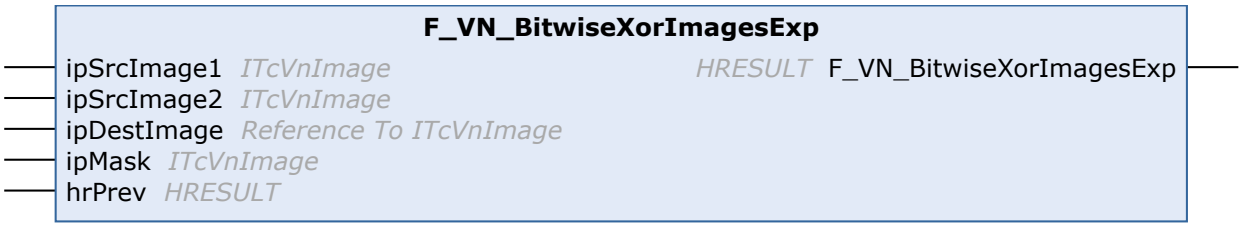
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.17 F_VN_BitwiseXorImagesExp



Element-wise application of a bit-wise XOR operator to two images. (expert function)

语法

定义：

```
FUNCTION F_VN_BitwiseXorImagesExp : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    ipMask      : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage [▶ 407]	掩膜用于将运算限制在特定像素位置（掩膜值 0：跳过像素，掩码值 > 0：对像素执行运算）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

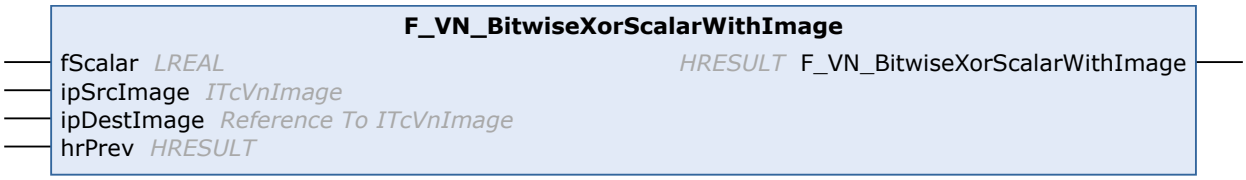
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.18 F_VN_BitwiseXorScalarWithImage



Bitwise xor a scalar value with each image pixel.

语法

定义：

```
FUNCTION F_VN_BitwiseXorScalarWithImage : HRESULT
VAR_INPUT
    fScalar     : LREAL;
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fScalar	LREAL	标量值
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

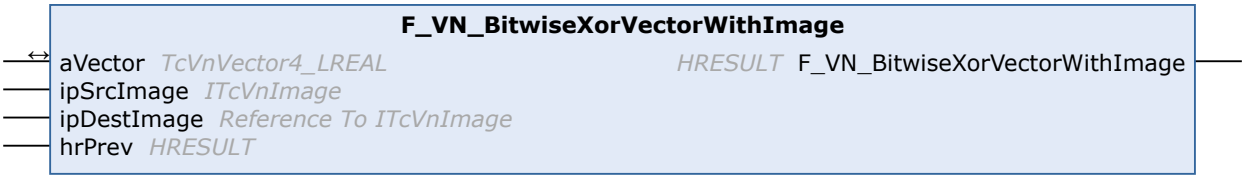
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.19 F_VN_BitwiseXorVectorWithImage



Bitwise xor a vector (1 element for each image channel) with each image pixel.

语法

定义：

```
FUNCTION F_VN_BitwiseXorVectorWithImage : HRESULT
VAR_IN_OUT
    aVector     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aVector	TcVnVector4_LREAL [▶ 153]	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.20 F_VN_BlendImages



Blends two images, i.e. dest = (src1 * weight1 + src2 * weight2) / (weight1 + weight2).
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_BlendImages : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipWeights1  : ITcVnImage;
    ipWeights2  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像（ET_USINT 或 ET_REAL）
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像（类型和大小与 ipSrcImage1 相同）
ipWeights1	ITcVnImage [▶ 407]	ipSrcImage1 的权重（1 通道，ET_REAL）
ipWeights2	ITcVnImage [▶ 407]	ipSrcImage2 的权重（1 通道，ET_REAL）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

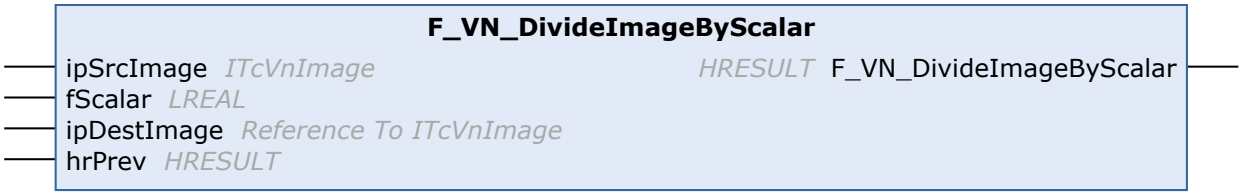
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.21 F_VN_DivideImageByScalar



Divide each image pixel by a scalar value using saturation arithmetics.

语法

定义:

```
FUNCTION F_VN_DivideImageByScalar : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    fScalar     : LREAL;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
fScalar	LREAL	标量值
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

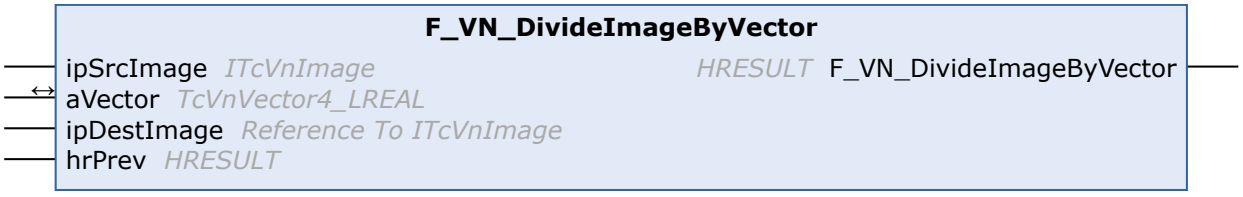
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.22 F_VN_DivideImageByVector



Divides each image pixel by a vector (1 element for each image channel) using saturation arithmetics.

语法

定义:

```
FUNCTION F_VN_DivideImageByVector : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
END_VAR
VAR_IN_OUT
    aVector     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aVector	TcVnVector4_LREAL [▶ 153]	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.23 F_VN_DivideImages



Element-wise division of two images using saturation arithmetics. (A division by zero equals zero.)

语法

定义：

```
FUNCTION F_VN_DivideImages : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一张源图像（被除数）
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像（除数）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

在除以零的情况下，函数F_VN_DivideImages针对各元素返回零。然而，这只有在执行PLC 任务 [▶ 63]的选项**浮点异常**被停用时才有效。否则会发生除以零的错误。

注意

浮点异常

如果正在执行的PLC 任务 [▶ 63]的选项**浮点异常**启用，该功能会不必要地引起错误。因此，停用这个选项。

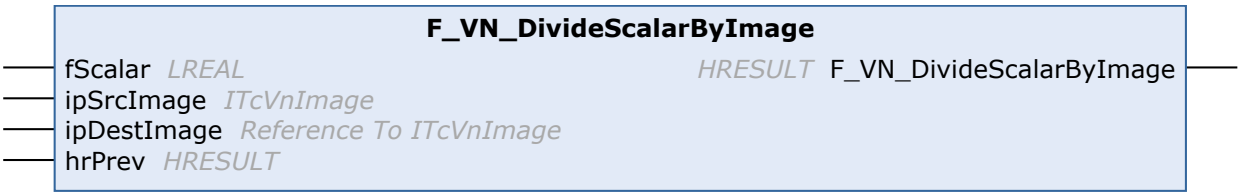
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.24 F_VN_DivideScalarByImage



Divides a scalar value by each image pixel using saturation arithmetics.

语法

定义：

```
FUNCTION F_VN_DivideScalarByImage : HRESULT
VAR_INPUT
    fScalar     : LREAL;
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fScalar	LREAL	标量值

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

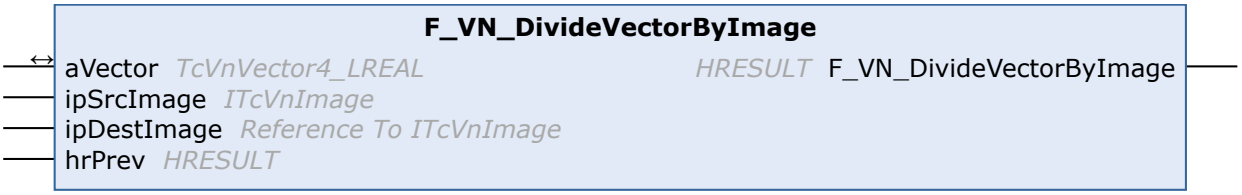
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.25 F_VN_DivideVectorByImage



Divides a vector (1 element for each image channel) by each image pixel using saturation arithmetics.

语法

定义：

```
FUNCTION F_VN_DivideVectorByImage : HRESULT
VAR_IN_OUT
    aVector     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aVector	TcVnVector4_LREAL [▶ 153]	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。

返回值

HRESULT [▶ 135]

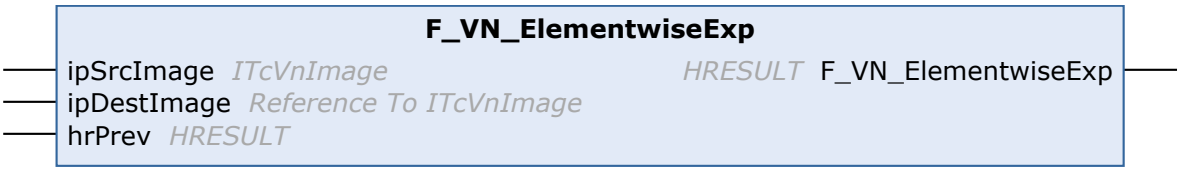
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.26 F_VN_ElementwiseExp



Computes the natural exponent of each pixel value.

语法

定义：

```
FUNCTION F_VN_ElementwiseExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（REAL 或 LREAL）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（类型和大小与 ipSrcImage 相同，如果需要，将创建适当的图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

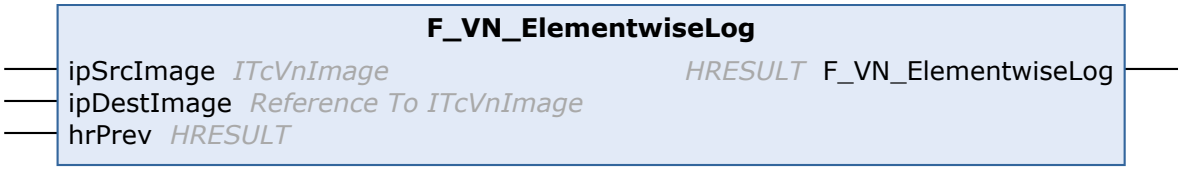
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.27 F_VN_ElementwiseLog



Computes the natural logarithm of each pixel value.

语法

定义：

```
FUNCTION F_VN_ElementwiseLog : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（REAL 或 LREAL）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（类型和大小与 ipSrcImage 相同，如果需要，将创建适当的图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.28 F_VN_MaxImage



Element-wise maximum of two images.

语法

定义：

```
FUNCTION F_VN_MaxImage : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
```

```
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.29 F_VN_MaxImageExp



Element-wise maximum of two images. (expert function)

语法

定义：

```
FUNCTION F_VN_MaxImageExp : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    ipMask      : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage [▶ 407]	USINT 类型的掩膜

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

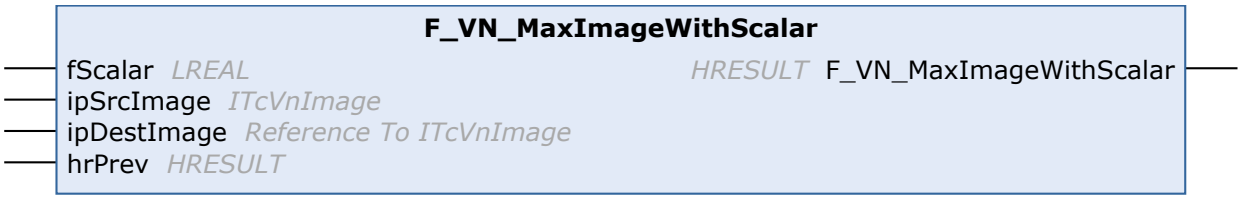
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.30 F_VN_MaxImageWithScalar



Element-wise maximum of image and scalar value.

语法

定义：

```
FUNCTION F_VN_MaxImageWithScalar : HRESULT
VAR_INPUT
    fScalar     : LREAL;
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fScalar	LREAL	标量值
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

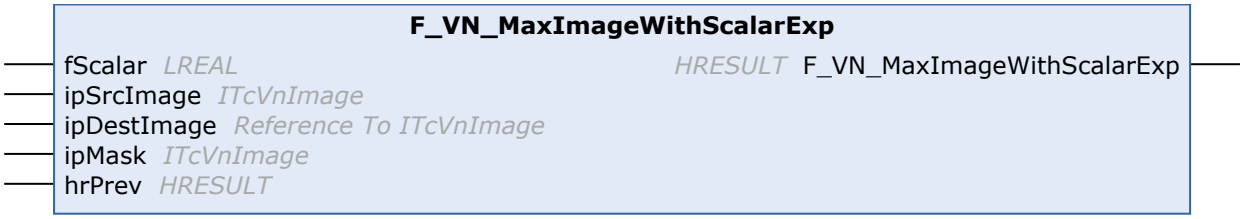
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.31 F_VN_MaxImageWithScalarExp



Element-wise maximum of image and scalar value. (expert function)

语法

定义：

```
FUNCTION F_VN_MaxImageWithScalarExp : HRESULT
VAR_INPUT
    fScalar     : LREAL;
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    ipMask      : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fScalar	LREAL	标量值
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage [▶ 407]	USINT 类型的掩膜
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.32 F_VN_MaxImageWithVector



Element-wise maximum of image and vector (1 element for each image channel).

语法

定义：

```
FUNCTION F_VN_MaxImageWithVector : HRESULT
VAR_IN_OUT
    aVector     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aVector	TcVnVector4_LREAL [▶ 153]	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。

返回值

HRESULT [▶ 135]

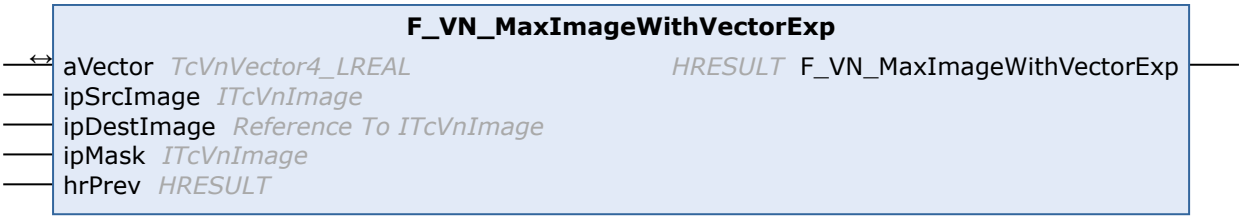
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.33 F_VN_MaxImageWithVectorExp



Element-wise maximum of image and vector (1 element for each image channel). (expert function)

语法

定义：

```
FUNCTION F_VN_MaxImageWithVectorExp : HRESULT
VAR_IN_OUT
    aVector     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    ipMask      : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage [▶ 407]	USINT 类型的掩膜
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aVector	TcVnVector4_LREAL [▶ 153]	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.34 F_VN_MinImage



Element-wise minimum of two images.

语法

定义：

```
FUNCTION F_VN_MinImage : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

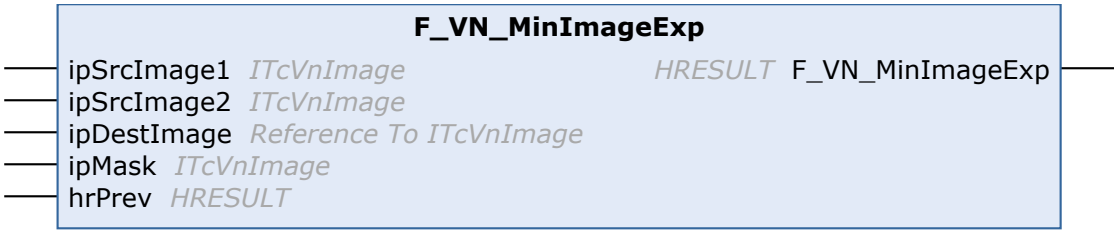
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.35 F_VN_MinImageExp



Element-wise minimum of two images. (expert function)

语法

定义：

```
FUNCTION F_VN_MinImageExp : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    ipMask      : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage [▶ 407]	USINT 类型的掩膜
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

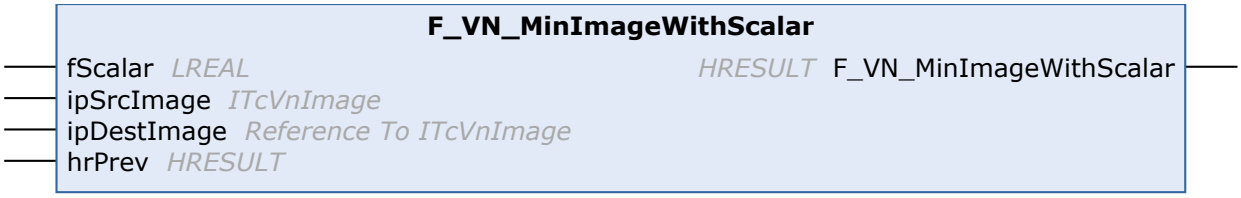
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.36 F_VN_MinImageWithScalar



Element-wise minimum of image and scalar value.

语法

定义：

```
FUNCTION F_VN_MinImageWithScalar : HRESULT
VAR_INPUT
    fScalar     : LREAL;
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fScalar	LREAL	标量值
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

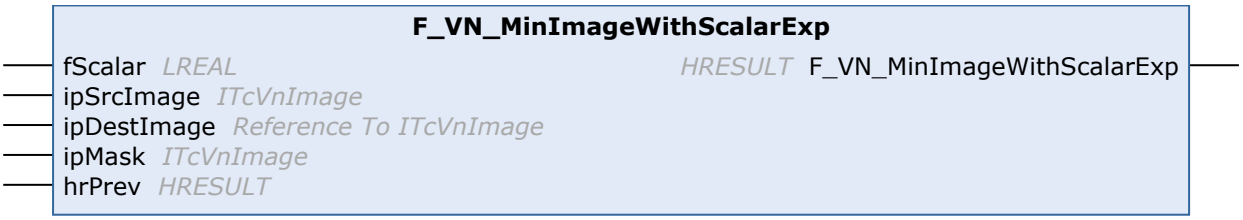
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.37 F_VN_MinImageWithScalarExp



Element-wise minimum of image and scalar value. (expert function)

语法

定义：

```
FUNCTION F_VN_MinImageWithScalarExp : HRESULT
VAR_INPUT
    fScalar     : LREAL;
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    ipMask      : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fScalar	LREAL	标量值
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage [▶ 407]	USINT 类型的掩膜
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.38 F_VN_MinImageWithVector



Element-wise minimum of image and vector (1 element for each image channel).

语法

定义：

```
FUNCTION F_VN_MinImageWithVector : HRESULT
VAR_IN_OUT
    aVector     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aVector	TcVnVector4_LREAL [▶ 153]	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。

返回值

HRESULT [▶ 135]

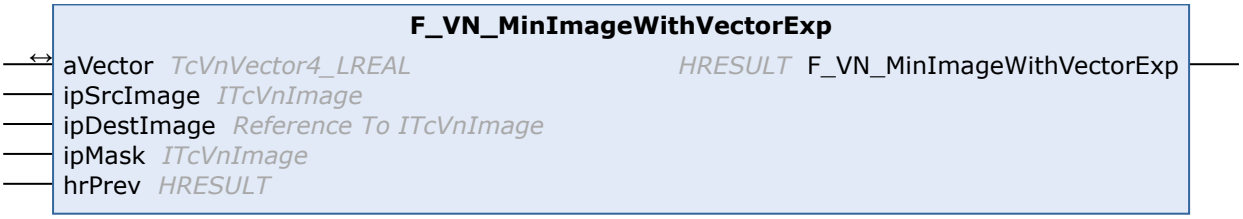
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.39 F_VN_MinImageWithVectorExp



Element-wise minimum of image and vector (1 element for each image channel). (expert function)

语法

定义：

```
FUNCTION F_VN_MinImageWithVectorExp : HRESULT
VAR_IN_OUT
    aVector     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    ipMask      : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage [▶ 407]	USINT 类型的掩膜
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aVector	TcVnVector4_LREAL [▶ 153]	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.40 F_VN_MultiplyImages



Element-wise multiplication of two images using saturation arithmetics.

语法

定义：

```
FUNCTION F_VN_MultiplyImages : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

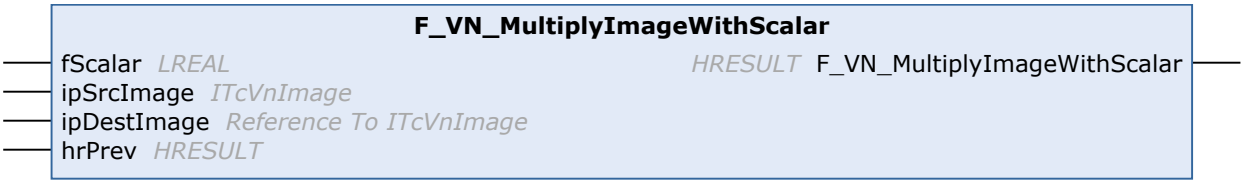
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.41 F_VN_MultiplyImageWithScalar



Multiply each image pixel by a scalar value using saturation arithmetics.

语法

定义：

```
FUNCTION F_VN_MultiplyImageWithScalar : HRESULT
VAR_INPUT
    fScalar     : LREAL;
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fScalar	LREAL	标量值
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.42 F_VN_MultiplyImageWithVector



Multiply each image pixel by a vector (1 element for each image channel) using saturation arithmetics.

语法

定义：

```
FUNCTION F_VN_MultiplyImageWithVector : HRESULT
VAR_IN_OUT
    aVector     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aVector	TcVnVector4_LREAL [▶ 153]	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.43 F_VN_SubtractImageFromScalar



Subtract each image pixel from a scalar value using saturation arithmetics.

语法

定义：

```
FUNCTION F_VN_SubtractImageFromScalar : HRESULT
VAR_INPUT
    fScalar     : LREAL;
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fScalar	LREAL	标量值
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

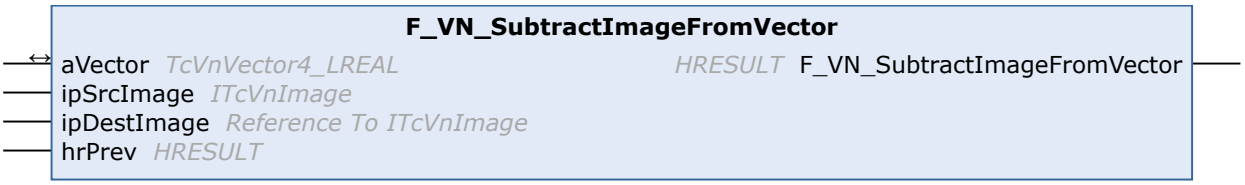
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.44 F_VN_SubtractImageFromVector



Subtract each image pixel from a vector (1 element for each image channel) using saturation arithmetics.

语法

定义：

```
FUNCTION F_VN_SubtractImageFromVector : HRESULT
VAR_IN_OUT
    aVector     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aVector	TcVnVector4_LREAL [▶ 153]	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。

返回值

HRESULT [▶ 135]

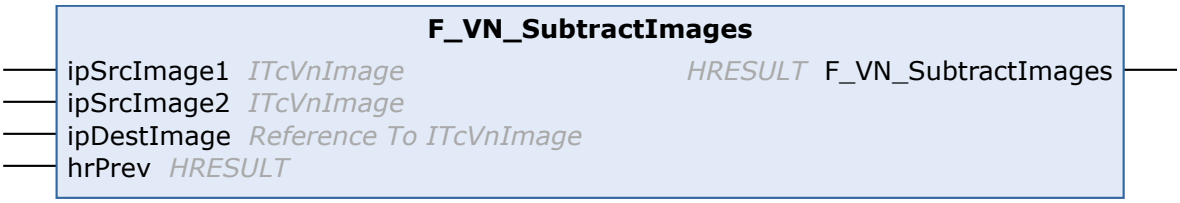
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.45 F_VN_SubtractImages



Element-wise subtraction of two images using saturation arithmetics.

语法

定义：

```
FUNCTION F_VN_SubtractImages : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	第一个源图像（被减数）
ipSrcImage2	ITcVnImage [▶ 407]	第二个源图像（减数）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

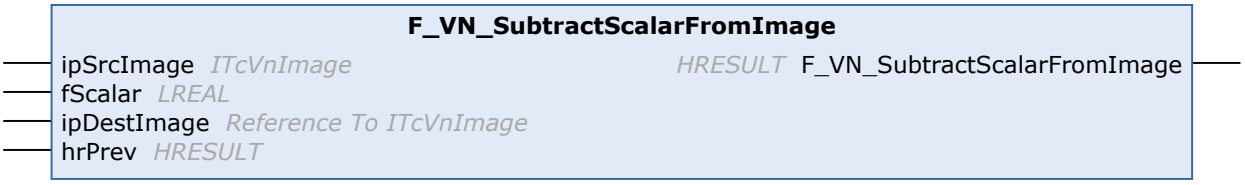
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.46 F_VN_SubtractScalarFromImage



Subtract a scalar value from each image pixel using saturation arithmetics.

语法

定义：

```
FUNCTION F_VN_SubtractScalarFromImage : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    fScalar     : LREAL;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
fScalar	LREAL	标量值
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

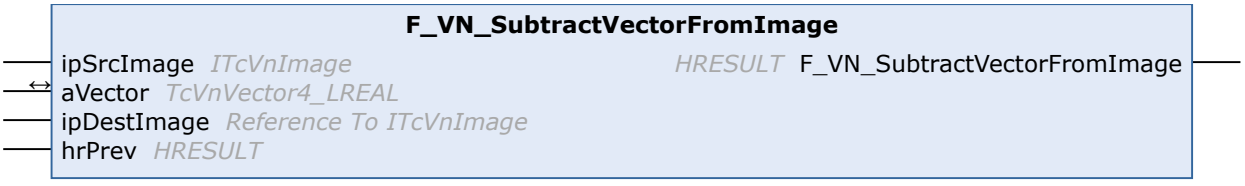
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.4.47 F_VN_SubtractVectorFromImage



Subtract a vector (1 element for each image channel) from each image pixel using saturation arithmetics.

语法

定义：

```
FUNCTION F_VN_SubtractVectorFromImage : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
END_VAR
VAR_IN_OUT
    aVector     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aVector	TcVnVector4_LREAL [▶ 153]	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

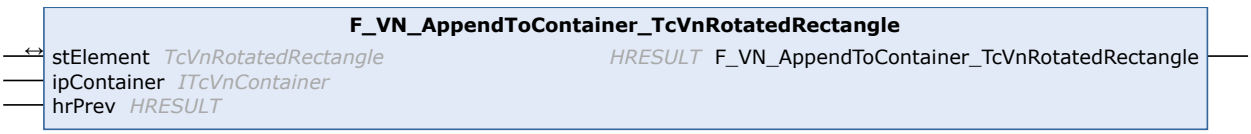
开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5 基本容器运算

6.1.4.5.1 F_VN_AppendToContainer

本章包含根据数据类型向容器中添加元素的功能。

6.1.4.5.1.1 F_VN_AppendToContainer_TcVnRotatedRectangle



将 TcVnRotatedRectangle 类型的单个元素添加到容器中。

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnRotatedRectangle : HRESULT
VAR_IN_OUT
    stElement   : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnRotatedRectangle [▶ 237]	添加到 ipContainer 的单个元素

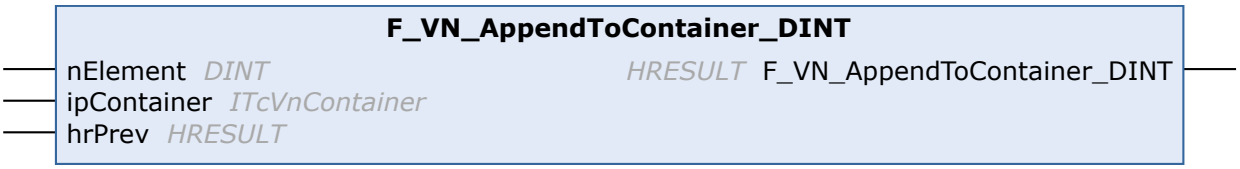
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.5.1.2 F_VN_AppendToContainer_DINT



Append a single element of type DINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_DINT : HRESULT
VAR_INPUT
    nElement    : DINT;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	DINT	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

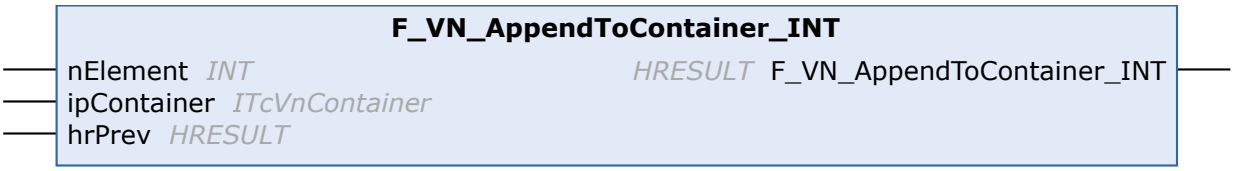
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.3 F_VN_AppendToContainer_INT



Append a single element of type INT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_INT : HRESULT
VAR_INPUT
    nElement    : INT;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	INT	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

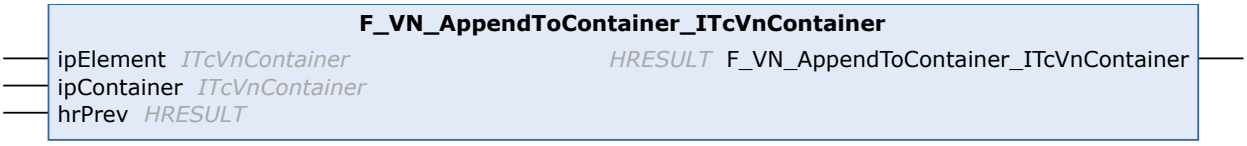
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.4 F_VN_AppendToContainer_ITcVnContainer



Append a single element to a container or concatenate containers (if ipElement has the same typeId as ipContainer).

语法

定义：

```
FUNCTION F_VN_AppendToContainer_ITcVnContainer : HRESULT
VAR_INPUT
    ipElement   : ITcVnContainer;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipElement	ITcVnContainer [▶ 365]	单个元素，添加到 ipContainer，或添加到包含多个元素的要连接的容器
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

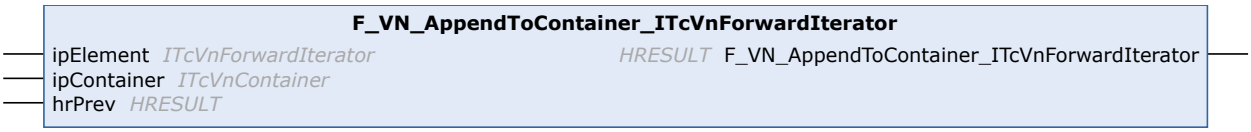
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.5 F_VN_AppendToContainer_ITcVnForwardIterator



Append a single element (represented by an iterator)to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_ITcVnForwardIterator : HRESULT
VAR_INPUT
    ipElement   : ITcVnForwardIterator;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipElement	ITcVnForwardIterator [▶ 355]	添加到 ipContainer 的单个元素

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

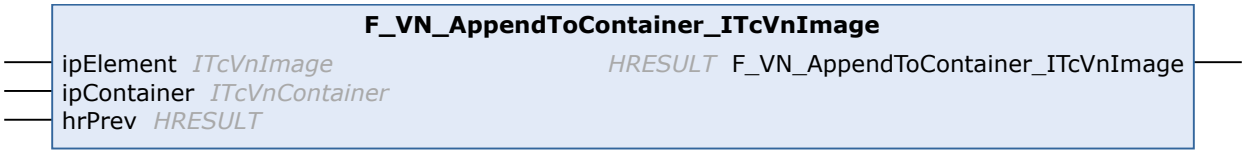
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.6 F_VN_AppendToContainer_ITcVnImage



Append a single element of type ITcVnImage to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_ITcVnImage : HRESULT
VAR_INPUT
    ipElement   : ITcVnImage;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipElement	ITcVnImage [▶ 407]	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

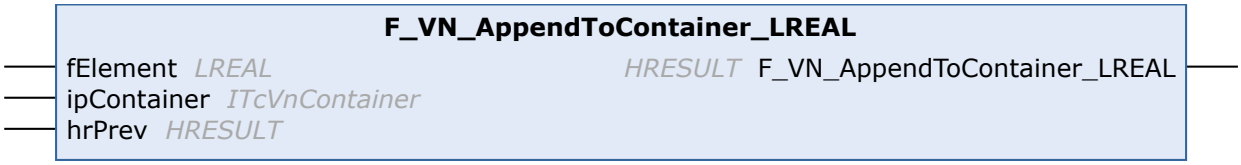
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.7 F_VN_AppendToContainer_LREAL



Append a single element of type LREAL to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_LREAL : HRESULT
VAR_INPUT
    fElement     : LREAL;
    ipContainer : ITcVnContainer;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
fElement	LREAL	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

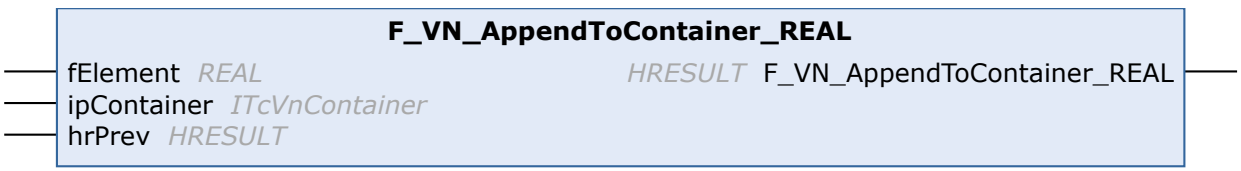
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.8 F_VN_AppendToContainer_REAL



Append a single element of type REAL to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_REAL : HRESULT
VAR_INPUT
    fElement    : REAL;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fElement	REAL	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

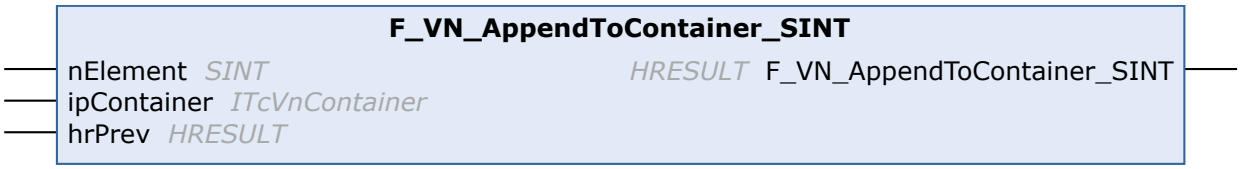
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.9 F_VN_AppendToContainer_SINT



Append a single element of type SINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_SINT : HRESULT
VAR_INPUT
    nElement    : SINT;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	SINT	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

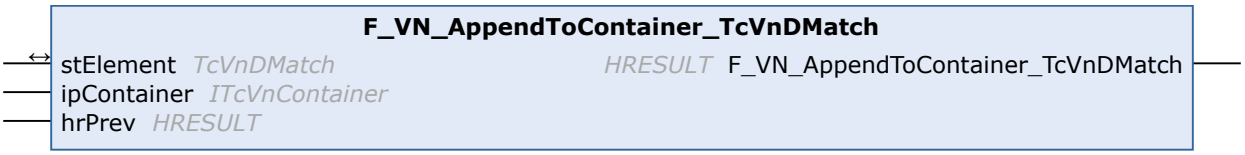
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.10 F_VN_AppendToContainer_TcVnDMatch



Append a single element of type TcVnDMatch to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnDMatch : HRESULT
VAR_IN_OUT
    stElement   : TcVnDMatch;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnDMatch [▶ 225]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

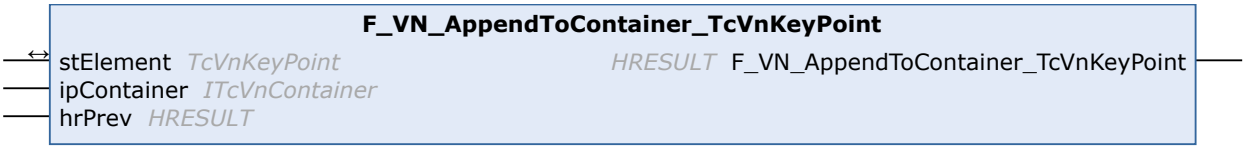
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.11 F_VN_AppendToContainer_TcVnKeyPoint



Append a single element of type TcVnKeyPoint to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnKeyPoint : HRESULT
VAR_IN_OUT
    stElement   : TcVnKeyPoint;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnKeyPoint [▶ 226]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

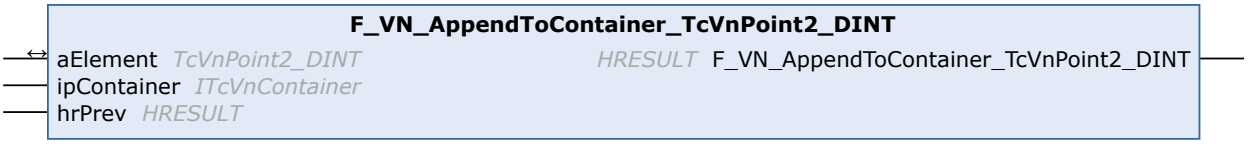
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.12 F_VN_AppendToContainer_TcVnPoint2_DINT



Append a single element of type TcVnPoint2_DINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnPoint2_DINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint2_DINT [▶ 152]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

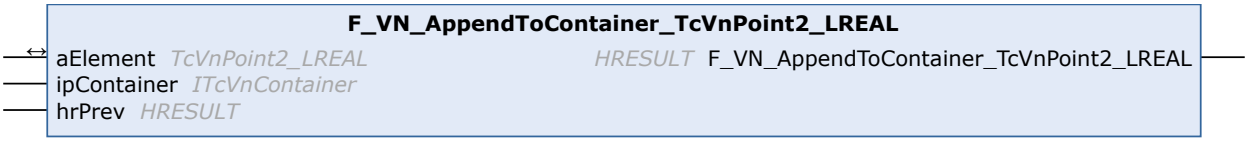
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.13 F_VN_AppendToContainer_TcVnPoint2_LREAL



Append a single element of type TcVnPoint2_LREAL to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnPoint2_LREAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnPoint2_LREAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint2_LREAL [▶ 152]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

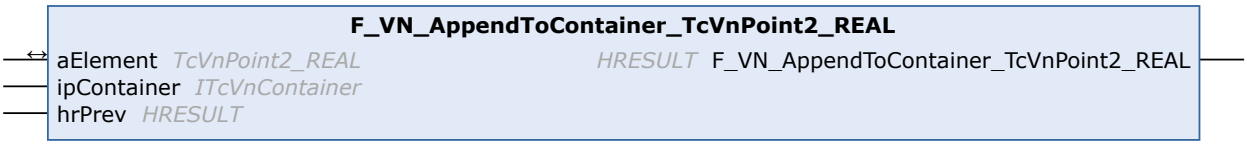
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.14 F_VN_AppendToContainer_TcVnPoint2_REAL



Append a single element of type TcVnPoint2_REAL to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnPoint2_REAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint2_REAL [▶ 152]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

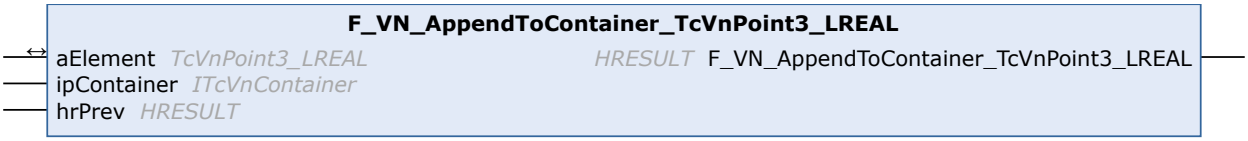
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.15 F_VN_AppendToContainer_TcVnPoint3_LREAL



Append a single element of type TcVnPoint3_LREAL to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnPoint3_LREAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnPoint3_LREAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint3_LREAL [▶ 152]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

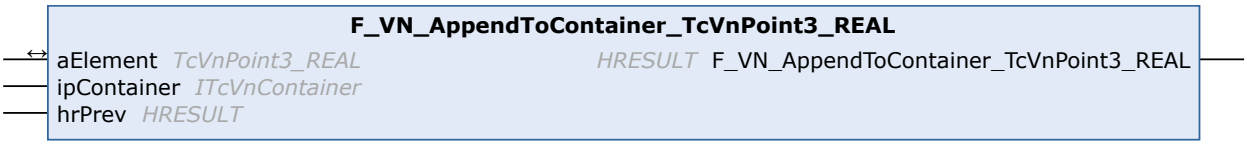
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.16 F_VN_AppendToContainer_TcVnPoint3_REAL



Append a single element of type TcVnPoint3_REAL to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnPoint3_REAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnPoint3_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint3_REAL [▶ 152]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

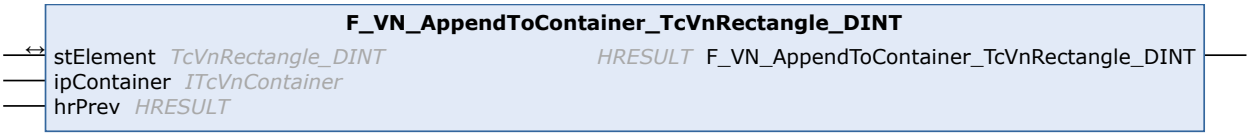
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.17 F_VN_AppendToContainer_TcVnRectangle_DINT



Append a single element of type TcVnRectangle_DINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnRectangle_DINT : HRESULT
VAR_IN_OUT
    stElement   : TcVnRectangle_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnRectangle_DINT [▶ 236]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

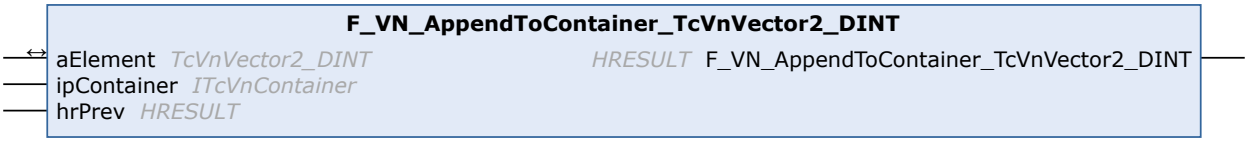
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.18 F_VN_AppendToContainer_TcVnVector2_DINT



Append a single element of type TcVnVector2_DINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector2_DINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector2_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_DINT [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

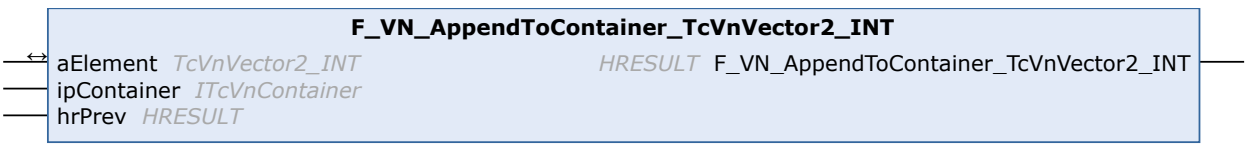
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.19 F_VN_AppendToContainer_TcVnVector2_INT



Append a single element of type TcVnVector2_INT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector2_INT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector2_INT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_INT [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

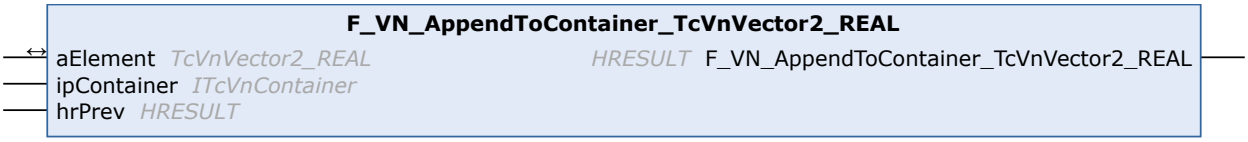
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.20 F_VN_AppendToContainer_TcVnVector2_REAL



Append a single element of type TcVnVector2_REAL to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector2_REAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector2_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_REAL [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

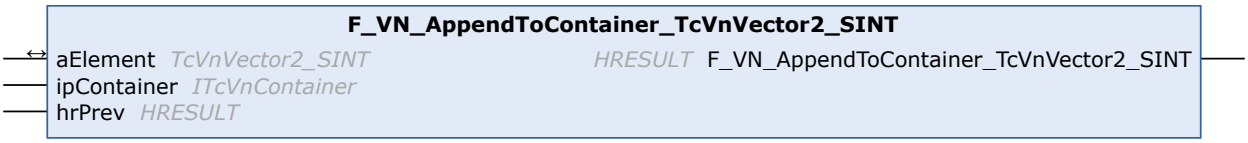
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.21 F_VN_AppendToContainer_TcVnVector2_SINT



Append a single element of type TcVnVector2_SINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector2_SINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector2_SINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_SINT [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

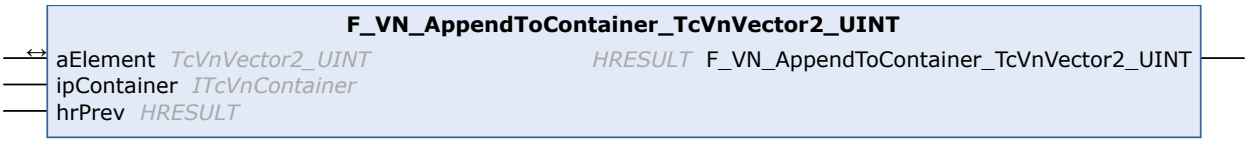
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.22 F_VN_AppendToContainer_TcVnVector2_UINT



Append a single element of type TcVnVector2_UINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector2_UINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector2_UINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_UINT [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

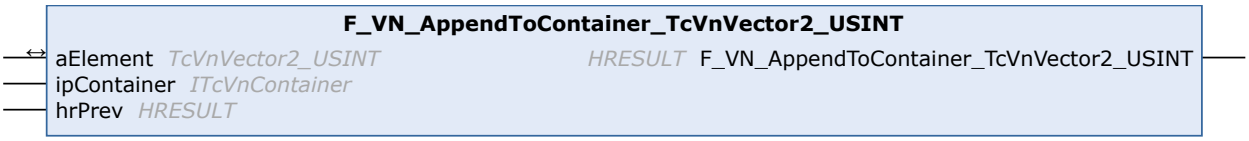
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.23 F_VN_AppendToContainer_TcVnVector2_USINT



Append a single element of type TcVnVector2_USINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector2_USINT : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector2_USINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_USINT [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

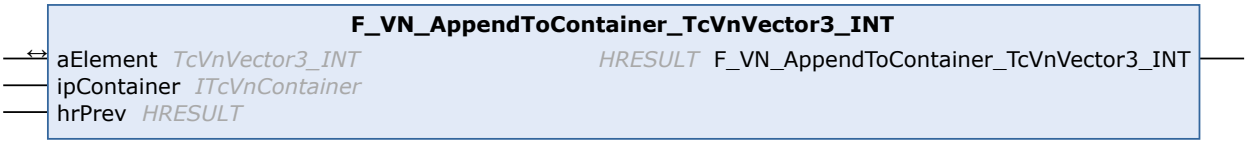
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.24 F_VN_AppendToContainer_TcVnVector3_INT



Append a single element of type TcVnVector3_INT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector3_INT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector3_INT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_INT [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

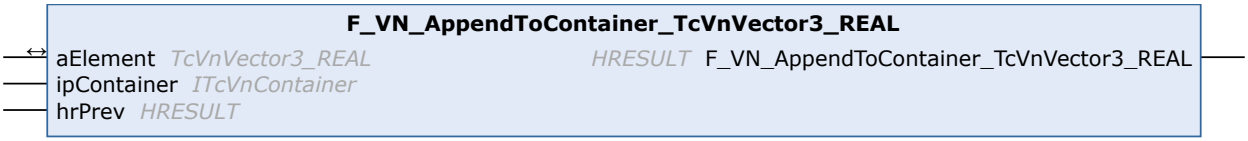
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.25 F_VN_AppendToContainer_TcVnVector3_REAL



Append a single element of type TcVnVector3_REAL to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector3_REAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector3_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_REAL [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

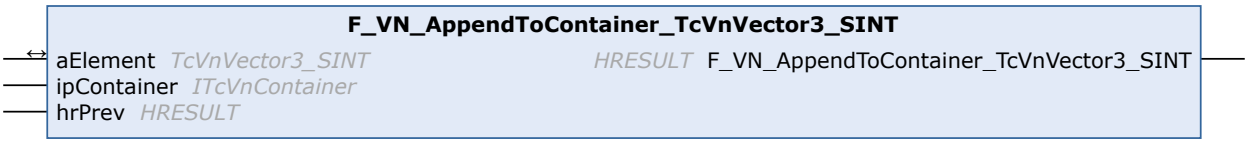
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.26 F_VN_AppendToContainer_TcVnVector3_SINT



Append a single element of type TcVnVector3_SINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector3_SINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector3_SINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_SINT [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

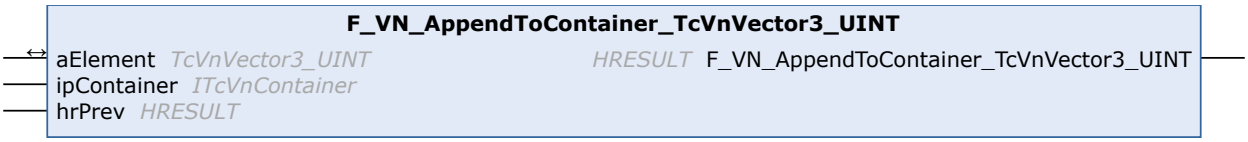
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.27 F_VN_AppendToContainer_TcVnVector3_UINT



Append a single element of type TcVnVector3_UINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector3_UINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector3_UINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_UINT [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

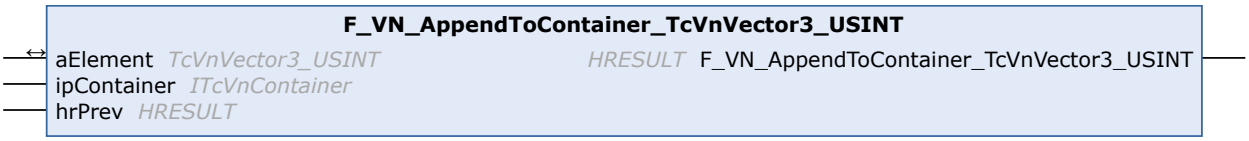
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.28 F_VN_AppendToContainer_TcVnVector3_USINT



Append a single element of type TcVnVector3_USINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector3_USINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector3_USINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_USINT [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

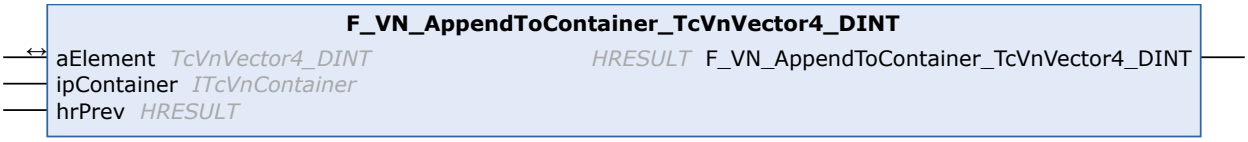
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.29 F_VN_AppendToContainer_TcVnVector4_DINT



Append a single element of type TcVnVector4_DINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector4_DINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector4_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_DINT [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

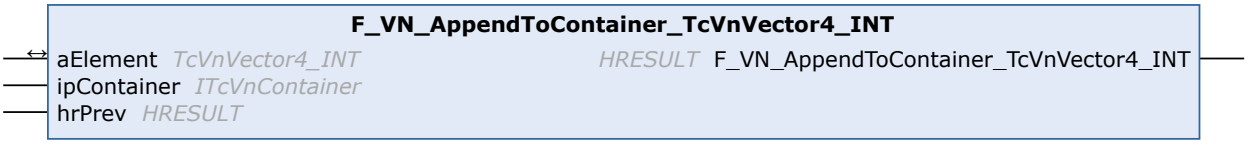
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.30 F_VN_AppendToContainer_TcVnVector4_INT



Append a single element of type TcVnVector4_INT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector4_INT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector4_INT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_INT [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.31 F_VN_AppendToContainer_TcVnVector4_LREAL



Append a single element of type TcVnVector4_LREAL to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector4_LREAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_LREAL [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

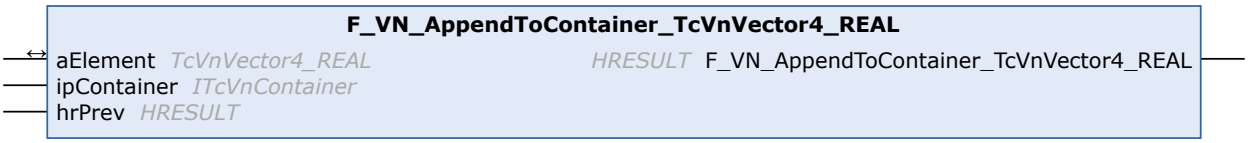
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.32 F_VN_AppendToContainer_TcVnVector4_REAL



Append a single element of type TcVnVector4_REAL to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector4_REAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector4_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_REAL [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

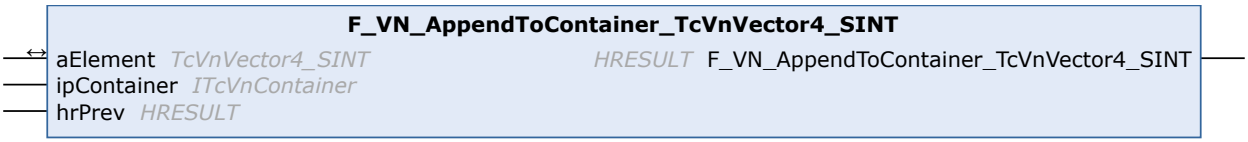
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.33 F_VN_AppendToContainer_TcVnVector4_SINT



Append a single element of type TcVnVector4_SINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector4_SINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector4_SINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_SINT [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

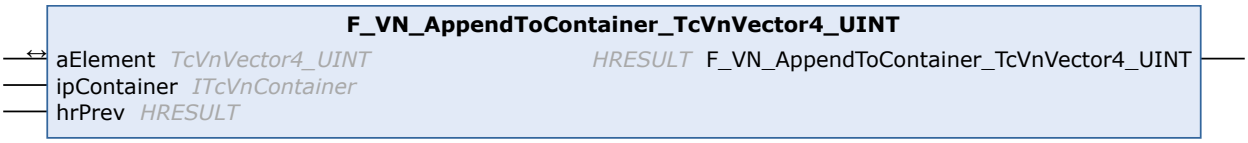
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.34 F_VN_AppendToContainer_TcVnVector4_UINT



Append a single element of type TcVnVector4_UINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector4_UINT : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector4_UINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_UINT [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

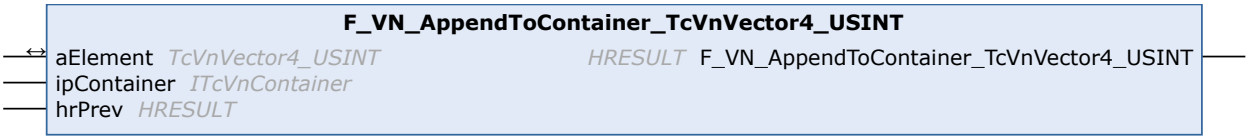
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.35 F_VN_AppendToContainer_TcVnVector4_USINT



Append a single element of type TcVnVector4_USINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_TcVnVector4_USINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector4_USINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_USINT [▶ 153]	添加到 ipContainer 的单个元素

返回值

HRESULT [▶ 135]

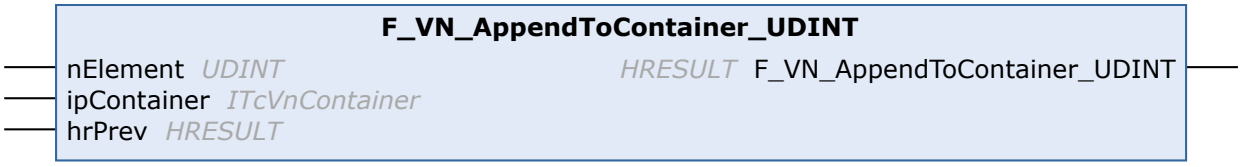
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.36 F_VN_AppendToContainer_UDINT



Append a single element of type UDINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_UDINT : HRESULT
VAR_INPUT
    nElement     : UDINT;
    ipContainer : ITcVnContainer;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	UDINT	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

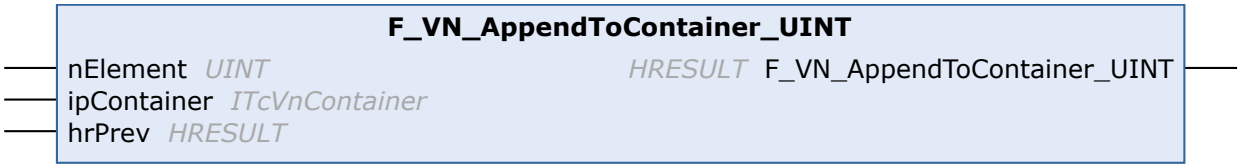
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.37 F_VN_AppendToContainer_UINT



Append a single element of type UINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_UINT : HRESULT
VAR_INPUT
    nElement     : UINT;
    ipContainer : ITcVnContainer;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	UINT	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

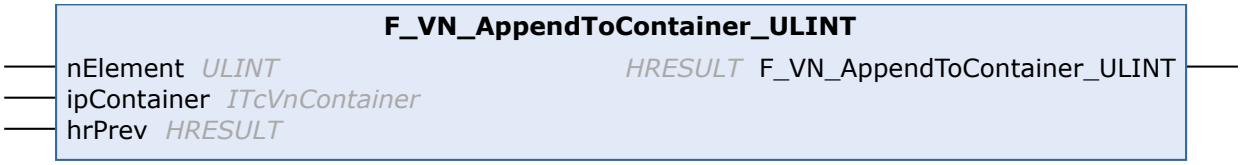
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.38 F_VN_AppendToContainer_ULINT



Append a single element of type ULINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_ULINT : HRESULT
VAR_INPUT
    nElement     : ULINT;
    ipContainer : ITcVnContainer;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	ULINT	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

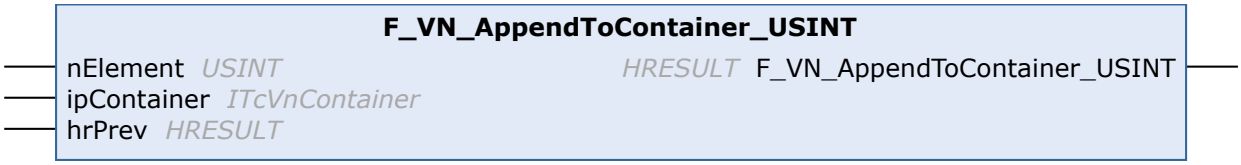
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.1.39 F_VN_AppendToContainer_USINT



Append a single element of type USINT to a container.

语法

定义：

```
FUNCTION F_VN_AppendToContainer_USINT : HRESULT
VAR_INPUT
    nElement    : USINT;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	USINT	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer [▶ 365]	将要添加元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

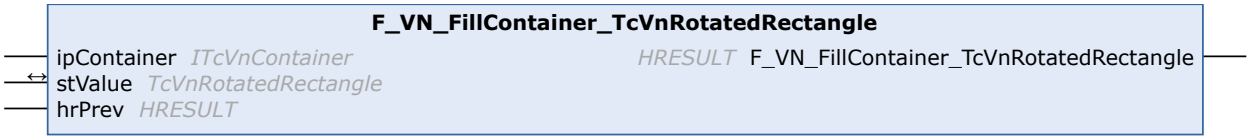
系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2 F_VN_FillContainer

本章包含根据数据类型用特殊值填充容器的功能。这个函数适用于包括基本元素的容器。

6.1.4.5.2.1 F_VN_FillContainer_TcVnRotatedRectangle



用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnRotatedRectangle : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stValue     : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stValue	TcVnRotatedRectangle [▶ 237]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.5.2.2 F_VN_FillContainer_DINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nValue      : DINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nValue	DINT	用于设置容器元素的值

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.3 F_VN_FillContainer_INT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nValue      : INT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nValue	INT	用于设置容器元素的值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

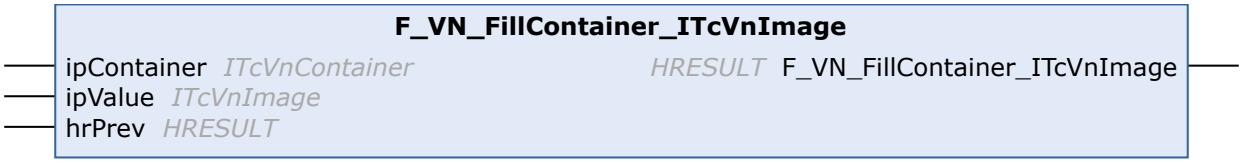
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.4 F_VN_FillContainer_ITcVnImage



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_ITcVnImage : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    ipValue     : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
ipValue	ITcVnImage [▶ 407]	用于设置容器元素的值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

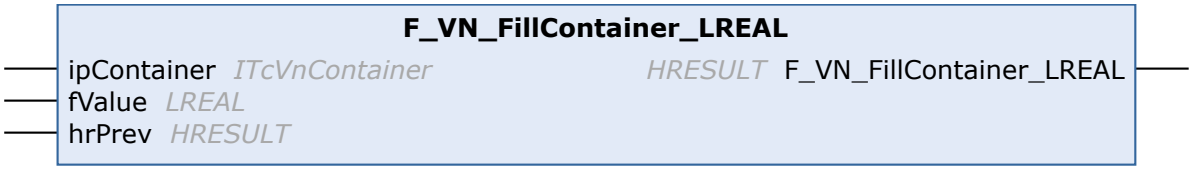
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.5 F_VN_FillContainer_LREAL



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    fValue      : LREAL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
fValue	LREAL	用于设置容器元素的值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.6 F_VN_FillContainer_REAL



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    fValue      : REAL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
fValue	REAL	用于设置容器元素的值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.7 F_VN_FillContainer_SINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nValue      : SINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nValue	SINT	用于设置容器元素的值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

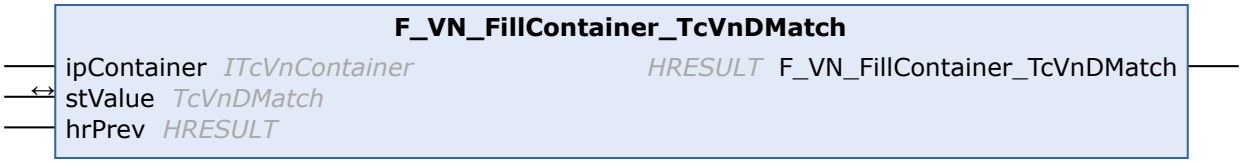
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.8 F_VN_FillContainer_TcVnDMatch



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnDMatch : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stValue     : TcVnDMatch;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stValue	TcVnDMatch [▶ 225]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

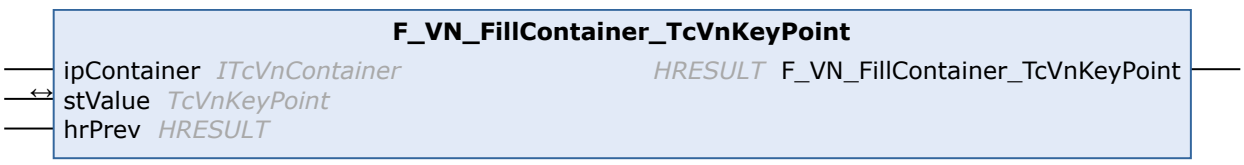
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.9 F_VN_FillContainer_TcVnKeyPoint



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnKeyPoint : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stValue     : TcVnKeyPoint;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stValue	TcVnKeyPoint [▶ 226]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.10 F_VN_FillContainer_TcVnPoint2_DINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnPoint2_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnPoint2_DINT [▶ 152]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

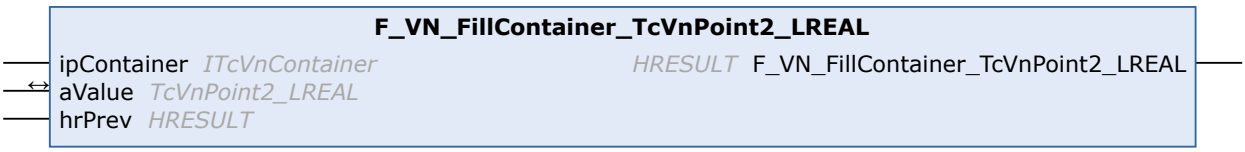
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.11 F_VN_FillContainer_TcVnPoint2_LREAL



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnPoint2_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnPoint2_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnPoint2_LREAL [▶ 152]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

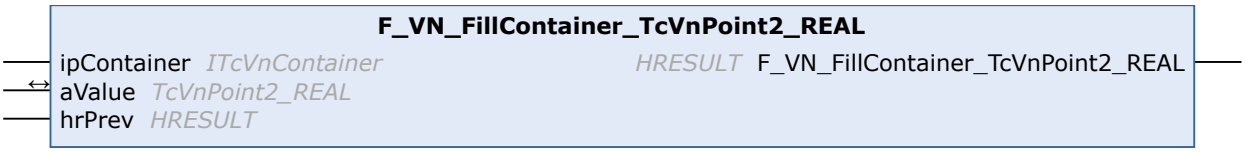
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.12 F_VN_FillContainer_TcVnPoint2_REAL



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnPoint2_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnPoint2_REAL [▶ 152]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.13 F_VN_FillContainer_TcVnPoint3_LREAL



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnPoint3_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnPoint3_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnPoint3_LREAL [▶ 152]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.14 F_VN_FillContainer_TcVnPoint3_REAL



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnPoint3_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnPoint3_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnPoint3_REAL [▶ 152]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

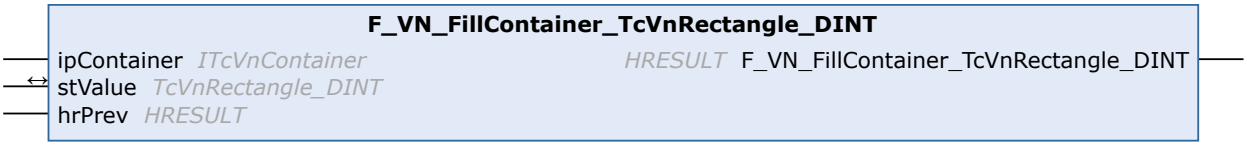
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.15 F_VN_FillContainer_TcVnRectangle_DINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnRectangle_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stValue     : TcVnRectangle_DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stValue	TcVnRectangle_DINT [▶ 236]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

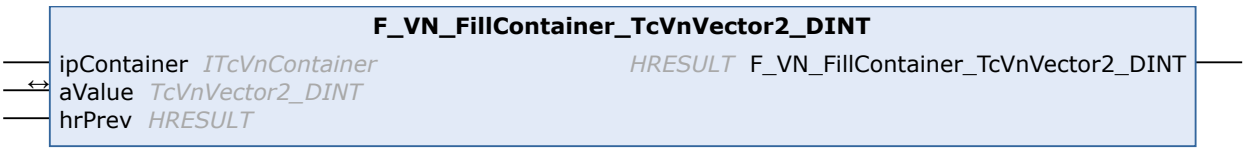
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.16 F_VN_FillContainer_TcVnVector2_DINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector2_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector2_DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_DINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

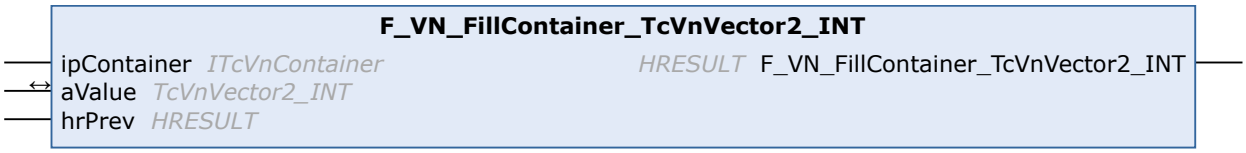
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.17 F_VN_FillContainer_TcVnVector2_INT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector2_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector2_INT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_INT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

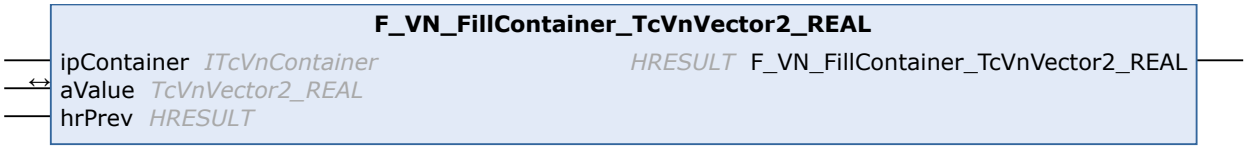
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.18 F_VN_FillContainer_TcVnVector2_REAL



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector2_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector2_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_REAL [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

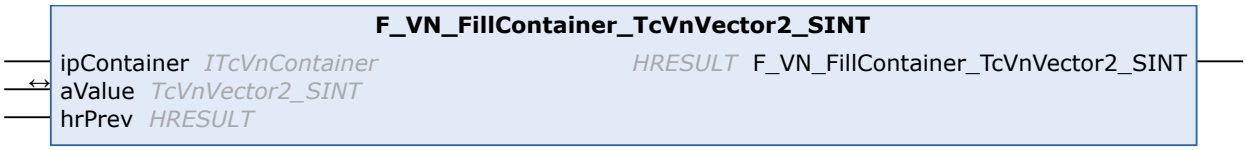
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.19 F_VN_FillContainer_TcVnVector2_SINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector2_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector2_SINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_SINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

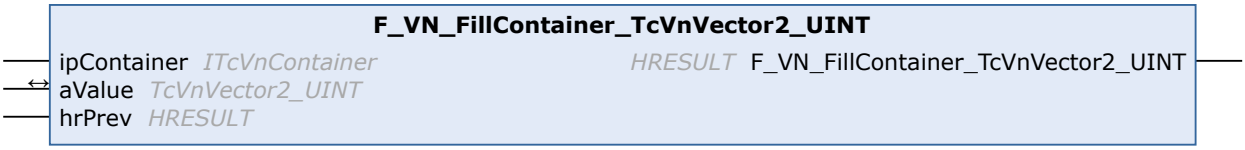
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.20 F_VN_FillContainer_TcVnVector2_UINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector2_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector2_UINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_UINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

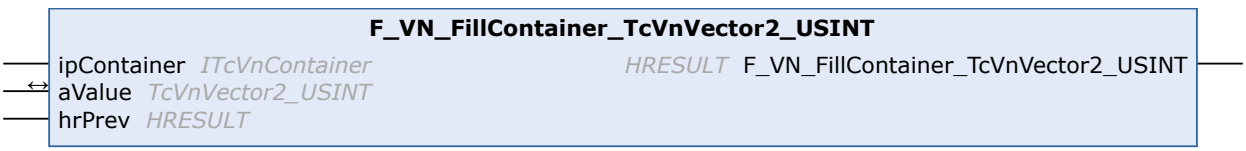
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.21 F_VN_FillContainer_TcVnVector2_USINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector2_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector2_USINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_USINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

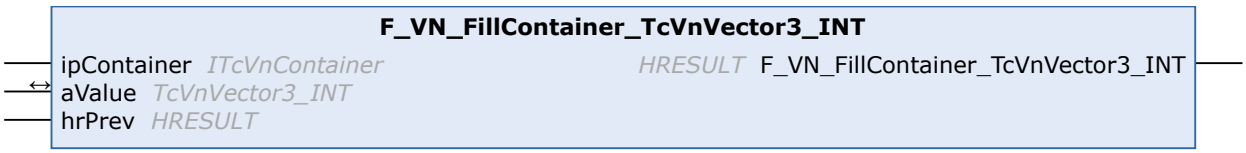
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.22 F_VN_FillContainer_TcVnVector3_INT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector3_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector3_INT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_INT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.23 F_VN_FillContainer_TcVnVector3_REAL



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector3_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector3_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_REAL [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

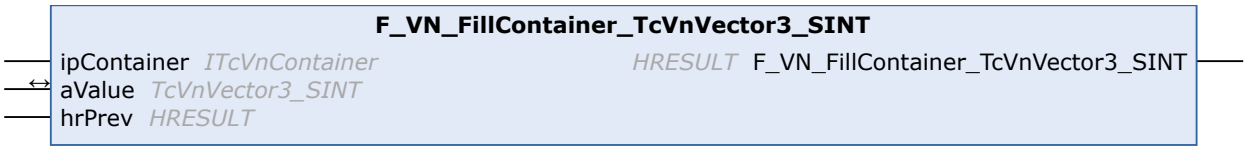
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.24 F_VN_FillContainer_TcVnVector3_SINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector3_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector3_SINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_SINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

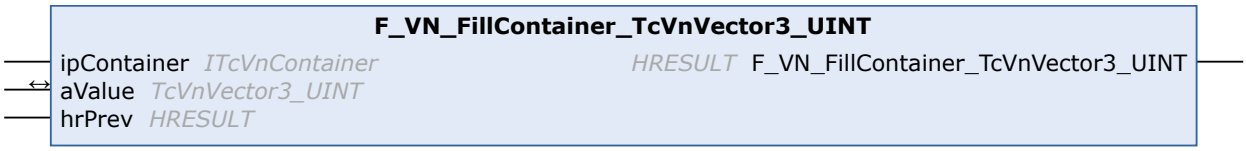
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.25 F_VN_FillContainer_TcVnVector3_UINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector3_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector3_UINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_UINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

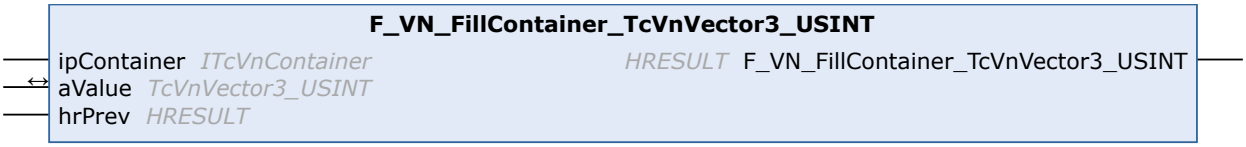
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.26 F_VN_FillContainer_TcVnVector3_USINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector3_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector3_USINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_USINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

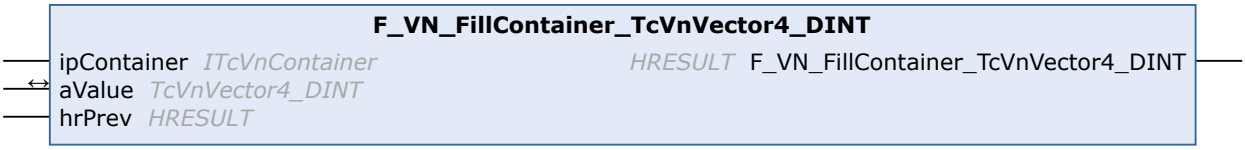
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.27 F_VN_FillContainer_TcVnVector4_DINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector4_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_DINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

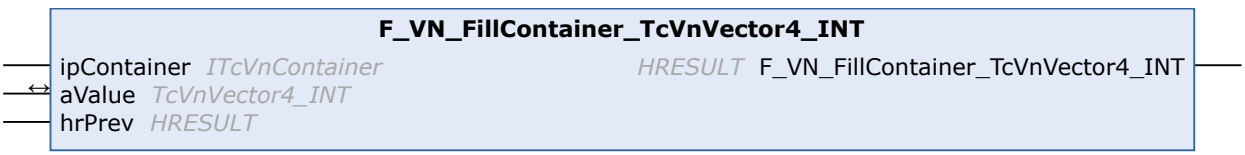
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.28 F_VN_FillContainer_TcVnVector4_INT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector4_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_INT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_INT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

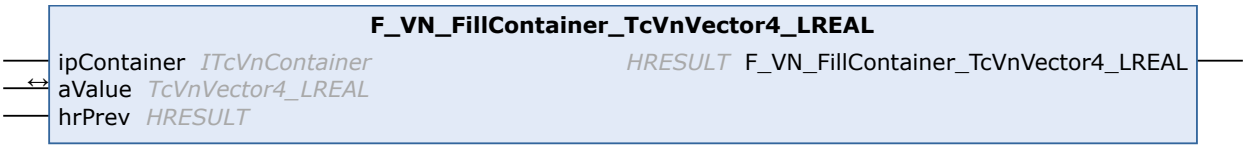
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.29 F_VN_FillContainer_TcVnVector4_LREAL



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector4_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_LREAL [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

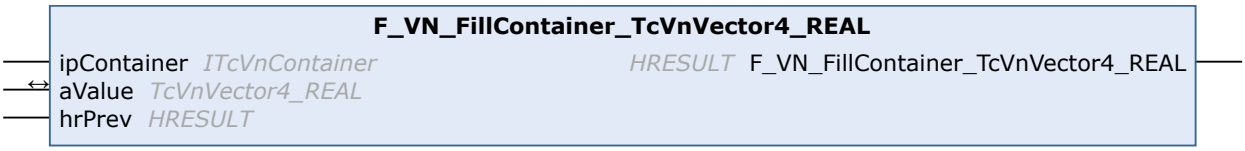
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.30 F_VN_FillContainer_TcVnVector4_REAL



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector4_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_REAL [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

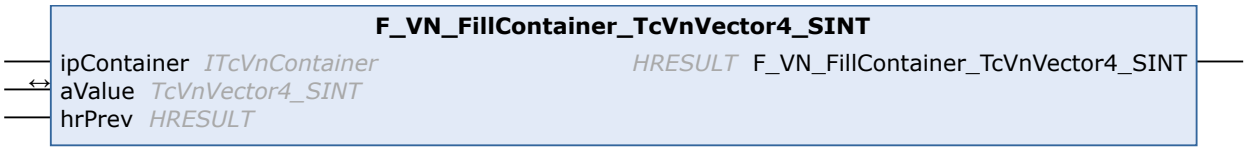
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.31 F_VN_FillContainer_TcVnVector4_SINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector4_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_SINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_SINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

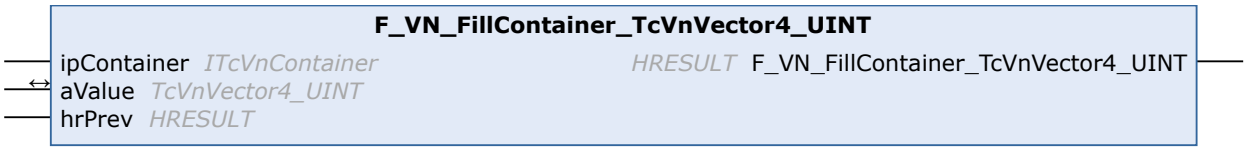
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.32 F_VN_FillContainer_TcVnVector4_UINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector4_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_UINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_UINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

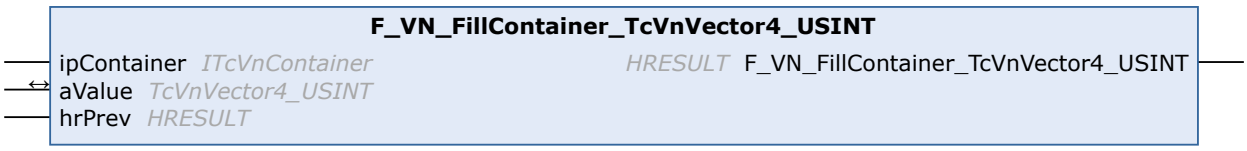
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.33 F_VN_FillContainer_TcVnVector4_USINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_TcVnVector4_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_USINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_USINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

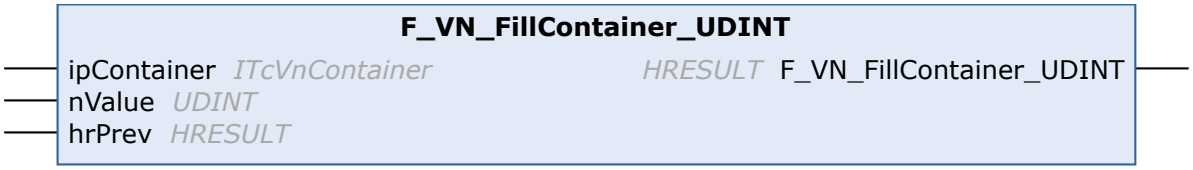
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.34 F_VN_FillContainer_UDINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_UDINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nValue      : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nValue	UDINT	用于设置容器元素的值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.35 F_VN_FillContainer_UINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nValue      : UINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nValue	UINT	用于设置容器元素的值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

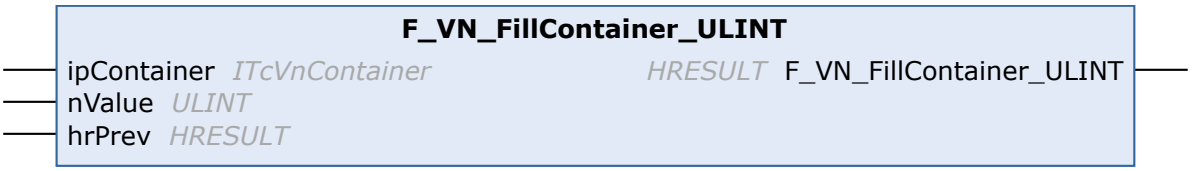
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.36 F_VN_FillContainer_ULINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_ULINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nValue      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nValue	ULINT	用于设置容器元素的值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

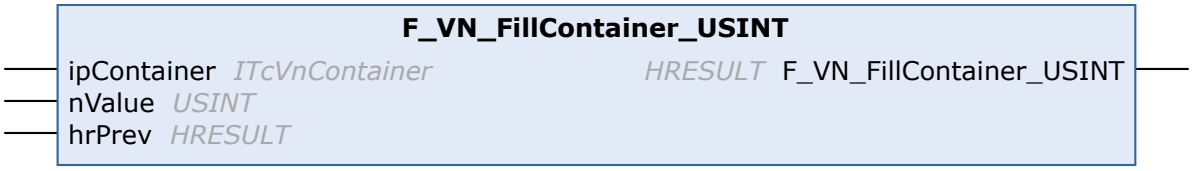
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.2.37 F_VN_FillContainer_USINT



Fill the container with the specified value. Only possible for containers with basic elements.

语法

定义：

```
FUNCTION F_VN_FillContainer_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nValue      : USINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nValue	USINT	用于设置容器元素的值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

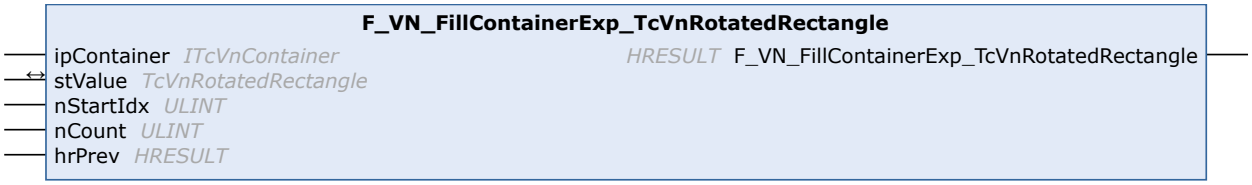
系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3 F_VN_FillContainerExp

本章包含专家函数，根据数据类型，可用于用特定的值填充容器，即使只是部分填充。这个函数适用于包括基本元素的容器。

6.1.4.5.3.1 F_VN_FillContainerExp_TcVnRotatedRectangle



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnRotatedRectangle : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stValue     : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stValue	TcVnRotatedRectangle [▶ 237]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.2 F_VN_FillContainerExp_DINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
```

```
    nValue      : DINT;
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nValue	DINT	用于设置容器元素的值
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.3 F_VN_FillContainerExp_INT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nValue      : INT;
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nValue	INT	用于设置容器元素的值

名称	类型	描述
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.4 F_VN_FillContainerExp_ITcVnImage



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_ITcVnImage : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    ipValue     : ITcVnImage;
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
ipValue	ITcVnImage [▶ 407]	用于设置容器元素的值
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.5 F_VN_FillContainerExp_LREAL



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    fValue      : LREAL;
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
fValue	LREAL	用于设置容器元素的值
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.6 F_VN_FillContainerExp_REAL



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    fValue      : REAL;
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
fValue	REAL	用于设置容器元素的值
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.7 F_VN_FillContainerExp_SINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nValue      : SINT;
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nValue	SINT	用于设置容器元素的值
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.8 F_VN_FillContainerExp_TcVnDMatch



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnDMatch : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stValue     : TcVnDMatch;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stValue	TcVnDMatch [▶ 225]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.9 F_VN_FillContainerExp_TcVnKeyPoint



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnKeyPoint : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stValue     : TcVnKeyPoint;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器

名称	类型	描述
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stValue	TcVnKeyPoint [▶ 226]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

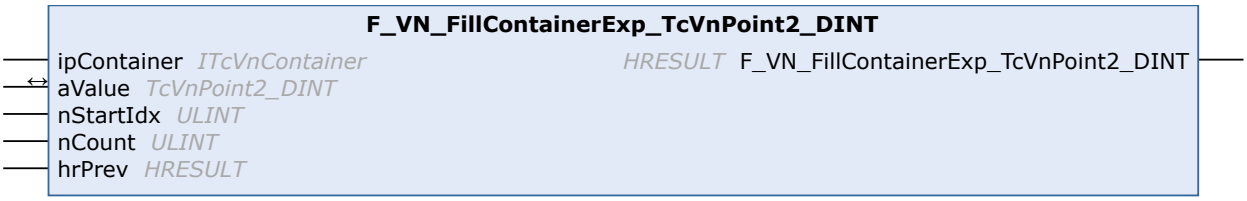
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.10 F_VN_FillContainerExp_TcVnPoint2_DINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnPoint2_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnPoint2_DINT [▶ 152]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.11 F_VN_FillContainerExp_TcVnPoint2_LREAL



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnPoint2_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnPoint2_LREAL;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnPoint2_LREAL [▶ 152]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

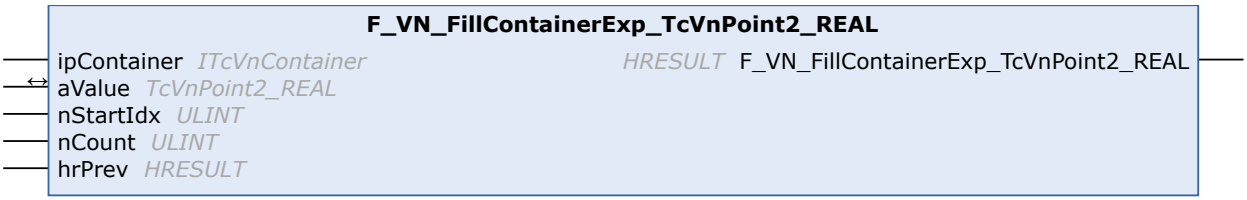
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.12 F_VN_FillContainerExp_TcVnPoint2_REAL



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnPoint2_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnPoint2_REAL [▶ 152]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.13 F_VN_FillContainerExp_TcVnPoint3_LREAL



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnPoint3_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnPoint3_LREAL;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnPoint3_LREAL [▶ 152]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

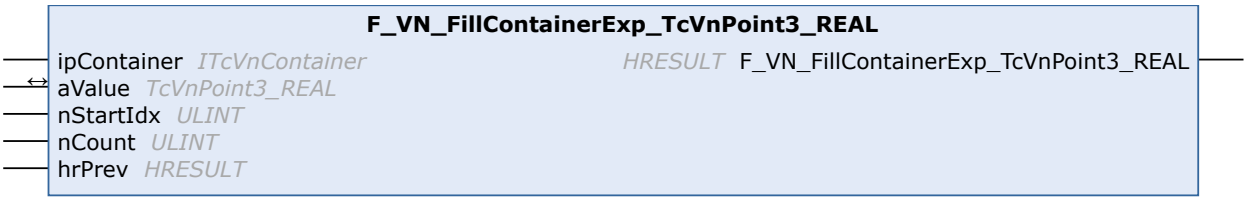
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.14 F_VN_FillContainerExp_TcVnPoint3_REAL



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnPoint3_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnPoint3_REAL;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnPoint3_REAL [▶ 152]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.15 F_VN_FillContainerExp_TcVnRectangle_DINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnRectangle_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stValue     : TcVnRectangle_DINT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stValue	TcVnRectangle_DINT [▶ 236]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.16 F_VN_FillContainerExp_TcVnVector2_DINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector2_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector2_DINT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_DINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

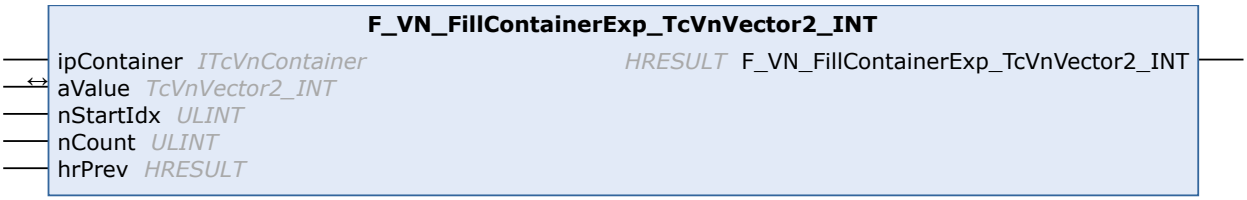
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.17 F_VN_FillContainerExp_TcVnVector2_INT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector2_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector2_INT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_INT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.18 F_VN_FillContainerExp_TcVnVector2_REAL



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector2_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector2_REAL;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_REAL [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.19 F_VN_FillContainerExp_TcVnVector2_SINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector2_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector2_SINT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_SINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.20 F_VN_FillContainerExp_TcVnVector2_UINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector2_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector2_UINT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_UINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

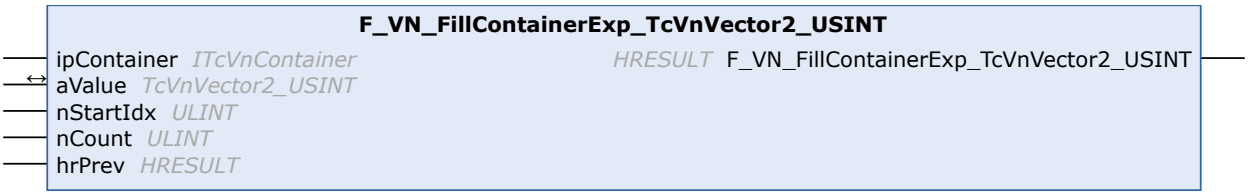
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.21 F_VN_FillContainerExp_TcVnVector2_USINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector2_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector2_USINT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector2_USINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

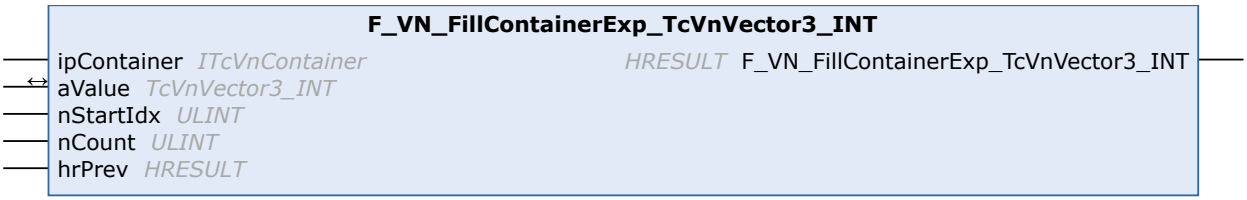
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.22 F_VN_FillContainerExp_TcVnVector3_INT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector3_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector3_INT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_INT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.23 F_VN_FillContainerExp_TcVnVector3_REAL



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector3_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector3_REAL;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_REAL [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.24 F_VN_FillContainerExp_TcVnVector3_SINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector3_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector3_SINT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_SINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.25 F_VN_FillContainerExp_TcVnVector3_UINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector3_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector3_UINT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_UINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

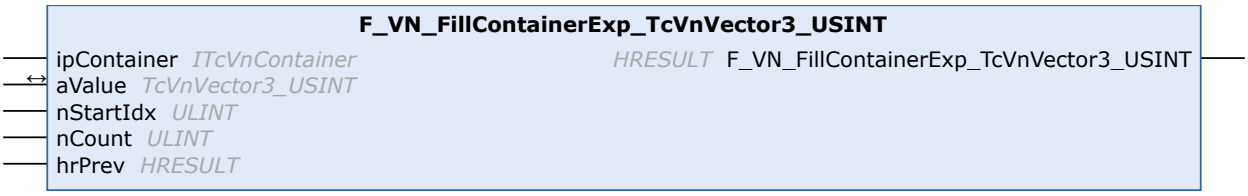
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.26 F_VN_FillContainerExp_TcVnVector3_USINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector3_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector3_USINT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector3_USINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.27 F_VN_FillContainerExp_TcVnVector4_DINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector4_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_DINT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_DINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

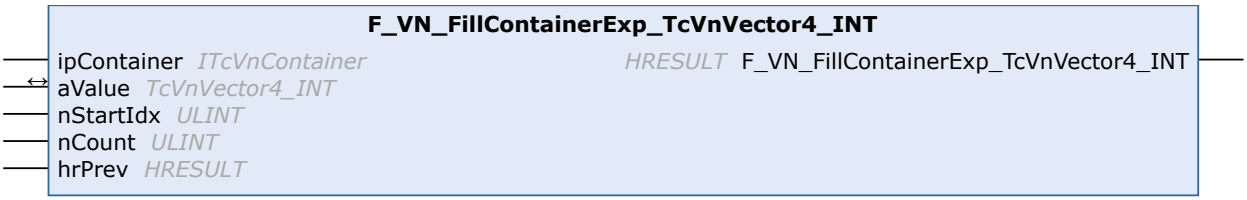
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.28 F_VN_FillContainerExp_TcVnVector4_INT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector4_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_INT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_INT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

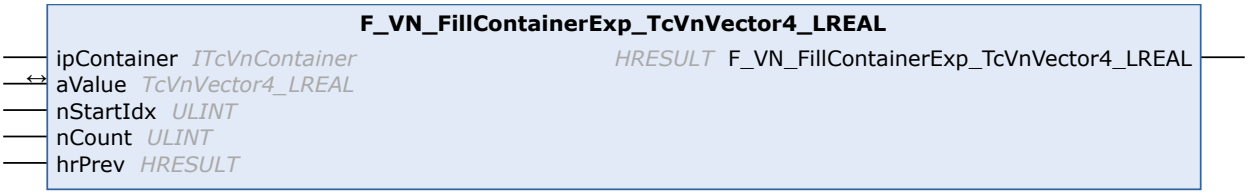
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.29 F_VN_FillContainerExp_TcVnVector4_LREAL



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector4_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_LREAL [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.30 F_VN_FillContainerExp_TcVnVector4_REAL



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector4_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_REAL;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_REAL [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.31 F_VN_FillContainerExp_TcVnVector4_SINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector4_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_SINT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_SINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.32 F_VN_FillContainerExp_TcVnVector4_UINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector4_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_UINT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_UINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.33 F_VN_FillContainerExp_TcVnVector4_USINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_TcVnVector4_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_USINT;
END_VAR
VAR_INPUT
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_USINT [▶ 153]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.34 F_VN_FillContainerExp_UDINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_UDINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nValue      : UDINT;
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nValue	UDINT	用于设置容器元素的值
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.35 F_VN_FillContainerExp_UINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nValue      : UINT;
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nValue	UINT	用于设置容器元素的值
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.36 F_VN_FillContainerExp_ULINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_ULINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nValue      : ULINT;
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nValue	ULINT	用于设置容器元素的值
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.3.37 F_VN_FillContainerExp_USINT



用指定值对容器进行部分填充。仅适用于具有基本元素的容器。（专家函数）

语法

定义：

```
FUNCTION F_VN_FillContainerExp_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nValue      : USINT;
    nStartIdx   : ULINT;
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nValue	USINT	用于设置容器元素的值
nStartIdx	ULINT	起始索引
nCount	ULINT	要设置的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4 F_VN_GetAt

如果需要访问一个容器中的单个元素，应该使用F_VN_GetAt函数。如果需要连续处理容器中的所有元素，使用迭代器 [▶ 764]更好。

6.1.4.5.4.1 F_VN_GetAt_TcVnRotatedRectangle



获取容器中指定索引处的 TcVnRotatedRectangle 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnRotatedRectangle : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stElement   : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnRotatedRectangle 元素的容器
nIndex	ULINT	请求元素的索引

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnRotatedRectangle [▶ 237]	返回指定索引处的元素

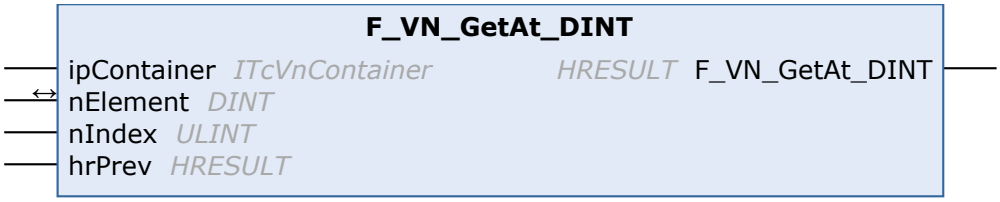
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.5.4.2 F_VN_GetAt_DINT



获取容器中指定索引处的 DINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nElement    : DINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 DINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nElement	DINT	返回指定索引处的元素

返回值

HRESULT [▶ 135]

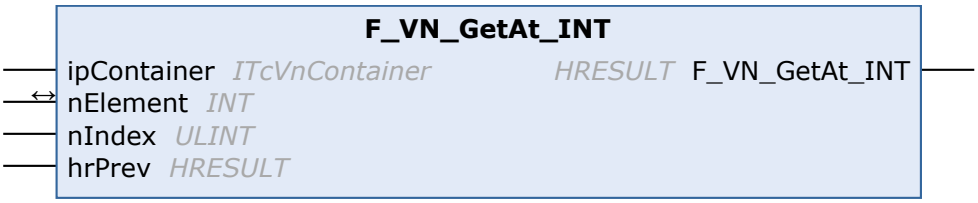
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.3 F_VN_GetAt_INT



获取容器中指定索引处的 INT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nElement    : INT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 INT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nElement	INT	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.4 F_VN_GetAt_ITcVnContainer



获取源容器中指定索引处的 ITcVnContainer 元素。（或者使用接口方法 .GetAt。）

语法

定义：

```
FUNCTION F_VN_GetAt_ITcVnContainer : HRESULT
VAR_INPUT
    ipSrcContainer  : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    nIndex          : ULINT;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	包含 ITcVnContainer 元素的容器
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回指定索引处的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.5 F_VN_GetAt_ITcVnImage



获取容器中指定索引处的 ITcVnImage 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_ITcVnImage : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    ipElement   : Reference To ITcVnImage;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 ITcVnImage 元素的容器
ipElement	引用 ITcVnImage [▶ 407]	返回指定索引处的元素
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

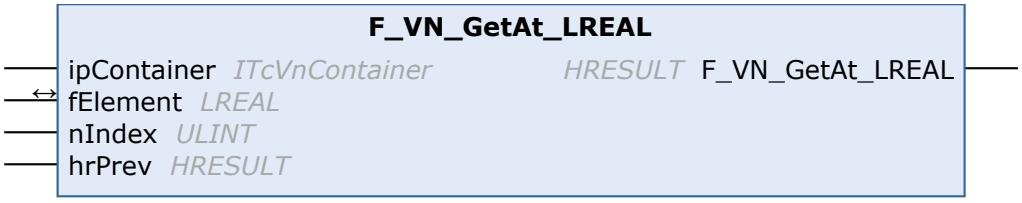
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.6 F_VN_GetAt_LREAL



获取容器中指定索引处的 LREAL 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fElement    : LREAL;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 LREAL 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fElement	LREAL	返回指定索引处的元素

返回值

HRESULT [▶ 135]

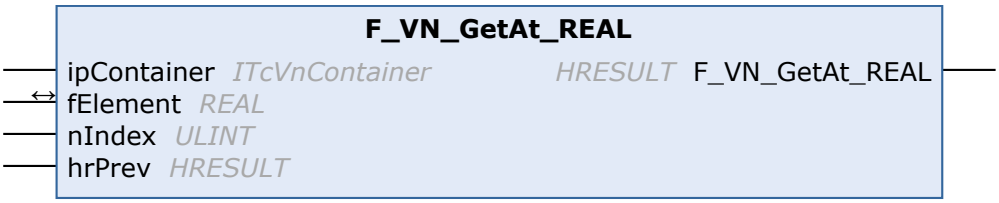
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.7 F_VN_GetAt_REAL



获取容器中指定索引处的 REAL 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fElement    : REAL;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 REAL 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fElement	REAL	返回指定索引处的元素

返回值

HRESULT [▶ 135]

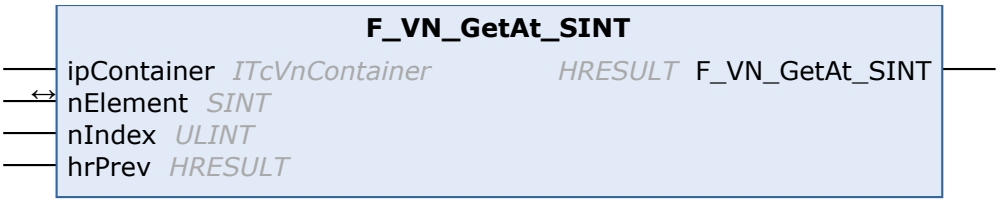
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.8 F_VN_GetAt_SINT



获取容器中指定索引处的 SINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nElement    : SINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	装有 SINT 元素的容器
nIndex	ULINT	请求元素的索引

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nElement	SINT	返回指定索引处的元素

返回值

HRESULT [▶ 135]

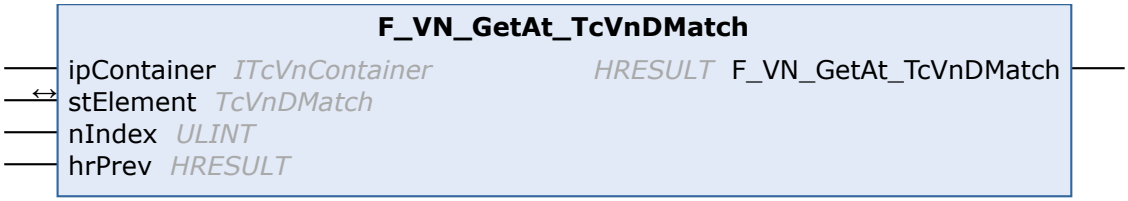
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.9 F_VN_GetAt_TcVnDMatch



获取容器中指定索引处的 TcVnDMatch 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnDMatch : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stElement   : TcVnDMatch;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnDMatch 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnDMatch [▶ 225]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.10 F_VN_GetAt_TcVnKeyPoint



获取容器中指定索引处的 TcVnKeyPoint 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnKeyPoint : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stElement   : TcVnKeyPoint;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnKeyPoint 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnKeyPoint [▶ 226]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.11 F_VN_GetAt_TcVnPoint2_DINT



获取容器中指定索引处的 TcVnPoint2_DINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnPoint2_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnPoint2_DINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint2_DINT [▶ 152]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.12 F_VN_GetAt_TcVnPoint2_LREAL



获取容器中指定索引处的 TcVnPoint2_LREAL 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnPoint2_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnPoint2_LREAL;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnPoint2_LREAL 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint2_LREAL [▶ 152]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.13 F_VN_GetAt_TcVnPoint2_REAL



获取容器中指定索引处的 TcVnPoint2_REAL 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnPoint2_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnPoint2_REAL 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint2_REAL [▶ 152]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.14 F_VN_GetAt_TcVnPoint3_LREAL



获取容器中指定索引处的 TcVnPoint3_LREAL 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnPoint3_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnPoint3_LREAL;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnPoint3_LREAL 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint3_LREAL [▶ 152]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.15 F_VN_GetAt_TcVnPoint3_REAL



获取容器中指定索引处的 TcVnPoint3_REAL 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnPoint3_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnPoint3_REAL;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnPoint3_REAL 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint3_REAL [▶ 152]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

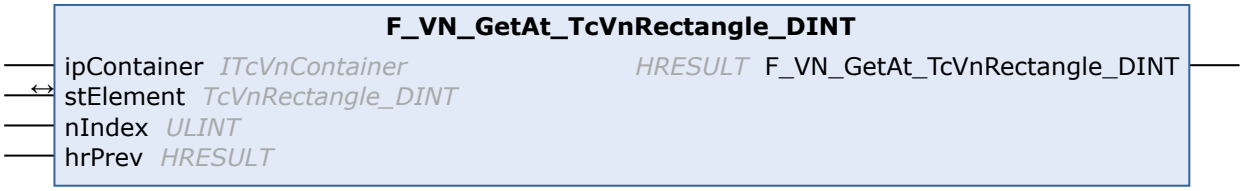
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.16 F_VN_GetAt_TcVnRectangle_DINT



获取容器中指定索引处的 TcVnRectangle_DINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnRectangle_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stElement   : TcVnRectangle_DINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnRectangle_DINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnRectangle_DINT [▶ 236]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.17 F_VN_GetAt_TcVnVector2_DINT



获取容器中指定索引处的 TcVnVector2_DINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector2_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector2_DINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_DINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_DINT [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.18 F_VN_GetAt_TcVnVector2_INT



获取容器中指定索引处的 TcVnVector2_INT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector2_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector2_INT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_INT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_INT [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.19 F_VN_GetAt_TcVnVector2_REAL



获取容器中指定索引处的 TcVnVector2_REAL 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector2_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector2_REAL;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_REAL 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_REAL [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

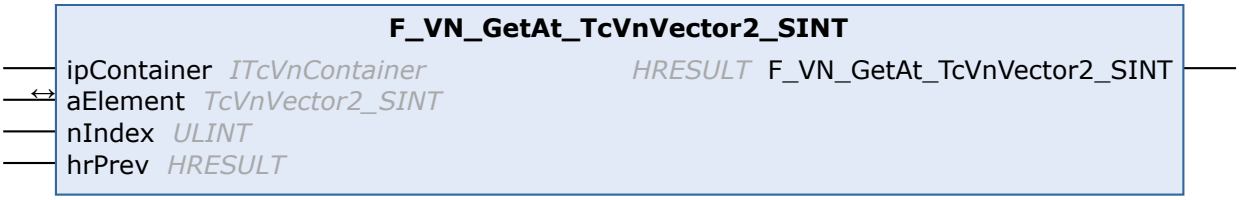
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.20 F_VN_GetAt_TcVnVector2_SINT



获取容器中指定索引处的 TcVnVector2_SINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector2_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector2_SINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_SINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_SINT [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

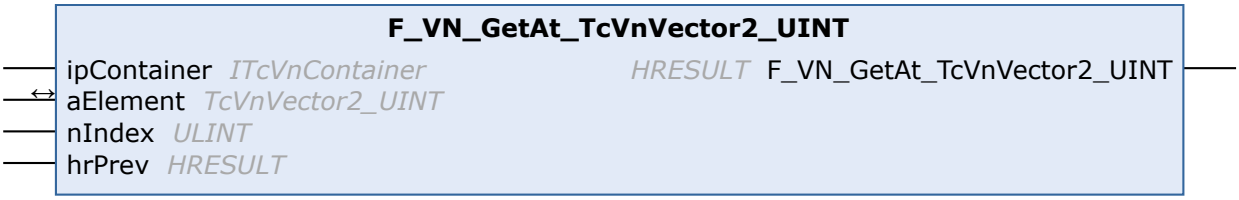
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.21 F_VN_GetAt_TcVnVector2_UINT



获取容器中指定索引处的 TcVnVector2_UINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector2_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector2_UINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_UINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_UINT [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

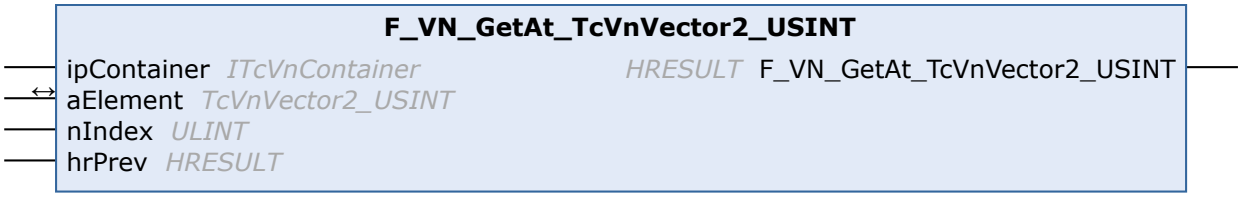
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.22 F_VN_GetAt_TcVnVector2_USINT



获取容器中指定索引处的 TcVnVector2_USINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector2_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector2_USINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_USINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_USINT [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.23 F_VN_GetAt_TcVnVector3_INT



获取容器中指定索引处的 TcVnVector3_INT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector3_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector3_INT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector3_INT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_INT [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.24 F_VN_GetAt_TcVnVector3_REAL



获取容器中指定索引处的 TcVnVector3_REAL 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector3_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector3_REAL;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector3_REAL 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_REAL [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

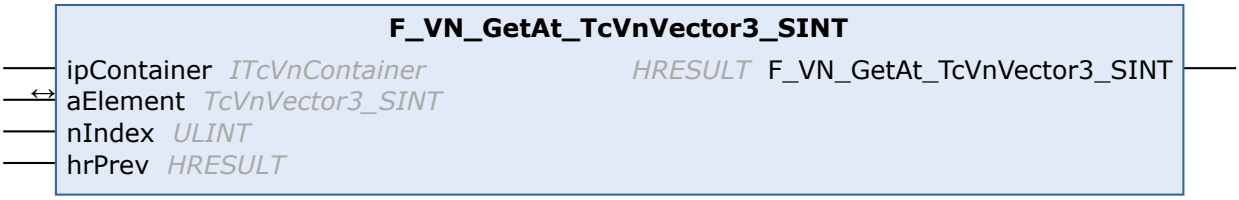
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.25 F_VN_GetAt_TcVnVector3_SINT



获取容器中指定索引处的 TcVnVector3_SINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector3_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector3_SINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector3_SINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_SINT [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

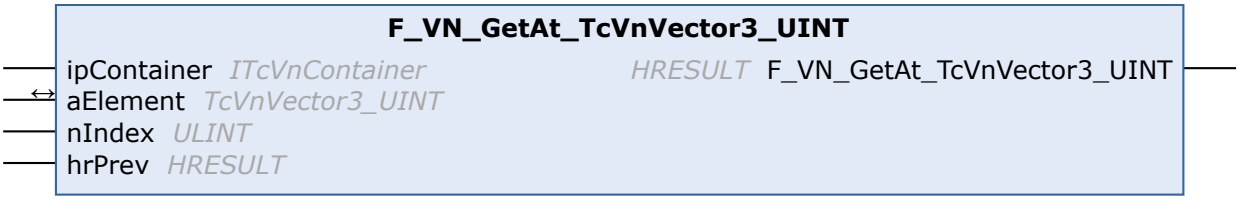
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.26 F_VN_GetAt_TcVnVector3_UINT



获取容器中指定索引处的 TcVnVector3_UINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector3_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector3_UINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector3_UINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_UINT [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

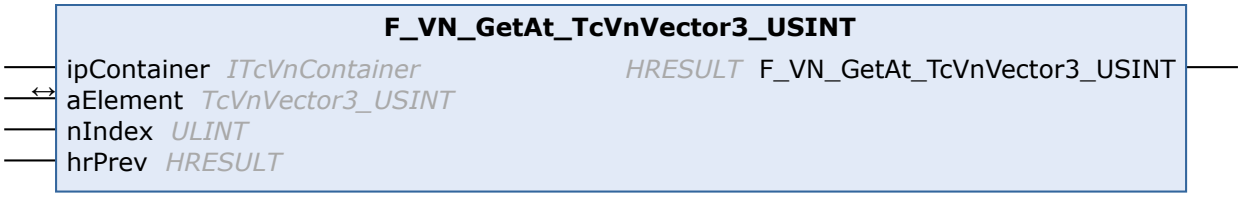
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.27 F_VN_GetAt_TcVnVector3_USINT



获取容器中指定索引处的 TcVnVector3_USINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector3_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector3_USINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector3_USINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_USINT [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.28 F_VN_GetAt_TcVnVector4_DINT



获取容器中指定索引处的 TcVnVector4_DINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector4_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector4_DINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_DINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_DINT [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.29 F_VN_GetAt_TcVnVector4_INT



获取容器中指定索引处的 TcVnVector4_INT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector4_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector4_INT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_INT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_INT [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

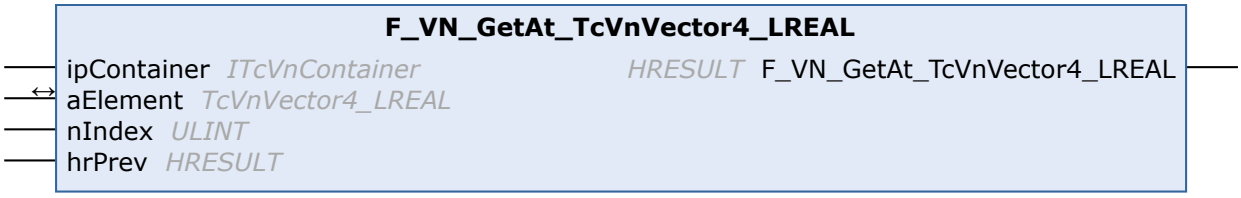
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.30 F_VN_GetAt_TcVnVector4_LREAL



获取容器中指定索引处的 TcVnVector4_LREAL 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector4_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_LREAL 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_LREAL [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

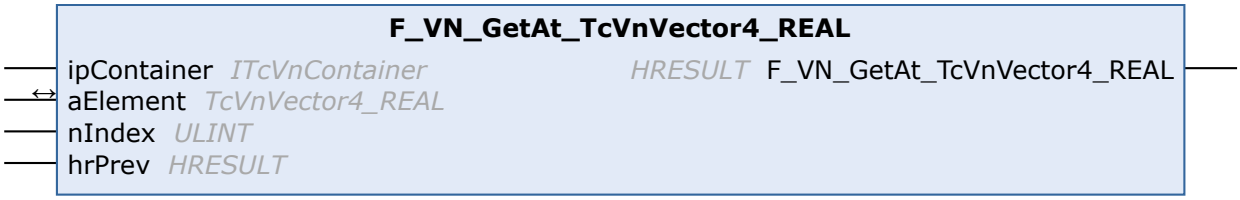
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.31 F_VN_GetAt_TcVnVector4_REAL



获取容器中指定索引处的 TcVnVector4_REAL 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector4_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector4_REAL;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_REAL 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_REAL [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

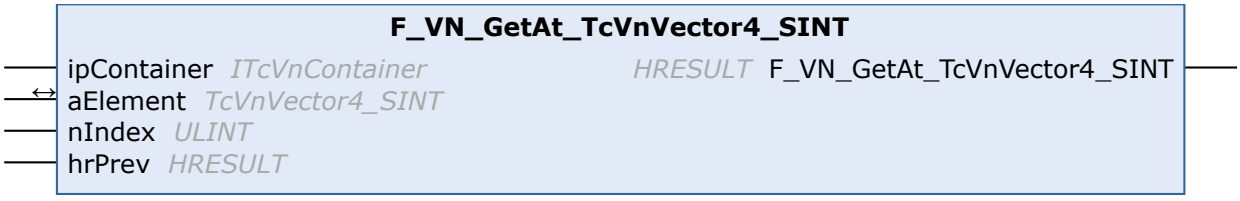
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.32 F_VN_GetAt_TcVnVector4_SINT



获取容器中指定索引处的 TcVnVector4_SINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector4_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector4_SINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_SINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_SINT [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

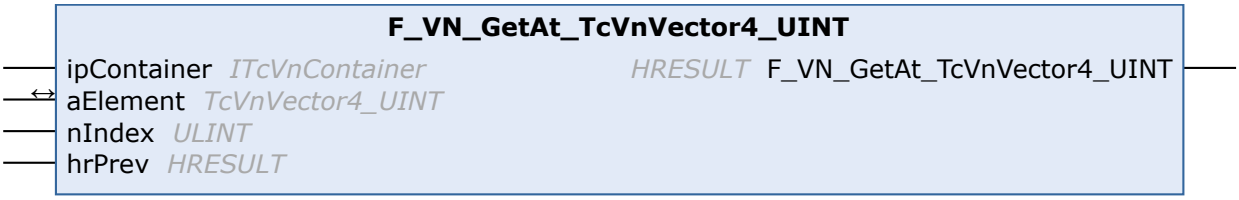
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.33 F_VN_GetAt_TcVnVector4_UINT



获取容器中指定索引处的 TcVnVector4_UINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector4_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector4_UINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_UINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_UINT [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

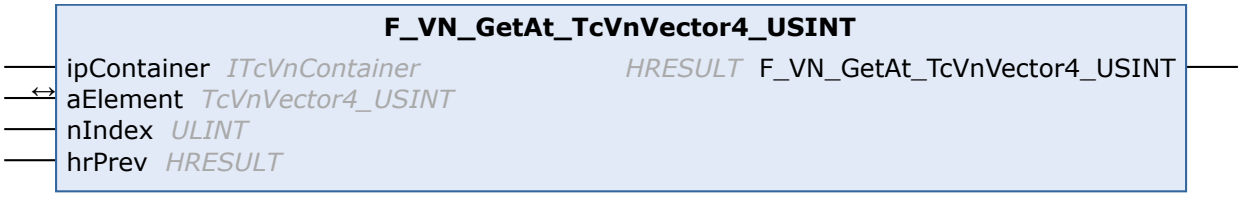
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.34 F_VN_GetAt_TcVnVector4_USINT



获取容器中指定索引处的 TcVnVector4_USINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_TcVnVector4_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aElement    : TcVnVector4_USINT;
END_VAR
VAR_INPUT
```

```
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_USINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_USINT [▶ 153]	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.35 F_VN_GetAt_UDINT



获取容器中指定索引处的 UDINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_UDINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nElement    : UDINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 UDINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nElement	UDINT	返回指定索引处的元素

返回值

HRESULT [▶ 135]

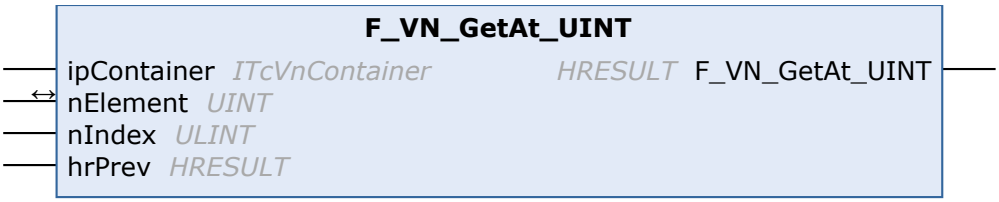
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.36 F_VN_GetAt_UINT



获取容器中指定索引处的 UINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nElement    : UINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 UINT 元素的容器
nIndex	ULINT	请求元素的索引

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nElement	UINT	返回指定索引处的元素

返回值

HRESULT [▶ 135]

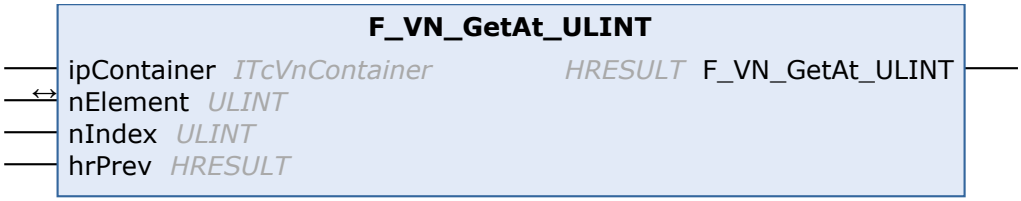
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.37 F_VN_GetAt_ULINT



获取容器中指定索引处的 ULINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_ULINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nElement    : ULINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 ULINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nElement	ULINT	返回指定索引处的元素

返回值

HRESULT [▶ 135]

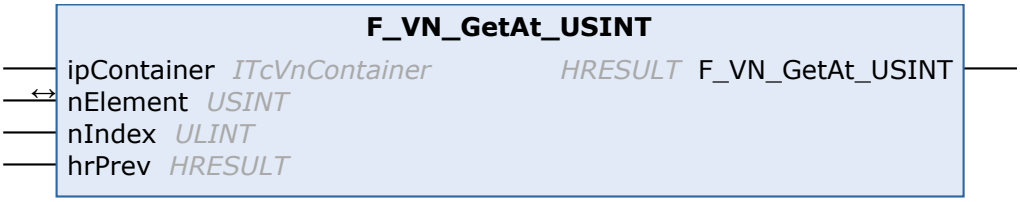
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.4.38 F_VN_GetAt_USINT



获取容器中指定索引处的 USINT 元素。

语法

定义：

```
FUNCTION F_VN_GetAt_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nElement    : USINT;
END_VAR
VAR_INPUT
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 USINT 元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nElement	USINT	返回指定索引处的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

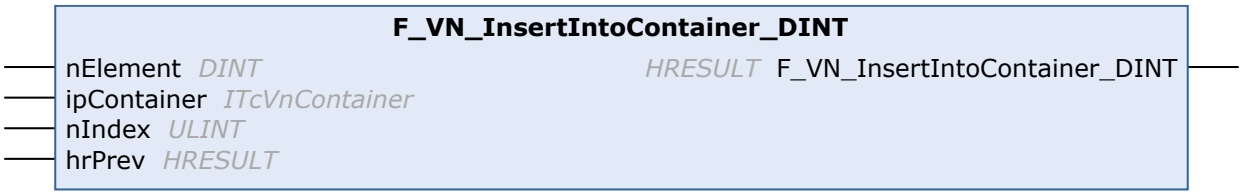
系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5 F_VN_InsertIntoContainer

本章包含根据数据类型在特定的容器位置插入一个元素的功能。

6.1.4.5.5.1 F_VN_InsertIntoContainer_DINT



将一个 DINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_DINT : HRESULT
VAR_INPUT
    nElement    : DINT;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	DINT	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.2 F_VN_InsertIntoContainer_INT



将一个 INT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_INT : HRESULT
VAR_INPUT
    nElement     : INT;
    ipContainer : ITcVnContainer;
    nIndex       : ULINT;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	INT	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

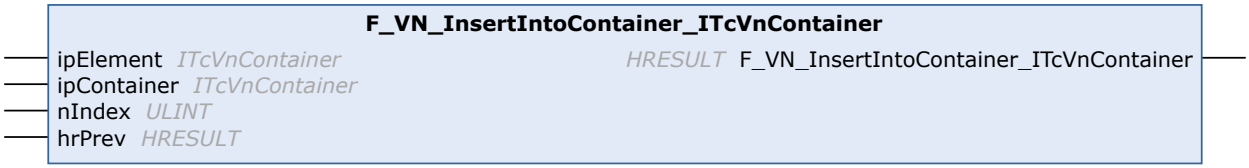
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.3 F_VN_InsertIntoContainer_ITcVnContainer



将一个元素（或一系列元素）插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_ITcVnContainer : HRESULT
VAR_INPUT
    ipElement   : ITcVnContainer;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipElement	ITcVnContainer [▶ 365]	插入到 ipContainer 中或包含多个元素的容器中的单个元素
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

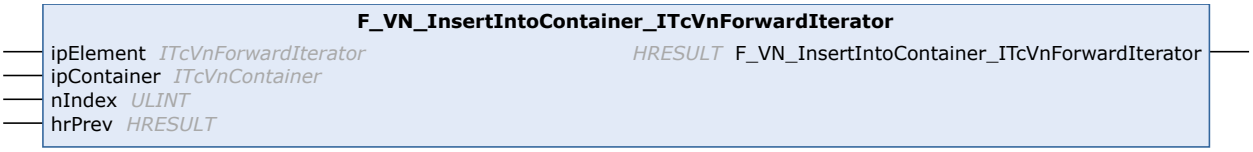
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.4 F_VN_InsertIntoContainer_ITcVnForwardIterator



将一个元素（由迭代器表示）插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_ITcVnForwardIterator : HRESULT
VAR_INPUT
    ipElement   : ITcVnForwardIterator;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipElement	ITcVnForwardIterator [▶ 355]	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

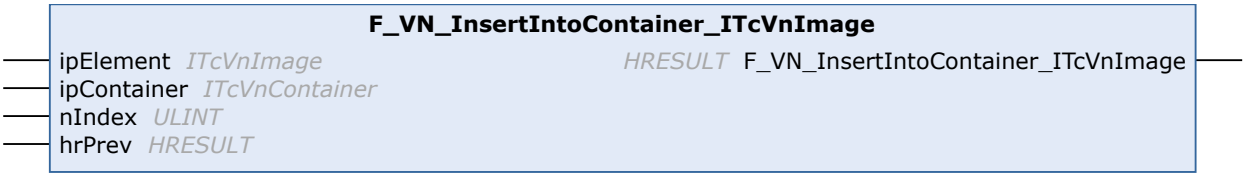
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.5 F_VN_InsertIntoContainer_ITcVnImage



将一个 ITcVnImage 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_ITcVnImage : HRESULT
VAR_INPUT
    ipElement   : ITcVnImage;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipElement	ITcVnImage [▶ 407]	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

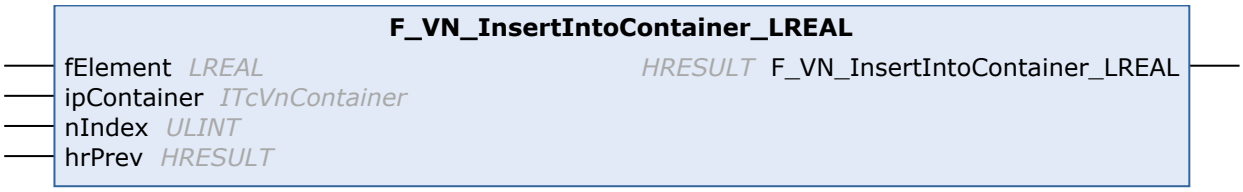
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.6 F_VN_InsertIntoContainer_LREAL



将一个 LREAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_LREAL : HRESULT
VAR_INPUT
    fElement     : LREAL;
    ipContainer : ITcVnContainer;
    nIndex       : ULINT;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
fElement	LREAL	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

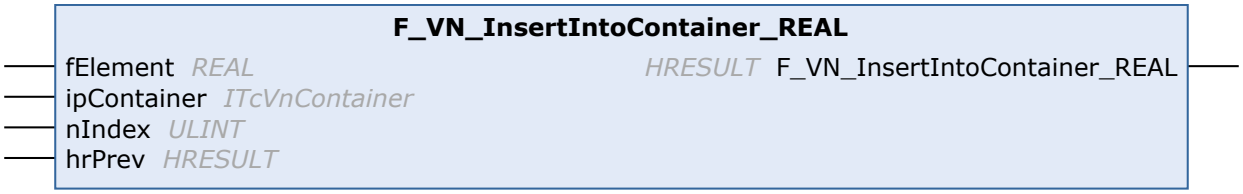
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.7 F_VN_InsertIntoContainer_REAL



将一个 REAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_REAL : HRESULT
VAR_INPUT
    fElement    : REAL;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fElement	REAL	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

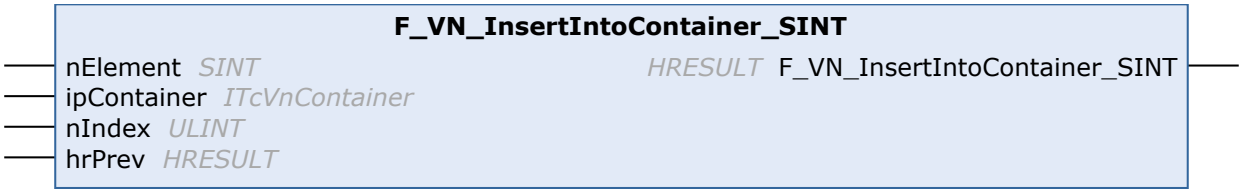
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.8 F_VN_InsertIntoContainer_SINT



将一个 SINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_SINT : HRESULT
VAR_INPUT
    nElement    : SINT;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	SINT	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

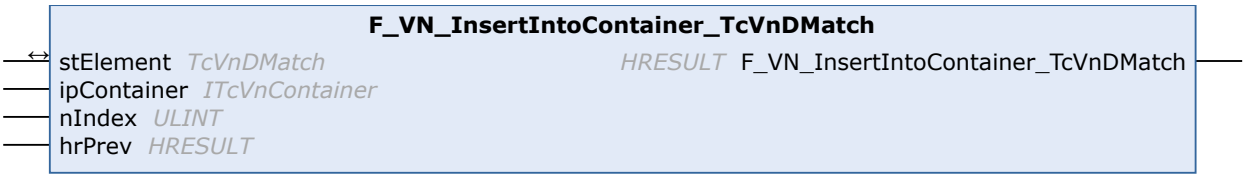
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.9 F_VN_InsertIntoContainer_TcVnDMatch



将一个 TcVnDMatch 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnDMatch : HRESULT
VAR_IN_OUT
    stElement   : TcVnDMatch;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnDMatch [▶ 225]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.10 F_VN_InsertIntoContainer_TcVnKeyPoint



将一个 TcVnKeyPoint 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnKeyPoint : HRESULT
VAR_IN_OUT
    stElement   : TcVnKeyPoint;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnKeyPoint [▶ 226]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

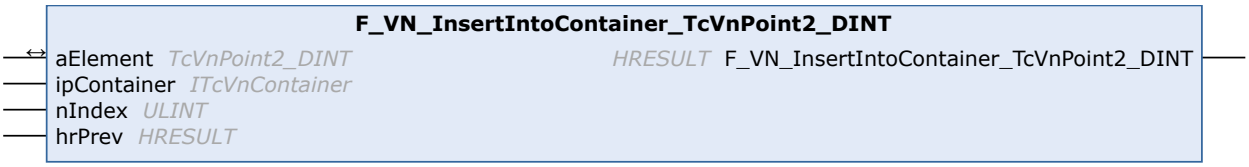
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.11 F_VN_InsertIntoContainer_TcVnPoint2_DINT



将一个 TcVnPoint2_DINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnPoint2_DINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint2_DINT [▶ 152]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

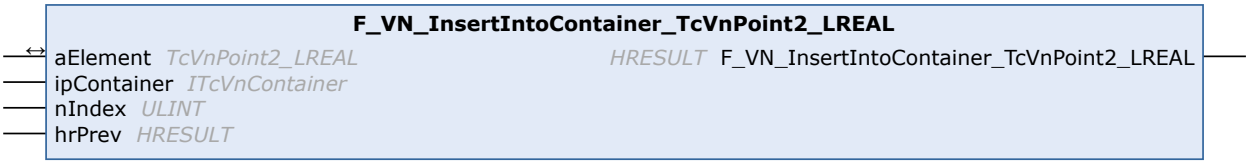
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.12 F_VN_InsertIntoContainer_TcVnPoint2_LREAL



将一个 TcVnPoint2_LREAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnPoint2_LREAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnPoint2_LREAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint2_LREAL [▶ 152]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

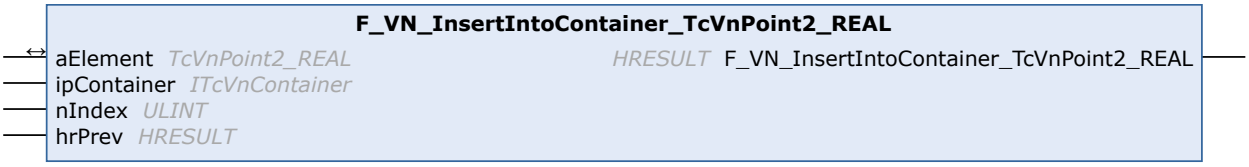
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.13 F_VN_InsertIntoContainer_TcVnPoint2_REAL



将一个 TcVnPoint2_REAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnPoint2_REAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint2_REAL [▶ 152]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

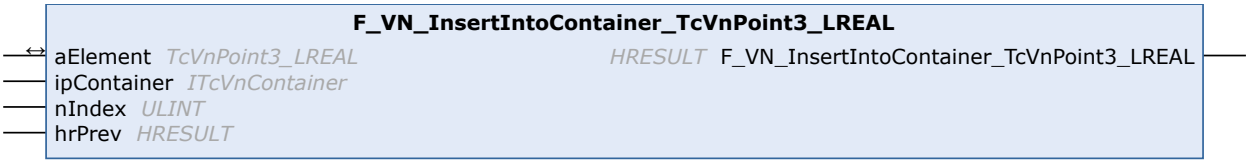
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.14 F_VN_InsertIntoContainer_TcVnPoint3_LREAL



将一个 TcVnPoint3_LREAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnPoint3_LREAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnPoint3_LREAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint3_LREAL [▶ 152]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

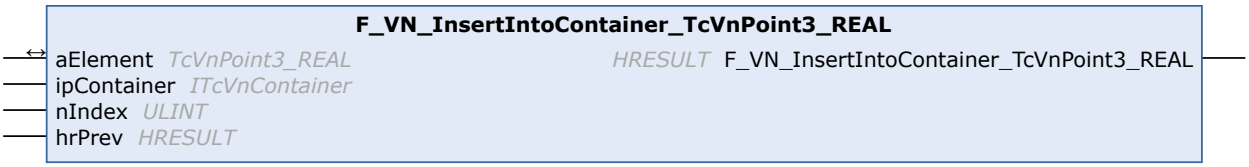
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.15 F_VN_InsertIntoContainer_TcVnPoint3_REAL



将一个 TcVnPoint3_REAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnPoint3_REAL : HRESULT
VAR_IN_OUT
    aElement    : TcVnPoint3_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint3_REAL [▶ 152]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

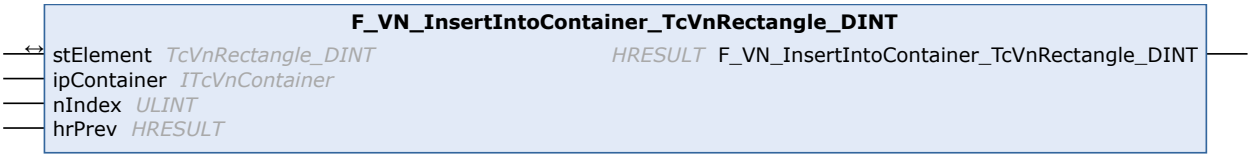
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.16 F_VN_InsertIntoContainer_TcVnRectangle_DINT



将一个 TcVnRectangle_DINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnRectangle_DINT : HRESULT
VAR_IN_OUT
    stElement   : TcVnRectangle_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnRectangle_DINT [▶ 236]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

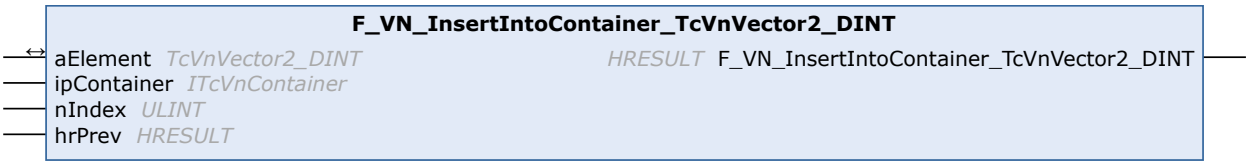
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.17 F_VN_InsertIntoContainer_TcVnVector2_DINT



将一个 TcVnVector2_DINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector2_DINT : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector2_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_DINT [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

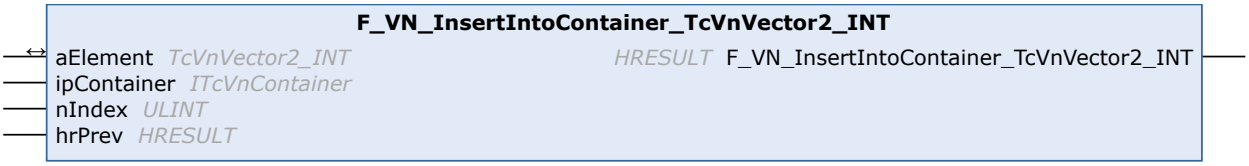
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.18 F_VN_InsertIntoContainer_TcVnVector2_INT



将一个 TcVnVector2_INT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector2_INT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector2_INT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_INT [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

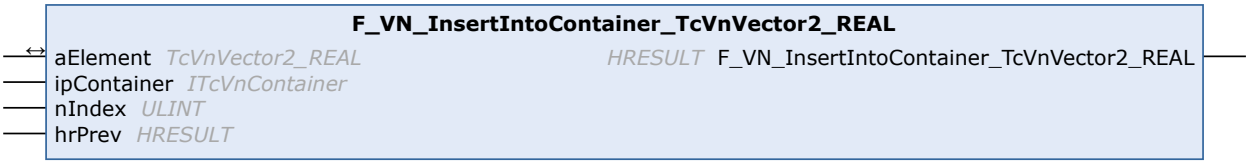
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.19 F_VN_InsertIntoContainer_TcVnVector2_REAL



将一个 TcVnVector2_REAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector2_REAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector2_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_REAL [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

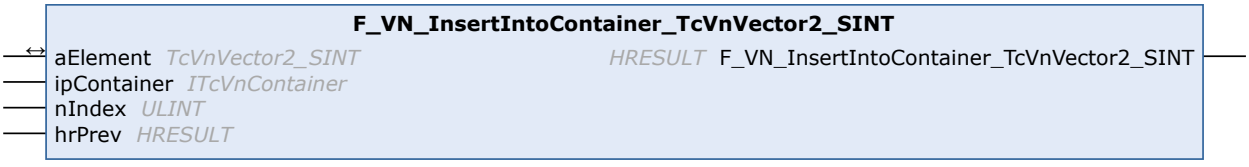
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.20 F_VN_InsertIntoContainer_TcVnVector2_SINT



将一个 TcVnVector2_SINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector2_SINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector2_SINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_SINT [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

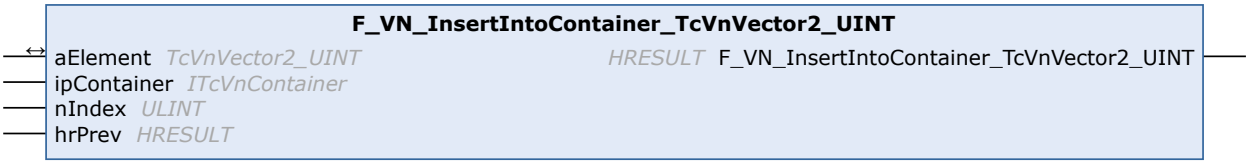
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.21 F_VN_InsertIntoContainer_TcVnVector2_UINT



将一个 TcVnVector2_UINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector2_UINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector2_UINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
```

```
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_UINT [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

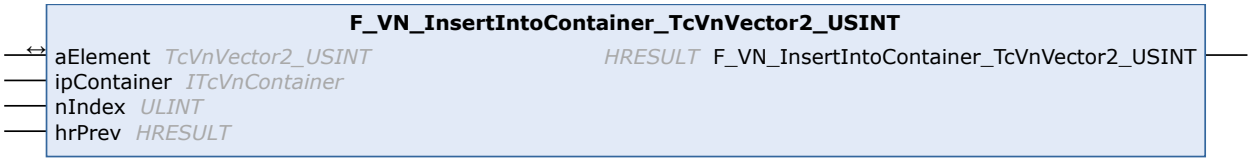
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.22 F_VN_InsertIntoContainer_TcVnVector2_USINT



将一个 TcVnVector2_USINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector2_USINT : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector2_USINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器

名称	类型	描述
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_USINT [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

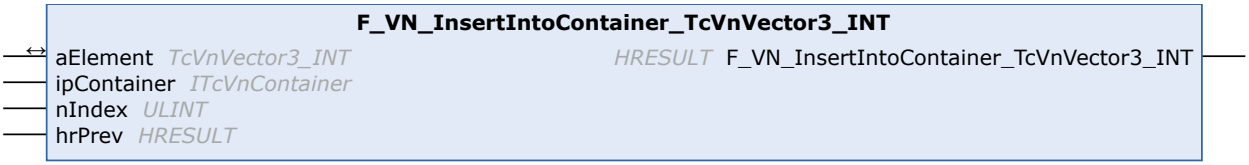
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.23 F_VN_InsertIntoContainer_TcVnVector3_INT



将一个 TcVnVector3_INT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector3_INT : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector3_INT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_INT [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

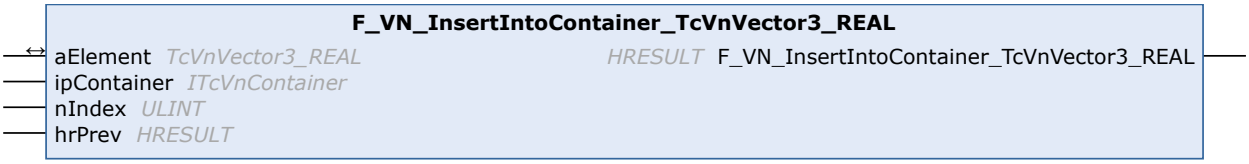
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.24 F_VN_InsertIntoContainer_TcVnVector3_REAL



将一个 TcVnVector3_REAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector3_REAL : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector3_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_REAL [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

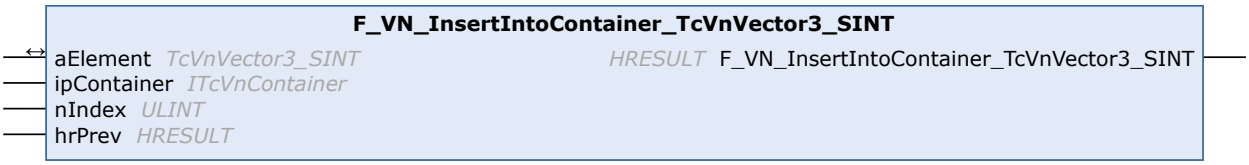
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.25 F_VN_InsertIntoContainer_TcVnVector3_SINT



将一个 TcVnVector3_SINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector3_SINT : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector3_SINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_SINT [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

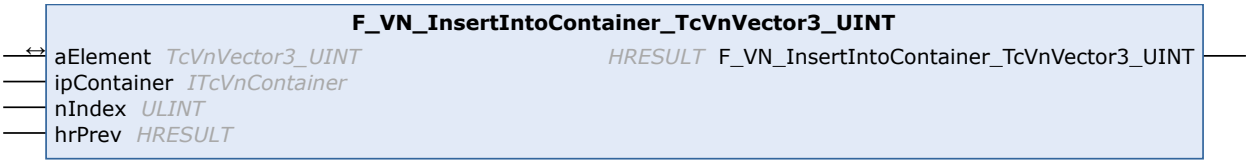
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.26 F_VN_InsertIntoContainer_TcVnVector3_UINT



将一个 TcVnVector3_UINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector3_UINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector3_UINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_UINT [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

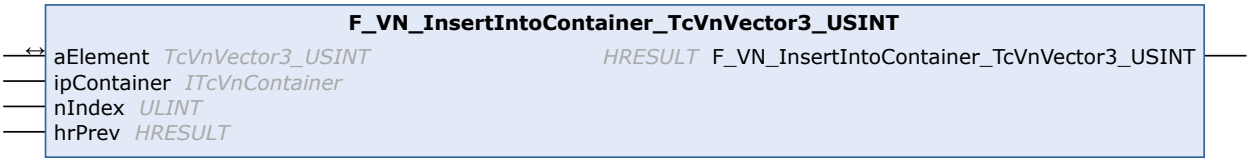
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.27 F_VN_InsertIntoContainer_TcVnVector3_USINT



将一个 TcVnVector3_USINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector3_USINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector3_USINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_USINT [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

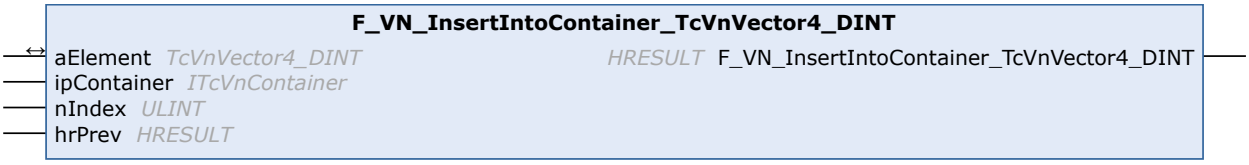
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.28 F_VN_InsertIntoContainer_TcVnVector4_DINT



将一个 TcVnVector4_DINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector4_DINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector4_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_DINT [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

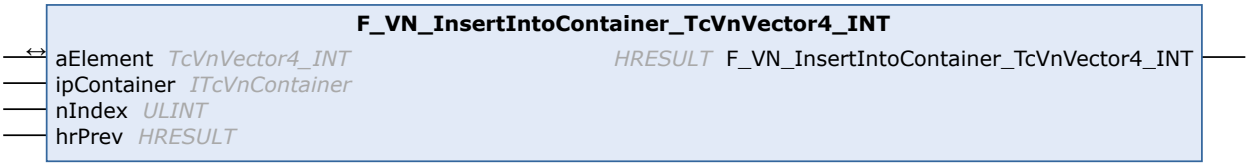
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.29 F_VN_InsertIntoContainer_TcVnVector4_INT



将一个 TcVnVector4_INT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector4_INT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector4_INT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
```

```
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_INT [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

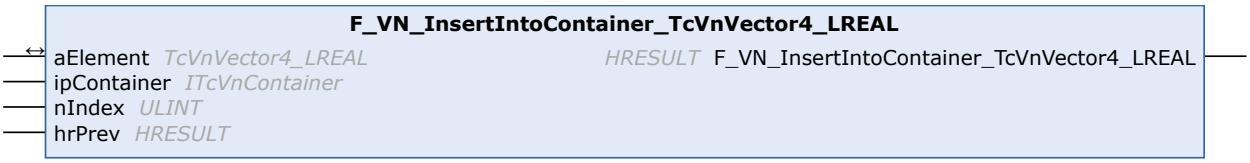
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.30 F_VN_InsertIntoContainer_TcVnVector4_LREAL



将一个 TcVnVector4_LREAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector4_LREAL : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_LREAL [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

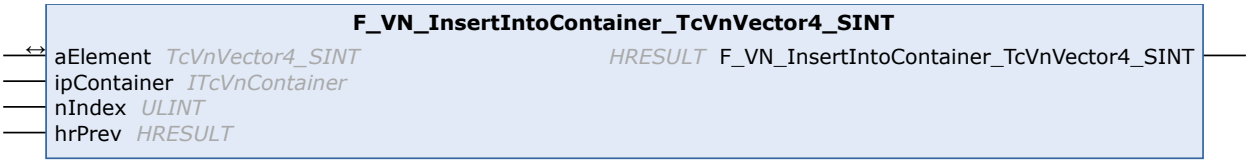
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.31 F_VN_InsertIntoContainer_TcVnVector4_SINT



将一个 TcVnVector4_SINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector4_SINT : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector4_SINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_SINT [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

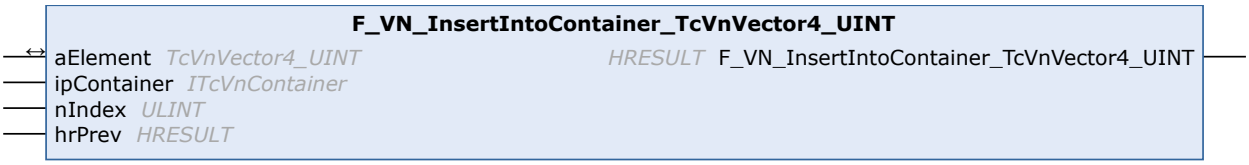
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.32 F_VN_InsertIntoContainer_TcVnVector4_UINT



将一个 TcVnVector4_UINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector4_UINT : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector4_UINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_UINT [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

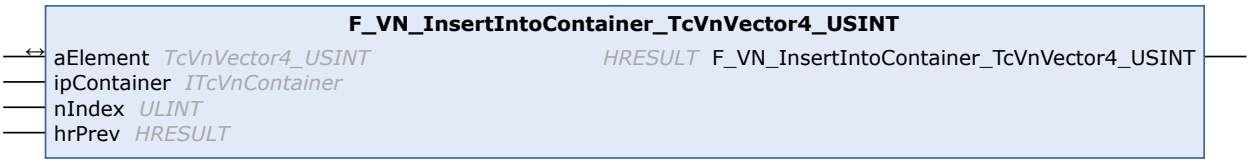
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.33 F_VN_InsertIntoContainer_TcVnVector4_USINT



将一个 TcVnVector4_USINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector4_USINT : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector4_USINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_USINT [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

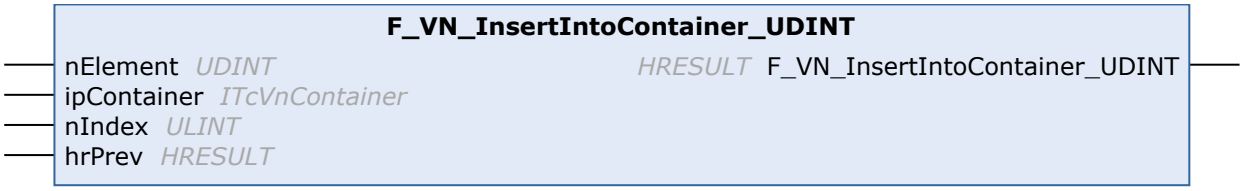
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.34 F_VN_InsertIntoContainer_UDINT



将一个 UDINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_UDINT : HRESULT
VAR_INPUT
    nElement     : UDINT;
    ipContainer : ITcVnContainer;
    nIndex       : ULINT;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	UDINT	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

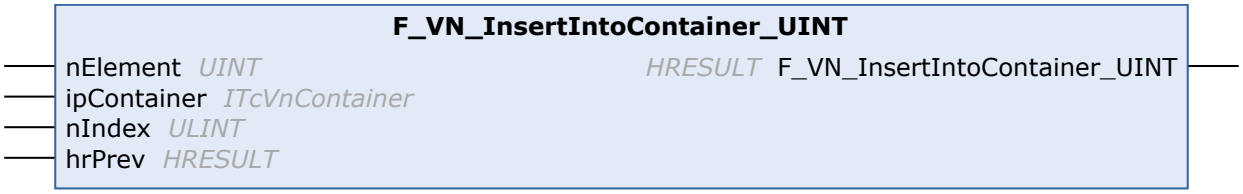
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.35 F_VN_InsertIntoContainer_UINT



将一个 UINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_UINT : HRESULT
VAR_INPUT
    nElement    : UINT;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	UINT	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

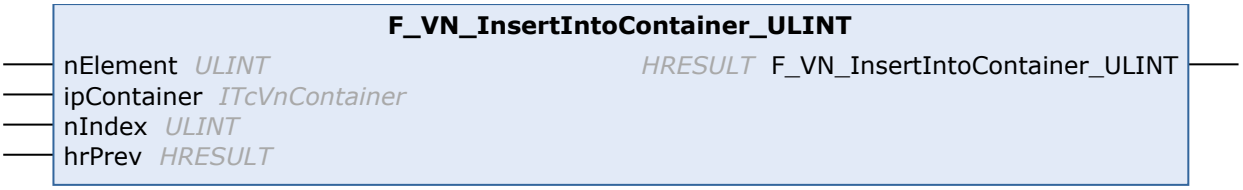
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.36 F_VN_InsertIntoContainer_ULINT



将一个 ULINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_ULINT : HRESULT
VAR_INPUT
    nElement    : ULINT;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	ULINT	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

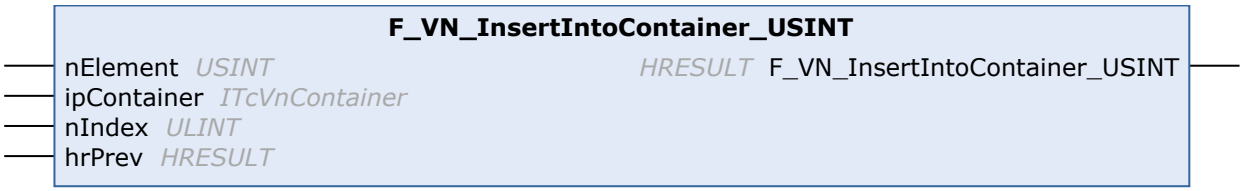
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.37 F_VN_InsertIntoContainer_USINT



将一个 USINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_USINT : HRESULT
VAR_INPUT
    nElement    : USINT;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	USINT	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

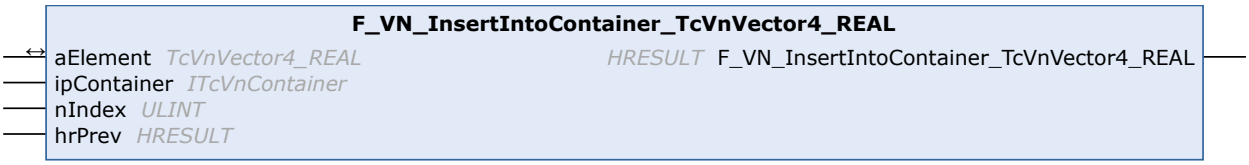
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.38 F_VN_InsertIntoContainer_TcVnVector4_REAL



将一个 TcVnVector4_REAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnVector4_REAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector4_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_REAL [▶ 153]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

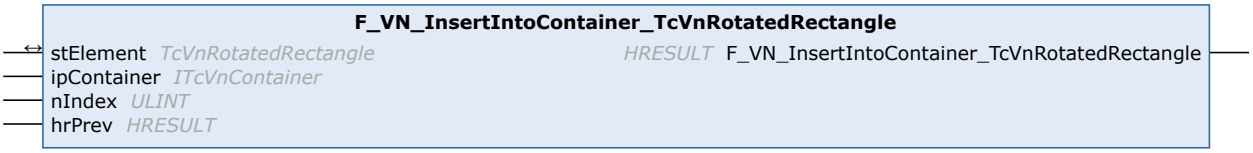
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.5.39 F_VN_InsertIntoContainer_TcVnRotatedRectangle



将一个 TcVnRotatedRectangle 类型的元素插入容器中的指定位置之前。

语法

定义：

```
FUNCTION F_VN_InsertIntoContainer_TcVnRotatedRectangle : HRESULT
VAR_IN_OUT
    stElement   : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	向其中插入元素的容器
nIndex	ULINT	在其之前插入元素的位置
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnRotatedRectangle [▶ 237]	插入到 ipContainer 中的单个元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

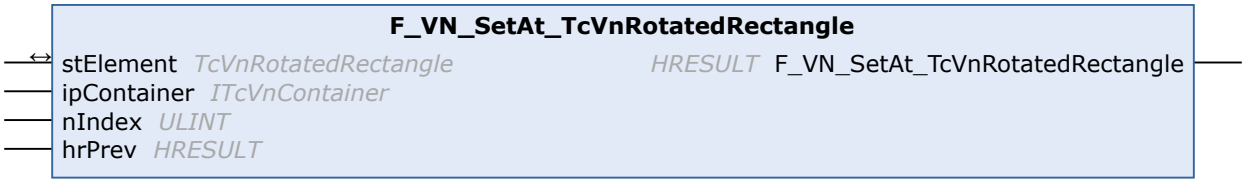
系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6 F_VN_SetAt

如果需要访问一个容器中的单个元素，应该使用F_VN_SetAt函数。如果需要连续处理容器中的所有元素，使用迭代器 [▶ 764]更好。

6.1.4.5.6.1 F_VN_SetAt_TcVnRotatedRectangle



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnRotatedRectangle : HRESULT
VAR_IN_OUT
    stElement   : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnRotatedRectangle 元素的容器，其中 nIndex 位置的元素由 stElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnRotatedRectangle [▶ 237]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.2 F_VN_SetAt_DINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_DINT : HRESULT
VAR_INPUT
    nElement    : DINT;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	DINT	要设置在指定容器位置的元素
ipContainer	ITcVnContainer [▶ 365]	包含 DINT 元素的容器，其中 nIndex 位置的元素由 nElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

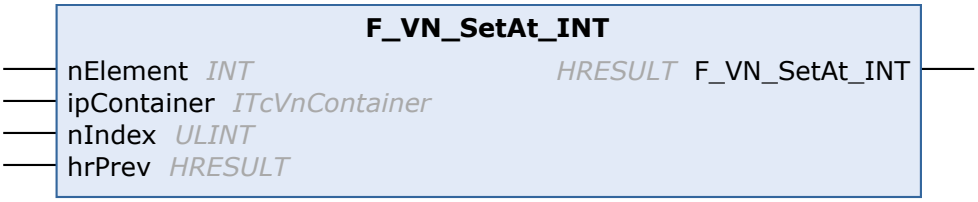
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.3 F_VN_SetAt_INT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_INT : HRESULT
VAR_INPUT
    nElement    : INT;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	INT	要设置在指定容器位置的元素
ipContainer	ITcVnContainer [▶ 365]	包含 INT 元素的容器，其中 nIndex 位置的元素由 nElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

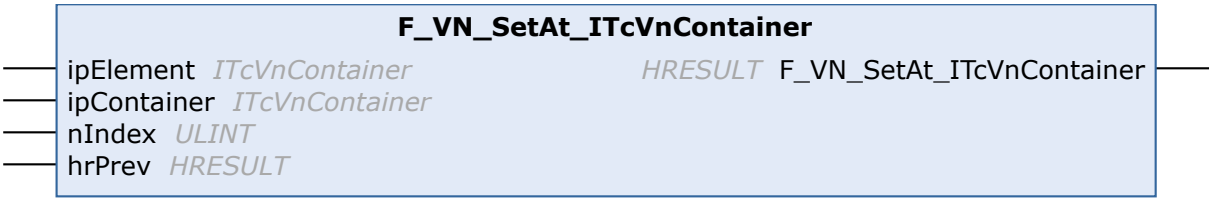
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.4 F_VN_SetAt_ITcVnContainer



设置容器中指定索引处的元素。（或者使用接口方法 .SetAt。）

语法

定义：

```
FUNCTION F_VN_SetAt_ITcVnContainer : HRESULT
VAR_INPUT
    ipElement   : ITcVnContainer;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipElement	ITcVnContainer [▶ 365]	要设置在指定容器位置的元素
ipContainer	ITcVnContainer [▶ 365]	包含 ITcVnContainer 元素的容器，其中 nIndex 位置的元素由 ipElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

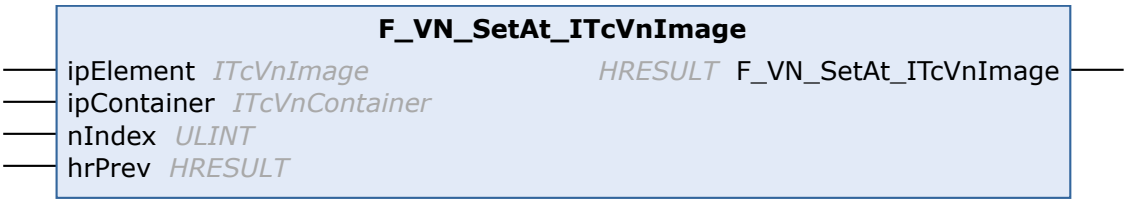
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.5 F_VN_SetAt_ITcVnImage



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_ITcVnImage : HRESULT
VAR_INPUT
    ipElement   : ITcVnImage;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipElement	ITcVnImage [▶ 407]	要设置在指定容器位置的元素
ipContainer	ITcVnContainer [▶ 365]	包含 ITcVnImage 元素的容器，其中 nIndex 位置的元素由 ipElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

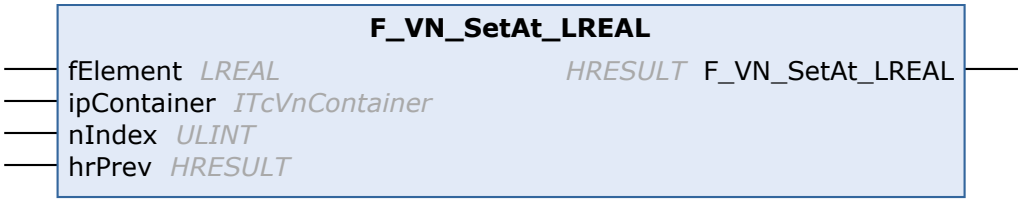
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.6 F_VN_SetAt_LREAL



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_LREAL : HRESULT
VAR_INPUT
    fElement    : LREAL;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fElement	LREAL	要设置在指定容器位置的元素
ipContainer	ITcVnContainer [▶ 365]	包含 LREAL 元素的容器，其中 nIndex 位置的元素由 fElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

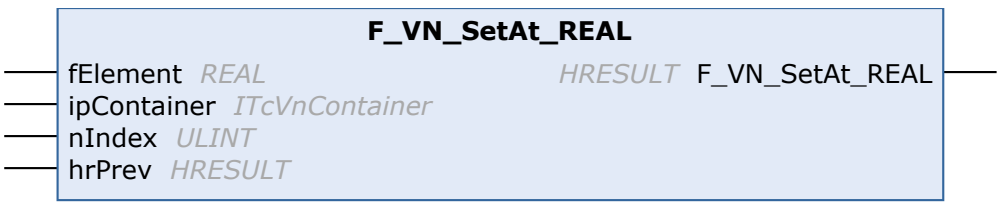
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.7 F_VN_SetAt_REAL



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_REAL : HRESULT
VAR_INPUT
    fElement    : REAL;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
fElement	REAL	要设置在指定容器位置的元素
ipContainer	ITcVnContainer [▶ 365]	包含 REAL 元素的容器，其中 nIndex 位置的元素由 fElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

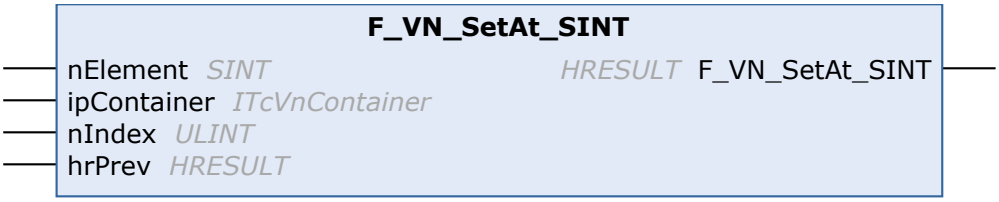
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.8 F_VN_SetAt_SINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_SINT : HRESULT
VAR_INPUT
    nElement    : SINT;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	SINT	要设置在指定容器位置的元素
ipContainer	ITcVnContainer [▶ 365]	包含 SINT 元素的容器，其中 nIndex 位置的元素由 nElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

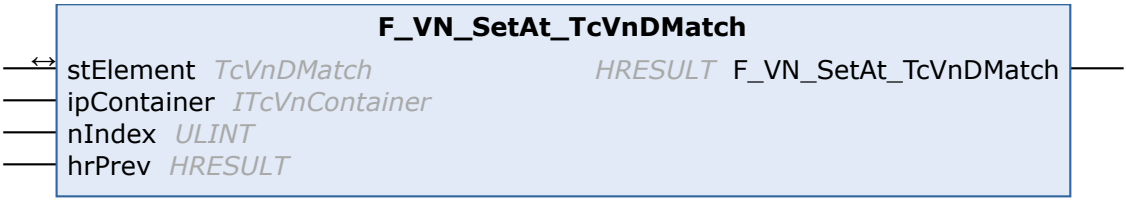
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.9 F_VN_SetAt_TcVnDMatch



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnDMatch : HRESULT
VAR_IN_OUT
    stElement   : TcVnDMatch;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnDMatch 元素的容器，其中 nIndex 位置的元素由 stElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnDMatch [▶ 225]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.10 F_VN_SetAt_TcVnKeyPoint



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnKeyPoint : HRESULT
VAR_IN_OUT
    stElement   : TcVnKeyPoint;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnKeyPoint 元素的容器，其中 nIndex 位置的元素由 stElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnKeyPoint [▶ 226]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.11 F_VN_SetAt_TcVnPoint2_DINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnPoint2_DINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnPoint2_DINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint2_DINT [▶ 152]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

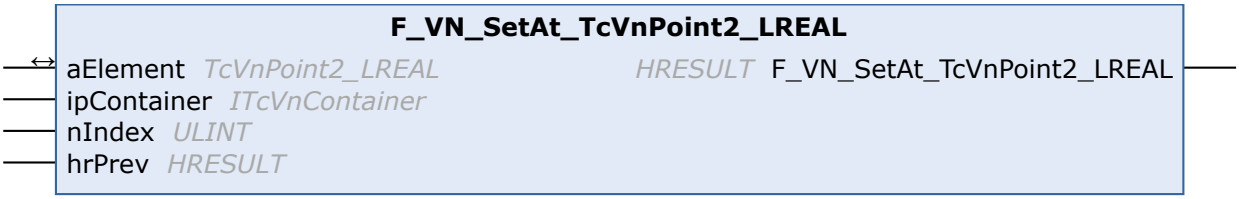
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.12 F_VN_SetAt_TcVnPoint2_LREAL



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnPoint2_LREAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnPoint2_LREAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnPoint2_LREAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint2_LREAL [▶ 152]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.13 F_VN_SetAt_TcVnPoint2_REAL



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnPoint2_REAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnPoint2_REAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint2_REAL [▶ 152]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

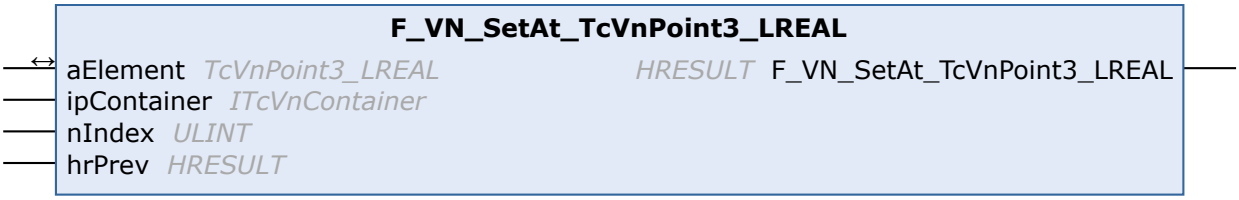
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.14 F_VN_SetAt_TcVnPoint3_LREAL



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnPoint3_LREAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnPoint3_LREAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnPoint3_LREAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint3_LREAL [▶ 152]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.15 F_VN_SetAt_TcVnPoint3_REAL



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnPoint3_REAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnPoint3_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnPoint3_REAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnPoint3_REAL [▶ 152]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

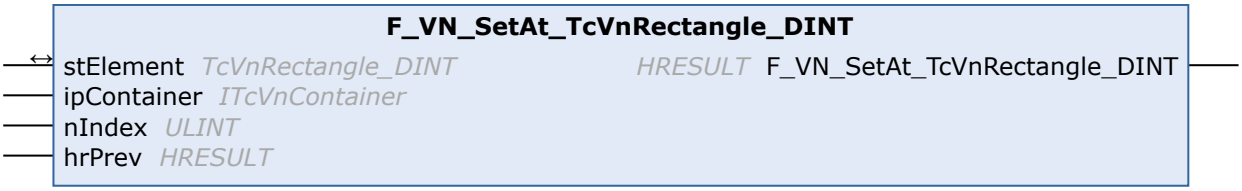
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.16 F_VN_SetAt_TcVnRectangle_DINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnRectangle_DINT : HRESULT
VAR_IN_OUT
    stElement    : TcVnRectangle_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnRectangle_DINT 元素的容器，其中 nIndex 位置的元素由 stElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stElement	TcVnRectangle_DINT [▶ 236]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

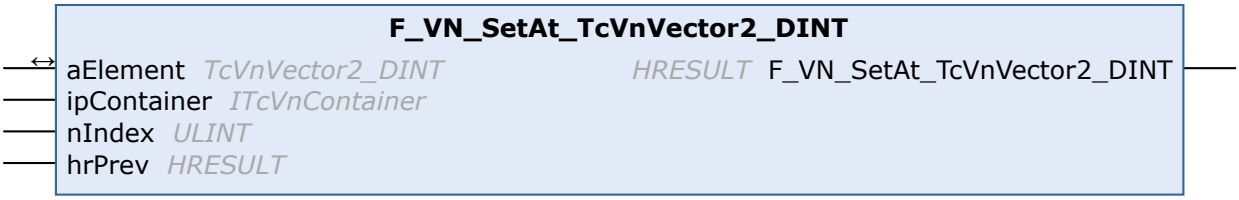
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.17 F_VN_SetAt_TcVnVector2_DINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector2_DINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector2_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_DINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_DINT [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

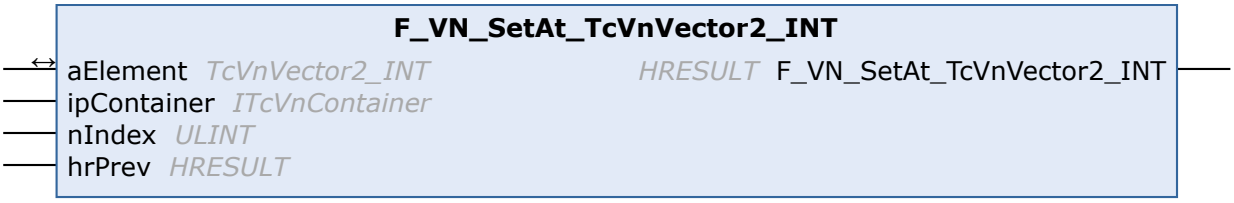
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.18 F_VN_SetAt_TcVnVector2_INT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector2_INT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector2_INT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_INT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_INT [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

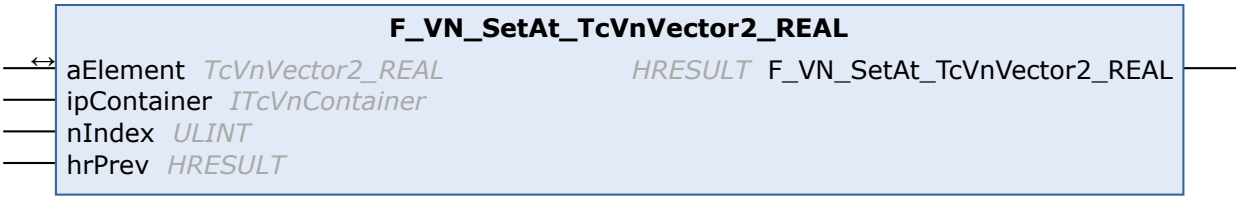
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.19 F_VN_SetAt_TcVnVector2_REAL



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector2_REAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector2_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_REAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_REAL [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

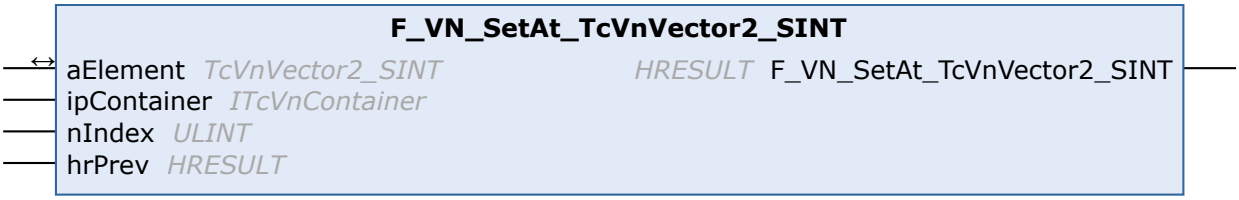
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.20 F_VN_SetAt_TcVnVector2_SINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector2_SINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector2_SINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_SINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_SINT [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

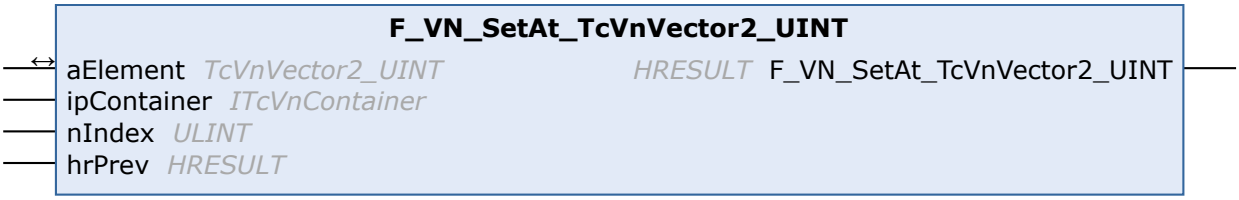
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.21 F_VN_SetAt_TcVnVector2_UINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector2_UINT : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector2_UINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_UINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_UINT [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

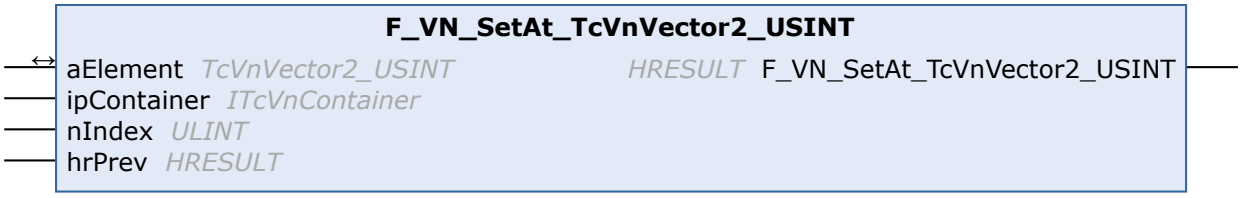
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.22 F_VN_SetAt_TcVnVector2_USINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector2_USINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector2_USINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector2_USINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector2_USINT [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

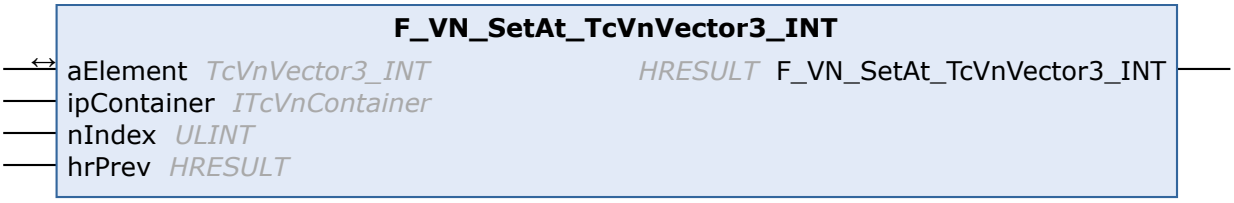
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.23 F_VN_SetAt_TcVnVector3_INT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector3_INT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector3_INT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector3_INT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_INT [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

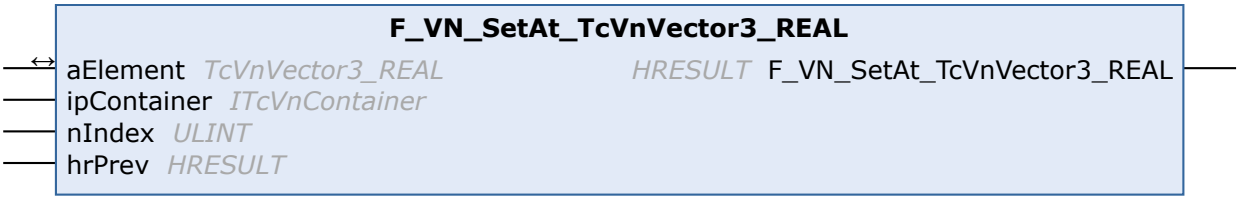
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.24 F_VN_SetAt_TcVnVector3_REAL



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector3_REAL : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector3_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector3_REAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_REAL [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

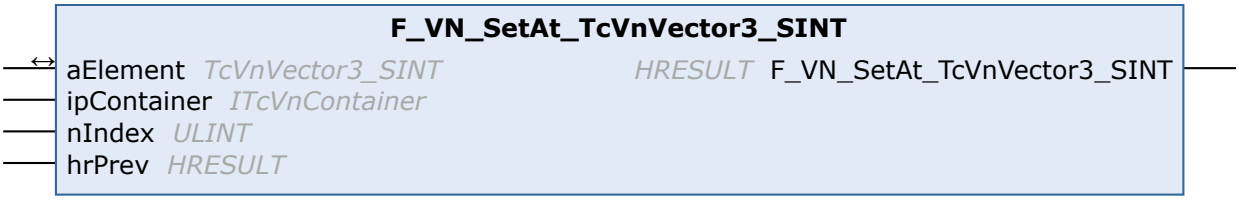
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.25 F_VN_SetAt_TcVnVector3_SINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector3_SINT : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector3_SINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector3_SINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_SINT [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

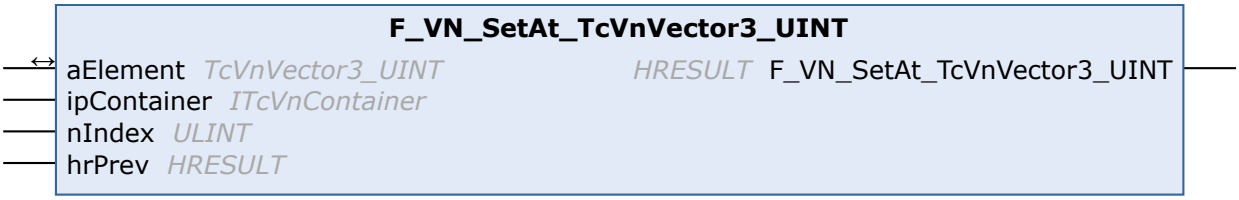
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.26 F_VN_SetAt_TcVnVector3_UINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector3_UINT : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector3_UINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector3_UINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_UINT [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

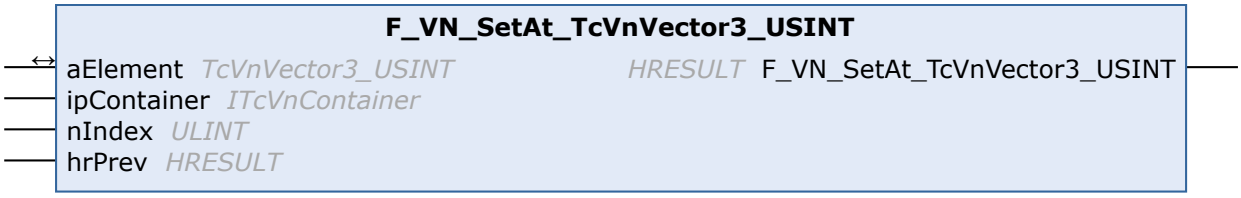
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.27 F_VN_SetAt_TcVnVector3_USINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector3_USINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector3_USINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector3_USINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector3_USINT [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

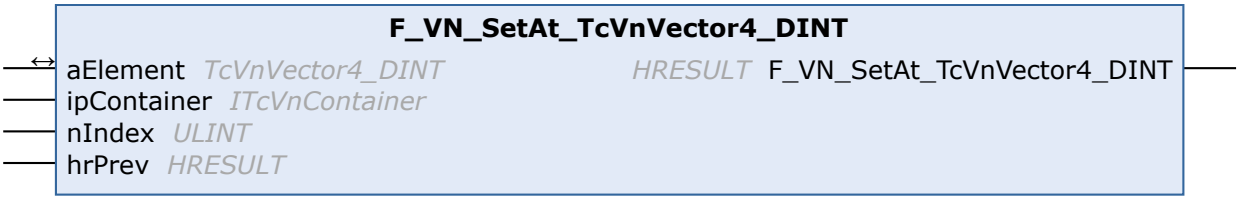
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.28 F_VN_SetAt_TcVnVector4_DINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector4_DINT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector4_DINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_DINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_DINT [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

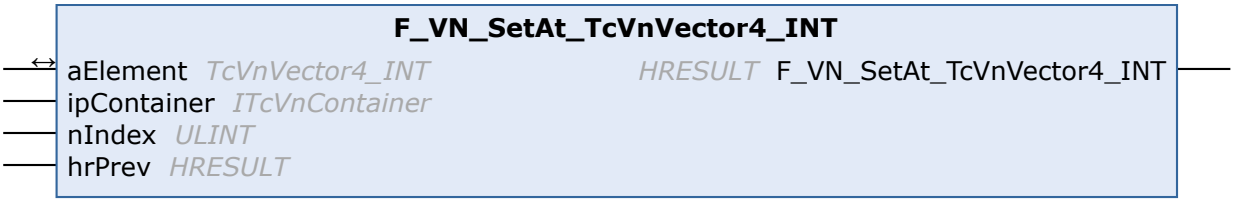
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.29 F_VN_SetAt_TcVnVector4_INT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector4_INT : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector4_INT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_INT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_INT [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

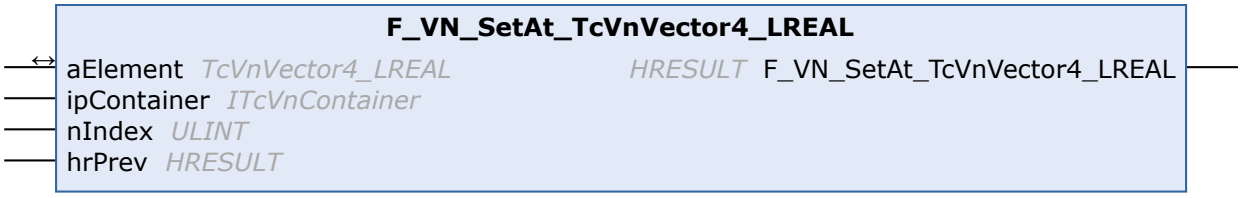
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.30 F_VN_SetAt_TcVnVector4_LREAL



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector4_LREAL : HRESULT
VAR_IN_OUT
    aElement     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_LREAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_LREAL [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

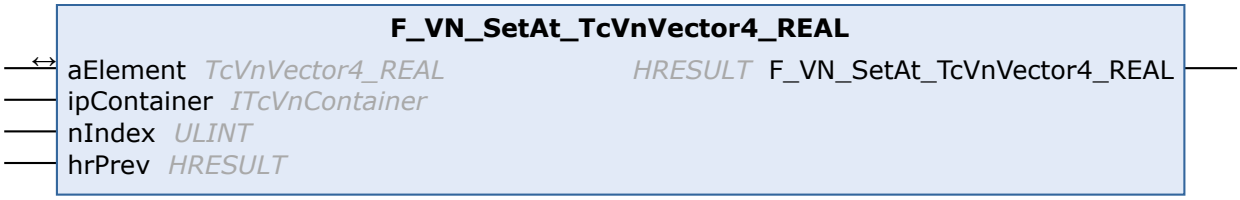
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.31 F_VN_SetAt_TcVnVector4_REAL



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector4_REAL : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector4_REAL;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_REAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_REAL [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

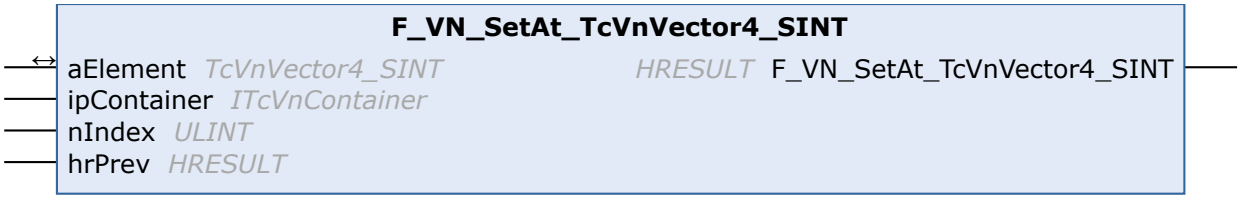
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.32 F_VN_SetAt_TcVnVector4_SINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector4_SINT : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector4_SINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_SINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_SINT [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

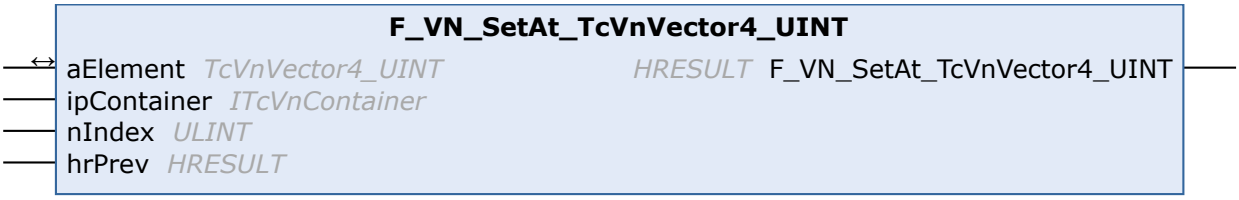
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.33 F_VN_SetAt_TcVnVector4_UINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector4_UINT : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector4_UINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_UINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_UINT [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

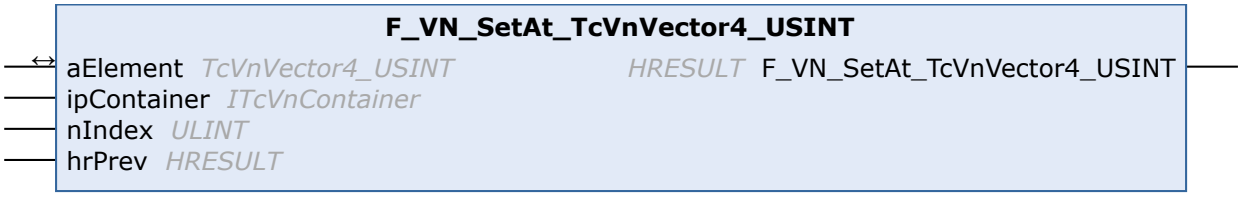
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.34 F_VN_SetAt_TcVnVector4_USINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_TcVnVector4_USINT : HRESULT
VAR_IN_OUT
    aElement    : TcVnVector4_USINT;
END_VAR
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含 TcVnVector4_USINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aElement	TcVnVector4_USINT [▶ 153]	要设置在指定容器位置的元素

返回值

HRESULT [▶ 135]

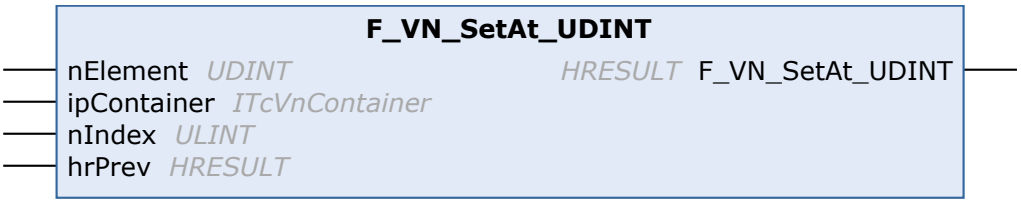
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.35 F_VN_SetAt_UDINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_UDINT : HRESULT
VAR_INPUT
    nElement    : UDINT;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	UDINT	要设置在指定容器位置的元素
ipContainer	ITcVnContainer [▶ 365]	包含 UDINT 元素的容器，其中 nIndex 位置的元素由 nElement 代替

名称	类型	描述
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

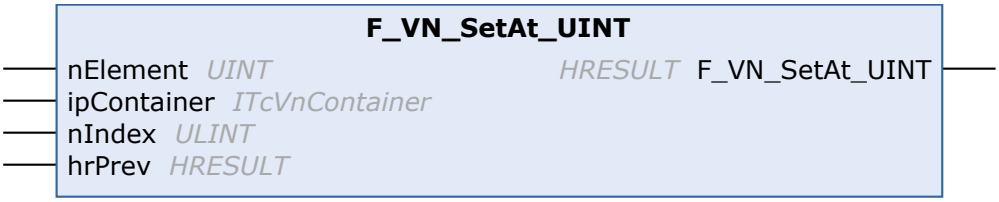
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.36 F_VN_SetAt_UINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_UINT : HRESULT
VAR_INPUT
    nElement    : UINT;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	UINT	要设置在指定容器位置的元素
ipContainer	ITcVnContainer [▶ 365]	包含 UINT 元素的容器，其中 nIndex 位置的元素由 nElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.37 F_VN_SetAt_ULINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_ULINT : HRESULT
VAR_INPUT
    nElement    : ULINT;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	ULINT	要设置在指定容器位置的元素
ipContainer	ITcVnContainer [▶ 365]	包含 ULINT 元素的容器，其中 nIndex 位置的元素由 nElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

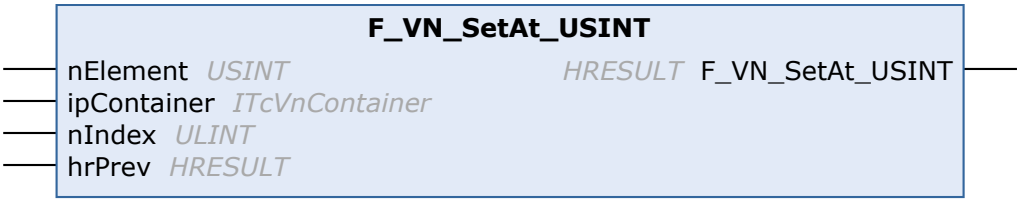
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.6.38 F_VN_SetAt_USINT



设置容器中指定索引处的元素。

语法

定义：

```
FUNCTION F_VN_SetAt_USINT : HRESULT
VAR_INPUT
    nElement    : USINT;
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nElement	USINT	要设置在指定容器位置的元素
ipContainer	ITcVnContainer [▶ 365]	包含 USINT 元素的容器，其中 nIndex 位置的元素被 nElement 代替
nIndex	ULINT	Index（索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.7 F_VN_AdvanceIterator



将迭代器向前推进指定的偏移量。

语法

定义：

```
FUNCTION F_VN_AdvanceIterator : HRESULT
VAR_INPUT
    ipIterator : ITcVnForwardIterator;
    nOffset    : LINT;
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipIterator	ITcVnForwardIterator [▶ 355]	要前进的迭代器
nOffset	LINT	ipIterator 前进的偏移量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

该函数`F_VN_AdvanceIterator`将一个迭代器 [▶ 354]增加或减少一定的偏移量`nOffset`。

如果偏移量`nOffset`始终为`1`，也可以使用更简单的函数F_VN_IncrementIterator [▶ 766]。

参数

迭代器

参数`ipIterator`类型为`ITcVnForwardIterator`。由于`ITcVnBidirectionalIterator`和`ITcVnRandomAccessIterator`从中得出，这些也可以转移到函数中。如果一个前向迭代器要被递减，内部会尝试将其转化为一个双向迭代器并进行递减。

偏移量

偏移量`nOffset`定义迭代器`ipIterator`需要递增或递减多少。如果`nOffset > 0`，递增；如果`nOffset < 0`，递减。

相关函数

- F_VN_IncrementIterator [▶ 766]
- F_VN_AdvanceIterator [▶ 744]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.8 F_VN_CheckIfEmpty



检查容器是否为空。（或者使用接口方法 .CheckIfEmpty。）

语法

定义：

```
FUNCTION F_VN_CheckIfEmpty : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.9 F_VN_CheckIfIteratorIsAtEnd



检查迭代器是否指向最后一个元素的后一个位置。（或者使用接口方法 .CheckIfEnd。）

语法

定义：

```
FUNCTION F_VN_CheckIfIteratorIsAtEnd : HRESULT
VAR_INPUT
    ipIterator : ITcVnForwardIterator;
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipIterator	ITcVnForwardIterator [▶ 355]	迭代器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.10 F_VN_ConvertContainerType



将一个容器转换为另一种类型（不支持结构体元素类型）。

语法

定义：

```
FUNCTION F_VN_ConvertContainerType : HRESULT
VAR_INPUT
    ipSrcContainer  : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    nDestTypeGuid   : GUID;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	源容器
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回转换后的容器
nDestTypeGuid	GUID	指定转换的目标容器类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_ConvertContainerType将一个容器转换为待指定的容器类型 [▶ 152]。

全局ContainerType [▶ 152]常量可用于通过参数nTypeGuid来确定容器类型。

HRESULT

70E	INCOMPATIBLE	相同的接口指针被用作输入容器和结果容器，或者一个指定了与nDestTypeGuid中不同类型的容器被用作接口指针。

应用

例如，一个典型的用例是对一个容器进行转换，以便可以在图像中进行绘制。一个类型为ContainerType_Vector_TcVnPoint2_REAL的轮廓，除其他外由函数F_VN_LocateEllipseExp [▶ 1486]返回，可以转换为类型ContainerType_Vector_TcVnPoint2_DINT，以便能够使用函数F_VN_DrawContours [▶ 1012]在图像中进行绘制：

```
hr := F_VN_LocateEllipseExp(ipImage, stEllipse, […], ipContourPoints, hr);
hr := F_VN_ConvertContainerType(
  ipSrcContainer := ipContourPoints,
  ipDestContainer := ipContourToDraw,
  nDestTypeGuid  := ContainerType_Vector_TcVnPoint2_DINT,
  hr
);
hr := F_VN_DrawContours(ipContourToDraw, -1, ipImageDisp, aBlue, 5, hr);
```

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.11 F_VN_CopyContainer



复制一个容器。

语法

定义：

```
FUNCTION F_VN_CopyContainer : HRESULT
VAR_INPUT
    ipSrcContainer  : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	源容器
ipDestContainer	引用 ITcVnContainer [▶ 365]	目标容器（与 ipSrcContainer 的类型 ID 相同；重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_CopyContainer不仅复制了容器 [▶ 146]的接口指针 [▶ 132]，还对该容器进行了深度复制。

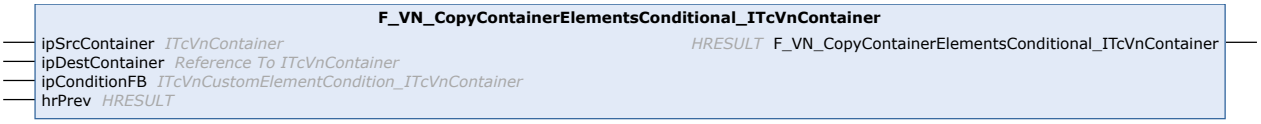
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.12 F_VN_CopyContainerElementsConditional_ITcVnContainer



根据自定义条件将容器元素复制到新容器中。

语法

定义：

```
FUNCTION F_VN_CopyContainerElementsConditional_ITcVnContainer : HRESULT
VAR_INPUT
    ipSrcContainer  : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    ipConditionFB   : ITcVnCustomElementCondition_ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	源容器
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回一个容器，其中包含符合条件的元素
ipConditionFB	ITcVnCustomElementCondition_ITcVnContainer [▶ 240]	自定义条件
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_CopyContainerElementsConditional过滤多维容器 [▶ 146]的元素。过滤条件将由功能块通过接口ITcVnCustomElementCondition_ITcVnContainer [▶ 240]指定。

参数

输入容器

输入容器ipSrcContainer必须是一个多维的容器；即该容器必须包含作为元素的子容器。因此，所有类型ContainerType_Vector_Vector_<...>的容器都可以使用。

结果容器

结果容器ipDestContainer包含所有满足ipConditionFB中条件的子容器。

过滤条件

过滤条件必须被定义为一个功能块的方法Condition。为了使这个功能块能够作为参数ipConditionFB转移到函数中，它必须实现接口ITcVnCustomElementCondition_ITcVnContainer [▶ 240]。

样本

- 容器元素选择 [▶ 2723]

相关函数

- F_VN_CopyContainerElementsConditional_ITcVnContainer [▶ 749]
- F_VN_CopyContainerElementsConditional_ITcVnForwardIterator [▶ 750]

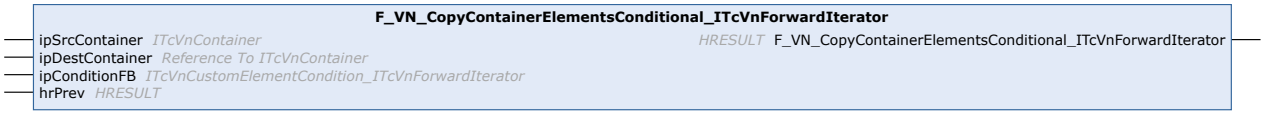
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.13 F_VN_CopyContainerElementsConditional_ITcVnForwardIterator



根据自定义条件将容器元素复制到新容器中。

语法

定义：

```
FUNCTION F_VN_CopyContainerElementsConditional_ITcVnForwardIterator : HRESULT
VAR_INPUT
    ipSrcContainer  : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    ipConditionFB   : ITcVnCustomElementCondition_ITcVnForwardIterator;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	源容器
ipDestContainer	引用 ITcVnContainer [▶ 365]	返回一个容器，其中包含符合条件的元素

名称	类型	描述
ipConditionFB	ITcVnCustomElementCondition_ITcVnForwardIterator [▶ 241]	自定义条件
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_CopyContainerElementsConditional_ITcVnForwardIterator过滤多维容器 [▶ 146]的元素。过滤条件将由功能块通过接口ITcVnCustomElementCondition_ITcVnForwardIterator [▶ 241]指定。

与F_VN_CopyContainerElementsConditional_ITcVnContainer [▶ 749]不同，子容器未转移到条件方法Condition，而是迭代器（它们各自指向子容器的第一个元素）。

参数

输入容器

输入容器ipSrcContainer必须是一个多维的容器；即该容器必须包含作为元素的子容器。因此，所有类型ContainerType_Vector_Vector_<…>的容器都可以使用。

结果容器

结果容器ipDestContainer包含所有满足ipConditionFB中条件的子容器。

过滤条件

过滤条件必须被定义为一个功能块的方法Condition。为了使这个功能块能够作为参数ipConditionFB转移到函数中，它必须实现接口ITcVnCustomElementCondition_ITcVnForwardIterator [▶ 241]。

样本

- 容器元素选择 [▶ 2723]

相关函数

- F_VN_CopyContainerElementsConditional_ITcVnContainer [▶ 749]
- F_VN_CopyContainerElementsConditional_ITcVnForwardIterator [▶ 750]

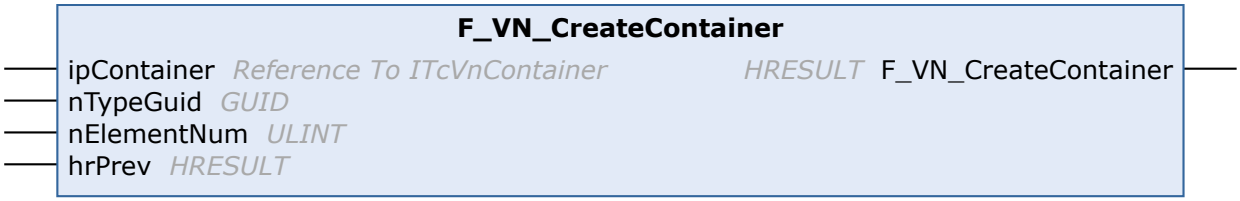
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.14 F_VN_CreateContainer



创建 GUID 类型为 nTypeGuid 的容器，并返回其容器接口。

语法

定义：

```
FUNCTION F_VN_CreateContainer : HRESULT
VAR_INPUT
    ipContainer : Reference To ITcVnContainer;
    nTypeGuid   : GUID;
    nElementNum : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	引用 ITcVnContainer [▶ 365]	返回创建的容器
nTypeGuid	GUID	要创建的容器的 GUID 类型
nElementNum	ULINT	元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

全局ContainerType [▶ 152]常量可用于通过参数`nTypeGuid`来确定容器类型。

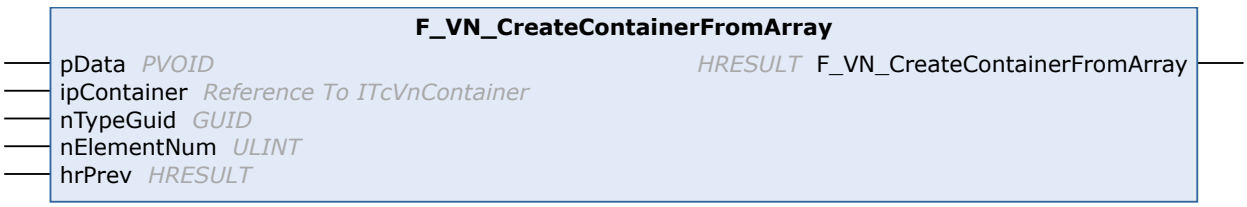
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.15 F_VN_CreateContainerFromArray



创建 GUID 类型为 nTypeGuid 的容器，使用提供的数据对其进行初始化，并返回其容器接口。仅支持用于基本容器类型，即不支持用于容器的容器。

语法

定义：

```
FUNCTION F_VN_CreateContainerFromArray : HRESULT
VAR_INPUT
    pData       : PVOID;
    ipContainer : Reference To ITcVnContainer;
    nTypeGuid   : GUID;
    nElementNum : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
pData	PVOID	指向数据数组的指针。确保数组至少包含 nElementNum 个元素，且数组元素类型与容器元素类型一致。
ipContainer	引用 ITcVnContainer [▶ 365]	返回创建的容器
nTypeGuid	GUID	要创建的容器的 GUID 类型
nElementNum	ULINT	从 pData 中复制的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

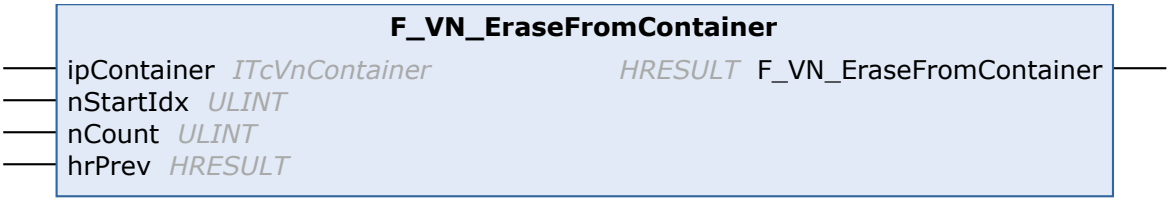
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.16 F_VN_EraseFromContainer



擦除容器中的元素。

语法

定义：

```
FUNCTION F_VN_EraseFromContainer : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nStartIdx   : ULINT;
```

```
    nCount      : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	擦除其中元素的容器
nStartIdx	ULINT	要擦除的第一个元素的索引
nCount	ULINT	要擦除的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.17 F_VN_ExportContainer



将容器元素导出到缓冲区（如数组）中。仅适用于具有基本元素的容器。

语法

定义：

```
FUNCTION F_VN_ExportContainer : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    pBuffer     : PVOID;
    nBufferSize : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
pBuffer	PVOID	用于存储容器元素的缓冲区（确保分配足够的内存！所需大小（以字节为单位）可通过函数 F_VN_ExportContainerSize 来确定）
nBufferSize	ULINT	缓冲区内存大小（以字节为单位）

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

该函数F_VN_ExportContainer将一个容器导出到一个数据缓冲区。请确保在指针的位置分配了足够的内存，以便将容器数据写入其中！所需的大小（以字节为单位）可以通过函数F_VN_ExportContainerSize [▶ 757]确定。你可以分配内存，例如通过创建一个适当大小的数组来分配。

应用

例如，包括10元素的Vector_REAL类型容器的出口如下：

```
VAR
    aArray: ARRAY [0..9] OF REAL;
    ipContainer: ITcVnContainer;
    nBufferSize: ULINT;
END_VAR

hr := F_VN_ExportContainerSize(ipContainer, nBufferSize, hr);
IF nBufferSize = SIZEOF(aArray) THEN
  hr := F_VN_ExportContainer(
    ipContainer:=ipContainer,
    pBuffer:=ADR(aArray),
    nBufferSize:=nBufferSize,
    hr
  );
END_IF
```

注意

内存访问

必须确保容器的类型和长度与指针处内存区域的大小一致。否则会导致系统崩溃或数据失效。

相关函数

- F_VN_ExportContainer [▶ 754]
- F_VN_ExportSubContainer [▶ 758]
- F_VN_ExportContainer_String [▶ 756]
- F_VN_ExportSubContainer_String [▶ 759]
- F_VN_ExportContainerSize [▶ 757]
- F_VN_ExportSubContainerSize [▶ 760]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.18 F_VN_ExportContainer_String



将容器元素导出为字符串。仅适用于 ContainerType_String_SINT 类型的容器。

语法

定义：

```
FUNCTION F_VN_ExportContainer_String : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    sText       : STRING;
    nMaxLength  : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	ContainerType_String_SINT 类型的容器
sText	STRING	确保选择足够的字符串大小！所需大小可使用函数 VnExportContainerSize 确定。
nMaxLength	ULINT	要导出的字符串的最大长度（包括 0 结尾）。如果容器内容较长，则在 nMaxLength - 1 处截断字符串，并添加 0 结尾。在这种情况下，将返回 S_FALSE。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

相关函数

- F_VN_ExportContainer [▶ 754]
- F_VN_ExportSubContainer [▶ 758]
- F_VN_ExportContainer_String [▶ 756]
- F_VN_ExportSubContainer_String [▶ 759]
- F_VN_ExportContainerSize [▶ 757]
- F_VN_ExportSubContainerSize [▶ 760]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.19 F_VN_ExportContainerSize



确定存储所有容器元素所需的缓冲区大小（以字节为单位）（number_of_Elements * size_per_Element）。仅适用于具有基本元素的容器。

语法

定义：

```
FUNCTION F_VN_ExportContainerSize : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nBufferSize : ULINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nBufferSize	ULINT	输出参数，包含所需的缓冲区大小

返回值

HRESULT [▶ 135]

相关函数

- F_VN_ExportContainer [▶ 754]
- F_VN_ExportSubContainer [▶ 758]
- F_VN_ExportContainer_String [▶ 756]
- F_VN_ExportSubContainer_String [▶ 759]
- F_VN_ExportContainerSize [▶ 757]
- F_VN_ExportSubContainerSize [▶ 760]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.20 F_VN_ExportSubContainer



将子容器的容器元素导出到缓冲区（如数组）中。仅适用于具有基本元素的二维容器。

语法

定义：

```
FUNCTION F_VN_ExportSubContainer : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    pBuffer     : PVOID;
    nBufferSize : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nIndex	ULINT	请求元素的索引
pBuffer	PVOID	用于存储容器元素的缓冲区（确保分配足够的内存！所需大小（以字节为单位）可通过函数 F_VN_ExportSubContainerSize 来确定）
nBufferSize	ULINT	缓冲区内存大小（以字节为单位）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

相关函数

- F_VN_ExportContainer [▶ 754]
- F_VN_ExportSubContainer [▶ 758]
- F_VN_ExportContainer_String [▶ 756]
- F_VN_ExportSubContainer_String [▶ 759]
- F_VN_ExportContainerSize [▶ 757]
- F_VN_ExportSubContainerSize [▶ 760]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.21 F_VN_ExportSubContainer_String



将子容器的容器元素导出为字符串。仅适用于 ContainerType_Vector_String_SINT 类型的二维容器。

语法

定义：

```
FUNCTION F_VN_ExportSubContainer_String : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
    sText       : STRING;
    nMaxLength  : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	ContainerType_Vector_String_SINT 类型的容器
nIndex	ULINT	请求元素的索引
sText	STRING	确保选择足够的字符串大小！所需大小可使用函数 VnExportSubContainerSize 确定。
nMaxLength	ULINT	要导出的字符串的最大长度（包括 0 结尾）。如果容器内容较长，则在 nMaxLength - 1 处截断字符串，并添加 0 结尾。在这种情况下，将返回 S_FALSE。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

应用

例如，在代码读取的情况下，使用函数F_VN_ExportSubContainer，以获得结果文本：

```
hr := F_VN_ReadPharmaCode(ipImage, ipDecodedText, hr);
IF hr = S_OK THEN
  hr := F_VN_ExportSubContainer_String(ipDecodedText, 0, sCodeAsString, 255, hr);
    // sCodeAsString contains the result
END_IF
```

相关函数

- F_VN_ExportContainer [▶ 754]
- F_VN_ExportSubContainer [▶ 758]
- F_VN_ExportContainer_String [▶ 756]
- F_VN_ExportSubContainer_String [▶ 759]
- F_VN_ExportContainerSize [▶ 757]
- F_VN_ExportSubContainerSize [▶ 760]

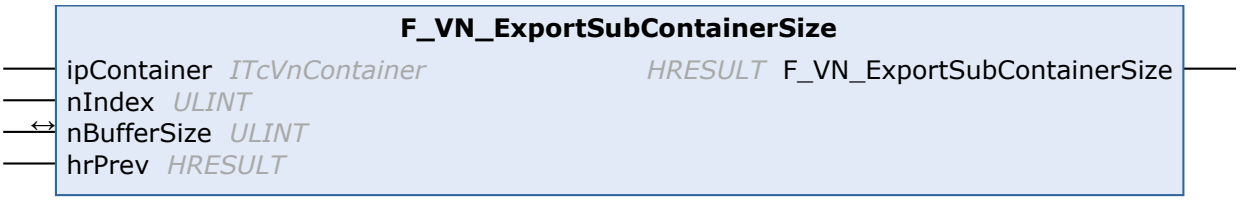
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.22 F_VN_ExportSubContainerSize



确定存储子容器中所有容器元素所需的缓冲区大小（以字节为单位）（number_of_Elements * size_per_Element）。仅适用于具有基本元素的二维容器。

语法

定义：

```
FUNCTION F_VN_ExportSubContainerSize : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nIndex      : ULINT;
END_VAR
VAR_IN_OUT
    nBufferSize : ULINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	包含基本元素的容器
nIndex	ULINT	请求元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nBufferSize	ULINT	输出参数，包含所需的缓冲区大小

返回值

HRESULT [▶ 135]

相关函数

- F_VN_ExportContainer [▶ 754]
- F_VN_ExportSubContainer [▶ 758]
- F_VN_ExportContainer_String [▶ 756]

- F_VN_ExportSubContainer_String [▶ 759]
- F_VN_ExportContainerSize [▶ 757]
- F_VN_ExportSubContainerSize [▶ 760]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.23 F_VN_ExtractContainerRange



将源容器的指定范围复制到目标容器中。（如果目标容器中已经包含元素，则这些元素将被移除）。

语法

定义：

```
FUNCTION F_VN_ExtractContainerRange : HRESULT
VAR_INPUT
    ipSrcContainer  : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    nFirstIdx       : ULINT;
    nLastIdx        : ULINT;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	源容器
ipDestContainer	引用 ITcVnContainer [▶ 365]	目标容器
nFirstIdx	ULINT	要复制的第一个元素的索引
nLastIdx	ULINT	要复制的最后一个元素的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

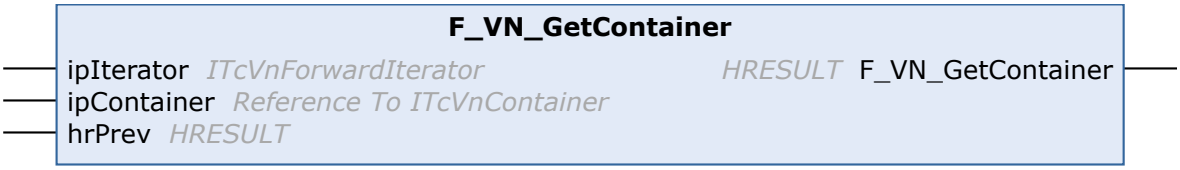
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.24 F_VN_GetContainer



获取转换为 ITcVnContainer 接口的当前元素的指针，并递增其引用计数器（仅适用于容器类型）。（或者使用接口方法 .GetContainer。）

语法

定义：

```
FUNCTION F_VN_GetContainer : HRESULT
VAR_INPUT
    ipIterator  : ITcVnForwardIterator;
    ipContainer : Reference To ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipIterator	ITcVnForwardIterator [▶ 355]	迭代器
ipContainer	引用 ITcVnContainer [▶ 365]	返回容器接口
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

深度复制

这个函数创建一个容器元素的深度拷贝。这意味着，如果想要改变容器中的元素，必须把更改单独写回容器。

如果想要在访问容器元素的同时递增迭代器，专家版F_VN_GetContainerExp [▶ 763]可用于此。

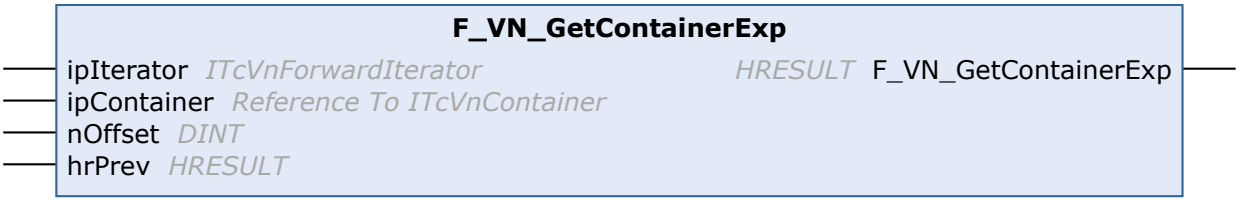
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.25 F_VN_GetContainerExp



获取转换为 ITcVnContainer 接口的当前元素的指针，并递增其引用计数器（仅适用于容器类型）。此外，还可以提供一个偏移量，用于之后递增或递减迭代器。（专家函数）

语法

定义：

```
FUNCTION F_VN_GetContainerExp : HRESULT
VAR_INPUT
    ipIterator  : ITcVnForwardIterator;
    ipContainer : Reference To ITcVnContainer;
    nOffset     : DINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipIterator	ITcVnForwardIterator [▶ 355]	迭代器
ipContainer	引用 ITcVnContainer [▶ 365]	返回容器接口
nOffset	DINT	之后递增（>0）或递减（<0）迭代器的偏移量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_GetContainerExp是F_VN_GetContainer [▶ 762]的专家版本。如果在通过迭代器访问容器元素的同时还也希望递增这个迭代器，那么函数可以提供帮助。将参数nOffset设为值1 进行正常、连续递增：

```
hr := F_VN_GetContainerExp(ipIterator, ipElement, 1, hr);
```

通过标准函数，你需要两个函数调用：

```
hr := F_VN_GetContainer(ipIterator, ipElement, hr);
hr := F_VN_IncrementIterator(ipIterator, hr);
```

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.26 F_VN_GetForwardIterator



获取容器的前向迭代器。（或者使用接口方法 .GetForwardIterator。）

语法

定义：

```
FUNCTION F_VN_GetForwardIterator : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    ipIterator  : Reference To ITcVnForwardIterator;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	容器
ipIterator	引用 ITcVnForwardIterator [▶ 355]	返回迭代器接口
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

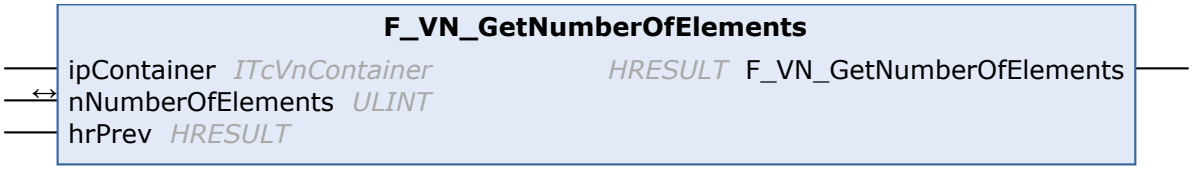
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.27 F_VN_GetNumberOfElements



获取容器中元素的数量。（或者使用接口方法 .getElementNum。）

语法

定义：

```
FUNCTION F_VN_GetNumberOfElements : HRESULT
VAR_INPUT
    ipContainer       : ITcVnContainer;
END_VAR
```

```
VAR_IN_OUT
    nNumberOfElements : ULINT;
END_VAR
VAR_INPUT
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nNumberOfElements	ULINT	返回 ipContainer 中元素的数量

返回值

HRESULT [▶ 135]

更多信息

这个函数不应用来迭代容器元素。相反，使用迭代器访问元素 [▶ 146]。特别是，不建议使用以下代码结构：

```
hr := F_VN_GetNumberOfElements(ipContainer, ipNumber, hr);
FOR i:=0 TO (ipNumber-1) DO
    // access container elements
END_FOR
```

由于`ipNumber`为`UDINT`（自然整数），所以返回值`ipNumber = 0`将导致下溢。

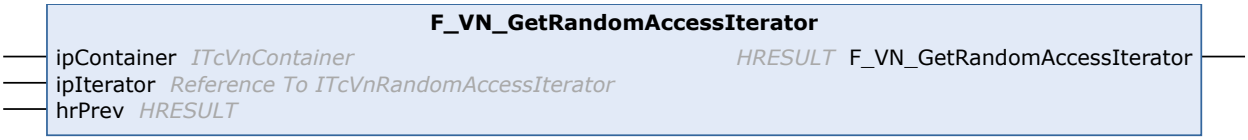
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.28 F_VN_GetRandomAccessIterator



获取容器的随机访问迭代器。（或者使用接口方法 .GetRandomAccessIterator。）

语法

定义：

```
FUNCTION F_VN_GetRandomAccessIterator : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    ipIterator  : Reference To ITcVnRandomAccessIterator;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	容器
ipIterator	引用 ITcVnRandomAccessIterator [▶ 361]	返回迭代器接口
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

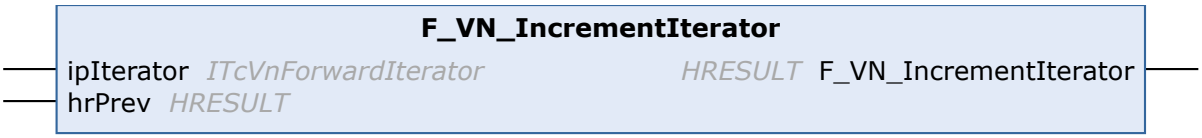
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.29 F_VN_IncrementIterator



递增迭代器。（或者使用接口方法 .Increment。）

语法

定义：

```
FUNCTION F_VN_IncrementIterator : HRESULT
VAR_INPUT
    ipIterator : ITcVnForwardIterator;
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipIterator	ITcVnForwardIterator [▶ 355]	迭代器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_IncrementIterator将迭代器 [▶ 354]递增一个元素。相反，如果希望递减迭代器或递增几个位置，请使用函数F_VN_AdvanceIterator [▶ 744]。

参数

迭代器

参数ipIterator类型为ITcVnForwardIterator。由于ITcVnBidirectionalIterator和ITcVnRandomAccessIterator从中得出，这些也可以转移到函数中。如果一个前向迭代器要被递减，内部会尝试将其转化为一个双向迭代器并进行递减。

相关函数

- F_VN_IncrementIterator [▶ 766]
- F_VN_AdvanceIterator [▶ 744]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.30 F_VN_IteratorDistance



计算 2 个迭代器之间的距离。

语法

定义：

```
FUNCTION F_VN_IteratorDistance : HRESULT
VAR_INPUT
    ipIteratorFirst : ITcVnForwardIterator;
    ipIteratorLast  : ITcVnForwardIterator;
END_VAR
VAR_IN_OUT
    nDistance       : LINT;
END_VAR
VAR_INPUT
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipIteratorFirst	ITcVnForwardIterator [▶ 355]	第一个迭代器
ipIteratorLast	ITcVnForwardIterator [▶ 355]	最后一个迭代器

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nDistance	LINT	返回计算出的距离

返回值

HRESULT [▶ 135]

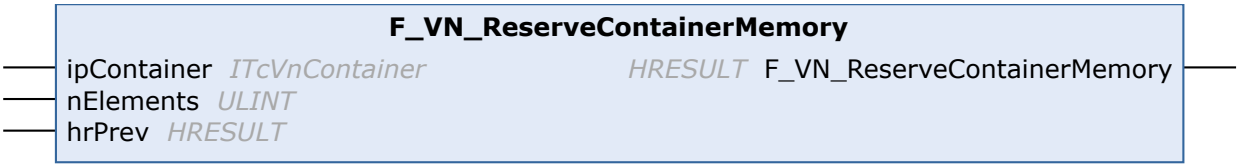
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.31 F_VN_ReserveContainerMemory



预留容器内存（在手动添加元素前应使用最大所需元素数量调用，以提高性能）

语法

定义：

```
FUNCTION F_VN_ReserveContainerMemory : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    nElements   : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	为其预留内存的容器
nElements	ULINT	容器应预留内存的元素数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

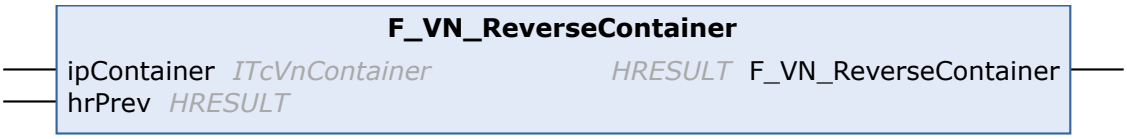
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.32 F_VN_ReverseContainer



反转容器元素。

语法

定义：

```
FUNCTION F_VN_ReverseContainer : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	要反转的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.33 F_VN_SetContainer



使用 ITcVnContainer 接口设置当前元素（仅适用于容器类型）。（或者使用接口方法 .SetContainer。）

语法

定义：

```
FUNCTION F_VN_SetContainer : HRESULT
VAR_INPUT
    ipIterator  : ITcVnForwardIterator;
    ipContainer : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipIterator	ITcVnForwardIterator [▶ 355]	迭代器
ipContainer	ITcVnContainer [▶ 365]	将其内容分配给当前元素的容器接口
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.34 F_VN_SetIteratorToBegin



将迭代器设置为容器的第一个元素。（或者使用接口方法 .SetToBegin。）

语法

定义：

```
FUNCTION F_VN_SetIteratorToBegin : HRESULT
VAR_INPUT
    ipIterator : ITcVnForwardIterator;
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipIterator	ITcVnForwardIterator [▶ 355]	迭代器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

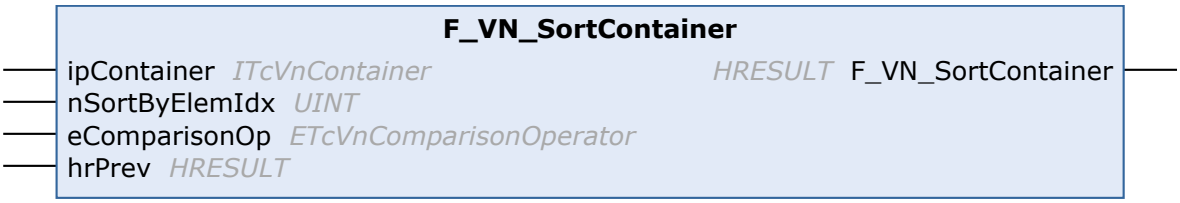
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.35 F_VN_SortContainer



Sort container elements.

Inputs

Name	Type	Description
ipContainer	ITcVnContainer [▶ 365]	Container to sort (only basic arithmetic, TcVnPoint and TcVnVector containers are supported)
nSortByElemIdx	UINT	Index of the element data to sort by (0 for arithmetic types, 0 or 1 for TcVnPoint2 and TcVnVector2, 0 to 2 for TcVnPoint3 and TcVnVector3, 0 to 3 for TcVnVector4)
eComparisonOp	ETcVnComparisonOperator [▶ 178]	Element comparison operator (LESS to sort ascending or GREATER to sort descending)
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

Return value

HRESULT [▶ 135]

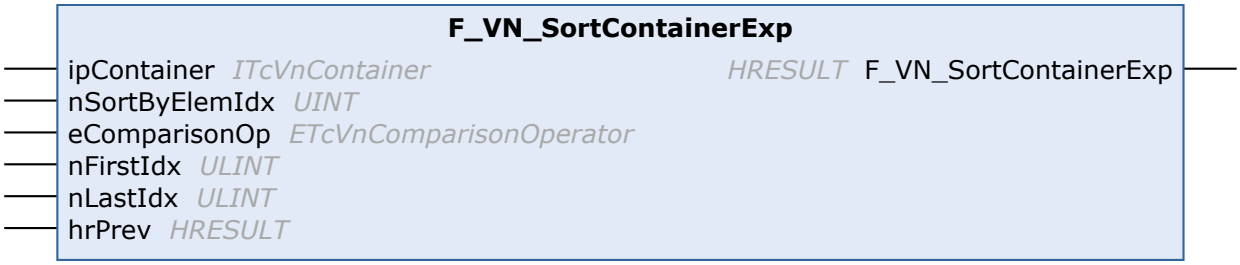
Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.5.36 F_VN_SortContainerExp



Sort container elements. (expert function)

Inputs

Name	Type	Description
ipContainer	ITcVnContainer [▶ 365]	Container to sort (only basic arithmetic, TcVnPoint and TcVnVector containers are supported)
nSortByElemIdx	UINT	Index of the element data to sort by (0 for arithmetic types, 0 or 1 for TcVnPoint2 and TcVnVector2, 0 to 2 for TcVnPoint3 and TcVnVector3, 0 to 3 for TcVnVector4)
eComparisonOp	ETcVnComparisonOperator [▶ 178]	Element comparison operator (LESS to sort ascending or GREATER to sort descending)
nFirstIdx	ULINT	First index of the container to sort (inclusive)
nLastIdx	ULINT	Last index of the container to sort (inclusive). Must be >= nFirstIdx. Automatically cropped to the last available index in the container, so it is valid to specify larger values to sort until the end.
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

Return value

HRESULT [▶ 135]

Required License

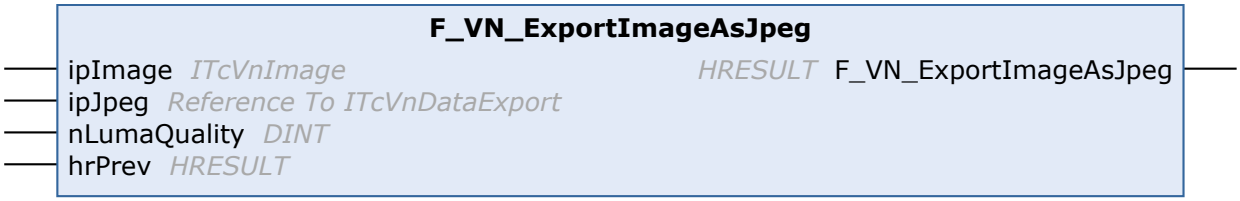
TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6 基本图像操作

6.1.4.6.1 F_VN_ExportImageAsJpeg



Exports the image as jpeg.

Syntax

Definition:

```
FUNCTION F_VN_ExportImageAsJpeg : HRESULT
VAR_INPUT
    ipImage      : ITcVnImage;
    ipJpeg       : Reference To ITcVnDataExport;
    nLumaQuality : DINT;
    hrPrev       : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipImage	ITcVnImage [▶ 407]	Source image (ET_USINT or ET_UINT with 1, 3 or 4 channels). ET_UINT only supports lossless compression. 3 channel expects RGB, 4 channel RGBA (alpha is simply ignored, jpeg does not support it).
ipJpeg	Reference To ITcVnDataExport [▶ 399]	Returns the jpeg image. The resulting data can be exported via RPC.
nLumaQuality	DINT	Luma quality, which is in a range of 0 - 100 for lossy or 101 - 107 for lossless compression (for lossless, this selects 1 - 7 as the predictor). Recommended: 90 for lossy or 101 for lossless. Lossless compression is significantly slower and leads to larger files, so usually a high-quality lossy compression is the better choice.
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

还请参阅有关此

F_VN_ExportImageAsJpegExp [▶ 773]

6.1.4.6.2 F_VN_ExportImageAsJpegExp



Exports the image as jpeg. (expert function)

Syntax

Definition:

```
FUNCTION F_VN_ExportImageAsJpegExp : HRESULT
VAR_INPUT
    ipImage         : ITcVnImage;
    ipJpeg          : Reference To ITcVnDataExport;
    nLumaQuality    : DINT;
    nChromaQuality  : DINT;
    eSamplingFactor : ETcVnJpegSampling;
    hrPrev          : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipImage	ITcVnImage [▶ 407]	Source image (ET_USINT or ET_UINT with 1, 3 or 4 channels). ET_UINT only supports lossless compression. 3 channel expects RGB, 4 channel RGBA (alpha is simply ignored, jpeg does not support it).
ipJpeg	Reference To ITcVnDataExport [▶ 399]	Returns the jpeg image. The resulting data can be exported via RPC.
nLumaQuality	DINT	Luma quality, which is in a range of 0 - 100 for lossy or 101 - 107 for lossless compression (for lossless, this selects 1 - 7 as the predictor). Recommended: 90 for lossy or 101 for lossless. Lossless compression is significantly slower and leads to larger files, so usually a high-quality lossy compression is the better choice.
nChromaQuality	DINT	Chroma quality (0 - 100) or -1 to use the same as luma. Only used for lossy compression. If chroma quality is different from luma quality, chroma subsampling is disabled automatically (samplingFactor JS_444).
eSamplingFactor	ETcVnJpegSampling [▶ 155]	Jpeg chroma sampling factor. Only used for lossy compression.
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Base

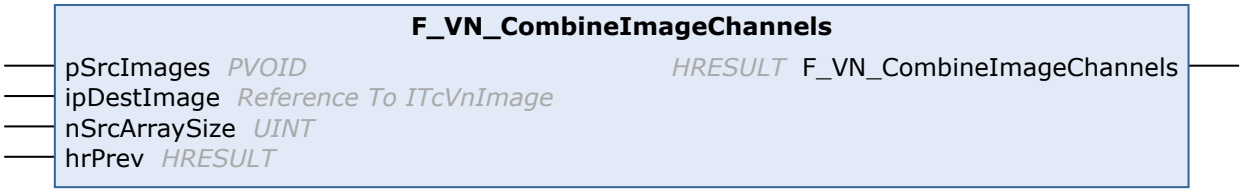
系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

还请参阅有关此

- F_VN_ExportImageAsJpeg [▶ 772]
- ETcVnJpegSampling [▶ 155]

6.1.4.6.3 F_VN_CombineImageChannels



Combines several single-channel images into one multi-channel image.

语法

定义：

```
FUNCTION F_VN_CombineImageChannels : HRESULT
VAR_INPUT
    pSrcImages    : PVOID;
    ipDestImage   : Reference To ITcVnImage;
    nSrcArraySize : UINT;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
pSrcImages	PVOID	指向单通道源图像数组的指针
ipDestImage	引用 ITcVnImage [▶ 407]	多通道目标图像（如果需要，将创建适当的目标图像）。
nSrcArraySize	UINT	pSrcImages 数组大小
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

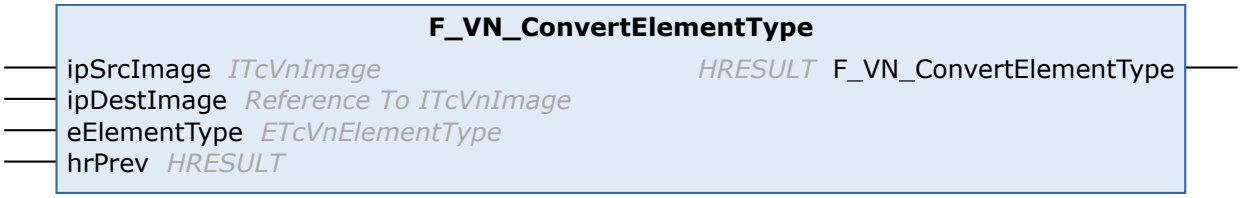
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.4 F_VN_ConvertElementType



Converts an image to another element type.

语法

定义：

```
FUNCTION F_VN_ConvertElementType : HRESULT
VAR_INPUT
    ipSrcImage   : ITcVnImage;
    ipDestImage  : Reference To ITcVnImage;
    eElementType : ETcVnElementType;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eElementType	ETcVnElementType [▶ 191]	所需的目标图像元素类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.5 F_VN_ConvertElementTypeExp



Converts an image to another element type. (expert function)

语法

定义：

```
FUNCTION F_VN_ConvertElementTypeExp : HRESULT
VAR_INPUT
    ipSrcImage   : ITcVnImage;
    ipDestImage  : Reference To ITcVnImage;
    eElementType : ETcVnElementType;
    fScaleFactor : LREAL;
    fDelta       : LREAL;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像

名称	类型	描述
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eElementType	ETcVnElementType [▶ 191]	所需的目标图像元素类型
fScaleFactor	LREAL	像素值的比例系数
fDelta	LREAL	与缩放像素值相加的值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.6 F_VN_CopyImage



Creates a deep copy of an image.

语法

定义：

```
FUNCTION F_VN_CopyImage : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_CopyImage创建输入图像ipSrcImage的深度拷贝。另外，函数F_VN_CopyImageRegion [▶ 778]可以用来创建图像区域的副本。

计算时间优化

关于计算时间的优化，检查每个图像副本的必要性。

相关函数

- F_VN_CopyImage [▶ 777]用于复制整个图像
- F_VN_CopyImageRegion [▶ 778]用于复制一个图像区域
- F_VN_CopyImageRegionToRegion [▶ 780]用于将一个图像区域复制到某个位置

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.7 F_VN_CopyImageRegion



Deep copy the specified region of interest into a new image.

语法

定义：

```
FUNCTION F_VN_CopyImageRegion : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    nX          : UDINT;
    nY          : UDINT;
    nWidth      : UDINT;
    nHeight     : UDINT;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
nX	UDINT	左边界（含基于 0 的索引）
nY	UDINT	上边界（含基于 0 的索引）
nWidth	UDINT	ROI 宽度

名称	类型	描述
nHeight	UDINT	ROI 高度
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_CopyImageRegion创建输入图像ipSrcImage图像区域的深度拷贝。图像区域由矩形指定位置和大小，如同ROI [▶ 150]。

计算时间优化

关于计算时间的优化，检查每个图像副本的必要性。

如果不需要复制，而只需要减少图像区域的大小，则可以设置感兴趣区域 [▶ 150]。

应用

例如，复制尺寸[240, 120]且左上角位于[50, 50]的图像区域如下所示：

```
hr := F_VN_CopyImageRegion(
    ipSrcImage  :=  ipImageIn,
    nX          :=  50,
    nY          :=  50,
    nWidth      :=  240,
    nHeight     :=  120,
    ipDestImage :=  ipImageWork,
    hrPrev      :=  hr,
);
```

相关函数

- F_VN_CopyImage [▶ 777]用于复制整个图像
- F_VN_CopyImageRegion [▶ 778]用于复制一个图像区域
- F_VN_CopyImageRegionToRegion [▶ 780]用于将一个图像区域复制到某个位置

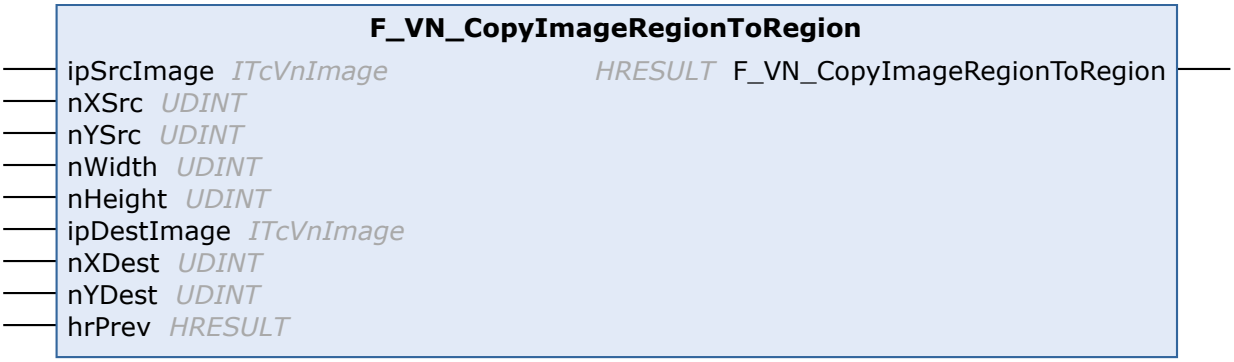
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.8 F_VN_CopyImageRegionToRegion



Copy an image region into another image region.

语法

定义：

```
FUNCTION F_VN_CopyImageRegionToRegion : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    nXSrc       : UDINT;
    nYSrc       : UDINT;
    nWidth      : UDINT;
    nHeight     : UDINT;
    ipDestImage : ITcVnImage;
    nXDest      : UDINT;
    nYDest      : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
nXSrc	UDINT	ipSrcImage 中的左边界（含基于 0 的索引）
nYSrc	UDINT	ipSrcImage 中的上边界（含基于 0 的索引）
nWidth	UDINT	ROI 宽度
nHeight	UDINT	ROI 高度
ipDestImage	ITcVnImage [▶ 407]	目标图像（与 ipSrcImage 类型相同）
nXDest	UDINT	ipDestImage 中的左边界（含基于 0 的索引）
nYDest	UDINT	ipDestImage 中的上边界（含基于 0 的索引）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_CopyImageRegionToRegion将输入图像ipSrcImage的一个图像区域复制到目标图像ipDestImage的某个地方。图像区域由矩形指定位置和大小，如同ROI [▶ 150]。

应用

例如，在结果图像ipImageRes左上角的位置aPosition复制尺寸[100, 100]的图像区域如下所示：

```
hr := F_VN_CopyImageRegionToRegion(
    ipSrcImage  :=  ipImageWork,
    nXSrc       :=  aPosition[0],
    nYSrc       :=  aPosition[1],
```

```
    nWidth      :=  100,
    nHeight     :=  100,
    ipDestImage :=  ipImageRes,
    nXDest      :=  0,
    nYDest      :=  0,
    hrPrev      :=  hr
);
```

样本

- 复制图像区域 [▶ 2726]

相关函数

- F_VN_CopyImage [▶ 777]用于复制整个图像
- F_VN_CopyImageRegion [▶ 778]用于复制一个图像区域
- F_VN_CopyImageRegionToRegion [▶ 780]用于将一个图像区域复制到某个位置

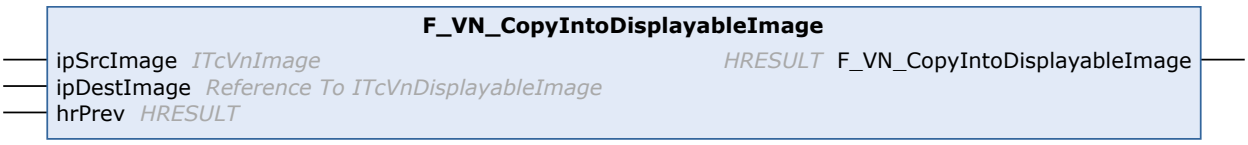
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.9 F_VN_CopyIntoDdisplayableImage



将一个图像复制到可显示图像中。如果不想在调用此函数后使用 ipSrcImage，则可能需要使用 F_VN_TransformIntoDisplayableImage 代替，以获得更好的性能。

语法

定义：

```
FUNCTION F_VN_CopyIntoDisplayableImage : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnDisplayableImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnDisplayableImage [▶ 407]	返回可显示图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

关于更多详情，请参见可显示的图像 [▶ 145]。

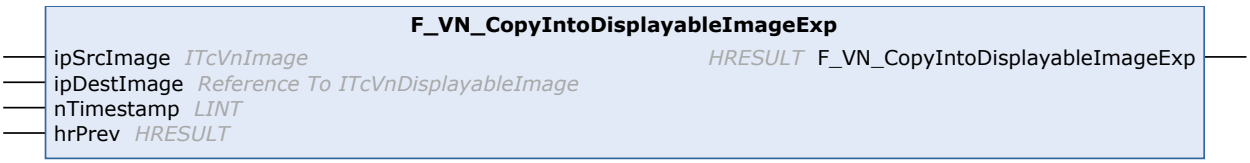
相关函数

- F_VN_TransformIntoDisplayableImage [▶ 818]
- F_VN_TransformIntoDisplayableImageExp [▶ 819]

所需的许可证

TC3 Vision Base

6.1.4.6.10 F_VN_CopyIntoDisplayableImageExp



将图像复制到可显示图像中，并设置指定时间戳，之后可通过 ITcVnImageTimestamps 读取该时间戳。如果不想在调用此函数后使用 ipSrcImage，则可能需要使用 F_VN_TransformIntoDisplayableImage 代替，以获得更好的性能。（专家函数）

语法

定义：

```
FUNCTION F_VN_CopyIntoDisplayableImageExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnDisplayableImage;
    nTimestamp  : LINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnDisplayableImage [▶ 407]	返回可显示图像
nTimestamp	LINT	添加到图像信息中的指定时间戳
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

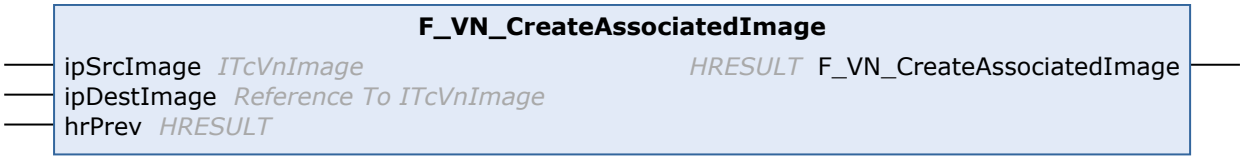
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.6.11 F_VN_CreateAssociatedImage



Creates a new image that shares its data with the source image. E.g. useful to work on different (disjoint) ROIs in parallel.

语法

定义：

```
FUNCTION F_VN_CreateAssociatedImage : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	返回创建的图像（重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.12 F_VN_CreateEmptyImage



Creates an empty image without allocating any data buffer. The initial reference count is set to one if a new image interface is created and kept, otherwise.

语法

定义：

```
FUNCTION F_VN_CreateEmptyImage : HRESULT
VAR_INPUT
    ipImage : Reference To ITcVnImage;
    hrPrev  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	引用 ITcVnImage [▶ 407]	返回创建的图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

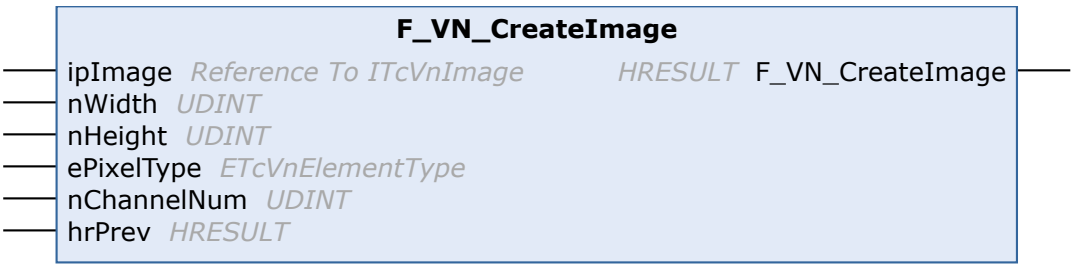
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.13 F_VN_CreateImage



Creates an image and allocate an appropriate data buffer. The initial reference count is set to one if a new image interface is created and kept, otherwise.

语法

定义：

```
FUNCTION F_VN_CreateImage : HRESULT
VAR_INPUT
    ipImage     : Reference To ITcVnImage;
    nWidth      : UDINT;
    nHeight     : UDINT;
    ePixelType  : ETcVnElementType;
    nChannelNum : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	引用 ITcVnImage [▶ 407]	返回创建的图像（重复使用非零接口指针。）
nWidth	UDINT	图像宽度
nHeight	UDINT	图像高度
ePixelType	ETcVnElementType [▶ 191]	像素类型
nChannelNum	UDINT	通道数量（1 至 255）

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.14 F_VN_CreateImageAndSetPixels



Creates an image, allocates an appropriate data buffer and sets all pixels to the specified value. The initial reference count is set to one if a new image interface is created and kept, otherwise.

语法

定义：

```
FUNCTION F_VN_CreateImageAndSetPixels : HRESULT
VAR_INPUT
    ipImage     : Reference To ITcVnImage;
    nWidth      : UDINT;
    nHeight     : UDINT;
    ePixelType  : ETcVnElementType;
    nChannelNum : UDINT;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	引用 ITcVnImage [▶ 407]	返回创建的图像（重复使用非零接口指针。）
nWidth	UDINT	图像宽度
nHeight	UDINT	图像高度
ePixelType	ETcVnElementType [▶ 191]	像素类型
nChannelNum	UDINT	通道数量（1 至 4）

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_LREAL [▶ 153]	像素值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.15 F_VN_CreateImageFromArray



Creates an image and initialize it with the provided data. The initial reference count is set to one if a new image interface is created and kept, otherwise.

语法

定义：

```
FUNCTION F_VN_CreateImageFromArray : HRESULT
VAR_INPUT
    pData       : PVOID;
    ipImage     : Reference To ITcVnImage;
    nWidth      : UDINT;
    nHeight     : UDINT;
    ePixelType  : ETcVnElementType;
    nChannelNum : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
pData	PVOID	指向一维数据数组的指针。确保数组至少包含 nWidth * nHeight * nChannelNum 元素，且数组元素类型与 ePixelType 匹配。
ipImage	引用 ITcVnImage [▶ 407]	返回创建的图像（重复使用非零接口指针。）

名称	类型	描述
nWidth	UDINT	图像宽度
nHeight	UDINT	图像高度
ePixelType	ETcVnElementType [▶ 191]	像素类型
nChannelNum	UDINT	通道数量（1 至 255）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.16 F_VN_CreateNDimensionalImage



Creates an N-dimensional image and allocates an appropriate data buffer. The initial reference count is set to one if a new image interface is created and kept, otherwise.

语法

定义：

```
FUNCTION F_VN_CreateNDimensionalImage : HRESULT
VAR_INPUT
    ipImage     : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aShape      : TcVnArray33_UDINT;
END_VAR
VAR_INPUT
    ePixelType : ETcVnElementType;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	引用 ITcVnImage [▶ 407]	返回创建的图像（重复使用非零接口指针。）
ePixelType	ETcVnElementType [▶ 191]	像素类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aShape	TcVnArray33_UDINT [▶ 153]	指定 ipDestImage 的形状。第一个条目是 ipImage 的维数（最多 32），然后是每个维度的大小。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.17 F_VN_CreateNDimensionalImageAndSetPixels



Creates an N-dimensional image, allocates an appropriate data buffer and sets all pixels to the specified value. The initial reference count is set to one if a new image interface is created and kept, otherwise.

语法

定义：

```
FUNCTION F_VN_CreateNDimensionalImageAndSetPixels : HRESULT
VAR_INPUT
    ipImage     : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aShape      : TcVnArray33_UDINT;
END_VAR
VAR_INPUT
    ePixelType : ETcVnElementType;
    fValue      : LREAL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	引用 ITcVnImage [▶ 407]	返回创建的图像（重复使用非零接口指针。）
ePixelType	ETcVnElementType [▶ 191]	像素类型
fValue	LREAL	像素值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aShape	TcVnArray33_UDINT [▶ 153]	指定 ipDestImage 的形状。第一个条目是 ipImage 的维数（最多 32），然后是每个维度的大小。

返回值

HRESULT [▶ 135]

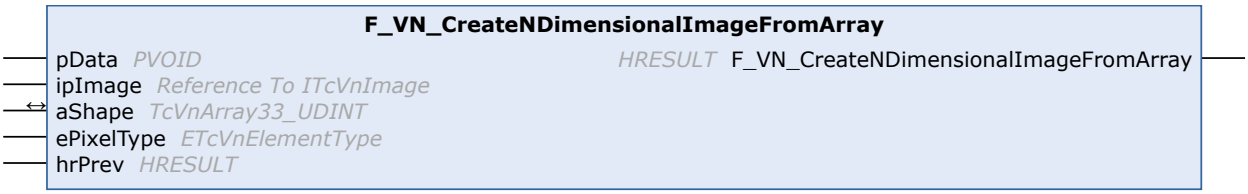
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.18 F_VN_CreateNDimensionalImageFromArray



Creates an N-dimensional image and initialize it with the provided data. The initial reference count is set to one if a new image interface is created and kept, otherwise.

语法

定义：

```
FUNCTION F_VN_CreateNDimensionalImageFromArray : HRESULT
VAR_INPUT
    pData       : PVOID;
    ipImage     : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aShape      : TcVnArray33_UDINT;
END_VAR
VAR_INPUT
    ePixelType : ETcVnElementType;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
pData	PVOID	指向一维数据数组的指针。确保数组包含的元素数量至少等于 aShape 中所有维度的乘积，且数组元素类型与 ePixelType 匹配。
ipImage	引用 ITcVnImage [▶ 407]	返回创建的图像（重复使用非零接口指针。）
ePixelType	ETcVnElementType [▶ 191]	像素类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aShape	TcVnArray33_UDINT [▶ 153]	指定 ipDestImage 的形状。第一个条目是 ipImage 的维数（最多 32），然后是每个维度的大小。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.19 F_VN_ExportImage



Exports the image data into a given buffer (e.g. an array). F_VN_ExportImageSize should be called before to get the required buffer size.

语法

定义：

```
FUNCTION F_VN_ExportImage : HRESULT
VAR_INPUT
    ipImage     : ITcVnImage;
    pBuffer     : PVOID;
    nBufferSize : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	Image
pBuffer	PVOID	指向图像数据存储缓冲区的指针（确保分配足够的内存！所需大小（以字节为单位）可通过函数 F_VN_ExportImageSize 来确定）
nBufferSize	ULINT	缓冲区内存大小（以字节为单位）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.20 F_VN_ExportImageAsBmp



Exports the image as bitmap into a given buffer (e.g. an array). F_VN_ExportImageAsBmpSize should be called before to get the required buffer size.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_ExportImageAsBmp : HRESULT
VAR_INPUT
    ipImage     : ITcVnImage;
    pBuffer     : PVOID;
    nBufferSize : ULINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	Image
pBuffer	PVOID	指向位图图像存储缓冲区的指针（确保分配足够的内存！所需大小（以字节为单位）可通过函数 F_VN_ExportImageAsBmpSize 来确定）
nBufferSize	ULINT	缓冲区内存大小（以字节为单位）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

注意接口ITcVnBitmapExport [▶ 388]的转换特性。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.21 F_VN_ExportImageAsBmpExp



Exports the image as bitmap into a given buffer (e.g. an array). F_VN_ExportImageAsBmpSizeExp should be called before to get the required buffer size. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_ExportImageAsBmpExp : HRESULT
VAR_INPUT
    ipImage     : ITcVnImage;
    pBuffer     : PVOID;
    nBufferSize : ULINT;
    nWidth      : UDINT;
    nHeight     : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	Image
pBuffer	PVOID	指向位图图像存储缓冲区的指针（确保分配足够的内存！所需大小（以字节为单位）可通过函数 F_VN_ExportImageAsBmpSizeExp 来确定）
nBufferSize	ULINT	缓冲区内存大小（以字节为单位）
nWidth	UDINT	所需的宽度（或使用 0 来保持原始宽度）
nHeight	UDINT	所需的高度（或使用 0 来保持原始高度）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.22 F_VN_ExportImageAsBmpSize



Get the required buffer size (in bytes) to store the image as bitmap.

语法

定义：

```
FUNCTION F_VN_ExportImageAsBmpSize : HRESULT
VAR_INPUT
    ipImage     : ITcVnImage;
END_VAR
VAR_IN_OUT
    nBufferSize : ULINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	Image
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nBufferSize	ULINT	返回所需的缓冲区大小（以字节为单位）

返回值

HRESULT [▶ 135]

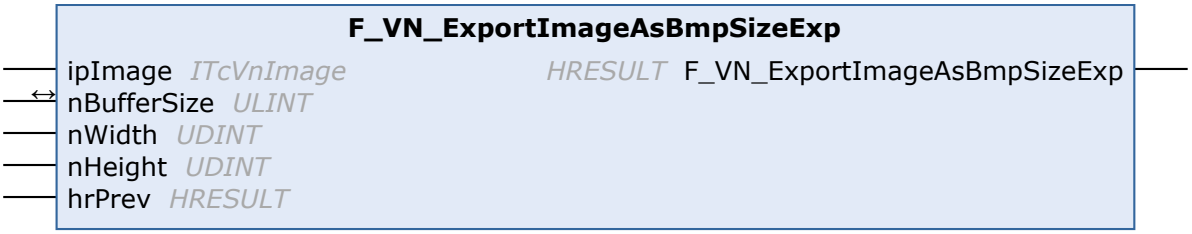
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.23 F_VN_ExportImageAsBmpSizeExp



Get the required buffer size (in bytes) to store the image as bitmap. (expert function)

语法

定义：

```
FUNCTION F_VN_ExportImageAsBmpSizeExp : HRESULT
VAR_INPUT
    ipImage     : ITcVnImage;
END_VAR
VAR_IN_OUT
    nBufferSize : ULINT;
END_VAR
VAR_INPUT
    nWidth      : UDINT;
    nHeight     : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	Image
nWidth	UDINT	所需的宽度（或使用 0 来保持原始宽度）
nHeight	UDINT	所需的高度（或使用 0 来保持原始高度）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nBufferSize	ULINT	返回所需的缓冲区大小（以字节为单位）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.24 F_VN_ExportImageSize



Get the required buffer size (in bytes) to store the image data.

语法

定义：

```
FUNCTION F_VN_ExportImageSize : HRESULT
VAR_INPUT
    ipImage     : ITcVnImage;
END_VAR
VAR_IN_OUT
    nBufferSize : ULINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	Image
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nBufferSize	ULINT	返回所需的缓冲区大小（以字节为单位）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.25 F_VN_FuseImages



Fuse 2 images vertically (intended for line scan cameras).

语法

定义：

```
FUNCTION F_VN_FuseImages : HRESULT
VAR_INPUT
    ipSrcImage1 : ITcVnImage;
    ipSrcImage2 : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    nFirstLine  : UDINT;
    nNumLines   : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	源图像 1
ipSrcImage2	ITcVnImage [▶ 407]	源图像 2
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像
nFirstLine	UDINT	ipSrcImage1 中的行索引，即 ipDestImage 中的第一行
nNumLines	UDINT	应复制到 ipDestImage 的行数，从 ipSrcImage1 中的 nFirstLine 开始。复制完 ipSrcImage1 的最后一行后，其余各行将从行索引 0 开始从 ipSrcImage2 复制。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

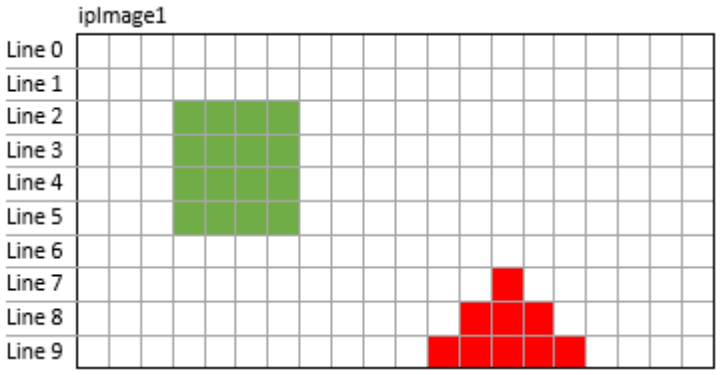
更多信息

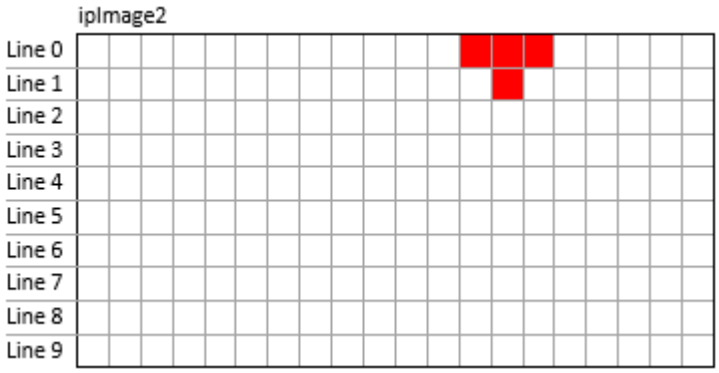
函数F_VN_FuseImages将两个图像合并形成一个新的连续图像。这对线扫描相机来说特别有意义。

函数F_VN_FuseImagesArray [▶ 797]可用于融合最多 10 个图像。

应用

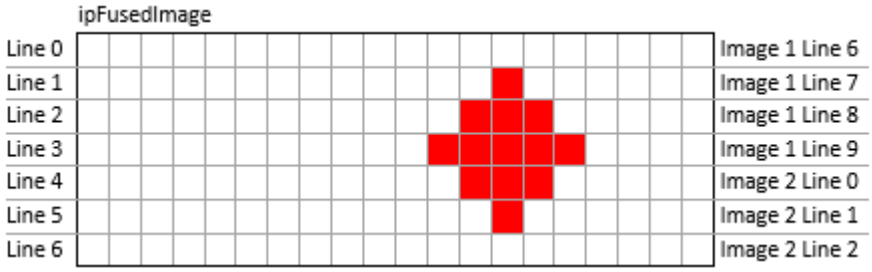
线扫描相机通常将一定数量的连续捕获的线作为二维图像返回。这个二维图像可以包含整个感兴趣的物体（下图中以绿色显示），也可以包含物体的一部分（图中以红色显示）。





如需分析红色物体，可以使用函数F_VN_FuseImages以将两个物体部分融合，从而形成一个完整的图像。如果知道哪些线与对象有关，则可以将链接限制在相关线上。

```
F_VN_FuseImages(ipImage1, ipImage2, ipFusedImage, 6, 7, S_OK);
```



样本

- 融合多个图像 [▶ 2725]

相关函数

- F_VN_FuseImages [▶ 795]用于融合两个图像。
- F_VN_FuseImagesArray [▶ 797]用于融合最多 10 个图像。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.26 F_VN_FuseImagesArray



Fuse up to 10 images vertically (intended for line scan cameras).

语法

定义：

```
FUNCTION F_VN_FuseImagesArray : HRESULT
VAR_IN_OUT
    aSrcImageArray : TcVnArray10_ITcVnImage;
END_VAR
VAR_INPUT
    nValidImages   : UDINT;
    ipDestImage    : Reference To ITcVnImage;
    nFirstLine     : UDINT;
    nNumLines      : UDINT;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
nValidImages	UDINT	aSrcImageArray 中有效图像的数量
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像
nFirstLine	UDINT	ipSrcImage1 中的行索引，即 ipDestImage 中的第一行
nNumLines	UDINT	应复制到 ipDestImage 的行数，从 aSrcImageArray[0] 中的 nFirstLine 开始。复制完 aSrcImageArray[0] 的最后一行后，其余各行将从 aSrcImageArray[1] 复制（从行索引 0 开始），以此类推。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aSrcImageArray	TcVnArray10_ITcVnImage [▶ 153]	包含 10 个元素的源图像数组（并非需要填充所有元素，nValidImages 指定实际图像数量）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_FuseImageArray是函数F_VN_FuseImages [▶ 795]的修改版，且可以融合最多 10 个图像而不是两个图像。

样本

- 融合多个图像 [▶ 2725]

相关函数

- F_VN_FuseImages [▶ 795]用于融合两个图像。
- F_VN_FuseImagesArray [▶ 797]用于融合最多 10 个图像。

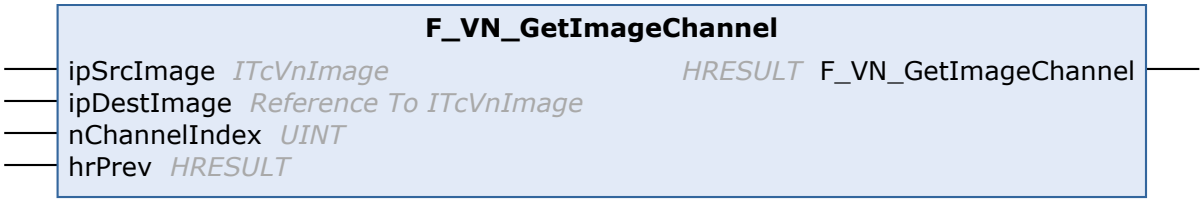
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.27 F_VN_GetImageChannel



Return the specified source image channel as a single-channel image.

语法

定义：

```
FUNCTION F_VN_GetImageChannel : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDestImage   : Reference To ITcVnImage;
    nChannelIndex : UINT;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
nChannelIndex	UINT	请求的源图像通道的索引
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

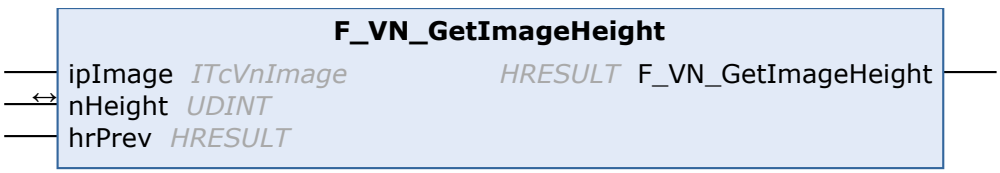
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.28 F_VN_GetImageHeight



Return the height of an image. (Alternatively use interface method .GetHeight.)

语法

定义：

```
FUNCTION F_VN_GetImageHeight : HRESULT
VAR_INPUT
    ipImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    nHeight : UDINT;
END_VAR
VAR_INPUT
    hrPrev  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	Image
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nHeight	UDINT	返回 ipImage 的高度

返回值

HRESULT [▶ 135]

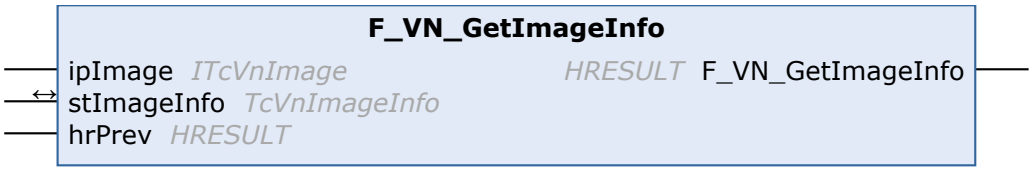
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.29 F_VN_GetImageInfo



Gets a struct containing all common meta infos of the image. (Alternatively use interface method .GetImageInfo.)

语法

定义：

```
FUNCTION F_VN_GetImageInfo : HRESULT
VAR_INPUT
    ipImage     : ITcVnImage;
END_VAR
VAR_IN_OUT
    stImageInfo : TcVnImageInfo;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	Image
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stImageInfo	TcVnImageInfo [▶ 225]	返回描述图像的结构体

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.30 F_VN_GetImageWidth



Return the width of an image. (Alternatively use interface method .GetWidth.)

语法

定义：

```
FUNCTION F_VN_GetImageWidth : HRESULT
VAR_INPUT
    ipImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    nWidth  : UDINT;
END_VAR
VAR_INPUT
    hrPrev  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	Image
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nWidth	UDINT	返回 ipImage 的宽度

返回值

HRESULT [▶ 135]

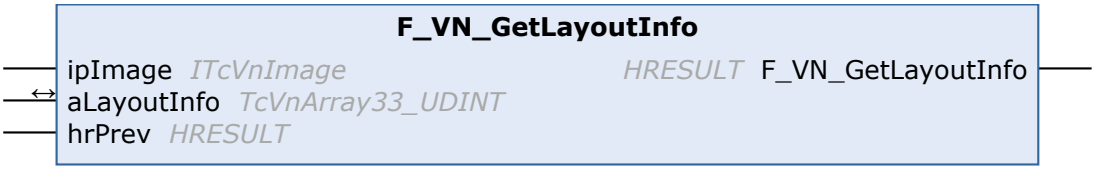
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.31 F_VN_GetLayoutInfo



Return the layout information of an image

语法

定义：

```
FUNCTION F_VN_GetLayoutInfo : HRESULT
VAR_INPUT
    ipImage     : ITcVnImage;
END_VAR
VAR_IN_OUT
    aLayoutInfo : TcVnArray33_UDINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	源图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aLayoutInfo	TcVnArray33_UDINT [▶ 153]	包含 ipImage 的布局信息。aLayoutInfo 的第一个条目是 ipImage 的维数（最多 32），然后是每个维度的大小。

返回值

HRESULT [▶ 135]

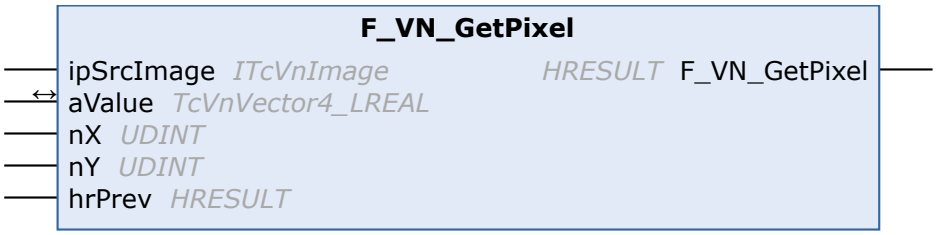
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.32 F_VN_GetPixel



Gets a specific pixel.

语法

定义：

```
FUNCTION F_VN_GetPixel : HRESULT
VAR_INPUT
    ipSrcImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aValue     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nX         : UDINT;
    nY         : UDINT;
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
nX	UDINT	像素的 x 坐标
nY	UDINT	像素的 y 坐标
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_LREAL [▶ 153]	返回像素值（未使用的通道设为 0）。

返回值

HRESULT [▶ 135]

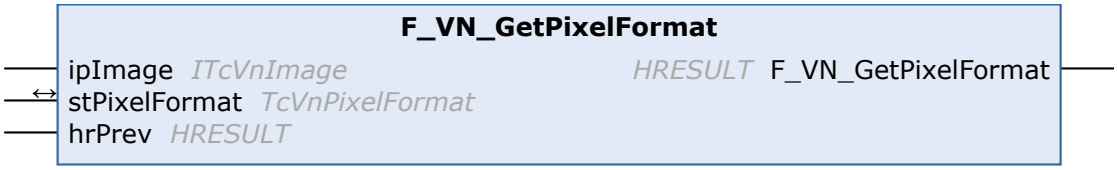
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.33 F_VN_GetPixelFormat



Gets the pixel format of an image. (Alternatively use interface method .GetPixelFormat.)

语法

定义：

```
FUNCTION F_VN_GetPixelFormat : HRESULT
VAR_INPUT
    ipImage       : ITcVnImage;
END_VAR
VAR_IN_OUT
    stPixelFormat : TcVnPixelFormat;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	Image
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stPixelFormat	TcVnPixelFormat [▶ 235]	返回描述像素格式的结构体

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.34 F_VN_GetRoi



Gets the coordinates of the region of interest (ROI) within the image.

语法

定义：

```
FUNCTION F_VN_GetRoi : HRESULT
VAR_INPUT
    ipSrcImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    stRoi      : TcVnRectangle_UDINT;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
StRoi	TcVnRectangle_UDINT [▶ 236]	返回感兴趣区域的坐标

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_GetRoi表明当前在输入图像ipSrcImage上设置了哪个感兴趣区域 [▶ 150]。例如，如果需要根据整个图像指定图像中的点，或者如果需要动态放大 ROI，则需要这些信息。

应用

例如，将一个点引用到原始图像的坐标上如下所示：

```
hr := F_VN_ContourCenterOfMass(ipContour, aCenter, hr);
hr := F_VN_GetRoi(ipImageWork, stRoi, hr);
aCenter[0] := aCenter[0] + stRoi.nX;
aCenter[1] := aCenter[1] + stRoi.nY;
```

相关函数

- F_VN_SetRoi [▶ 814]用于设置感兴趣区域
- F_VN_GetRoi [▶ 805]用于检索设定的感兴趣区域
- F_VN_ResetRoi [▶ 810]用于重设整个图像的感兴趣区域
- F_VN_CopyImageRegion [▶ 779]用于复制一个图像区域

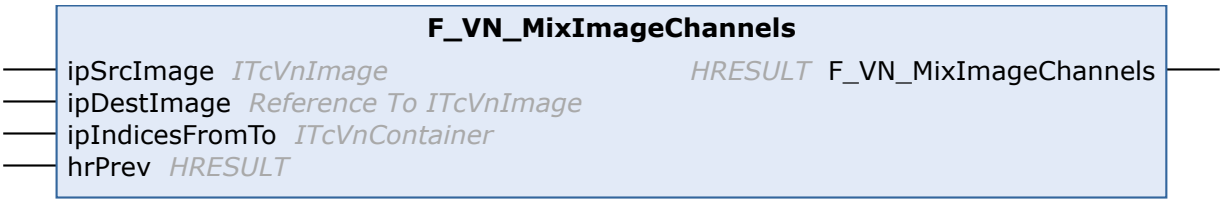
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.35 F_VN_MixImageChannels



Mix image channels by copying the specified channels of the source image into the specified channels of the destination image.

语法

定义：

```
FUNCTION F_VN_MixImageChannels : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipDestImage     : Reference To ITcVnImage;
    ipIndicesFromTo : ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
ipIndicesFromTo	ITcVnContainer [▶ 365]	索引对（ContainerType_Vector_TcVnVector2_DINT），指定应将哪个源通道（TcVnVector2_DINT [0]）复制到哪个目标通道（TcVnVector2_DINT [1]）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.36 F_VN_PadImageBorder



在原始图像边界周围添加填充（零）。

语法

定义：

```
FUNCTION F_VN_PadImageBorder : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipPaddedImage : Reference To ITcVnImage;
    nTopBorder    : UDINT;
    nBottomBorder : UDINT;
    nLeftBorder   : UDINT;
    nRightBorder  : UDINT;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipPaddedImage	引用 ITcVnImage [▶ 407]	填充的目标图像（与 ipSrcImage 类型相同，如果需要，将创建适当的目标图像。）
nTopBorder	UDINT	上边界上方的填充高度（以像素为单位）
nBottomBorder	UDINT	下边界下方的填充高度（以像素为单位）
nLeftBorder	UDINT	左边界前的填充宽度（以像素为单位）
nRightBorder	UDINT	右边界后的填充宽度（以像素为单位）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.37 F_VN_PadImageBorderExp



在原始图像边界周围添加填充。（专家函数）

语法

定义：

```
FUNCTION F_VN_PadImageBorderExp : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipPaddedImage : Reference To ITcVnImage;
    nTopBorder    : UDINT;
    nBottomBorder : UDINT;
    nLeftBorder   : UDINT;
    nRightBorder  : UDINT;
    ePaddingType  : ETcVnBorderInterpolationMethod;
END_VAR
VAR_IN_OUT
    aPaddingValue : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipPaddedImage	引用 ITcVnImage [▶ 407]	填充的目标图像（与 ipSrcImage 类型相同，如果需要，将创建适当的目标图像。）
nTopBorder	UDINT	上边界上方的填充高度（以像素为单位）
nBottomBorder	UDINT	下边界下方的填充高度（以像素为单位）
nLeftBorder	UDINT	左边界前的填充宽度（以像素为单位）
nRightBorder	UDINT	右边界后的填充宽度（以像素为单位）
ePaddingType	ETcVnBorderInterpolationMethod [▶ 160]	指定如何确定填充区域的像素值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aPaddingValue	TcVnVector4_LREAL [▶ 153]	如果使用 CONSTANT，则指定填充值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.38 F_VN_ReinterpretUnsupportedImage



使用附加信息重新解释不受支持的图像数据。如果成功，该图像将变为受支持状态，并带有像素格式标志 bSigned 和 bFloat、像素元素大小 nElementSize 以及颜色通道数 nChannels。该信息根据指定的像素元素类型 ePixelType 确定。

语法

定义：

```
FUNCTION F_VN_ReinterpretUnsupportedImage : HRESULT
VAR_INPUT
    ipImage     : ITcVnImage;
    ePixelType  : ETcVnElementType;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	要重新解释的图像
ePixelType	ETcVnElementType [▶ 191]	用于重新解释图像数据的像素元素类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

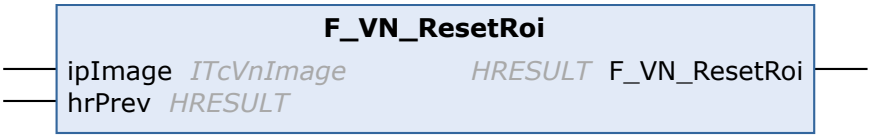
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.39 F_VN_ResetRoi



重置图像的感兴趣区域（ROI）。（执行此操作后，ROI 将设置为整个图像）。

语法

定义：

```
FUNCTION F_VN_ResetRoi : HRESULT
VAR_INPUT
    ipImage : ITcVnImage;
    hrPrev  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	Image
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

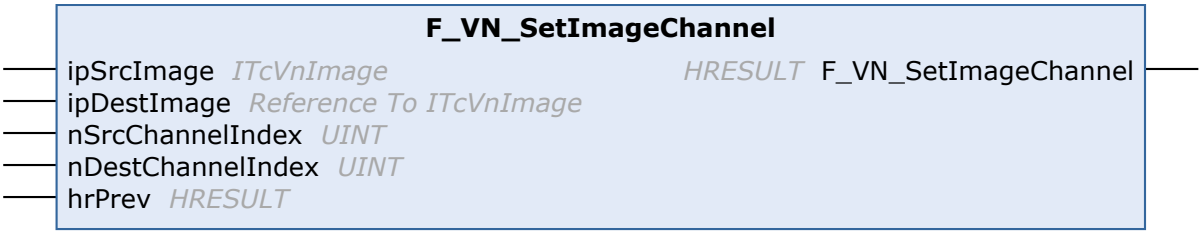
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.40 F_VN_SetImageChannel



Sets the specified destination image channel to the values of the specified source image channel.

语法

定义：

```
FUNCTION F_VN_SetImageChannel : HRESULT
VAR_INPUT
    ipSrcImage        : ITcVnImage;
    ipDestImage       : Reference To ITcVnImage;
    nSrcChannelIndex  : UINT;
    nDestChannelIndex : UINT;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
nSrcChannelIndex	UINT	源图像通道的索引
nDestChannelIndex	UINT	目标图像通道的索引，它将被指定的源图像通道取代
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

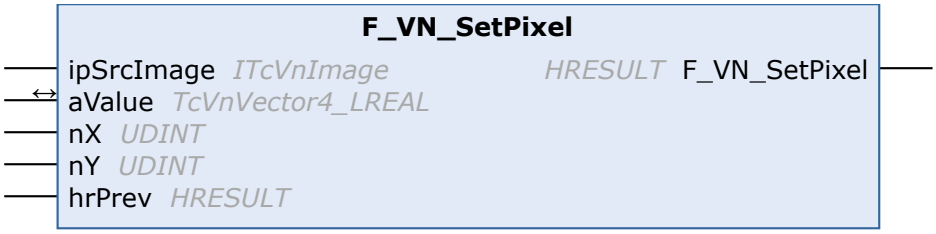
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.41 F_VN_SetPixel



Sets a specific pixel.

语法

定义：

```
FUNCTION F_VN_SetPixel : HRESULT
VAR_INPUT
    ipSrcImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aValue     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nX         : UDINT;
    nY         : UDINT;
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
nX	UDINT	像素的 x 坐标

名称	类型	描述
nY	UDINT	像素的 y 坐标
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_LREAL [▶ 153]	要设置的像素值（忽略其它通道）。

返回值

HRESULT [▶ 135]

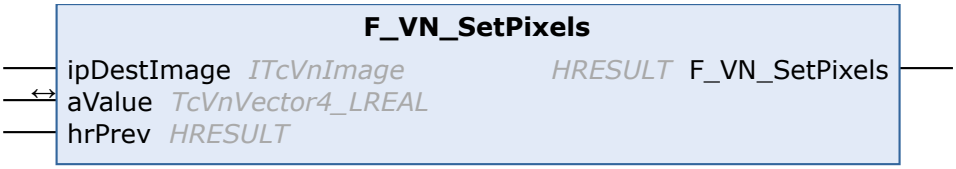
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.42 F_VN_SetPixels



Sets all pixels of an image to a given value.

语法

定义：

```
FUNCTION F_VN_SetPixels : HRESULT
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_LREAL [▶ 153]	像素值

返回值

HRESULT [▶ 135]

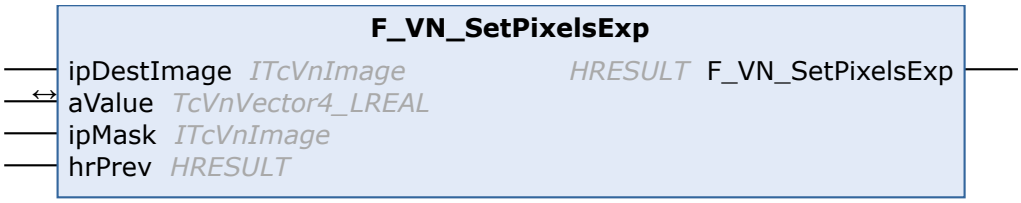
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.43 F_VN_SetPixelsExp



Sets all pixels of an image to a given value. (expert function)

语法

定义：

```
FUNCTION F_VN_SetPixelsExp : HRESULT
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aValue      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipMask      : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
ipMask	ITcVnImage [▶ 407]	掩膜图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aValue	TcVnVector4_LREAL [▶ 153]	像素值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.44 F_VN_SetRoi



Sets a region of interest (ROI) within an image.

语法

定义：

```
FUNCTION F_VN_SetRoi : HRESULT
VAR_INPUT
    nX          : UDINT;
    nY          : UDINT;
    nWidth      : UDINT;
    nHeight     : UDINT;
    ipDestImage : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nX	UDINT	左边界（含基于 0 的索引）
nY	UDINT	上边界（含基于 0 的索引）
nWidth	UDINT	ROI 宽度
nHeight	UDINT	ROI 高度
ipDestImage	ITcVnImage [▶ 407]	目标图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_SetRoi设置图像的感兴趣区域 [▶ 150]。如果不单独传输矩形参数，而是作为一个类型为 TcVnRectangle_UDINT [▶ 236]的结构，可以使用函数F_VN_SetRoi_TcVnRectangle_UDINT [▶ 816]。

应用

例如，设置尺寸[240, 120]且位置[50, 50]的 ROI 如下所示：

```
hr := F_VN_SetRoi(50, 50, 240, 120, ipImageWork, hr);
```

相关函数

- F_VN_SetRoi [▶ 814]用于设置感兴趣区域
- F_VN_GetRoi [▶ 805]用于检索设定的感兴趣区域
- F_VN_ResetRoi [▶ 810]用于重设整个图像的感兴趣区域
- F_VN_CopyImageRegion [▶ 779]用于复制一个图像区域

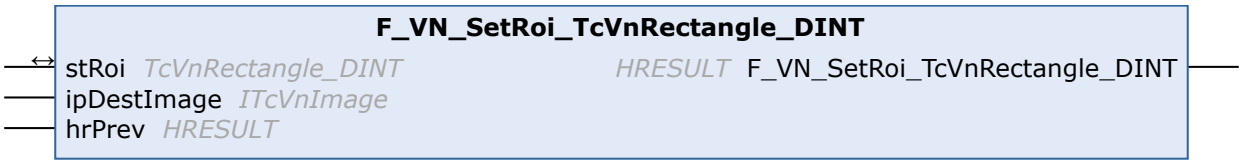
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.45 F_VN_SetRoi_TcVnRectangle_DINT



Sets a region of interest (ROI) within an image.

语法

定义：

```
FUNCTION F_VN_SetRoi_TcVnRectangle_DINT : HRESULT
VAR_IN_OUT
    stRoi       : TcVnRectangle_DINT;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
StRoi	TcVnRectangle_DINT [▶ 236]	感兴趣区域

返回值

HRESULT [▶ 135]

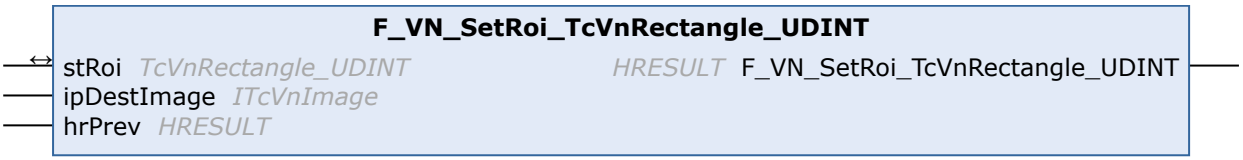
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.46 F_VN_SetRoi_TcVnRectangle_UDINT



在图像中设置感兴趣区域（ROI）。

语法

定义：

```
FUNCTION F_VN_SetRoi_TcVnRectangle_UDINT : HRESULT
VAR_IN_OUT
    stRoi       : TcVnRectangle_UDINT;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
StRoi	TcVnRectangle_UDINT [▶ 236]	感兴趣区域

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.6.47 F_VN_SplitImageChannels



Split a multi-channel image into multiple single channel images (1 for each source image channel).

语法

定义：

```
FUNCTION F_VN_SplitImageChannels : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    pDestImages     : PVOID;
    nArraySize      : UINT;
    hrPrev          : HRESULT;
END_VAR
VAR_OUTPUT
    nImageChannels : UINT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	多通道源图像
pDestImages	PVOID	指向 ITcVnImage 数组的指针（如果需要，将创建适当的目标图像）
nArraySize	UINT	pDestImages 数组元素的数量（数组必须 >= ipSourceImage 通道）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输出

名称	类型	描述
nImageChannels	UINT	实际源图像通道

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.48 F_VN_TransformIntoDisplayableImage



Transform an image into a displayable image. The source image will be released and zeroed while existing destination images will be released and overwritten with the function result. The source image must not be used anywhere else and the transformation is very fast. If you want to use ipSrcImage after this function call, use F_VN_CopyIntoDisplayableImage instead.

语法

定义：

```
FUNCTION F_VN_TransformIntoDisplayableImage : HRESULT
VAR_INPUT
    ipSrcImage  : Reference To ITcVnImage;
    ipDestImage : Reference To ITcVnDisplayableImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	引用 ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnDisplayableImage [▶ 407]	返回可显示图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

关于更多详情，请参见可显示的图像 [▶ 145]。

注意

与 Ads Communicator 对象一起使用

请注意以下情况：待转换图像来自于 Ads Communicator 对象处于活动状态的相机对象。这时，可能会出现这样的情况：图像在物体中没有被内部释放，因此不能被转化为可显示的图像。在这种情况下，`bAllowDeepCopy := TRUE`时，使用F_VN_CopyIntoDisplayableImage [▶ 781]或F_VN_TransformIntoDisplayableImageExp [▶ 819]。

相关函数

- F_VN_CopyIntoDisplayableImage [▶ 781]
- F_VN_TransformIntoDisplayableImageExp [▶ 819]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.6.49 F_VN_TransformIntoDisplayableImageExp



将图像转换为可显示图像。源图像将被释放并归零，而现有的目标图像将被释放并被函数结果覆盖。如果 bAllowDeepCopy 等于 false，则源图像不得在其它任何地方使用，而且变换速度总是非常快。否则，如果 bAllowDeepCopy 等于 true，可能会根据需要创建源图像的深拷贝，这将导致执行时间延长。如果要在调用此函数后使用 ipSrcImage，请使用 F_VN_CopyIntoDisplayableImage 代替。（专家函数）

语法

定义：

```
FUNCTION F_VN_TransformIntoDisplayableImageExp : HRESULT
VAR_INPUT
    ipSrcImage     : Reference To ITcVnImage;
    ipDestImage    : Reference To ITcVnDisplayableImage;
    bAllowDeepCopy : BOOL;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	引用 ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnDisplayableImage [▶ 407]	返回可显示图像
bAllowDeepCopy	BOOL	如有需要，允许图像深拷贝
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

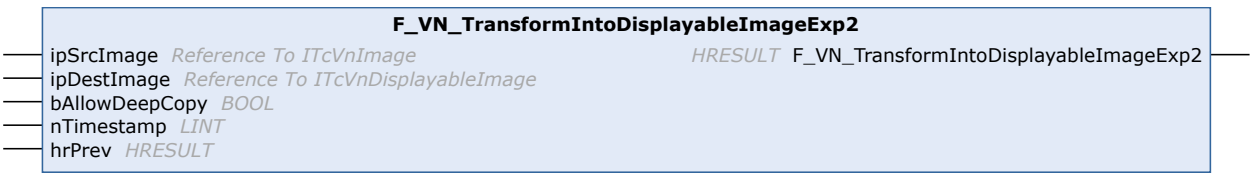
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.6.50 F_VN_TransformIntoDisplayableImageExp2



将图像转换为可显示图像，并设置指定时间戳，之后可通过 ITcVnImageTimestamps 读取该时间戳。源图像将被释放并归零，而现有的目标图像将被释放并被函数结果覆盖。如果 bAllowDeepCopy 等于 false，则源图像不得在其它任何地方使用，而且变换速度总是非常快。否则，如果 bAllowDeepCopy 等于 true，可能会根据需要创建源图像的深拷贝，这将导致执行时间延长。如果要在调用此函数后使用 ipSrcImage，请使用 F_VN_CopyIntoDisplayableImage 代替。（专家函数）

语法

定义：

```
FUNCTION F_VN_TransformIntoDisplayableImageExp2 : HRESULT
VAR_INPUT
    ipSrcImage     : Reference To ITcVnImage;
    ipDestImage    : Reference To ITcVnDisplayableImage;
    bAllowDeepCopy : BOOL;
    nTimestamp     : LINT;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	引用 ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnDisplayableImage [▶ 407]	返回可显示图像
bAllowDeepCopy	BOOL	如有需要，允许图像深拷贝
nTimestamp	LINT	添加到图像信息中的指定时间戳
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.7 编码质量

该组包含对各种一维条码和二维条码进行质量评估的函数。评估以国际标准为基础。编码质量检查的目的是确保相应的编码在使用时能被可靠地解码。

为了获得可比较的结果，这些标准规定了测量和评估特定属性以及得出编码质量总体评估的方法。其目的是尽可能预测在不同应用、环境和条件下，使用不同类型的识读器时，编码的预期识读率。因此，通常会为每种应用确定最低可接受质量水平。

一方面，标准描述了所使用的系统必须满足的要求，以便获得可用的结果。另一方面，标准提供了诸如光源、镜头设置和几何结构等建议，以便能够将评级与其他系统进行比较。为了建立一个符合标准的系统，最好先获得相关的质量评估规范和需要检测的编码。

这些函数也可用于非标准系统。这样做的唯一后果是，确定的质量无法与其他系统相比。一般来说，不同的系统在各个评估类别上会有细微差别。

根据 ISO/IEC 15416:2016 标准，Code 39、Code 128、EAN 8、EAN 13、ITF、UPC-A、UPC-E 编码类型的条形码打印质量可通过以下函数确定：

- F_VN_GradeBarcode(Exp) [▶ 821]

根据 ISO/IEC 15415:2011 标准，可以使用以下函数确定 ECC200 Data Matrix（数据矩阵）编码和二维码类型的二维矩阵符号的打印质量：

- F_VN_GradeDataMatrixCode(Exp) [▶ 827]
- F_VN_GradeQRCode(Exp) [▶ 832]

HRESULT 的解释

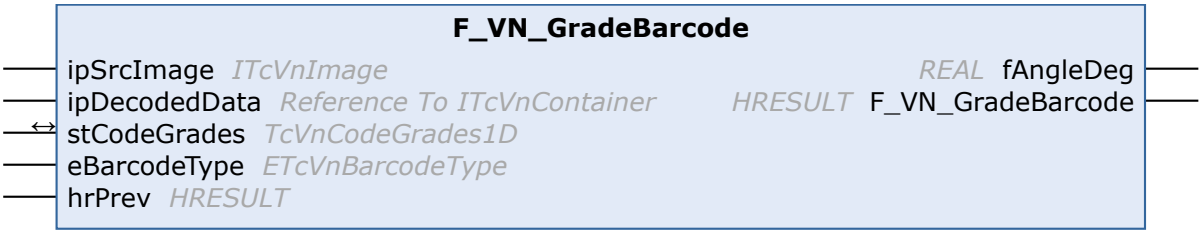
对于 Code Grading（编码分级）函数，HRESULT [▶ 135] 的返回值如下使用：

编码	名称	描述
16#000	S_OK	该函数已成功执行，所有评估均按照标准进行。
16#001	S_FALSE	函数已成功执行，但对于一维条码，并非所有扫描线都能解码，对于二维条码，则并非所有检查都能按照标准进行。因此，总分设为 0 分，但可以对各项结果进行评估。
16#256	s_watchdogtimeout	函数被 Watchdog（看门狗）中止。可根据终止时间对各项结果进行评估
16#70C	NOTFOUND	函数未成功执行，一维条码无法解码扫描线，二维条码未找到或无法解码。
16#7xx	所有错误代码	函数未成功执行。请参见：ADS Return Codes（ADS 返回代码） [▶ 2857]

还请参阅有关此

编码质量 [▶ 2728]

6.1.4.7.1 F_VN_GradeBarcode



根据 ISO/IEC 15416:2016 标准对 1D 条形码进行分级。需要均匀照度、良好的对焦以及图像传感器平面与编码平面平行，才能获得有意义的结果。

语法

定义：

```
FUNCTION F_VN_GradeBarcode : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDecodedData : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    stCodeGrades  : TcVnCodeGrades1D;
END_VAR
VAR_INPUT
    eBarcodeType  : ETcVnBarcodeType;
    hrPrev        : HRESULT;
END_VAR
VAR_OUTPUT
    fAngleDeg     : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像，仅包含水平对齐的条形码区域（USINT，1 通道，包括左右两侧的安静区，不包括顶部或底部超过 10% 总高度的文本或空白空间）
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码数据（ContainerType_String_SINT）
eBarcodeType	ETcVnBarcodeType [▶ 158]	条形码类型（支持：CODE39、CODE128、EAN8、EAN13、ITF、UPCA、UPCE）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stCodeGrades	TcVnCodeGrades1D [▶ 223]	返回编码等级，范围从 0（非常差）至 4（非常好）

输出

名称	类型	描述
fAngleDeg	REAL	以度为单位返回大致的顺时针旋转角度

返回值

HRESULT [▶ 135]

更多信息

`F_VN_GradeBarcode` 函数会根据 ISO/IEC 15416:2016 标准确定条形码的打印质量。

算法

该函数首先会搜索图像中水平排列的编码，然后搜索垂直排列的编码。

参数

输入图像

输入图像 `ipSrcImage` 必须是元素类型为 `USINT`（8 位）的单通道图像。输入图像可能只包含要检查的编码，包括相应的安静区。条形码必须水平或垂直对齐，上下的文字或空白不得超过总高度的 10%。水平排列时，安静区位于代码的左右两侧。

要获得有意义的结果，需要均匀照度，不存在曝光不足或曝光过度，编码平面对焦良好以及图像传感器平面与其平行对齐。此外，图像分辨率必须足够高，以匹配线宽。对于条形码，最小条形宽度至少应覆盖 3 个像素，这样才能获得良好的效果。

读取数据

从编码中读取的数据以字符串形式返回容器 `ipDecodedData` 中。容器类型为 `ContainerType_String_SINT`。成功找到编码并解码后，便可以使用函数 `F_VN_ExportContainer_String` 将内容导出为字符串。

```
hr := F_VN_ExportSubContainer_String(ipDecodedData, sText, nMaxLength, hr);
```

不同的解码结果

使用相应的参考算法进行解码会导致结果出现偏差。因此，编码内容会被其他设备或函数读取，但参考算法却失效。

质量评估报告

类型为 TcVnCodeGrades1D 的返回值 `stCodeGrades` 包含已确定的各个结果和总体结果 `fOverall`。质量等级从 0（很差）到 4（很好）不等。结果是几条扫描线的平均值。

参数	描述
fDecode	评估编码是否能用参考算法解码。
fSymbolContrast	评估全局符号对比度，即整个扫描线上最亮和最暗值之间的差值。
fMinReflectance	评估扫描线最暗值与最亮值的比率。
fMinEdgeContrast	评估扫描线相邻元素的最小局部对比度。
fModulation	评估最小局部对比度与全局符号对比度之间的比率。
fDefects	评估元素和安静区的不规则性。
fDecodability	评估使用参考算法解码的可靠程度。
fOverall	编码的总分是所有扫描线在各项测试中取得的最低分数的平均值。如果不同的扫描线产生不同的解码数据，则总分为 0。

条形码类型

参数 `eBarcodeType` 定义了要评估打印质量的条形码类型。支持枚举 ETcVnBarcodeType [▶ 158] 中的以下条形码类型：

- `TCVN_BT_CODE39`
- `TCVN_BT_CODE128`
- `TCVN_BT_EAN8`
- `TCVN_BT_EAN13`
- `TCVN_BT_ITF`
- `TCVN_BT_UPCA`
- `TCVN_BT_UPCE`

旋转角度（返回值）

`fAngleDeg` 以 90°为单位返回顺时针旋转角度。

专家参数

专家变体 F_VN_GradeBarcodeExp [▶ 824] 包含附加参数。

应用

例如，条形码的打印质量可以通过这种方式来确定：

```
hr := F_VN_GradeBarcode(
    ipSrcImage       := ipSrcImage,
    ipDecodedData    := ipDecodedData,
    stCodeGrades     := stCodeGrades,
    eBarcodeType     := eBarcodeType,
    hrPrev           := hr,
    fAngleDeg        => fAngleDeg);
```

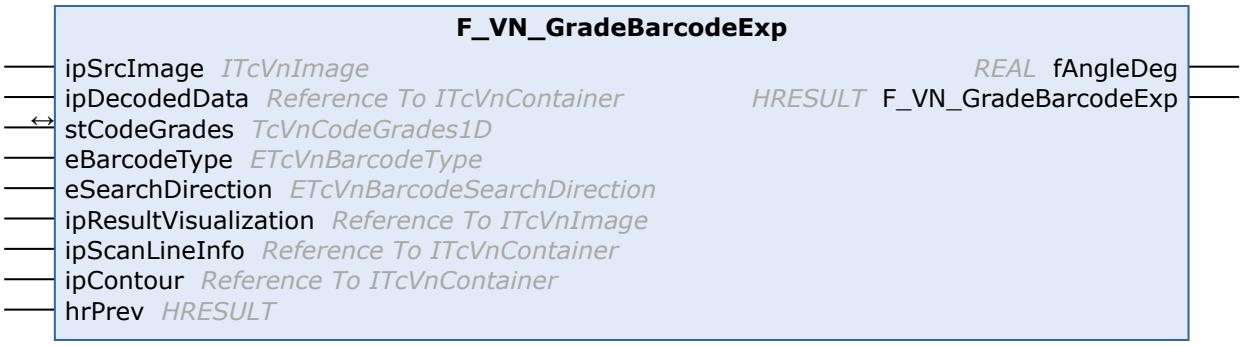
所需的许可证

TC3 Vision Code Quality

还请参阅有关此

编码质量 [▶ 2728]

6.1.4.7.2 F_VN_GradeBarcodeExp



根据 ISO/IEC 15416:2016 标准对 1D 条形码进行分级。需要均匀照度、良好的对焦以及图像传感器平面与编码平面平行，才能获得有意义的结果。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_GradeBarcodeExp : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
    ipDecodedData         : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    stCodeGrades          : TcVnCodeGrades1D;
END_VAR
VAR_INPUT
    eBarcodeType          : ETcVnBarcodeType;
    eSearchDirection      : ETcVnBarcodeSearchDirection;
    ipResultVisualization : Reference To ITcVnImage;
    ipScanLineInfo        : Reference To ITcVnContainer;
    ipContour             : Reference To ITcVnContainer;
    hrPrev                : HRESULT;
END_VAR
VAR_OUTPUT
    fAngleDeg             : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像，仅包含水平对齐的条形码区域（USINT，1 通道，包括左右两侧的安静区，不包括顶部或底部超过 10% 总高度的文本或空白空间）
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码数据（ContainerType_String_SINT）
eBarcodeType	ETcVnBarcodeType [▶ 158]	条形码类型（支持：CODE39、CODE128、EAN8、EAN13、ITF、UPCA、UPCE）
eSearchDirection	ETcVnBarcodeSearchDirection [▶ 158]	条形码搜索方向（BSD_ANY 首先尝试水平方向，然后是垂直方向）
ipResultVisualization	引用 ITcVnImage [▶ 407]	返回扫描线的可视化信息（可选项，如果不需要，则设为 0）。等级用颜色标示：>= 3 为绿色，>= 2 为浅蓝色，>= 1 为橙色，< 1 为红色
ipScanLineInfo	引用 ITcVnContainer [▶ 365]	返回用于进一步评估的附加扫描线信息，例如用于可视化的 PlotIntensityProfile（可选项，如果不需要，则设为 0；ContainerType_Vector_TcVnVector3_REAL，包含 [行索引、整体等级、阈值]）

名称	类型	描述
ipContour	引用 ITcVnContainer [▶ 365]	返回 4 个大致编码角点，包括安静区（可选项，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_DINT）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stCodeGrades	TcVnCodeGrades1D [▶ 223]	返回编码等级，范围从 0（非常差）至 4（非常好）

输出

名称	类型	描述
fAngleDeg	REAL	以度为单位返回大致的顺时针旋转角度

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_GradeBarcodeExp` 是 F_VN_GradeBarcode [▶ 821] 的专家变体。它包含额外的参数。

参数

输入图像

输入图像 `ipSrcImage` 必须是元素类型为 `USINT`（8 位）的单通道图像。输入图像可能只包含要检查的编码，包括相应的安静区。条形码必须水平或垂直对齐，上下的文字或空白不得超过总高度的 10%。水平排列时，安静区位于代码的左右两侧。

要获得有意义的结果，需要均匀照度，不存在曝光不足或曝光过度，编码平面对焦良好以及图像传感器平面与其平行对齐。此外，图像分辨率必须足够高，以匹配线宽。对于条形码，最小条形宽度至少应覆盖 3 个像素，这样才能获得良好的效果。

读取数据

从编码中读取的数据以字符串形式返回容器 `ipDecodedData` 中。容器类型为 `ContainerType_String_SINT`。成功找到编码并解码后，便可以使用函数 `F_VN_ExportContainer_String` 将内容导出为字符串。

```
hr := F_VN_ExportSubContainer_String(ipDecodedData, sText, nMaxLength, hr);
```

不同的解码结果

使用相应的参考算法进行解码会导致结果出现偏差。因此，编码内容会被其他设备或函数读取，但参考算法却失效。

质量评估报告

类型为 TcVnCodeGrades1D 的返回值 `stCodeGrades` 包含已确定的各个结果和总体结果 `fOverall`。质量等级从 0（很差）到 4（很好）不等。结果是几条扫描线的平均值。

参数	描述
fDecode	评估编码是否能用参考算法解码。
fSymbolContrast	评估全局符号对比度，即整个扫描线上最亮和最暗值之间的差值。

参数	描述
fMinReflectance	评估扫描线最暗值与最亮值的比率。
fMinEdgeContrast	评估扫描线相邻元素的最小局部对比度。
fModulation	评估最小局部对比度与全局符号对比度之间的比率。
fDefects	评估元素和安静区的不规则性。
fDecodability	评估使用参考算法解码的可靠程度。
fOverall	编码的总分是所有扫描线在各项测试中取得的最低分数的平均值。如果不同的扫描线产生不同的解码数据，则总分为 0。

条形码类型

参数 `eBarcodeType` 定义了要评估打印质量的条形码类型。支持枚举 ETcVnBarcodeType [▶ 158] 中的以下条形码类型：

- `TCVN_BT_CODE39`
- `TCVN_BT_CODE128`
- `TCVN_BT_EAN8`
- `TCVN_BT_EAN13`
- `TCVN_BT_ITF`
- `TCVN_BT_UPCA`
- `TCVN_BT_UPCE`

搜索方向

可通过 `eSearchDirection` 指定搜索方向：

- 通过 `TCVN_BSD_ANY`，首先在水平方向上进行搜索，然后在垂直方向上进行搜索。
- 通过 `TCVN_BSD_HORIZONTAL`，仅在水平方向上进行搜索。
- 通过 `TCVN_BSD_VERTICAL`，仅在垂直方向上进行搜索。

结果呈现

`ipResultVisualization` 会返回一个图像，其中包含扫描线和相应评估结果的可视化。线条根据各个等级用不同颜色标示：`>= 3` 为绿色，`>= 2` 为浅蓝色，`>= 1` 为橙色，`< 1` 为红色。该参数为可选参数，如果不需要，可设置为 0。

扫描线信息

`ipScanLineInfo` 会返回各个扫描线的其他信息，以便进行更详细的评估。返回值的类型为 `ContainerType_Vector_TcVnVector3_REAL`，并根据方向指定图像行或图像列的索引、总分和为每条扫描线确定的全局阈值。例如，可以使用 F_VN_PlotIntensityProfile [▶ 1050] 对各个线条进行可视化。该参数为可选参数，如果不需要，可设置为 0。

编码的位置

参数 `ipContour` 会将编码角点（包括安静区）返回为 `ContainerType_Vector_TcVnPoint2_DINT` 类型的四个等高线点。计算仅基于可解码的扫描线。例如，可以使用 F_VN_DrawContours [▶ 1012] 直接绘制等高线，也可以使用 Contour Analysis（等高线分析）函数进行进一步分析。该参数为可选参数，如果不需要，可设置为 0。

旋转角度（返回值）

`fAngleDeg` 以 90°为单位返回顺时针旋转角度。

应用

例如，条形码的打印质量可以通过这种方式来确定：

```
hr := F_VN_GradeBarcodeExp(
    ipSrcImage             := ipSrcImage,
    ipDecodedData          := ipDecodedData,
    stCodeGrades           := stCodeGrades,
    eBarcodeType           := eBarcodeType,
    eSearchDirection       := eSearchDirection,
    ipResultVisualization := ipResultVisualization,
    ipScanLineInfo         := ipScanLineInfo,
    ipContour              := ipContour,
    hrPrev                 := hr,
    fAngleDeg              => fAngleDeg);
```

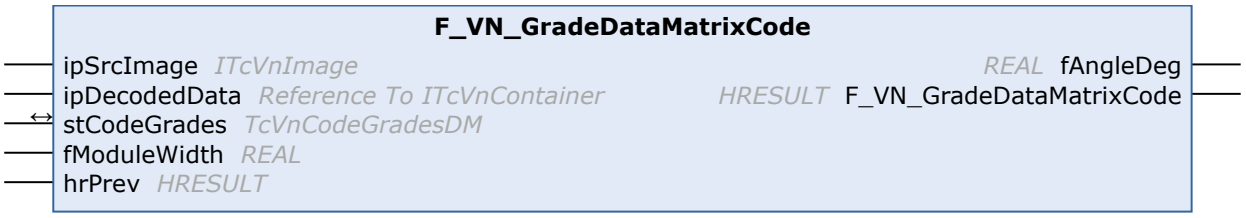
所需的许可证

TC3 Vision Code Quality

还请参阅有关此

- 编码质量 [▶ 2728]

6.1.4.7.3 F_VN_GradeDataMatrixCode



根据 ISO/IEC 15415:2011 标准对数据矩阵编码（ECC200）进行分级。需要均匀照度、良好的对焦以及图像传感器平面与编码平面平行，才能获得有意义的结果。
可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_GradeDataMatrixCode : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDecodedData : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    stCodeGrades  : TcVnCodeGradesDM;
END_VAR
VAR_INPUT
    fModuleWidth  : REAL;
    hrPrev        : HRESULT;
END_VAR
VAR_OUTPUT
    fAngleDeg     : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像，仅包含数据矩阵编码区域（USINT，1 通道，包括安静区）
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码数据（ContainerType_String_SINT）
fModuleWidth	REAL	图像中编码的最小模块宽度（单位：像素）。必须至少为 3，最好在 5 - 8 之间。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stCodeGrades	TcVnCodeGradesDM [▶ 224]	返回编码等级，范围从 0（非常差）至 4（非常好）

输出

名称	类型	描述
fAngleDeg	REAL	以度为单位返回顺时针旋转角度

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_GradeDataMatrixCode` 会根据 ISO/IEC 15415:2011 标准确定 ECC200 数据矩阵代码的打印质量。

参数

输入图像

输入图像 `ipSrcImage` 必须是元素类型为 `USINT`（8 位）的单通道图像。输入图像可能只包含要检查的代码，包括相应的安静区。

要获得有意义的结果，需要均匀照度，不存在曝光不足或曝光过度，编码平面对焦良好以及图像传感器平面与其平行对齐。此外，图像分辨率必须足够高，以满足模块尺寸的要求。对于二维条码，最小的模块至少应覆盖 5 个像素，这样才能获得良好的效果。

读取数据

从编码中读取的数据以字符串形式返回容器 `ipDecodedData` 中。容器类型为 `ContainerType_String_SINT`。成功找到编码并解码后，便可以使用函数 `F_VN_ExportContainer_String` 将内容导出为字符串。

```
hr := F_VN_ExportSubContainer_String(ipDecodedData, sText, nMaxLength, hr);
```

不同的解码结果

使用相应的参考算法进行解码会导致结果出现偏差。因此，编码内容会被其他设备或函数读取，但参考算法却失效。

质量评估报告

类型为 TcVnCodeGradesDM 的返回值 `stCodeGrades` 包含已确定的各个结果和总体结果 `nOverall`。质量等级从 0（很差）到 4（很好）不等。

参数	描述
nDecode	评估代码是否能用参考算法解码。
nSymbolContrast	评估全局符号对比度，即测试区域内最亮和最暗值之间的差异。
nModulation	评估暗模块或亮模块的均匀性。
nReflectanceMargin	评估与全局阈值相比，每个模块能被正确区分为亮或暗的程度。
nFixedPatternDamage	评估各种固定代码组件，如查找器或对齐模式。
nAxialNonuniformity	评估代码的统一缩放。
nGridNonuniformity	评估代码网格的偏差。
nUnusedErrorCorrection	评估错误校正的剩余容量。

参数	描述
nOverall	编码的总评分是各项考试的最低评分。

最小模块宽度

用 fModuleWidth 指定编码的预期最小模块宽度（单位：像素）。如果图像中的模块大小与此相差很大，就会导致结果不准确或没有结果，并增加计算时间。

旋转角度（返回值）

fAngleDeg 会返回顺时针旋转角度（单位：度）。

专家参数

专家变体 F_VN_GradeDataMatrixCodeExp [▶ 829] 包含附加参数。

应用

例如，Data Matrix Code（数据矩阵代码）的打印质量可以用这种方法来确定：

```
hr := F_VN_GradeDataMatrixCode(
    ipSrcImage       := ipSrcImage,
    ipDecodedData    := ipDecodedData,
    stCodeGrades     := stCodeGrades,
    fModuleWidth     := fModuleWidth,
    hrPrev           := hr,
    fAngleDeg        => fAngleDeg);
```

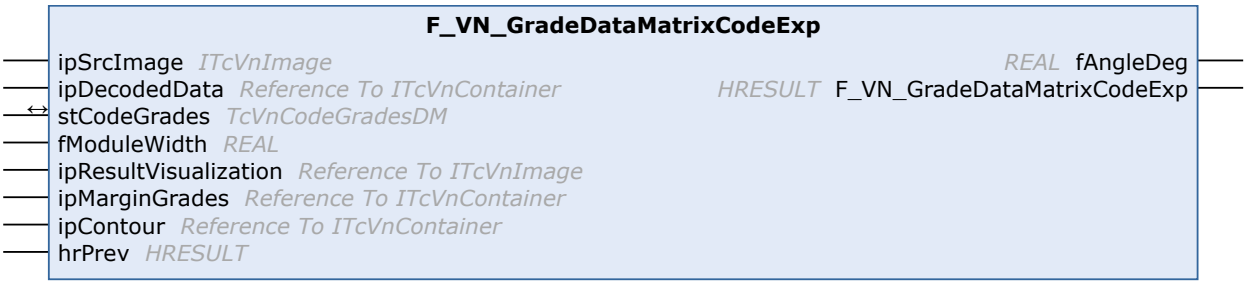
所需的许可证

TC3 Vision Code Quality

还请参阅有关此

- 编码质量 [▶ 2728]

6.1.4.7.4 F_VN_GradeDataMatrixCodeExp



根据 ISO/IEC 15415:2011 标准对数据矩阵编码（ECC200）进行分级。需要均匀照度、良好的对焦以及图像传感器平面与编码平面平行，才能获得有意义的结果。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被看门狗取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_GradeDataMatrixCodeExp : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
    ipDecodedData         : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    stCodeGrades          : TcVnCodeGradesDM;
END_VAR
VAR_INPUT
    fModuleWidth          : REAL;
    ipResultVisualization : Reference To ITcVnImage;
    ipMarginGrades        : Reference To ITcVnContainer;
```

```
    ipContour               : Reference To ITcVnContainer;
    hrPrev                  : HRESULT;
END_VAR
VAR_OUTPUT
    fAngleDeg               : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像，仅包含数据矩阵编码区域（USINT，1 通道，包括安静区）
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码数据（ContainerType_String_SINT）
fModuleWidth	REAL	图像中编码的最小模块宽度（单位：像素）。必须至少为 3，最好在 5 - 8 之间。
ipResultVisualization	引用 ITcVnImage [▶ 407]	返回扫描网格的可视化信息（可选项，如果不需要，则设为 0）。模块边距等级用颜色标示：>= 3 为绿色，>= 2 为浅蓝色，>= 1 为橙色，< 1 为红色
ipMarginGrades	引用 ITcVnContainer [▶ 365]	返回每个模块的单个边距等级，例如用于创建自定义可视化（可选项，如果不需要，则设为 0；ContainerType_Vector_TcVnVector3_REAL，包含 [x 位置、y 位置、等级]）
ipContour	引用 ITcVnContainer [▶ 365]	返回 4 个编码角点，不包括安静区（可选项，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_DINT）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stCodeGrades	TcVnCodeGradesDM [▶ 224]	返回编码等级，范围从 0（非常差）至 4（非常好）

输出

名称	类型	描述
fAngleDeg	REAL	以度为单位返回顺时针旋转角度

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_GradeDataMatrixCodeExp` 是 F_VN_GradeDataMatrixCode [▶ 827] 的专家变体。它包含额外的参数。

参数

输入图像

输入图像 `ipSrcImage` 必须是元素类型为 `USINT`（8 位）的单通道图像。输入图像可能只包含要检查的代码，包括相应的安静区。

要获得有意义的结果，需要均匀照度，不存在曝光不足或曝光过度，编码平面对焦良好以及图像传感器平面与其平行对齐。此外，图像分辨率必须足够高，以满足模块尺寸的要求。对于二维条码，最小的模块至少应覆盖 5 个像素，这样才能获得良好的效果。

读取数据

从编码中读取的数据以字符串形式返回容器 `ipDecodedData` 中。容器类型为 `ContainerType_String_SINT`。成功找到编码并解码后，便可以使用函数 `F_VN_ExportContainer_String` 将内容导出为字符串。

```
hr := F_VN_ExportSubContainer_String(ipDecodedData, sText, nMaxLength, hr);
```

不同的解码结果

使用相应的参考算法进行解码会导致结果出现偏差。因此，编码内容会被其他设备或函数读取，但参考算法却失效。

质量评估报告

类型为 TcVnCodeGradesDM 的返回值 `stCodeGrades` 包含已确定的各个结果和总体结果 `nOverall`。质量等级从 0（很差）到 4（很好）不等。

参数	描述
nDecode	评估代码是否能用参考算法解码。
nSymbolContrast	评估全局符号对比度，即测试区域内最亮和最暗值之间的差异。
nModulation	评估暗模块或亮模块的均匀性。
nReflectanceMargin	评估与全局阈值相比，每个模块能被正确区分为亮或暗的程度。
nFixedPatternDamage	评估各种固定代码组件，如查找器或对齐模式。
nAxialNonuniformity	评估代码的统一缩放。
nGridNonuniformity	评估代码网格的偏差。
nUnusedErrorCorrection	评估错误校正的剩余容量。
nOverall	编码的总评分是各项考试的最低评分。

最小模块宽度

用 `fModuleWidth` 指定编码的预期最小模块宽度（单位：像素）。如果图像中的模块大小与此相差很大，就会导致结果不准确或没有结果，并增加计算时间。

结果呈现

`ipResultVisualization` 会返回一个图像，其中包含扫描网格和相应评估结果的可视化。模块按各个等级用颜色标示：`>= 3` 为绿色，`>= 2` 为浅蓝色，`>= 1` 为橙色，`< 1` 为红色。该参数为可选参数，如果不需要，可设置为 0。

边距信息

`ipMarginGrades` 会返回每个模块的各个边距评级的更多信息，以便进行更详细的评估。`ContainerType_Vector_TcVnVector3_REAL` 类型的返回向量包含中心点的 X 和 Y 位置以及每个模块的等级。该参数为可选参数，如果不需要，可设置为 0。

编码的位置

参数 `ipContour` 会将编码角点作为类型 `ContainerType_Vector_TcVnPoint2_DINT` 的四个轮廓点返回。角点位于外部模块的边缘。例如，可以使用 F_VN_DrawContours [▶ 1012] 直接绘制等高线，也可以使用 Contour Analysis（等高线分析）函数进行进一步分析。该参数为可选参数，如果不需要，可设置为 0。

旋转角度（返回值）

`fAngleDeg` 会返回顺时针旋转角度（单位：度）。

应用

例如，Data Matrix Code（数据矩阵编码）的打印质量可以用这种方法来确定：

```
hr := F_VN_GradeDataMatrixCodeExp(
    ipSrcImage              := ipSrcImage,
    ipDecodedData           := ipDecodedData,
    stCodeGrades            := stCodeGrades,
    fModuleWidth            := fModuleWidth,
    ipResultVisualization   := ipResultVisualization,
    ipMarginGrades          := ipMarginGrades,
    ipContour               := ipContour,
    hrPrev                  := hr,
    fAngleDeg               => fAngleDeg);
```

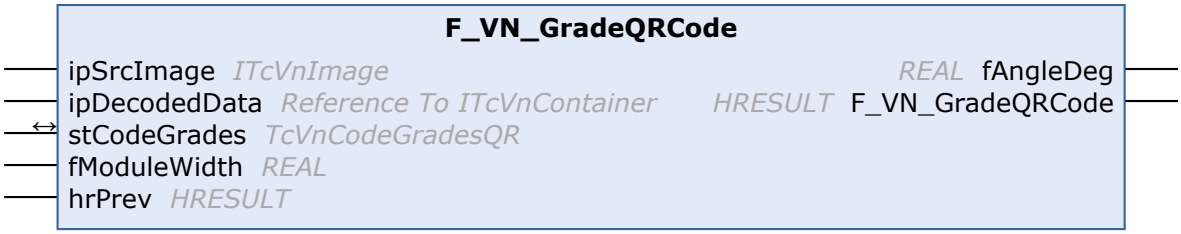
所需的许可证

TC3 Vision Code Quality

还请参阅有关此

编码质量 [▶ 2728]

6.1.4.7.5 F_VN_GradeQRCode



根据 ISO/IEC 15415:2011 标准对二维码进行分级。需要均匀照度、良好的对焦以及图像传感器平面与编码平面平行，才能获得有意义的结果。
可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_GradeQRCode : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDecodedData : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    stCodeGrades  : TcVnCodeGradesQR;
END_VAR
VAR_INPUT
    fModuleWidth  : REAL;
    hrPrev        : HRESULT;
END_VAR
VAR_OUTPUT
    fAngleDeg     : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像，仅包含二维码区域（USINT，1 通道，包括安静区）
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码数据（ContainerType_String_SINT）
fModuleWidth	REAL	图像中编码的最小模块宽度（单位：像素）。必须至少为 3，最好在 5 - 8 之间。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stCodeGrades	TcVnCodeGradesQR [▶ 224]	返回编码等级，范围从 0（非常差）至 4（非常好）

输出

名称	类型	描述
fAngleDeg	REAL	以度为单位返回顺时针旋转角度

返回值

HRESULT [▶ 135]

更多信息

`F_VN_GradeQRCode` 函数根据 ISO/IEC 15415:2011 标准确定二维码的打印质量。

参数

输入图像

输入图像 `ipSrcImage` 必须是元素类型为 `USINT`（8 位）的单通道图像。输入图像可能只包含要检查的代码，包括相应的安静区。

要获得有意义的结果，需要均匀照度，不存在曝光不足或曝光过度，编码平面对焦良好以及图像传感器平面与其平行对齐。此外，图像分辨率必须足够高，以满足模块尺寸的要求。对于二维条码，最小的模块至少应覆盖 5 个像素，这样才能获得良好的效果。

读取数据

从编码中读取的数据以字符串形式返回容器 `ipDecodedData` 中。容器类型为 `ContainerType_String_SINT`。成功找到编码并解码后，便可以使用函数 `F_VN_ExportContainer_String` 将内容导出为字符串。

```
hr := F_VN_ExportSubContainer_String(ipDecodedData, sText, nMaxLength, hr);
```

不同的解码结果

使用相应的参考算法进行解码会导致结果出现偏差。因此，编码内容会被其他设备或函数读取，但参考算法却失效。

质量评估报告

类型为 TcVnCodeGradesQR 的返回值 `stCodeGrades` 包含已确定的单个结果和总结果 `nOverall`。质量等级从 0（很差）到 4（很好）不等。

参数	描述
nDecode	评估代码是否能用参考算法解码。
nSymbolContrast	评估全局符号对比度，即测试区域内最亮和最暗值之间的差异。
nModulation	评估暗模块或亮模块的均匀性。
nReflectanceMargin	评估与全局阈值相比，每个模块能被正确区分为亮或暗的程度。
nFixedPatternDamage	评估各种固定代码组件，如查找器或对齐模式。
nAxialNonuniformity	评估代码的统一缩放。
nGridNonuniformity	评估代码网格的偏差。
nUnusedErrorCorrection	评估错误校正的剩余容量。

参数	描述
nFormatInfo	评估围绕查找器模式编码的格式信息。
nVersionInfo	从第 7 版开始评估代码中包含的版本信息。
nOverall	编码的总评分是各项考试的最低评分。

最小模块宽度

用 `fModuleWidth` 指定编码的预期最小模块宽度（单位：像素）。如果图像中的模块大小与此相差很大，就会导致结果不准确或没有结果，并增加计算时间。

旋转角度（返回值）

`fAngleDeg` 会返回顺时针旋转角度（单位：度）。

专家参数

专家变体 F_VN_GradeQRCodeExp [▶ 834] 包含附加参数。

应用

例如，二维码的打印质量就可以通过这种方法来确定：

```
hr := F_VN_GradeQRCode(
    ipSrcImage        := ipSrcImage,
    ipDecodedData     := ipDecodedData,
    stCodeGrades      := stCodeGrades,
    fModuleWidth      := fModuleWidth,
    hrPrev            := hr,
    fAngleDeg         => fAngleDeg);
```

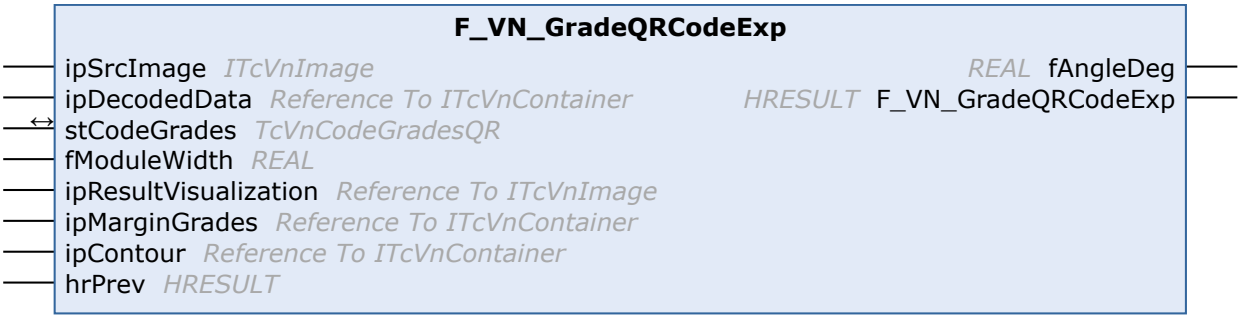
所需的许可证

TC3 Vision Code Quality

还请参阅有关此

编码质量 [▶ 2728]

6.1.4.7.6 F_VN_GradeQRCodeExp



根据 ISO/IEC 15415:2011 标准对二维码进行分级。需要均匀照度、良好的对焦以及图像传感器平面与编码平面平行，才能获得有意义的结果。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被看门狗取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_GradeQRCodeExp : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
    ipDecodedData         : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
```

```
    stCodeGrades          : TcVnCodeGradesQR;
END_VAR
VAR_INPUT
    fModuleWidth          : REAL;
    ipResultVisualization : Reference To ITcVnImage;
    ipMarginGrades        : Reference To ITcVnContainer;
    ipContour             : Reference To ITcVnContainer;
    hrPrev                : HRESULT;
END_VAR
VAR_OUTPUT
    fAngleDeg             : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像，仅包含二维码区域（USINT，1 通道，包括安静区）
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码数据（ContainerType_String_SINT）
fModuleWidth	REAL	图像中编码的最小模块宽度（单位：像素）。必须至少为 3，最好在 5 - 8 之间。
ipResultVisualization	引用 ITcVnImage [▶ 407]	返回扫描网格的可视化信息（可选项，如果不需要，则设为 0）。模块边距等级用颜色标示：>= 3 为绿色，>= 2 为浅蓝色，>= 1 为橙色，< 1 为红色
ipMarginGrades	引用 ITcVnContainer [▶ 365]	返回每个模块的单个边距等级，例如用于创建自定义可视化（可选项，如果不需要，则设为 0；ContainerType_Vector_TcVnVector3_REAL，包含 [x 位置、y 位置、等级]）
ipContour	引用 ITcVnContainer [▶ 365]	返回 4 个编码角点，不包括安静区（可选项，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_DINT）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stCodeGrades	TcVnCodeGradesQR [▶ 224]	返回编码等级，范围从 0（非常差）至 4（非常好）

输出

名称	类型	描述
fAngleDeg	REAL	以度为单位返回顺时针旋转角度

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_GradeQRCodeExp` 是 F_VN_GradeQRCode [▶ 832] 的专家变体。它包含额外的参数。

参数

输入图像

输入图像 `ipSrcImage` 必须是元素类型为 `USINT`（8 位）的单通道图像。输入图像可能只包含要检查的代码，包括相应的安静区。

要获得有意义的结果，需要均匀照度，不存在曝光不足或曝光过度，编码平面对焦良好以及图像传感器平面与其平行对齐。此外，图像分辨率必须足够高，以满足模块尺寸的要求。对于二维条码，最小的模块至少应覆盖 5 个像素，这样才能获得良好的效果。

读取数据

从编码中读取的数据以字符串形式返回容器 `ipDecodedData` 中。容器类型为 `ContainerType_String_SINT`。成功找到编码并解码后，便可以使用函数 `F_VN_ExportContainer_String` 将内容导出为字符串。

```
hr := F_VN_ExportSubContainer_String(ipDecodedData, sText, nMaxLength, hr);
```

不同的解码结果

使用相应的参考算法进行解码会导致结果出现偏差。因此，编码内容会被其他设备或函数读取，但参考算法却失效。

质量评估报告

类型为 TcVnCodeGradesQR 的返回值 `stCodeGrades` 包含已确定的单个结果和总结果 `nOverall`。质量等级从 0（很差）到 4（很好）不等。

参数	描述
nDecode	评估代码是否能用参考算法解码。
nSymbolContrast	评估全局符号对比度，即测试区域内最亮和最暗值之间的差异。
nModulation	评估暗模块或亮模块的均匀性。
nReflectanceMargin	评估与全局阈值相比，每个模块能被正确区分为亮或暗的程度。
nFixedPatternDamage	评估各种固定代码组件，如查找器或对齐模式。
nAxialNonuniformity	评估代码的统一缩放。
nGridNonuniformity	评估代码网格的偏差。
nUnusedErrorCorrection	评估错误校正的剩余容量。
nFormatInfo	评估围绕查找器模式编码的格式信息。
nVersionInfo	从第 7 版开始评估代码中包含的版本信息。
nOverall	编码的总评分是各项考试的最低评分。

最小模块宽度

用 `fModuleWidth` 指定编码的预期最小模块宽度（单位：像素）。如果图像中的模块大小与此相差很大，就会导致结果不准确或没有结果，并增加计算时间。

结果呈现

`ipResultVisualization` 会返回一个图像，其中包含扫描网格和相应评估结果的可视化。模块按各个等级用颜色标示：`>= 3` 为绿色，`>= 2` 为浅蓝色，`>= 1` 为橙色，`< 1` 为红色。该参数为可选参数，如果不需要，可设置为 0。

边距信息

`ipMarginGrades` 会返回每个模块的各个边距评级的更多信息，以便进行更详细的评估。`ContainerType_Vector_TcVnVector3_REAL` 类型的返回向量包含中心点的 X 和 Y 位置以及每个模块的等级。该参数为可选参数，如果不需要，可设置为 0。

编码的位置

参数 `ipContour` 会将编码角点作为类型 `ContainerType_Vector_TcVnPoint2_DINT` 的四个轮廓点返回。角点位于外部模块的中间。例如，可以使用 F_VN_DrawContours [▶ 1012] 直接绘制等高线，也可以使用 Contour Analysis（等高线分析）函数进行进一步分析。该参数为可选参数，如果不需要，可设置为 0。

旋转角度（返回值）

`fAngleDeg` 会返回顺时针旋转角度（单位：度）。

应用

例如，二维码的打印质量就可以通过这种方法来确定：

```
hr := F_VN_GradeQRCodeExp(
    ipSrcImage              := ipSrcImage,
    ipDecodedData           := ipDecodedData,
    stCodeGrades            := stCodeGrades,
    fModuleWidth            := fModuleWidth,
    ipResultVisualization   := ipResultVisualization,
    ipMarginGrades          := ipMarginGrades,
    ipContour               := ipContour,
    hrPrev                  := hr,
    fAngleDeg               => fAngleDeg);
```

所需的许可证

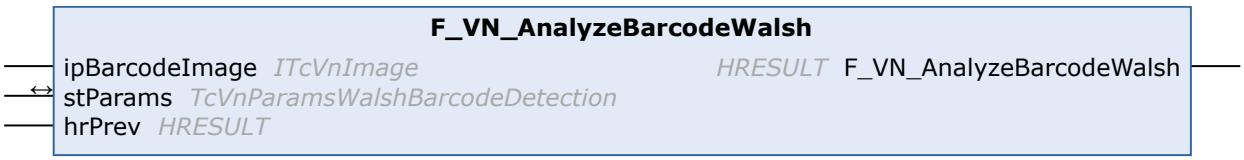
TC3 Vision Code Quality

还请参阅有关此

- 编码质量 [▶ 2728]

6.1.4.8 编码读取

6.1.4.8.1 F_VN_AnalyzeBarcodeWalsh



Analyze an exemplary barcode image and estimate the following parameters for calls of F_VN_DetectBarcodesWalsh: nMaxBarDistance, fMinAspectRatio, fMaxAspectRatio, nIntensityDifference, nMinArea, and nChannelIndex. The depicted barcode must be representative for the barcodes to be detected by F_VN_DetectBarcodesWalsh.
Can use available TwinCAT Job Tasks for executing parallel code regions.

Syntax

Definition:

```
FUNCTION F_VN_AnalyzeBarcodeWalsh : HRESULT
VAR_INPUT
    ipBarcodeImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    stParams       : TcVnParamsWalshBarcodeDetection;
END_VAR
VAR_INPUT
    hrPrev         : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipBarcodeImage	ITcVnImage [▶ 407]	Image showing a horizontal barcode surrounded by a narrow non-barcode region in each direction. The total size of the non-barcode region must not exceed 30% of the image size.
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

In/Outputs

Name	Type	Description
stParams	TcVnParamsWalshBarcodeDetection [▶ 218]	Returns the estimated parameters for use with F_VN_DetectBarcodesWalsh. The values of the other parameters are not changed.

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Code Reading

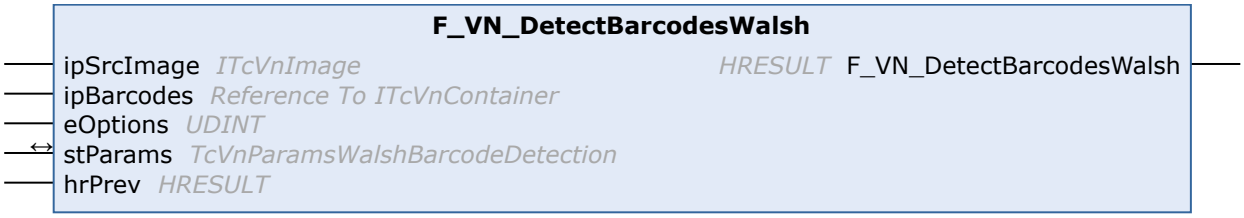
系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

还请参阅有关此

- F_VN_DetectBarcodesWalsh [▶ 838]
- TcVnParamsWalshBarcodeDetection [▶ 218]

6.1.4.8.2 F_VN_DetectBarcodesWalsh



Detect barcodes using the Walsh transform.
Can use available TwinCAT Job Tasks for executing parallel code regions.

Syntax

Definition:

```
FUNCTION F_VN_DetectBarcodesWalsh : HRESULT
VAR_INPUT
    ipSrcImage : ITcVnImage;
    ipBarcodes : Reference To ITcVnContainer;
    eOptions   : UDINT;
END_VAR
VAR_IN_OUT
    stParams   : TcVnParamsWalshBarcodeDetection;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipSrcImage	ITcVnImage [▶ 407]	Source image
ipBarcodes	Reference To ITcVnContainer [▶ 365]	Returns the code positions as rotated rectangles (ContainerType_Vector_TcVnRotatedRectangle; Non-zero interface pointers are reused.)

Name	Type	Description
eOptions	UDINT	General flags controlling the detection process. (see ETcVnDetectBarcodesWalshOptions; default: TCVN_DBWO_HORIZONTAL)
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

In/Outputs

Name	Type	Description
stParams	TcVnParamsWalshBarcodeDetection [▶ 218]	Parameters for configuring the individual detection steps.

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Code Reading

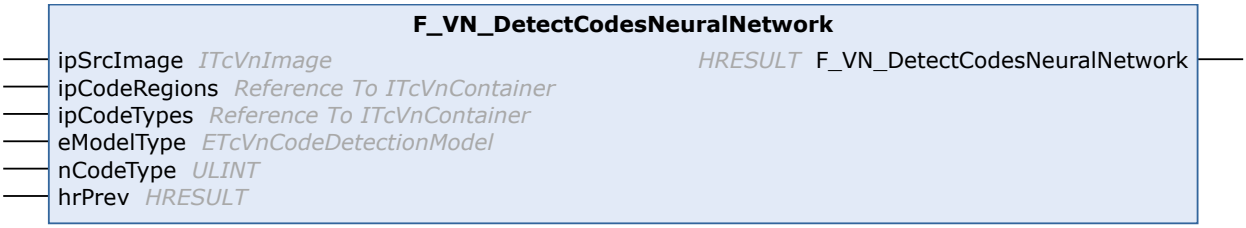
系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

还请参阅有关此

- F_VN_DetectCodesNeuralNetwork [▶ 839]
- ETcVnDetectBarcodesWalshOptions [▶ 154]
- TcVnParamsWalshBarcodeDetection [▶ 218]
- F_VN_AnalyzeBarcodeWalsh [▶ 837]

6.1.4.8.3 F_VN_DetectCodesNeuralNetwork



使用基于 eModelType 的经过训练的神经网络模型，在提供的图像中检测各种编码类型，例如条形码、QR 码、数据矩阵码和 DotCode (点码）。检测到的编码区域和类型（如支持）将按其模型置信度从高到低排序。可以使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_DetectCodesNeuralNetwork : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipCodeRegions : Reference To ITcVnContainer;
    ipCodeTypes   : Reference To ITcVnContainer;
    eModelType    : ETcVnCodeDetectionModel;
    nCodeType     : ULINT;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 元素，1 或 3 通道）。该函数会自动将输入归一化并转换为选定 eModelType 所需的格式
ipCodeRegions	引用 ITcVnContainer [▶ 365]	根据 eModelType 以直立矩形或旋转矩形的形式返回码位置（ContainerType_Vector_TcVnRectangle_DINT 或 ContainerType_Vector_TcVnRotatedRectangle；重复使用非零接口指针。）
ipCodeTypes	引用 ITcVnContainer [▶ 365]	返回每个检测到的编码（ContainerType_Vector_ULINT；重复使用非零接口指针）的编码类型（ETcVnCodeType）；如果不需要，则设为 0。
eModelType	ETcVnCodeDetectionModel [▶ 164]	用于编码检测的神经网络模型的类型。提供多种不同的神经网络模型，包括专用于一维条码、二维码或所有编码类型的模型。
nCodeType	ULINT	指定要检测的编码类型。通过执行按位 OR 运算，可以将 ETcVnCodeType 枚举中的多个值组合在一起。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

`F_VN_DetectCodesNeuralNetwork` 函数使用神经网络来定位输入图像中的一维条码和二维码，并对其类型进行分类。与传统的读码功能不同，此处不对编码内容进行解码。`F_VN_DetectCodesNeuralNetwork` 用于预定位，神经网络模型以边界框和编码类型的形式提供概率性检测结果。

这些编码区域可以在下一步中使用相应的读码功能进行解码。由于指定了区域，编码的定位被限制在较小的图像区域内，从而减少了执行时间。这样还可以直接使用需要预定义 ROI 的读码功能。

参数

ipSrcImage

输入图像 `ipSrcImage` 必须是 `USINT` 类型（8 位）的单通道或三通道图像。

根据所选模型的不同，图像会在内部自动进行转换。模型的具体输入格式和尺寸在 ETcVnCodeDetectionModel [▶ 164] 的描述中列出。为了减少内部转换步骤，从而减少所需的处理时间，如果图像在调用前已具有相同通道数的格式，则建议直接以该格式传输图像。

通常应确保输入图像的长宽比与模型的长宽比尽可能接近。否则，内部缩放可能会导致几何畸变，从而影响检测质量。

ipCodeRegions

参数 `ipCodeRegions` 以 `ContainerType_Vector_TcVnRectangle_DINT` 或 `ContainerType_Vector_TcVnRotatedRectangle` 类型容器的形式返回在输入图像中定位到的编码区域。使用哪种容器类型取决于所传输的 `eModelType`。列表中的条目按置信度降序排列，从概率最高的条目开始。

ipCodeTypes

参数 `ipCodeTypes` 按照与 `ipCodeRegions` 相同的顺序返回识别出的编码的分类类型。返回的容器为 `ContainerType_Vector_ULINT` 类型，包含 ETcVnCodeType [▶ 165] 枚举的值。返回哪些编码类型取决于所用模型的识别范围。

eModelType

ETcVnCodeDetectionModel [▶ 164] 类型的参数 `eModelType` 指定了编码识别的神经网络模型。模型的选择基于两个维度：识别范围（支持哪些编码类型）和输入分辨率（精度与速度）。

模型类型	识别范围

TCVN_CDM_DETECT_1_x	带分类的一维条码和二维码：
	• TCVN_CT_1D
	• TCVN_CT_2D
	  ◦ TCVN_CT_QR
	  ◦ TCVN_CT_DATAMATRIX
	  ◦ TCVN_CT_DOTCODE

TCVN_CDM_DETECT_2_x	仅一维条码（TCVN_CT_1D）
TCVN_CDM_DETECT_3_x	仅二维码（TCVN_CT_2D）

以上每种模型类型均提供三种分辨率级别。后缀（_1、_2、_3）定义了图像在内部缩放的目标尺寸：

- ……_1：小图像尺寸（处理速度最快，适用于编码较大或性能要求较高的应用）
- ……_2：中等图像尺寸（速度与精度的平衡比。推荐用于大多数应用）
- ……_3：大图像尺寸（分辨率最高，对较小或较远的编码识别效果最好，处理时间最长）

模型的选择会影响检测速度和精度。识别范围较小的模型（例如仅支持一维条码或二维码的模型）具有更高的性能。

nCodeType

参数 `nCodeType` 可用作过滤器，用于将检测范围限制在特定的编码类型（ETcVnCodeType [▶ 165]）。要检测模型所支持的所有编码，可以使用值 `TCVN_CT_ANY`。或者，可以使用 `OR` 链接将特定的类型组合起来（例如 `TCVN_CT_QR_CODE OR TCVN_CT_DATAMATRIX`）。请注意，该过滤器仅适用于所选模型 `eModelType` 的识别范围。如果过滤器与识别范围不匹配，则返回 `0x70E HRESULT`。

应用

执行的前提是，使用功能块 FB_VN_InitializeFunction [▶ 1584] 对要使用的模型进行一次性初始化。

以下程序段演示了如何调用 `F_VN_DetectCodesNeuralNetwork` 函数，以同时检测一维条码和二维码：

```
hr := F_VN_DetectCodesNeuralNetwork(
        ipSrcImage      := ipImageIn,
        ipCodeRegions   := ipCodeRegions,
        ipCodeTypes     := ipCodeTypeResult,
        eModelType      := TCVN_CDM_DETECT_1_2,
        nCodeType       := TCVN_CT_ANY,
        hrPrev          := hr);
```

然后可以将检测到的区域设置为 ROI [▶ 150]，并使用合适的读码功能进行解码。为确保解码的可靠性，建议提前使用 F_VN_ResizeRegion_TcVnRectangle_DINT [▶ 1155] 对找到的区域进行扩大处理。这样可以确保将解码所需的没有噪声干扰区包含在图像部分中。

性能说明

模型的执行可从 CPU 并行化中显著受益。因此，建议在隔离的 CPU 核上执行 PLC 代码，并将其分配给专门的工作任务 [▶ 66]。

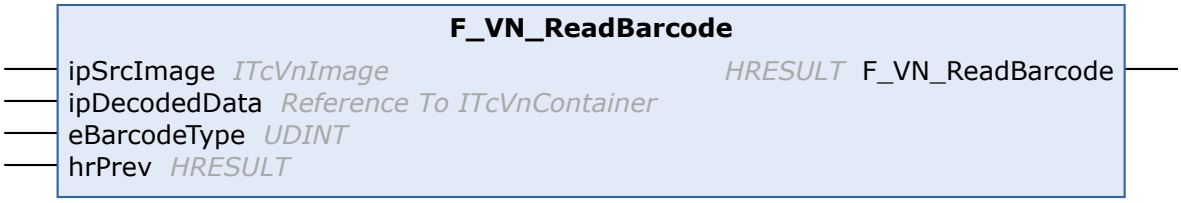
示例

- 编码检测 [▶ 2731]
- 读码过程中的结果评估 [▶ 2745]

所需的许可证

TC3 Vision Code Reading

6.1.4.8.4 F_VN_ReadBarcode



检测并解释所提供图像中的一维条码。 可由 Watchdog（看门狗）取消。

语法

定义：

```
FUNCTION F_VN_ReadBarcode : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDecodedData : Reference To ITcVnContainer;
    eBarcodeType  : UDINT;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 元素，1 通道或 3 通道（3 通道输入应为 RGB 并内部转换为灰色））
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_Vector_String_SINT）。
eBarcodeType	UDINT	要搜索的条形码类型（ETcVnBarcodeType）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

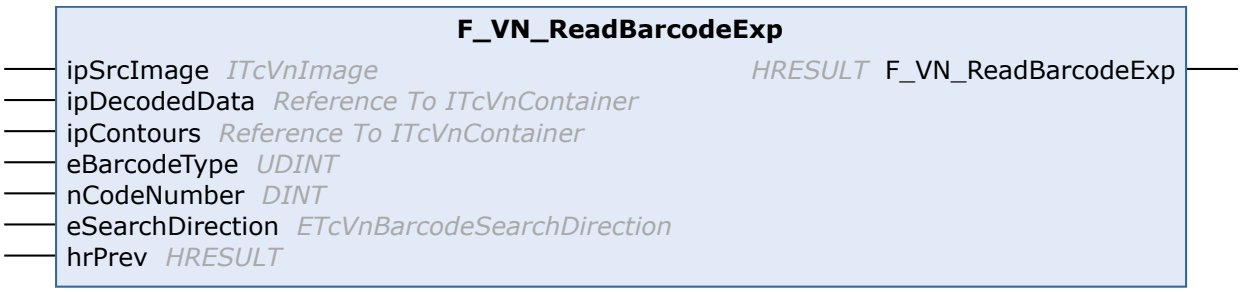
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Code Reading

6.1.4.8.5 F_VN_ReadBarcodeExp



检测并解释所提供图像中的一维条码。 可由 Watchdog（看门狗）取消。

语法

定义：

```
FUNCTION F_VN_ReadBarcodeExp : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
```

```
    ipDecodedData    : Reference To ITcVnContainer;
    ipContours       : Reference To ITcVnContainer;
    eBarcodeType     : UDINT;
    nCodeNumber      : DINT;
    eSearchDirection : ETcVnBarcodeSearchDirection;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 元素，1 通道或 3 通道（3 通道输入应为 RGB 并内部转换为灰色））
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_Vector_String_SINT）。
ipContours	引用 ITcVnContainer [▶ 365]	以轮廓形式返回编码位置（可选项，如果不需要，则设为 0；ContainerType_Vector_Vector_TcVnPoint2_DINT）
eBarcodeType	UDINT	要搜索的条形码类型（ETcVnBarcodeType）
nCodeNumber	DINT	在提供的图像中应检测到的一维码数量（目前仅支持 1 个）。
eSearchDirection	ETcVnBarcodeSearchDirection [▶ 158]	条形码搜索方向。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

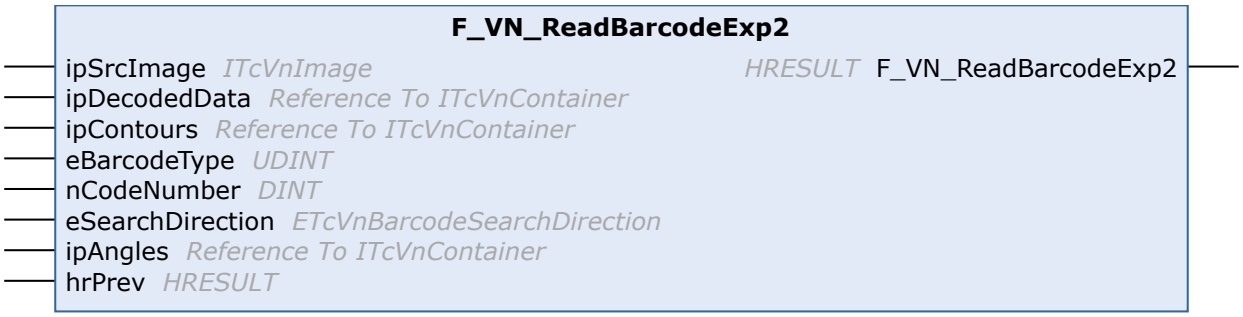
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Code Reading

6.1.4.8.6 F_VN_ReadBarcodeExp2



检测并解释所提供图像中的一维码。 可由看门狗取消。

语法

定义：

```
FUNCTION F_VN_ReadBarcodeExp2 : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
    ipDecodedData    : Reference To ITcVnContainer;
    ipContours       : Reference To ITcVnContainer;
    eBarcodeType     : UDINT;
    nCodeNumber      : DINT;
    eSearchDirection : ETcVnBarcodeSearchDirection;
    ipAngles         : Reference To ITcVnContainer;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 元素，1 通道或 3 通道（3 通道输入应为 RGB 并内部转换为灰色））
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_Vector_String_SINT）。
ipContours	引用 ITcVnContainer [▶ 365]	以轮廓形式返回编码位置（可选项，如果不需要，则设为 0；ContainerType_Vector_Vector_TcVnPoint2_DINT）
eBarcodeType	UDINT	要搜索的条形码类型（ETcVnBarcodeType）
nCodeNumber	DINT	在提供的图像中应检测到的一维码数量（目前仅支持 1 个）。
eSearchDirection	ETcVnBarcodeSearchDirection [▶ 158]	条形码搜索方向。
ipAngles	引用 ITcVnContainer [▶ 365]	返回顺时针编码旋转角度（可选项，如果不需要，则设为 0；ContainerType_Vector_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_ReadBarcodeExp` 是 F_VN_ReadBarcode [▶ 842] 的专家变体。它包含用于返回编码位置和设置搜索方向的附加参数。

参数

输入图像

输入图像`ipSrcImage`必须为元素类型`USINT`（8 位）的 1 通道或 3 通道图像。3 通道图像将被解释为 RGB 并在内部转换为灰度图像。

阅读数据

从代码中读取的数据在容器`ipDecodedData`中以字符串形式返回。容器类型是`ContainerType_Vector_String_SINT`。

条形码类型

参数`eBarcodeType`定义了要在图像中搜索的条形码类型。枚举ETcVnBarcodeType [▶ 158]中支持所有条码类型。可以选择一个以上的类型。建议将选择限制在最终应用中实际出现的类型，以改善运行时间。

```
eBarcodeType := TCVN_BT_EAN13 + TCVN_BT_CODE128;
```

使用`TCVN_BT_ANY`搜索所有可用的条形码类型。

编码的位置

参数 `ipContour` 会返回作为轮廓找到的编码位置。

编码数量

参数 `nCodeNumber` 定义了要搜索的编码数量。

不支持搜索多个条形码

目前不支持在一个图像中搜索多个条形码。因此，必须设为参数 nCodeNumber := 1。

搜索方向

可通过 `eSearchDirection` 指定搜索方向：

- 通过 `TCVN_BSD_ANY`，首先在水平方向上进行搜索，然后在垂直方向上进行搜索。
- 通过 `TCVN_BSD_HORIZONTAL`，仅在水平方向上进行搜索。
- 通过 `TCVN_BSD_VERTICAL`，仅在垂直方向上进行搜索。

旋转角度

参数 `ipAngles` 会返回以顺时针方向找到的代码的旋转角度（单位：90°）。容器类型为 `ContainerType_Vector_REAL`。该参数为可选参数，如果不需要，可设置为 0。

样本

- EAN-13 条形码读取 [▶ 2734]
- 读码过程中的结果评估 [▶ 2745]

相关函数

- F_VN_GradeBarcode [▶ 821]
- F_VN_GradeDataMatrixCode [▶ 827]
- F_VN_GradeQRCode [▶ 832]
- F_VN_ReadBarcode [▶ 842]
- F_VN_ReadBarcodeRoi [▶ 845]
- F_VN_ReadPharmaCode [▶ 861]
- F_VN_ReadDataMatrixCode [▶ 850]
- F_VN_ReadDataMatrixCodeRoi [▶ 855]
- F_VN_ReadQRCode [▶ 864]
- F_VN_ReadDotCodeRoi [▶ 858]

所需的许可证

TC3 Vision Code Reading

6.1.4.8.7 F_VN_ReadBarcodeRoi



检测并解释所提供图像中的一维条码。为获得最佳性能，编码应该水平对齐并位于图像中心。

语法

定义：

```
FUNCTION F_VN_ReadBarcodeRoi : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDecodedData : Reference To ITcVnContainer;
```

```
    eBarcodeType   : ETcVnBarcodeType;
    hrPrev         : HRESULT;
END_VAR
VAR_OUTPUT
    fAngleDeg      : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道）
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码数据（ContainerType_String_SINT）
eBarcodeType	ETcVnBarcodeType [▶ 158]	条形码类型（支持：CODE39、CODE128、EAN8、EAN13、ITF、UPCA、UPCE）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输出

名称	类型	描述
fAngleDeg	REAL	以度为单位返回大致的顺时针旋转角度

返回值

HRESULT [▶ 135]

更多信息

`F_VN_ReadBarcodeRoi` 函数经过优化，可在规定图像区域（ROI）内检测单个条形码。

算法

为此，该函数使用了一种定制算法，要求编码在传输图像中居中。该函数首先会搜索图像中水平排列的编码，然后搜索垂直排列的编码。

参数

输入图像

`ipSrcImage` 输入图像必须是元素类型为 `USINT`（8 位）的单通道图像。编码必须在图像中心水平或垂直对齐。

读取数据

从编码中读取的数据以字符串形式返回 `ipDecodedData` 容器中。容器类型为 `ContainerType_String_SINT`。成功找到编码并解码后，便可以使用 F_VN_ExportContainer_String [▶ 756] 函数将内容导出为字符串。

```
hr := F_VN_ExportContainer_String(ipDecodedData, sText, nMaxLength, hr);
```

条形码类型

`eBarcodeType` 参数定义了要读取的条形码类型。每次函数调用只能传输一种条形码类型。支持 ETcVnBarcodeType [▶ 158] 枚举中的以下条形码类型：

- `TCVN_BT_CODE39`
- `TCVN_BT_CODE128`
- `TCVN_BT_EAN8`
- `TCVN_BT_EAN13`
- `TCVN_BT_ITF`

- TCVN_BT_UPCA
- TCVN_BT_UPCE

旋转角度 (返回值)

fAngleDeg 返回值以 90° 为单位返回顺时针旋转角度。

专家参数

专家版 F_VN_ReadBarcodeRoiExp [▶ 847] 包含附加参数。

样本

- EAN-13 条形码读取 [▶ 2734]
- 读码过程中的结果评估 [▶ 2745]

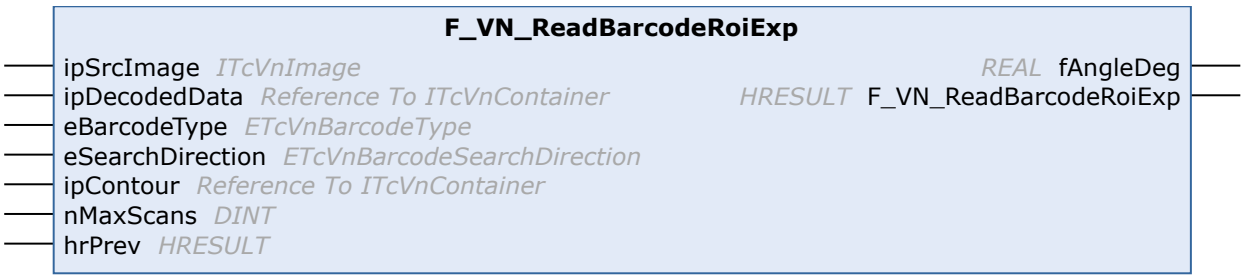
相关函数

- F_VN_GradeBarcode [▶ 821]
- F_VN_GradeDataMatrixCode [▶ 827]
- F_VN_GradeQRCode [▶ 832]
- F_VN_ReadBarcode [▶ 842]
- F_VN_ReadBarcodeRoi [▶ 845]
- F_VN_ReadPharmaCode [▶ 861]
- F_VN_ReadDataMatrixCode [▶ 850]
- F_VN_ReadDataMatrixCodeRoi [▶ 855]
- F_VN_ReadQRCode [▶ 864]
- F_VN_ReadDotCodeRoi [▶ 858]

所需的许可证

TC3 Vision Code Reading

6.1.4.8.8 F_VN_ReadBarcodeRoiExp



检测并解释所提供图像中的一维条码。为获得最佳性能，编码应该水平对齐并位于图像中心。（专家函数）

语法

定义：

```
FUNCTION F_VN_ReadBarcodeRoiExp : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
    ipDecodedData    : Reference To ITcVnContainer;
    eBarcodeType     : ETcVnBarcodeType;
    eSearchDirection : ETcVnBarcodeSearchDirection;
    ipContour        : Reference To ITcVnContainer;
    nMaxScans        : DINT;
    hrPrev           : HRESULT;
END_VAR
```

```
VAR_OUTPUT
    fAngleDeg        : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道）
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码数据（ContainerType_String_SINT）
eBarcodeType	ETcVnBarcodeType [▶ 158]	条形码类型（支持：CODE39、CODE128、EAN8、EAN13、ITF、UPCA、UPCE）
eSearchDirection	ETcVnBarcodeSearchDirection [▶ 158]	条形码搜索方向（BSD_ANY 首先尝试水平方向，然后是垂直方向）
ipContour	引用 ITcVnContainer [▶ 365]	返回 4 个码角点的大致位置，每个位置均包含了静区所占的区域。宽度是准确的，但出于性能原因，不测量实际高度。相反，将其静态设置为宽度的 10%，并以已解码的扫描线为中心。（可选项，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_DINT）
nMaxScans	DINT	为查找条形码而评估的扫描线的最大数量。扫描始终从图像中心开始，然后以（ipSrcImage 高度 / nMaxScans）为步长，交替向上和向下进行，距离逐渐增大。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输出

名称	类型	描述
fAngleDeg	REAL	以度为单位返回大致的顺时针旋转角度

返回值

HRESULT [▶ 135]

更多信息

`F_VN_ReadBarcodeRoiExp` 函数是 F_VN_ReadBarcodeRoi [▶ 845] 的专家版本。它包含额外的参数。

参数

输入图像

`ipSrcImage` 输入图像必须是元素类型为 `USINT`（8 位）的单通道图像。编码必须在图像中心水平或垂直对齐。

读取数据

从编码中读取的数据以字符串形式返回 `ipDecodedData` 容器中。容器类型为 `ContainerType_String_SINT`。成功找到编码并解码后，便可以使用 F_VN_ExportContainer_String [▶ 756] 函数将内容导出为字符串。

```
hr := F_VN_ExportContainer_String(ipDecodedData, sText, nMaxLength, hr);
```

条形码类型

`eBarcodeType` 参数定义了要读取的条形码类型。每次函数调用只能传输一种条形码类型。支持 ETcVnBarcodeType [▶ 158] 枚举中的以下条形码类型：

- `TCVN_BT_CODE39`

- TCVN_BT_CODE128
- TCVN_BT_EAN8
- TCVN_BT_EAN13
- TCVN_BT_ITF
- TCVN_BT_UPCA
- TCVN_BT_UPCE

搜索方向

可通过 eSearchDirection 指定搜索方向：

- 通过 TCVN_BSD_ANY，首先在水平方向上进行搜索，然后在垂直方向上进行搜索。
- 通过 TCVN_BSD_HORIZONTAL，仅在水平方向上进行搜索。
- 通过 TCVN_BSD_VERTICAL，仅在垂直方向上进行搜索。

编码的位置

ipContour 参数会将编码角点（包括没有噪声干扰区）作为类型 ContainerType_Vector_TcVnPoint2_DINT 的四个轮廓点返回。该位置基于可解码的第一条扫描线。将高度静态设置为宽度的 10%，并以扫描线为中心。例如，可以使用 F_VN_DrawContours [▶ 1012] 直接绘制等高线，也可以使用 Contour Analysis（等高线分析）函数进行进一步分析。该参数为可选参数，如果不需要，可设置为 0。

扫描线的最大数量

为查找条形码而评估的扫描线的最大数量由 nMaxScans 定义。扫描始终从图像中心开始，然后随着距离的增加，以 ipSrcImage height / nMaxScans 为步长交替向上和向下移动。

旋转角度（返回值）

fAngleDeg 返回值以 90° 为单位返回顺时针旋转角度。

样本

- EAN-13 条形码读取 [▶ 2734]
- 读码过程中的结果评估 [▶ 2745]

相关函数

- F_VN_GradeBarcode [▶ 821]
- F_VN_GradeDataMatrixCode [▶ 827]
- F_VN_GradeQRCode [▶ 832]
- F_VN_ReadBarcode [▶ 842]
- F_VN_ReadBarcodeRoi [▶ 845]
- F_VN_ReadPharmaCode [▶ 861]
- F_VN_ReadDataMatrixCode [▶ 850]
- F_VN_ReadDataMatrixCodeRoi [▶ 855]
- F_VN_ReadQRCode [▶ 864]
- F_VN_ReadDotCodeRoi [▶ 858]

所需的许可证

TC3 Vision Code Reading

6.1.4.8.9 F_VN_ReadDataMatrixCode



Detect and interpret a data matrix code (ECC200) within the provided image.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_ReadDataMatrixCode : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDecodedData : Reference To ITcVnContainer;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 元素，1 或 3 通道）
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_Vector_String_SINT）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

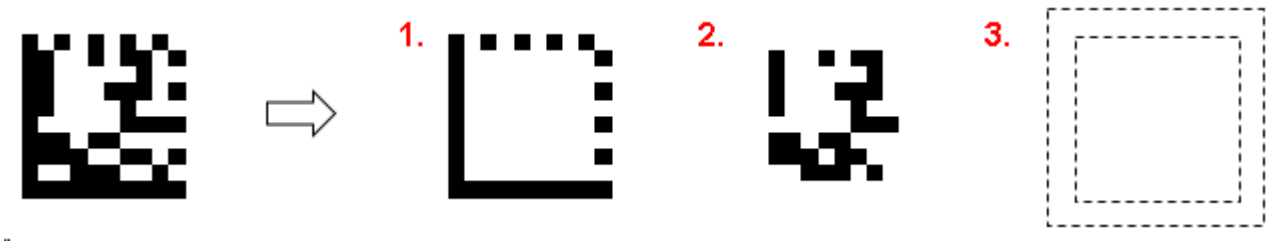
更多信息

符合 IEC 16022 标准的数据矩阵条形码是一种二维码，便于以最小所需空间存储大量数据。因此，它通常用于组件和产品标记。

函数 F_VN_ReadDataMatrixCode 只能识别和解码图像中的 ECC 200 代码类型。这以每图像一个代码为基础。如果要在一个图像中识别和解码多个数据矩阵代码，必须使用函数 F_VN_ReadDataMatrixCodeExp [▶ 852]。

数据矩阵代码结构

数据矩阵代码由矩形或圆点组成一个正方形或长方形符号，而后者又可分为三个区域：



1. 寻像模板
   实线定位器 (L) 用于定位和对齐代码，而模板线定位器 (P) 则用于确定单元大小和排列。
2. 数据区
   数据区由寻像模板包围。

3. 静区
   静区是代码背景色中寻像模板周围至少 1.5 个单元宽的区域。

寻像模板或静音区中的缺陷/损坏可能会导致无法检测到代码。

错误纠正

数据矩阵代码 ECC 200（ECC = 错误纠正代码）是普遍接受的版本，因为它的纠正效果更好。除了实际数据外，它们还根据里德所罗门代码进行冗余编码。该版本的数据矩阵代码可通过该函数解码。如有损坏，可执行错误纠正。

代码搜索

函数 F_VN_ReadDataMatrixCode 从图像中心线开始搜索图像中的数据矩阵代码。

从容器中读取数据

如果成功找到并解码了代码，代码内容将在容器`ipDecodedData`中返回，并可以使用函数 F_VN_ExportSubContainer_String [▶ 759]导出为一个字符串。

```
hr := F_VN_ExportSubContainer_String( ipDecodedData, 0, sText, nMaxLength, hr);
```

HRESULT

与所有 TwinCAT Vision API 函数一样，该函数也返回一个HRESULT [▶ 135]，以表明执行是否成功。在成功的情况下，在随后对结果的处理中，可以区分以下成功代码。为了区分这些情况，`HRESULT`变量可以直接与`S_OK`或`S_FALSE`进行比较。

代码	名称	描述
16#000	S_OK	该功能成功执行。 在图像中发现了指定数量的代码。
16#001	S_FALSE	该功能成功执行。 发现的代码比图片中显示的少。
16#256	S_WATCHDOGTIMEOUT	该功能被看门狗 [▶ 140]中止。

示例

- 数据矩阵码读取 [▶ 2737]
- 读码过程中的结果评估 [▶ 2745]

相关函数

- F_VN_ReadDataMatrixCodeExp [▶ 852]
- F_VN_ReadBarcode [▶ 842] 或 F_VN_ReadBarcodeExp [▶ 842]
- F_VN_ReadPharmaCode [▶ 861] 或 F_VN_ReadPharmaCodeExp [▶ 861]
- F_VN_ReadQRCode [▶ 864] 或 F_VN_ReadQRCodeExp [▶ 866]

所需的许可证

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.8.10 F_VN_ReadDataMatrixCodeExp



Detect and interpret a data matrix code (ECC200) within the provided image.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_ReadDataMatrixCodeExp : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipDecodedData   : Reference To ITcVnContainer;
    ipContours      : Reference To ITcVnContainer;
    nCodeNumber     : DINT;
    eSearchStrategy : UDINT;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 元素，1 或 3 通道）
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_Vector_String_SINT）。
ipContours	引用 ITcVnContainer [▶ 365]	以轮廓形式返回编码位置（可选项，如果不需要，则设为 0；ContainerType_Vector_Vector_TcVnPoint2_DINT）
nCodeNumber	DINT	在提供的图像中应检测到的数据矩阵编码数量。（如果设置为 -1，则会尝试检测所有数据矩阵编码）。
eSearchStrategy	UDINT	使用的搜索策略（ETcVn2dCodeSearchStrategy）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_ReadDataMatrixCodeExp 对应于函数F_VN_ReadDataMatrixCode [▶ 850]，由以下参数扩展。

ipContours

参数ipContour返回找到的代码位置，作为轮廓。

nCodeNumber

参数nCodeNumber定义了需要搜索多少个代码。在-1，搜索继续进行，直到在图像中没有进一步发现代码或看门狗中止功能。

电子搜索策略

参数eSearchStrategy定义了代码搜索的搜索策略。标准选项TCVN_CSS_DEFAULT导致搜索策略TCVN_CSS_FIRST_NOT_FLIPPED。关于搜索策略的描述，见代码阅读搜索策略。

搜索策略

目前，这个功能只支持对镜像的搜索策略的设置。如果在原始图像中没有找到所有预期的代码，反转总是自动进行。

样本

- 数据矩阵码读取 [▶ 2737]
- 读码过程中的结果评估 [▶ 2745]

相关函数

- F_VN_ReadDataMatrixCode [▶ 850]
- F_VN_ReadBarcode [▶ 842]或F_VN_ReadBarcodeExp [▶ 842]
- F_VN_ReadPharmaCode [▶ 861]或F_VN_ReadPharmaCodeExp [▶ 861]
- F_VN_ReadQRCode [▶ 864]或 F_VN_ReadQRCodeExp [▶ 866]

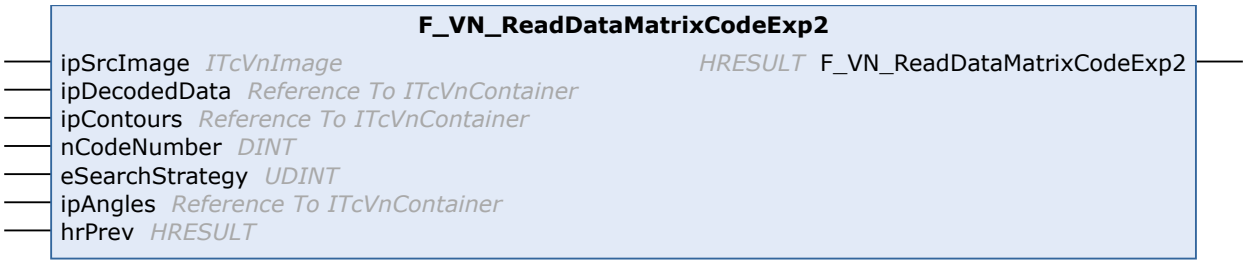
所需的许可证

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.8.11 F_VN_ReadDataMatrixCodeExp2



Detect and interpret a data matrix code (ECC200) within the provided image.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_ReadDataMatrixCodeExp2 : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipDecodedData   : Reference To ITcVnContainer;
    ipContours      : Reference To ITcVnContainer;
    nCodeNumber     : DINT;
    eSearchStrategy : UDINT;
    ipAngles        : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 元素，1 或 3 通道）
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_Vector_String_SINT）。

名称	类型	描述
ipContours	引用 ITcVnContainer [▶ 365]	以轮廓形式返回编码位置（可选项，如果不需要，则设为 0；ContainerType_Vector_Vector_TcVnPoint2_DINT）
nCodeNumber	DINT	在提供的图像中应检测到的数据矩阵编码数量。（如果设置为 -1，则会尝试检测所有数据矩阵编码）。
eSearchStrategy	UDINT	使用的搜索策略（ETcVn2dCodeSearchStrategy）
ipAngles	引用 ITcVnContainer [▶ 365]	返回顺时针编码旋转角度（可选项，如果不需要，则设为 0；ContainerType_Vector_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_ReadDataMatrixCodeExp 与通过以下参数扩展的功能 F_VN_ReadDataMatrixCode [▶ 850] 相对应。

ipContours

参数 `ipContour` 会返回作为轮廓找到的代码位置。

nCodeNumber

参数 `nCodeNumber` 定义了要搜索的编码数量。-1 时，搜索继续进行，直到在图像中找不到更多编码或 Watchdog（看门狗）中止该函数。

eSearchStrategy

参数 `eSearchStrategy` 定义了代码搜索的搜索策略。标准选项 `TCVN_CSS_DEFAULT` 下的搜索策略为 `TCVN_CSS_FIRST_NOT_FLIPPED`。关于搜索策略的说明，请参见 Code Reading（读码）搜索策略。

搜索策略

目前，该函数仅支持镜像搜索策略的设置。如果在原始图像中没有找到所有预期的编码，反转总是自动进行。

旋转角度

参数 `ipAngles` 会返回代码顺时针旋转角度（单位：度）。容器类型为 `ContainerType_Vector_REAL`。该参数为可选参数，如果不需要，可设置为 0。

示例

- 数据矩阵码读取 [▶ 2737]
- 读码过程中的结果评估 [▶ 2745]

相关函数

- F_VN_ReadDataMatrixCode [▶ 850]
- F_VN_ReadBarcode [▶ 842] 或 F_VN_ReadBarcodeExp [▶ 842]
- F_VN_ReadPharmaCode [▶ 861] 或 F_VN_ReadPharmaCodeExp [▶ 861]
- F_VN_ReadQRCode [▶ 864] 或 F_VN_ReadQRCodeExp [▶ 866]

所需的许可证

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.8.12 F_VN_ReadDataMatrixCodeRoi



在提供的图像中检测并解释数据矩阵编码（ECC200）。编码应该位于图像中心。
可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_ReadDataMatrixCodeRoi : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDecodedData : Reference To ITcVnContainer;
    fModuleWidth  : REAL;
    hrPrev        : HRESULT;
END_VAR
VAR_OUTPUT
    fAngleDeg     : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 元素，1 通道）
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_String_SINT）
fModuleWidth	REAL	图像中编码的最小模块宽度（单位：像素）。必须至少为 3，最好在 5 - 8 之间。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输出

名称	类型	描述
fAngleDeg	REAL	以度为单位返回顺时针旋转角度

返回值

HRESULT [▶ 135]

更多信息

`F_VN_ReadDataMatrixCodeRoi` 函数经过优化，可在规定图像区域内检测单个 Data Matrix（数据矩阵）编码 (ECC200)。为此，该函数使用了一种定制算法，要求编码在传输图像中居中。

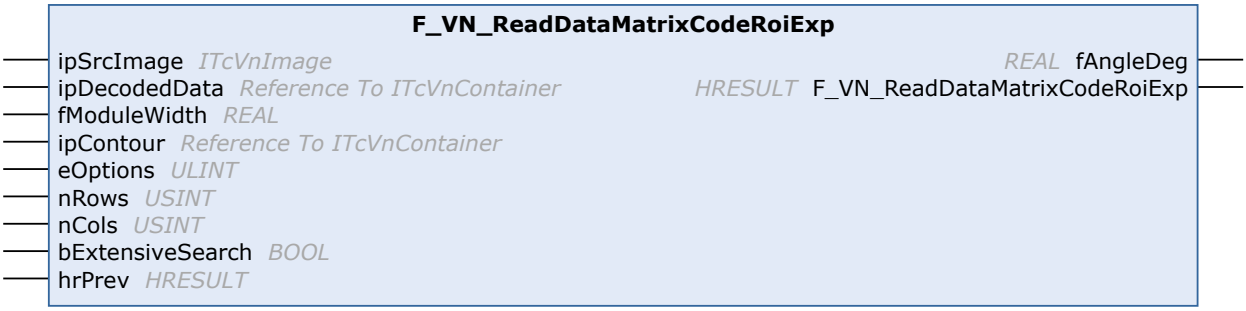
fModuleWidth

`fModuleWidth` 参数定义了要读取的 Data Matrix（数据矩阵）编码的最小预期模块宽度（单位：像素）。一个模块对应编码中的一个黑色或白色方格。如果图像中的模块尺寸与此相差较大，可能会导致计算时间延长或无结果。

所需的许可证

TC3 Vision Code Reading

6.1.4.8.13 F_VN_ReadDataMatrixCodeRoiExp



在提供的图像中检测并解释数据矩阵编码（ECC200）。编码应该位于图像中心。（专家函数）可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_ReadDataMatrixCodeRoiExp : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
    ipDecodedData    : Reference To ITcVnContainer;
    fModuleWidth     : REAL;
    ipContour        : Reference To ITcVnContainer;
    eOptions         : ULINT;
    nRows            : USINT;
    nCols            : USINT;
    bExtensiveSearch : BOOL;
    hrPrev           : HRESULT;
END_VAR
VAR_OUTPUT
    fAngleDeg        : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 元素，1 通道）
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_String_SINT）
fModuleWidth	REAL	图像中编码的最小模块宽度（单位：像素）。必须至少为 3，最好在 5 - 8 之间。
ipContour	引用 ITcVnContainer [▶ 365]	返回检测到的编码的轮廓（ContainerType_Vector_TcVnPoint2_DINT；可选项，如果不需要，则设为 0）
eOptions	ULINT	指定一个或多个 ETcVnDataMatrixOptions 的组合
nRows	USINT	指定编码模块行数，包括查找器和时钟模式，不包括安静区（8 - 144；设为 0，可搜索任何大小）
nCols	USINT	指定编码模块列数，包括查找器和时钟模式，不包括安静区（10 - 144；设为 0，可搜索任何大小）
bExtensiveSearch	BOOL	如果为 true，则在指定选项失败时尝试不同的方法
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输出

名称	类型	描述
fAngleDeg	REAL	以度为单位返回顺时针旋转角度

返回值

HRESULT [▶ 135]

更多信息

`F_VN_ReadDataMatrixCodeRoiExp` 函数是 F_VN_ReadDataMatrixCodeRoi [▶ 855] 的专家变体，包含附加参数。

ipContour

参数 `ipContour` 会将编码角点作为类型 `ContainerType_Vector_TcVnPoint2_DINT` 的四个轮廓点返回。角点位于外部模块的边缘。例如，可以使用 F_VN_DrawContours [▶ 1012] 直接绘制等高线，也可以使用 Contour Analysis（等高线分析）函数进行进一步分析。该参数为可选参数，如果不需要，可设置为 0。

eOptions

`eOptions` 参数定义了要使用的检测策略。有三组相互排斥的选项，每组只能选择一个选项：

1. 图像预处理：
   - `TCVN_DMO_BINARY_INPUT`
   - `TCVN_DMO_GLOBAL_THRESHOLD`
2. 网格检测：
   - `TCVN_DMO_BINARY_GRID`
   - `TCVN_DMO_STATIC_GRID`
3. 编码颜色模式：
   - `TCVN_DMO_REGULAR_COLOR`
   - `TCVN_DMO_INVERTED_COLOR`

每组都可以单独配置，例如 `TCVN_DMO_GLOBAL_THRESHOLD OR TCVN_DMO_STATIC_GRID OR TCVN_DMO_REGULAR_COLOR`。`TCVN_DMO_BINARY_GRID OR TCVN_DMO_STATIC_GRID` 等无效组合会导致错误 `ADS_E_INVALIDPARM`。

默认设置 `TCVN_DMO_DEFAULT` 使用自适应局部阈值、动态网格（基于灰度图像）并检测两种颜色模式。为优化计算时间，建议始终明确指定颜色模式。

有关识别策略的说明，请参见 ETcVnDataMatrixOptions [▶ 185]。

nRows 和 nCols

参数 `nRows` 和 `nCols` 固定了垂直（高度）或水平（宽度）方向上的预期模块数，其依据是 L 图案的角点位于左下方的非旋转编码。Data Matrix（数据矩阵）编码的定位图案和时钟图案都必须包含在计数中。建议您务必设置这些值，以减少所需的最长计算时间。

bExtensiveSearch

`bExtensiveSearch` 参数可激活高级搜索例行程序，在初始分析失败时自动应用其他解码策略。

所需的许可证

TC3 Vision Code Reading

6.1.4.8.14 F_VN_ReadDotCodeRoi



检测并解释所提供图像中的 DotCode (点码）。编码应该位于图像中心，并水平或垂直对齐。
可以使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_ReadDotCodeRoi : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDecodedData : Reference To ITcVnContainer;
    fModuleWidth  : REAL;
    hrPrev        : HRESULT;
END_VAR
VAR_OUTPUT
    fAngleDeg     : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 元素，1 通道）
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_String_SINT）
fModuleWidth	REAL	图像中编码的平均点直径（以像素为单位）。必须至少为 3，最好在 5 - 8 之间。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输出

名称	类型	描述
fAngleDeg	REAL	以度为单位返回顺时针旋转角度

返回值

HRESULT [▶ 135]

更多信息

`F_VN_ReadDotCodeRoi` 函数在指定的感兴趣区域（ROI）内搜索单个 DotCode（点码） 并对其进行解码。这要求编码在所提供的图像中居中。

fModuleWidth

`fModuleWidth` 参数定义了单个 DotCode（点码） 模块（点）的预期平均直径（以像素为单位）。模块大小至少须为 3 个像素，建议使用 5 到 8 个像素以获得最佳且可靠的结果。与参数设置值存在显著偏差可能会导致处理时间增加或无结果。

旋转角度 (返回值)

返回值 `fAngleDeg` 返回顺时针旋转角度。

示例

- DotCode 读取 [▶ 2743]

- 读码过程中的结果评估 [▶ 2745]

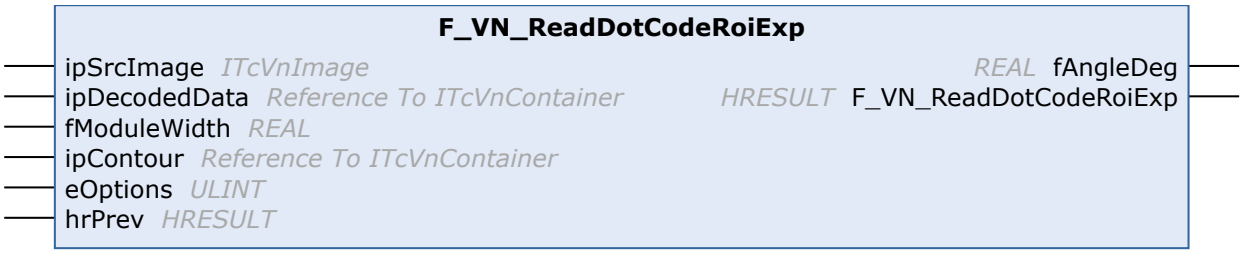
相关函数

- F_VN_GradeBarcode [▶ 821]
- F_VN_GradeDataMatrixCode [▶ 827]
- F_VN_GradeQRCode [▶ 832]
- F_VN_ReadBarcode [▶ 842]
- F_VN_ReadBarcodeRoi [▶ 845]
- F_VN_ReadPharmaCode [▶ 861]
- F_VN_ReadDataMatrixCode [▶ 850]
- F_VN_ReadDataMatrixCodeRoi [▶ 855]
- F_VN_ReadQRCode [▶ 864]
- F_VN_ReadDotCodeRoi [▶ 858]

所需的许可证

TC3 Vision Code Reading

6.1.4.8.15 F_VN_ReadDotCodeRoiExp



检测并解释所提供图像中的 DotCode (点码）。编码应在图像中心，并水平或垂直对齐。（专家函数）可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_ReadDotCodeRoiExp : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDecodedData : Reference To ITcVnContainer;
    fModuleWidth  : REAL;
    ipContour     : Reference To ITcVnContainer;
    eOptions      : ULINT;
    hrPrev        : HRESULT;
END_VAR
VAR_OUTPUT
    fAngleDeg     : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 元素，1 通道）
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_String_SINT）
fModuleWidth	REAL	图像中编码的平均点直径（以像素为单位）。必须至少为 3，最好在 5 - 8 之间。
ipContour	引用 ITcVnContainer [▶ 365]	返回检测到的编码的轮廓（ContainerType_Vector_TcVnPoint2_DINT；可选项，如果不需要，则设为 0）

名称	类型	描述
eOptions	ULINT	指定一个或多个 ETcVnDotCodeOptions 的组合
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输出

名称	类型	描述
fAngleDeg	REAL	以度为单位返回顺时针旋转角度

返回值

HRESULT [▶ 135]

更多信息

`F_VN_ReadDotCodeRoiExp` 函数是 F_VN_ReadDotCodeRoi [▶ 858] 的专家变体，包含附加参数。

ipContour

参数 `ipContour` 会将编码角点作为类型 `ContainerType_Vector_TcVnPoint2_DINT` 的四个轮廓点返回。这些坐标对应于 DotCode（点码） 网格中理论边角模块的中心，无论相应边角处是否实际印有点。例如，可以使用 F_VN_DrawContours [▶ 1012] 直接绘制等高线，也可以使用 Contour Analysis（等高线分析）函数进行进一步分析。该参数为可选参数，如果不需要，可设置为 0。

eOptions

`eOptions` 参数（ETcVnDotCodeOptions [▶ 188]）定义了识别与预处理策略。一些选项可以组合使用（使用按位 `OR` 运算）。可提供下列设置：

- `TCVN_DCO_DEFAULT`：预期为灰度图像，并使用标准算法。
- `TCVN_DCO_BINARY_INPUT`：预期为已二值化的图像（像素值仅为 0 或 255）。这会停用内部预处理。
- `TCVN_DCO_DETECTBLOBS`：修改算法，并使用Blob检测来查找各个点。（不能与 `BINARY_INPUT` 或 `FILTERCONTOURS` 组合使用）。
- `TCVN_DCO_FILTERCONTOURS`：激活一个预处理步骤，该步骤会滤除图像边缘的轮廓以及尺寸偏差较大（相对于 `fModuleWidth`）的轮廓。
- `TCVN_DCO_REGULAR_COLOR`：将搜索范围限制为仅检测常规编码（浅色背景上的深色点）。
- `TCVN_DCO_INVERTED_COLOR`：将搜索范围限制为仅检测反向编码（深色背景上的浅色点）。

示例

- DotCode 读取 [▶ 2743]
- 读码过程中的结果评估 [▶ 2745]

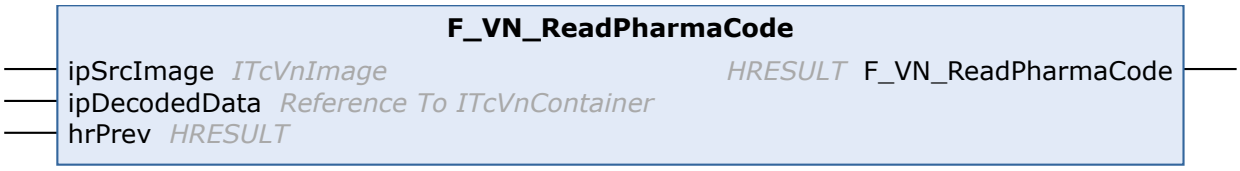
相关函数

- F_VN_GradeBarcode [▶ 821]
- F_VN_GradeDataMatrixCode [▶ 827]
- F_VN_GradeQRCode [▶ 832]
- F_VN_ReadBarcode [▶ 842]
- F_VN_ReadBarcodeRoi [▶ 845]
- F_VN_ReadPharmaCode [▶ 861]
- F_VN_ReadDataMatrixCode [▶ 850]
- F_VN_ReadDataMatrixCodeRoi [▶ 855]
- F_VN_ReadQRCode [▶ 864]
- F_VN_ReadDotCodeRoi [▶ 858]

所需的许可证

TC3 Vision Code Reading

6.1.4.8.16 F_VN_ReadPharmaCode



检测并解释所提供图像中的pharma码。 可由 Watchdog（看门狗）取消。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_ReadPharmaCode : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDecodedData : Reference To ITcVnContainer;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道（所有元素类型）或带有 TCVN_ET_USINT、TCVN_ET_UINT 或 TCVN_ET_REAL 类型元素的 3 通道（3 通道输入应为 RGB 并在内部转换为灰色））
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_Vector_String_SINT）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

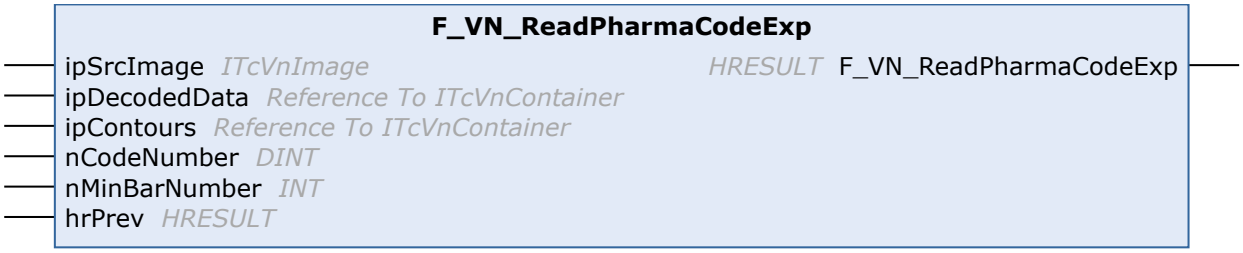
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Code Reading

6.1.4.8.17 F_VN_ReadPharmaCodeExp



Detect and interpret a pharma code within the provided image. Can be canceled by Watchdog.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_ReadPharmaCodeExp : HRESULT
VAR_INPUT
    ipSrcImage     : ITcVnImage;
    ipDecodedData : Reference To ITcVnContainer;
    ipContours     : Reference To ITcVnContainer;
    nCodeNumber    : DINT;
    nMinBarNumber : INT;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道（所有元素类型）或带有 TCVN_ET_USINT、TCVN_ET_UINT 或 TCVN_ET_REAL 类型元素的 3 通道（3 通道输入应为 RGB 并在内部转换为灰色））
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_Vector_String_SINT）。
ipContours	引用 ITcVnContainer [▶ 365]	以轮廓形式返回编码位置（可选项，如果不需要，则设为 0；ContainerType_Vector_Vector_TcVnPoint2_DINT）
nCodeNumber	DINT	在提供的图像中应检测到的药品二进制码数量（目前仅支持 1 个）。
nMinBarNumber	INT	编码必须具有的最少（暗）条数。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

这个专家函数扩展了F_VN_ReadPharmaCode [▶ 861]函数，内容如下：

- 通过参数`ipContours`返回找到的代码的位置
- 指定图像中需要找到的代码数量。一旦在图像中发现`nCodeNumber`或更多的代码，搜索就会终止。如果找到的代码少于`nCodeNumber`，函数返回`S_FALSE`。
- 通过参数`nMinBarNumber`，函数找到的代码必须具有的最小暗条数。

样本

- 药品编码读取 [▶ 2739]
- 读码过程中的结果评估 [▶ 2745]

相关函数

- F_VN_ReadPharmaCode [▶ 861]
- F_VN_ReadBarcode [▶ 842]或F_VN_ReadBarcodeExp
- F_VN_ReadDataMatrixCode [▶ 850]或F_VN_ReadDataMatrixCodeExp [▶ 852]
- F_VN_ReadQRCode [▶ 864]或F_VN_ReadQRCodeExp [▶ 866]

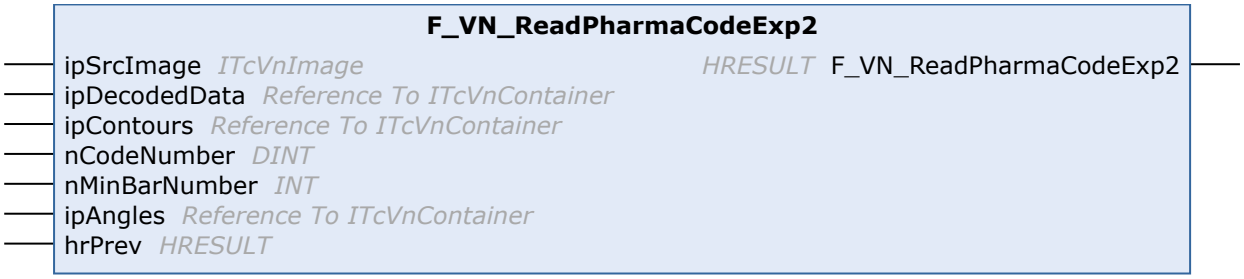
所需的许可证

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.8.18 F_VN_ReadPharmaCodeExp2



Detect and interpret a pharma code within the provided image. Can be canceled by Watchdog. Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_ReadPharmaCodeExp2 : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDecodedData : Reference To ITcVnContainer;
    ipContours    : Reference To ITcVnContainer;
    nCodeNumber   : DINT;
    nMinBarNumber : INT;
    ipAngles      : Reference To ITcVnContainer;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道（所有元素类型）或带有 TCVN_ET_USINT、TCVN_ET_UINT 或 TCVN_ET_REAL 类型元素的 3 通道（3 通道输入应为 RGB 并在内部转换为灰色））
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_Vector_String_SINT）。
ipContours	引用 ITcVnContainer [▶ 365]	以轮廓形式返回编码位置（可选项，如果不需要，则设为 0；ContainerType_Vector_Vector_TcVnPoint2_DINT）
nCodeNumber	DINT	在提供的图像中应检测到的药品二进制码数量（目前仅支持 1 个）。
nMinBarNumber	INT	编码必须具有的最少（暗）条数。
ipAngles	引用 ITcVnContainer [▶ 365]	返回顺时针编码旋转角度（可选项，如果不需要，则设为 0；ContainerType_Vector_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

该 Expert（专家）函数对函数 F_VN_ReadPharmaCode [▶ 861] 进行了如下扩展：

- 通过参数 `ipContours` 返回找到的编码位置
- 指定要在图像中找到的编码数。一旦在图像中找到 `nCodeNumber` 或更多编码，搜索就会终止。如果找到的编码少于 `nCodeNumber`，则该函数会返回 `S_FALSE`。
- 该函数通过参数 `nMinBarNumber` 找到编码所需的最小暗条数。

旋转角度

参数 `ipAngles` 会返回以顺时针方向从 -90° 到 +90° 所找到的代码的大致旋转角度（单位：度）。容器类型为 `ContainerType_Vector_REAL`。该参数为可选参数，如果不需要，可设置为 0。

读取方向的差异

由于药品码不包含起始或终止标识符，因此读取结果会因读取方向不同而不同。因此，无法通过旋转角度来确定编码是否旋转了 180°。为确保编码符合预期方向，必须将图像旋转 180°，然后再次读取编码。然后将两个读取结果与预期值进行比较，以明确确定方向。

示例

- 药品编码读取 [▶ 2739]
- 读码过程中的结果评估 [▶ 2745]

相关函数

- F_VN_ReadPharmaCode [▶ 861]
- F_VN_ReadBarcode [▶ 842] 或 F_VN_ReadBarcodeExp
- F_VN_ReadDataMatrixCode [▶ 850] 或 VN_ReadDataMatrixCodeExp [▶ 852]
- F_VN_ReadQRCode [▶ 864] 或 F_VN_ReadQRCodeExp [▶ 866]

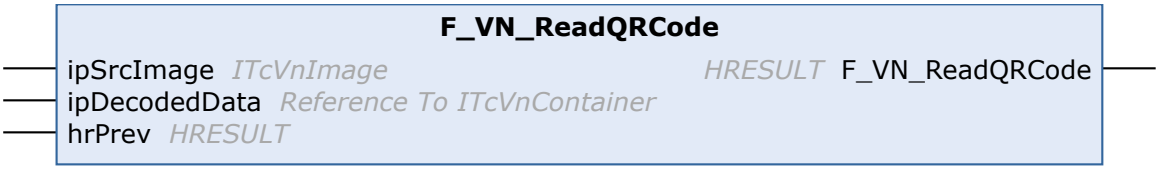
所需的许可证

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.8.19 F_VN_ReadQRCode



Detect and interpret a QR code within the provided image. Can be canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_ReadQRCode : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDecodedData : Reference To ITcVnContainer;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 元素，1 通道或 3 通道（3 通道输入应为 RGB 并内部转换为灰色））
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_Vector_String_SINT）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

根据 IEC18004 标准，QR 码（快速响应码）是一种二维代码，在德国的移动标签领域特别常见。由于可以对汉字/假名字符进行编码，QR 码也已在亚洲的行业中得到确立。

F_VN_ReadQRCode 函数可以识别并解码图像中的这种代码类型。这是以每张图片一个代码为基础。

QR 码结构



1.		位置标记
2.		对齐标记
3.		定时排队
4.		版本信息
5.		数据格式
6.		数据领域
7.		安静区，至少 4 个单元宽

缺陷/损坏可以妨碍检测和解码。

代码搜索

函数 F_VN_ReadQRCode 从左上到右下逐行搜索图像中的 QR 码。

从容器中读取数据

如果成功找到并解码了代码，代码内容将在容器ipDecodedData中返回，并可以使用函数 F_VN_ExportSubContainer_String [▶ 759]导出为一个字符串。

```
hr := F_VN_ExportSubContainer_String( ipDecodedData, 0, sText, nMaxLength, hr);
```

HRESULT

与所有 TwinCAT Vision API 函数一样，该函数也返回一个HRESULT [▶ 135]，以表明执行是否成功。在成功的情况下，在随后对结果的处理中，可以区分以下成功代码。为了区分这些情况，HRESULT变量可以直接与S_OK或S_FALSE进行比较。

代码	名称	描述
16#000	S_OK	该功能成功执行。 在图像中发现了指定数量的代码。
16#001	S_FALSE	该功能成功执行。 发现的代码比图片中显示的少。
16#256	S_WATCHDOGTIMEOUT	该功能被看门狗 [▶ 140]中止。

样本

- QR 码读取 [▶ 2741]
- 读码过程中的结果评估 [▶ 2745]

相关函数

- F_VN_ReadQRCodeExp [▶ 866]
- F_VN_ReadBarcode [▶ 842]或F_VN_ReadBarcodeExp [▶ 842]
- F_VN_ReadDataMatrixCode [▶ 850]或F_VN_ReadDataMatrixCodeExp [▶ 852]
- F_VN_ReadPharmaCode [▶ 861]或F_VN_ReadPharmaCodeExp [▶ 861]

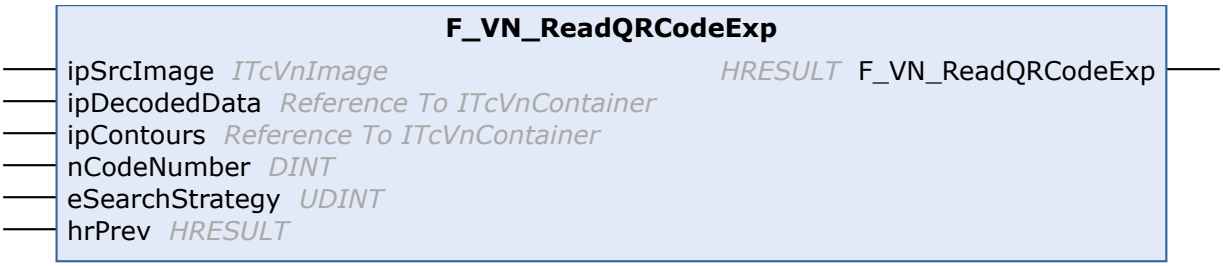
所需的许可证

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.8.20 F_VN_ReadQRCodeExp



Detect and interpret a QR code within the provided image. Can be canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_ReadQRCodeExp : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipDecodedData   : Reference To ITcVnContainer;
    ipContours      : Reference To ITcVnContainer;
    nCodeNumber     : DINT;
    eSearchStrategy : UDINT;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 元素，1 通道或 3 通道（3 通道输入应为 RGB 并内部转换为灰色））
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_Vector_String_SINT）。
ipContours	引用 ITcVnContainer [▶ 365]	以轮廓形式返回编码位置（可选项，如果不需要，则设为 0；ContainerType_Vector_Vector_TcVnPoint2_DINT）
nCodeNumber	DINT	在提供的图像中应检测到的二维码数量（目前仅支持 1 个）。
eSearchStrategy	UDINT	使用的搜索策略（ETcVn2dCodeSearchStrategy）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

注意

浮点异常

如果正在执行的PLC 任务 [▶ 63]的选项**浮点异常**启用，该功能会不必要地引起错误。因此，停用这个选项。

函数 F_VN_ReadQRCodeExp 对应于函数F_VN_ReadQRCode [▶ 864]，并由以下参数扩展。

ipContour

参数ipContour返回找到的代码位置，作为轮廓。

nCodeNumber

参数nCodeNumber定义了需要搜索多少个代码。

目前不支持搜索一个以上的 QR 码

目前不支持在一张图像中搜索多个 QR 码。因此，参数 nCodeNumber 必须设置为 1。

电子搜索策略

参数eSearchStrategy定义了代码搜索的搜索策略。标准选项TCVN_CSS_DEFAULT导致搜索策略TCVN_CSS_FIRST_NOT_INVERTED和TCVN_CSS_FIRST_NOT_FLIPPED。关于搜索策略的描述，见代码阅读搜索策略。

样本

- QR 码读取 [▶ 2741]

- 读码过程中的结果评估 [▶ 2745]

相关函数

- F_VN_ReadQRCode [▶ 864]
- F_VN_ReadBarcode [▶ 842]或F_VN_ReadBarcodeExp [▶ 842]
- F_VN_ReadDataMatrixCode [▶ 850]或F_VN_ReadDataMatrixCodeExp [▶ 852]
- F_VN_ReadPharmaCode [▶ 861]或F_VN_ReadPharmaCodeExp [▶ 861]

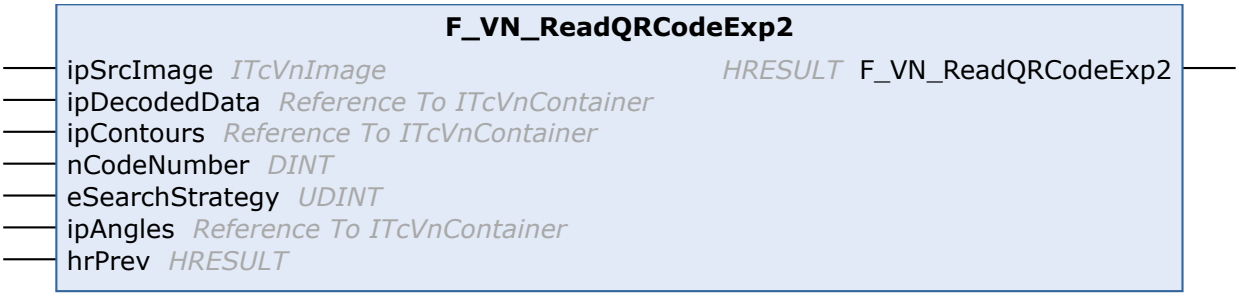
所需的许可证

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.8.21 F_VN_ReadQRCodeExp2



Detect and interpret a QR code within the provided image. Can be canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_ReadQRCodeExp2 : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipDecodedData   : Reference To ITcVnContainer;
    ipContours      : Reference To ITcVnContainer;
    nCodeNumber     : DINT;
    eSearchStrategy : UDINT;
    ipAngles        : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 元素，1 通道或 3 通道（3 通道输入应为 RGB 并内部转换为灰色））
ipDecodedData	引用 ITcVnContainer [▶ 365]	返回解码后的编码（ContainerType_Vector_String_SINT）。
ipContours	引用 ITcVnContainer [▶ 365]	以轮廓形式返回编码位置（可选项，如果不需要，则设为 0；ContainerType_Vector_Vector_TcVnPoint2_DINT）
nCodeNumber	DINT	在提供的图像中应检测到的二维码数量（目前仅支持 1 个）。
eSearchStrategy	UDINT	使用的搜索策略（ETcVn2dCodeSearchStrategy）

名称	类型	描述
ipAngles	引用 ITcVnContainer [▶ 365]	返回顺时针编码旋转角度（可选项，如果不需要，则设为 0；ContainerType_Vector_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

注意

浮点异常

如果正在执行的PLC 任务 [▶ 63]的选项**浮点异常**启用，该功能会不必要地引起错误。因此，停用这个选项。

函数 F_VN_ReadQRCodeExp 与通过以下参数扩展的功能 F_VN_ReadQRCode [▶ 864] 相对应。

ipContour

参数 `ipContour` 会返回作为轮廓找到的编码位置。

nCodeNumber

参数 `nCodeNumber` 定义了要搜索的编码数量。

目前不支持搜索多个二维码

目前不支持在一个图像中搜索多个二维码。因此，必须设为参数 nCodeNumber := 1。

eSearchStrategy

参数 `eSearchStrategy` 定义了编码搜索的搜索策略。标准选项 `TCVN_CSS_DEFAULT` 下的搜索策略为 `TCVN_CSS_FIRST_NOT_INVERTED` 和 `TCVN_CSS_FIRST_NOT_FLIPPED`。关于搜索策略的说明，请参见 Code Reading（读码）搜索策略。

旋转角度

参数 `ipAngles` 会返回找到的编码的顺时针旋转角度（单位：度）。容器类型为 `ContainerType_Vector_REAL`。该参数为可选参数，如果不需要，可设置为 0。

示例

- QR 码读取 [▶ 2741]
- 读码过程中的结果评估 [▶ 2745]

相关函数

- F_VN_ReadQRCode [▶ 864]
- F_VN_ReadBarcode [▶ 842] 或 F_VN_ReadBarcodeExp [▶ 842]
- F_VN_ReadDataMatrixCode [▶ 850] 或 VN_ReadDataMatrixCodeExp [▶ 852]
- F_VN_ReadPharmaCode [▶ 861] 或 F_VN_ReadPharmaCodeExp [▶ 861]

所需的许可证

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

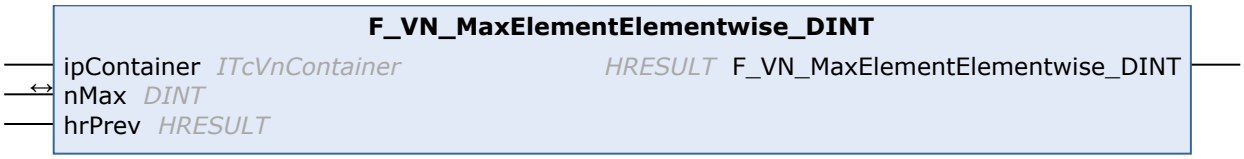
6.1.4.9 容器统计

6.1.4.9.1 F_VN_MaxElementElementwise

本章包含一些函数，通过这些函数可根据每个元素的数据类型来确定最大值。这个函数适用于包括基本元素的容器。

例如，对于 TcVnPoint2_DINT，最大 X 值和最大 Y 值相互独立确定，所以这些值不一定来自于同一个点。

6.1.4.9.1.1 F_VN_MaxElementElementwise_DINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMax        : DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMax	DINT	返回请求的元素

返回值

HRESULT [▶ 135]

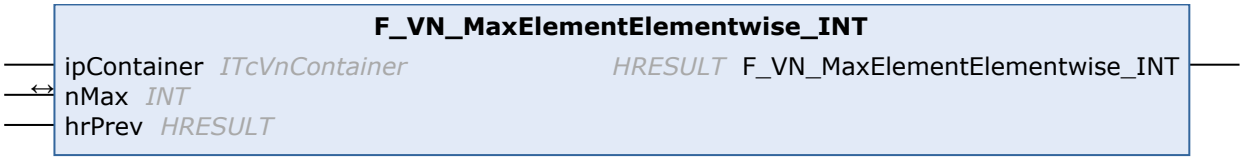
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.2 F_VN_MaxElementElementwise_INT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMax        : INT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMax	INT	返回请求的元素

返回值

HRESULT [▶ 135]

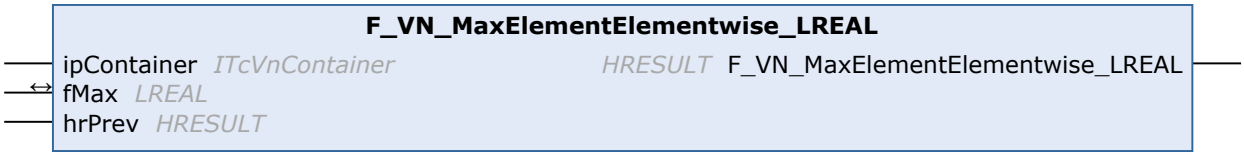
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.3 F_VN_MaxElementElementwise_LREAL



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fMax        : LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fMax	LREAL	返回请求的元素

返回值

HRESULT [▶ 135]

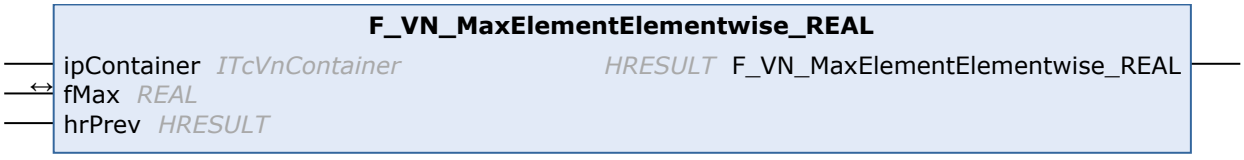
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.4 F_VN_MaxElementElementwise_REAL



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fMax        : REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fMax	REAL	返回请求的元素

返回值

HRESULT [▶ 135]

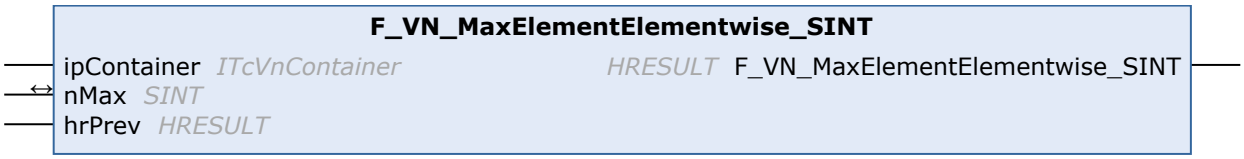
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.5 F_VN_MaxElementElementwise_SINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMax        : SINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMax	SINT	返回请求的元素

返回值

HRESULT [▶ 135]

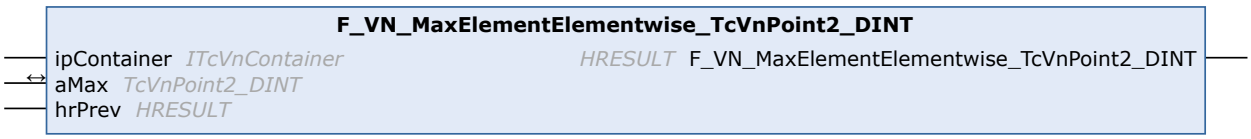
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.6 F_VN_MaxElementElementwise_TcVnPoint2_DINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnPoint2_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnPoint2_DINT [▶ 152]	返回请求的元素

返回值

HRESULT [▶ 135]

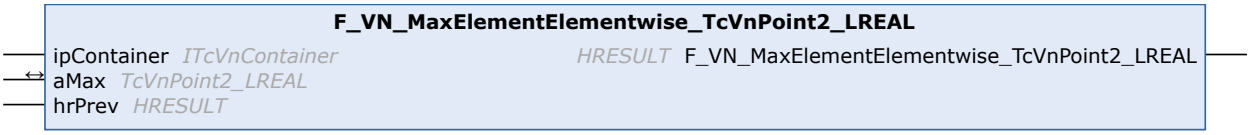
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.7 F_VN_MaxElementElementwise_TcVnPoint2_LREAL



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnPoint2_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnPoint2_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnPoint2_LREAL [▶ 152]	返回请求的元素

返回值

HRESULT [▶ 135]

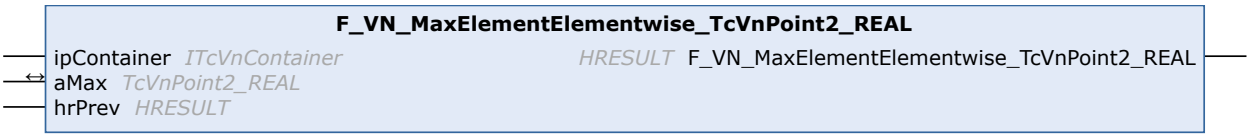
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.8 F_VN_MaxElementElementwise_TcVnPoint2_REAL



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnPoint2_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnPoint2_REAL [▶ 152]	返回请求的元素

返回值

HRESULT [▶ 135]

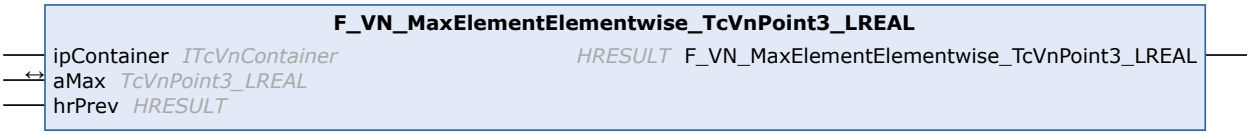
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.9 F_VN_MaxElementElementwise_TcVnPoint3_LREAL



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnPoint3_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnPoint3_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnPoint3_LREAL [▶ 152]	返回请求的元素

返回值

HRESULT [▶ 135]

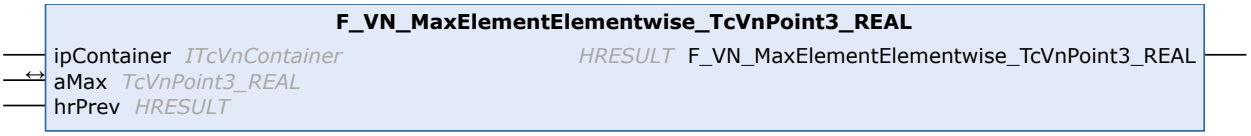
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.10 F_VN_MaxElementElementwise_TcVnPoint3_REAL



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnPoint3_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnPoint3_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnPoint3_REAL [▶ 152]	返回请求的元素

返回值

HRESULT [▶ 135]

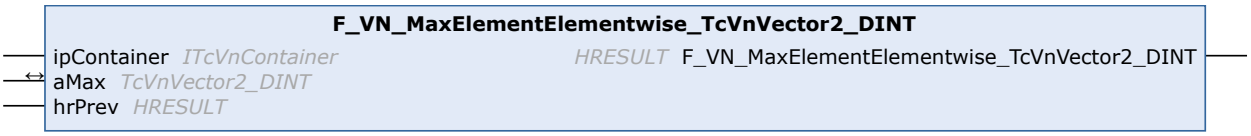
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.11 F_VN_MaxElementElementwise_TcVnVector2_DINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector2_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector2_DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector2_DINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

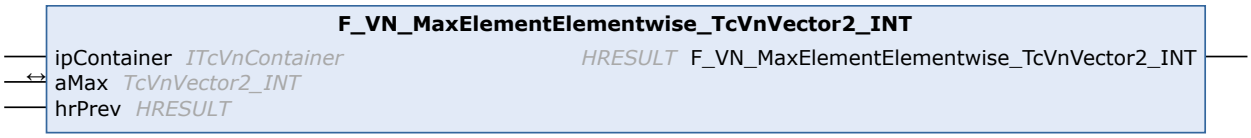
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.12 F_VN_MaxElementElementwise_TcVnVector2_INT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector2_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector2_INT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector2_INT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

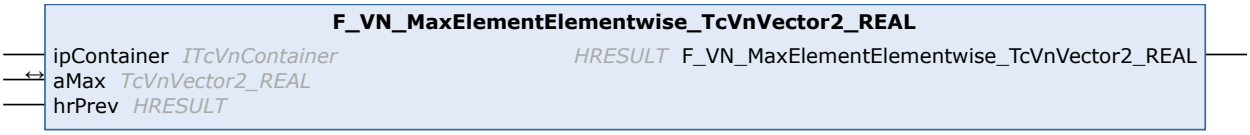
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.13 F_VN_MaxElementElementwise_TcVnVector2_REAL



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector2_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector2_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector2_REAL [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

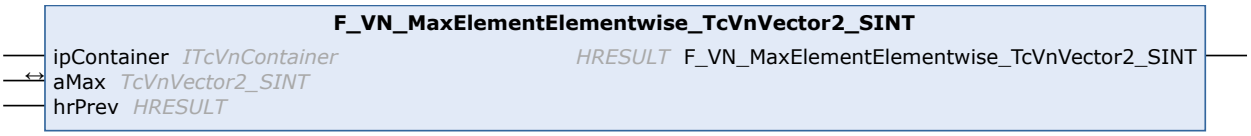
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.14 F_VN_MaxElementElementwise_TcVnVector2_SINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector2_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector2_SINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector2_SINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

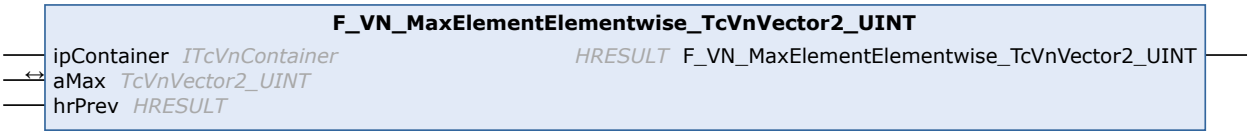
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.15 F_VN_MaxElementElementwise_TcVnVector2_UINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector2_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector2_UINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector2_UINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

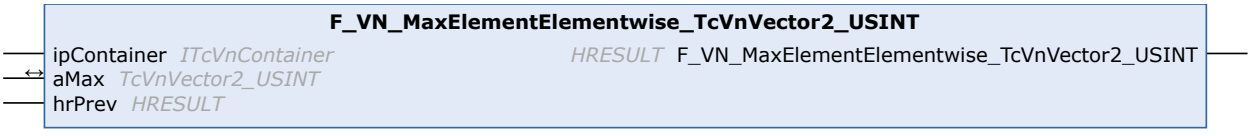
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.16 F_VN_MaxElementElementwise_TcVnVector2_USINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector2_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector2_USINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector2_USINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

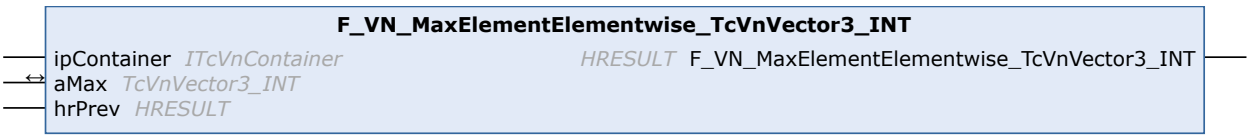
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.17 F_VN_MaxElementElementwise_TcVnVector3_INT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector3_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector3_INT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector3_INT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

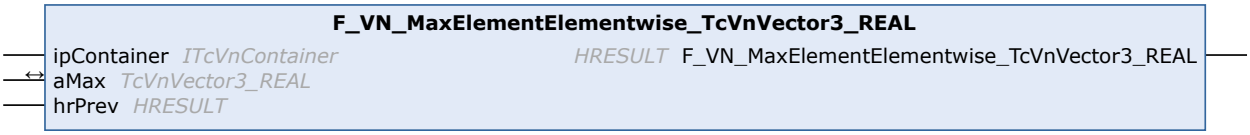
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.18 F_VN_MaxElementElementwise_TcVnVector3_REAL



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector3_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector3_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector3_REAL [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

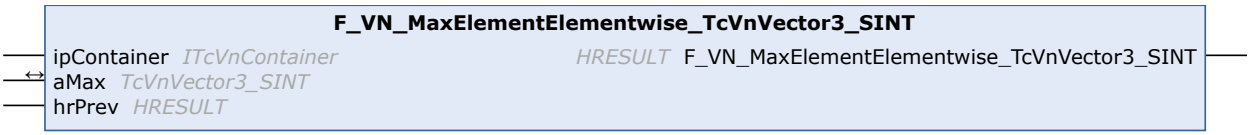
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.19 F_VN_MaxElementElementwise_TcVnVector3_SINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector3_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector3_SINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector3_SINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

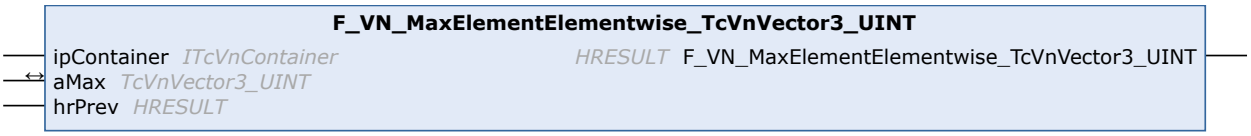
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.20 F_VN_MaxElementElementwise_TcVnVector3_UINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector3_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector3_UINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector3_UINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

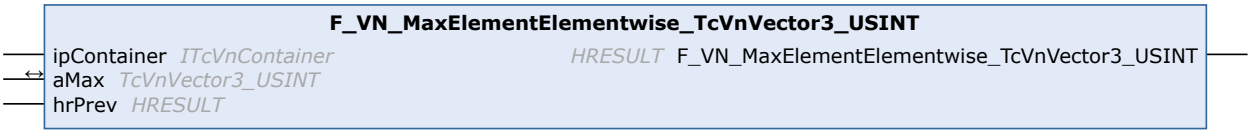
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.21 F_VN_MaxElementElementwise_TcVnVector3_USINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector3_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector3_USINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector3_USINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

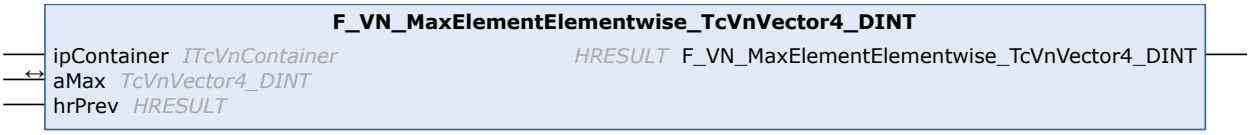
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.22 F_VN_MaxElementElementwise_TcVnVector4_DINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector4_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector4_DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector4_DINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

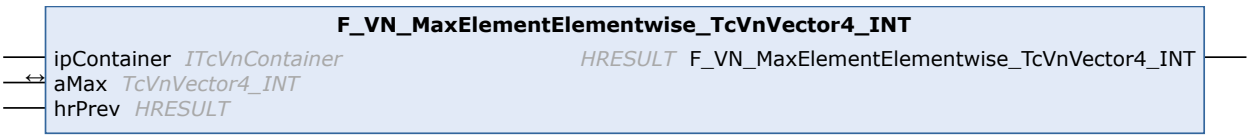
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.23 F_VN_MaxElementElementwise_TcVnVector4_INT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector4_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector4_INT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector4_INT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

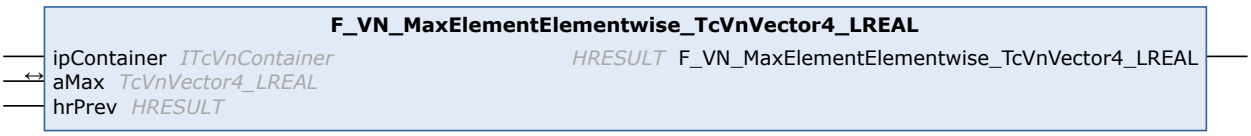
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.24 F_VN_MaxElementElementwise_TcVnVector4_LREAL



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector4_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector4_LREAL [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

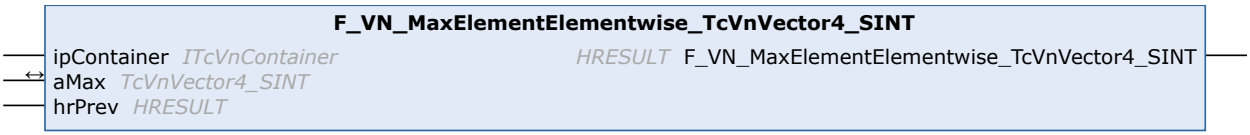
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.25 F_VN_MaxElementElementwise_TcVnVector4_SINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector4_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector4_SINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector4_SINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

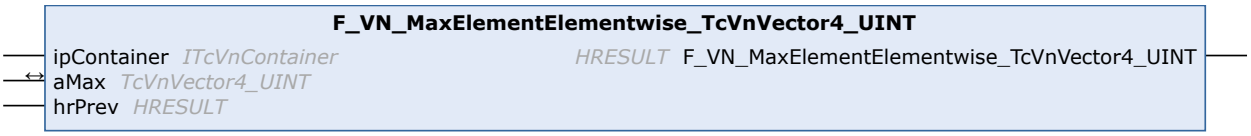
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.26 F_VN_MaxElementElementwise_TcVnVector4_UINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector4_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector4_UINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector4_UINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

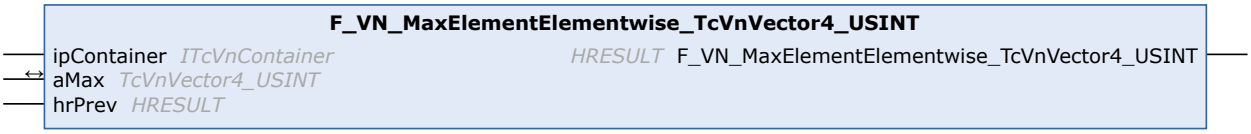
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.27 F_VN_MaxElementElementwise_TcVnVector4_USINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector4_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector4_USINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector4_USINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

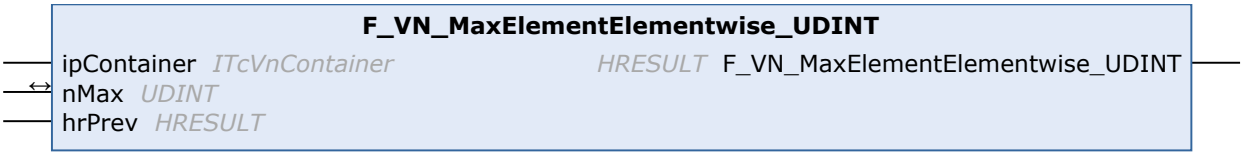
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.28 F_VN_MaxElementElementwise_UDINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_UDINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMax        : UDINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMax	UDINT	返回请求的元素

返回值

HRESULT [▶ 135]

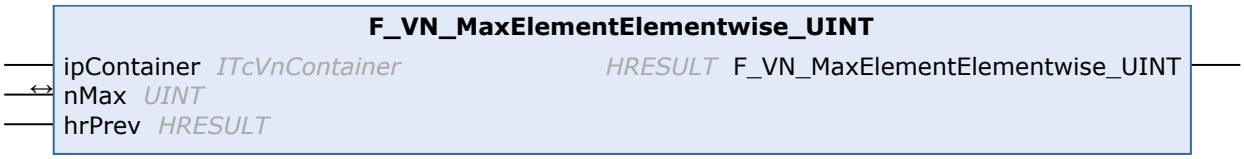
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.29 F_VN_MaxElementElementwise_UINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMax        : UINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMax	UINT	返回请求的元素

返回值

HRESULT [▶ 135]

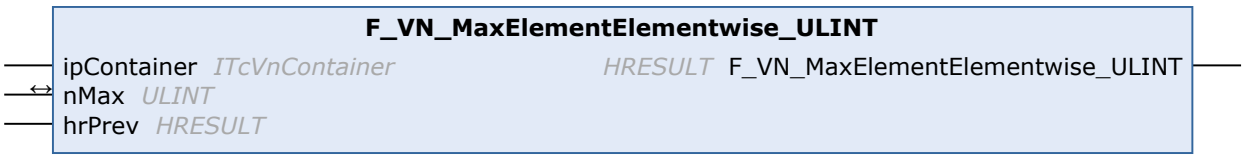
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.30 F_VN_MaxElementElementwise_ULINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_ULINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMax        : ULINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMax	ULINT	返回请求的元素

返回值

HRESULT [▶ 135]

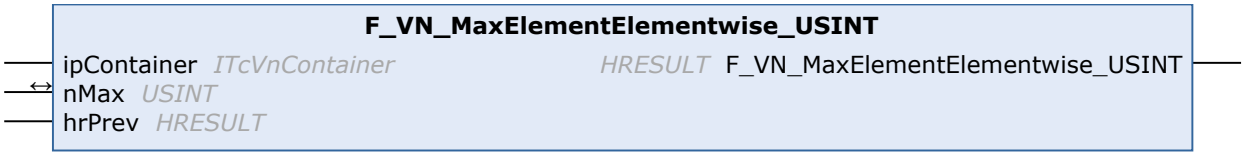
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.31 F_VN_MaxElementElementwise_USINT



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMax        : USINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMax	USINT	返回请求的元素

返回值

HRESULT [▶ 135]

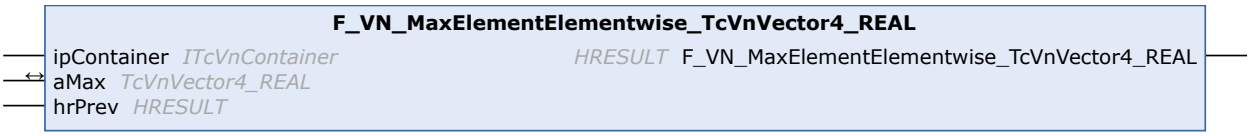
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.1.32 F_VN_MaxElementElementwise_TcVnVector4_REAL



Gets the element wise maximum container element.

语法

定义：

```
FUNCTION F_VN_MaxElementElementwise_TcVnVector4_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMax        : TcVnVector4_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMax	TcVnVector4_REAL [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

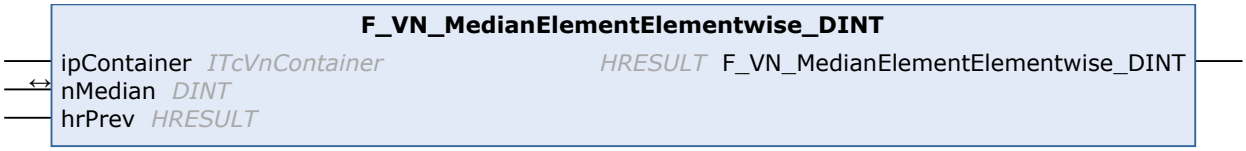
开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2 F_VN_MedianElementElementwise

本章包含根据每个元素的数据类型确定中位数的函数。这个函数适用于包括基本元素的容器。

例如，对于 TcVnPoint2_DINT 来说，中位 X 值和中位 Y 值相互独立确定，所以这些值不一定来自于同一个点。

6.1.4.9.2.1 F_VN_MedianElementElementwise_DINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMedian     : DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMedian	DINT	返回请求的元素

返回值

HRESULT [▶ 135]

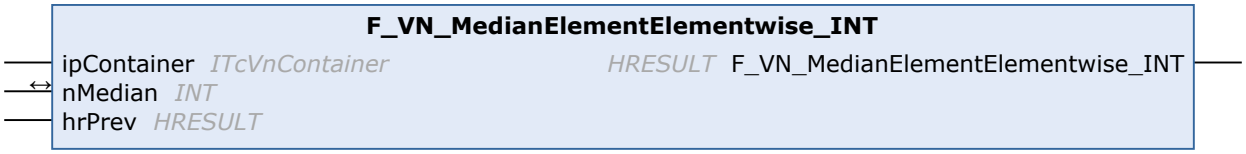
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.2 F_VN_MedianElementElementwise_INT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMedian     : INT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMedian	INT	返回请求的元素

返回值

HRESULT [▶ 135]

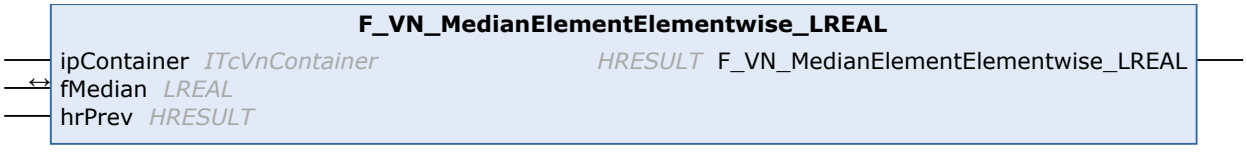
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.3 F_VN_MedianElementElementwise_LREAL



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fMedian     : LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fMedian	LREAL	返回请求的元素

返回值

HRESULT [▶ 135]

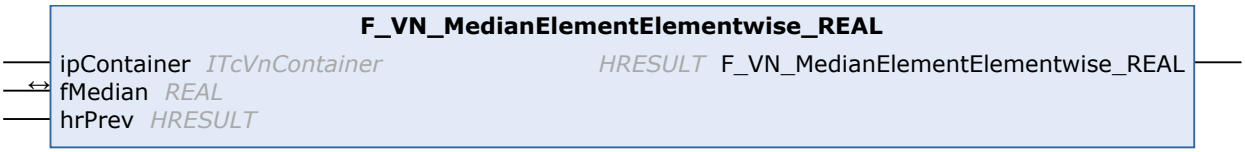
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.4 F_VN_MedianElementElementwise_REAL



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fMedian     : REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fMedian	REAL	返回请求的元素

返回值

HRESULT [▶ 135]

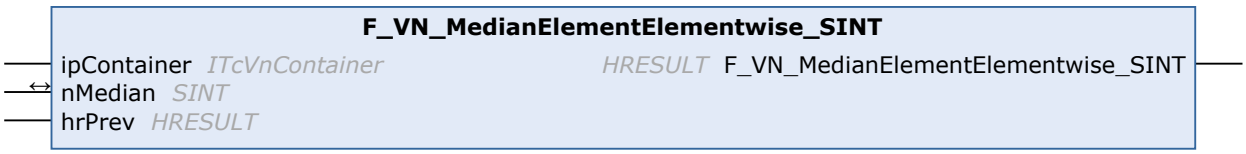
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.5 F_VN_MedianElementElementwise_SINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMedian     : SINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMedian	SINT	返回请求的元素

返回值

HRESULT [▶ 135]

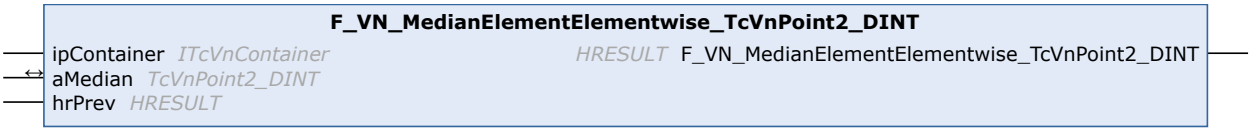
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.6 F_VN_MedianElementElementwise_TcVnPoint2_DINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnPoint2_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnPoint2_DINT [▶ 152]	返回请求的元素

返回值

HRESULT [▶ 135]

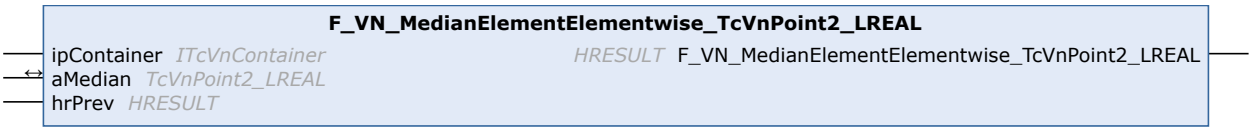
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.7 F_VN_MedianElementElementwise_TcVnPoint2_LREAL



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnPoint2_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnPoint2_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnPoint2_LREAL [▶ 152]	返回请求的元素

返回值

HRESULT [▶ 135]

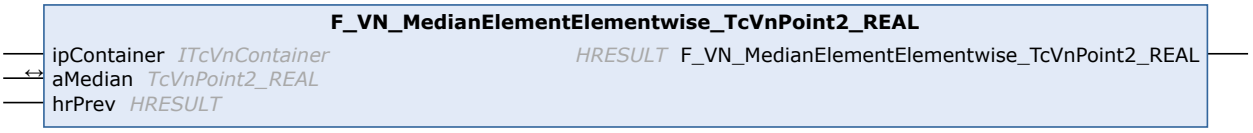
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.8 F_VN_MedianElementElementwise_TcVnPoint2_REAL



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnPoint2_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnPoint2_REAL [▶ 152]	返回请求的元素

返回值

HRESULT [▶ 135]

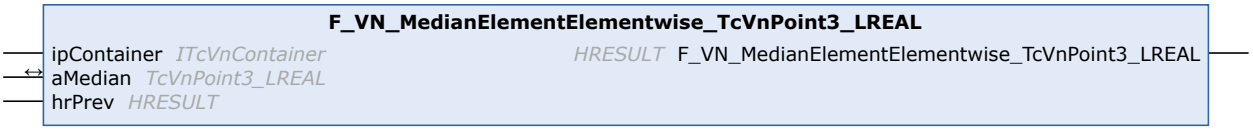
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.9 F_VN_MedianElementElementwise_TcVnPoint3_LREAL



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnPoint3_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnPoint3_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnPoint3_LREAL [▶ 152]	返回请求的元素

返回值

HRESULT [▶ 135]

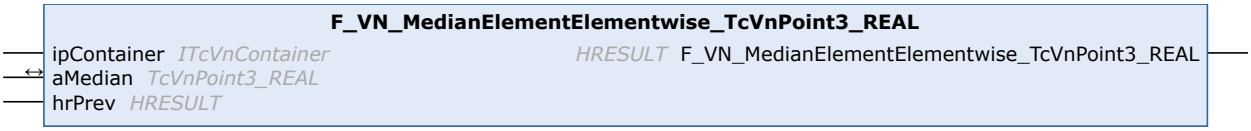
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.10 F_VN_MedianElementElementwise_TcVnPoint3_REAL



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnPoint3_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnPoint3_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnPoint3_REAL [▶ 152]	返回请求的元素

返回值

HRESULT [▶ 135]

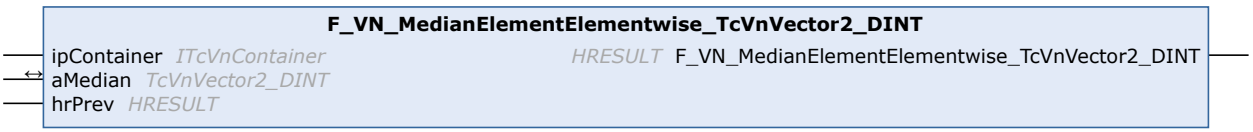
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.11 F_VN_MedianElementElementwise_TcVnVector2_DINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector2_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector2_DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector2_DINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

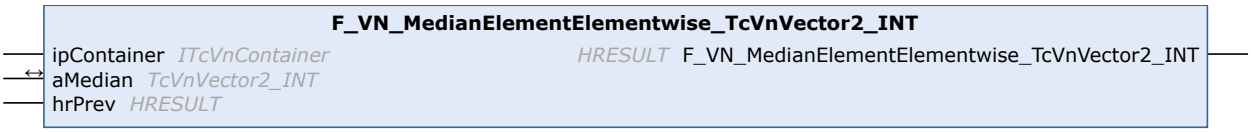
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.12 F_VN_MedianElementElementwise_TcVnVector2_INT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector2_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector2_INT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector2_INT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

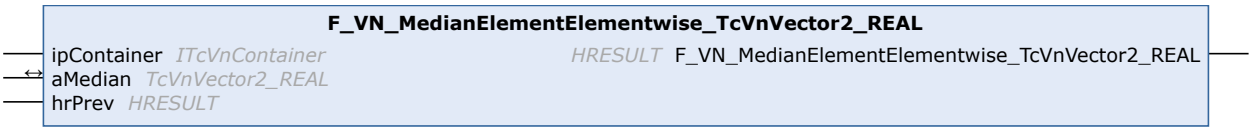
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.13 F_VN_MedianElementElementwise_TcVnVector2_REAL



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector2_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector2_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector2_REAL [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

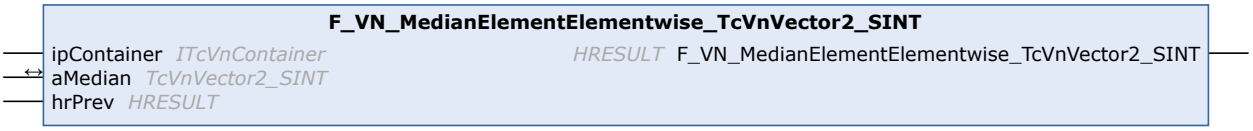
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.14 F_VN_MedianElementElementwise_TcVnVector2_SINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector2_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector2_SINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector2_SINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

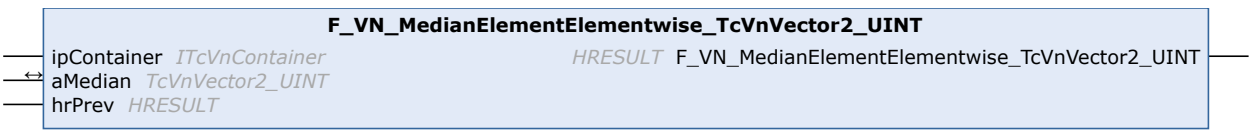
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.15 F_VN_MedianElementElementwise_TcVnVector2_UINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector2_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector2_UINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector2_UINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

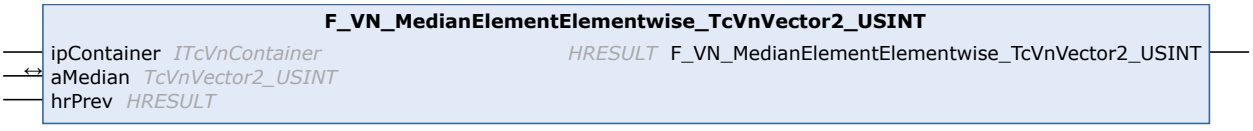
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.16 F_VN_MedianElementElementwise_TcVnVector2_USINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector2_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector2_USINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector2_USINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

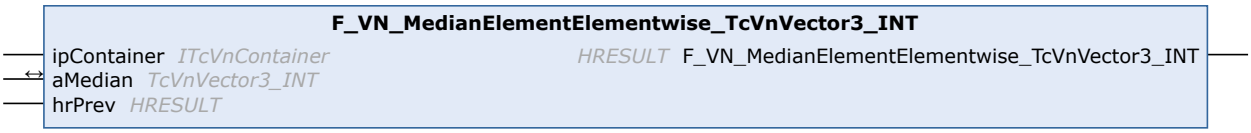
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.17 F_VN_MedianElementElementwise_TcVnVector3_INT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector3_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector3_INT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector3_INT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

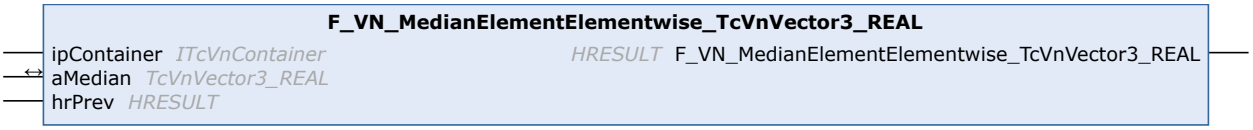
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.18 F_VN_MedianElementElementwise_TcVnVector3_REAL



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector3_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector3_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector3_REAL [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

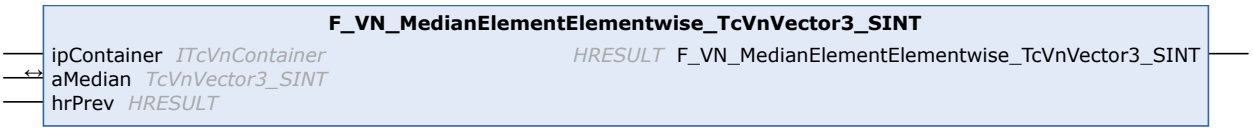
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.19 F_VN_MedianElementElementwise_TcVnVector3_SINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector3_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector3_SINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector3_SINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

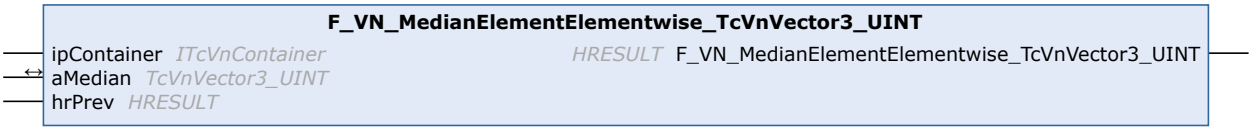
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.20 F_VN_MedianElementElementwise_TcVnVector3_UINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector3_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector3_UINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector3_UINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

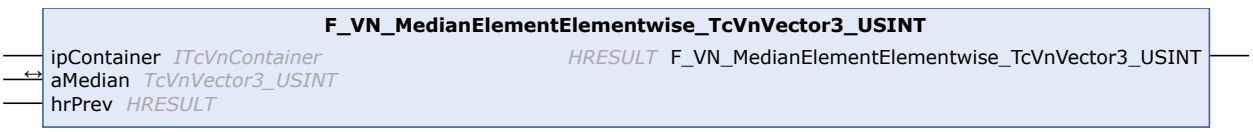
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.21 F_VN_MedianElementElementwise_TcVnVector3_USINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector3_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector3_USINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector3_USINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

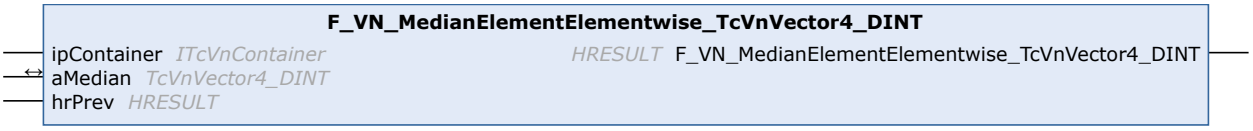
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.22 F_VN_MedianElementElementwise_TcVnVector4_DINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector4_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector4_DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector4_DINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

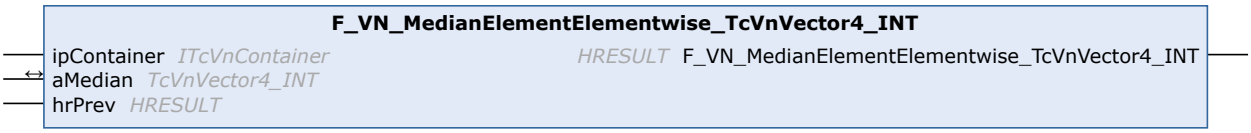
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.23 F_VN_MedianElementElementwise_TcVnVector4_INT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector4_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector4_INT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector4_INT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

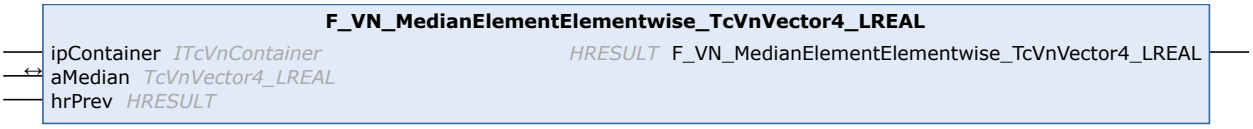
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.24 F_VN_MedianElementElementwise_TcVnVector4_LREAL



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector4_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector4_LREAL [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

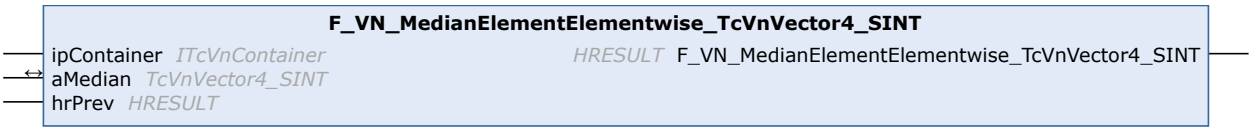
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.25 F_VN_MedianElementElementwise_TcVnVector4_SINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector4_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector4_SINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector4_SINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

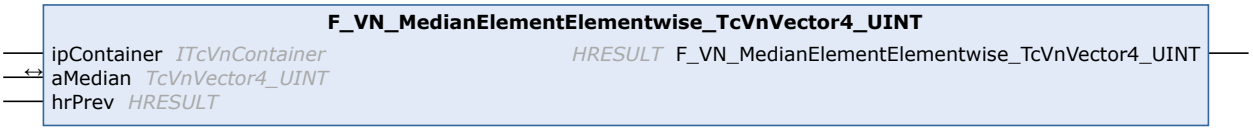
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.26 F_VN_MedianElementElementwise_TcVnVector4_UINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector4_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector4_UINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector4_UINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

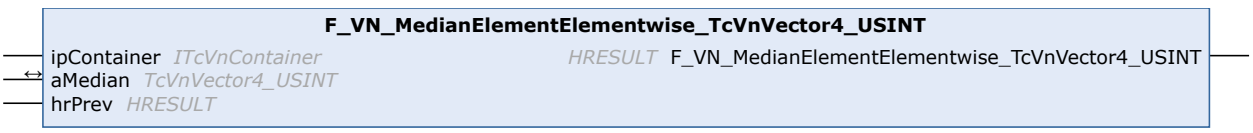
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.27 F_VN_MedianElementElementwise_TcVnVector4_USINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector4_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector4_USINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector4_USINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

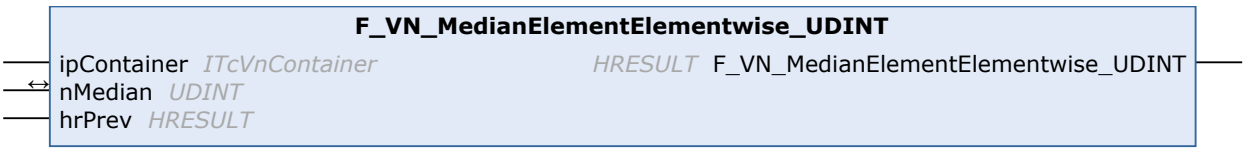
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.28 F_VN_MedianElementElementwise_UDINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_UDINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMedian     : UDINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMedian	UDINT	返回请求的元素

返回值

HRESULT [▶ 135]

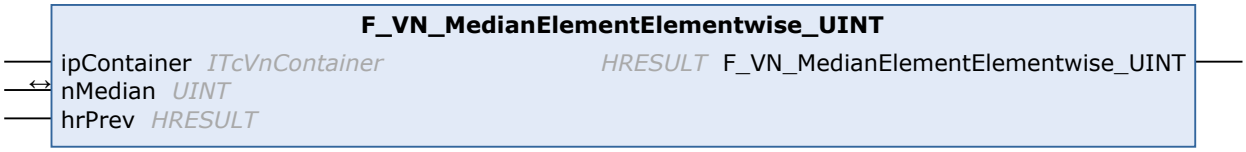
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.29 F_VN_MedianElementElementwise_UINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMedian     : UINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMedian	UINT	返回请求的元素

返回值

HRESULT [▶ 135]

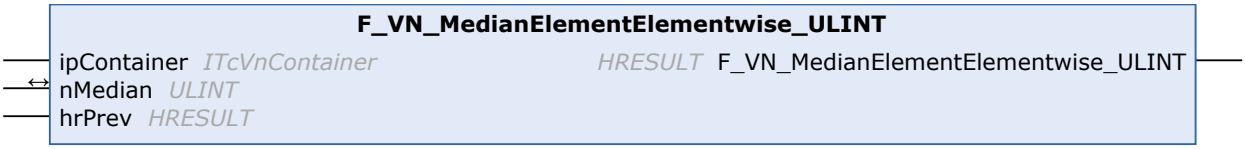
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.30 F_VN_MedianElementElementwise_ULINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_ULINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMedian     : ULINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMedian	ULINT	返回请求的元素

返回值

HRESULT [▶ 135]

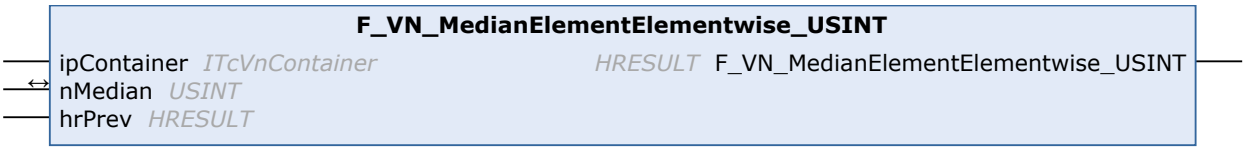
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.31 F_VN_MedianElementElementwise_USINT



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMedian     : USINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMedian	USINT	返回请求的元素

返回值

HRESULT [▶ 135]

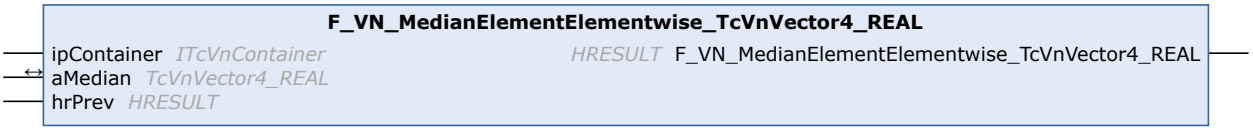
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.2.32 F_VN_MedianElementElementwise_TcVnVector4_REAL



Gets the element wise median container element.

语法

定义：

```
FUNCTION F_VN_MedianElementElementwise_TcVnVector4_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMedian     : TcVnVector4_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector4_REAL [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

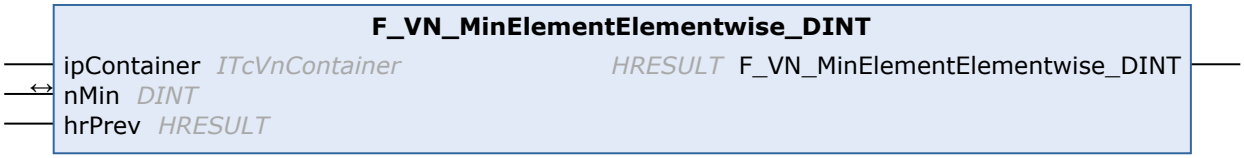
开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3 F_VN_MinElementElementwise

本章包含根据每个元素的数据类型来确定最小值的函数。这个函数适用于包括基本元素的容器。

例如，对于 TcVnPoint2_DINT，最小 X 值和最小 Y 值相互独立确定，所以这些值不一定来自于同一个点。

6.1.4.9.3.1 F_VN_MinElementElementwise_DINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMin        : DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMin	DINT	返回请求的元素

返回值

HRESULT [▶ 135]

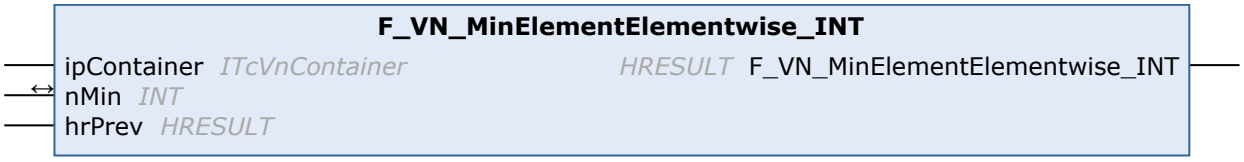
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.2 F_VN_MinElementElementwise_INT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMin        : INT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMin	INT	返回请求的元素

返回值

HRESULT [▶ 135]

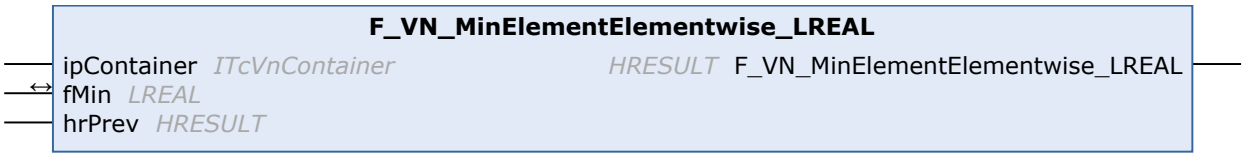
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.3 F_VN_MinElementElementwise_LREAL



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fMin        : LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fMin	LREAL	返回请求的元素

返回值

HRESULT [▶ 135]

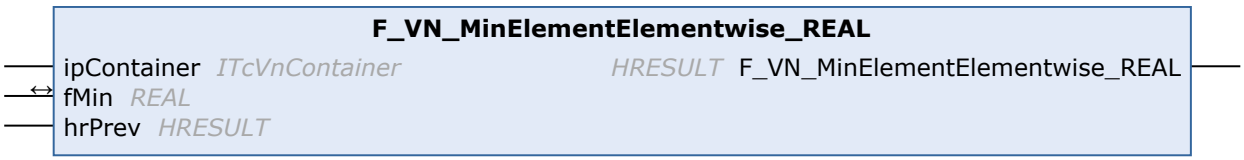
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.4 F_VN_MinElementElementwise_REAL



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fMin        : REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fMin	REAL	返回请求的元素

返回值

HRESULT [▶ 135]

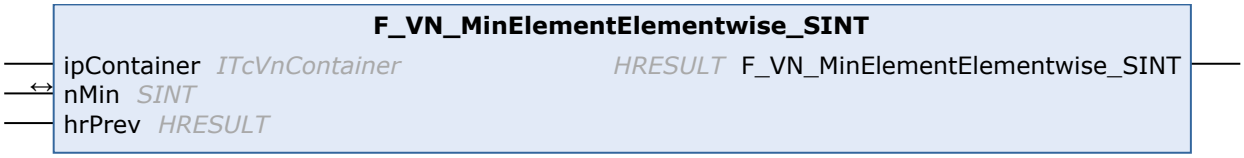
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.5 F_VN_MinElementElementwise_SINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMin        : SINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMin	SINT	返回请求的元素

返回值

HRESULT [▶ 135]

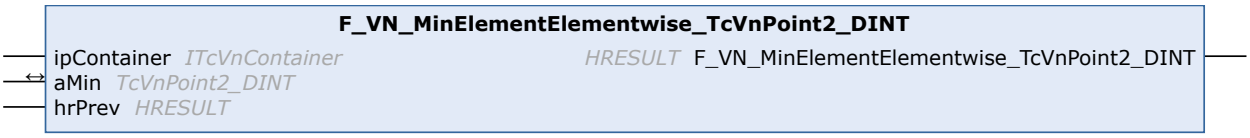
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.6 F_VN_MinElementElementwise_TcVnPoint2_DINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnPoint2_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnPoint2_DINT [▶ 152]	返回请求的元素

返回值

HRESULT [▶ 135]

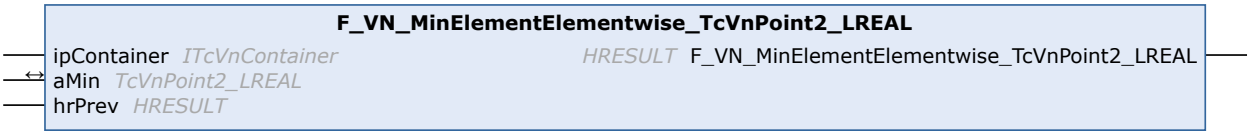
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.7 F_VN_MinElementElementwise_TcVnPoint2_LREAL



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnPoint2_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnPoint2_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnPoint2_LREAL [▶ 152]	返回请求的元素

返回值

HRESULT [▶ 135]

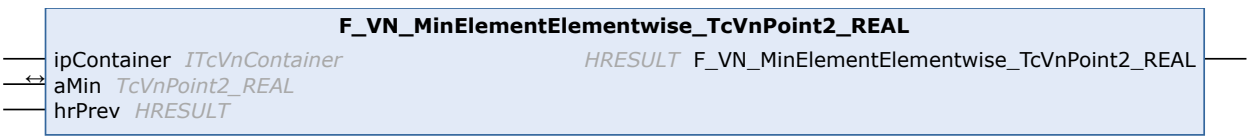
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.8 F_VN_MinElementElementwise_TcVnPoint2_REAL



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnPoint2_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnPoint2_REAL [▶ 152]	返回请求的元素

返回值

HRESULT [▶ 135]

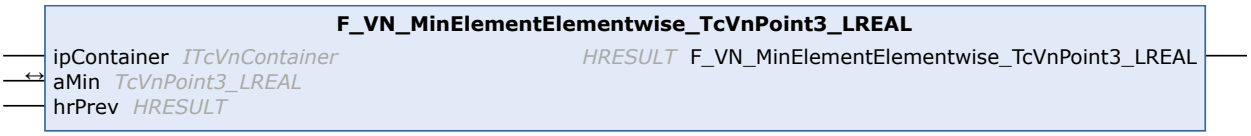
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.9 F_VN_MinElementElementwise_TcVnPoint3_LREAL



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnPoint3_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnPoint3_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnPoint3_LREAL [▶ 152]	返回请求的元素

返回值

HRESULT [▶ 135]

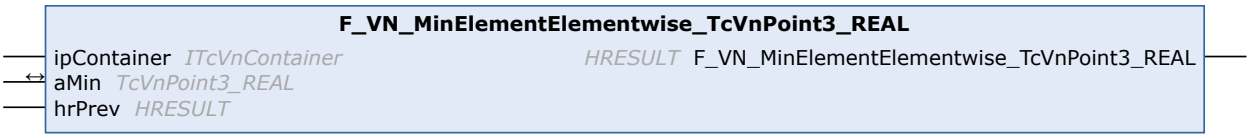
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.10 F_VN_MinElementElementwise_TcVnPoint3_REAL



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnPoint3_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnPoint3_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnPoint3_REAL [▶ 152]	返回请求的元素

返回值

HRESULT [▶ 135]

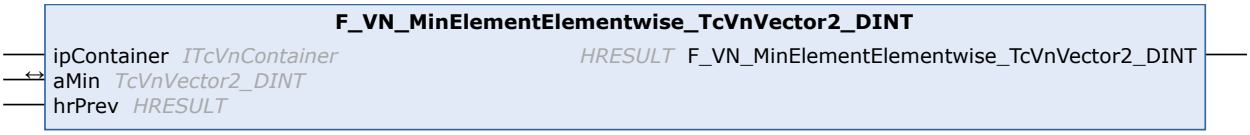
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.11 F_VN_MinElementElementwise_TcVnVector2_DINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector2_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector2_DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector2_DINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

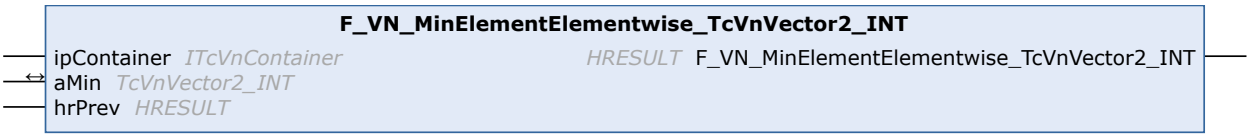
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.12 F_VN_MinElementElementwise_TcVnVector2_INT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector2_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector2_INT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector2_INT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

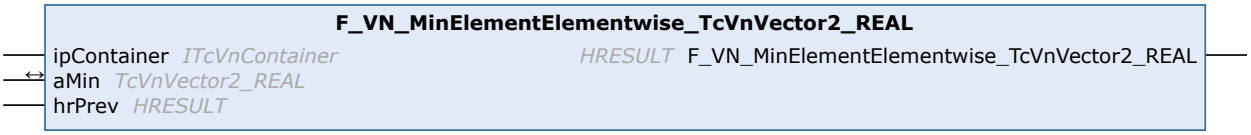
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.13 F_VN_MinElementElementwise_TcVnVector2_REAL



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector2_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector2_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector2_REAL [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

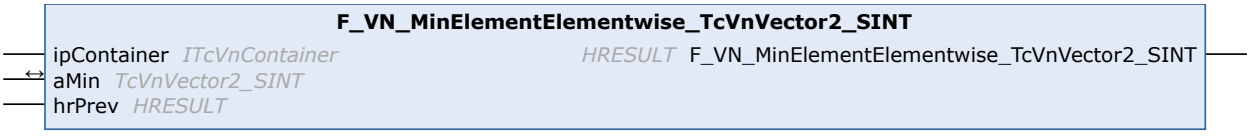
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.14 F_VN_MinElementElementwise_TcVnVector2_SINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector2_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector2_SINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector2_SINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

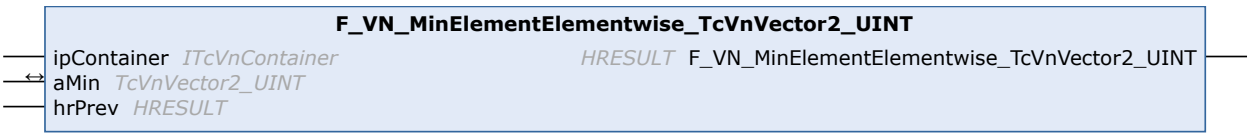
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.15 F_VN_MinElementElementwise_TcVnVector2_UINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector2_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector2_UINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector2_UINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

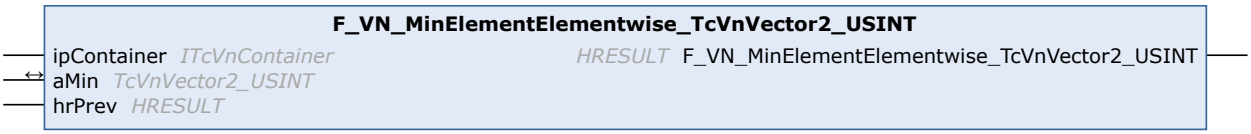
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.16 F_VN_MinElementElementwise_TcVnVector2_USINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector2_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector2_USINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector2_USINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

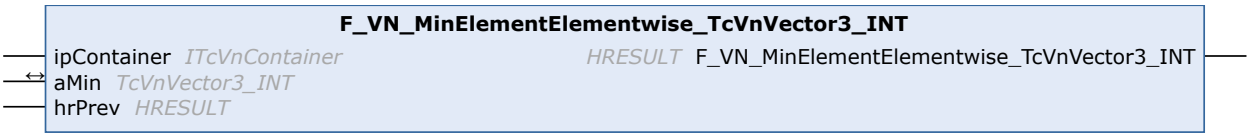
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.17 F_VN_MinElementElementwise_TcVnVector3_INT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector3_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector3_INT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector3_INT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

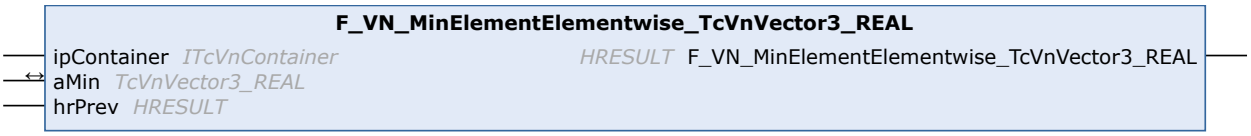
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.18 F_VN_MinElementElementwise_TcVnVector3_REAL



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector3_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector3_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector3_REAL [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

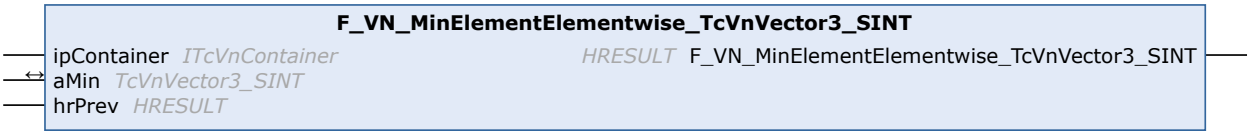
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.19 F_VN_MinElementElementwise_TcVnVector3_SINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector3_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector3_SINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector3_SINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

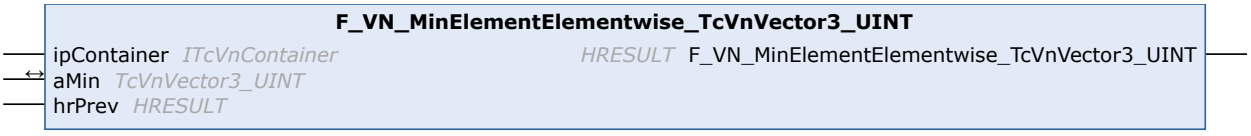
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.20 F_VN_MinElementElementwise_TcVnVector3_UINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector3_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector3_UINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector3_UINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

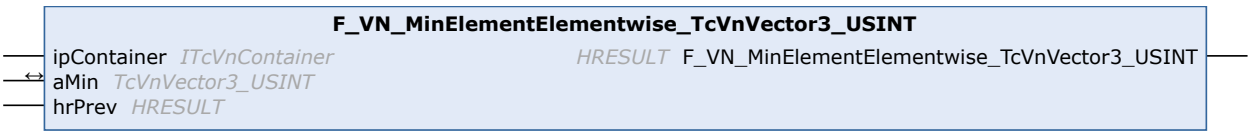
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.21 F_VN_MinElementElementwise_TcVnVector3_USINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector3_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector3_USINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector3_USINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

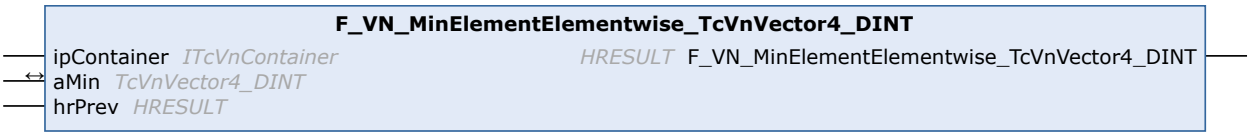
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.22 F_VN_MinElementElementwise_TcVnVector4_DINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector4_DINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector4_DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector4_DINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

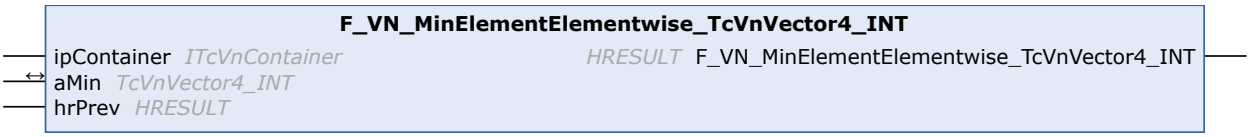
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.23 F_VN_MinElementElementwise_TcVnVector4_INT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector4_INT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector4_INT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector4_INT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

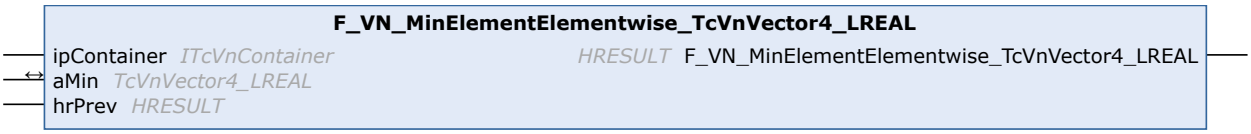
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.24 F_VN_MinElementElementwise_TcVnVector4_LREAL



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector4_LREAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector4_LREAL [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

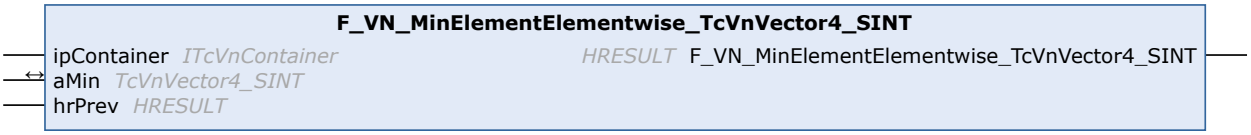
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.25 F_VN_MinElementElementwise_TcVnVector4_SINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector4_SINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector4_SINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector4_SINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

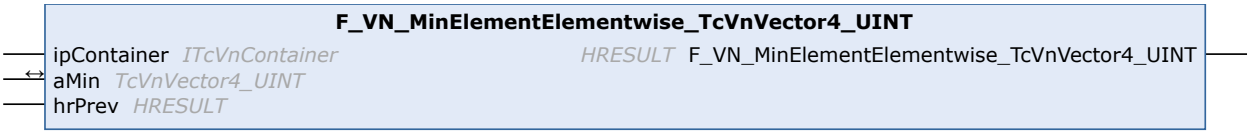
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.26 F_VN_MinElementElementwise_TcVnVector4_UINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector4_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector4_UINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector4_UINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

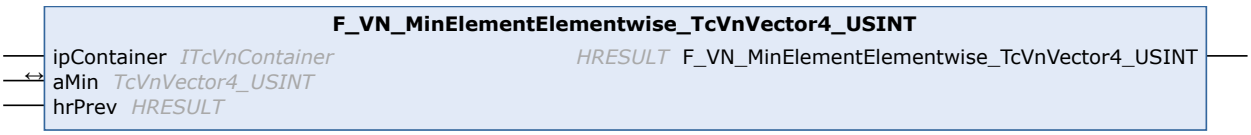
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.27 F_VN_MinElementElementwise_TcVnVector4_USINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector4_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector4_USINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector4_USINT [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

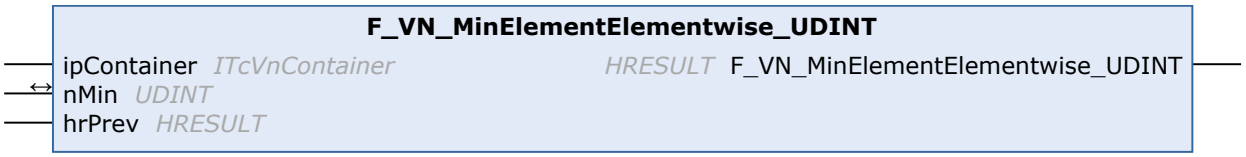
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.28 F_VN_MinElementElementwise_UDINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_UDINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMin        : UDINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMin	UDINT	返回请求的元素

返回值

HRESULT [▶ 135]

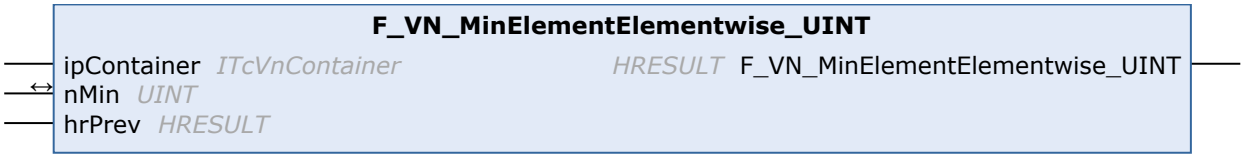
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.29 F_VN_MinElementElementwise_UINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_UINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMin        : UINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMin	UINT	返回请求的元素

返回值

HRESULT [▶ 135]

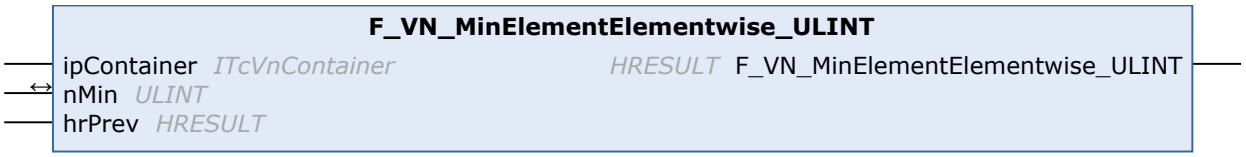
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.30 F_VN_MinElementElementwise_ULINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_ULINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMin        : ULINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMin	ULINT	返回请求的元素

返回值

HRESULT [▶ 135]

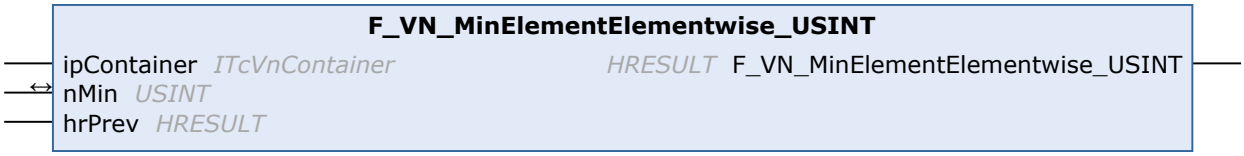
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.31 F_VN_MinElementElementwise_USINT



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_USINT : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    nMin        : USINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nMin	USINT	返回请求的元素

返回值

HRESULT [▶ 135]

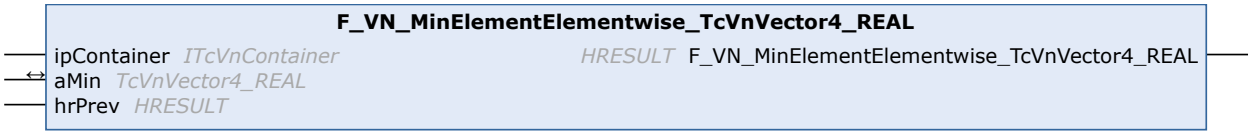
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.3.32 F_VN_MinElementElementwise_TcVnVector4_REAL



Gets the element wise minimum container element.

语法

定义：

```
FUNCTION F_VN_MinElementElementwise_TcVnVector4_REAL : HRESULT
VAR_INPUT
    ipContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aMin        : TcVnVector4_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer	ITcVnContainer [▶ 365]	源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMin	TcVnVector4_REAL [▶ 153]	返回请求的元素

返回值

HRESULT [▶ 135]

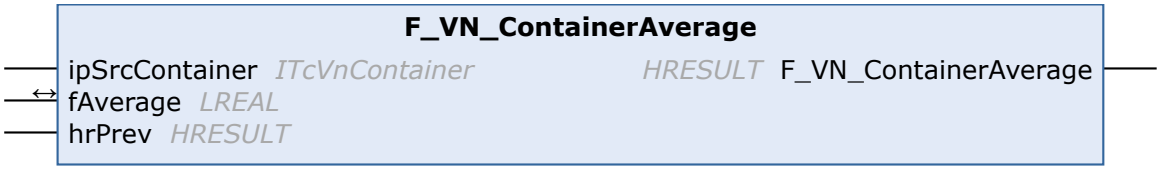
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.4 F_VN_ContainerAverage



Calculate the average of elements in a container.

语法

定义：

```
FUNCTION F_VN_ContainerAverage : HRESULT
VAR_INPUT
    ipSrcContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fAverage       : LREAL;
END_VAR
VAR_INPUT
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	源容器（支持算术、点和向量元素类型。对于点和向量，使用欧几里得范数）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fAverage	LREAL	返回计算出的平均值

返回值

HRESULT [▶ 135]

更多信息

下表显示了哪些功能可用于哪种容器类型：

容器类型	可用功能
元素的容器	F_VN_ContainerAverage [▶ 950] F_VN_ContainerAverageVariance [▶ 955]
向量的容器，每个都有 2 个元素	F_VN_ContainerAverageElementwise2 [▶ 951] F_VN_ContainerAverageVarianceElementwise2 [▶ 956]
向量的容器，每个都有 3 个元素	F_VN_ContainerAverageElementwise3 [▶ 953] F_VN_ContainerAverageVarianceElementwise3 [▶ 958]
向量的容器，每个都有 4 个元素	F_VN_ContainerAverageElementwise4 [▶ 954] F_VN_ContainerAverageVarianceElementwise4 [▶ 959]

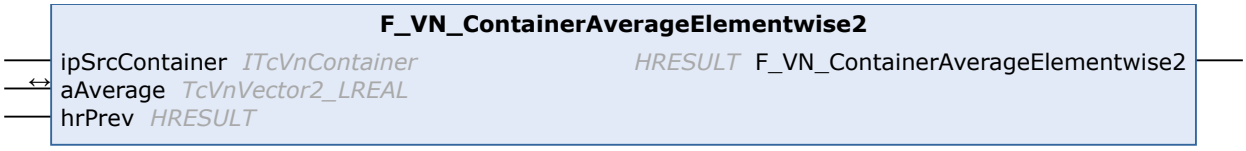
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.5 F_VN_ContainerAverageElementwise2



Calculate the elementwise average of 2D elements in a container.

语法

定义：

```
FUNCTION F_VN_ContainerAverageElementwise2 : HRESULT
VAR_INPUT
    ipSrcContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aAverage       : TcVnVector2_LREAL;
END_VAR
VAR_INPUT
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	带有二维点或向量的源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aAverage	TcVnVector2_LREAL [▶ 153]	返回计算出的平均值

返回值

HRESULT [▶ 135]

更多信息

对于对每个容器元素有两个基本子元素的容器，这个函数逐个元素计算平均值。

下表显示了哪些功能可用于哪种容器类型：

容器类型	可用功能
元素的容器	F_VN_ContainerAverage [▶ 950] F_VN_ContainerAverageVariance [▶ 955]
向量的容器，每个都有 2 个元素	F_VN_ContainerAverageElementwise2 [▶ 951] F_VN_ContainerAverageVarianceElementwise2 [▶ 956]
向量的容器，每个都有 3 个元素	F_VN_ContainerAverageElementwise3 [▶ 953] F_VN_ContainerAverageVarianceElementwise3 [▶ 958]
向量的容器，每个都有 4 个元素	F_VN_ContainerAverageElementwise4 [▶ 954] F_VN_ContainerAverageVarianceElementwise4 [▶ 959]

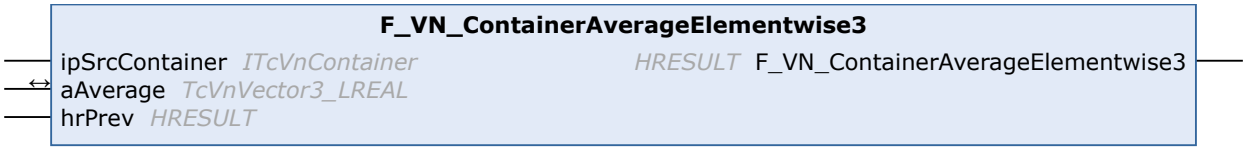
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.6 F_VN_ContainerAverageElementwise3



Calculate the elementwise average of 3D elements in a container.

语法

定义：

```
FUNCTION F_VN_ContainerAverageElementwise3 : HRESULT
VAR_INPUT
    ipSrcContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aAverage       : TcVnVector3_LREAL;
END_VAR
VAR_INPUT
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	带有三维点或向量的源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aAverage	TcVnVector3_LREAL [▶ 153]	返回计算出的平均值

返回值

HRESULT [▶ 135]

更多信息

下表显示了哪些功能可用于哪种容器类型：

容器类型	可用功能
元素的容器	F_VN_ContainerAverage [▶ 950] F_VN_ContainerAverageVariance [▶ 955]
向量的容器，每个都有 2 个元素	F_VN_ContainerAverageElementwise2 [▶ 951] F_VN_ContainerAverageVarianceElementwise2 [▶ 956]
向量的容器，每个都有 3 个元素	F_VN_ContainerAverageElementwise3 [▶ 953]

容器类型	可用功能
	F_VN_ContainerAverageVarianceElementwise3 [▶ 958]
向量的容器，每个都有 4 个元素	F_VN_ContainerAverageElementwise4 [▶ 954] F_VN_ContainerAverageVarianceElementwise4 [▶ 959]

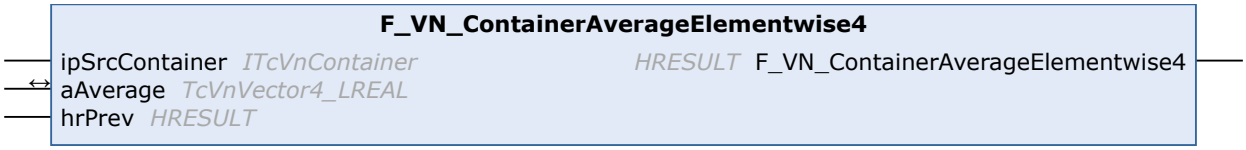
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.7 F_VN_ContainerAverageElementwise4



Calculate the elementwise average of 4D elements in a container.

语法

定义：

```
FUNCTION F_VN_ContainerAverageElementwise4 : HRESULT
VAR_INPUT
    ipSrcContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aAverage       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	带有四维向量的源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aAverage	TcVnVector4_LREAL [▶ 153]	返回计算出的平均值

返回值

HRESULT [▶ 135]

更多信息

下表显示了哪些功能可用于哪种容器类型：

容器类型	可用功能
元素的容器	F_VN_ContainerAverage [▶ 950] F_VN_ContainerAverageVariance [▶ 955]
向量的容器，每个都有 2 个元素	F_VN_ContainerAverageElementwise2 [▶ 951] F_VN_ContainerAverageVarianceElementwise2 [▶ 956]
向量的容器，每个都有 3 个元素	F_VN_ContainerAverageElementwise3 [▶ 953] F_VN_ContainerAverageVarianceElementwise3 [▶ 958]
向量的容器，每个都有 4 个元素	F_VN_ContainerAverageElementwise4 [▶ 954] F_VN_ContainerAverageVarianceElementwise4 [▶ 959]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.8 F_VN_ContainerAverageVariance



Calculate the average and variance of elements in a container.

语法

定义：

```
FUNCTION F_VN_ContainerAverageVariance : HRESULT
VAR_INPUT
    ipSrcContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fAverage       : LREAL;
    fVariance      : LREAL;
END_VAR
VAR_INPUT
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	源容器（支持算术、点和向量元素类型。对于点和向量，使用欧几里得范数）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fAverage	LREAL	返回计算出的平均值
fVariance	LREAL	返回计算出的方差值

返回值

HRESULT [▶ 135]

更多信息

下表显示了哪些功能可用于哪种容器类型：

容器类型	可用功能
元素的容器	F_VN_ContainerAverage [▶ 950] F_VN_ContainerAverageVariance [▶ 955]
向量的容器，每个都有 2 个元素	F_VN_ContainerAverageElementwise2 [▶ 951] F_VN_ContainerAverageVarianceElementwise2 [▶ 956]
向量的容器，每个都有 3 个元素	F_VN_ContainerAverageElementwise3 [▶ 953] F_VN_ContainerAverageVarianceElementwise3 [▶ 958]
向量的容器，每个都有 4 个元素	F_VN_ContainerAverageElementwise4 [▶ 954] F_VN_ContainerAverageVarianceElementwise4 [▶ 959]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.9 F_VN_ContainerAverageVarianceElementwise2



Calculate the elementwise average and variance of 2D elements in a container.

语法

定义：

```
FUNCTION F_VN_ContainerAverageVarianceElementwise2 : HRESULT
VAR_INPUT
    ipSrcContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aAverage       : TcVnVector2_LREAL;
    aVariance      : TcVnVector2_LREAL;
END_VAR
```

```
VAR_INPUT
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	带有二维点或向量的源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aAverage	TcVnVector2_LREAL [▶ 153]	返回计算出的平均值
aVariance	TcVnVector2_LREAL [▶ 153]	返回计算出的方差值

返回值

HRESULT [▶ 135]

更多信息

下表显示了哪些功能可用于哪种容器类型：

容器类型	可用功能
元素的容器	F_VN_ContainerAverage [▶ 950] F_VN_ContainerAverageVariance [▶ 955]
向量的容器，每个都有 2 个元素	F_VN_ContainerAverageElementwise2 [▶ 951] F_VN_ContainerAverageVarianceElementwise2 [▶ 956]
向量的容器，每个都有 3 个元素	F_VN_ContainerAverageElementwise3 [▶ 953] F_VN_ContainerAverageVarianceElementwise3 [▶ 958]
向量的容器，每个都有 4 个元素	F_VN_ContainerAverageElementwise4 [▶ 954] F_VN_ContainerAverageVarianceElementwise4 [▶ 959]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.10 F_VN_ContainerAverageVarianceElementwise3



Calculate the elementwise average and variance of 3D elements in a container.

语法

定义：

```
FUNCTION F_VN_ContainerAverageVarianceElementwise3 : HRESULT
VAR_INPUT
    ipSrcContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aAverage       : TcVnVector3_LREAL;
    aVariance      : TcVnVector3_LREAL;
END_VAR
VAR_INPUT
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	带有三维点或向量的源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aAverage	TcVnVector3_LREAL [▶ 153]	返回计算出的平均值
aVariance	TcVnVector3_LREAL [▶ 153]	返回计算出的方差值

返回值

HRESULT [▶ 135]

更多信息

下表显示了哪些功能可用于哪种容器类型：

容器类型	可用功能
元素的容器	F_VN_ContainerAverage [▶ 950] F_VN_ContainerAverageVariance [▶ 955]
向量的容器，每个都有 2 个元素	F_VN_ContainerAverageElementwise2 [▶ 951] F_VN_ContainerAverageVarianceElementwise2 [▶ 956]
向量的容器，每个都有 3 个元素	F_VN_ContainerAverageElementwise3 [▶ 953] F_VN_ContainerAverageVarianceElementwise3 [▶ 958]
向量的容器，每个都有 4 个元素	F_VN_ContainerAverageElementwise4 [▶ 954]

容器类型	可用功能
	F_VN_ContainerAverageVarianceElementwise4 [▶ 959]

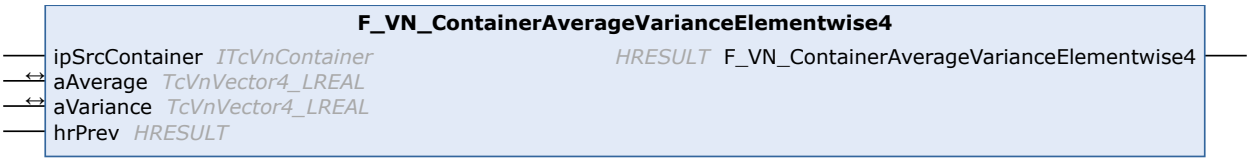
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.11 F_VN_ContainerAverageVarianceElementwise4



Calculate the elementwise average and variance of 4D elements in a container.

语法

定义：

```
FUNCTION F_VN_ContainerAverageVarianceElementwise4 : HRESULT
VAR_INPUT
    ipSrcContainer : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aAverage       : TcVnVector4_LREAL;
    aVariance      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	带有四维向量的源容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aAverage	TcVnVector4_LREAL [▶ 153]	返回计算出的平均值
aVariance	TcVnVector4_LREAL [▶ 153]	返回计算出的方差值

返回值

HRESULT [▶ 135]

更多信息

下表显示了哪些功能可用于哪种容器类型：

容器类型	可用功能
元素的容器	F_VN_ContainerAverage [▶ 950] F_VN_ContainerAverageVariance [▶ 955]
向量的容器，每个都有 2 个元素	F_VN_ContainerAverageElementwise2 [▶ 951] F_VN_ContainerAverageVarianceElementwise2 [▶ 956]
向量的容器，每个都有 3 个元素	F_VN_ContainerAverageElementwise3 [▶ 953] F_VN_ContainerAverageVarianceElementwise3 [▶ 958]
向量的容器，每个都有 4 个元素	F_VN_ContainerAverageElementwise4 [▶ 954] F_VN_ContainerAverageVarianceElementwise4 [▶ 959]

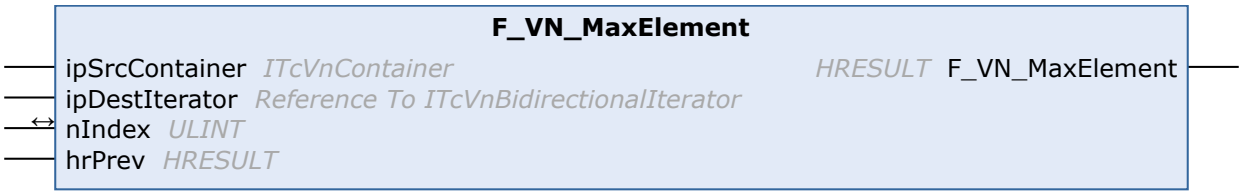
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.12 F_VN_MaxElement



Gets the maximum element of a container.

语法

定义：

```
FUNCTION F_VN_MaxElement : HRESULT
VAR_INPUT
    ipSrcContainer : ITcVnContainer;
    ipDestIterator : Reference To ITcVnBidirectionalIterator;
END_VAR
VAR_IN_OUT
    nIndex         : ULINT;
END_VAR
VAR_INPUT
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	源容器（支持算术、点和向量元素类型。对于点和向量，使用欧几里得范数）。

名称	类型	描述
ipDestIterator	引用 ITcVnBidirectionalIterator [▶ 354]	返回请求元素的迭代器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nIndex	ULINT	返回请求的元素在容器中的位置

返回值

HRESULT [▶ 135]

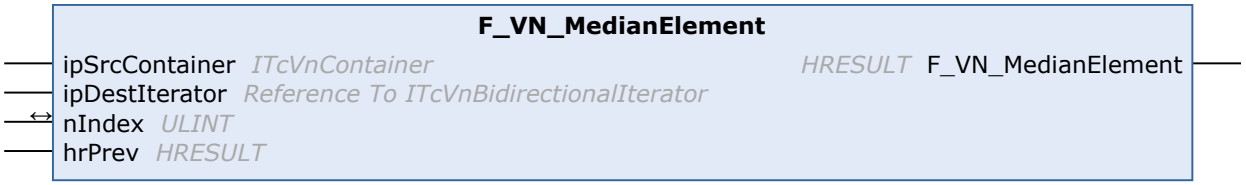
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.13 F_VN_MedianElement



Gets the median element of a container.

语法

定义：

```
FUNCTION F_VN_MedianElement : HRESULT
VAR_INPUT
    ipSrcContainer : ITcVnContainer;
    ipDestIterator : Reference To ITcVnBidirectionalIterator;
END_VAR
VAR_IN_OUT
    nIndex         : ULINT;
END_VAR
VAR_INPUT
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	源容器（支持算术、点和向量元素类型。对于点和向量，使用欧几里得范数）。
ipDestIterator	引用 ITcVnBidirectionalIterator [▶ 354]	返回请求元素的迭代器

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nIndex	ULINT	返回请求的元素在容器中的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.9.14 F_VN_MinElement



Gets the minimum element of a container.

语法

定义：

```
FUNCTION F_VN_MinElement : HRESULT
VAR_INPUT
    ipSrcContainer : ITcVnContainer;
    ipDestIterator : Reference To ITcVnBidirectionalIterator;
END_VAR
VAR_IN_OUT
    nIndex         : ULINT;
END_VAR
VAR_INPUT
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContainer	ITcVnContainer [▶ 365]	源容器（支持算术、点和向量元素类型。对于点和向量，使用欧几里得范数）。
ipDestIterator	引用 ITcVnBidirectionalIterator [▶ 354]	返回请求元素的迭代器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nIndex	ULINT	返回请求的元素在容器中的位置

返回值

HRESULT [▶ 135]

所需的许可证

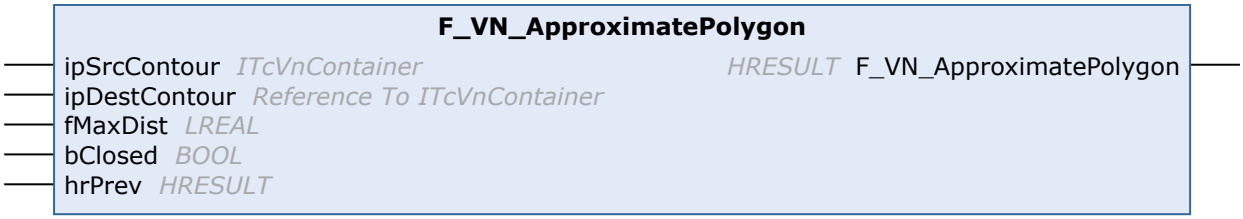
TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10 轮廓分析

6.1.4.10.1 F_VN_ApproximatePolygon



Approximate a contour to a simplified polygon (using the Douglas-Peucker algorithm).

语法

定义：

```
FUNCTION F_VN_ApproximatePolygon : HRESULT
VAR_INPUT
    ipSrcContour  : ITcVnContainer;
    ipDestContour : Reference To ITcVnContainer;
    fMaxDist      : LREAL;
    bClosed       : BOOL;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcContour	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
ipDestContour	引用 ITcVnContainer [▶ 365]	返回一个以近似多边形点填充的容器（与 ipSrcContour 的类型 ID 相同；重复使用非零接口指针。）
fMaxDist	LREAL	原始轮廓与近似值之间的最大距离
bClosed	BOOL	指定轮廓是否封闭（首尾点相连）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.2 F_VN_CheckIfPointIsInsideContour



Checks if a point is inside a contour (and optionally return the distance to it).

语法

定义：

```
FUNCTION F_VN_CheckIfPointIsInsideContour : HRESULT
VAR_INPUT
    ipContour         : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aPoint            : TcVnPoint2_REAL;
    fDist             : LREAL;
END_VAR
VAR_INPUT
    bMeasureDistance : BOOL;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL）
bMeasureDistance	BOOL	如果为 true，fDist 将返回到最近轮廓边缘的距离。否则，fDist 只返回 -1、0 或 +1（如果不需要距离，建议设置为 false，这样速度更快）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aPoint	TcVnPoint2_REAL [▶ 152]	待检查的点位置
fDist	LREAL	返回结果（> 0：点在轮廓内；0：点在轮廓上；< 0：点在轮廓外）

返回值

HRESULT [▶ 135]

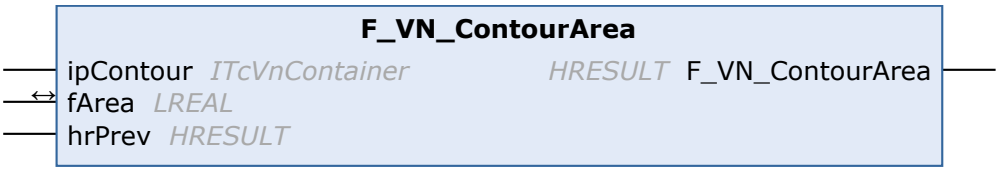
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.3 F_VN_ContourArea



Estimate the contour area using Green's formula.

语法

定义：

```
FUNCTION F_VN_ContourArea : HRESULT
VAR_INPUT
    ipContour : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fArea     : LREAL;
END_VAR
VAR_INPUT
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fArea	LREAL	返回估算的轮廓面积（实际轮廓面积可能因轮廓形状而异）。

返回值

HRESULT [▶ 135]

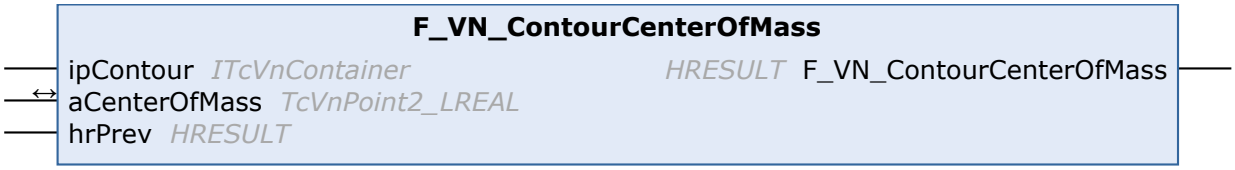
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.4 F_VN_ContourCenterOfMass



Computes the center of mass of a contour.

语法

定义：

```
FUNCTION F_VN_ContourCenterOfMass : HRESULT
VAR_INPUT
    ipContour     : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aCenterOfMass : TcVnPoint2_LREAL;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCenterOfMass	TcVnPoint2_LREAL [▶ 152]	返回轮廓的质心

返回值

HRESULT [▶ 135]

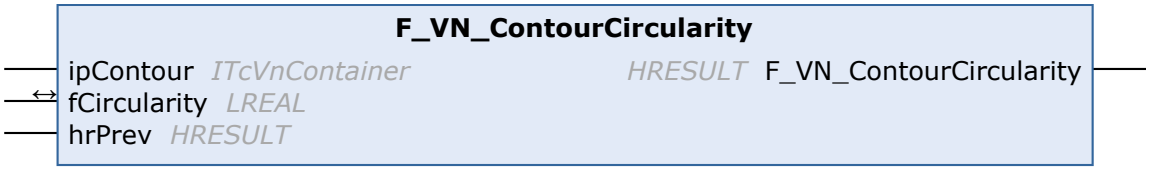
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.5 F_VN_ContourCircularity



Computes the circularity of a contour (1.0: ideal circle, 0.0: straight line).

语法

定义：

```
FUNCTION F_VN_ContourCircularity : HRESULT
VAR_INPUT
    ipContour    : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fCircularity : LREAL;
END_VAR
VAR_INPUT
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fCircularity	LREAL	返回轮廓的圆度 [0..1]

返回值

HRESULT [▶ 135]

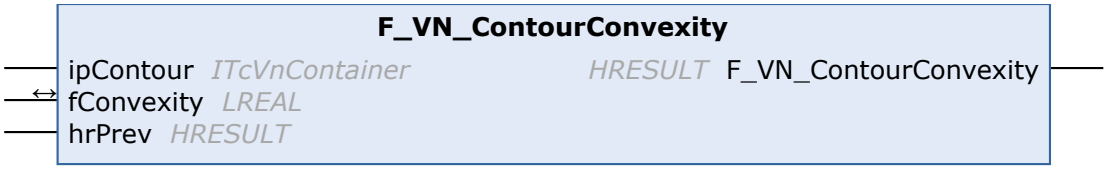
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.6 F_VN_ContourConvexity



Computes the convexity of a contour as the ratio between the contour's area and the area of the convex hull (1.0: convex shapes (i.e. rectangle, circle, ..., etc.), 0.0: zero area (i.e. a line or an empty contour)).

语法

定义：

```
FUNCTION F_VN_ContourConvexity : HRESULT
VAR_INPUT
    ipContour  : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fConvexity : LREAL;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fConvexity	LREAL	返回轮廓的凸度 [0..1]

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.7 F_VN_ContourEccentricity



Computes the eccentricity of a contour (0.0: circular, 1.0: linear).

语法

定义：

```
FUNCTION F_VN_ContourEccentricity : HRESULT
VAR_INPUT
    ipContour     : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fEccentricity : LREAL;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fEccentricity	LREAL	返回轮廓的偏心率 [0..1]

返回值

HRESULT [▶ 135]

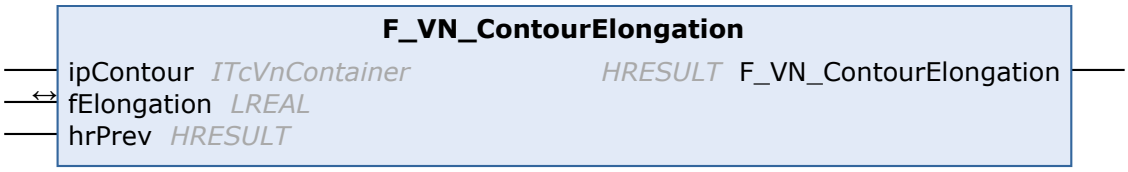
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.8 F_VN_ContourElongation



Computes the elongation factor of a contour.

语法

定义：

```
FUNCTION F_VN_ContourElongation : HRESULT
VAR_INPUT
    ipContour   : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fElongation : LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fElongation	LREAL	返回轮廓的伸长率

返回值

HRESULT [▶ 135]

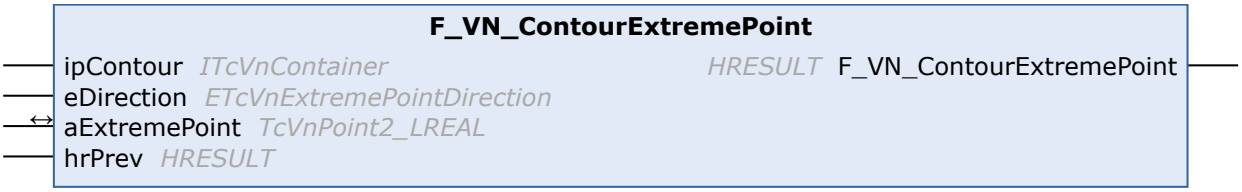
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.9 F_VN_ContourExtremePoint



Finds the contour extreme point.

语法

定义：

```
FUNCTION F_VN_ContourExtremePoint : HRESULT
VAR_INPUT
    ipContour     : ITcVnContainer;
    eDirection    : ETcVnExtremePointDirection;
END_VAR
VAR_IN_OUT
    aExtremePoint : TcVnPoint2_LREAL;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	包含二维点的源容器（TcVnPoint2_DINT 或 TcVnPoint2_REAL 或 TcVnPoint2_LREAL）

名称	类型	描述
eDirection	ETcVnExtremePointDirection [▶ 193]	选择极值点的搜索方向
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aExtremePoint	TcVnPoint2_LREAL [▶ 152]	根据 eDirection 返回轮廓的极值点

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.10 F_VN_ContourInertiaRatio



Computes the inertia ratio of a contour which reflects how a contour/shape is elongated (i.e. circle: 1.0, line: 0.0).

语法

定义：

```
FUNCTION F_VN_ContourInertiaRatio : HRESULT
VAR_INPUT
    ipContour     : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fInertiaRatio : LREAL;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fInertiaRatio	LREAL	返回轮廓的惯性比 [0..1]

返回值

HRESULT [▶ 135]

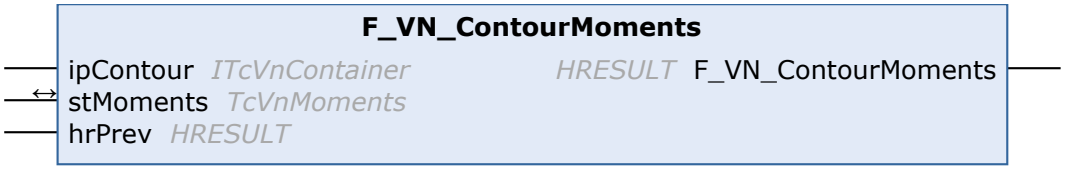
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.11 F_VN_ContourMoments



Computes the spatial moments, the central moments, and the central normalized moments of a contour up to the third order.

语法

定义：

```
FUNCTION F_VN_ContourMoments : HRESULT
VAR_INPUT
    ipContour : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stMoments : TcVnMoments;
END_VAR
VAR_INPUT
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stMoments	TcVnMoments [▶ 226]	返回包含矩的结构体

返回值

HRESULT [▶ 135]

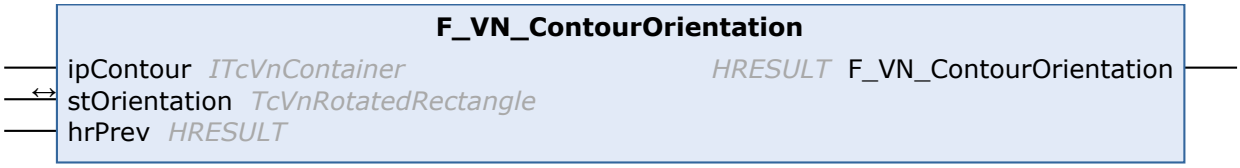
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.12 F_VN_ContourOrientation



Calculate the orientation of a contour or a set of points using TCVN_OM_PCA.

语法

定义：

```
FUNCTION F_VN_ContourOrientation : HRESULT
VAR_INPUT
    ipContour     : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stOrientation : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	源点集（ContainerType_Vector_TcVnPoint2_DINT、ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL。此容器的元素为各个点）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stOrientation	TcVnRotatedRectangle [▶ 237]	旋转后的矩形，包含中心点、轴长和顺时针方向的轮廓旋转角。

返回值

HRESULT [▶ 135]

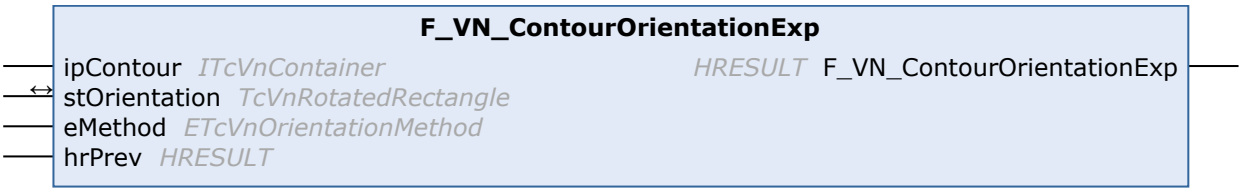
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.13 F_VN_ContourOrientationExp



Calculate the orientation of a contour or a set of points. (expert function)

语法

定义：

```
FUNCTION F_VN_ContourOrientationExp : HRESULT
VAR_INPUT
    ipContour     : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stOrientation : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    eMethod       : ETcVnOrientationMethod;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	源点集（ContainerType_Vector_TcVnPoint2_DINT、ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL。此容器的元素为各个点）。
eMethod	ETcVnOrientationMethod [▶ 205]	计算方向的方法。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stOrientation	TcVnRotatedRectangle [▶ 237]	旋转后的矩形，包含中心点、轴长和顺时针方向的轮廓旋转角（取决于所使用的计算方法）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.14 F_VN_ContourPerimeter



Computes the perimeter of a contour (curve length if the contour is not closed).

语法

定义：

```
FUNCTION F_VN_ContourPerimeter : HRESULT
VAR_INPUT
    ipContour  : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fPerimeter : LREAL;
END_VAR
VAR_INPUT
    bClosed    : BOOL;
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
bClosed	BOOL	指定轮廓是否封闭（首尾点相连）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fPerimeter	LREAL	返回轮廓的周长

返回值

HRESULT [▶ 135]

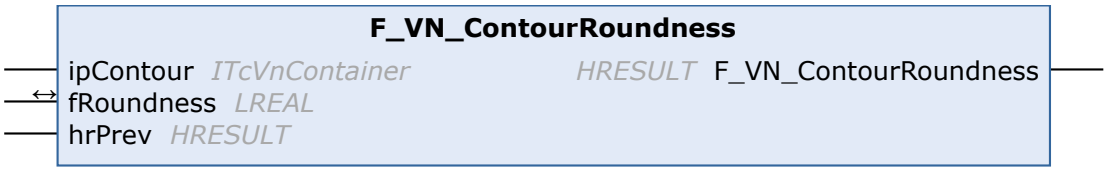
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.15 F_VN_ContourRoundness



Computes the roundness of a contour (perimeter^2 / area).

语法

定义：

```
FUNCTION F_VN_ContourRoundness : HRESULT
VAR_INPUT
    ipContour  : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fRoundness : LREAL;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fRoundness	LREAL	返回轮廓的圆度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.16 F_VN_ConvexHullPoints



Determines the convex hull of a point set.

语法

定义：

```
FUNCTION F_VN_ConvexHullPoints : HRESULT
VAR_INPUT
    ipPointSet    : ITcVnContainer;
    ipConvexHull : Reference To ITcVnContainer;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipPointSet	ITcVnContainer [▶ 365]	源点集（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为各个点。）
ipConvexHull	引用 ITcVnContainer [▶ 365]	返回一个以已排序凸包点填充的容器（与 ipPointSet 的类型 ID 相同；重复使用非零接口指针）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

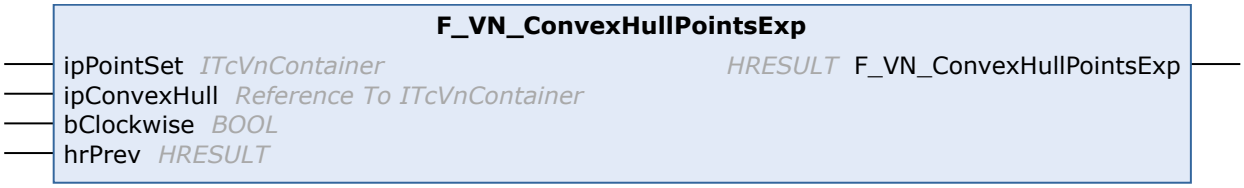
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.17 F_VN_ConvexHullPointsExp



Determines the convex hull of a point set. (expert function)

语法

定义：

```
FUNCTION F_VN_ConvexHullPointsExp : HRESULT
VAR_INPUT
    ipPointSet    : ITcVnContainer;
    ipConvexHull : Reference To ITcVnContainer;
    bClockwise    : BOOL;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipPointSet	ITcVnContainer [▶ 365]	源点集（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为各个点。）
ipConvexHull	引用 ITcVnContainer [▶ 365]	返回一个以已排序凸包点填充的容器（与 ipPointSet 的类型 ID 相同；重复使用非零接口指针）。
bClockwise	BOOL	假定为笛卡尔（Cartesian）坐标，选择凸包点的排序方向。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

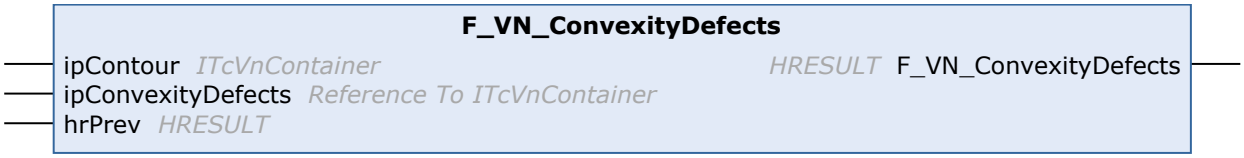
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.18 F_VN_ConvexityDefects



Finds the convexity defects of a point set.

语法

定义：

```
FUNCTION F_VN_ConvexityDefects : HRESULT
VAR_INPUT
    ipContour          : ITcVnContainer;
    ipConvexityDefects : Reference To ITcVnContainer;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	源轮廓（ContainerType_Vector_TcVnPoint2_DINT；此容器的元素为各个点。）
ipConvexityDefects	引用 ITcVnContainer [▶ 365]	返回一个以凸度缺陷填充的容器（ContainerType_Vector_TcVnVector4_DINT，包含 [startIndex、endIndex、furthestPointIndex、fixedPointDistance（8 个尾数位，即除以 256 得到以像素为单位的距离）]；重复使用非零接口指针）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

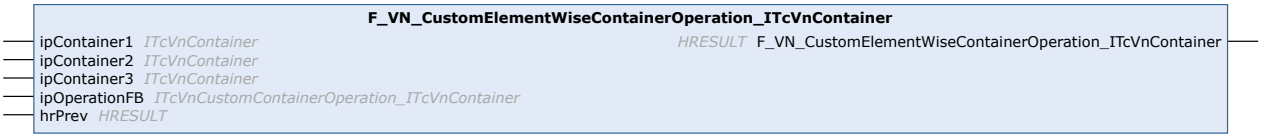
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.19 F_VN_CustomElementWiseContainerOperation_ITcVnContainer



Performs a custom, element wise operation on a set of up to 3 containers.

语法

定义：

```
FUNCTION F_VN_CustomElementWiseContainerOperation_ITcVnContainer : HRESULT
VAR_INPUT
    ipContainer1  : ITcVnContainer;
    ipContainer2  : ITcVnContainer;
    ipContainer3  : ITcVnContainer;
    ipOperationFB : ITcVnCustomContainerOperation_ITcVnContainer;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer1	ITcVnContainer [▶ 365]	容器 1
ipContainer2	ITcVnContainer [▶ 365]	容器 2
ipContainer3	ITcVnContainer [▶ 365]	容器 3
ipOperationFB	ITcVnCustomContainerOperation_ITcVnContainer [▶ 238]	对容器元素进行自定义运算

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

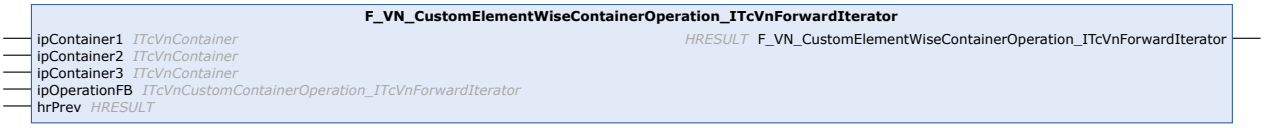
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.20 F_VN_CustomElementWiseContainerOperation_ITcVnForwardIterator



Performs a custom, element wise operation on a set of up to 3 containers.

语法

定义：

```
FUNCTION F_VN_CustomElementWiseContainerOperation_ITcVnForwardIterator : HRESULT
VAR_INPUT
    ipContainer1   : ITcVnContainer;
    ipContainer2   : ITcVnContainer;
    ipContainer3   : ITcVnContainer;
    ipOperationFB : ITcVnCustomContainerOperation_ITcVnForwardIterator;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContainer1	ITcVnContainer [▶ 365]	容器 1
ipContainer2	ITcVnContainer [▶ 365]	容器 2
ipContainer3	ITcVnContainer [▶ 365]	容器 3
ipOperationFB	ITcVnCustomContainerOperation_ITcVnForwardIterator [▶ 239]	对元素进行自定义运算
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

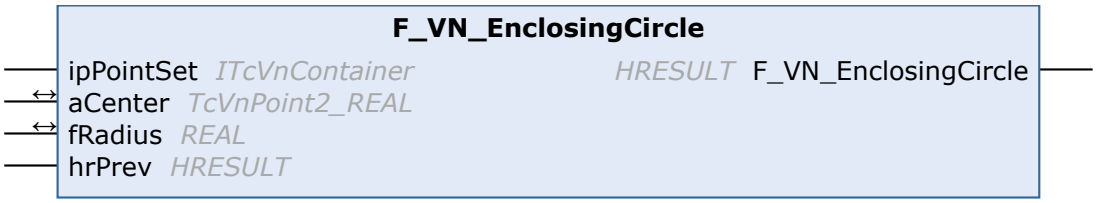
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.21 F_VN_EnclosingCircle



Searchs for a minimum area circle enclosing a set of points.

语法

定义：

```
FUNCTION F_VN_EnclosingCircle : HRESULT
VAR_INPUT
    ipPointSet : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aCenter    : TcVnPoint2_REAL;
    fRadius    : REAL;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipPointSet	ITcVnContainer [▶ 365]	源点集（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为各个点。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCenter	TcVnPoint2_REAL [▶ 152]	返回圆心
fRadius	REAL	返回圆的半径

返回值

HRESULT [▶ 135]

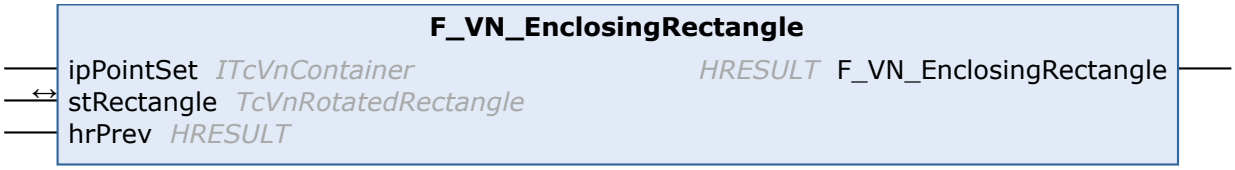
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.22 F_VN_EnclosingRectangle



Searches for a minimum area rectangle enclosing a set of points.

语法

定义：

```
FUNCTION F_VN_EnclosingRectangle : HRESULT
VAR_INPUT
    ipPointSet  : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stRectangle : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipPointSet	ITcVnContainer [▶ 365]	源点集（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为各个点。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stRectangle	TcVnRotatedRectangle [▶ 237]	返回测定的矩形

返回值

HRESULT [▶ 135]

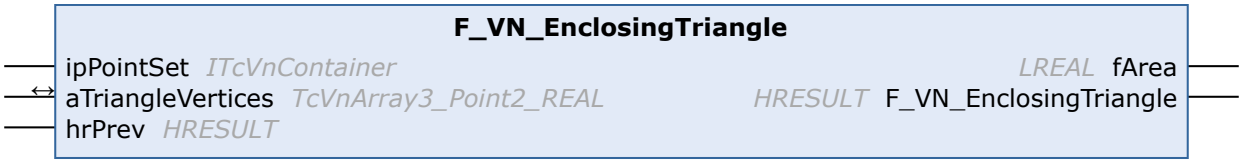
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.23 F_VN_EnclosingTriangle



Searches for a minimum area triangle enclosing a set of points.

语法

定义：

```
FUNCTION F_VN_EnclosingTriangle : HRESULT
VAR_INPUT
    ipPointSet        : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aTriangleVertices : TcVnArray3_Point2_REAL;
END_VAR
VAR_INPUT
    hrPrev            : HRESULT;
END_VAR
VAR_OUTPUT
    fArea             : LREAL;
END_VAR
```

输入

名称	类型	描述
ipPointSet	ITcVnContainer [▶ 365]	源点集（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为各个点。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTriangleVertices	TcVnArray3_Point2_REAL [▶ 153]	返回 3 个三角形顶点

输出

名称	类型	描述
fArea	LREAL	返回三角形面积

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.24 F_VN_FitEllipse



Fit an ellipse in a point set.

语法

定义：

```
FUNCTION F_VN_FitEllipse : HRESULT
VAR_INPUT
    ipPointSet : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stEllipse  : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipPointSet	ITcVnContainer [▶ 365]	源点集（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；至少需要 5 个合理的点！此容器的元素为各个点）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stEllipse	TcVnRotatedRectangle [▶ 237]	产生的椭圆，由旋转矩形描述。

返回值

HRESULT [▶ 135]

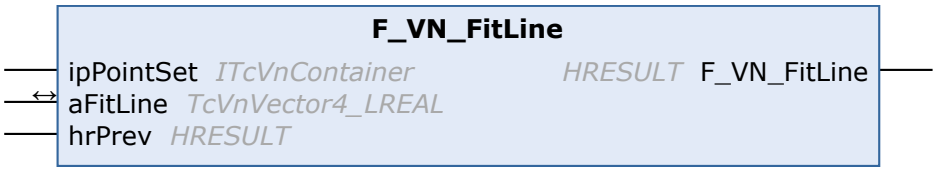
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.25 F_VN_FitLine



Fit a line into a point set.

语法

定义：

```
FUNCTION F_VN_FitLine : HRESULT
VAR_INPUT
    ipPointSet : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aFitLine   : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipPointSet	ITcVnContainer [▶ 365]	源点集（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为各个点。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aFitLine	TcVnVector4_LREAL [▶ 153]	产生的直线。第一个和第二个元素描述的是该直线共线向量的 x 和 y 分量。第三个和第四个元素描述该直线上一个点的 x 和 y 分量。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.26 F_VN_FitLineExp



Fit a line into a point set. (expert function)

语法

定义：

```
FUNCTION F_VN_FitLineExp : HRESULT
VAR_INPUT
    ipPointSet     : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aFitLine       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    eDistanceType : ETcVnDistanceType;
    fParam         : LREAL;
    fEpsRadius     : LREAL;
    fEpsAngle      : LREAL;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipPointSet	ITcVnContainer [▶ 365]	源点集（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为各个点。）
eDistanceType	ETcVnDistanceType [▶ 188]	距离计算方法（支持：L1、L2、L12、FAIR、WELSCH、HUBER）
fParam	LREAL	某个 eDistanceType 的数值参数（c）（应 >= 0）。如果为 0，则在内部选择一个最佳值。
fEpsRadius	LREAL	半径精度（直线与坐标原点的距离，应大于 0）。数值越小，精度越高。
fEpsAngle	LREAL	角度精度（应大于 0）。数值越小，精度越高。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aFitLine	TcVnVector4_LREAL [▶ 153]	产生的直线。第一个和第二个元素描述的是该直线共线向量的 x 和 y 分量。第三个和第四个元素描述该直线上一个点的 x 和 y 分量。

返回值

HRESULT [▶ 135]

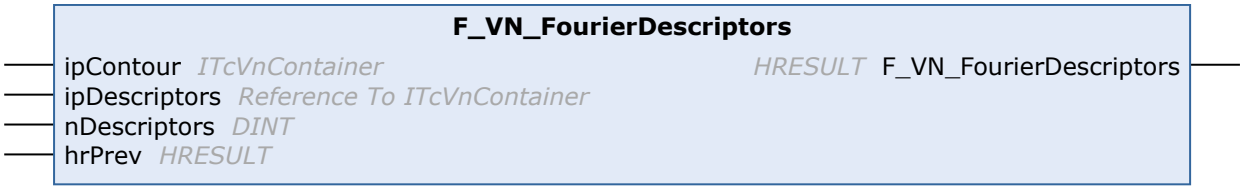
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.27 F_VN_FourierDescriptors



Computes the fourier descriptors for a closed contour.

语法

定义：

```
FUNCTION F_VN_FourierDescriptors : HRESULT
VAR_INPUT
    ipContour     : ITcVnContainer;
    ipDescriptors : Reference To ITcVnContainer;
    nDescriptors  : DINT;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	封闭轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL。强烈建议提供完整的轮廓，即轮廓检测算法使用 TCVN_CAM_NONE）。
ipDescriptors	引用 ITcVnContainer [▶ 365]	返回 ipContour 的傅立叶描述符（Re0、Im0、Re1、Im1、...；如果 ipContour 属于 ContainerType_Vector_LREAL 类型，则返回 ContainerType_Vector_TcVnPoint2_LREAL，否则返回 ContainerType_Vector_REAL。）
nDescriptors	DINT	指定返回的描述符数量（设为 -1 则返回所有算得的描述符，即至少与轮廓点的数量一样多）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.28 F_VN_MatchContours



Compare contours using the Hu moment invariants. In case of multiple contours in each container, the best matches are found and the average dissimilarity over all matched contours is returned.

语法

定义：

```
FUNCTION F_VN_MatchContours : HRESULT
VAR_INPUT
    ipContour1        : ITcVnContainer;
    ipContour2        : ITcVnContainer;
    eComparisonMethod : ETcVnContoursMatchComparisonMethod;
END_VAR
VAR_IN_OUT
    fDissimilarity    : LREAL;
END_VAR
VAR_INPUT
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour1	ITcVnContainer [▶ 365]	第一个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；该容器的元素为轮廓的点）或多条轮廓线的集合（ContainerType_Vector_Vector_TcVnPoint2_DINT）
ipContour2	ITcVnContainer [▶ 365]	第二个轮廓（与 ipContour1 类型相同）
eComparisonMethod	ETcVnContoursMatchComparisonMethod [▶ 184]	用于比较轮廓 Hu 不变矩的方法
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fDissimilarity	LREAL	根据所选的比较方法，返回轮廓的相异度

返回值

HRESULT [▶ 135]

示例

- 匹配轮廓 1vsN（手动形状） [▶ 2747]
- 匹配轮廓（提取的形状） [▶ 2750]

相关函数

- F_VN_ContourMoments [▶ 972]
- F_VN_HuMomentInvariants [▶ 1523]

- F_VN_MatchContours [▶ 988]
- F_VN_MatchContours1vsN [▶ 989]

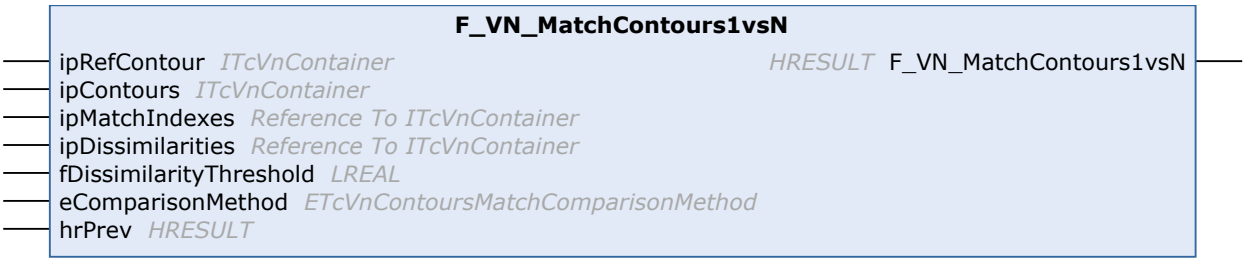
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.29 F_VN_MatchContours1vsN



Compare a reference contour with multiple other contours using the Hu moment invariants. Returns a sorted list of best matches.

语法

定义：

```
FUNCTION F_VN_MatchContours1vsN : HRESULT
VAR_INPUT
    ipRefContour            : ITcVnContainer;
    ipContours              : ITcVnContainer;
    ipMatchIndexes          : Reference To ITcVnContainer;
    ipDissimilarities       : Reference To ITcVnContainer;
    fDissimilarityThreshold : LREAL;
    eComparisonMethod       : ETcVnContoursMatchComparisonMethod;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipRefContour	ITcVnContainer [▶ 365]	参考轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓上的点）
ipContours	ITcVnContainer [▶ 365]	多个轮廓的集合（CTcVnContainer_Vector_Vector_TcVnPoint2_DINT）
ipMatchIndexes	引用 ITcVnContainer [▶ 365]	返回最佳匹配项的索引（CTcVnContainer_Vector_ULINT；已排序，第一个元素为最佳匹配项，即相异度最低的元素）
ipDissimilarities	引用 ITcVnContainer [▶ 365]	返回算得的最佳匹配项的相似度（CTcVnContainer_Vector_LREAL；可选，如果不需要，则设为 0；根据 ipMatchIndexes 排序）
fDissimilarityThreshold	LREAL	忽略不相关的匹配，即相异度 > fDissimilarityThreshold
eComparisonMethod	ETcVnContoursMatchComparisonMethod [▶ 184]	用于比较轮廓 Hu 不变矩的方法

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

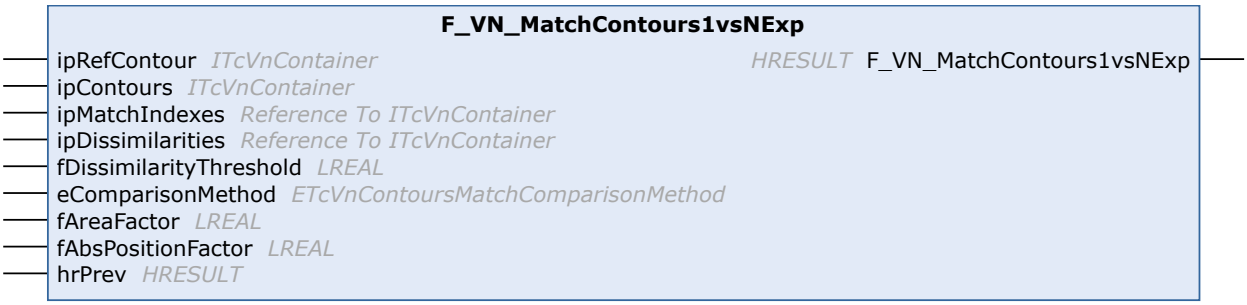
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.30 F_VN_MatchContours1vsNExp



Compare a reference contour with multiple other contours using the Hu moment invariants (and optionally further aspects). Returns a sorted list of best matches. (expert function)

语法

定义：

```
FUNCTION F_VN_MatchContours1vsNExp : HRESULT
VAR_INPUT
    ipRefContour            : ITcVnContainer;
    ipContours              : ITcVnContainer;
    ipMatchIndexes          : Reference To ITcVnContainer;
    ipDissimilarities       : Reference To ITcVnContainer;
    fDissimilarityThreshold : LREAL;
    eComparisonMethod       : ETcVnContoursMatchComparisonMethod;
    fAreaFactor             : LREAL;
    fAbsPositionFactor      : LREAL;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipRefContour	ITcVnContainer [▶ 365]	参考轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓上的点）
ipContours	ITcVnContainer [▶ 365]	多个轮廓的集合（CTcVnContainer_Vector_Vector_TcVnPoint2_DINT）
ipMatchIndexes	引用 ITcVnContainer [▶ 365]	返回最佳匹配项的索引（CTcVnContainer_Vector_ULINT；已排序，第一个元素为最佳匹配项，即相异度最低的元素）

名称	类型	描述
ipDissimilarities	引用 ITcVnContainer [▶ 365]	返回算得的最佳匹配项的相似度（CTcVnContainer_Vector_LREAL；可选，如果不需要，则设为 0；根据 ipMatchIndexes 排序）
fDissimilarityThreshold	LREAL	忽略不相关的匹配，即相异度 > fDissimilarityThreshold
eComparisonMethod	ETcVnContoursMatchComparisonMethod [▶ 184]	用于比较轮廓 Hu 不变矩的方法
fAreaFactor	LREAL	如果 > 0，则轮廓之间的相对面积差将按此系数缩放，并与算得的相异度相加
fAbsPositionFactor	LREAL	如果 > 0，则轮廓之间的绝对位置差（即几何轮廓中心的坐标）按此系数缩放，并与算得的相异度相加
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

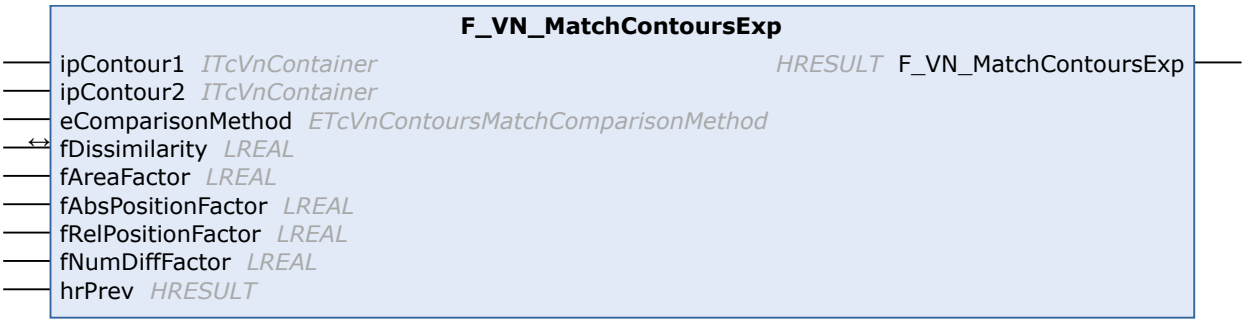
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.31 F_VN_MatchContoursExp



Compare contours using the Hu moment invariants (and optionally further aspects). In case of multiple contours in each container, the best matches are found and the average dissimilarity over all matched contours is returned. (expert function)

语法

定义：

```
FUNCTION F_VN_MatchContoursExp : HRESULT
VAR_INPUT
    ipContour1          : ITcVnContainer;
    ipContour2          : ITcVnContainer;
    eComparisonMethod   : ETcVnContoursMatchComparisonMethod;
END_VAR
VAR_IN_OUT
    fDissimilarity      : LREAL;
END_VAR
VAR_INPUT
    fAreaFactor         : LREAL;
```

```
    fAbsPositionFactor : LREAL;
    fRelPositionFactor : LREAL;
    fNumDiffFactor     : LREAL;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour1	ITcVnContainer [▶ 365]	第一个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；该容器的元素为轮廓的点）或多条轮廓线的集合（ContainerType_Vector_Vector_TcVnPoint2_DINT）
ipContour2	ITcVnContainer [▶ 365]	第二个轮廓（与 ipContour1 类型相同）
eComparisonMethod	ETcVnContoursMatchComparisonMethod [▶ 184]	用于比较轮廓 Hu 不变矩的方法
fAreaFactor	LREAL	如果 > 0，则轮廓之间的相对面积差将按此系数缩放，并与算得的相异度相加
fAbsPositionFactor	LREAL	如果 > 0，则轮廓之间的绝对位置差（即几何轮廓中心的坐标）按此系数缩放，并与算得的相异度相加
fRelPositionFactor	LREAL	如果 > 0，则轮廓之间的相对位置差将按此系数缩放，并与算得的相异度相加（仅在匹配多个轮廓时使用）
fNumDiffFactor	LREAL	如果 > 0，则两个容器中的轮廓数量之差将按此系数缩放，并与算得的相异度相加（仅在匹配多个轮廓时使用）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fDissimilarity	LREAL	根据所选的比较方法，返回轮廓的相异度

返回值

HRESULT [▶ 135]

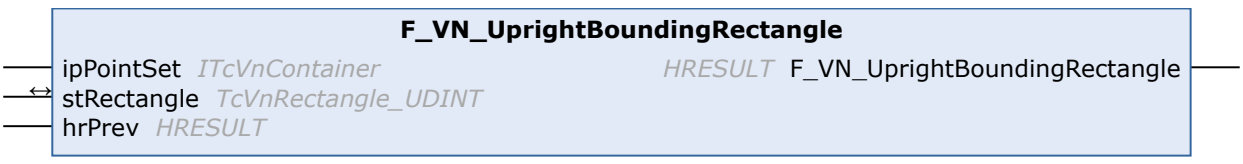
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.32 F_VN_UprightBoundingRectangle



Determines the upright bounding rectangle of a set of points.

语法

定义：

```
FUNCTION F_VN_UprightBoundingRectangle : HRESULT
VAR_INPUT
    ipPointSet  : ITcVnContainer;
END_VAR
VAR_IN_OUT
    stRectangle : TcVnRectangle_UDINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipPointSet	ITcVnContainer [▶ 365]	源点集（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为各个点。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stRectangle	TcVnRectangle_UDINT [▶ 236]	返回测定的矩形

返回值

HRESULT [▶ 135]

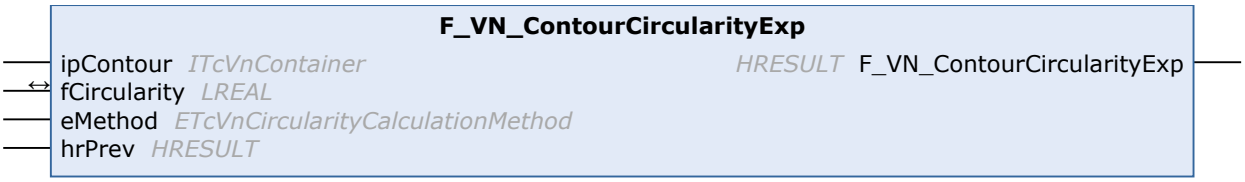
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.10.33 F_VN_ContourCircularityExp



计算轮廓的圆度（1.0：理想圆，0.0：直线）。（专家函数）

语法

定义：

```
FUNCTION F_VN_ContourCircularityExp : HRESULT
VAR_INPUT
    ipContour    : ITcVnContainer;
END_VAR
VAR_IN_OUT
```

```
    fCircularity : LREAL;
END_VAR
VAR_INPUT
    eMethod      : ETcVnCircularityCalculationMethod;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContour	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
eMethod	ETcVnCircularityCalculationMethod [▶ 163]	用于计算圆度的方法
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fCircularity	LREAL	返回轮廓的圆度 [0..1]

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.11 控制函数

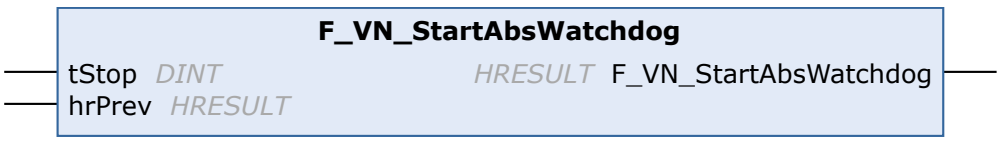
该组包含用于顺序控制的函数，例如用于处理看门狗 [▶ 140]。

函数

看门狗

- F_VN_StartAbsWatchdog(Exp) [▶ 995]
- F_VN_StartRelWatchdog(Exp) [▶ 997]
- F_VN_StopWatchdog [▶ 998]

6.1.4.11.1 F_VN_StartAbsWatchdog



Starts a cooperative watchdog given an absolute stop time.

语法

定义：

```
FUNCTION F_VN_StartAbsWatchdog : HRESULT
VAR_INPUT
    tStop  : DINT;
    hrPrev : HRESULT;
END_VAR
```

输入

名称	类型	描述
tStop	DINT	停止时间，单位：us
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

关于更多信息，可查看看门狗 [▶ 140]一章。

超过 tStop 时间

请注意，当看门狗被触发时，tStop 时间会稍微超过（几微秒）。

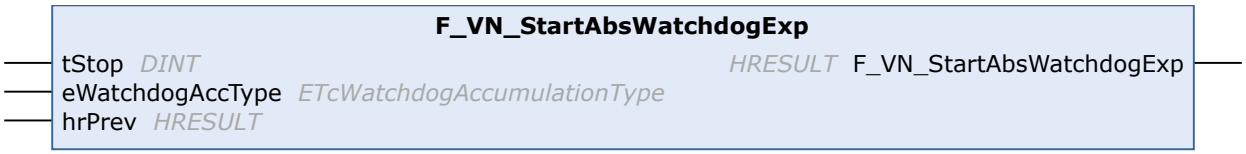
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.11.2 F_VN_StartAbsWatchdogExp



Start a cooperative watchdog given an absolute stop time. (expert function)

语法

定义：

```
FUNCTION F_VN_StartAbsWatchdogExp : HRESULT
VAR_INPUT
    tStop            : DINT;
    eWatchdogAccType : ETcWatchdogAccumulationType;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
tStop	DINT	停止时间，单位：us
eWatchdogAccType	ETcWatchdogAccumulationType [▶ 216]	用于合并该看门狗所包含的多个函数结果的累加方法
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

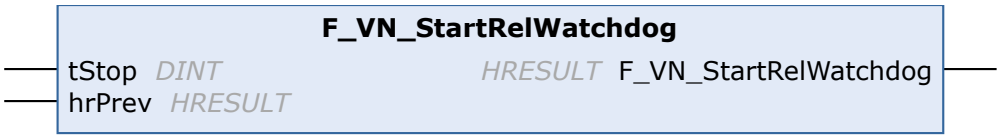
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.11.3 F_VN_StartRelWatchdog



Starts a cooperative watchdog given a stop time relative to the current time.

语法

定义：

```
FUNCTION F_VN_StartRelWatchdog : HRESULT
VAR_INPUT
    tStop  : DINT;
    hrPrev : HRESULT;
END_VAR
```

输入

名称	类型	描述
tStop	DINT	停止时间，单位：us
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

关于更多信息，可查看看门狗 [▶ 140]一章。

超过 tStop 时间

请注意，当看门狗被触发时，tStop 时间会稍微超过（几微秒）。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.11.4 F_VN_StartRelWatchdogExp



Start a cooperative watchdog given a stop time relative to the current time. (expert function)

语法

定义：

```
FUNCTION F_VN_StartRelWatchdogExp : HRESULT
VAR_INPUT
    tStop            : DINT;
    eWatchdogAccType : ETcWatchdogAccumulationType;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
tStop	DINT	停止时间，单位：us
eWatchdogAccType	ETcWatchdogAccumulationType [▶ 216]	用于合并该看门狗所包含的多个函数结果的累加方法
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.11.5 F_VN_StopWatchdog



Stops a watchdog and provide runtime information.

语法

定义：

```
FUNCTION F_VN_StopWatchdog : HRESULT
VAR_INPUT
    hrStartWatchdog     : HRESULT;
END_VAR
VAR_OUTPUT
    nFunctionsMonitored : ULINT;
    nFractionProcessed  : UDINT;
    tRest               : DINT;
END_VAR
```

输入

名称	类型	描述
hrStartWatchdog	HRESULT [▶ 135]	HRESULT 表示用于启动看门狗的函数的结果（如果 SUCCEEDED（hrStartWatchdog）等于 false，则不执行任何操作。）

输出

名称	类型	描述
nFunctionsMonitored	ULINT	返回受监控的函数数量
nFractionProcessed	UDINT	返回被监控函数的累计处理进度（以百分比为单位）
tRest	DINT	返回以 us 为单位的剩余计算时间（可能为负值）

返回值

HRESULT [▶ 135]

更多信息

输出 nFractionProcessed 必须根据看门狗的参数化进行不同的解释。这在启动看门狗时通过 ETcWatchdogAccumulationType [▶ 216]设置。

关于更多信息，可查看看门狗 [▶ 140]。

所需的许可证

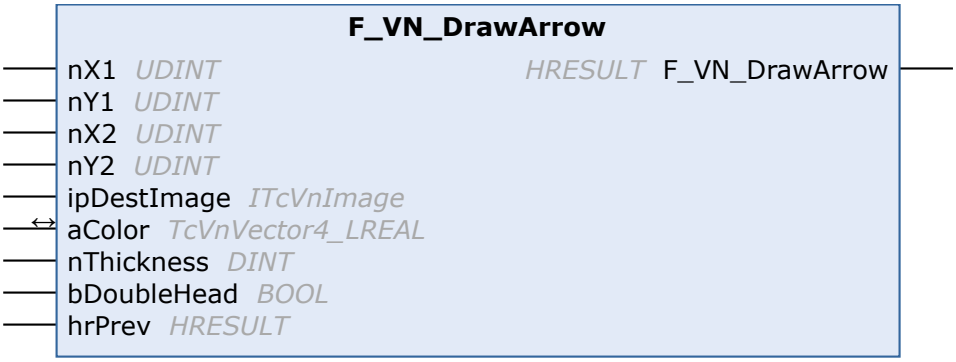
TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12 绘图

6.1.4.12.1 F_VN_DrawArrow



Draw an arrowed line.

语法

定义：

```
FUNCTION F_VN_DrawArrow : HRESULT
VAR_INPUT
    nX1          : UDINT;
    nY1          : UDINT;
    nX2          : UDINT;
    nY2          : UDINT;
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness   : DINT;
    bDoubleHead : BOOL;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
nX1	UDINT	起点的 x 坐标
nY1	UDINT	起点的 y 坐标
nX2	UDINT	终点的 x 坐标
nY2	UDINT	终点的 y 坐标
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细
bDoubleHead	BOOL	在直线两侧各画一个箭头。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

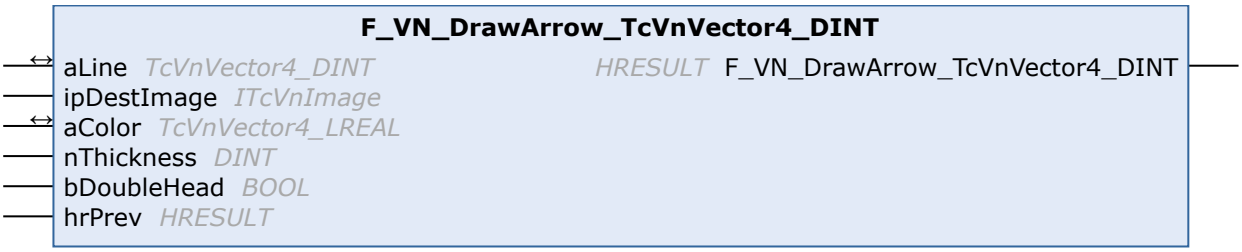
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.2 F_VN_DrawArrow_TcVnVector4_DINT



Draw an arrowed line.

语法

定义：

```
FUNCTION F_VN_DrawArrow_TcVnVector4_DINT : HRESULT
VAR_IN_OUT
    aLine       : TcVnVector4_DINT;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    bDoubleHead : BOOL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细
bDoubleHead	BOOL	在直线两侧各画一个箭头。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aLine	TcVnVector4_DINT [▶ 153]	线段 [x1，y1，x2，y2] 的起点和终点
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

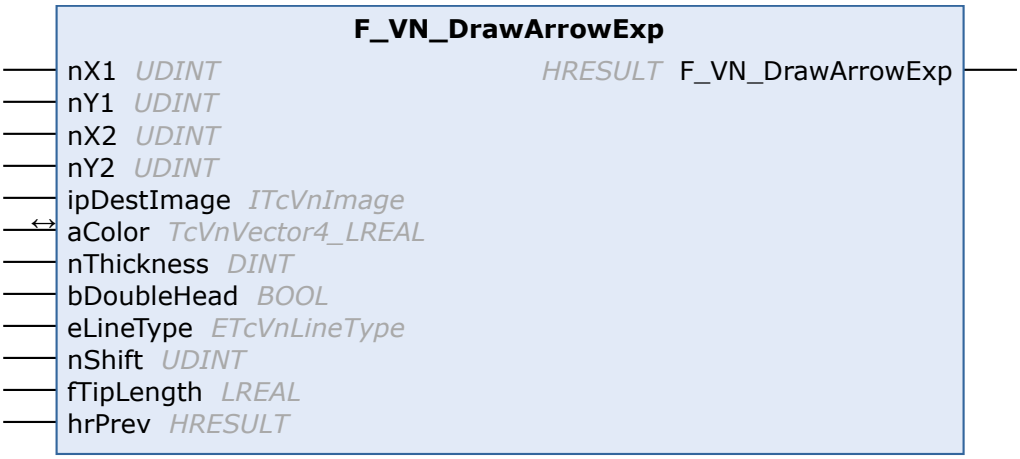
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.3 F_VN_DrawArrowExp



Draws an arrowed line. (expert function)

Syntax

Definition:

```
FUNCTION F_VN_DrawArrowExp : HRESULT
VAR_INPUT
    nX1          : UDINT;
    nY1          : UDINT;
    nX2          : UDINT;
    nY2          : UDINT;
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness   : DINT;
    bDoubleHead : BOOL;
    eLineType    : ETcVnLineType;
    nShift       : UDINT;
    fTipLength   : LREAL;
    hrPrev       : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
nX1	UDINT	x coordinate of the start point
nY1	UDINT	y coordinate of the start point
nX2	UDINT	x coordinate of the end point
nY2	UDINT	y coordinate of the end point

Name	Type	Description
ipDestImage	ITcVnImage [▶ 407]	Destination image
nThickness	DINT	Line thickness
bDoubleHead	BOOL	To draw an arrow on each side of the line.
eLineType	ETcVnLineType [▶ 201]	Line type
nShift	UDINT	Fractional bits of the coordinates (bit shift)
fTipLength	LREAL	Length of the arrow's tip as a fraction of the line's length
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

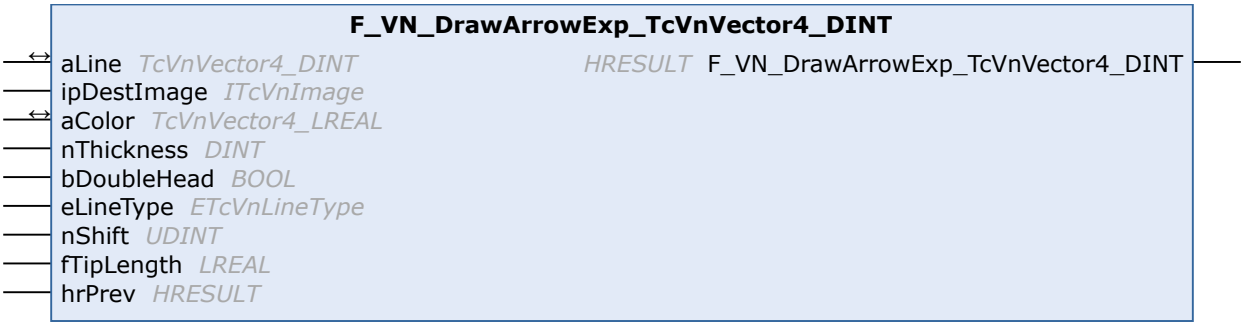
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.4 F_VN_DrawArrowExp_TcVnVector4_DINT



Draws an arrowed line. (expert function)

Syntax

Definition:

```
FUNCTION F_VN_DrawArrowExp_TcVnVector4_DINT : HRESULT
VAR_IN_OUT
    aLine       : TcVnVector4_DINT;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
```

```
    bDoubleHead : BOOL;
    eLineType   : ETcVnLineType;
    nShift      : UDINT;
    fTipLength  : LREAL;
    hrPrev      : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipDestImage	ITcVnImage [▶ 407]	Destination image
nThickness	DINT	Line thickness
bDoubleHead	BOOL	To draw an arrow on each side of the line.
eLineType	ETcVnLineType [▶ 201]	Line type
nShift	UDINT	Fractional bits of the coordinates (bit shift)
fTipLength	LREAL	Length of the arrow's tip as a fraction of the line's length
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

输入/输出

名称	类型	描述
aLine	TcVnVector4_DINT [▶ 153]	线段 [x1，y1，x2，y2] 的起点和终点
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

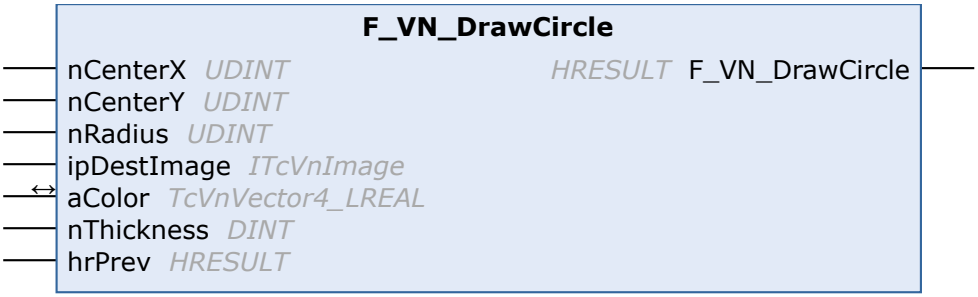
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.5 F_VN_DrawCircle



Draws a circle.

语法

定义：

```
FUNCTION F_VN_DrawCircle : HRESULT
VAR_INPUT
    nCenterX    : UDINT;
    nCenterY    : UDINT;
    nRadius     : UDINT;
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nCenterX	UDINT	中心的 x 坐标
nCenterY	UDINT	中心的 y 坐标
nRadius	UDINT	半径
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充圆形）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

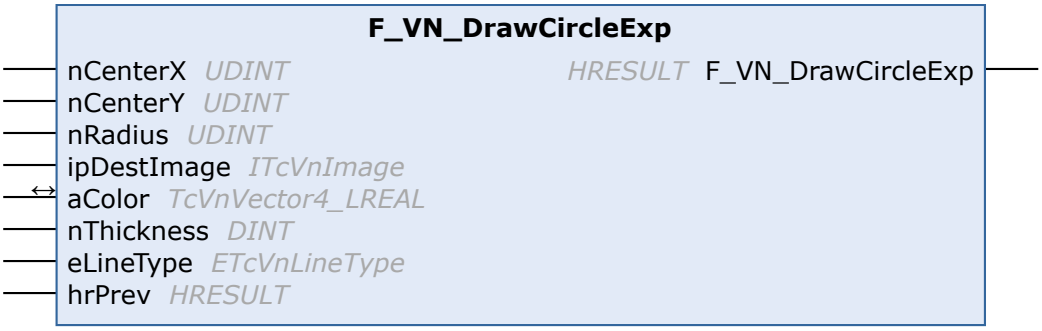
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.6 F_VN_DrawCircleExp



Draws a circle.

语法

定义：

```
FUNCTION F_VN_DrawCircleExp : HRESULT
VAR_INPUT
    nCenterX    : UDINT;
    nCenterY    : UDINT;
    nRadius     : UDINT;
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    eLineType   : ETcVnLineType;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nCenterX	UDINT	中心的 x 坐标
nCenterY	UDINT	中心的 y 坐标
nRadius	UDINT	半径
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充圆形）
eLineType	ETcVnLineType [▶ 201]	线条类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

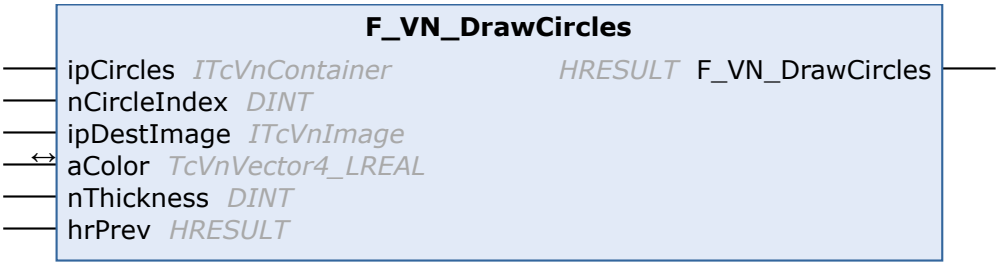
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.7 F_VN_DrawCircles



Draws circles.

语法

定义：

```
FUNCTION F_VN_DrawCircles : HRESULT
VAR_INPUT
    ipCircles    : ITcVnContainer;
    nCircleIndex : DINT;
    ipDestImage  : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness   : DINT;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipCircles	ITcVnContainer [▶ 365]	包含圆的容器（ContainerType_Vector_TcVnVector3_REAL；每个容器元素都包含圆心的 x 坐标 [0]、圆心的 y 坐标 [1] 和半径 [2]。）
nCircleIndex	DINT	要绘制的特定圆的索引（如果为负值，则绘制容器内的所有圆）
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充圆形）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

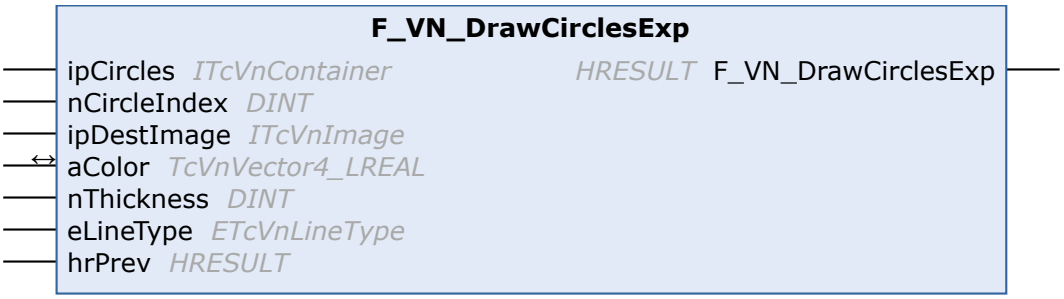
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.8 F_VN_DrawCirclesExp



Draws circles. (expert function)

语法

定义：

```
FUNCTION F_VN_DrawCirclesExp : HRESULT
VAR_INPUT
    ipCircles    : ITcVnContainer;
    nCircleIndex : DINT;
    ipDestImage  : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness   : DINT;
    eLineType    : ETcVnLineType;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipCircles	ITcVnContainer [▶ 365]	包含圆的容器（ContainerType_Vector_TcVnVector3_REAL；每个容器元素都包含圆心的 x 坐标 [0]、圆心的 y 坐标 [1] 和半径 [2]。）
nCircleIndex	DINT	要绘制的特定圆的索引（如果为负值，则绘制容器内的所有圆）
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充圆形）
eLineType	ETcVnLineType [▶ 201]	线条类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

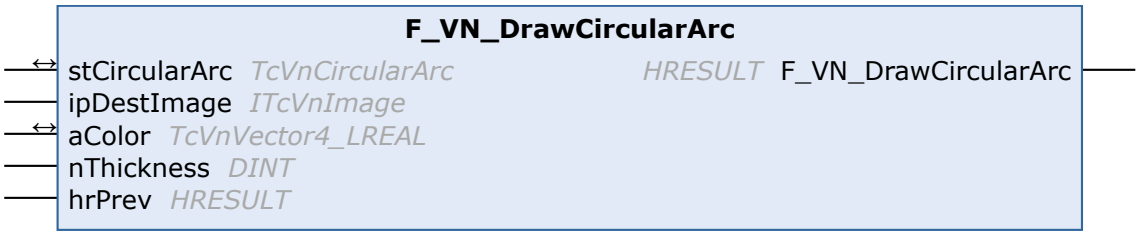
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.9 F_VN_DrawCircularArc



Draws a circular arc.

语法

定义：

```
FUNCTION F_VN_DrawCircularArc : HRESULT
VAR_IN_OUT
    stCircularArc : TcVnCircularArc;
END_VAR
VAR_INPUT
    ipDestImage   : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor        : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness    : DINT;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充圆弧）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stCircularArc	TcVnCircularArc [▶ 223]	圆弧定义
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

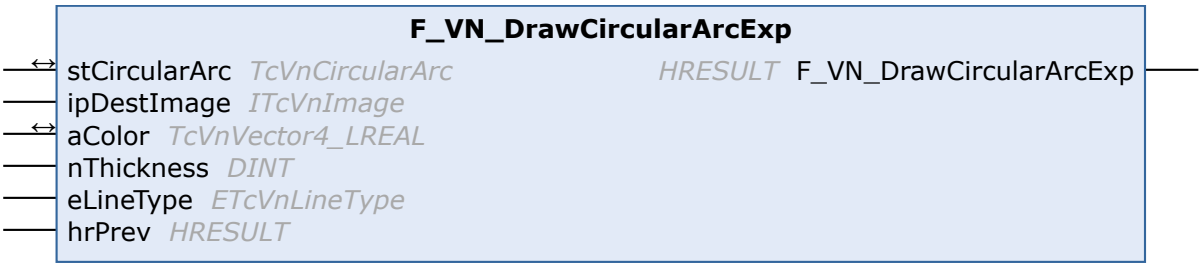
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.10 F_VN_DrawCircularArcExp



Draws a circular arc. (expert function)

语法

定义：

```
FUNCTION F_VN_DrawCircularArcExp : HRESULT
VAR_IN_OUT
    stCircularArc : TcVnCircularArc;
END_VAR
VAR_INPUT
    ipDestImage   : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor        : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness    : DINT;
    eLineType     : ETcVnLineType;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充圆弧）
eLineType	ETcVnLineType [▶ 201]	线条类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stCircularArc	TcVnCircularArc [▶ 223]	圆弧定义
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

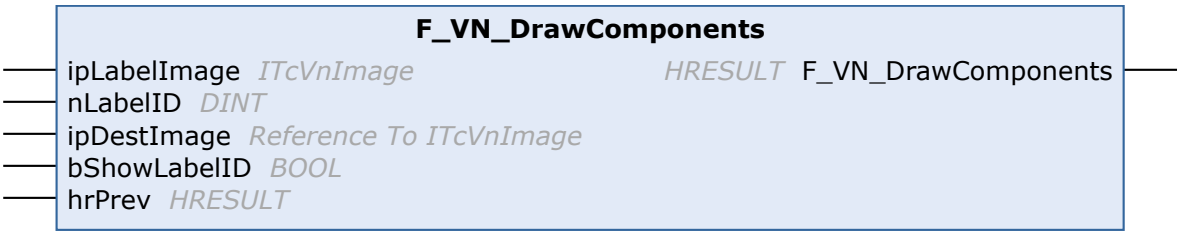
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.11 F_VN_DrawComponents



Draw the results of connected component function.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_DrawComponents : HRESULT
VAR_INPUT
    ipLabelImage : ITcVnImage;
    nLabelID     : DINT;
    ipDestImage  : Reference To ITcVnImage;
    bShowLabelID : BOOL;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipLabelImage	ITcVnImage [▶ 407]	图像包含每个像素的标签（1 通道 DINT 或 UINT）。
nLabelID	DINT	要绘制的特定标签的值（如果为负值，则绘制图像中的所有分量）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。输出图像为 1 或 3 通道 USINT）。
bShowLabelID	BOOL	在目标图像的分量中心显示标签 ID
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

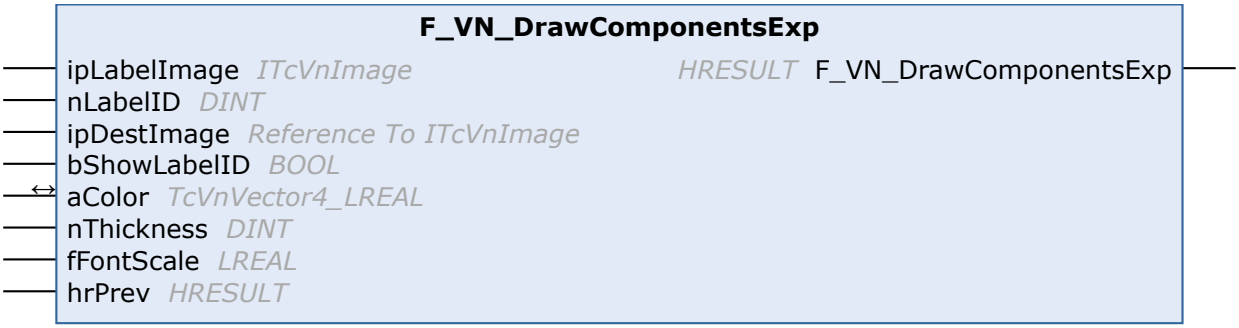
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.12 F_VN_DrawComponentsExp



Draw the results of connected component function. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_DrawComponentsExp : HRESULT
VAR_INPUT
    ipLabelImage : ITcVnImage;
    nLabelID     : DINT;
    ipDestImage  : Reference To ITcVnImage;
    bShowLabelID : BOOL;
END_VAR
VAR_IN_OUT
    aColor       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness   : DINT;
    fFontScale   : LREAL;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipLabelImage	ITcVnImage [▶ 407]	图像包含每个像素的标签（1 通道 DINT 或 UINT）。
nLabelID	DINT	要绘制的特定标签的值（如果为负值，则绘制图像中的所有分量）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。输出图像为 1 或 3 通道 USINT）。
bShowLabelID	BOOL	在目标图像的分量中心显示标签 ID
nThickness	DINT	文本线条粗细
fFontScale	LREAL	文本缩放系数
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	绘制分量的颜色（对于 {-1、-1、-1、-1}，将为每个分量随机选择一种颜色）

返回值

HRESULT [▶ 135]

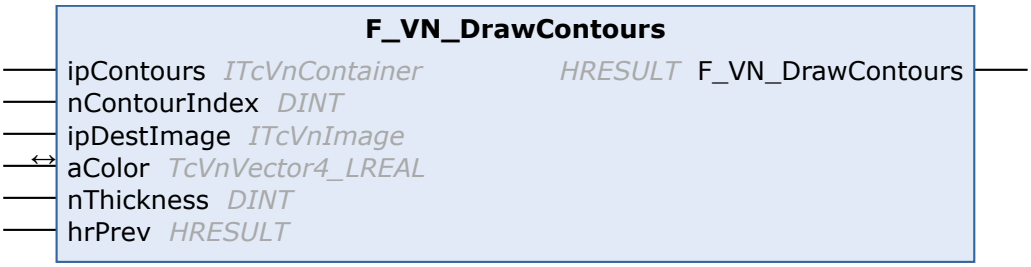
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.13 F_VN_DrawContours



Draws a single point set or multiple point sets that are interpreted as contours.

语法

定义：

```
FUNCTION F_VN_DrawContours : HRESULT
VAR_INPUT
    ipContours    : ITcVnContainer;
    nContourIndex : DINT;
    ipDestImage   : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor        : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness    : DINT;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContours	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT）或多个轮廓（ContainerType_Vector_Vector_TcVnPoint2_DINT）
nContourIndex	DINT	要绘制的特定轮廓的索引（如果为负值，则绘制容器内的所有轮廓）
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充轮廓）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_DrawContours在目标图像中绘制轮廓。一个轮廓的描述基于存储在容器中的一组点。请注意，只能绘制整数的轮廓点，因此，容器必须是类型ContainerType_Vector_TcVnPoint2_DINT。如果使用浮点数描述点，F_VN_ConvertContainerType [▶ 747]可以用来转换点。

除了具有一个轮廓的容器外，具有几个轮廓的容器也可以转移到函数中。其类型为ContainerType_Vector_Vector_TcVnPoint2_DINT。输入变量nContourIndex表示要画哪条轮廓。如果需要绘制所有的轮廓，nContourIndex必须设置为负值。一次性绘制整个容器的轮廓（将nContourIndex设置为负值）比在一个循环中重复调用该函数一个接一个地绘制更有效率。

专家参数

专家版F_VN_DrawContoursExp [▶ 1014]包含额外的参数。

应用

例如，在一个轮廓列表ipContours中使用厚度为 5px 的红线画出的所有轮廓如下所示：

```
hr := F_VN_DrawContours(
    ipContours      :=  ipContours,
    nContourIndex   :=  -1,
    ipDestImage     :=  ipImageRes,
    aColor          :=  aColorRed,
    nThickness      :=  5,
    hrPrev          :=  hr
);
```

返回 ContainerType_Vector_Vector_TcVnPoint2_DINT 类型轮廓的典型函数

- F_VN_DetectBlobs [▶ 1172]
- F_VN_FindContours [▶ 1180]或F_VN_FindContoursExp [▶ 1182]
- F_VN_FindContourHierarchyExp [▶ 1176]

样本

- 带看门狗监控的斑点检测 [▶ 2755]

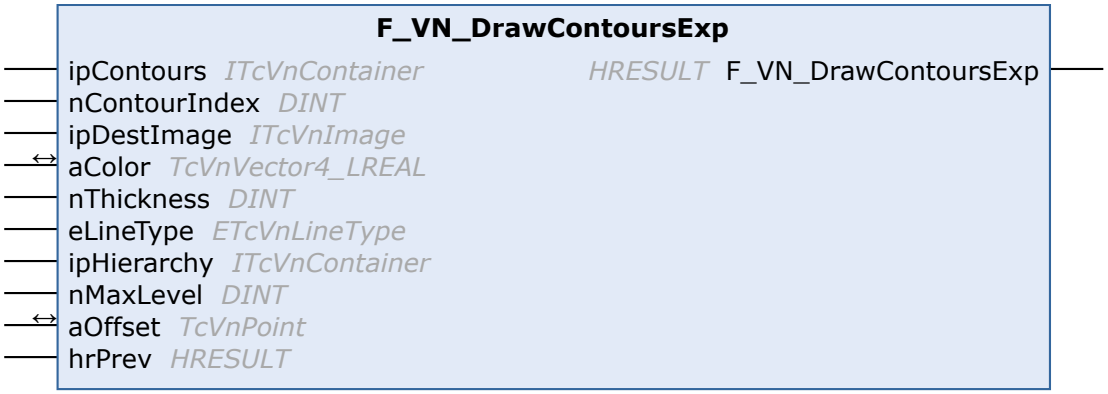
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.14 F_VN_DrawContoursExp



Draws a single point set or multiple point sets that are interpreted as contours. (expert function)

语法

定义：

```
FUNCTION F_VN_DrawContoursExp : HRESULT
VAR_INPUT
    ipContours    : ITcVnContainer;
    nContourIndex : DINT;
    ipDestImage   : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor        : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness    : DINT;
    eLineType     : ETcVnLineType;
    ipHierarchy   : ITcVnContainer;
    nMaxLevel     : DINT;
END_VAR
VAR_IN_OUT
    aOffset       : TcVnPoint;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContours	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT）或多个轮廓（ContainerType_Vector_Vector_TcVnPoint2_DINT）
nContourIndex	DINT	要绘制的特定轮廓的索引（如果为负值，则绘制容器内的所有轮廓）
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充轮廓）
eLineType	ETcVnLineType [▶ 201]	线条类型
ipHierarchy	ITcVnContainer [▶ 365]	轮廓层次（ContainerType_Vector_TcVnVector4_DINT）
nMaxLevel	DINT	要绘制的轮廓的最大等级
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色
aOffset	TcVnPoint [▶ 152]	每个轮廓点移动的偏移量

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_DrawContoursExp 对应于函数F_VN_DrawContours [▶ 1012]，由额外的输入变量扩展。

因此，除了找到的轮廓外，还可以考虑函数F_VN_FindContourHierarchyExp [▶ 1175]返回的层次结构描述。绘制轮廓的层次结构通过`ipHierarchy`和输入变量`nMaxLevel`指定。如果 0 被转移到`ipHierarchy`，那么层次结构就被忽略了。

`aOffset`可以用来绘制所有在 x 和 y 方向上有统一偏移的轮廓点。这一点很有意义，例如，如果轮廓是在一个 ROI 中搜索的，但要画到原始图像中。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.15 F_VN_DrawEllipse



Draws an ellipse.

语法

定义：

```
FUNCTION F_VN_DrawEllipse : HRESULT
VAR_IN_OUT
    stEllipse    : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness   : DINT;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充椭圆）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stEllipse	TcVnRotatedRectangle [▶ 237]	要绘制的椭圆（旋转角，单位：度）
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.16 F_VN_DrawEllipseExp



Draws an ellipse. (expert function)

语法

定义：

```
FUNCTION F_VN_DrawEllipseExp : HRESULT
VAR_IN_OUT
    stEllipse   : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    eLineType   : ETcVnLineType;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充椭圆）
eLineType	ETcVnLineType [▶ 201]	线条类型

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stEllipse	TcVnRotatedRectangle [▶ 237]	要绘制的椭圆（旋转角，单位：度）
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

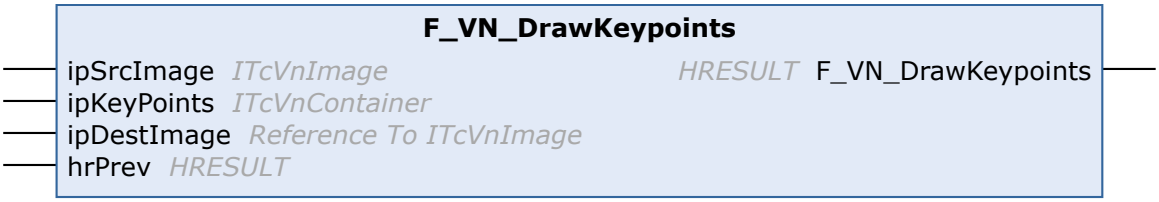
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.17 F_VN_DrawKeypoints



Draws the keypoints for visualization purpose.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_DrawKeypoints : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipKeyPoints : ITcVnContainer;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 类型元素）
ipKeyPoints	ITcVnContainer [▶ 365]	包含关键点的容器（ContainerType_Vector_TcVnKeyPoint）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的彩色图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

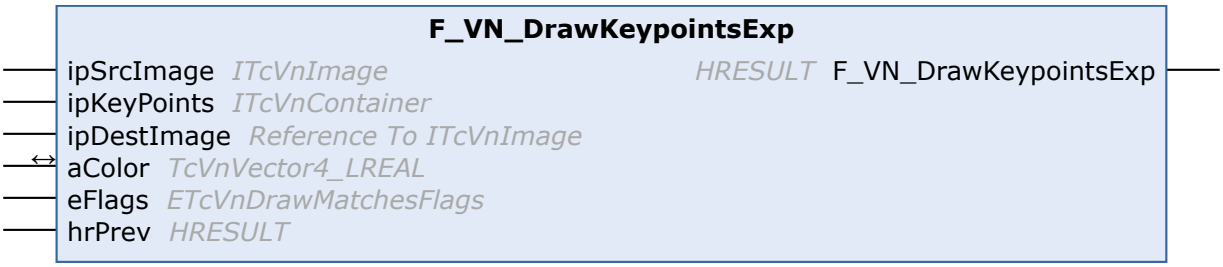
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.18 F_VN_DrawKeypointsExp



Draws the keypoints for visualization purpose. (expert function)

语法

定义：

```
FUNCTION F_VN_DrawKeypointsExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipKeyPoints : ITcVnContainer;
    ipDestImage : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    eFlags      : ETcVnDrawMatchesFlags;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 类型元素）
ipKeyPoints	ITcVnContainer [▶ 365]	包含关键点的容器（ContainerType_Vector_TcVnKeyPoint）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的彩色图像。）
eFlags	ETcVnDrawMatchesFlags [▶ 189]	标志组合，支持过度绘制现有目标图像和/或绘制附加（富）关键点信息（尺寸和方向）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	绘制关键点的颜色（对于 {-1、-1、-1、-1}，会为每个点随机选择一种颜色）

返回值

HRESULT [▶ 135]

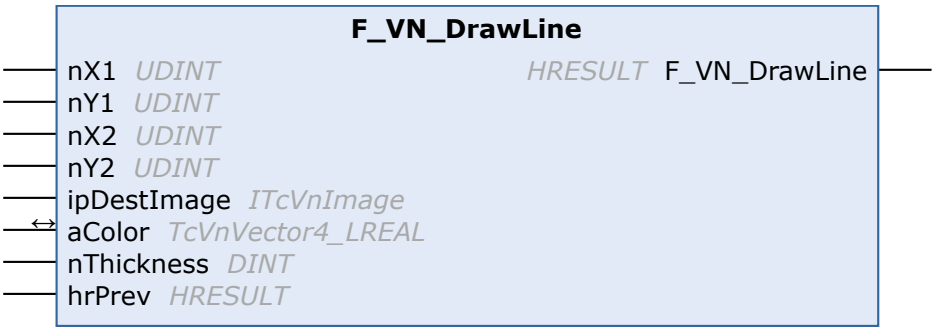
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.19 F_VN_DrawLine



Draws a line.

语法

定义：

```
FUNCTION F_VN_DrawLine : HRESULT
VAR_INPUT
    nX1          : UDINT;
    nY1          : UDINT;
    nX2          : UDINT;
    nY2          : UDINT;
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness   : DINT;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
nX1	UDINT	起点的 x 坐标
nY1	UDINT	起点的 y 坐标
nX2	UDINT	终点的 x 坐标
nY2	UDINT	终点的 y 坐标
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

HRESULT

16#734	OUTOFRANGE	部分线条位于图像区域之外。使用函数 F_VN_ClipLineToBoundary_ITcVnImage [▶ 1516]以确保指定的线适合在图像区域内。

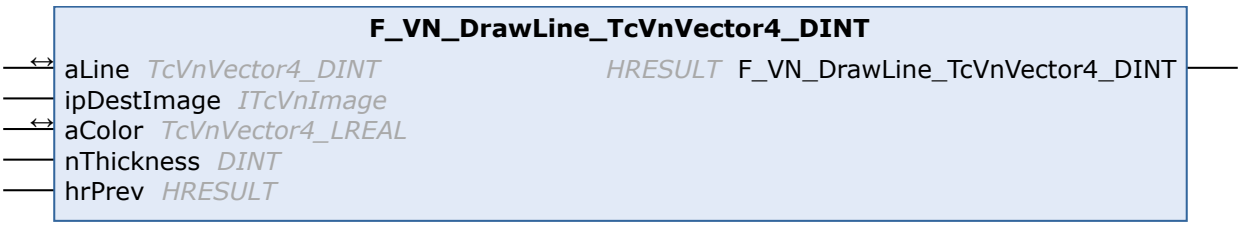
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.20 F_VN_DrawLine_TcVnVector4_DINT



Draws a line.

语法

定义：

```
FUNCTION F_VN_DrawLine_TcVnVector4_DINT : HRESULT
VAR_IN_OUT
    aLine       : TcVnVector4_DINT;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aLine	TcVnVector4_DINT [▶ 153]	线段 [x1，y1，x2，y2] 的起点和终点
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

HRESULT

16#734	OUTOFRANGE	部分线条位于图像区域之外。使用函数 F_VN_ClipLineToBoundary_ITcVnImage [▶ 1516]以确保指定的线适合在图像区域内。

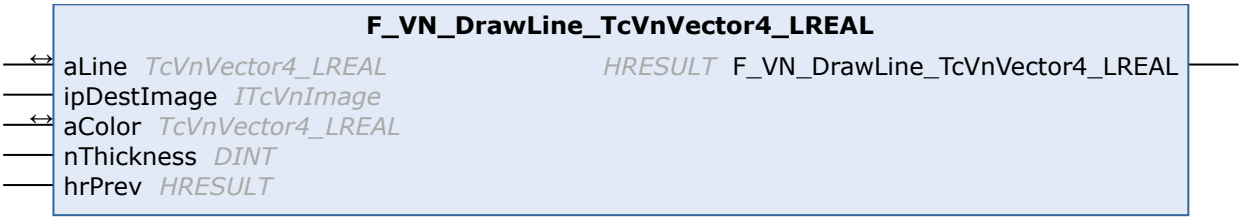
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.21 F_VN_DrawLine_TcVnVector4_LREAL



Draws a line.

语法

定义：

```
FUNCTION F_VN_DrawLine_TcVnVector4_LREAL : HRESULT
VAR_IN_OUT
    aLine       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aLine	TcVnVector4_LREAL [▶ 153]	第一个和第二个元素描述了方向向量的 x 和 y 分量。第三个和第四个元素描述位置向量的 x 和 y 分量。
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

注意

浮点异常

如果正在执行的PLC 任务 [▶ 63]的选项**浮点异常**启用，该功能会不必要地引起错误。因此，停用这个选项。

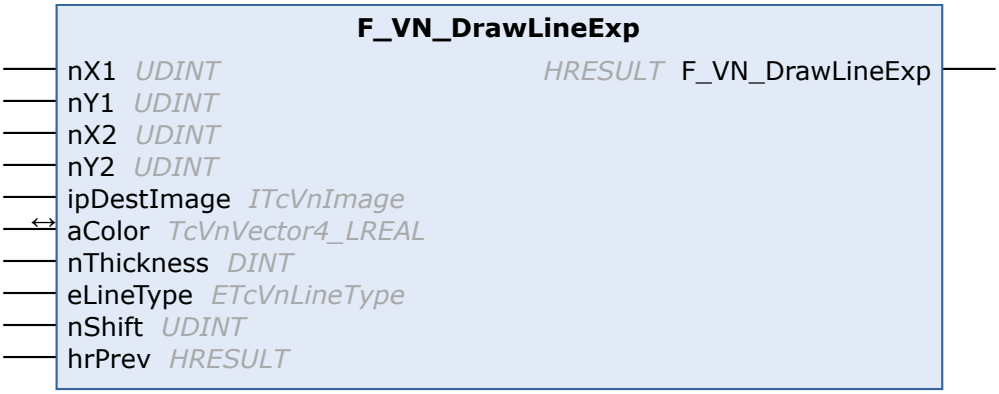
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.22 F_VN_DrawLineExp



Draws a line. (expert function)

语法

定义：

```
FUNCTION F_VN_DrawLineExp : HRESULT
VAR_INPUT
    nX1          : UDINT;
    nY1          : UDINT;
    nX2          : UDINT;
    nY2          : UDINT;
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness   : DINT;
    eLineType    : ETcVnLineType;
    nShift       : UDINT;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
nX1	UDINT	起点的 x 坐标
nY1	UDINT	起点的 y 坐标
nX2	UDINT	终点的 x 坐标
nY2	UDINT	终点的 y 坐标
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细
eLineType	ETcVnLineType [▶ 201]	线条类型
nShift	UDINT	坐标的小数位数（位移）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

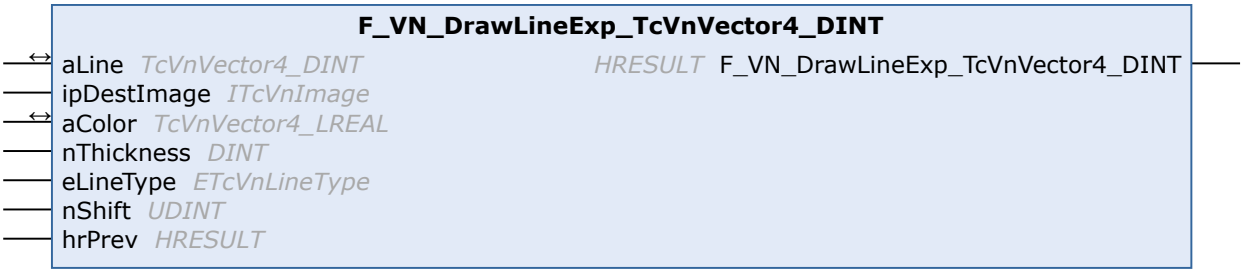
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.23 F_VN_DrawLineExp_TcVnVector4_DINT



Draws a line. (expert function)

语法

定义：

```
FUNCTION F_VN_DrawLineExp_TcVnVector4_DINT : HRESULT
VAR_IN_OUT
    aLine       : TcVnVector4_DINT;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    eLineType   : ETcVnLineType;
    nShift      : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细
eLineType	ETcVnLineType [▶ 201]	线条类型
nShift	UDINT	坐标的小数位数（位移）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aLine	TcVnVector4_DINT [▶ 153]	线段 [x1，y1，x2，y2] 的起点和终点
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

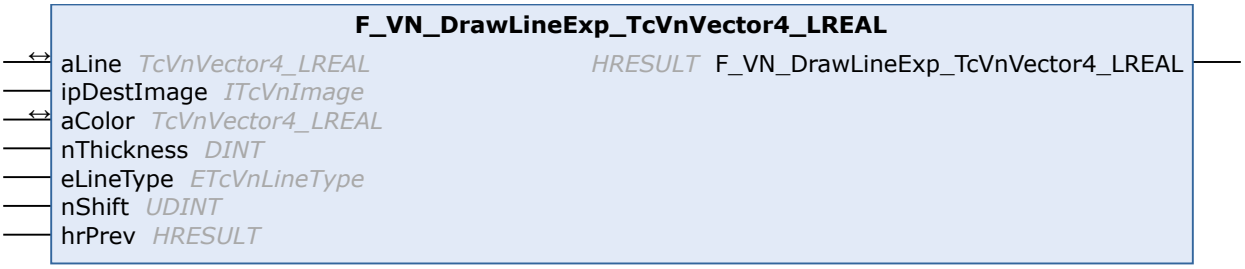
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.24 F_VN_DrawLineExp_TcVnVector4_LREAL



Draws a line. (expert function)

语法

定义：

```
FUNCTION F_VN_DrawLineExp_TcVnVector4_LREAL : HRESULT
VAR_IN_OUT
    aLine       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    eLineType   : ETcVnLineType;
    nShift      : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细
eLineType	ETcVnLineType [▶ 201]	线条类型
nShift	UDINT	坐标的小数位数（位移）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aLine	TcVnVector4_LREAL [▶ 153]	第一个和第二个元素描述了方向向量的 x 和 y 分量。第三个和第四个元素描述位置向量的 x 和 y 分量。
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

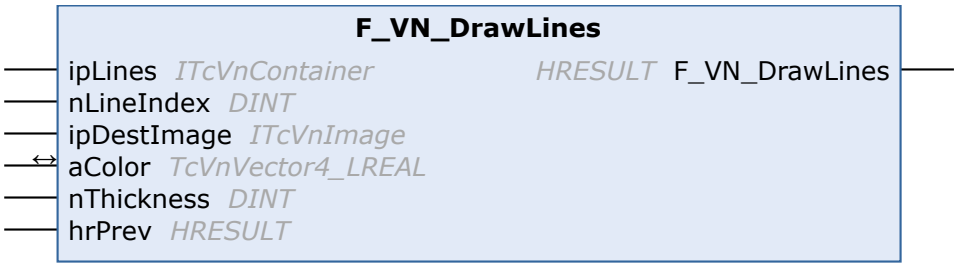
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.25 F_VN_DrawLines



Draws lines.

语法

定义：

```
FUNCTION F_VN_DrawLines : HRESULT
VAR_INPUT
    ipLines     : ITcVnContainer;
    nLineIndex  : DINT;
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipLines	ITcVnContainer [▶ 365]	带有线条描述的容器（ContainerType_Vector_TcVnVector2_REAL：与原点的距离 [0]（以像素为单位）和旋转角 [1]（以弧度为单位）。ContainerType_Vector_TcVnVector4_LREAL：第一个和第二个元素描述的是该直线共线向量的 x 和 y 分量。第三个和第四个元素描述该直线上一个点的 x 和 y 分量。ContainerType_Vector_TcVnVector4_DINT: 起点和终点 [x1，y1，x2，y2]）
nLineIndex	DINT	要绘制的特定直线的索引（如果为负值，则绘制容器内的所有直线）
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

HRESULT

16#734	OUTOFRANGE	部分线条位于图像区域之外。使用函数 F_VN_ClipLineToBoundary_ITcVnImage [▶ 1516]以确保指定的线适合在图像区域内。

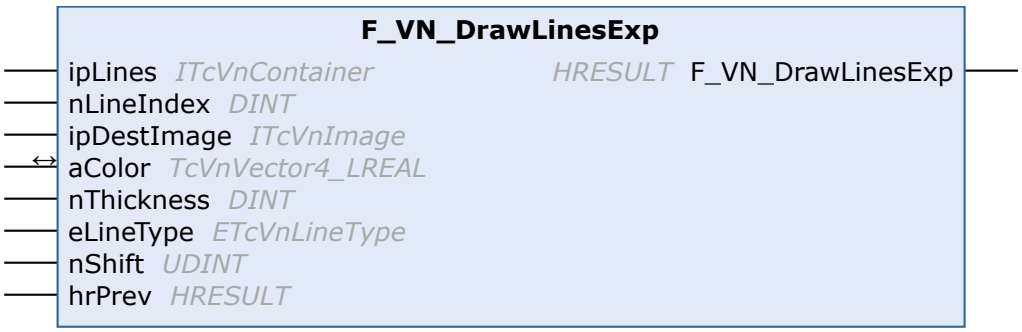
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.26 F_VN_DrawLinesExp



Draws lines. (expert function)

语法

定义：

```
FUNCTION F_VN_DrawLinesExp : HRESULT
VAR_INPUT
    ipLines     : ITcVnContainer;
    nLineIndex  : DINT;
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    eLineType   : ETcVnLineType;
    nShift      : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipLines	ITcVnContainer [▶ 365]	带有线条描述的容器（ContainerType_Vector_TcVnVector2_REAL：与原点的距离 [0]（以像素为单位）和旋转角 [1]（以弧度为单位）。ContainerType_Vector_TcVnVector4_LREAL：第一个和第二个元素描述的是该直线共线向量的 x 和 y 分量。第三个和第四个元素描述该直线上一个点的 x 和 y 分量。ContainerType_Vector_TcVnVector4_DINT: 起点和终点 [x1，y1，x2，y2]）
nLineIndex	DINT	要绘制的特定直线的索引（如果为负值，则绘制容器内的所有直线）
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细
eLineType	ETcVnLineType [▶ 201]	线条类型
nShift	UDINT	坐标的小数位数（位移）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

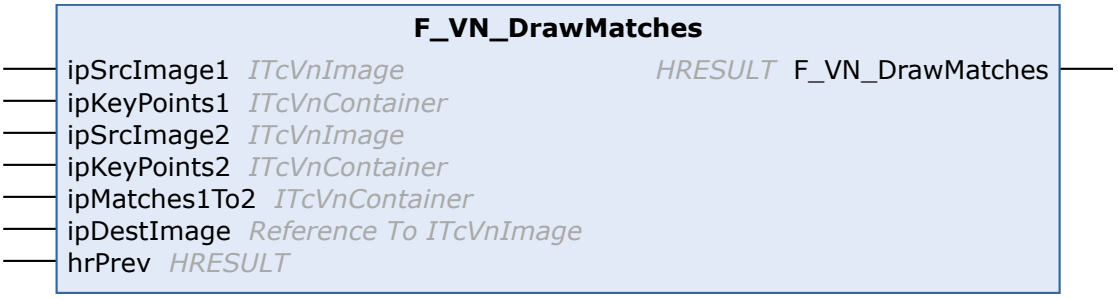
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.27 F_VN_DrawMatches



Draws the keypoints and matches for visualization purpose.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_DrawMatches : HRESULT
VAR_INPUT
    ipSrcImage1   : ITcVnImage;
    ipKeyPoints1  : ITcVnContainer;
    ipSrcImage2   : ITcVnImage;
    ipKeyPoints2  : ITcVnContainer;
    ipMatches1To2 : ITcVnContainer;
    ipDestImage   : Reference To ITcVnImage;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	源图像 1（USINT 类型元素）
ipKeyPoints1	ITcVnContainer [▶ 365]	包含关键点的容器 1（ContainerType_Vector_TcVnKeyPoint）
ipSrcImage2	ITcVnImage [▶ 407]	源图像 2（USINT 类型元素，通道数量必须与 ipSrcImage1 相同）
ipKeyPoints2	ITcVnContainer [▶ 365]	包含关键点的容器 2（ContainerType_Vector_TcVnKeyPoint）
ipMatches1To2	ITcVnContainer [▶ 365]	描述符匹配的容器（ContainerType_Vector_TcVnDMatch）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的彩色图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

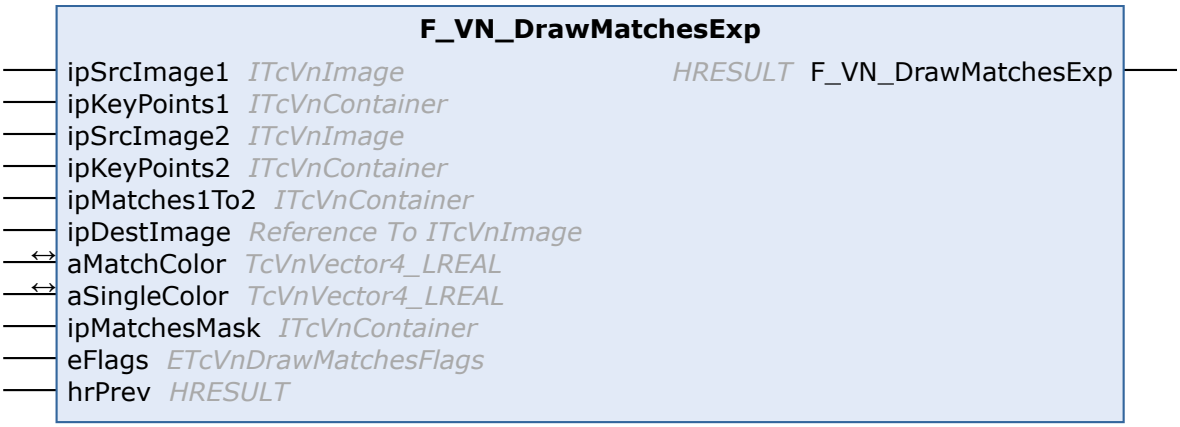
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.28 F_VN_DrawMatchesExp



Draws the keypoints and matches for visualization purpose. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_DrawMatchesExp : HRESULT
VAR_INPUT
    ipSrcImage1    : ITcVnImage;
    ipKeyPoints1   : ITcVnContainer;
    ipSrcImage2    : ITcVnImage;
    ipKeyPoints2   : ITcVnContainer;
    ipMatches1To2  : ITcVnContainer;
    ipDestImage    : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aMatchColor    : TcVnVector4_LREAL;
    aSingleColor   : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipMatchesMask : ITcVnContainer;
    eFlags        : ETcVnDrawMatchesFlags;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage1	ITcVnImage [▶ 407]	源图像 1（USINT 类型元素）
ipKeyPoints1	ITcVnContainer [▶ 365]	包含关键点的容器 1（ContainerType_Vector_TcVnKeyPoint）
ipSrcImage2	ITcVnImage [▶ 407]	源图像 2（USINT 类型元素，通道数量必须与 ipSrcImage1 相同）
ipKeyPoints2	ITcVnContainer [▶ 365]	包含关键点的容器 2（ContainerType_Vector_TcVnKeyPoint）
ipMatches1To2	ITcVnContainer [▶ 365]	描述符匹配的容器（ContainerType_Vector_TcVnDMatch）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的彩色图像。）
ipMatchesMask	ITcVnContainer [▶ 365]	用于选择要绘制的匹配项的掩膜（ContainerType_Vector_SINT；如果要绘制所有匹配项，则设为 0。）
eFlags	ETcVnDrawMatchesFlags [▶ 189]	标志组合，支持过度绘制现有目标图像和/或绘制附加（富）关键点信息（大小和方向）和/或跳过单个关键点
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMatchColor	TcVnVector4_LREAL [▶ 153]	绘制匹配项的颜色（对于 {-1、-1、-1、-1}，会为每个点和直线随机选择一种颜色）
aSingleColor	TcVnVector4_LREAL [▶ 153]	绘制单个关键点的颜色（对于 {-1、-1、-1、-1}，会为每个点随机选择一种颜色）

返回值

HRESULT [▶ 135]

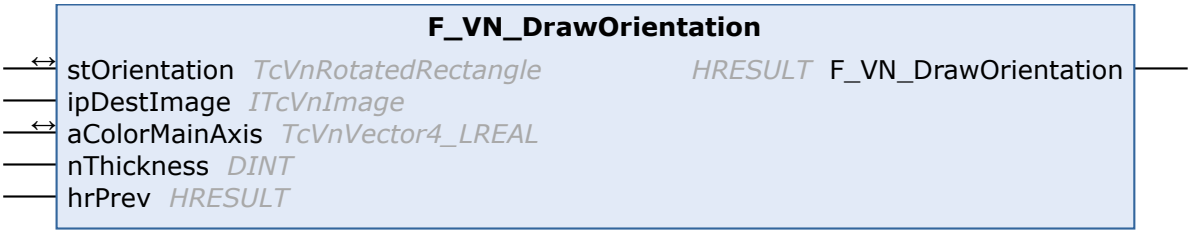
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.29 F_VN_DrawOrientation



Draw the main axis of a set of points/contour based on a rotated rectangle.

语法

定义：

```
FUNCTION F_VN_DrawOrientation : HRESULT
VAR_IN_OUT
    stOrientation  : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    ipDestImage    : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColorMainAxis : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness     : DINT;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stOrientation	TcVnRotatedRectangle [▶ 237]	旋转矩形，包含中心点、轴长和顺时针方向的轮廓旋转角
aColorMainAxis	TcVnVector4_LREAL [▶ 153]	主轴的颜色

返回值

HRESULT [▶ 135]

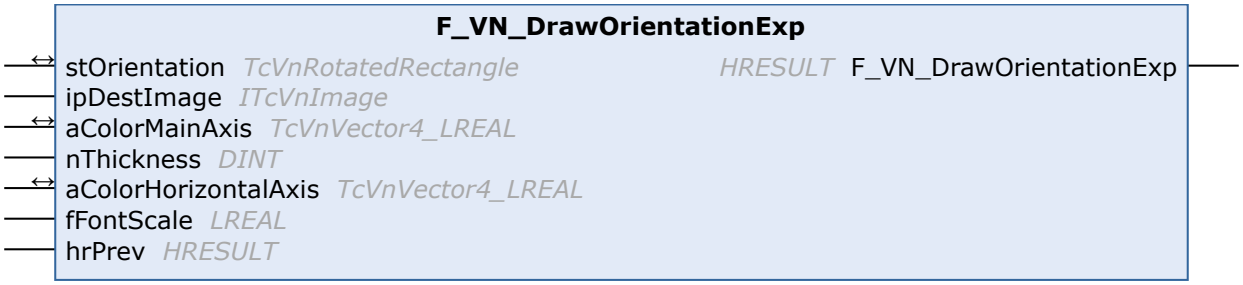
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.30 F_VN_DrawOrientationExp



Draw the main axis, the horizontal axis and the angle of a set of points/contour based on a rotated rectangle. (expert function)

语法

定义：

```
FUNCTION F_VN_DrawOrientationExp : HRESULT
VAR_IN_OUT
    stOrientation        : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    ipDestImage          : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColorMainAxis       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness           : DINT;
END_VAR
VAR_IN_OUT
    aColorHorizontalAxis : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    fFontScale           : LREAL;
    hrPrev               : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细
fFontScale	LREAL	文本缩放系数
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stOrientation	TcVnRotatedRectangle [▶ 237]	旋转矩形，包含中心点、轴长和顺时针方向的轮廓旋转角
aColorMainAxis	TcVnVector4_LREAL [▶ 153]	主轴的颜色
aColorHorizontalAxis	TcVnVector4_LREAL [▶ 153]	水平轴的颜色

返回值

HRESULT [▶ 135]

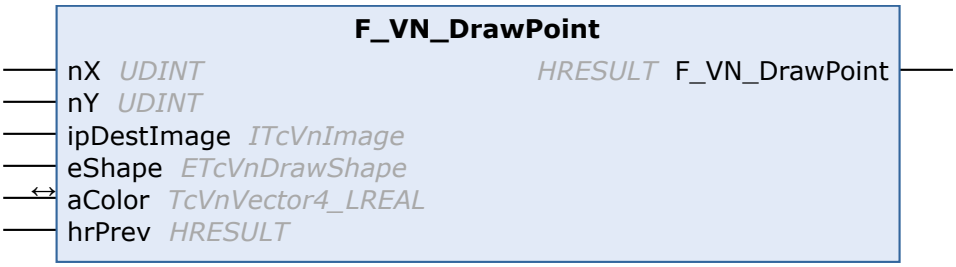
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.31 F_VN_DrawPoint



Draws a point.

语法

定义：

```
FUNCTION F_VN_DrawPoint : HRESULT
VAR_INPUT
    nX          : UDINT;
    nY          : UDINT;
    ipDestImage : ITcVnImage;
    eShape      : ETcVnDrawShape;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nX	UDINT	点的 x 坐标
nY	UDINT	点的 y 坐标
ipDestImage	ITcVnImage [▶ 407]	目标图像
eShape	ETcVnDrawShape [▶ 190]	点形状
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色（对于 {-1、-1、-1、-1}，随机选择一种颜色）

返回值

HRESULT [▶ 135]

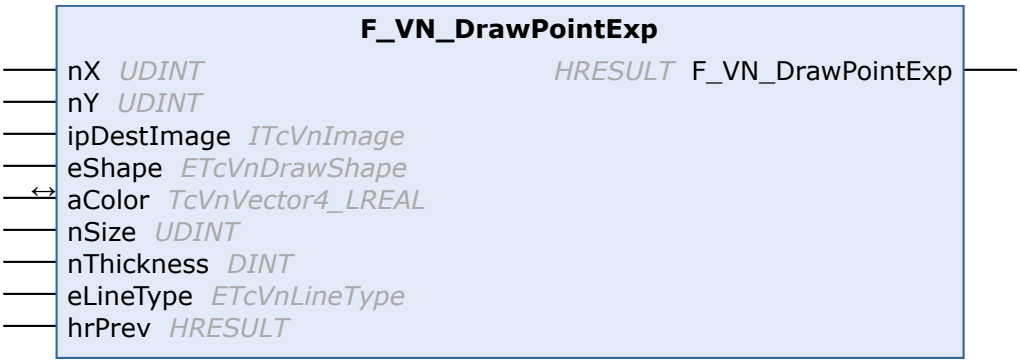
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.32 F_VN_DrawPointExp



Draw a point. (expert function)

语法

定义：

```
FUNCTION F_VN_DrawPointExp : HRESULT
VAR_INPUT
    nX          : UDINT;
    nY          : UDINT;
    ipDestImage : ITcVnImage;
    eShape      : ETcVnDrawShape;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nSize       : UDINT;
    nThickness  : DINT;
    eLineType   : ETcVnLineType;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nX	UDINT	点的 x 坐标
nY	UDINT	点的 y 坐标
ipDestImage	ITcVnImage [▶ 407]	目标图像
eShape	ETcVnDrawShape [▶ 190]	点形状
nSize	UDINT	形状大小（半宽）
nThickness	DINT	线条粗细（如果为负值，则填充封闭的形状）
eLineType	ETcVnLineType [▶ 201]	线条类型

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色（对于 {-1、-1、-1、-1}，随机选择一种颜色）

返回值

HRESULT [▶ 135]

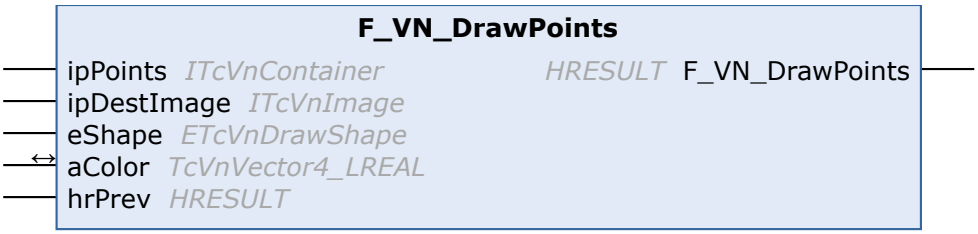
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.33 F_VN_DrawPoints



Draw a collection of points.

语法

定义：

```
FUNCTION F_VN_DrawPoints : HRESULT
VAR_INPUT
    ipPoints    : ITcVnContainer;
    ipDestImage : ITcVnImage;
    eShape      : ETcVnDrawShape;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipPoints	ITcVnContainer [▶ 365]	包含 TcVnPoint2_DINT 或 TcVnPoint2_REAL 元素的容器
ipDestImage	ITcVnImage [▶ 407]	目标图像
eShape	ETcVnDrawShape [▶ 190]	点形状

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色（对于 {-1、-1、-1、-1}，为每个点随机选择一种颜色）

返回值

HRESULT [▶ 135]

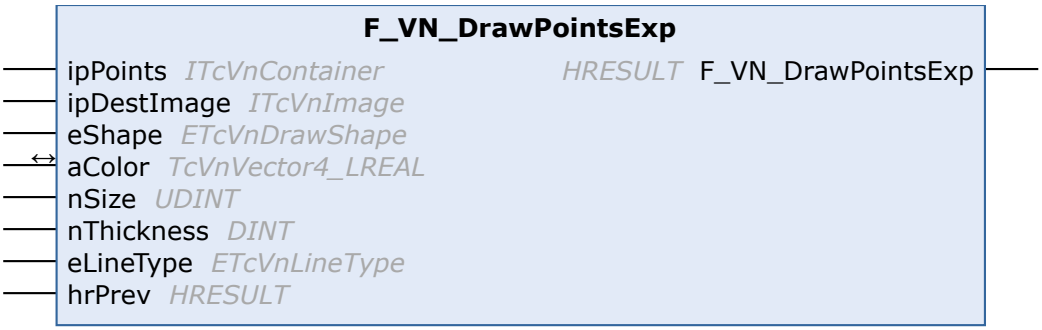
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.34 F_VN_DrawPointsExp



Draw a collection of points. (expert function)

语法

定义：

```
FUNCTION F_VN_DrawPointsExp : HRESULT
VAR_INPUT
    ipPoints    : ITcVnContainer;
    ipDestImage : ITcVnImage;
    eShape      : ETcVnDrawShape;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nSize       : UDINT;
    nThickness  : DINT;
    eLineType   : ETcVnLineType;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipPoints	ITcVnContainer [▶ 365]	包含 TcVnPoint2_DINT 或 TcVnPoint2_REAL 元素的容器
ipDestImage	ITcVnImage [▶ 407]	目标图像
eShape	ETcVnDrawShape [▶ 190]	点形状
nSize	UDINT	形状大小（半宽）
nThickness	DINT	线条粗细（如果为负值，则填充封闭的形状）
eLineType	ETcVnLineType [▶ 201]	线条类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色（对于 {-1、-1、-1、-1}，为每个点随机选择一种颜色）

返回值

HRESULT [▶ 135]

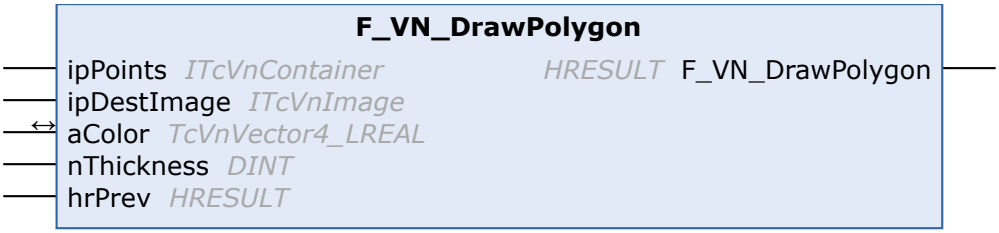
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.35 F_VN_DrawPolygon



Draw a pologon using collection of points.

语法

定义：

```
FUNCTION F_VN_DrawPolygon : HRESULT
VAR_INPUT
    ipPoints    : ITcVnContainer;
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipPoints	ITcVnContainer [▶ 365]	包含 TcVnPoint2_DINT、TcVnPoint2_REAL 或 TcVnPoint2_LREAL 元素的容器
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充多边形）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

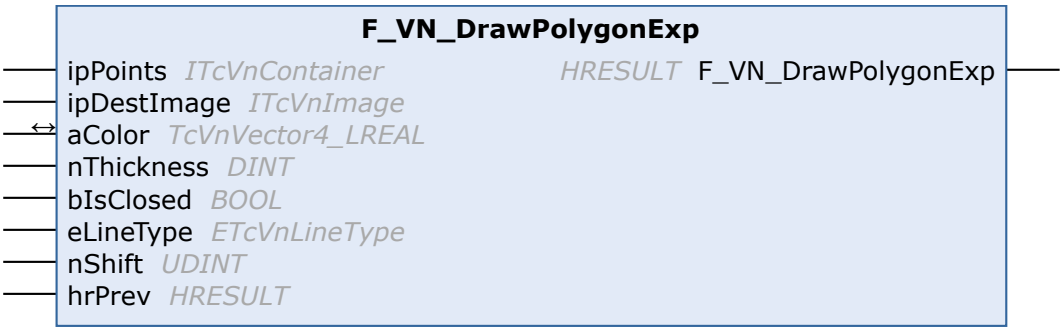
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.36 F_VN_DrawPolygonExp



Draw a pologon using collection of points. (expert function)

语法

定义：

```
FUNCTION F_VN_DrawPolygonExp : HRESULT
VAR_INPUT
    ipPoints    : ITcVnContainer;
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    bIsClosed   : BOOL;
    eLineType   : ETcVnLineType;
```

```
    nShift      : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipPoints	ITcVnContainer [▶ 365]	包含 TcVnPoint2_DINT、TcVnPoint2_REAL 或 TcVnPoint2_LREAL 元素的容器
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充形状）
bIsClosed	BOOL	如果为 true，函数会从最后一个点向第一个点绘制一条线来封闭多边形。
eLineType	ETcVnLineType [▶ 201]	线条类型
nShift	UDINT	坐标的小数位数（位移）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

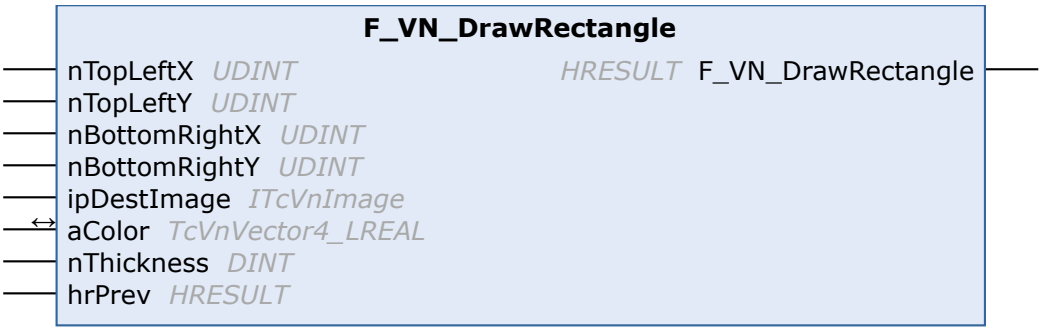
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.37 F_VN_DrawRectangle



Draw a rectangle.

语法

定义：

```
FUNCTION F_VN_DrawRectangle : HRESULT
VAR_INPUT
    nTopLeftX     : UDINT;
    nTopLeftY     : UDINT;
    nBottomRightX : UDINT;
    nBottomRightY : UDINT;
    ipDestImage   : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor        : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness    : DINT;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
nTopLeftX	UDINT	左上角的 x 坐标
nTopLeftY	UDINT	左上角的 y 坐标
nBottomRightX	UDINT	右下角的 x 坐标
nBottomRightY	UDINT	右下角的 y 坐标
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充矩形）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

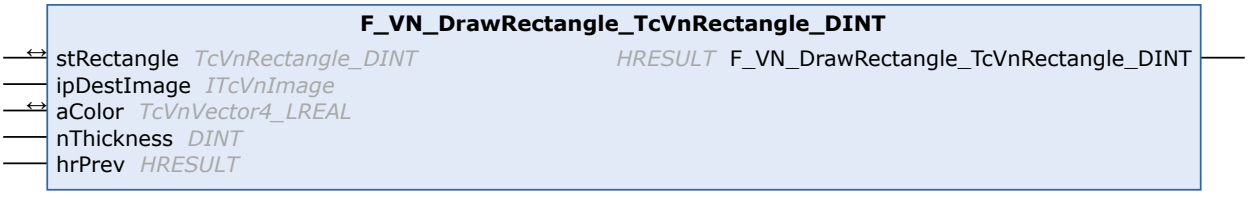
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.38 F_VN_DrawRectangle_TcVnRectangle_DINT



Draw a rectangle.

语法

定义：

```
FUNCTION F_VN_DrawRectangle_TcVnRectangle_DINT : HRESULT
VAR_IN_OUT
    stRectangle : TcVnRectangle_DINT;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充矩形）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stRectangle	TcVnRectangle_DINT [▶ 236]	要绘制的矩形
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.39 F_VN_DrawRectangle_TcVnRectangle_UDINT



Draw a rectangle.

语法

定义：

```
FUNCTION F_VN_DrawRectangle_TcVnRectangle_UDINT : HRESULT
VAR_IN_OUT
    stRectangle : TcVnRectangle_UDINT;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充矩形）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stRectangle	TcVnRectangle_UDINT [▶ 236]	要绘制的矩形
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

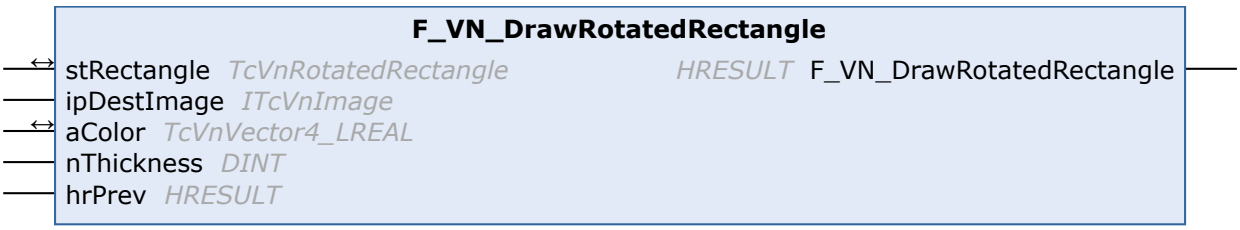
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.40 F_VN_DrawRotatedRectangle



Draw a rectangle.

语法

定义：

```
FUNCTION F_VN_DrawRotatedRectangle : HRESULT
VAR_IN_OUT
    stRectangle : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充矩形）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stRectangle	TcVnRotatedRectangle [▶ 237]	要绘制的矩形（旋转角，单位：度）
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

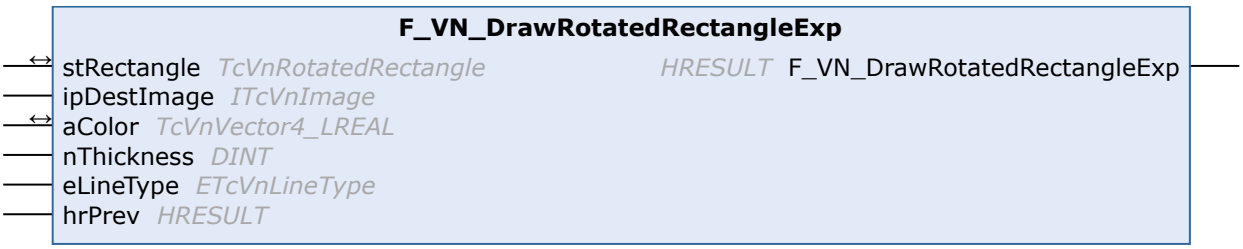
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.41 F_VN_DrawRotatedRectangleExp



Draw a rectangle. (expert function)

语法

定义：

```
FUNCTION F_VN_DrawRotatedRectangleExp : HRESULT
VAR_IN_OUT
    stRectangle : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness  : DINT;
    eLineType   : ETcVnLineType;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
nThickness	DINT	线条粗细（如果为负值，则填充矩形）
eLineType	ETcVnLineType [▶ 201]	线条类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stRectangle	TcVnRotatedRectangle [▶ 237]	要绘制的矩形（旋转角，单位：度）
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.42 F_VN_FillCircle



Paint a filled circle.

语法

定义：

```
FUNCTION F_VN_FillCircle : HRESULT
VAR_INPUT
    nCenterX    : UDINT;
    nCenterY    : UDINT;
    nRadius     : UDINT;
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nCenterX	UDINT	中心的 x 坐标
nCenterY	UDINT	中心的 y 坐标
nRadius	UDINT	半径
ipDestImage	ITcVnImage [▶ 407]	目标图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.43 F_VN_FillContours



Paint a single point set or multiple point sets that are interpreted as contours.

语法

定义：

```
FUNCTION F_VN_FillContours : HRESULT
VAR_INPUT
    ipContours    : ITcVnContainer;
    nContourIndex : DINT;
    ipDestImage   : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor        : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipContours	ITcVnContainer [▶ 365]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT）或多个轮廓（ContainerType_Vector_Vector_TcVnPoint2_DINT）
nContourIndex	DINT	要绘制的特定轮廓的索引（如果为负值，则绘制容器内的所有轮廓）
ipDestImage	ITcVnImage [▶ 407]	目标图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

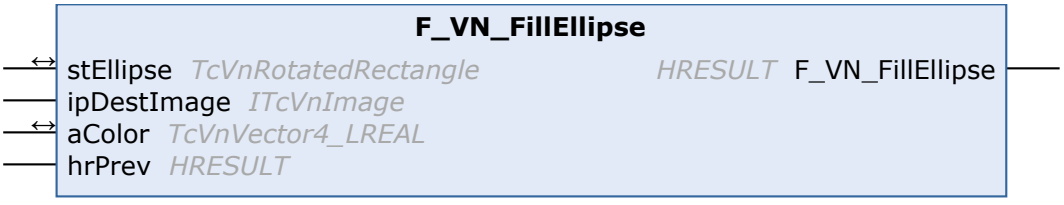
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.44 F_VN_FillEllipse



Paint a filled ellipse.

语法

定义：

```
FUNCTION F_VN_FillEllipse : HRESULT
VAR_IN_OUT
    stEllipse   : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stEllipse	TcVnRotatedRectangle [▶ 237]	要绘制的椭圆（旋转角，单位：度）
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.45 F_VN_FillPolygon



Paint a filled pologon using a set of points.

语法

定义：

```
FUNCTION F_VN_FillPolygon : HRESULT
VAR_INPUT
    ipPoints    : ITcVnContainer;
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipPoints	ITcVnContainer [▶ 365]	包含 TcVnPoint2_DINT、TcVnPoint2_REAL 或 TcVnPoint2_LREAL 元素的容器
ipDestImage	ITcVnImage [▶ 407]	目标图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

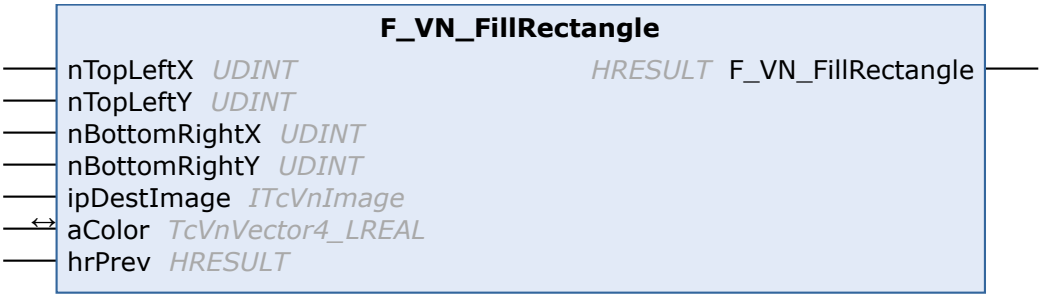
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.46 F_VN_FillRectangle



Paint a filled rectangle.

语法

定义：

```
FUNCTION F_VN_FillRectangle : HRESULT
VAR_INPUT
    nTopLeftX     : UDINT;
```

```
    nTopLeftY     : UDINT;
    nBottomRightX : UDINT;
    nBottomRightY : UDINT;
    ipDestImage   : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor        : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
nTopLeftX	UDINT	左上角的 x 坐标
nTopLeftY	UDINT	左上角的 y 坐标
nBottomRightX	UDINT	右下角的 x 坐标
nBottomRightY	UDINT	右下角的 y 坐标
ipDestImage	ITcVnImage [▶ 407]	目标图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.47 F_VN_FillRotatedRectangle



Paint a filled rectangle.

语法

定义：

```
FUNCTION F_VN_FillRotatedRectangle : HRESULT
VAR_IN_OUT
    stRectangle : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    ipDestImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aColor      : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipDestImage	ITcVnImage [▶ 407]	目标图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stRectangle	TcVnRotatedRectangle [▶ 237]	要绘制的矩形（旋转角，单位：度）
aColor	TcVnVector4_LREAL [▶ 153]	颜色

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.48 F_VN_PlotIntensityProfile



Plots the pixel intensity profile along a line segment in an image.

语法

定义：

```
FUNCTION F_VN_PlotIntensityProfile : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
END_VAR
```

```
VAR_IN_OUT
    aStartPoint : TcVnPoint2_REAL;
    aEndPoint   : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（USINT，1 通道。x：位置，y：强度，原点：左下角。如有需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aStartPoint	TcVnPoint2_REAL [▶ 152]	线段的起点
aEndPoint	TcVnPoint2_REAL [▶ 152]	线段的终点

返回值

HRESULT [▶ 135]

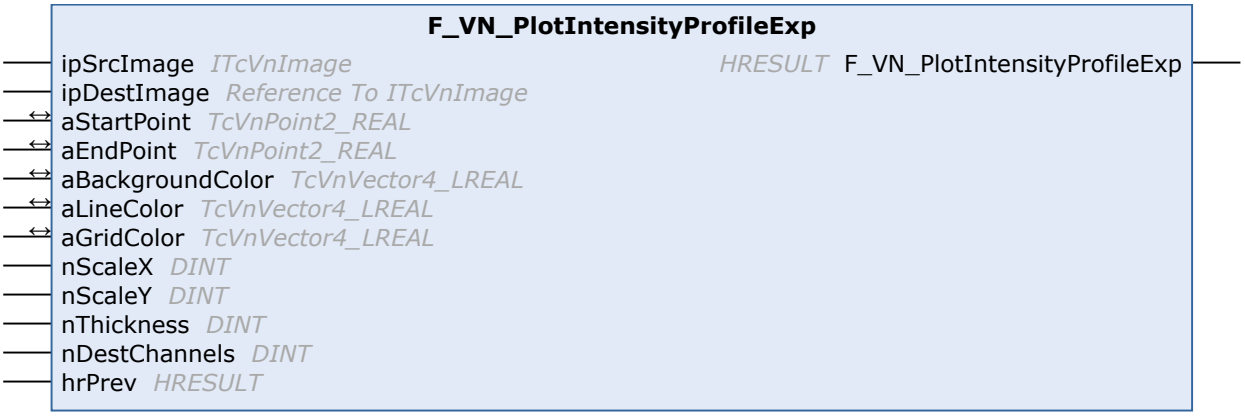
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.49 F_VN_PlotIntensityProfileExp



Plots the pixel intensity profile along a line segment in an image. (expert function)

语法

定义：

```
FUNCTION F_VN_PlotIntensityProfileExp : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
    ipDestImage      : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aStartPoint      : TcVnPoint2_REAL;
    aEndPoint        : TcVnPoint2_REAL;
    aBackgroundColor : TcVnVector4_LREAL;
    aLineColor       : TcVnVector4_LREAL;
    aGridColor       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nScaleX          : DINT;
    nScaleY          : DINT;
    nThickness       : DINT;
    nDestChannels    : DINT;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（USINT，1 通道。x：位置，y：强度，原点：左下角。如有需要，将创建适当的目标图像。）
nScaleX	DINT	在 x 方向上缩放（ipDestImage 的宽度为 nScaleX *（ipSrcImage 宽度 - 1）+ 1）
nScaleY	DINT	按 Y 方向缩放（ipDestImage 的高度为 nScaleY * 255 + 1）
nThickness	DINT	线条粗细
nDestChannels	DINT	ipDestImage 通道（1 或 3）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aStartPoint	TcVnPoint2_REAL [▶ 152]	线段的起点
aEndPoint	TcVnPoint2_REAL [▶ 152]	线段的终点
aBackgroundColor	TcVnVector4_LREAL [▶ 153]	Background color（背景颜色）
aLineColor	TcVnVector4_LREAL [▶ 153]	线条颜色
aGridColor	TcVnVector4_LREAL [▶ 153]	网格颜色

返回值

HRESULT [▶ 135]

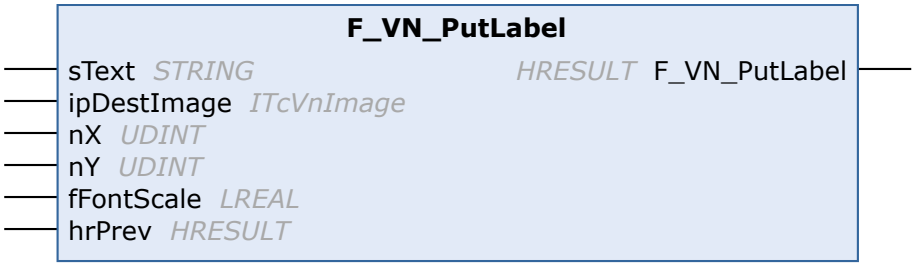
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.50 F_VN_PutLabel



Write a label (black text on white background) into an image.

语法

定义：

```
FUNCTION F_VN_PutLabel : HRESULT
VAR_INPUT
    sText       : STRING;
    ipDestImage : ITcVnImage;
    nX          : UDINT;
    nY          : UDINT;
    fFontScale  : LREAL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
sText	STRING	Text
ipDestImage	ITcVnImage [▶ 407]	目标图像
nX	UDINT	x 坐标（左下角）
nY	UDINT	y 坐标（左下角）
fFontScale	LREAL	缩放系数
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

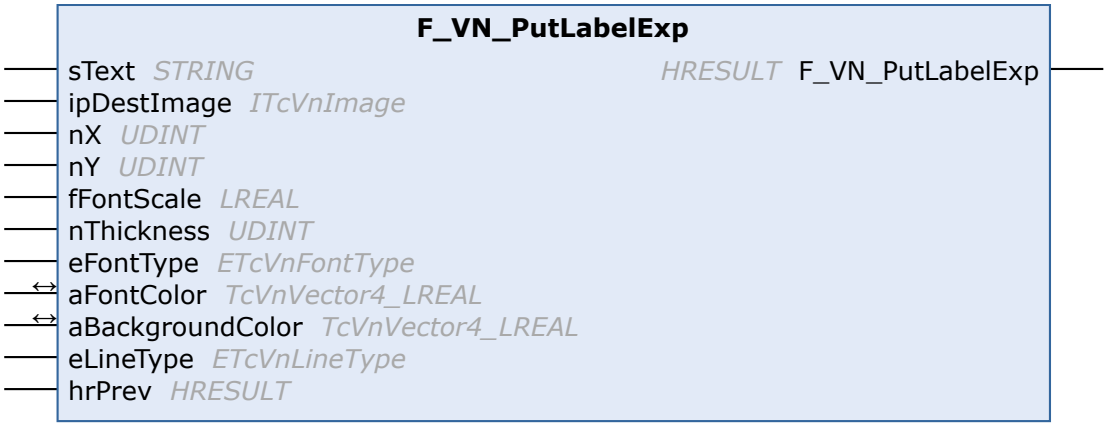
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.51 F_VN_PutLabelExp



Write a label (text on unified background) into an image. (expert function)

语法

定义：

```
FUNCTION F_VN_PutLabelExp : HRESULT
VAR_INPUT
    sText             : STRING;
    ipDestImage       : ITcVnImage;
    nX                : UDINT;
    nY                : UDINT;
    fFontScale        : LREAL;
    nThickness        : UDINT;
    eFontType         : ETcVnFontType;
END_VAR
VAR_IN_OUT
    aFontColor        : TcVnVector4_LREAL;
    aBackgroundColor  : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    eLineType         : ETcVnLineType;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
sText	STRING	Text
ipDestImage	ITcVnImage [▶ 407]	目标图像
nX	UDINT	x 坐标（左下角）
nY	UDINT	y 坐标（左下角）
fFontScale	LREAL	缩放系数
nThickness	UDINT	线条粗细
eFontType	ETcVnFontType [▶ 196]	字体类型
eLineType	ETcVnLineType [▶ 201]	线条类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aFontColor	TcVnVector4_LREAL [▶ 153]	字体颜色
aBackgroundColor	TcVnVector4_LREAL [▶ 153]	Background color（背景颜色）

返回值

HRESULT [▶ 135]

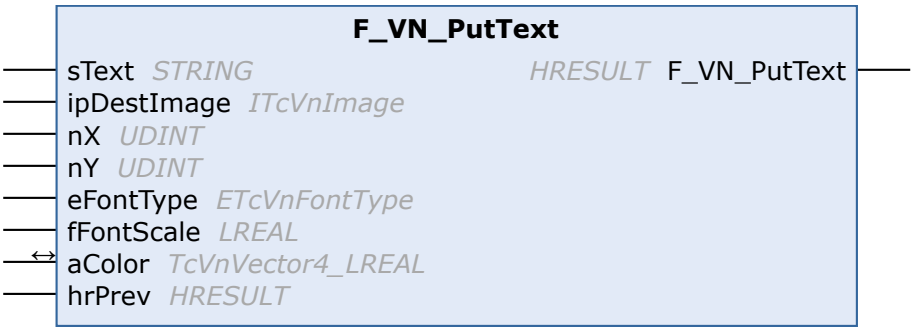
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.12.52 F_VN_PutText



将文本写入图像。

语法

定义：

```
FUNCTION F_VN_PutText : HRESULT
VAR_INPUT
    sText        : STRING;
    ipDestImage  : ITcVnImage;
    nX           : UDINT;
    nY           : UDINT;
    eFontType    : ETcVnFontType;
    fFontScale   : LREAL;
END_VAR
VAR_IN_OUT
    aColor       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
sText	STRING	Text
ipDestImage	ITcVnImage [▶ 407]	目标图像
nX	UDINT	x 坐标（左下角）
nY	UDINT	y 坐标（左下角）
eFontType	ETcVnFontType [▶ 196]	字体类型
fFontScale	LREAL	缩放系数
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	文本颜色

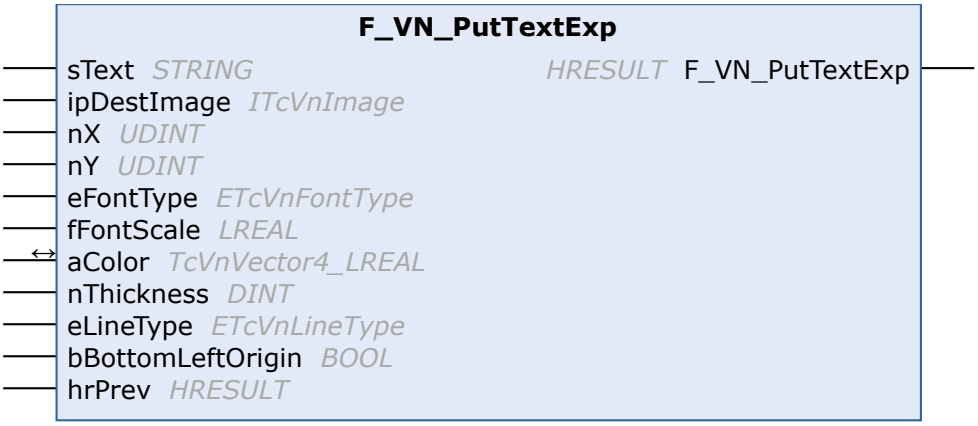
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.12.53 F_VN_PutTextExp



Write text into an image. (expert function)

语法

定义:

```
FUNCTION F_VN_PutTextExp : HRESULT
VAR_INPUT
    sText              : STRING;
    ipDestImage        : ITcVnImage;
    nX                 : UDINT;
    nY                 : UDINT;
    eFontType          : ETcVnFontType;
    fFontScale         : LREAL;
END_VAR
VAR_IN_OUT
    aColor             : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nThickness         : DINT;
    eLineType          : ETcVnLineType;
    bBottomLeftOrigin  : BOOL;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
sText	STRING	Text
ipDestImage	ITcVnImage [▶ 407]	目标图像
nX	UDINT	x 坐标（左下角）
nY	UDINT	y 坐标（左下角）
eFontType	ETcVnFontType [▶ 196]	字体类型

名称	类型	描述
fFontScale	LREAL	缩放系数
nThickness	DINT	线条粗细
eLineType	ETcVnLineType [▶ 201]	线条类型
bBottomLeftOrigin	BOOL	如果为 true，则将图像原点设置为左下角
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aColor	TcVnVector4_LREAL [▶ 153]	文本颜色

返回值

HRESULT [▶ 135]

所需的许可证

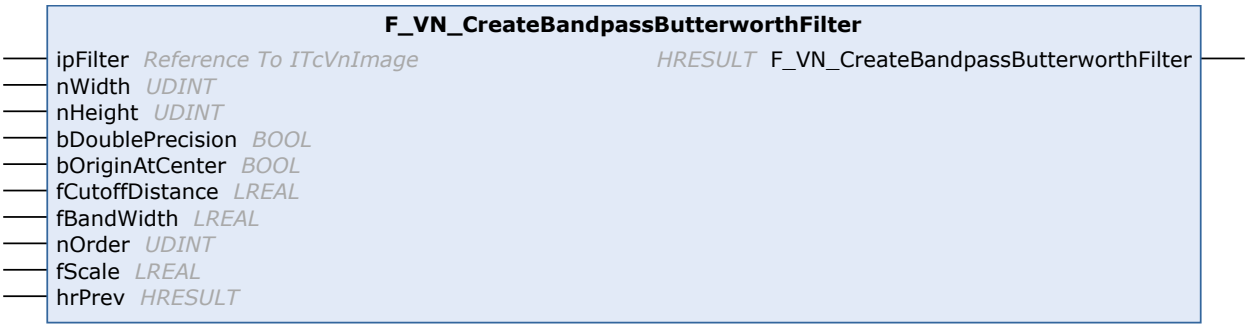
TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.13 傅立叶分析

6.1.4.13.1 F_VN_CreateBandpassButterworthFilter



Creates a bandpass Butterworth filter, which can be applied to a frequency domain image by element-wise multiplication.

语法

定义：

```
FUNCTION F_VN_CreateBandpassButterworthFilter : HRESULT
VAR_INPUT
    ipFilter          : Reference To ITcVnImage;
    nWidth            : UDINT;
    nHeight           : UDINT;
    bDoublePrecision  : BOOL;
    bOriginAtCenter   : BOOL;
    fCutoffDistance   : LREAL;
```

```
    fBandWidth        : LREAL;
    nOrder            : UDINT;
    fScale            : LREAL;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipFilter	引用 ITcVnImage [▶ 407]	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	UDINT	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	UDINT	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	LREAL	巴特沃斯（Butterworth）滤波器的截断距离（> 0）。
fBandWidth	LREAL	巴特沃斯（Butterworth）滤波器的带宽（> 0）。
nOrder	UDINT	巴特沃斯（Butterworth）滤波器的阶数（> 0）。
fScale	LREAL	巴特沃斯（Butterworth）滤波器分母项的比例系数（> 0，例如 0.414 或 1.0）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

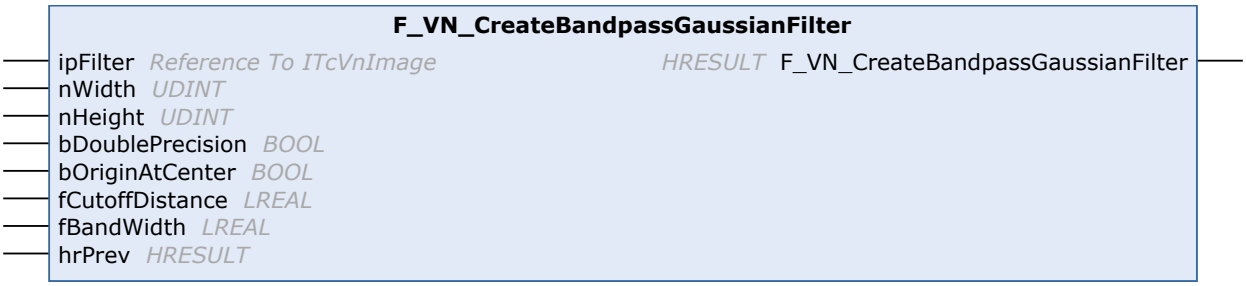
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.13.2 F_VN_CreateBandpassGaussianFilter



Creates a bandpass Gaussian filter, which can be applied to a frequency domain image by element-wise multiplication.

语法

定义：

```
FUNCTION F_VN_CreateBandpassGaussianFilter : HRESULT
VAR_INPUT
    ipFilter         : Reference To ITcVnImage;
    nWidth           : UDINT;
    nHeight          : UDINT;
    bDoublePrecision : BOOL;
    bOriginAtCenter  : BOOL;
    fCutoffDistance  : LREAL;
    fBandWidth       : LREAL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipFilter	引用 ITcVnImage [▶ 407]	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	UDINT	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	UDINT	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	LREAL	高斯（Gaussian）滤波器的截断距离（> 0）。
fBandWidth	LREAL	高斯（Gaussian）滤波器的带宽（> 0）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

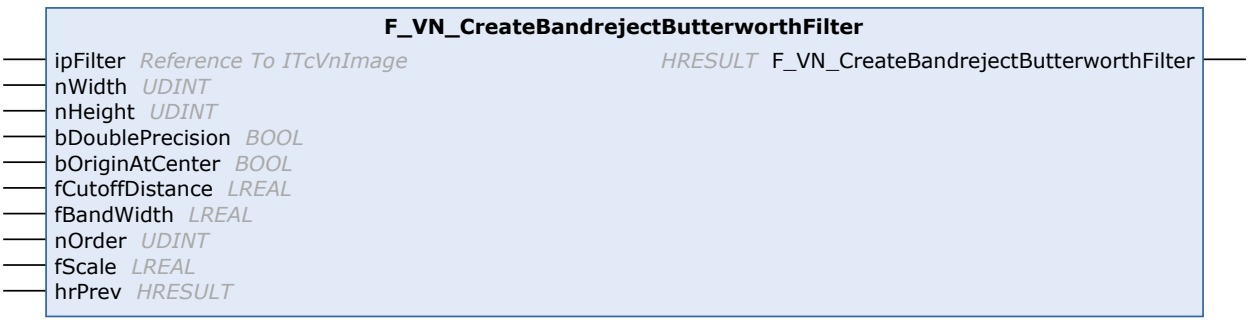
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.13.3 F_VN_CreateBandrejectButterworthFilter



Creates a bandreject Butterworth filter, which can be applied to a frequency domain image by element-wise multiplication.

语法

定义：

```
FUNCTION F_VN_CreateBandrejectButterworthFilter : HRESULT
VAR_INPUT
    ipFilter         : Reference To ITcVnImage;
    nWidth           : UDINT;
    nHeight          : UDINT;
    bDoublePrecision : BOOL;
    bOriginAtCenter  : BOOL;
    fCutoffDistance  : LREAL;
    fBandWidth       : LREAL;
    nOrder           : UDINT;
    fScale           : LREAL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipFilter	引用 ITcVnImage [▶ 407]	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	UDINT	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	UDINT	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	LREAL	巴特沃斯（Butterworth）滤波器的截断距离（> 0）。
fBandWidth	LREAL	巴特沃斯（Butterworth）滤波器的带宽（> 0）。
nOrder	UDINT	巴特沃斯（Butterworth）滤波器的阶数（> 0）。
fScale	LREAL	巴特沃斯（Butterworth）滤波器分母项的比例系数（> 0，例如 0.414 或 1.0）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

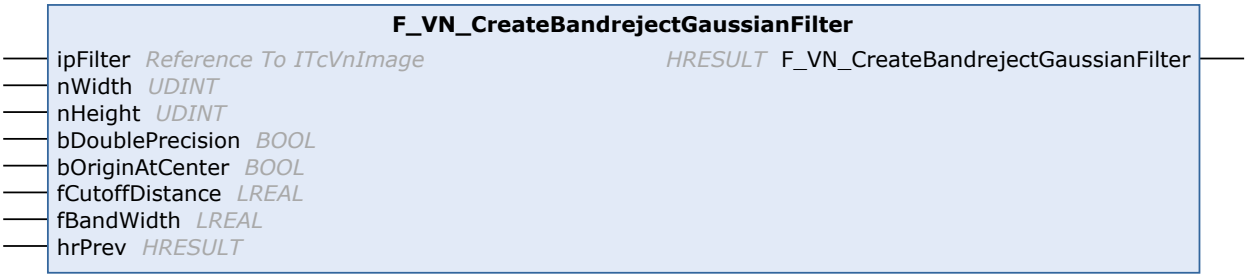
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.13.4 F_VN_CreateBandrejectGaussianFilter



Creates a bandreject Gaussian filter, which can be applied to a frequency domain image by element-wise multiplication.

语法

定义：

```
FUNCTION F_VN_CreateBandrejectGaussianFilter : HRESULT
VAR_INPUT
    ipFilter         : Reference To ITcVnImage;
    nWidth           : UDINT;
    nHeight          : UDINT;
    bDoublePrecision : BOOL;
    bOriginAtCenter  : BOOL;
    fCutoffDistance  : LREAL;
    fBandWidth       : LREAL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipFilter	引用 ITcVnImage [▶ 407]	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	UDINT	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	UDINT	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	LREAL	高斯（Gaussian）滤波器的截断距离（> 0）。
fBandWidth	LREAL	高斯（Gaussian）滤波器的带宽（> 0）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

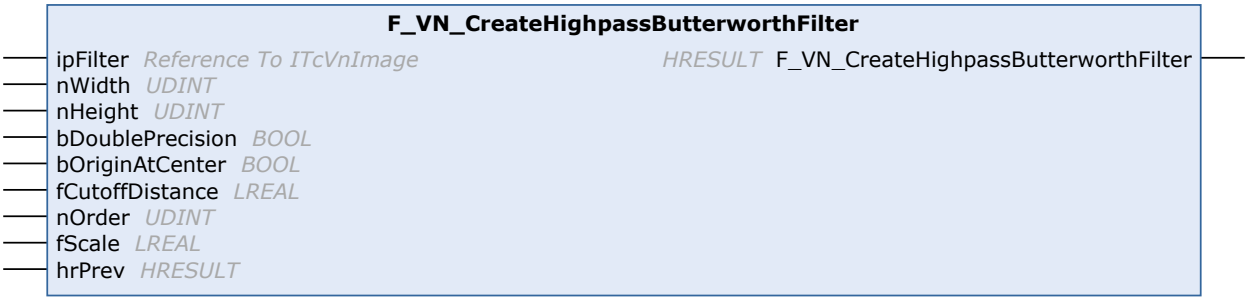
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.13.5 F_VN_CreateHighpassButterworthFilter



Creates a highpass Butterworth filter, which can be applied to a frequency domain image by element-wise multiplication.

语法

定义：

```
FUNCTION F_VN_CreateHighpassButterworthFilter : HRESULT
VAR_INPUT
    ipFilter         : Reference To ITcVnImage;
    nWidth           : UDINT;
    nHeight          : UDINT;
    bDoublePrecision : BOOL;
    bOriginAtCenter  : BOOL;
    fCutoffDistance  : LREAL;
    nOrder           : UDINT;
    fScale           : LREAL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipFilter	引用 ITcVnImage [▶ 407]	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	UDINT	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	UDINT	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	LREAL	巴特沃斯（Butterworth）滤波器的截断距离（> 0）。
nOrder	UDINT	巴特沃斯（Butterworth）滤波器的阶数（> 0）。
fScale	LREAL	巴特沃斯（Butterworth）滤波器分母项的比例系数（> 0，例如 0.414 或 1.0）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

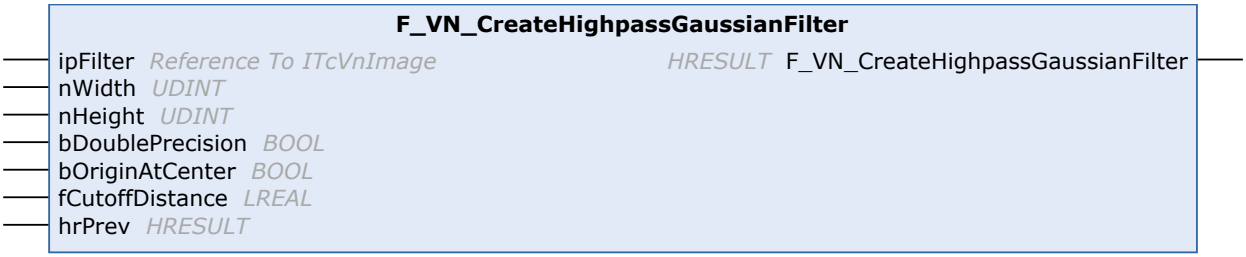
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.13.6 F_VN_CreateHighpassGaussianFilter



Creates a highpass Gaussian filter, which can be applied to a frequency domain image by element-wise multiplication.

语法

定义：

```
FUNCTION F_VN_CreateHighpassGaussianFilter : HRESULT
VAR_INPUT
    ipFilter         : Reference To ITcVnImage;
    nWidth           : UDINT;
    nHeight          : UDINT;
    bDoublePrecision : BOOL;
    bOriginAtCenter  : BOOL;
    fCutoffDistance  : LREAL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipFilter	引用 ITcVnImage [▶ 407]	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	UDINT	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	UDINT	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	LREAL	高斯（Gaussian）滤波器的截断距离（> 0）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

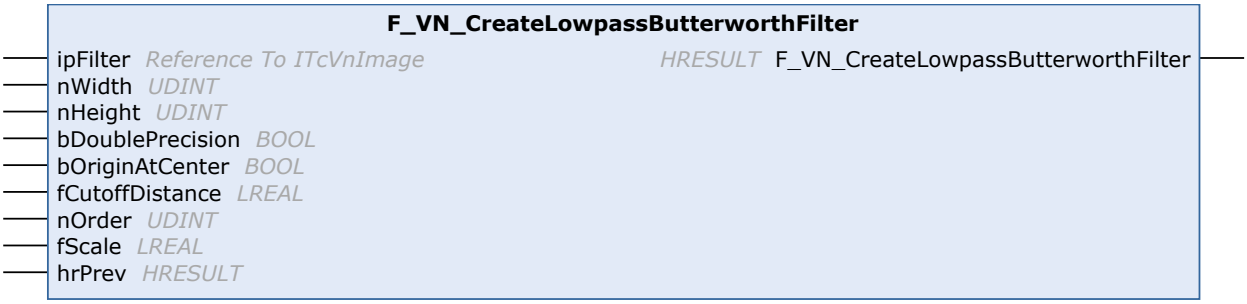
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.13.7 F_VN_CreateLowpassButterworthFilter



Creates a lowpass Butterworth filter, which can be applied to a frequency domain image by element-wise multiplication.

语法

定义：

```
FUNCTION F_VN_CreateLowpassButterworthFilter : HRESULT
VAR_INPUT
    ipFilter         : Reference To ITcVnImage;
    nWidth           : UDINT;
    nHeight          : UDINT;
    bDoublePrecision : BOOL;
    bOriginAtCenter  : BOOL;
    fCutoffDistance  : LREAL;
    nOrder           : UDINT;
    fScale           : LREAL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipFilter	引用 ITcVnImage [▶ 407]	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	UDINT	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	UDINT	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	LREAL	巴特沃斯（Butterworth）滤波器的截断距离（> 0）。
nOrder	UDINT	巴特沃斯（Butterworth）滤波器的阶数（> 0）。
fScale	LREAL	巴特沃斯（Butterworth）滤波器分母项的比例系数（> 0，例如 0.414 或 1.0）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

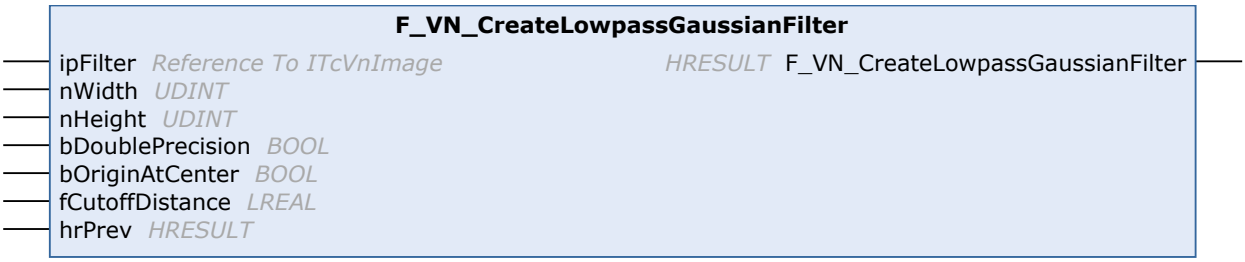
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.13.8 F_VN_CreateLowpassGaussianFilter



Creates a lowpass Gaussian filter, which can be applied to a frequency domain image by element-wise multiplication.

语法

定义：

```
FUNCTION F_VN_CreateLowpassGaussianFilter : HRESULT
VAR_INPUT
    ipFilter         : Reference To ITcVnImage;
    nWidth           : UDINT;
    nHeight          : UDINT;
    bDoublePrecision : BOOL;
    bOriginAtCenter  : BOOL;
    fCutoffDistance  : LREAL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipFilter	引用 ITcVnImage [▶ 407]	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	UDINT	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	UDINT	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	LREAL	高斯（Gaussian）滤波器的截断距离（> 0）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

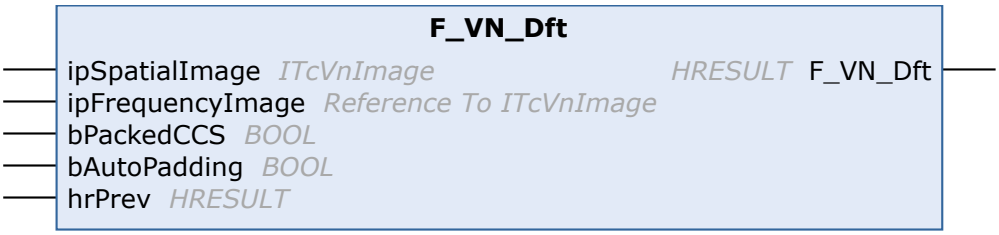
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.13.9 F_VN_Dft



Computes the DFT for a given image.

语法

定义：

```
FUNCTION F_VN_Dft : HRESULT
VAR_INPUT
    ipSpatialImage   : ITcVnImage;
    ipFrequencyImage : Reference To ITcVnImage;
    bPackedCCS       : BOOL;
    bAutoPadding     : BOOL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSpatialImage	ITcVnImage [▶ 407]	源图像（REAL 或 LREAL 类型的元素，1（Re）或 2（Re + Im）通道）
ipFrequencyImage	引用 ITcVnImage [▶ 407]	目标图像（类型与 ipSpatialImage 相同，但通道数量和大小可能因 bPackedCCS 和 bAutoPadding 而异）。
bPackedCCS	BOOL	如果为 true，且 ipSpatialImage 只有 1 个通道，则结果图像将只有 1 个通道，并带有打包的复合共轭对称形式结果。否则，结果图像将有 2 个独立通道（Re + Im），包含整个光谱。
bAutoPadding	BOOL	如果为 true，输入图像会自动填充（0）到最佳尺寸（如果需要），以加快 dft 的速度（创建临时副本，使 ipSpatialImage 的内容保持不变，这也需要一些额外的计算能力。因此，建议将有填充的执行时间和无填充的执行时间进行比较）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.13.10 F_VN_InverseDft



Compute the inverse DFT for a given frequency image.

语法

定义：

```
FUNCTION F_VN_InverseDft : HRESULT
VAR_INPUT
    ipFrequencyImage : ITcVnImage;
    ipSpatialImage   : Reference To ITcVnImage;
    bRealOutput      : BOOL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipFrequencyImage	ITcVnImage [▶ 407]	源频率图像（REAL 或 LREAL 类型的元素，1（打包 CCS）或 2（Re + Im）通道）
ipSpatialImage	引用 ITcVnImage [▶ 407]	目标图像（类型与 ipFrequencyImage 相同，但通道数量可因 bRealOutput 而异。如有需要，将创建适当的图像）。
bRealOutput	BOOL	仅当 ipFrequencyImage 有 2 个通道时才相关。如果为 true，结果图像将只有 1 个通道。否则，结果图像将有 2 个独立的通道（Re + Im）。如果 ipFrequencyImage 只有 1 个通道，则假定为打包 CCS 格式，结果图像将始终只有 1 个通道。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

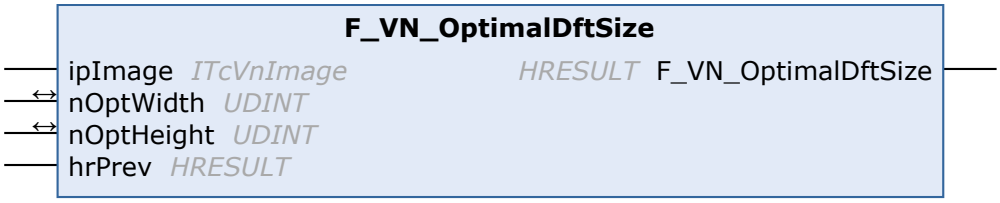
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.13.11 F_VN_OptimalDftSize



Computes the optimal image size for applying a dft (can lead to better performance).

语法

定义：

```
FUNCTION F_VN_OptimalDftSize : HRESULT
VAR_INPUT
    ipImage    : ITcVnImage;
END_VAR
VAR_IN_OUT
    nOptWidth  : UDINT;
    nOptHeight : UDINT;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	为其计算最佳宽度和高度的源图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nOptWidth	UDINT	返回图像的最佳宽度
nOptHeight	UDINT	返回图像的最佳高度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14 几何变换和坐标变换

6.1.4.14.1 F_VN_MergeRegions



Merge rectangles that overlap or whose edge-to-edge gap is within the specified distance.

Syntax

Definition:

```
FUNCTION F_VN_MergeRegions : HRESULT
VAR_INPUT
    ipSrcContainer  : ITcVnContainer;
    ipDestContainer : Reference To ITcVnContainer;
    nDistance       : UDINT;
    nOptions        : ULINT;
    hrPrev          : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipSrcContainer	ITcVnContainer [▶ 365]	Source container containing rectangles to be merged (ContainerType_Vector_TcVnRectangle_DINT)
ipDestContainer	Reference To ITcVnContainer [▶ 365]	Returns the resulting container, same type as ipSrcContainer
nDistance	UDINT	Maximum edge-to-edge gap (in pixels) for two non-overlapping rectangles to be merged (overlapping rectangles are always merged)
nOptions	ULINT	Specifies the method for merging regions. Valid values are defined in ETcVnMergeRegionsOptions.
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

还请参阅有关此

- ETcVnMergeRegionsOptions [▶ 156]
- F_VN_DetectCodesNeuralNetwork [▶ 839]

6.1.4.14.2 标定

6.1.4.14.2.1 F_VN_SortAxisAlignedPatternPoints



Sorts the points of a horizontally aligned asymmetric or symmetric calibration pattern in place. Rows top to bottom (ascending y), and points within each row left to right (ascending x). Individual pattern rows must not have overlapping y-coordinates.

Syntax

Definition:

```
FUNCTION F_VN_SortAxisAlignedPatternPoints : HRESULT
VAR_INPUT
    ipImagePoints : ITcVnContainer;
    nPatternRows  : UDINT;
    nPatternCols  : UDINT;
    hrPrev        : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipImagePoints	ITcVnContainer [▶ 365]	Detected pattern point positions in the image coordinate system (will be sorted by this function, ContainerType_Vector_TcVnPoint2_REAL)
nPatternRows	UDINT	Number of pattern rows
nPatternCols	UDINT	Number of pattern columns
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

Return value

HRESULT [▶ 135]

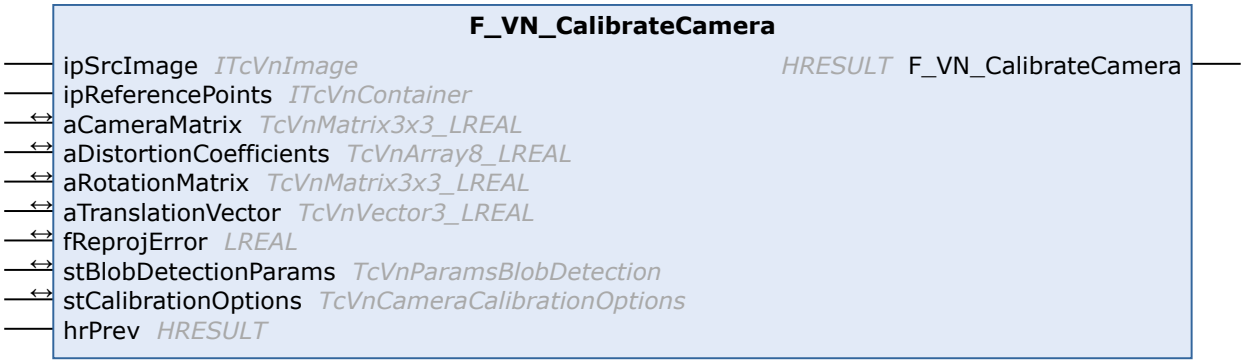
Required License

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.2 F_VN_CalibrateCamera



Compute the camera parameters (intrinsic + extrinsic) by evaluating an image containing a calibration pattern (circles).
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_CalibrateCamera : HRESULT
VAR_INPUT
    ipSrcImage              : ITcVnImage;
    ipReferencePoints       : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aRotationMatrix         : TcVnMatrix3x3_LREAL;
    aTranslationVector      : TcVnVector3_LREAL;
    fReprojError            : LREAL;
    stBlobDetectionParams   : TcVnParamsBlobDetection;
    stCalibrationOptions    : TcVnCameraCalibrationOptions;
END_VAR
VAR_INPUT
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	包含标定板的源图像（1 通道，USINT 元素）
ipReferencePoints	ITcVnContainer [▶ 365]	标定板参考点的位置（ContainerType_Vector_TcVnPoint3_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	返回镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	返回平移向量
fReprojError	LREAL	返回重投影误差

名称	类型	描述
stBlobDetectionParams	TcVnParamsBlobDetection [▶ 229]	用于内部使用的 F_VN_DetectBlobs 函数的参数
stCalibrationOptions	TcVnCameraCalibrationOptions [▶ 222]	标定选项

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_CalibrateCamera根据包含校准图案的输入图像计算相机的内在和外在参数。校准图案的元素必须可检测为斑点。

输入图像

输入图像ipSrcImage必须是位深为 8 位的单通道灰度图像。该图像必须包含一个与参考点ipReferencePoints相匹配的带圆圈的校准图案。该函数期望在白色背景上搭配黑色填充圆圈。为了反转图像，可以使用斑点函数的ETcVnThresholdType = TCVN_TT_BINARY_INV。

专家版F_VN_CalibrateCameraExp [▶ 1073]需要使用多个输入图像进行校准。

参考点

容器ipReferencePoints必须包含输入图像上说明的参考点ipSrcImage，作为校准模式。这些点必须被指定为类型为REAL的 3 维点，以便容器类型为ContainerType_Vector_TcVnPoint3_REAL。

参考点可按以下方式产生：

- 手动创建一个 XML 文件，并用功能块FB_VN_ReadCalibrationPattern [▶ 1586]读取。
- 校准助手 [▶ 106]中的配置，通过点击**写入结果**保存在 TcCOM 对象中，并使用功能块FB_VN_GevCameraControl [▶ 1609]的方法GetCalibPatternRef [▶ 1615]读取。设定的中心点被考虑在内。
- 将参考点单独插入一个容器中。

相机参数的矩阵

这些矩阵返回相机模型的计算参数。然后，它们可用于补偿客观畸变，并用于像素坐标和全局坐标之间的转换。

- 相机矩阵
- 畸变系数
- 旋转矩阵
- 平移矢量

舍入误差

由于舍入误差，计算出的参数值可能与校准助手中的计算值略有偏差。

再投影误差

斑点检测的参数

没有棋盘图案

由于参考点被检测为一个斑点结构，因此无法使用棋盘图案。

校准选项

默认设置

如需从校准向导 [▶ 106]获得默认设置，位bFixAspectRatio、bFixPrincipalPoint、bFixK5、bFixK6和bRationalModel必须设置为 TRUE。

bFixPrincipalPoint = “修复中心点”

该位bFixPrincipalPoint对应于相机校准助手中的“修复中心点”选项。

样本

- 校准助手 [▶ 2822]

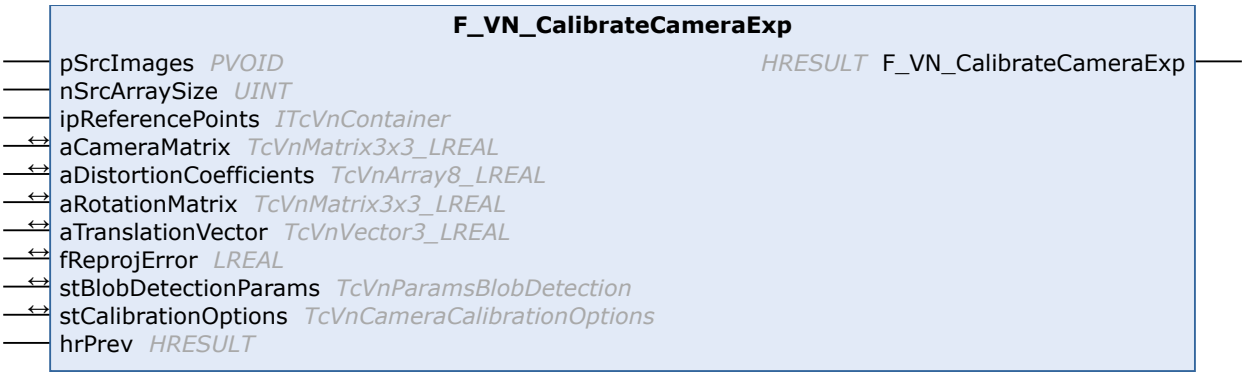
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.3 F_VN_CalibrateCameraExp



Compute the camera parameters (intrinsic + extrinsic) by evaluating images containing a calibration pattern (circles). The extrinsic parameters are computed for the first image in the array. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_CalibrateCameraExp : HRESULT
VAR_INPUT
    pSrcImages              : PVOID;
    nSrcArraySize           : UINT;
    ipReferencePoints       : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
```

```
    aRotationMatrix          : TcVnMatrix3x3_LREAL;
    aTranslationVector       : TcVnVector3_LREAL;
    fReprojError             : LREAL;
    stBlobDetectionParams    : TcVnParamsBlobDetection;
    stCalibrationOptions     : TcVnCameraCalibrationOptions;
END_VAR
VAR_INPUT
    hrPrev                   : HRESULT;
END_VAR
```

输入

名称	类型	描述
pSrcImages	PVOID	指向源图像数组的指针，每个源图像包含相同的标定板（1 通道，USINT 元素）
nSrcArraySize	UINT	pSrcImages 数组大小
ipReferencePoints	ITcVnContainer [▶ 365]	标定板参考点的位置（ContainerType_Vector_TcVnPoint3_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	返回镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	返回平移向量
fReprojError	LREAL	返回重投影误差
stBlobDetectionParams	TcVnParamsBlobDetection [▶ 229]	用于内部使用的 F_VN_DetectBlobs 函数的参数
stCalibrationOptions	TcVnCameraCalibrationOptions [▶ 222]	标定选项

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CalibrateCameraExp` 是 F_VN_CalibrateCamera [▶ 1071] 的专家版本。它可以使用多个输入图像来标定相机。

参数

输入图像

在指针 `pSrcImages` 和图像数量 `nSrcArraySize` 的帮助下，将用于相机标定的输入图像传递给函数。在其他方面，各个图像必须满足与标准函数相同的要求。

注意

指针处理

确保指针定义和数组大小与实际数据相符。否则，可能会发生不允许的内存访问，导致系统崩溃。

参考点

容器ipReferencePoints必须包含输入图像上说明的参考点ipSrcImage，作为校准模式。这些点必须被指定为类型为REAL的 3 维点，以便容器类型为ContainerType_Vector_TcVnPoint3_REAL。

参考点可按以下方式产生：

- 手动创建一个 XML 文件，并用功能块FB_VN_ReadCalibrationPattern [▶ 1586]读取。
- 校准助手 [▶ 106]中的配置，通过点击**写入结果**保存在 TcCOM 对象中，并使用功能块 FB_VN_GevCameraControl [▶ 1609]的方法GetCalibPatternRef [▶ 1615]读取。设定的中心点被考虑在内。
- 将参考点单独插入一个容器中。

相机参数的矩阵

这些矩阵返回相机模型的计算参数。然后，它们可用于补偿客观畸变，并用于像素坐标和全局坐标之间的转换。

- 相机矩阵
- 畸变系数
- 旋转矩阵
- 平移矢量

舍入误差

由于舍入误差，计算出的参数值可能与校准助手中的计算值略有偏差。

再投影误差

斑点检测的参数

没有棋盘图案

由于参考点被检测为一个斑点结构，因此无法使用棋盘图案。

校准选项

默认设置

如需从校准向导 [▶ 106]获得默认设置，位bFixAspectRatio、bFixPrincipalPoint、bFixK5、bFixK6和bRationalModel必须设置为 TRUE。

bFixPrincipalPoint = “修复中心点”

该位bFixPrincipalPoint对应于相机校准助手中的“修复中心点”选项。

应用

样本

- 校准助手 [▶ 2822]

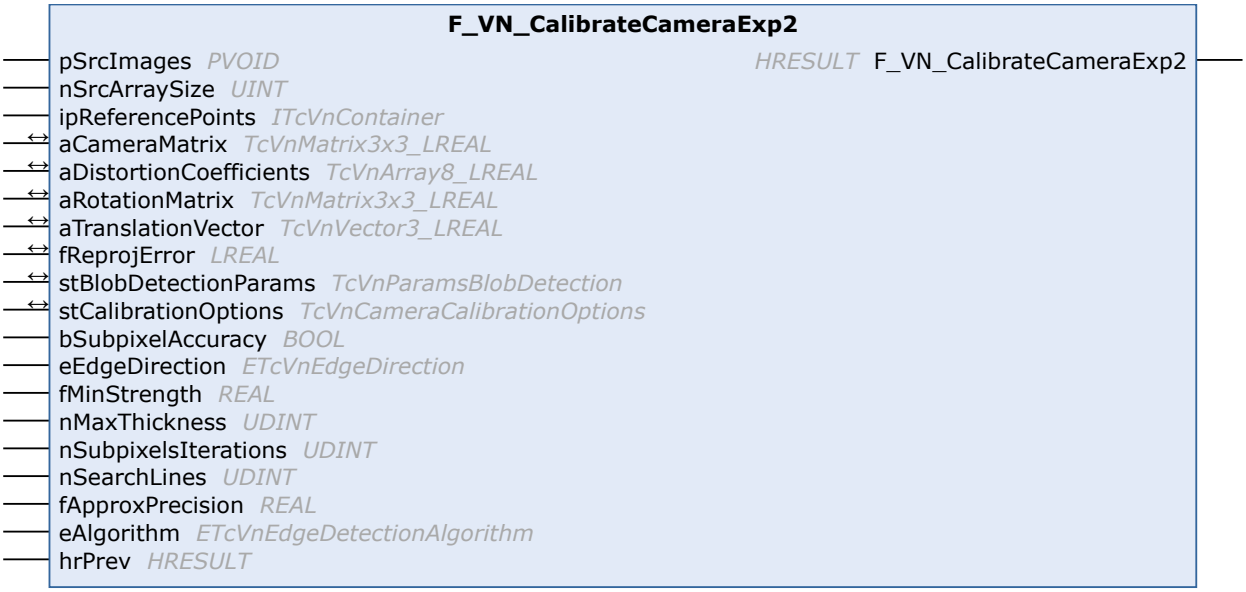
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.4 F_VN_CalibrateCameraExp2



Compute the camera parameters (intrinsic + extrinsic) by evaluating images containing a calibration pattern (circles). The extrinsic parameters are computed for the first image in the array. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_CalibrateCameraExp2 : HRESULT
VAR_INPUT
    pSrcImages              : PVOID;
    nSrcArraySize           : UINT;
    ipReferencePoints       : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aRotationMatrix         : TcVnMatrix3x3_LREAL;
    aTranslationVector      : TcVnVector3_LREAL;
    fReprojError            : LREAL;
    stBlobDetectionParams   : TcVnParamsBlobDetection;
    stCalibrationOptions    : TcVnCameraCalibrationOptions;
END_VAR
VAR_INPUT
    bSubpixelAccuracy       : BOOL;
    eEdgeDirection          : ETcVnEdgeDirection;
    fMinStrength            : REAL;
    nMaxThickness           : UDINT;
    nSubpixelsIterations    : UDINT;
    nSearchLines            : UDINT;
    fApproxPrecision        : REAL;
    eAlgorithm              : ETcVnEdgeDetectionAlgorithm;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
pSrcImages	PVOID	指向源图像数组的指针，每个源图像包含相同的标定板（1 通道，USINT 元素）

名称	类型	描述
nSrcArraySize	UINT	pSrcImages 数组大小
ipReferencePoints	ITcVnContainer [▶ 365]	标定板参考点的位置（ContainerType_Vector_TcVnPoint3_REAL）
bSubpixelAccuracy	BOOL	如果为 true，则以亚像素精度检测图案点
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向（从中心到椭圆外侧）
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
nSearchLines	UDINT	指定搜索线的数量，在所有方向上平均分配（必须 >= 8 且是 4 的倍数）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	返回镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	返回平移向量
fReprojError	LREAL	返回重投影误差
stBlobDetectionParams	TcVnParamsBlobDetection [▶ 229]	用于内部使用的 F_VN_DetectBlobs 函数的参数
stCalibrationOptions	TcVnCameraCalibrationOptions [▶ 222]	标定选项

返回值

HRESULT [▶ 135]

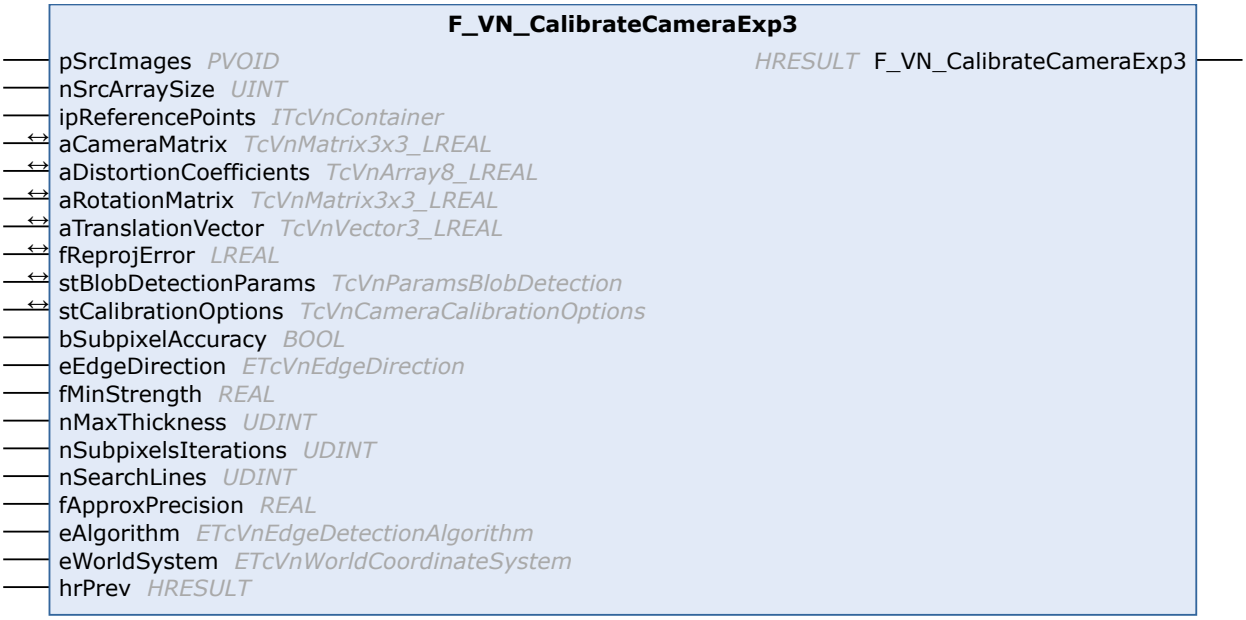
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.5 F_VN_CalibrateCameraExp3



Compute the camera parameters (intrinsic + extrinsic) by evaluating images containing a calibration pattern (circles). The extrinsic parameters are computed for the first image in the array. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_CalibrateCameraExp3 : HRESULT
VAR_INPUT
    pSrcImages              : PVOID;
    nSrcArraySize           : UINT;
    ipReferencePoints       : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aRotationMatrix         : TcVnMatrix3x3_LREAL;
    aTranslationVector      : TcVnVector3_LREAL;
    fReprojError            : LREAL;
    stBlobDetectionParams   : TcVnParamsBlobDetection;
    stCalibrationOptions    : TcVnCameraCalibrationOptions;
END_VAR
VAR_INPUT
    bSubpixelAccuracy       : BOOL;
    eEdgeDirection          : ETcVnEdgeDirection;
    fMinStrength            : REAL;
    nMaxThickness           : UDINT;
    nSubpixelsIterations    : UDINT;
    nSearchLines            : UDINT;
    fApproxPrecision        : REAL;
    eAlgorithm              : ETcVnEdgeDetectionAlgorithm;
    eWorldSystem            : ETcVnWorldCoordinateSystem;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
pSrcImages	PVOID	指向源图像数组的指针，每个源图像包含相同的标定板（1 通道，USINT 元素）
nSrcArraySize	UINT	pSrcImages 数组大小
ipReferencePoints	ITcVnContainer [▶ 365]	标定板参考点的位置（ContainerType_Vector_TcVnPoint3_REAL）
bSubpixelAccuracy	BOOL	如果为 true，则以亚像素精度检测图案点
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向（从中心到椭圆外侧）
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
nSearchLines	UDINT	指定搜索线的数量，在所有方向上平均分配（必须 >= 8 且是 4 的倍数）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
eWorldSystem	ETcVnWorldCoordinateSystem [▶ 215]	世界坐标系方向
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	返回镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	返回平移向量
fReprojError	LREAL	返回重投影误差
stBlobDetectionParams	TcVnParamsBlobDetection [▶ 229]	用于内部使用的 F_VN_DetectBlobs 函数的参数
stCalibrationOptions	TcVnCameraCalibrationOptions [▶ 222]	标定选项

返回值

HRESULT [▶ 135]

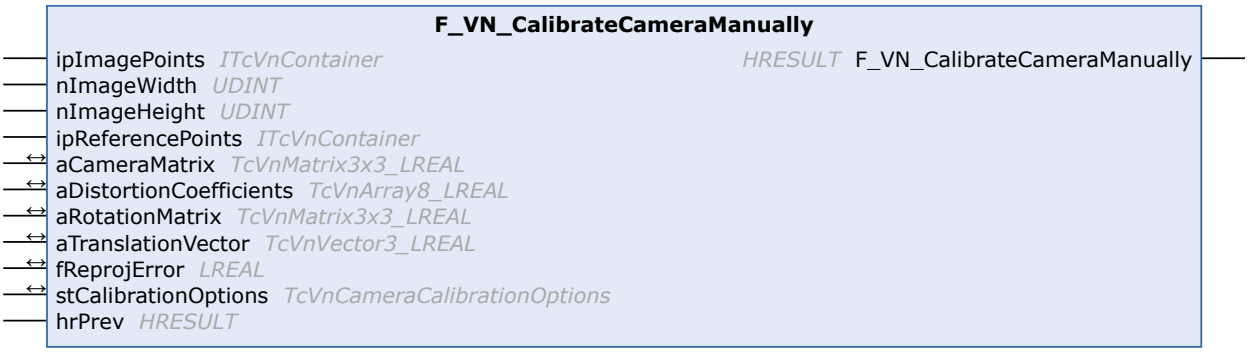
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.6 F_VN_CalibrateCameraManually



Compute the camera parameters (intrinsic + extrinsic). The extrinsic parameters are computed for the first inner container.

语法

定义：

```
FUNCTION F_VN_CalibrateCameraManually : HRESULT
VAR_INPUT
    ipImagePoints           : ITcVnContainer;
    nImageWidth             : UDINT;
    nImageHeight            : UDINT;
    ipReferencePoints       : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aRotationMatrix         : TcVnMatrix3x3_LREAL;
    aTranslationVector      : TcVnVector3_LREAL;
    fReprojError            : LREAL;
    stCalibrationOptions    : TcVnCameraCalibrationOptions;
END_VAR
VAR_INPUT
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImagePoints	ITcVnContainer [▶ 365]	提供多个图像的图像点（ContainerType_Vector_Vector_TcVnPoint2_REAL，每个内部容器至少 6 个点）。容器和点的阶数必须与 ipReferencePoints 匹配。
nImageWidth	UDINT	图像宽度
nImageHeight	UDINT	图像高度

名称	类型	描述
ipReferencePoints	ITcVnContainer [▶ 365]	世界参考点（ContainerType_Vector_Vector_TcVnPoint3_REAL）。内部容器的数量及其点数必须与 ipImagePoints 匹配。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	返回镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回（第一个内部容器的）旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	返回（第一个内部容器的）平移向量
fReprojError	LREAL	返回重投影误差
stCalibrationOptions	TcVnCameraCalibrationOptions [▶ 222]	标定选项

返回值

HRESULT [▶ 135]

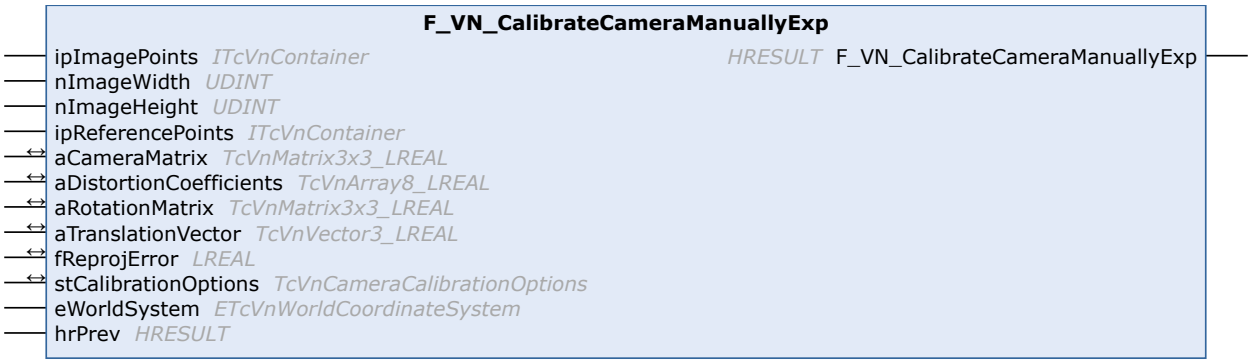
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.7 F_VN_CalibrateCameraManuallyExp



Compute the camera parameters (intrinsic + extrinsic). The extrinsic parameters are computed for the first inner container. (expert function)

语法

定义：

```
FUNCTION F_VN_CalibrateCameraManuallyExp : HRESULT
VAR_INPUT
    ipImagePoints           : ITcVnContainer;
    nImageWidth             : UDINT;
    nImageHeight            : UDINT;
    ipReferencePoints       : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aRotationMatrix         : TcVnMatrix3x3_LREAL;
    aTranslationVector      : TcVnVector3_LREAL;
    fReprojError            : LREAL;
    stCalibrationOptions    : TcVnCameraCalibrationOptions;
END_VAR
VAR_INPUT
    eWorldSystem            : ETcVnWorldCoordinateSystem;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImagePoints	ITcVnContainer [▶ 365]	提供多个图像的图像点（ContainerType_Vector_Vector_TcVnPoint2_REAL，每个内部容器至少 6 个点）。容器和点的阶数必须与 ipReferencePoints 匹配。
nImageWidth	UDINT	图像宽度
nImageHeight	UDINT	图像高度
ipReferencePoints	ITcVnContainer [▶ 365]	世界参考点（ContainerType_Vector_Vector_TcVnPoint3_REAL）。内部容器的数量及其点数必须与 ipImagePoints 匹配。
eWorldSystem	ETcVnWorldCoordinateSystem [▶ 215]	世界坐标系方向
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	返回镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回（第一个内部容器的）旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	返回（第一个内部容器的）平移向量
fReprojError	LREAL	返回重投影误差
stCalibrationOptions	TcVnCameraCalibrationOptions [▶ 222]	标定选项

返回值

HRESULT [▶ 135]

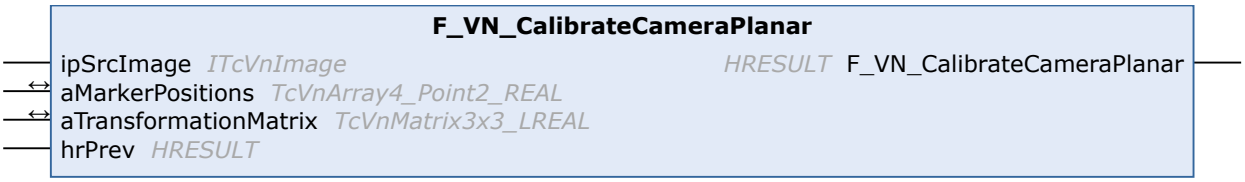
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.8 F_VN_CalibrateCameraPlanar



Calibrate camera using a planar calibration pattern comprised of four circles marking the corners of a rectangle.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_CalibrateCameraPlanar : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
END_VAR
VAR_IN_OUT
    aMarkerPositions      : TcVnArray4_Point2_REAL;
    aTransformationMatrix : TcVnMatrix3x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	二值源图像（背景为零，标记非零）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMarkerPositions	TcVnArray4_Point2_REAL [▶ 153]	在其平面内使用世界单位的标记位置（右侧坐标系）
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回一个 3×3 矩阵，用于将图像坐标转换为参考坐标系

返回值

HRESULT [▶ 135]

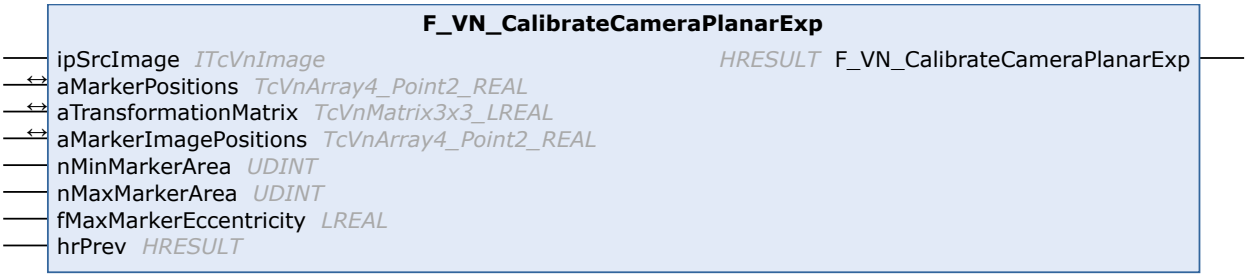
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.9 F_VN_CalibrateCameraPlanarExp



Calibrate camera using a planar calibration pattern comprised of four circles marking the corners of a rectangle. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_CalibrateCameraPlanarExp : HRESULT
VAR_INPUT
    ipSrcImage             : ITcVnImage;
END_VAR
VAR_IN_OUT
    aMarkerPositions       : TcVnArray4_Point2_REAL;
    aTransformationMatrix  : TcVnMatrix3x3_LREAL;
    aMarkerImagePositions  : TcVnArray4_Point2_REAL;
END_VAR
VAR_INPUT
    nMinMarkerArea         : UDINT;
    nMaxMarkerArea         : UDINT;
    fMaxMarkerEccentricity : LREAL;
    hrPrev                 : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	二值源图像（背景为零，标记非零）
nMinMarkerArea	UDINT	标记检测的最小面积（较小的标记将被忽略。）
nMaxMarkerArea	UDINT	标记检测的最面积（较大的标记将被忽略）。
fMaxMarkerEccentricity	LREAL	用于标记检测的最大偏心率（偏心率用于测量偏离圆形的程度（偏心率 = 0：圆形物体；偏心率 = 1.0：线形物体）。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMarkerPositions	TcVnArray4_Point2_REAL [▶ 153]	在其平面内使用世界单位的标记位置（右侧坐标系）
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回一个 3×3 矩阵，用于将图像坐标转换为参考坐标系

名称	类型	描述
aMarkerImage Positions	TcVnArray4_Point2_REAL [▶ 153]	返回标记位置的图像坐标（左侧坐标系）

返回值

HRESULT [▶ 135]

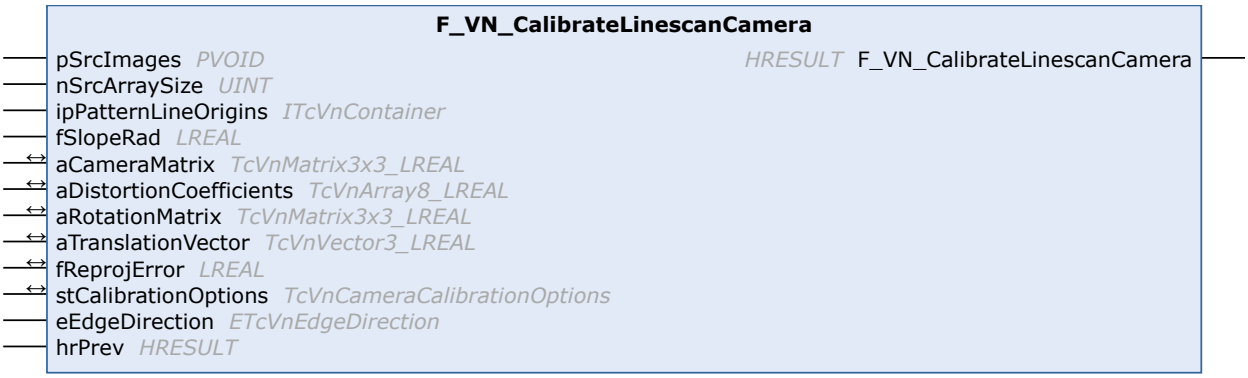
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.10 F_VN_CalibrateLinescanCamera



Compute the camera parameters (intrinsic + extrinsic) by evaluating images containing a line calibration pattern. The pattern must consist of alternating vertical and diagonal lines, with each diagonal having the same constant slope: |\|\|...\|. The results are only valid for the x-direction of a line scan image. To get accurate x-values for a transformation into world coordinates, the y-coordinate must be set to half the sensor width and so equal to the cy-value of the camera matrix. For other y-values, the results are approximated and can be inaccurate. The extrinsic parameters are computed for the first image in the array.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_CalibrateLinescanCamera : HRESULT
VAR_INPUT
    pSrcImages              : PVOID;
    nSrcArraySize           : UINT;
    ipPatternLineOrigins    : ITcVnContainer;
    fSlopeRad               : LREAL;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aRotationMatrix         : TcVnMatrix3x3_LREAL;
    aTranslationVector      : TcVnVector3_LREAL;
    fReprojError            : LREAL;
    stCalibrationOptions    : TcVnCameraCalibrationOptions;
END_VAR
VAR_INPUT
    eEdgeDirection          : ETcVnEdgeDirection;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
pSrcImages	PVOID	指向包含 1 个或多个源图像（1 通道）的数组的指针。如果图像包含不止 1 行，则每行必须包含相同标定位置的图像。
nSrcArraySize	UINT	pSrcImages 数组大小
ipPatternLineOrigins	ITcVnContainer [▶ 365]	标定板线原点的 X 位置（ContainerType_Vector_REAL，垂直线和其后的对角线的原点通常相同）
fSlopeRad	LREAL	对角线相对于垂直线的斜率，单位：弧度（> 0，< PI/2）（即 0 为垂直线，PI/2 为水平线）
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	返回镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	返回平移向量
fReprojError	LREAL	返回重投影误差
stCalibrationOptions	TcVnCameraCalibrationOptions [▶ 222]	标定选项（建议设置 bFixAspectRatio、bFixPrincipalPoint 和 bZeroTangentDist）

返回值

HRESULT [▶ 135]

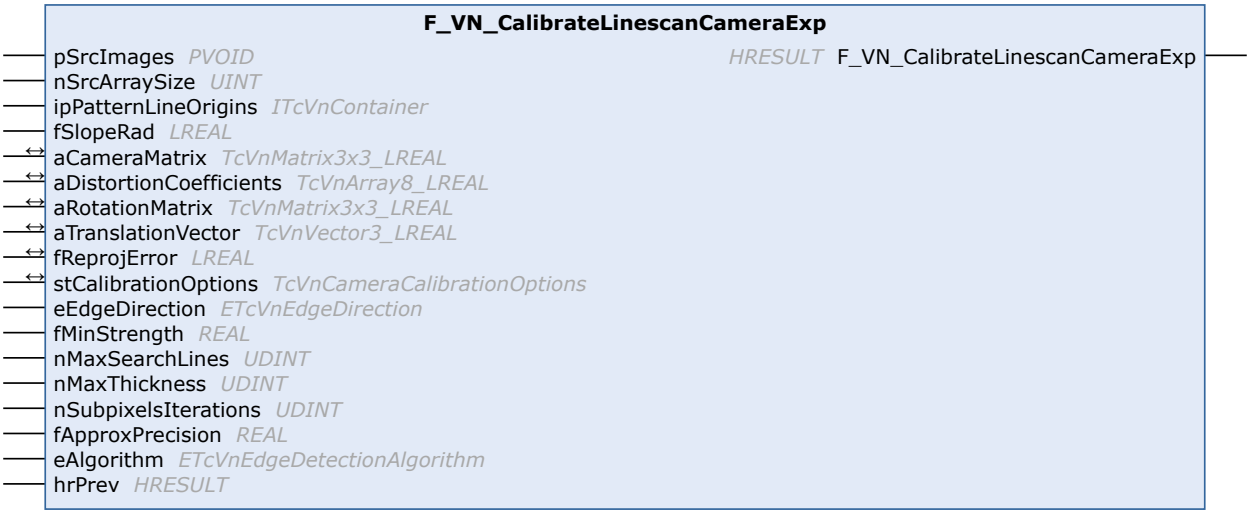
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.11 F_VN_CalibrateLinescanCameraExp



Compute the camera parameters (intrinsic + extrinsic) by evaluating images containing a line calibration pattern. The pattern must consist of alternating vertical and diagonal lines, with each diagonal having the same constant slope: |\|\|...\|. The results are only valid for the x-direction of a line scan image. To get accurate x-values for a transformation into world coordinates, the y-coordinate must be set to half the sensor width and so equal to the cy-value of the camera matrix. For other y-values, the results are approximated and can be inaccurate. The extrinsic parameters are computed for the first image in the array. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_CalibrateLinescanCameraExp : HRESULT
VAR_INPUT
    pSrcImages              : PVOID;
    nSrcArraySize           : UINT;
    ipPatternLineOrigins    : ITcVnContainer;
    fSlopeRad               : LREAL;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aRotationMatrix         : TcVnMatrix3x3_LREAL;
    aTranslationVector      : TcVnVector3_LREAL;
    fReprojError            : LREAL;
    stCalibrationOptions    : TcVnCameraCalibrationOptions;
END_VAR
VAR_INPUT
    eEdgeDirection          : ETcVnEdgeDirection;
    fMinStrength            : REAL;
    nMaxSearchLines         : UDINT;
    nMaxThickness           : UDINT;
    nSubpixelsIterations    : UDINT;
    fApproxPrecision        : REAL;
    eAlgorithm              : ETcVnEdgeDetectionAlgorithm;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
pSrcImages	PVOID	指向包含 1 个或多个源图像（1 通道）的数组的指针。如果图像包含不止 1 行，则每行必须包含相同标定位置的图像。
nSrcArraySize	UINT	pSrcImages 数组大小
ipPatternLineOrigins	ITcVnContainer [▶ 365]	标定板线原点的 X 位置（ContainerType_Vector_REAL，垂直线和其后的对角线的原点通常相同）
fSlopeRad	LREAL	对角线相对于垂直线的斜率，单位：弧度（> 0，< PI/2）（即 0 为垂直线，PI/2 为水平线）

名称	类型	描述
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nMaxSearchLines	UDINT	搜索线最大数量（在图像高度上平均分配，每行最多 1 条搜索线）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	返回镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	返回平移向量
fReprojError	LREAL	返回重投影误差
stCalibrationOptions	TcVnCameraCalibrationOptions [▶ 222]	标定选项（建议设置 bFixAspectRatio、bFixPrincipalPoint 和 bZeroTangentDist）

返回值

HRESULT [▶ 135]

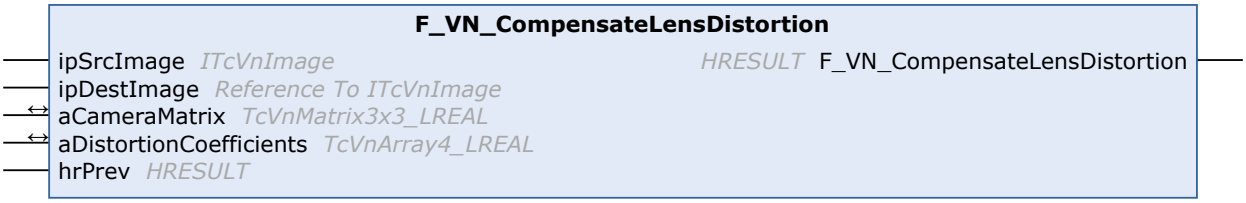
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.12 F_VN_CompensateLensDistortion



转换图像，以补偿镜头畸变。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_CompensateLensDistortion : HRESULT
VAR_INPUT
    ipSrcImage              : ITcVnImage;
    ipDestImage             : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray4_LREAL;
END_VAR
VAR_INPUT
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（必须与 ipSrcImage 不同！如有需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	包含内参的相机矩阵
aDistortionCoefficients	TcVnArray4_LREAL [▶ 153]	镜头畸变系数 [k1、k2、p1、p2]

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.1.4.14.2.13 F_VN_CompensateLensDistortionExp1



转换图像，以补偿镜头畸变。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_CompensateLensDistortionExp1 : HRESULT
VAR_INPUT
    ipSrcImage              : ITcVnImage;
    ipDestImage             : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
END_VAR
VAR_INPUT
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（必须与 ipSrcImage 不同！如有需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	包含内参的相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.1.4.14.2.14 F_VN_CompensateLensDistortionExp2



转换图像，以补偿镜头畸变。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_CompensateLensDistortionExp2 : HRESULT
VAR_INPUT
    ipSrcImage              : ITcVnImage;
    ipDestImage             : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aNewCameraMatrix        : TcVnMatrix3x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（必须与 ipSrcImage 不同！如有需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	包含内参的相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aNewCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	允许对结果图像进行额外的缩放和移动

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.1.4.14.2.15 F_VN_CompensateLensDistortionForPoints



Transforms point coordinates to compensate the lens distortion.

语法

定义：

```
FUNCTION F_VN_CompensateLensDistortionForPoints : HRESULT
VAR_INPUT
    ipSrcPoints             : ITcVnContainer;
    ipDestPoints            : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray4_LREAL;
END_VAR
```

```
VAR_INPUT
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPoints	ITcVnContainer [▶ 365]	包含源点坐标的容器（ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_DINT）
ipDestPoints	引用 ITcVnContainer [▶ 365]	返回包含转换后点坐标的容器（ContainerType_Vector_TcVnPoint2_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	包含内参的相机矩阵
aDistortionCoefficients	TcVnArray4_LREAL [▶ 153]	镜头畸变系数 [k1、k2、p1、p2]

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.16 F_VN_CompensateLensDistortionForPointsExp1



Transforms point coordinates to compensate the lens distortion. (expert function)

语法

定义：

```
FUNCTION F_VN_CompensateLensDistortionForPointsExp1 : HRESULT
VAR_INPUT
    ipSrcPoints             : ITcVnContainer;
    ipDestPoints            : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
END_VAR
```

```
VAR_INPUT
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPoints	ITcVnContainer [▶ 365]	包含源点坐标的容器（ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_DINT）
ipDestPoints	引用 ITcVnContainer [▶ 365]	返回包含转换后点坐标的容器（ContainerType_Vector_TcVnPoint2_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	包含内参的相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.17 F_VN_CompensateLensDistortionForPointsExp2



Transforms point coordinates to compensate the lens distortion. (expert function)

语法

定义：

```
FUNCTION F_VN_CompensateLensDistortionForPointsExp2 : HRESULT
VAR_INPUT
    ipSrcPoints             : ITcVnContainer;
    ipDestPoints            : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aNewCameraMatrix        : TcVnMatrix3x3_LREAL;
```

```
END_VAR
VAR_INPUT
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPoints	ITcVnContainer [▶ 365]	包含源点坐标的容器（ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_DINT）
ipDestPoints	引用 ITcVnContainer [▶ 365]	返回包含转换后点坐标的容器（ContainerType_Vector_TcVnPoint2_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	包含内参的相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aNewCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	允许对结果点坐标进行额外移动

返回值

HRESULT [▶ 135]

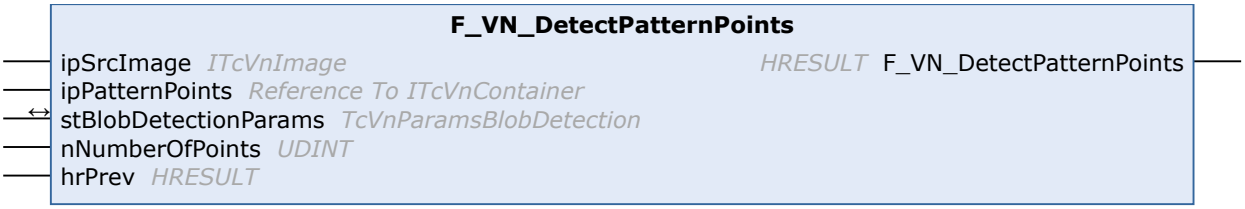
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.18 F_VN_DetectPatternPoints



Detects calibration pattern points (circles) within the provided image.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_DetectPatternPoints : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
    ipPatternPoints       : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    stBlobDetectionParams : TcVnParamsBlobDetection;
END_VAR
VAR_INPUT
    nNumberOfPoints       : UDINT;
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道）
ipPatternPoints	引用 ITcVnContainer [▶ 365]	返回标定板的点位置（ContainerType_Vector_TcVnPoint2_REAL；重复使用非零接口指针。）
nNumberOfPoints	UDINT	预期的图案点数
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stBlobDetectionParams	TcVnParamsBlobDetection [▶ 229]	用于检测和筛选轮廓的参数。

返回值

HRESULT [▶ 135]

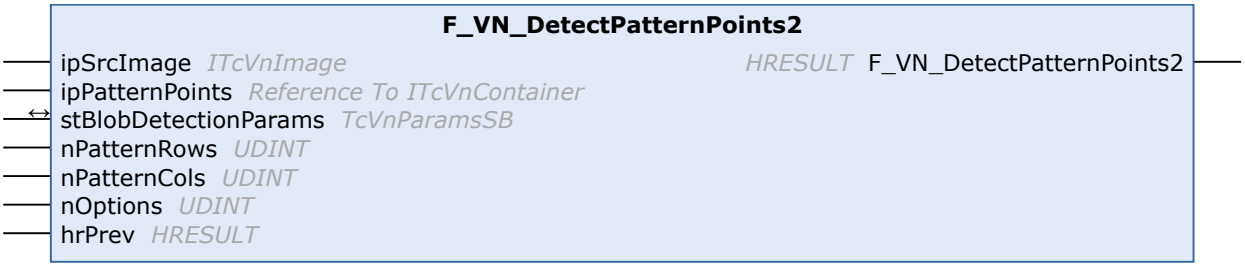
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.19 F_VN_DetectPatternPoints2



在提供的图像中检测标定板点（圆），并对其进行排序。
可以使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_DetectPatternPoints2 : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
    ipPatternPoints       : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    stBlobDetectionParams : TcVnParamsSB;
END_VAR
VAR_INPUT
    nPatternRows          : UDINT;
    nPatternCols          : UDINT;
    nOptions              : UDINT;
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT、1 或 3 通道）
ipPatternPoints	引用 ITcVnContainer [▶ 365]	返回标定板的点位置，按行从左到右排序（ContainerType_Vector_TcVnPoint2_REAL；重复使用非零接口指针。）
nPatternRows	UDINT	预期的图案行数
nPatternCols	UDINT	预期的图案列数
nOptions	UDINT	提供来自 ETcVnDetectPatternPointsOptions 的选项。始终需要选择 DPPO_SYMMETRIC 或 DPPO_ASYMMETRIC。可以选择通过按位 OR 与 DPPO_CLUSTERING 组合使用。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stBlobDetectionParams	TcVnParamsSB [▶ 234]	用于检测和筛选轮廓的参数。至少应为 fMinArea 和 fMaxArea 指定合适的值。

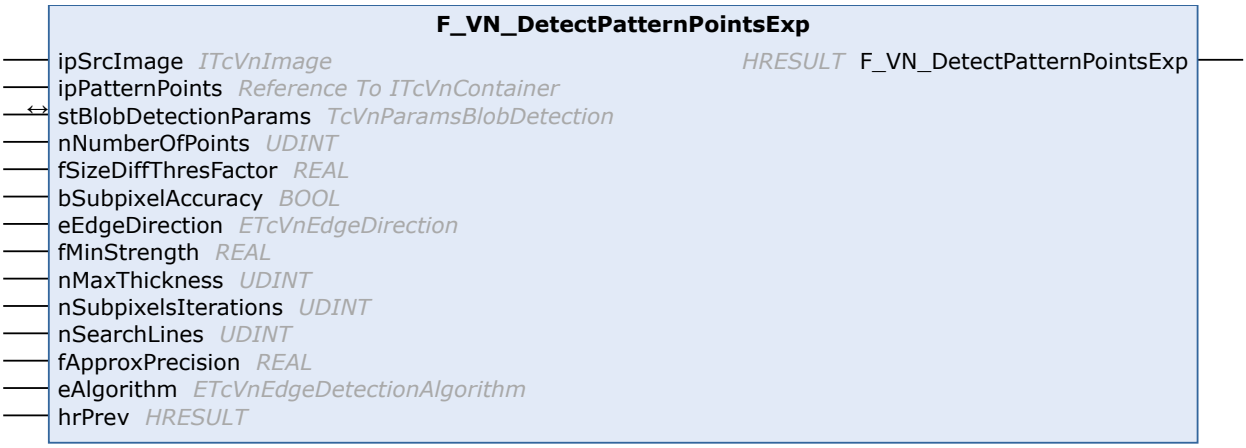
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.1.4.14.2.20 F_VN_DetectPatternPointsExp



在提供的图像中检测标定板点（圆）。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_DetectPatternPointsExp : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
    ipPatternPoints       : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    stBlobDetectionParams : TcVnParamsBlobDetection;
END_VAR
VAR_INPUT
    nNumberOfPoints       : UDINT;
    fSizeDiffThresFactor  : REAL;
    bSubpixelAccuracy     : BOOL;
    eEdgeDirection        : ETcVnEdgeDirection;
    fMinStrength          : REAL;
    nMaxThickness         : UDINT;
    nSubpixelsIterations  : UDINT;
    nSearchLines          : UDINT;
    fApproxPrecision      : REAL;
    eAlgorithm            : ETcVnEdgeDetectionAlgorithm;
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道）
ipPatternPoints	引用 ITcVnContainer [▶ 365]	返回标定板的点位置（ContainerType_Vector_TcVnPoint2_REAL；重复使用非零接口指针。）
nNumberOfPoints	UDINT	预期的图案点数
fSizeDiffThresFactor	REAL	用于过滤轮廓的阈值：轮廓面积与平均轮廓面积的偏差超过（平均面积 * fSizeDiffThresFactor）时被过滤
bSubpixelAccuracy	BOOL	如果为 true，则以亚像素精度检测图案点
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向（从圆心到外圆）
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
nSearchLines	UDINT	指定搜索线的数量，在所有方向上平均分配（必须 >= 8 且是 4 的倍数）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stBlobDetectionParams	TcVnParamsBlobDetection [▶ 229]	用于检测和筛选轮廓的参数。

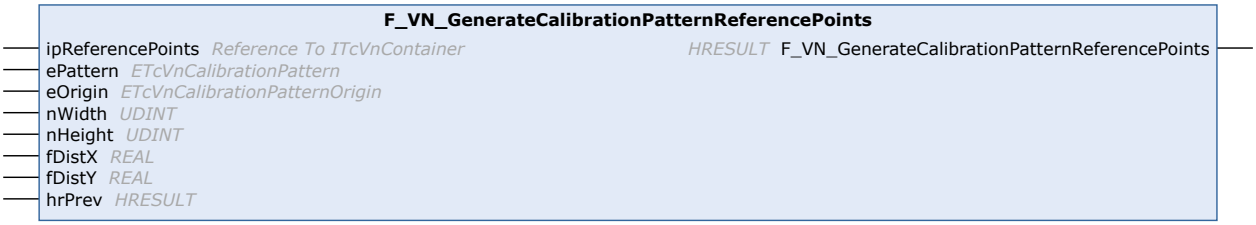
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.1.4.14.2.21 F_VN_GenerateCalibrationPatternReferencePoints



Generate reference points for common calibration patterns.

语法

定义：

```
FUNCTION F_VN_GenerateCalibrationPatternReferencePoints : HRESULT
VAR_INPUT
    ipReferencePoints : Reference To ITcVnContainer;
    ePattern          : ETcVnCalibrationPattern;
    eOrigin           : ETcVnCalibrationPatternOrigin;
    nWidth            : UDINT;
    nHeight           : UDINT;
    fDistX            : REAL;
    fDistY            : REAL;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipReferencePoints	引用 ITcVnContainer [▶ 365]	返回生成的参考点（ContainerType_Vector_TcVnPoint3_REAL）。
ePattern	ETcVnCalibrationPattern [▶ 161]	标定板类型
eOrigin	ETcVnCalibrationPatternOrigin [▶ 161]	标定板原点
nWidth	UDINT	图案宽度，即 x 方向上的图案点数
nHeight	UDINT	图案高度，即 y 方向上的图案点数
fDistX	REAL	x 方向上图案点之间的距离
fDistY	REAL	y 方向上图案点之间的距离
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

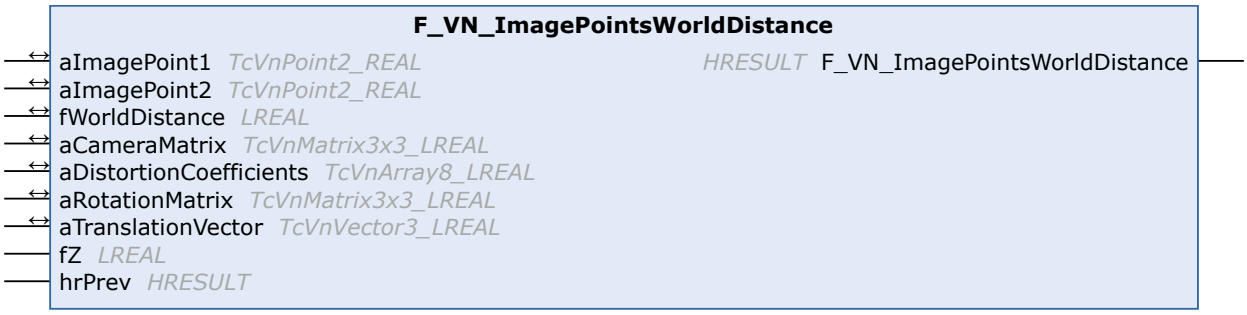
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.22 F_VN_ImagePointsWorldDistance



Computes the distance in the world coordinate system between two points (in the same world coordinate z layer) given in image coordinates.

语法

定义：

```
FUNCTION F_VN_ImagePointsWorldDistance : HRESULT
VAR_IN_OUT
    aImagePoint1            : TcVnPoint2_REAL;
    aImagePoint2            : TcVnPoint2_REAL;
    fWorldDistance          : LREAL;
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aRotationMatrix         : TcVnMatrix3x3_LREAL;
    aTranslationVector      : TcVnVector3_LREAL;
END_VAR
VAR_INPUT
    fZ                      : LREAL;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
fZ	LREAL	给定点的 z 坐标（世界坐标系）（0 表示在（顶层）标定板处）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aImagePoint1	TcVnPoint2_REAL [▶ 152]	图像坐标系中的第一个点
aImagePoint2	TcVnPoint2_REAL [▶ 152]	图像坐标系中的第二点

名称	类型	描述
fWorldDistance	LREAL	返回 aImagePoint1 和 aImagePoint2 在世界坐标系中的距离
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	包含内参的 3x3 相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	3x3 旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	平移矢量

返回值

HRESULT [▶ 135]

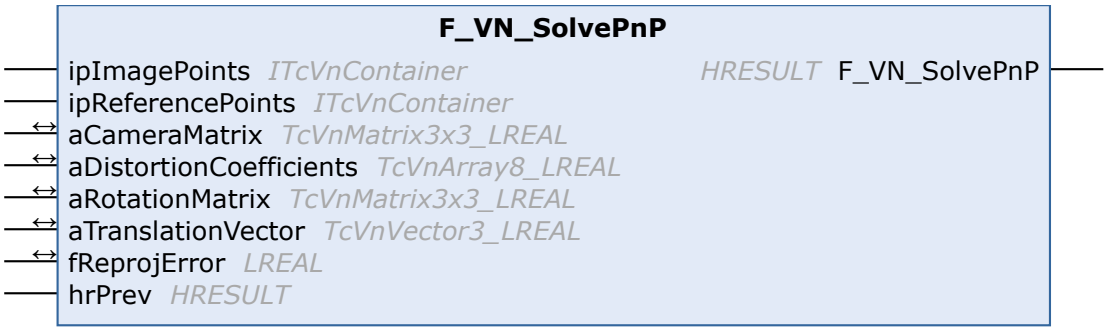
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.23 F_VN_SolvePnP



Computes the rotation matrix and translation vector from 3D-2D point correspondences.

语法

定义：

```
FUNCTION F_VN_SolvePnP : HRESULT
VAR_INPUT
    ipImagePoints           : ITcVnContainer;
    ipReferencePoints       : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aRotationMatrix         : TcVnMatrix3x3_LREAL;
    aTranslationVector      : TcVnVector3_LREAL;
    fReprojError            : LREAL;
END_VAR
VAR_INPUT
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImagePoints	ITcVnContainer [▶ 365]	图像点（ContainerType_Vector_TcVnPoint2_REAL，至少需要有 4 个点的顺序与 ipReferencePoints 中的顺序相同）
ipReferencePoints	ITcVnContainer [▶ 365]	世界参考点（ContainerType_Vector_TcVnPoint3_REAL，与 ipImagePoints 中的点数相同）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	相机矩阵（例如，使用其中一个 CalibrateCamera 函数计算得出）
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]（例如，使用其中一个 CalibrateCamera 函数计算得出）
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	返回平移向量
fReprojError	LREAL	返回重投影误差

返回值

HRESULT [▶ 135]

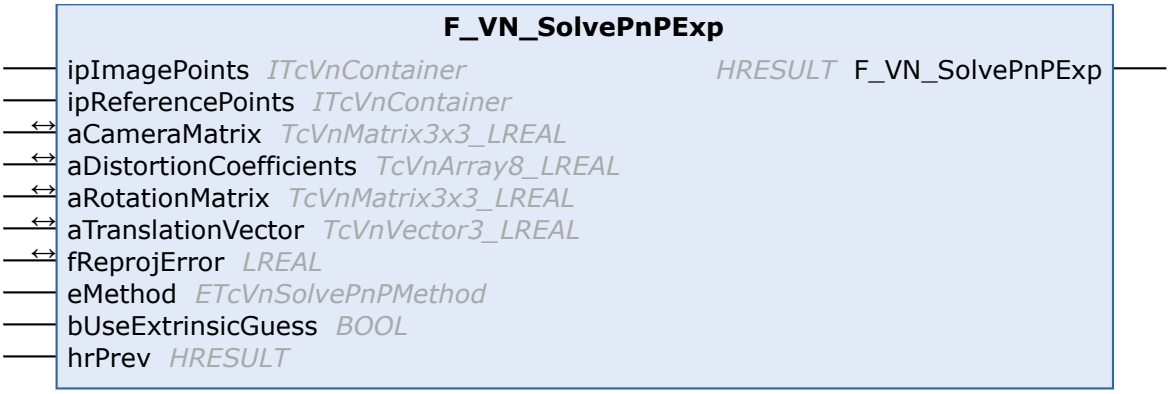
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.24 F_VN_SolvePnPExp



Computes the rotation matrix and translation vector from 3D-2D point correspondences. (expert function)

语法

定义：

```
FUNCTION F_VN_SolvePnPExp : HRESULT
VAR_INPUT
    ipImagePoints           : ITcVnContainer;
    ipReferencePoints       : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aRotationMatrix         : TcVnMatrix3x3_LREAL;
    aTranslationVector      : TcVnVector3_LREAL;
    fReprojError            : LREAL;
END_VAR
VAR_INPUT
    eMethod                 : ETcVnSolvePnPMethod;
    bUseExtrinsicGuess      : BOOL;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImagePoints	ITcVnContainer [▶ 365]	图像点（ContainerType_Vector_TcVnPoint2_REAL，至少需要有 4 个点的顺序与 ipReferencePoints 中的顺序相同）
ipReferencePoints	ITcVnContainer [▶ 365]	世界参考点（ContainerType_Vector_TcVnPoint3_REAL，与 ipImagePoints 中的点数相同）
eMethod	ETcVnSolvePnPMethod [▶ 210]	PnP 求解算法
bUseExtrinsicGuess	BOOL	如果为 true，所提供的旋转矩阵和平移向量将用作初始猜测，以实现进一步优化（仅支持 SPM_ITERATIVE 方法）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	相机矩阵（例如，使用其中一个 CalibrateCamera 函数计算得出）
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]（例如，使用其中一个 CalibrateCamera 函数计算得出）
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	返回平移向量
fReprojError	LREAL	返回重投影误差

返回值

HRESULT [▶ 135]

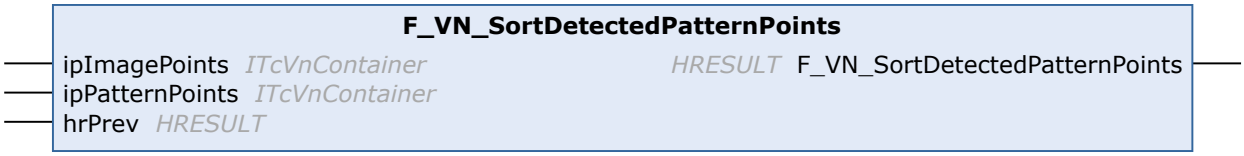
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.25 F_VN_SortDetectedPatternPoints



Sort the detected pattern points according to the relative positions of the calibration pattern points.

语法

定义：

```
FUNCTION F_VN_SortDetectedPatternPoints : HRESULT
VAR_INPUT
    ipImagePoints   : ITcVnContainer;
    ipPatternPoints : ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImagePoints	ITcVnContainer [▶ 365]	图像坐标系中检测到的图案点位置（将通过函数 ContainerType_Vector_TcVnPoint2_REAL 进行排序）
ipPatternPoints	ITcVnContainer [▶ 365]	标定板坐标系中的图案点，用作对 ipImagePoints（ContainerType_Vector_TcVnPoint3_REAL）进行排序时的参考值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

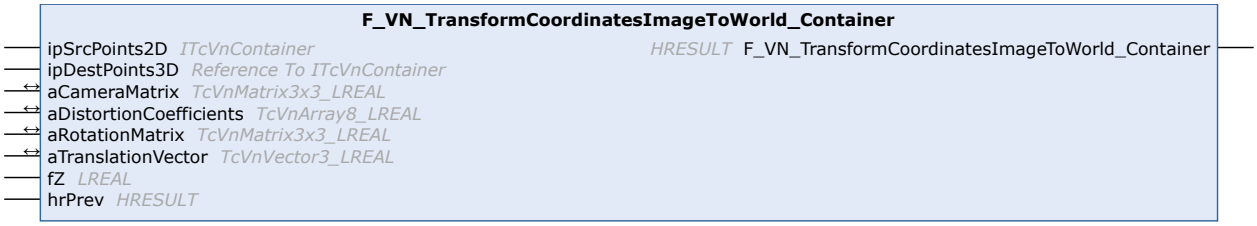
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.26 F_VN_TransformCoordinatesImageToWorld_Container



Transform 2D image point coordinates to 3D world coordinates (using intrinsic + extrinsic calibration results).

语法

定义：

```
FUNCTION F_VN_TransformCoordinatesImageToWorld_Container : HRESULT
VAR_INPUT
    ipSrcPoints2D           : ITcVnContainer;
    ipDestPoints3D          : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aRotationMatrix         : TcVnMatrix3x3_LREAL;
    aTranslationVector      : TcVnVector3_LREAL;
END_VAR
VAR_INPUT
    fZ                      : LREAL;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPoints2D	ITcVnContainer [▶ 365]	包含二维源点的容器（TcVnPoint2_DINT 或 TcVnPoint2_REAL 或 TcVnPoint2_LREAL）
ipDestPoints3D	引用 ITcVnContainer [▶ 365]	返回变换后的三维点（TcVnPoint3_LREAL）
fZ	LREAL	变换点的 z 坐标（世界坐标系）（0 表示在（顶层）标定板处）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	包含内参的 3x3 相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	3x3 旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	平移矢量

返回值

HRESULT [▶ 135]

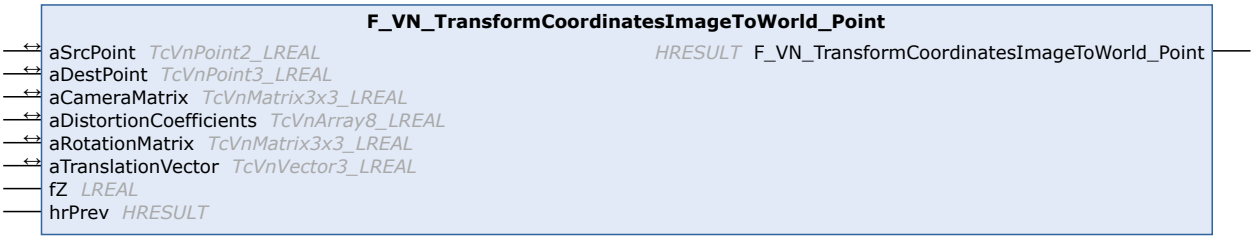
所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.27 F_VN_TransformCoordinatesImageToWorld_Point



Transform 2D image point coordinate to 3D world coordinate (using intrinsic + extrinsic calibration results).

语法

定义：

```
FUNCTION F_VN_TransformCoordinatesImageToWorld_Point : HRESULT
VAR_IN_OUT
    aSrcPoint               : TcVnPoint2_LREAL;
    aDestPoint              : TcVnPoint3_LREAL;
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aRotationMatrix         : TcVnMatrix3x3_LREAL;
    aTranslationVector      : TcVnVector3_LREAL;
END_VAR
VAR_INPUT
    fZ                      : LREAL;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
fZ	LREAL	变换点的 z 坐标（世界坐标系）（0 表示在（顶层）标定板处）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aSrcPoint	TcVnPoint2_LREAL [▶ 152]	图像坐标系中的点
aDestPoint	TcVnPoint3_LREAL [▶ 152]	返回世界坐标系中的点
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	包含内参的 3x3 相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	3x3 旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	平移矢量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.28 F_VN_TransformCoordinatesPlanar_Container



Compute real-world coordinates for given image points or vice versa.

语法

定义：

```
FUNCTION F_VN_TransformCoordinatesPlanar_Container : HRESULT
VAR_INPUT
    ipSrcPoints           : ITcVnContainer;
    ipDestPoints          : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix3x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPoints	ITcVnContainer [▶ 365]	包含二维源点的容器（TcVnPoint2_DINT 或 TcVnPoint2_REAL 或 TcVnPoint2_LREAL）
ipDestPoints	引用 ITcVnContainer [▶ 365]	返回变换后的二维点（TcVnPoint2_LREAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	通过 VnCalibrateCameraPlanar() 获得的变换矩阵或通过 VnInvertMatrix3x3() 获得的逆矩阵

返回值

HRESULT [▶ 135]

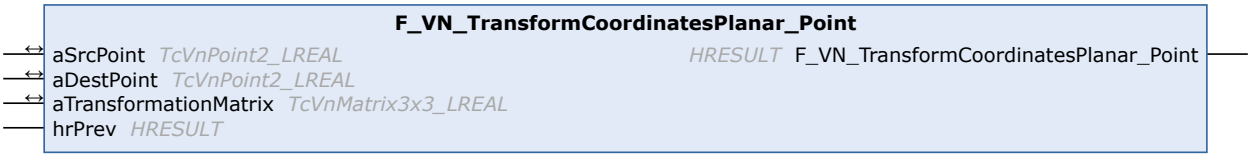
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.2.29 F_VN_TransformCoordinatesPlanar_Point



为给定的图像点计算世界坐标，反之亦然。

语法

定义：

```
FUNCTION F_VN_TransformCoordinatesPlanar_Point : HRESULT
VAR_IN_OUT
    aSrcPoint             : TcVnPoint2_LREAL;
    aDestPoint            : TcVnPoint2_LREAL;
    aTransformationMatrix : TcVnMatrix3x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aSrcPoint	TcVnPoint2_LREAL [▶ 152]	源坐标系中的点
aDestPoint	TcVnPoint2_LREAL [▶ 152]	返回目标坐标系中的点
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	通过 VnCalibrateCameraPlanar() 获得的变换矩阵或通过 VnInvertMatrix3x3() 获得的逆矩阵

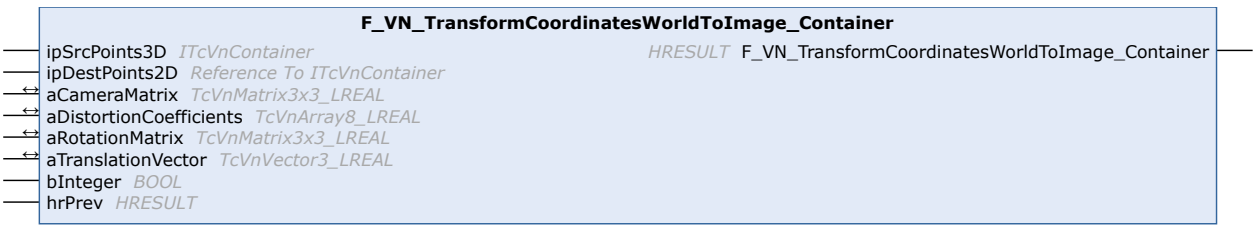
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.2.30 F_VN_TransformCoordinatesWorldToImage_Container



将三维世界坐标转换为二维图像坐标（使用内参和外参标定结果）。

语法

定义：

```
FUNCTION F_VN_TransformCoordinatesWorldToImage_Container : HRESULT
VAR_INPUT
    ipSrcPoints3D           : ITcVnContainer;
    ipDestPoints2D          : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aRotationMatrix         : TcVnMatrix3x3_LREAL;
    aTranslationVector      : TcVnVector3_LREAL;
END_VAR
VAR_INPUT
    bInteger                : BOOL;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPoints3D	ITcVnContainer [▶ 365]	包含三维源点的容器（TcVnPoint3_LREAL）
ipDestPoints2D	引用 ITcVnContainer [▶ 365]	返回变换后的二维点（TcVnPoint2_REAL 或 TcVnPoint2_DINT，取决于 bInteger）
bInteger	BOOL	选择函数返回的是 TcVnPoint2_DINT 还是 TcVnPoint2_REAL 容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	包含内参的 3x3 相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	3x3 旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	平移矢量

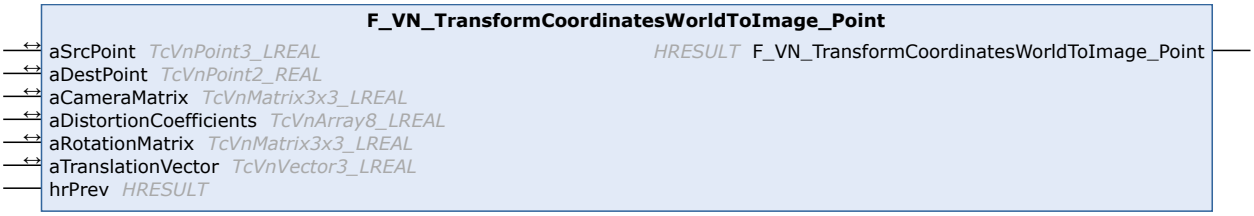
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.1.4.14.2.31 F_VN_TransformCoordinatesWorldToImage_Point



将三维世界坐标转换为二维图像坐标（使用内参和外参标定结果）。

语法

定义：

```
FUNCTION F_VN_TransformCoordinatesWorldToImage_Point : HRESULT
VAR_IN_OUT
    aSrcPoint               : TcVnPoint3_LREAL;
    aDestPoint              : TcVnPoint2_REAL;
    aCameraMatrix           : TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : TcVnArray8_LREAL;
    aRotationMatrix         : TcVnMatrix3x3_LREAL;
    aTranslationVector      : TcVnVector3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aSrcPoint	TcVnPoint3_LREAL [▶ 152]	三维源点（世界坐标系）
aDestPoint	TcVnPoint2_REAL [▶ 152]	返回变换后的二维点（图像坐标系）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	包含内参的 3x3 相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	3x3 旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	平移矢量

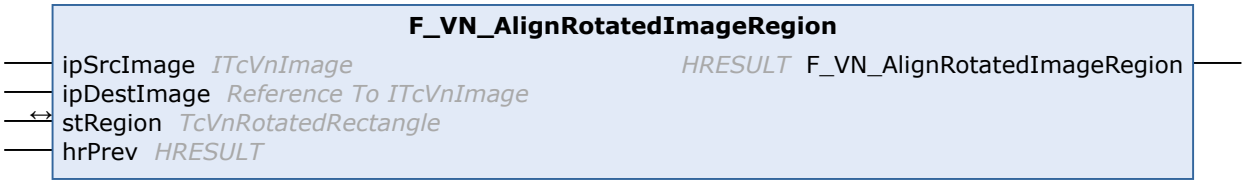
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.1.4.14.3 F_VN_AlignRotatedImageRegion



复制提供的图像区域并将其与图像轴对齐。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_AlignRotatedImageRegion : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    stRegion    : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stRegion	TcVnRotatedRectangle [▶ 237]	要对齐的图像区域

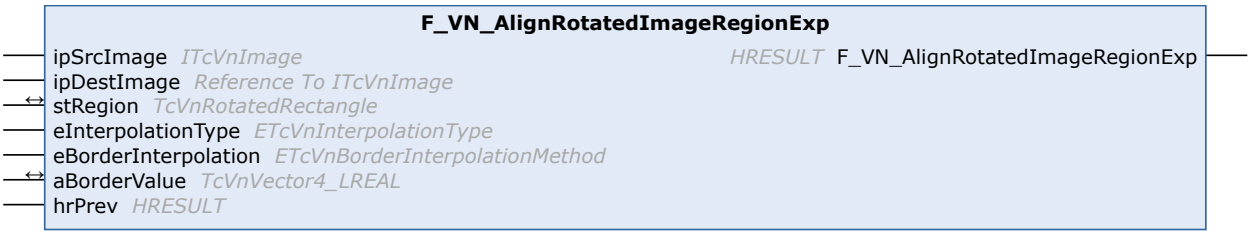
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.4 F_VN_AlignRotatedImageRegionExp



复制提供的图像区域并将其与图像轴对齐。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_AlignRotatedImageRegionExp : HRESULT
VAR_INPUT
    ipSrcImage           : ITcVnImage;
    ipDestImage          : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    stRegion             : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    eInterpolationType   : ETcVnInterpolationType;
    eBorderInterpolation : ETcVnBorderInterpolationMethod;
END_VAR
VAR_IN_OUT
    aBorderValue         : TcVnVector4_LREAL;
END_VAR
```

```
VAR_INPUT
    hrPrev              : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
eInterpolationType	ETcVnInterpolationType [▶ 198]	插值法（不支持 BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）
eBorderInterpolation	ETcVnBorderInterpolationMethod [▶ 160]	边界插值法（不支持 ISOLATED）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stRegion	TcVnRotatedRectangle [▶ 237]	要对齐的图像区域
aBorderValue	TcVnVector4_LREAL [▶ 153]	边界值（如果使用 CONSTANT）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.5 F_VN_ApplyRotationToAffineTransformation



对现有的二维仿射变换矩阵进行旋转。

语法

定义：

```
FUNCTION F_VN_ApplyRotationToAffineTransformation : HRESULT
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    fAngle              : LREAL;
    hrPrev              : HRESULT;
END_VAR
```

输入

名称	类型	描述
fAngle	LREAL	以弧度为单位的角度（正值表示逆时针方向）

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	对其应用旋转的仿射变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.6 F_VN_ApplyRotationToAffineTransformationExp



对现有的二维仿射变换矩阵进行旋转。（专家函数）

语法

定义：

```
FUNCTION F_VN_ApplyRotationToAffineTransformationExp : HRESULT
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    fAngle                : LREAL;
    bUsePreMultiplication : BOOL;
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
fAngle	LREAL	以弧度为单位的角度（正值表示逆时针方向）
bUsePreMultiplication	BOOL	选择使用预乘还是后乘
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	对其应用旋转的仿射变换矩阵

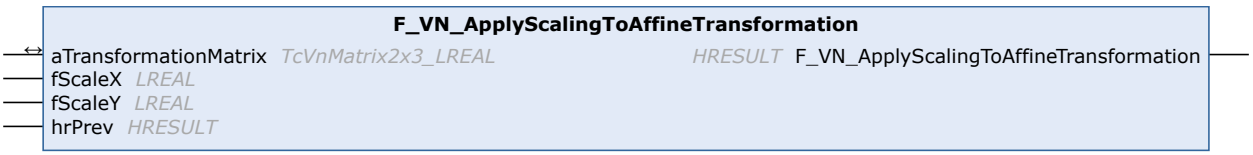
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.7 F_VN_ApplyScalingToAffineTransformation



对现有的二维仿射变换矩阵进行缩放。

语法

定义：

```
FUNCTION F_VN_ApplyScalingToAffineTransformation : HRESULT
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    fScaleX               : LREAL;
    fScaleY               : LREAL;
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
fScaleX	LREAL	x 方向的缩放系数
fScaleY	LREAL	y 方向的缩放系数
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	对其进行缩放的仿射变换矩阵

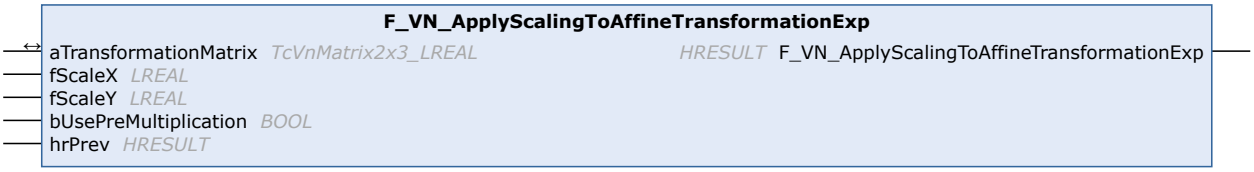
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.8 F_VN_ApplyScalingToAffineTransformationExp



对现有的二维仿射变换矩阵进行缩放。（专家函数）

语法

定义：

```
FUNCTION F_VN_ApplyScalingToAffineTransformationExp : HRESULT
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    fScaleX               : LREAL;
    fScaleY               : LREAL;
    bUsePreMultiplication : BOOL;
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
fScaleX	LREAL	x 方向的缩放系数
fScaleY	LREAL	y 方向的缩放系数
bUsePreMultiplication	BOOL	选择使用预乘还是后乘
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	对其进行缩放的仿射变换矩阵

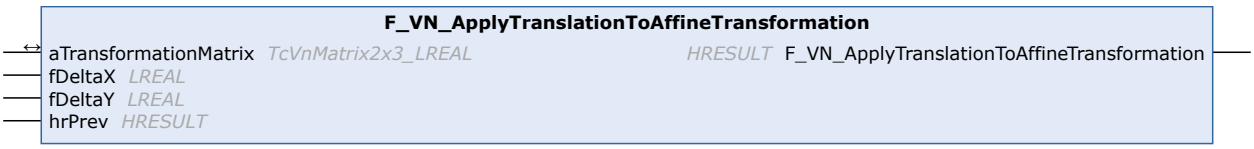
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.9 F_VN_ApplyTranslationToAffineTransformation



对现有的二维仿射变换矩阵进行平移。

语法

定义：

```
FUNCTION F_VN_ApplyTranslationToAffineTransformation : HRESULT
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    fDeltaX               : LREAL;
    fDeltaY               : LREAL;
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
fDeltaX	LREAL	x 方向的平移
fDeltaY	LREAL	y 方向的平移
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	对其应用平移的仿射变换矩阵

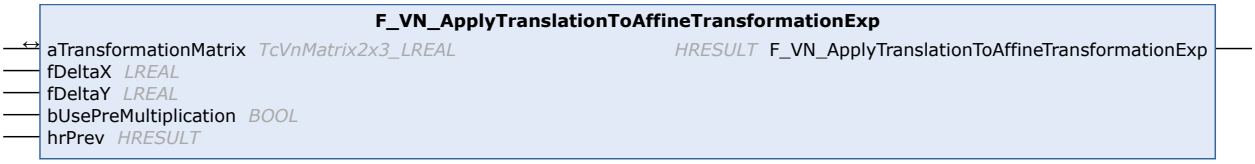
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.10 F_VN_ApplyTranslationToAffineTransformationExp



对现有的二维仿射变换矩阵进行平移。（专家函数）

语法

定义：

```
FUNCTION F_VN_ApplyTranslationToAffineTransformationExp : HRESULT
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    fDeltaX               : LREAL;
    fDeltaY               : LREAL;
    bUsePreMultiplication : BOOL;
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
fDeltaX	LREAL	x 方向的平移
fDeltaY	LREAL	y 方向的平移

名称	类型	描述
bUsePreMultiplication	BOOL	选择使用预乘还是后乘
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	对其应用平移的仿射变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.11 F_VN_ApplyYAxisInversionToAffineTransformation



Apply an inversion of the y-axis direction to an existing 2D affine transformation matrix.

语法

定义：

```
FUNCTION F_VN_ApplyYAxisInversionToAffineTransformation : HRESULT
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	对其进行 y 轴方向反转的仿射变换矩阵

返回值

HRESULT [▶ 135]

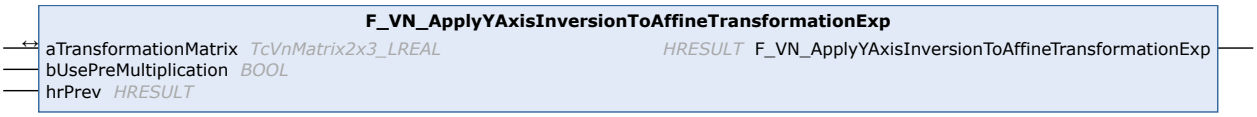
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.12 F_VN_ApplyYAxisInversionToAffineTransformationExp



Apply an inversion of the y-axis direction to an existing 2D affine transformation matrix. (expert function)

语法

定义：

```
FUNCTION F_VN_ApplyYAxisInversionToAffineTransformationExp : HRESULT
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    bUsePreMultiplication : BOOL;
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
bUsePreMultiplication	BOOL	选择使用预乘还是后乘
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	对其进行 y 轴方向反转的仿射变换矩阵

返回值

HRESULT [▶ 135]

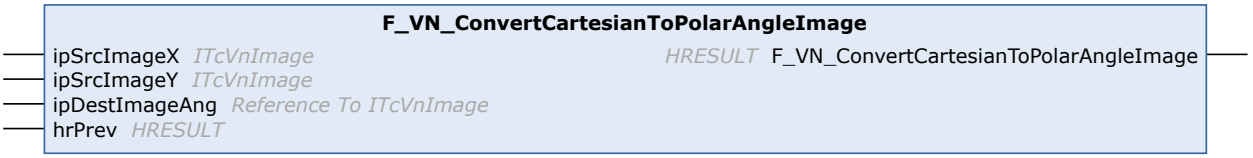
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.13 F_VN_ConvertCartesianToPolarAngleImage



Converts cartesian coordinates (x, y) to polar angle.

语法

定义：

```
FUNCTION F_VN_ConvertCartesianToPolarAngleImage : HRESULT
VAR_INPUT
    ipSrcImageX    : ITcVnImage;
    ipSrcImageY    : ITcVnImage;
    ipDestImageAng : Reference To ITcVnImage;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImageX	ITcVnImage [▶ 407]	包含 x 值的源图像（REAL 或 LREAL 类型元素）
ipSrcImageY	ITcVnImage [▶ 407]	包含 y 值的源图像（与 ipSrcImageX 的元素类型相同）
ipDestImageAng	引用 ITcVnImage [▶ 407]	包含角度的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.14 F_VN_ConvertCartesianToPolarAngleImageExp



Converts cartesian coordinates (x, y) to polar angle. (expert function)

语法

定义：

```
FUNCTION F_VN_ConvertCartesianToPolarAngleImageExp : HRESULT
VAR_INPUT
    ipSrcImageX     : ITcVnImage;
    ipSrcImageY     : ITcVnImage;
    ipDestImageAng  : Reference To ITcVnImage;
    bAngleInDegrees : BOOL;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImageX	ITcVnImage [▶ 407]	包含 x 值的源图像（REAL 或 LREAL 类型元素）
ipSrcImageY	ITcVnImage [▶ 407]	包含 y 值的源图像（与 ipSrcImageX 的元素类型相同）
ipDestImageAng	引用 ITcVnImage [▶ 407]	包含角度的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
bAngleInDegrees	BOOL	指定角度的单位是度还是弧度
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

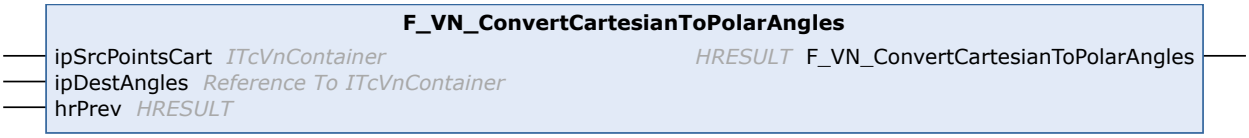
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.15 F_VN_ConvertCartesianToPolarAngles



Converts cartesian coordinates (x, y) to polar angle.

语法

定义：

```
FUNCTION F_VN_ConvertCartesianToPolarAngles : HRESULT
VAR_INPUT
    ipSrcPointsCart : ITcVnContainer;
    ipDestAngles    : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPointsCart	ITcVnContainer [▶ 365]	包含笛卡尔点的容器（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL）
ipDestAngles	引用 ITcVnContainer [▶ 365]	返回一个以角度填充的容器（如果源点的类型为 TcVnPoint2_DINT 或 TcVnPoint2_REAL，则返回 ContainerType_Vector_REAL；否则返回 ContainerType_Vector_LREAL。重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

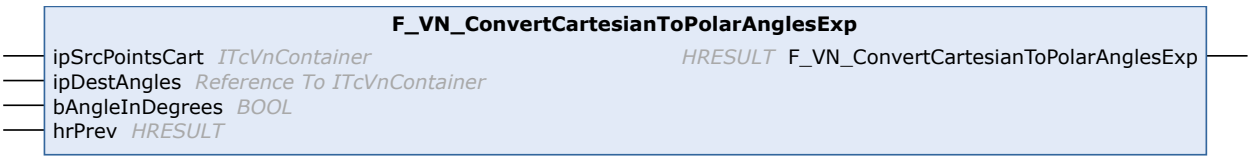
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.16 F_VN_ConvertCartesianToPolarAnglesExp



Converts cartesian coordinates (x, y) to polar angle. (expert function)

语法

定义：

```
FUNCTION F_VN_ConvertCartesianToPolarAnglesExp : HRESULT
VAR_INPUT
    ipSrcPointsCart : ITcVnContainer;
    ipDestAngles    : Reference To ITcVnContainer;
    bAngleInDegrees : BOOL;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPointsCart	ITcVnContainer [▶ 365]	包含笛卡尔点的容器（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL）

名称	类型	描述
ipDestAngles	引用 ITcVnContainer [▶ 365]	返回一个以角度填充的容器（如果源点的类型为 TcVnPoint2_DINT 或 TcVnPoint2_REAL，则返回 ContainerType_Vector_REAL；否则返回 ContainerType_Vector_LREAL。重复使用非零接口指针。）
bAngleInDegrees	BOOL	指定角度的单位是度还是弧度
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

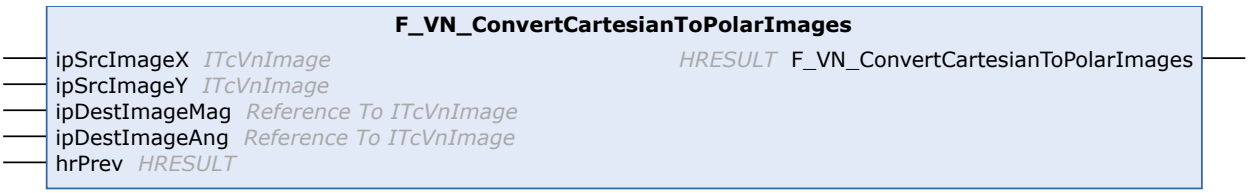
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.17 F_VN_ConvertCartesianToPolarImages



Converts cartesian coordinates (x, y) to polar coordinates (magnitude, angle).

语法

定义：

```
FUNCTION F_VN_ConvertCartesianToPolarImages : HRESULT
VAR_INPUT
    ipSrcImageX    : ITcVnImage;
    ipSrcImageY    : ITcVnImage;
    ipDestImageMag : Reference To ITcVnImage;
    ipDestImageAng : Reference To ITcVnImage;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImageX	ITcVnImage [▶ 407]	包含 x 值的源图像（REAL 或 LREAL 类型元素）
ipSrcImageY	ITcVnImage [▶ 407]	包含 y 值的源图像（与 ipSrcImageX 的元素类型相同）
ipDestImageMag	引用 ITcVnImage [▶ 407]	包含幅值的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
ipDestImageAng	引用 ITcVnImage [▶ 407]	包含角度的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.18 F_VN_ConvertCartesianToPolarImagesExp



Converts cartesian coordinates (x, y) to polar coordinates (magnitude, angle). (expert function)

语法

定义：

```
FUNCTION F_VN_ConvertCartesianToPolarImagesExp : HRESULT
VAR_INPUT
    ipSrcImageX     : ITcVnImage;
    ipSrcImageY     : ITcVnImage;
    ipDestImageMag  : Reference To ITcVnImage;
    ipDestImageAng  : Reference To ITcVnImage;
    bAngleInDegrees : BOOL;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImageX	ITcVnImage [▶ 407]	包含 x 值的源图像（REAL 或 LREAL 类型元素）
ipSrcImageY	ITcVnImage [▶ 407]	包含 y 值的源图像（与 ipSrcImageX 的元素类型相同）
ipDestImageMag	引用 ITcVnImage [▶ 407]	包含幅值的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
ipDestImageAng	引用 ITcVnImage [▶ 407]	包含角度的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
bAngleInDegrees	BOOL	指定角度的单位是度还是弧度
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

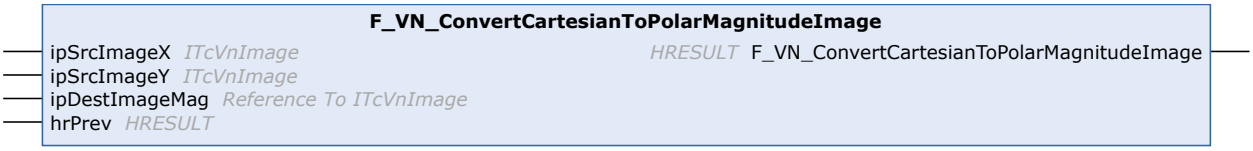
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.19 F_VN_ConvertCartesianToPolarMagnitudeImage



Converts cartesian coordinates (x, y) to polar magnitude.

语法

定义：

```
FUNCTION F_VN_ConvertCartesianToPolarMagnitudeImage : HRESULT
VAR_INPUT
    ipSrcImageX    : ITcVnImage;
    ipSrcImageY    : ITcVnImage;
    ipDestImageMag : Reference To ITcVnImage;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImageX	ITcVnImage [▶ 407]	包含 x 值的源图像（REAL 或 LREAL 类型元素）
ipSrcImageY	ITcVnImage [▶ 407]	包含 y 值的源图像（与 ipSrcImageX 的元素类型相同）
ipDestImageMag	引用 ITcVnImage [▶ 407]	包含幅值的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

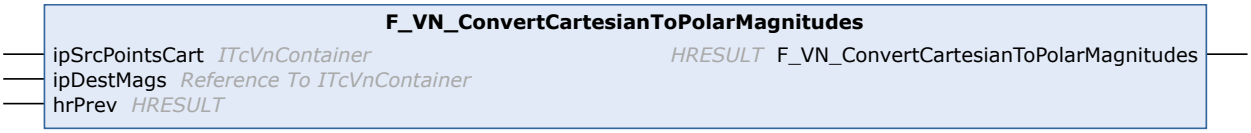
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.20 F_VN_ConvertCartesianToPolarMagnitudes



Converts cartesian coordinates (x, y) to polar magnitude.

语法

定义：

```
FUNCTION F_VN_ConvertCartesianToPolarMagnitudes : HRESULT
VAR_INPUT
    ipSrcPointsCart : ITcVnContainer;
    ipDestMags      : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPointsCart	ITcVnContainer [▶ 365]	包含笛卡尔点的容器（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL）
ipDestMags	引用 ITcVnContainer [▶ 365]	返回一个以幅值填充的容器（如果源点的类型是 TcVnPoint2_DINT 或 TcVnPoint2_REAL，则返回 ContainerType_Vector_REAL；否则返回 ContainerType_Vector_LREAL。重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

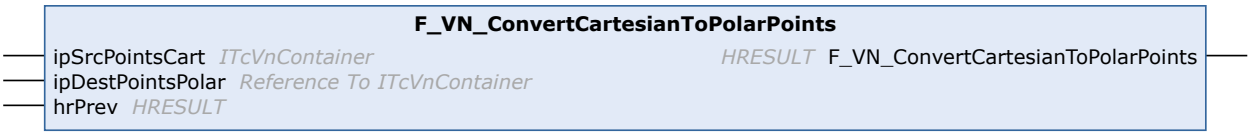
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.21 F_VN_ConvertCartesianToPolarPoints



Converts cartesian coordinates (x, y) to polar coordinates (magnitude, angle).

语法

定义：

```
FUNCTION F_VN_ConvertCartesianToPolarPoints : HRESULT
VAR_INPUT
    ipSrcPointsCart   : ITcVnContainer;
    ipDestPointsPolar : Reference To ITcVnContainer;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPointsCart	ITcVnContainer [▶ 365]	包含笛卡尔点的容器（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL）
ipDestPointsPolar	引用 ITcVnContainer [▶ 365]	返回一个以极坐标转换后的点填充的容器（如果源点的类型是 TcVnPoint2_DINT 或 TcVnPoint2_REAL，则返回 ContainerType_Vector_TcVnPoint2_REAL；否则返回 ContainerType_Vector_TcVnPoint2_LREAL。重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

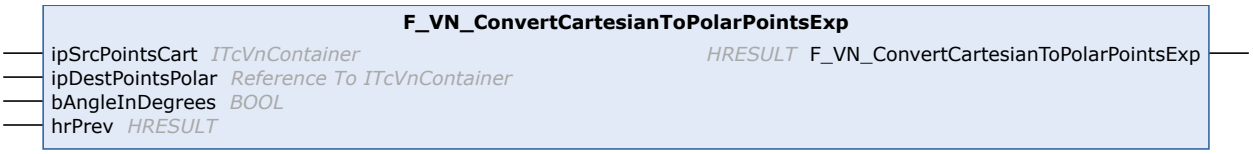
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.22 F_VN_ConvertCartesianToPolarPointsExp



将笛卡尔坐标（x、y）转换为极坐标（幅值、角度）。（专家函数）

语法

定义：

```
FUNCTION F_VN_ConvertCartesianToPolarPointsExp : HRESULT
VAR_INPUT
    ipSrcPointsCart   : ITcVnContainer;
    ipDestPointsPolar : Reference To ITcVnContainer;
    bAngleInDegrees   : BOOL;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPointsCart	ITcVnContainer [▶ 365]	包含笛卡尔点的容器（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL）

名称	类型	描述
ipDestPointsPolar	引用 ITcVnContainer [▶ 365]	返回一个以极坐标转换后的点填充的容器（如果源点的类型是 TcVnPoint2_DINT 或 TcVnPoint2_REAL，则返回 ContainerType_Vector_TcVnPoint2_REAL；否则返回 ContainerType_Vector_TcVnPoint2_LREAL。重复使用非零接口指针。）
bAngleInDegrees	BOOL	指定角度的单位是度还是弧度
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

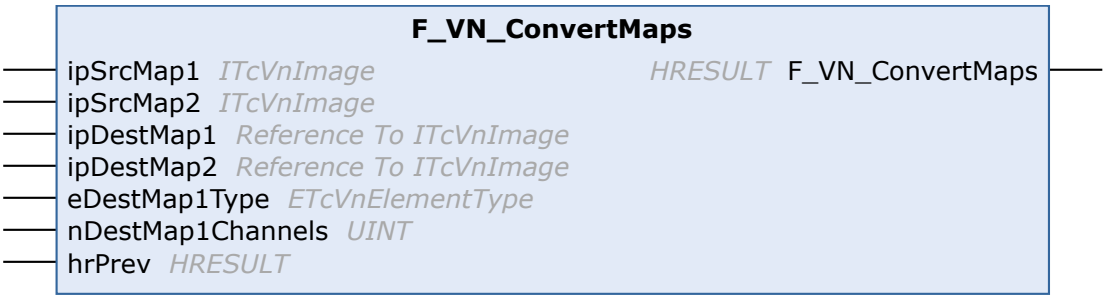
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.23 F_VN_ConvertMaps



在不同映射类型之间进行转换，通常从浮点转换为定点，以加速 RemapImage。支持的映射对包括（map1 = ET_REAL，2 通道；map2 = 0）、（map1 = ET_REAL，1 通道；map2 = ET_REAL，1 通道）、（map1 = ET_INT，2 通道；map2 = ET_UINT，1 通道）、（map1 = ET_INT，2 通道；map2 = 0）。

语法

定义：

```
FUNCTION F_VN_ConvertMaps : HRESULT
VAR_INPUT
    ipSrcMap1          : ITcVnImage;
    ipSrcMap2          : ITcVnImage;
    ipDestMap1         : Reference To ITcVnImage;
    ipDestMap2         : Reference To ITcVnImage;
    eDestMap1Type      : ETcVnElementType;
    nDestMap1Channels  : UINT;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcMap1	ITcVnImage [▶ 407]	源映射 1（ET_REAL 带 2 通道（包含 mapX、mapY），或 ET_REAL 带 1 通道（包含 mapX），或 ET_INT 带 2 通道（包含取整后的 mapX、mapY））
ipSrcMap2	ITcVnImage [▶ 407]	源映射 2（ET_REAL 带 1 通道（包含 mapY），或 ET_UINT 带 1 通道（如果 ipSrcMap1 为 ET_INT 带 2 通道，则包含插值表系数），如果不需要，则设为 0。
ipDestMap1	引用 ITcVnImage [▶ 407]	目标映射 1（不得与 ipSrcMap1 或 ipSrcMap2 相同！如有需要，将创建适当的目标图像。）

名称	类型	描述
ipDestMap2	引用 ITcVnImage [▶ 407]	目标映射 2（不得与 ipSrcMap1 或 ipSrcMap2 相同！如果不需要，即如果 ipDestMap1 为 ET_REAL 带 2 通道，则设为 0。否则，如有需要，将创建适当的目标图像。）
eDestMap1Type	ETcVnElementType [▶ 191]	生成的 ipDestMap1 的请求类型（ET_REAL 或 ET_INT）。ipDestMap2 的类型将自动确定，即 ET_REAL 或 ET_UINT 或无。
nDestMap1Channels	UINT	生成的 ipDestMap1 的请求通道数（1 或 2，如果 eDestMap1Type 为 ET_INT，则必须为 2）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

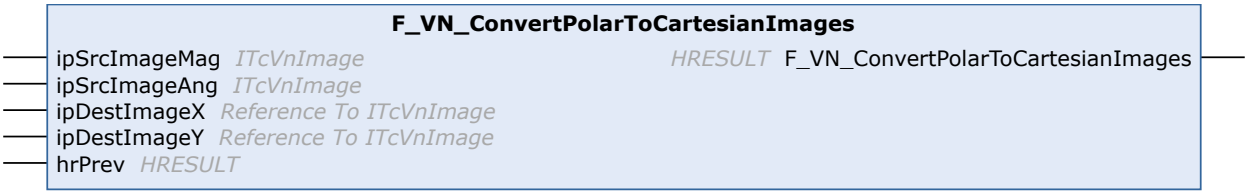
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.24 F_VN_ConvertPolarToCartesianImages



将极坐标（幅值、角度）转换为笛卡尔坐标（x、y）。

语法

定义：

```
FUNCTION F_VN_ConvertPolarToCartesianImages : HRESULT
VAR_INPUT
    ipSrcImageMag : ITcVnImage;
    ipSrcImageAng : ITcVnImage;
    ipDestImageX  : Reference To ITcVnImage;
    ipDestImageY  : Reference To ITcVnImage;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImageMag	ITcVnImage [▶ 407]	包含幅值的源图像（REAL 或 LREAL 类型元素）
ipSrcImageAng	ITcVnImage [▶ 407]	包含角度的源图像（与 ipSrcImageMag 的元素类型相同）
ipDestImageX	引用 ITcVnImage [▶ 407]	包含 x 值的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
ipDestImageY	引用 ITcVnImage [▶ 407]	包含 y 值的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.25 F_VN_ConvertPolarToCartesianImagesExp



将极坐标（幅值、角度）转换为笛卡尔坐标（x、y）。（专家函数）

语法

定义：

```
FUNCTION F_VN_ConvertPolarToCartesianImagesExp : HRESULT
VAR_INPUT
    ipSrcImageMag   : ITcVnImage;
    ipSrcImageAng   : ITcVnImage;
    ipDestImageX    : Reference To ITcVnImage;
    ipDestImageY    : Reference To ITcVnImage;
    bAngleInDegrees : BOOL;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImageMag	ITcVnImage [▶ 407]	包含幅值的源图像（REAL 或 LREAL 类型元素）
ipSrcImageAng	ITcVnImage [▶ 407]	包含角度的源图像（与 ipSrcImageMag 的元素类型相同）
ipDestImageX	引用 ITcVnImage [▶ 407]	包含 x 值的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
ipDestImageY	引用 ITcVnImage [▶ 407]	包含 y 值的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
bAngleInDegrees	BOOL	指定角度的单位是度还是弧度
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

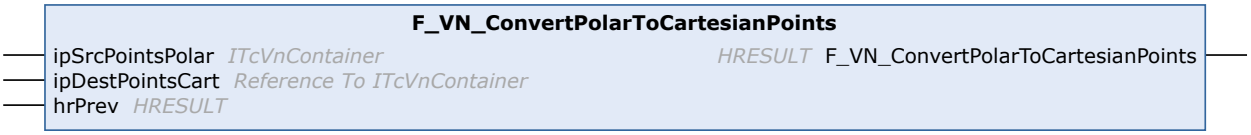
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.26 F_VN_ConvertPolarToCartesianPoints



将极坐标（幅值、角度）转换为笛卡尔坐标（x、y）。

语法

定义：

```
FUNCTION F_VN_ConvertPolarToCartesianPoints : HRESULT
VAR_INPUT
    ipSrcPointsPolar : ITcVnContainer;
    ipDestPointsCart : Reference To ITcVnContainer;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPointsPolar	ITcVnContainer [▶ 365]	包含极坐标的容器（ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL）
ipDestPointsCart	引用 ITcVnContainer [▶ 365]	返回以笛卡尔坐标转换后的点填充的容器（与 ipSrcPointsPolar 的类型 ID 相同。重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.27 F_VN_ConvertPolarToCartesianPointsExp



将极坐标（幅值、角度）转换为笛卡尔坐标（x、y）。（专家函数）

语法

定义：

```
FUNCTION F_VN_ConvertPolarToCartesianPointsExp : HRESULT
VAR_INPUT
    ipSrcPointsPolar : ITcVnContainer;
    ipDestPointsCart : Reference To ITcVnContainer;
    bAngleInDegrees  : BOOL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPointsPolar	ITcVnContainer [▶ 365]	包含极坐标的容器（ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL）
ipDestPointsCart	引用 ITcVnContainer [▶ 365]	返回以笛卡尔坐标转换后的点填充的容器（与 ipSrcPointsPolar 的类型 ID 相同。重复使用非零接口指针。）
bAngleInDegrees	BOOL	指定角度的单位是度还是弧度
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.28 F_VN_DecomposeAffineTransformation



分解仿射变换矩阵，并返回以弧度为单位的旋转角（顺时针）。分解法假定变换顺序如下：平移、旋转、缩放和倾斜。

语法

定义：

```
FUNCTION F_VN_DecomposeAffineTransformation : HRESULT
VAR_IN_OUT
    aAffineTransform : TcVnMatrix2x3_LREAL;
    fRotationAngle   : LREAL;
END_VAR
VAR_INPUT
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aAffineTransform	TcVnMatrix2x3_LREAL [▶ 153]	仿射变换矩阵
fRotationAngle	LREAL	顺时针方向的旋转角（单位：弧度）。旋转中心为原点（0，0）

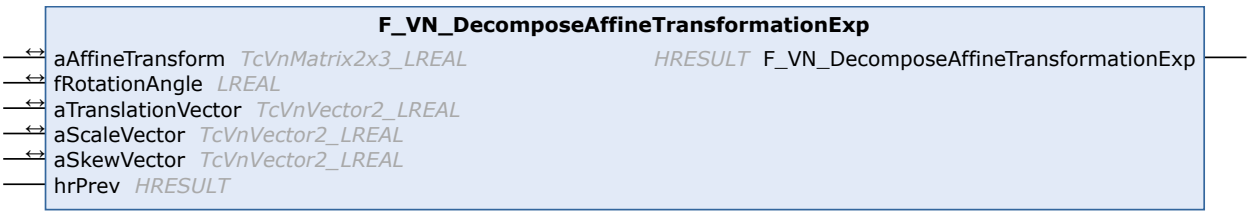
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.29 F_VN_DecomposeAffineTransformationExp



分解仿射变换矩阵，并返回以弧度为单位的旋转角（顺时针）、平移向量（像素）、缩放比例和倾斜向量（以弧度为单位的角）。分解法假定变换顺序如下：平移、旋转、缩放和倾斜。

语法

定义：

```
FUNCTION F_VN_DecomposeAffineTransformationExp : HRESULT
VAR_IN_OUT
    aAffineTransform    : TcVnMatrix2x3_LREAL;
    fRotationAngle      : LREAL;
    aTranslationVector  : TcVnVector2_LREAL;
    aScaleVector        : TcVnVector2_LREAL;
    aSkewVector         : TcVnVector2_LREAL;
END_VAR
VAR_INPUT
    hrPrev              : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aAffineTransform	TcVnMatrix2x3_LREAL [▶ 153]	仿射变换矩阵
fRotationAngle	LREAL	顺时针方向的旋转角（单位：弧度）。旋转中心为原点（0，0）
aTranslationVector	TcVnVector2_LREAL [▶ 153]	二维平移向量。数值单位为像素
aScaleVector	TcVnVector2_LREAL [▶ 153]	二维比例向量
aSkewVector	TcVnVector2_LREAL [▶ 153]	二维倾斜向量。数值是 x 和 y 方向的倾斜角，单位为弧度

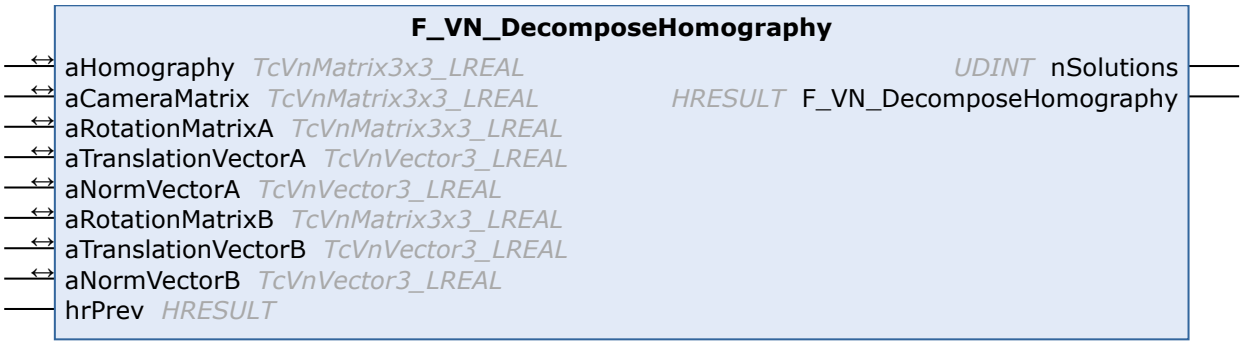
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.30 F_VN_DecomposeHomography



将单应性矩阵分解为旋转、平移和平面法线四个解。返回两个解（Ra、Ta、Nb）和（Rb、Tb、Nb）。第三个和第四个解的计算方法如下：（Ra、-Ta、-Na）和（Rb、-Tb、-Nb）。

语法

定义：

```
FUNCTION F_VN_DecomposeHomography : HRESULT
VAR_IN_OUT
    aHomography         : TcVnMatrix3x3_LREAL;
    aCameraMatrix       : TcVnMatrix3x3_LREAL;
    aRotationMatrixA    : TcVnMatrix3x3_LREAL;
    aTranslationVectorA : TcVnVector3_LREAL;
    aNormVectorA        : TcVnVector3_LREAL;
    aRotationMatrixB    : TcVnMatrix3x3_LREAL;
    aTranslationVectorB : TcVnVector3_LREAL;
    aNormVectorB        : TcVnVector3_LREAL;
END_VAR
VAR_INPUT
    hrPrev              : HRESULT;
END_VAR
VAR_OUTPUT
    nSolutions          : UDINT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aHomography	TcVnMatrix3x3_LREAL [▶ 153]	单应性矩阵（两个平面之间的透视变换）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	相机矩阵
aRotationMatrixA	TcVnMatrix3x3_LREAL [▶ 153]	第一个解的旋转矩阵（Ra）
aTranslationVectorA	TcVnVector3_LREAL [▶ 153]	第一个解的平移向量（Ta）
aNormVectorA	TcVnVector3_LREAL [▶ 153]	第一个解的范数向量（Na）
aRotationMatrixB	TcVnMatrix3x3_LREAL [▶ 153]	第二个解的旋转矩阵（Rb）

名称	类型	描述
aTranslationVectorB	TcVnVector3_LREAL [▶ 153]	第二个解的平移向量（Tb）
aNormVectorB	TcVnVector3_LREAL [▶ 153]	第二个解的范数向量 Nb

输出

名称	类型	描述
nSolutions	UDINT	返回解的数量。正常情况下等于 4。

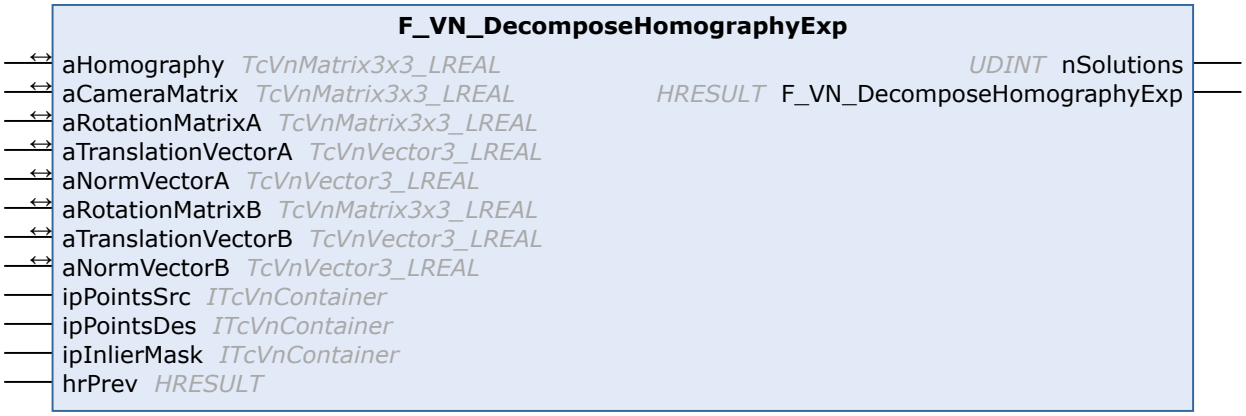
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.31 F_VN_DecomposeHomographyExp



分解单应性矩阵，并验证旋转、平移和平面法线的四个解。它利用相机前的可见参考点来确认四个解中有哪几个解（最多两个）与所有参考点一致。

语法

定义：

```
FUNCTION F_VN_DecomposeHomographyExp : HRESULT
VAR_IN_OUT
    aHomography         : TcVnMatrix3x3_LREAL;
    aCameraMatrix       : TcVnMatrix3x3_LREAL;
    aRotationMatrixA    : TcVnMatrix3x3_LREAL;
    aTranslationVectorA : TcVnVector3_LREAL;
    aNormVectorA        : TcVnVector3_LREAL;
    aRotationMatrixB    : TcVnMatrix3x3_LREAL;
    aTranslationVectorB : TcVnVector3_LREAL;
    aNormVectorB        : TcVnVector3_LREAL;
END_VAR
VAR_INPUT
    ipPointsSrc         : ITcVnContainer;
    ipPointsDes         : ITcVnContainer;
    ipInlierMask        : ITcVnContainer;
    hrPrev              : HRESULT;
END_VAR
VAR_OUTPUT
    nSolutions          : UDINT;
END_VAR
```

输入

名称	类型	描述
ipPointsSrc	ITcVnContainer [▶ 365]	包含源点的容器（ContainerType_Vector_TcVnPoint2_REAL）
ipPointsDes	ITcVnContainer [▶ 365]	包含目标点的容器（数量与 ipPoints1 相同，ContainerType_Vector_TcVnPoint2_REAL）。
ipInlierMask	ITcVnContainer [▶ 365]	标记内群值的掩码（可选，如果不可用，则设为 0；ContainerType_Vector_SINT 或 ContainerType_Vector_USINT）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aHomography	TcVnMatrix3x3_LREAL [▶ 153]	单应性矩阵（两个平面之间的透视变换）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	相机矩阵
aRotationMatrixA	TcVnMatrix3x3_LREAL [▶ 153]	第一个解的旋转矩阵（Ra）
aTranslationVectorA	TcVnVector3_LREAL [▶ 153]	第一个解的平移向量（Ta）
aNormVectorA	TcVnVector3_LREAL [▶ 153]	第一个解的范数向量（Na）
aRotationMatrixB	TcVnMatrix3x3_LREAL [▶ 153]	第二个解的旋转矩阵（Rb）
aTranslationVectorB	TcVnVector3_LREAL [▶ 153]	第二个解的平移向量（Tb）
aNormVectorB	TcVnVector3_LREAL [▶ 153]	第二个解的范数向量（Nb）

输出

名称	类型	描述
nSolutions	UDINT	返回潜在真解的数量 [0、1 或 2]。 0：没有确认给定相应点的潜在真解。 1：确认解（Ra、Ta、Na）。 2：解（Ra、Ta、Na）和（Rb、Tb、Nb）均得到确认。 在所有情况下，函数都会返回两个解（Ra、Ta、Nb）和（Rb、Tb、Nb）。第三个和第四个解的计算方法如下：（Ra、-Ta、-Na）和（Rb、-Tb、-Nb）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.32 F_VN_FlipImage



翻转图像（镜像并移回之前的坐标）。

语法

定义：

```
FUNCTION F_VN_FlipImage : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    eFlipAxis   : ETcVnFlipAxis;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eFlipAxis	ETcVnFlipAxis [▶ 196]	选择绕其翻转图像的轴线
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

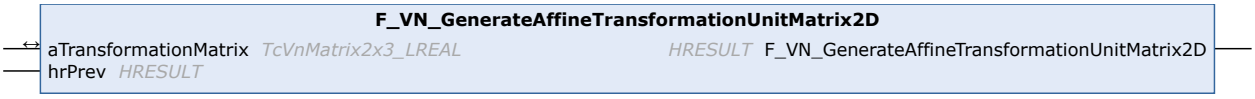
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.33 F_VN_GenerateAffineTransformationUnitMatrix2D



生成仿射变换二维单位矩阵。

语法

定义：

```
FUNCTION F_VN_GenerateAffineTransformationUnitMatrix2D : HRESULT
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	结果仿射变换单位矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.34 F_VN_GetAffineTransformation



计算三个相应点对之间的仿射变换。这些点标记相应三角形的角。

语法

定义：

```
FUNCTION F_VN_GetAffineTransformation : HRESULT
VAR_IN_OUT
    aSrcPoints            : TcVnArray3_Point2_REAL;
    aDestPoints           : TcVnArray3_Point2_REAL;
    aTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aSrcPoints	TcVnArray3_Point2_REAL [▶ 153]	源点
aDestPoints	TcVnArray3_Point2_REAL [▶ 153]	目标点

名称	类型	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	结果变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.35 F_VN_GetAffineTransformation2D



使用 RANSAC 算法估算两个平面点集之间的二维仿射变换。

语法

定义：

```
FUNCTION F_VN_GetAffineTransformation2D : HRESULT
VAR_INPUT
    ipSrcPoints      : ITcVnContainer;
    ipDestPoints     : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aAffineTransform : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPoints	ITcVnContainer [▶ 365]	至少包含 3 个源点的容器（ContainerType_Vector_TcVnPoint2_REAL）
ipDestPoints	ITcVnContainer [▶ 365]	包含目标点的容器（数量与 ipSrcPoints 相同，ContainerType_Vector_TcVnPoint2_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aAffineTransform	TcVnMatrix2x3_LREAL [▶ 153]	返回将源点转换为目标点的仿射变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.36 F_VN_GetAffineTransformation2DExp



估算两个平面点集之间的仿射变换。（专家函数）

语法

定义：

```
FUNCTION F_VN_GetAffineTransformation2DExp : HRESULT
VAR_INPUT
    ipSrcPoints      : ITcVnContainer;
    ipDestPoints     : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aAffineTransform : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    eAlgorithm       : ETcVnEstimationAlgorithm;
    fReprojThreshold : LREAL;
    ipInlierMask     : Reference To ITcVnContainer;
    nMaxIterations   : UDINT;
    fConfidence      : LREAL;
    nRefineIters     : UDINT;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPoints	ITcVnContainer [▶ 365]	至少包含 3 个源点的容器（ContainerType_Vector_TcVnPoint2_REAL）
ipDestPoints	ITcVnContainer [▶ 365]	包含目标点的容器（数量与 ipSrcPoints 相同，ContainerType_Vector_TcVnPoint2_REAL）
eAlgorithm	ETcVnEstimationAlgorithm [▶ 192]	估计算法（仅支持 RANSAC 和 LMEDS）
fReprojThreshold	LREAL	将点对视为内群时允许的最大重投影误差
ipInlierMask	引用 ITcVnContainer [▶ 365]	返回标记内群值的掩码（可选，如果不需要，则设为 0；ContainerType_Vector_USINT）
nMaxIterations	UDINT	最大迭代次数
fConfidence	LREAL	置信度（0..1）
nRefineIters	UDINT	Levenberg-Marquardt 算法进一步改进仿射变换（仅使用内群值）的最大迭代次数。设为 0 则禁用改进。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aAffineTransform	TcVnMatrix2x3_LREAL [▶ 153]	返回将源点转换为目标点的仿射变换矩阵

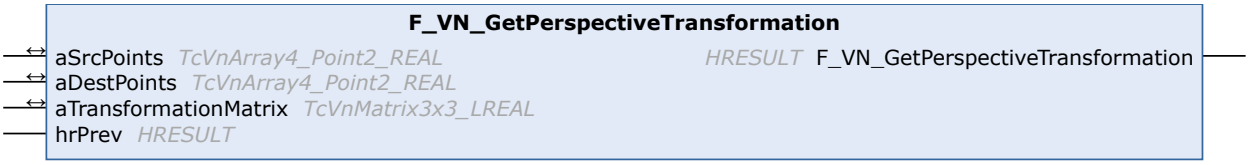
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.37 F_VN_GetPerspectiveTransformation



计算四个对应点对之间的透视变换。这些点标记相应矩形的角。

语法

定义：

```
FUNCTION F_VN_GetPerspectiveTransformation : HRESULT
VAR_IN_OUT
    aSrcPoints            : TcVnArray4_Point2_REAL;
    aDestPoints           : TcVnArray4_Point2_REAL;
    aTransformationMatrix : TcVnMatrix3x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aSrcPoints	TcVnArray4_Point2_REAL [▶ 153]	源点
aDestPoints	TcVnArray4_Point2_REAL [▶ 153]	目标点
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	结果变换矩阵

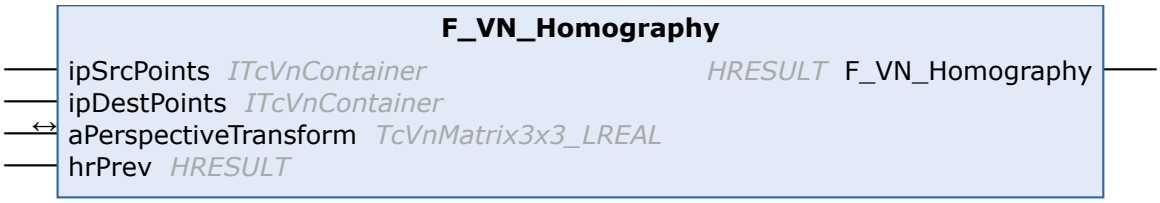
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.38 F_VN_Homography



使用 RANSAC 算法查找两个平面点集之间的单应性（透视变换）。

语法

定义：

```
FUNCTION F_VN_Homography : HRESULT
VAR_INPUT
    ipSrcPoints           : ITcVnContainer;
    ipDestPoints          : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aPerspectiveTransform : TcVnMatrix3x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPoints	ITcVnContainer [▶ 365]	至少包含 4 个源点的容器（ContainerType_Vector_TcVnPoint2_REAL）
ipDestPoints	ITcVnContainer [▶ 365]	包含目标点的容器（数量与 ipSrcPoints 相同，ContainerType_Vector_TcVnPoint2_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aPerspectiveTransform	TcVnMatrix3x3_LREAL [▶ 153]	返回将源点转换为目标点的透视变换矩阵

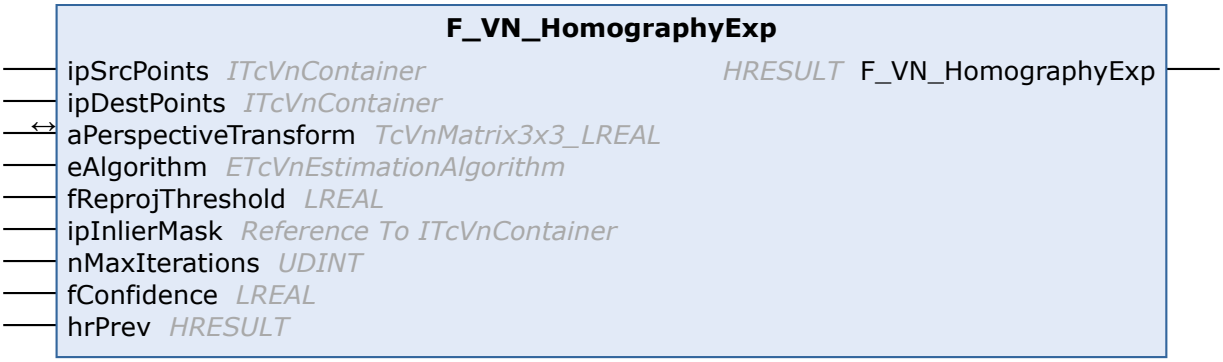
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.39 F_VN_HomographyExp



找出两个平面点集之间的单应性（透视变换）。（专家函数）

语法

定义：

```
FUNCTION F_VN_HomographyExp : HRESULT
VAR_INPUT
    ipSrcPoints           : ITcVnContainer;
    ipDestPoints          : ITcVnContainer;
END_VAR
VAR_IN_OUT
    aPerspectiveTransform : TcVnMatrix3x3_LREAL;
END_VAR
VAR_INPUT
    eAlgorithm            : ETcVnEstimationAlgorithm;
    fReprojThreshold      : LREAL;
    ipInlierMask          : Reference To ITcVnContainer;
    nMaxIterations        : UDINT;
    fConfidence           : LREAL;
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPoints	ITcVnContainer [▶ 365]	至少包含 4 个源点的容器（ContainerType_Vector_TcVnPoint2_REAL）
ipDestPoints	ITcVnContainer [▶ 365]	包含目标点的容器（数量与 ipSrcPoints 相同，ContainerType_Vector_TcVnPoint2_REAL）
eAlgorithm	ETcVnEstimationAlgorithm [▶ 192]	估计算法
fReprojThreshold	LREAL	将点对视为内群时允许的最大重投影误差（仅适用于 RANSAC 和 RHO）
ipInlierMask	引用 ITcVnContainer [▶ 365]	返回标记内群值的掩码（可选，如果不需要，则设为 0；ContainerType_Vector_SINT；仅适用于 RANSAC 和 LMEDS）
nMaxIterations	UDINT	最大 RANSAC 迭代次数
fConfidence	LREAL	置信度（0..1）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aPerspectiveTransform	TcVnMatrix3x3_LREAL [▶ 153]	返回将源点转换为目标点的透视变换矩阵

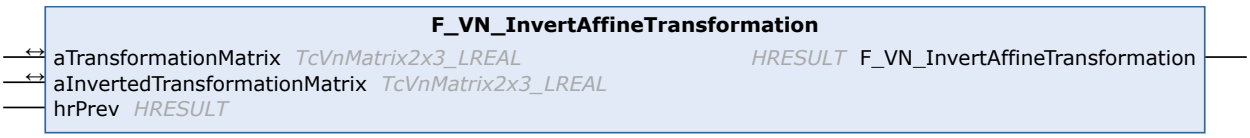
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.40 F_VN_InvertAffineTransformation



Invert a 2D affine transformation matrix. The inverted transformation compensates the affine transformation on an image if it is used in the F_VN_WarpAffine function.

语法

定义:

```
FUNCTION F_VN_InvertAffineTransformation : HRESULT
VAR_IN_OUT
    aTransformationMatrix         : TcVnMatrix2x3_LREAL;
    aInvertedTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                        : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	源仿射变换矩阵
aInvertedTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	结果逆仿射变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.41 F_VN_PerspectiveTransformation



Apply a perspective transformation (rotation + translation, e.g. from extrinsic calibration) to 3D point coordinates.

语法

定义：

```
FUNCTION F_VN_PerspectiveTransformation : HRESULT
VAR_INPUT
    ipSrcPoints         : ITcVnContainer;
    ipDestPoints        : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aRotationMatrix     : TcVnMatrix3x3_LREAL;
    aTranslationVector : TcVnVector3_LREAL;
END_VAR
VAR_INPUT
    hrPrev              : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPoints	ITcVnContainer [▶ 365]	包含三维源点的容器（TcVnPoint3_REAL 或 TcVnPoint3_LREAL）
ipDestPoints	引用 ITcVnContainer [▶ 365]	返回变换后的点（与 ipSrcPoints 类型相同）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	3x3 旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 153]	平移矢量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.42 F_VN_PyramidDown



Downsamples an image to half width and height.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_PyramidDown : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.43 F_VN_PyramidUp



Upsamples an image to double width and height.

语法

定义：

```
FUNCTION F_VN_PyramidUp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
```

```
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.44 F_VN_RemapImage



使用自定义映射对图像进行重映射，即 ipDestImage(x,y) = ipSrcImage(mapX(x,y), mapY(x,y))。支持的映射对包括（ipMap1 = ET_REAL，2 通道；ipMap2 = 0）、（ipMap1 = ET_REAL，1 通道；ipMap2 = ET_REAL，1 通道）、（ipMap1 = ET_INT，2 通道；ipMap2 = ET_UINT，1 通道）、（ipMap1 = ET_INT，2 通道；ipMap2 = 0）。
可以使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_RemapImage : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    ipMap1      : ITcVnImage;
    ipMap2      : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（不得与 ipSrcImage 相同！与 ipMap1 尺寸相同，与 ipSrcImage 类型相同。如有需要，将创建适当的目标图像。）

名称	类型	描述
ipMap1	ITcVnImage [▶ 407]	映射 1（ET_REAL 带 2 通道（包含 mapX、mapY），或 ET_REAL 带 1 通道（包含 mapX），或 ET_INT 带 2 通道（包含取整后的 mapX、mapY））
ipMap2	ITcVnImage [▶ 407]	映射 2（ET_REAL 带 1 通道（包含 mapY），或 ET_UINT 带 1 通道（如果 ipMap1 为 ET_INT 带 2 通道，则包含插值表系数），如果不需要，则设为 0。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

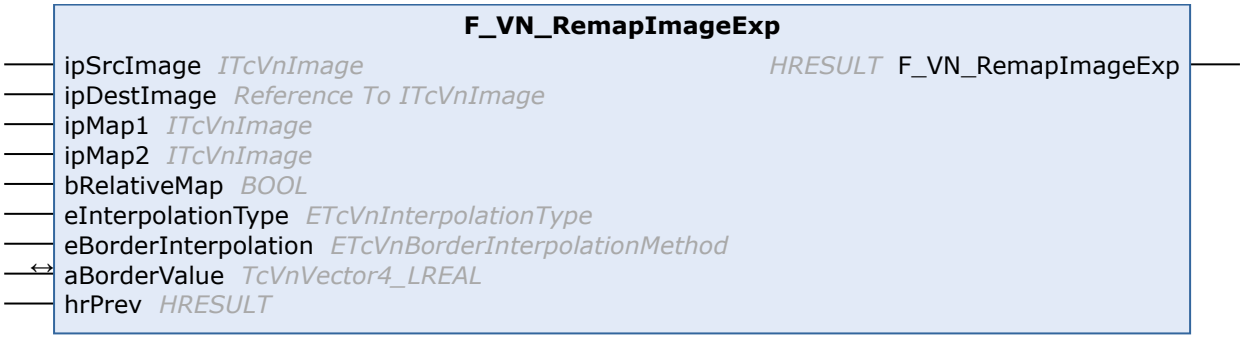
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.45 F_VN_RemapImageExp



使用自定义映射对图像进行重映射，即 ipDestImage(x,y) = ipSrcImage(mapX(x,y), mapY(x,y))。支持的映射对包括（ipMap1 = ET_REAL，2 通道；ipMap2 = 0）、（ipMap1 = ET_REAL，1 通道；ipMap2 = ET_REAL，1 通道）、（ipMap1 = ET_INT，2 通道；ipMap2 = ET_UINT，1 通道）、（ipMap1 = ET_INT，2 通道；ipMap2 = 0）。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_RemapImageExp : HRESULT
VAR_INPUT
    ipSrcImage           : ITcVnImage;
    ipDestImage          : Reference To ITcVnImage;
    ipMap1               : ITcVnImage;
    ipMap2               : ITcVnImage;
    bRelativeMap         : BOOL;
    eInterpolationType   : ETcVnInterpolationType;
    eBorderInterpolation : ETcVnBorderInterpolationMethod;
END_VAR
VAR_IN_OUT
    aBorderValue         : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev               : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（不得与 ipSrcImage 相同！与 ipMap1 尺寸相同，与 ipSrcImage 类型相同。如有需要，将创建适当的目标图像。）

名称	类型	描述
ipMap1	ITcVnImage [▶ 407]	映射 1（ET_REAL 带 2 通道（包含 mapX、mapY），或 ET_REAL 带 1 通道（包含 mapX），或 ET_INT 带 2 通道（包含取整后的 mapX、mapY））
ipMap2	ITcVnImage [▶ 407]	映射 2（ET_REAL 带 1 通道（包含 mapY），或 ET_UINT 带 1 通道（如果 ipMap1 为 ET_INT 带 2 通道，则包含插值表系数），如果不需要，则设为 0。
bRelativeMap	BOOL	如果为 true，则映射包含相对坐标，即 ipDestImage(x,y) = ipSrcImage(x + mapX(x,y), y + mapY(x,y))。
eInterpolationType	ETcVnInterpolationType [▶ 198]	插值类型（不支持 AREA_BASED、BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）
eBorderInterpolation	ETcVnBorderInterpolationMethod [▶ 160]	边界插值法（不支持 ISOLATED）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aBorderValue	TcVnVector4_LREAL [▶ 153]	边界值（如果使用 CONSTANT）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.46 F_VN_RemapImageToLogPolarSpace



将图像重映射到对数极空间。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_RemapImageToLogPolarSpace : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aCenter     : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（不得与 ipSrcImage 相同！如有需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCenter	TcVnPoint2_REAL [▶ 152]	变换中心点

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.47 F_VN_RemapImageToLogPolarSpaceExp



将图像重映射到对数极空间。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_RemapImageToLogPolarSpaceExp : HRESULT
VAR_INPUT
    ipSrcImage          : ITcVnImage;
    ipDestImage         : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aCenter             : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    fScale              : LREAL;
    eInterpolationType  : ETcVnInterpolationType;
    hrPrev              : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（不得与 ipSrcImage 相同！如有需要，将创建适当的目标图像。）
fScale	LREAL	幅值标度参数，即图像宽度/ln（半径）（自动选择则设为 0）

名称	类型	描述
eInterpolationType	ETcVnInterpolationType [▶ 198]	插值类型（不支持 BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCenter	TcVnPoint2_REAL [▶ 152]	变换中心点

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.48 F_VN_RemapImageToLogPolarSpaceExp2



将图像重映射到对数极空间。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_RemapImageToLogPolarSpaceExp2 : HRESULT
VAR_INPUT
    ipSrcImage          : ITcVnImage;
    ipDestImage         : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aCenter             : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    fScale              : LREAL;
    eInterpolationType  : ETcVnInterpolationType;
    nDestWidth          : DINT;
    nDestHeight         : DINT;
    hrPrev              : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（不得与 ipSrcImage 相同！如有需要，将创建适当的目标图像。）
fScale	LREAL	幅值标度参数，即图像宽度/ln（半径）（自动选择则设为 0）
eInterpolationType	ETcVnInterpolationType [▶ 198]	插值类型（不支持 BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）

名称	类型	描述
nDestWidth	DINT	目标图像宽度（设为 -1 表示源图像宽度，0 表示自动缩放至转换的半径，> 0 表示用户自定义的宽度）
nDestHeight	DINT	目标图像高度（设为 -1 表示源图像高度，0 表示自动缩放至转换的半径 * PI，> 0 表示用户自定义的高度）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCenter	TcVnPoint2_REAL [▶ 152]	变换中心点

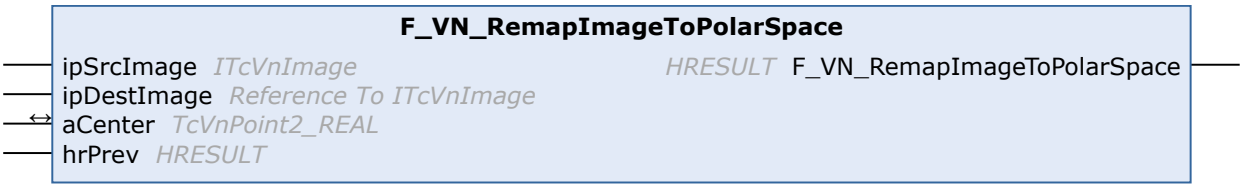
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.49 F_VN_RemapImageToPolarSpace



将图像重映射到极空间。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_RemapImageToPolarSpace : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aCenter     : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（不得与 ipSrcImage 相同！如有需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCenter	TcVnPoint2_REAL [▶ 152]	变换中心点

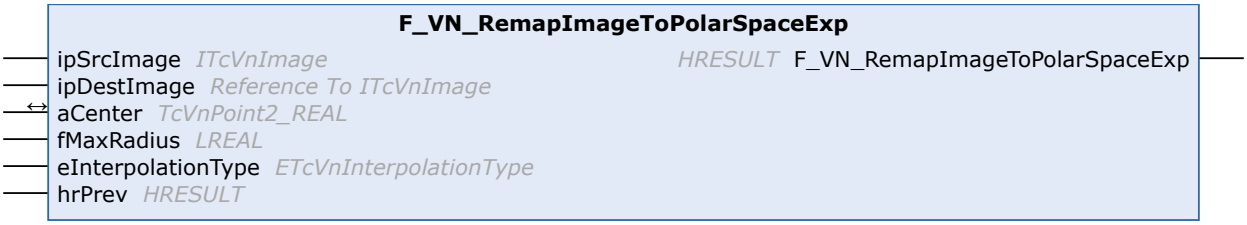
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.50 F_VN_RemapImageToPolarSpaceExp



将图像重新映射到极坐标空间。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_RemapImageToPolarSpaceExp : HRESULT
VAR_INPUT
    ipSrcImage          : ITcVnImage;
    ipDestImage         : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aCenter             : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    fMaxRadius          : LREAL;
    eInterpolationType  : ETcVnInterpolationType;
    hrPrev              : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（不得与 ipSrcImage 相同！如有需要，将创建适当的目标图像。）
fMaxRadius	LREAL	变换的最大半径（自动选择则设为 0）
eInterpolationType	ETcVnInterpolationType [▶ 198]	插值类型（不支持 BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCenter	TcVnPoint2_REAL [▶ 152]	变换中心点

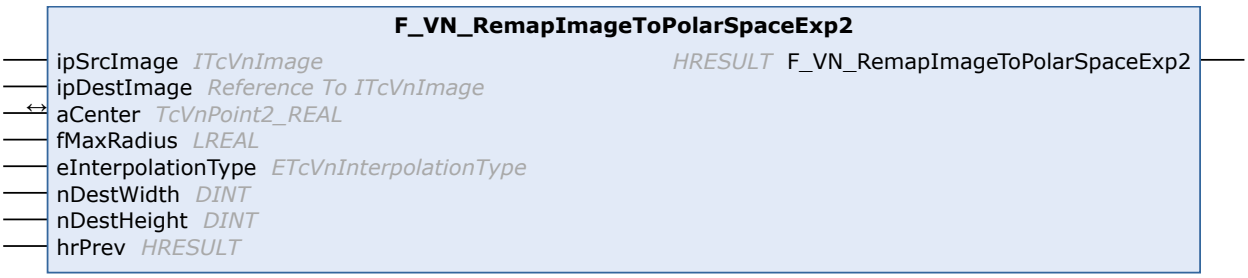
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.51 F_VN_RemapImageToPolarSpaceExp2



将图像重新映射到极坐标空间。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_RemapImageToPolarSpaceExp2 : HRESULT
VAR_INPUT
    ipSrcImage          : ITcVnImage;
    ipDestImage         : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aCenter             : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    fMaxRadius          : LREAL;
    eInterpolationType  : ETcVnInterpolationType;
    nDestWidth          : DINT;
    nDestHeight         : DINT;
    hrPrev              : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（不得与 ipSrcImage 相同！如有需要，将创建适当的目标图像。）
fMaxRadius	LREAL	变换的最大半径（自动选择则设为 0）
eInterpolationType	ETcVnInterpolationType [▶ 198]	插值类型（不支持 BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）
nDestWidth	DINT	目标图像宽度（设为 -1 表示源图像宽度，0 表示自动缩放至转换的半径，> 0 表示用户自定义的宽度）
nDestHeight	DINT	目标图像高度（设为 -1 表示源图像高度，0 表示自动缩放至转换的半径 * PI，> 0 表示用户自定义的高度）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCenter	TcVnPoint2_REAL [▶ 152]	变换中心点

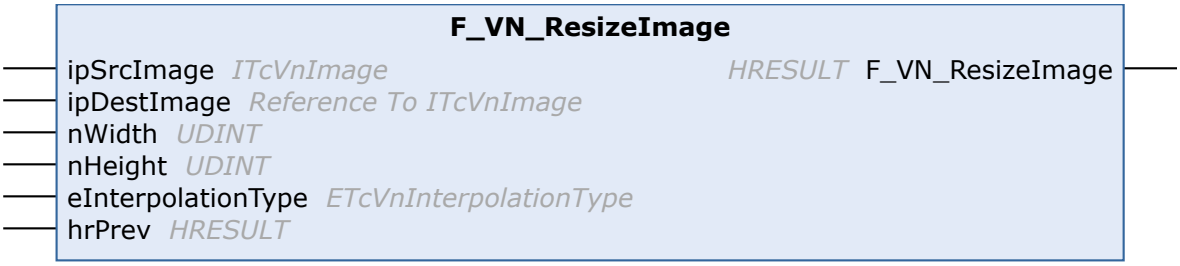
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.52 F_VN_ResizeImage



使用特定的插值类型重新定义图像大小。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被看门狗取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_ResizeImage : HRESULT
VAR_INPUT
    ipSrcImage          : ITcVnImage;
    ipDestImage         : Reference To ITcVnImage;
    nWidth              : UDINT;
    nHeight             : UDINT;
    eInterpolationType  : ETcVnInterpolationType;
    hrPrev              : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
nWidth	UDINT	新宽度
nHeight	UDINT	新高度
eInterpolationType	ETcVnInterpolationType [▶ 198]	插值类型（SINT 或 DINT 类型的图像仅支持 NEAREST_NEIGHBOR）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

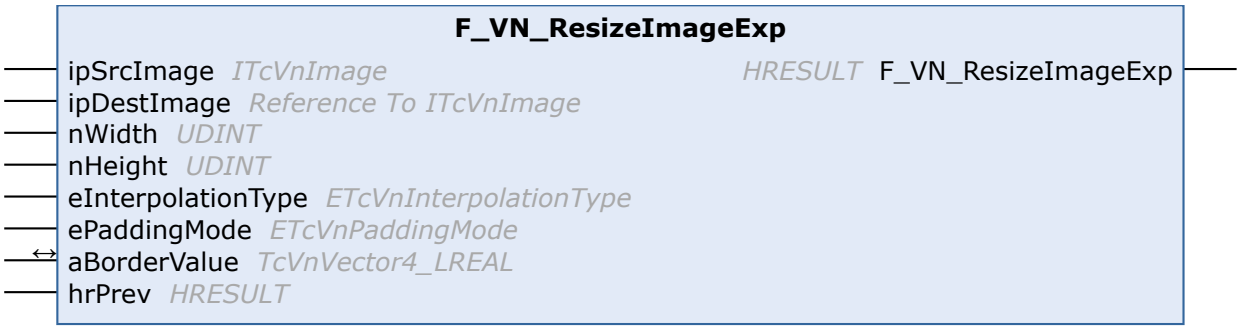
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.53 F_VN_ResizeImageExp



使用特定的插值类型调整图像大小，同时保持其长宽比。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被看门狗取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_ResizeImageExp : HRESULT
VAR_INPUT
    ipSrcImage          : ITcVnImage;
    ipDestImage         : Reference To ITcVnImage;
    nWidth              : UDINT;
    nHeight             : UDINT;
    eInterpolationType  : ETcVnInterpolationType;
    ePaddingMode        : ETcVnPaddingMode;
END_VAR
VAR_IN_OUT
    aBorderValue        : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev              : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
nWidth	UDINT	新宽度
nHeight	UDINT	新高度
eInterpolationType	ETcVnInterpolationType [▶ 198]	插值类型（SINT 或 DINT 类型的图像仅支持 NEAREST_NEIGHBOR）
ePaddingMode	ETcVnPaddingMode [▶ 205]	图像填充模式
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aBorderValue	TcVnVector4_LREAL [▶ 153]	边界值，如果使用 TCVN_PM_LETTERBOX

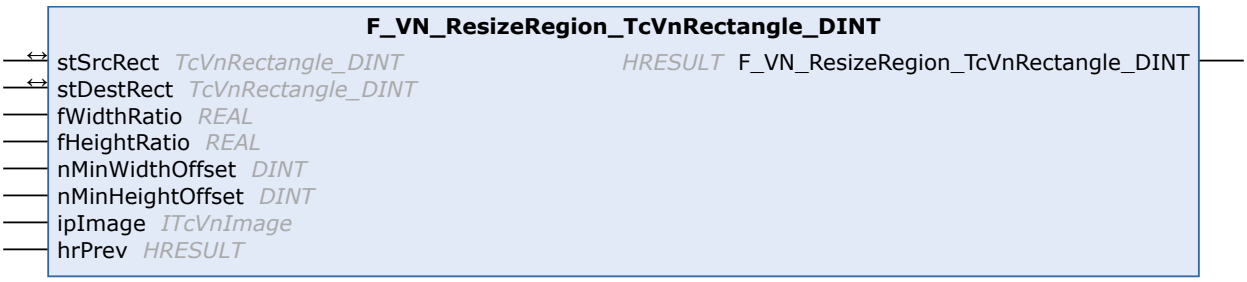
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.54 F_VN_ResizeRegion_TcVnRectangle_DINT



按给定矩形当前大小的比例和/或通过显式的绝对偏移量来扩展或缩小该矩形。水平和垂直缩放比定义了按比例的尺寸变化。如果缩放比等于 1.0，则仅由相应的最小偏移量决定尺寸变化。如果提供了参考图像，则生成的矩形将被裁剪到图像边界内。如果发生了裁剪，则函数返回 S_FALSE。

语法

定义：

```
FUNCTION F_VN_ResizeRegion_TcVnRectangle_DINT : HRESULT
VAR_IN_OUT
    stSrcRect         : TcVnRectangle_DINT;
    stDestRect        : TcVnRectangle_DINT;
END_VAR
VAR_INPUT
    fWidthRatio       : REAL;
    fHeightRatio      : REAL;
    nMinWidthOffset   : DINT;
    nMinHeightOffset  : DINT;
    ipImage           : ITcVnImage;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
fWidthRatio	REAL	水平缩放比。如果 fWidthRatio <> 1（> 1 表示扩展，< 1 表示收缩），则 nMinWidthOffset 的绝对值可用作最小宽度变化约束。如果 fWidthRatio = 1.0，则宽度变化仅由 nMinWidthOffset 定义。
fHeightRatio	REAL	垂直缩放比。如果 fHeightRatio <> 1（> 1 表示扩展，< 1 表示收缩），则 nMinHeightOffset 的绝对值可用作最小宽度变化约束。如果 fHeightRatio = 1.0，则高度变化仅由 nMinHeightOffset 定义。
nMinWidthOffset	DINT	最小宽度偏移量（以像素为单位），围绕矩形中心对称应用。如果 fWidthRatio <> 1.0，则 nMinWidthOffset 的绝对值定义了最小宽度变化。如果 fWidthRatio = 1.0，则符号定义了操作方向（> 0 表示扩展，< 0 表示收缩，= 0 表示宽度不变）。
nMinHeightOffset	DINT	最小高度偏移量（以像素为单位），围绕矩形中心对称应用。如果 fHeightRatio <> 1.0，则 nMinHeightOffset 的绝对值定义了最小高度变化。如果 fHeightRatio = 1.0，则符号定义了操作方向（> 0 表示扩展，< 0 表示收缩，= 0 表示高度不变）。
ipImage	ITcVnImage [▶ 407]	可选参考图像，用于定义裁剪边界，如果不需要，则设为 0。

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stSrcRect	TcVnRectangle_DINT [▶ 236]	要扩展或收缩的源矩形。
stDestRect	TcVnRectangle_DINT [▶ 236]	扩展或收缩后生成的矩形。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.14.55 F_VN_RotateImage



Rotate an image by 90, 180, or 270 degrees in clockwise direction.

语法

定义：

```
FUNCTION F_VN_RotateImage : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    eAngle      : ETcVnRotationAngle;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
eAngle	ETcVnRotationAngle [▶ 209]	旋转角
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

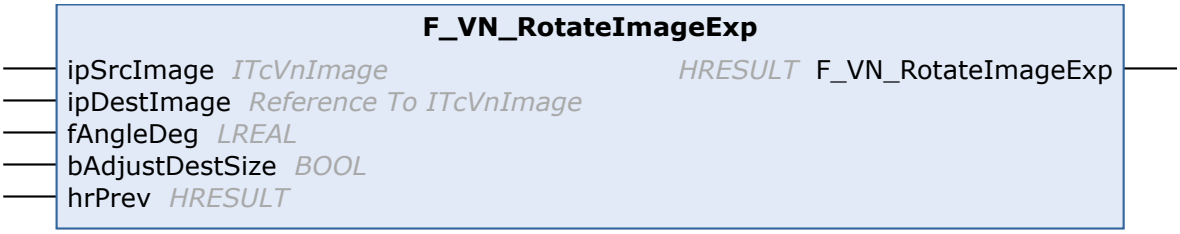
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.56 F_VN_RotateImageExp



Rotate an image by an individual rotation angle in degrees.
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_RotateImageExp : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipDestImage     : Reference To ITcVnImage;
    fAngleDeg       : LREAL;
    bAdjustDestSize : BOOL;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
fAngleDeg	LREAL	旋转角（以度为单位）（正：逆时针，负：顺时针）
bAdjustDestSize	BOOL	如果为 true，将调整 ipDestImage 的大小，从而包含整个旋转后的 ipSrcImage。否则，ipDestImage 大小将设置为 ipSrcImage 大小。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

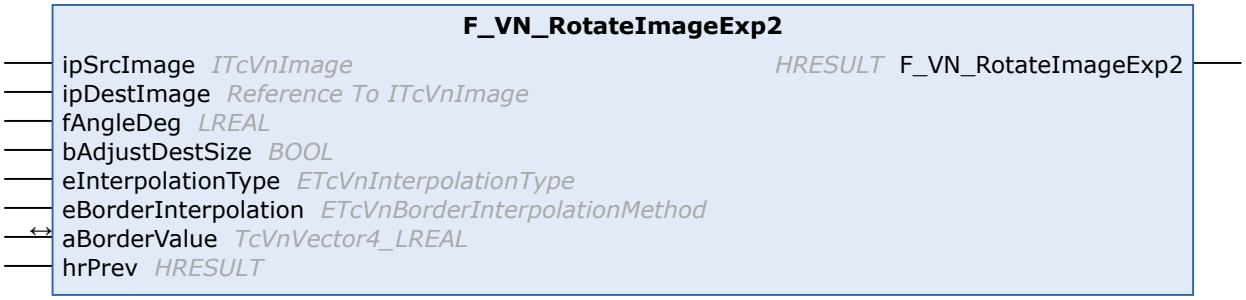
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.57 F_VN_RotateImageExp2



Rotate an image by an individual rotation angle in degrees. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_RotateImageExp2 : HRESULT
VAR_INPUT
    ipSrcImage           : ITcVnImage;
    ipDestImage          : Reference To ITcVnImage;
    fAngleDeg            : LREAL;
    bAdjustDestSize      : BOOL;
    eInterpolationType   : ETcVnInterpolationType;
    eBorderInterpolation : ETcVnBorderInterpolationMethod;
END_VAR
VAR_IN_OUT
    aBorderValue         : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev               : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
fAngleDeg	LREAL	旋转角（以度为单位）（正：逆时针，负：顺时针）
bAdjustDestSize	BOOL	如果为 true，将调整 ipDestImage 的大小，从而包含整个旋转后的 ipSrcImage。否则，ipDestImage 大小将设置为 ipSrcImage 大小。
eInterpolationType	ETcVnInterpolationType [▶ 198]	插值法（不支持 BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）
eBorderInterpolation	ETcVnBorderInterpolationMethod [▶ 160]	边界插值法（不支持 ISOLATED）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aBorderValue	TcVnVector4_LREAL [▶ 153]	边界值（如果使用 CONSTANT）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.58 F_VN_WarpAffine



Apply an affine transformation to an image.
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_WarpAffine : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
    ipDestImage           : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	仿射变换矩阵

返回值

HRESULT [▶ 135]

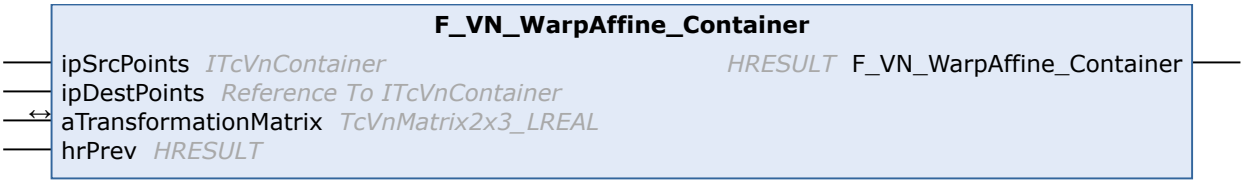
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.59 F_VN_WarpAffine_Container



Apply an affine transformation to a container of 2D points.

语法

定义：

```
FUNCTION F_VN_WarpAffine_Container : HRESULT
VAR_INPUT
    ipSrcPoints           : ITcVnContainer;
    ipDestPoints          : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPoints	ITcVnContainer [▶ 365]	源点（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL）
ipDestPoints	引用 ITcVnContainer [▶ 365]	返回变换后的点（与 ipSrcPoints 的类型 ID 相同；重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	仿射变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.60 F_VN_WarpAffine_Point



Apply an affine transformation to a 2D point.

语法

定义：

```
FUNCTION F_VN_WarpAffine_Point : HRESULT
VAR_IN_OUT
    aSrcPoint             : TcVnPoint2_LREAL;
    aDestPoint            : TcVnPoint2_LREAL;
    aTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aSrcPoint	TcVnPoint2_LREAL [▶ 152]	源点
aDestPoint	TcVnPoint2_LREAL [▶ 152]	目标点
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	仿射变换矩阵

返回值

HRESULT [▶ 135]

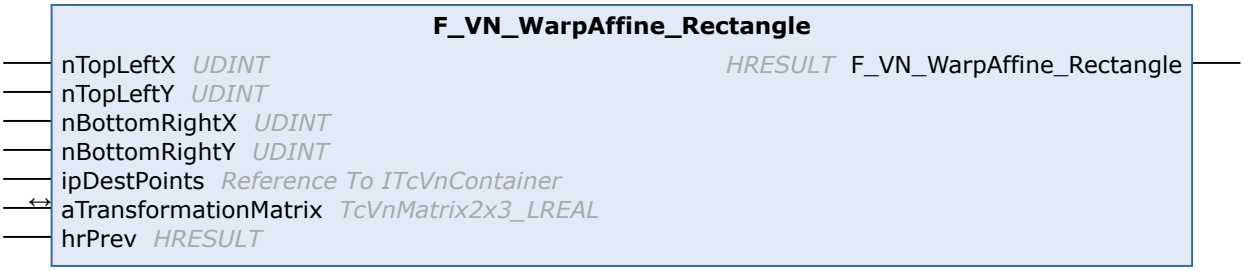
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.61 F_VN_WarpAffine_Rectangle



Use an affine transform to warp a rectangle.

语法

定义：

```
FUNCTION F_VN_WarpAffine_Rectangle : HRESULT
VAR_INPUT
    nTopLeftX             : UDINT;
    nTopLeftY             : UDINT;
    nBottomRightX         : UDINT;
    nBottomRightY         : UDINT;
    ipDestPoints          : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
nTopLeftX	UDINT	左上角的 x 坐标
nTopLeftY	UDINT	左上角的 y 坐标
nBottomRightX	UDINT	右下角的 x 坐标
nBottomRightY	UDINT	右下角的 y 坐标
ipDestPoints	引用 ITcVnContainer [▶ 365]	返回 4 个变换后的点（ContainerType_Vector_TcVnPoint2_REAL）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	仿射变换矩阵

返回值

HRESULT [▶ 135]

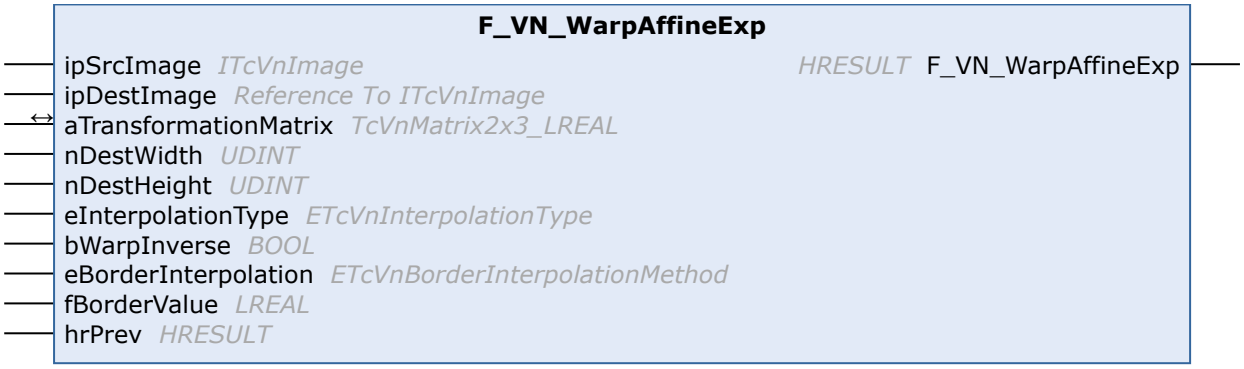
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.62 F_VN_WarpAffineExp



Apply an affine transformation to an image. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_WarpAffineExp : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
    ipDestImage           : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix2x3_LREAL;
END_VAR
VAR_INPUT
    nDestWidth            : UDINT;
    nDestHeight           : UDINT;
    eInterpolationType    : ETcVnInterpolationType;
    bWarpInverse          : BOOL;
    eBorderInterpolation  : ETcVnBorderInterpolationMethod;
    fBorderValue          : LREAL;
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
nDestWidth	UDINT	目标图像的宽度
nDestHeight	UDINT	目标图像的高度
eInterpolationType	ETcVnInterpolationType [▶ 198]	插值法（不支持 BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）
bWarpInverse	BOOL	将 aTransformationMatrix 作为逆变换处理
eBorderInterpolation	ETcVnBorderInterpolationMethod [▶ 160]	边界插值法（不支持 ISOLATED）
fBorderValue	LREAL	边界值（如果使用 CONSTANT）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 153]	仿射变换矩阵

返回值

HRESULT [▶ 135]

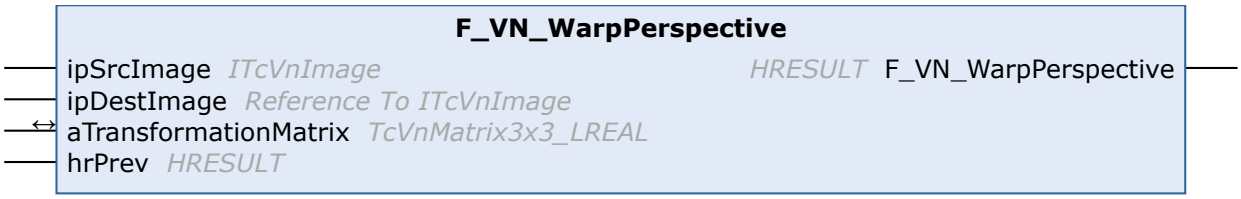
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.63 F_VN_WarpPerspective



Apply a perspective transformation to an image.
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_WarpPerspective : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
    ipDestImage           : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix3x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	透视变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.64 F_VN_WarpPerspective_Container



Apply a perspective transformation to a container of 2D points.

语法

定义：

```
FUNCTION F_VN_WarpPerspective_Container : HRESULT
VAR_INPUT
    ipSrcPoints           : ITcVnContainer;
    ipDestPoints          : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix3x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcPoints	ITcVnContainer [▶ 365]	源点（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL）
ipDestPoints	引用 ITcVnContainer [▶ 365]	返回变换后的点（与 ipSrcPoints 的类型 ID 相同；重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	透视变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.65 F_VN_WarpPerspective_Point



Apply a perspective transformation to a 2D point.

语法

定义：

```
FUNCTION F_VN_WarpPerspective_Point : HRESULT
VAR_IN_OUT
    aSrcPoint             : TcVnPoint2_LREAL;
    aDestPoint            : TcVnPoint2_LREAL;
    aTransformationMatrix : TcVnMatrix3x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aSrcPoint	TcVnPoint2_LREAL [▶ 152]	源点
aDestPoint	TcVnPoint2_LREAL [▶ 152]	目标点
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	透视变换矩阵

返回值

HRESULT [▶ 135]

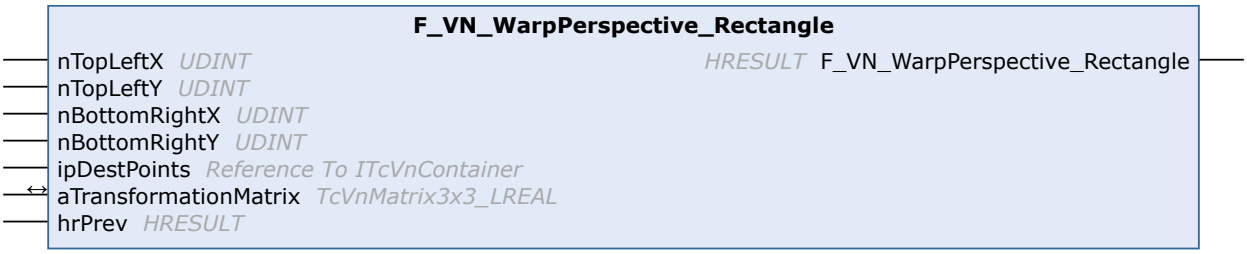
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.66 F_VN_WarpPerspective_Rectangle



Use a perspective transform to warp a rectangle.

语法

定义：

```
FUNCTION F_VN_WarpPerspective_Rectangle : HRESULT
VAR_INPUT
    nTopLeftX             : UDINT;
    nTopLeftY             : UDINT;
    nBottomRightX         : UDINT;
    nBottomRightY         : UDINT;
    ipDestPoints          : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix3x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
nTopLeftX	UDINT	左上角的 x 坐标
nTopLeftY	UDINT	左上角的 y 坐标
nBottomRightX	UDINT	右下角的 x 坐标
nBottomRightY	UDINT	右下角的 y 坐标
ipDestPoints	引用 ITcVnContainer [▶ 365]	返回 4 个变换后的点（ContainerType_Vector_TcVnPoint2_REAL）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	透视变换矩阵

返回值

HRESULT [▶ 135]

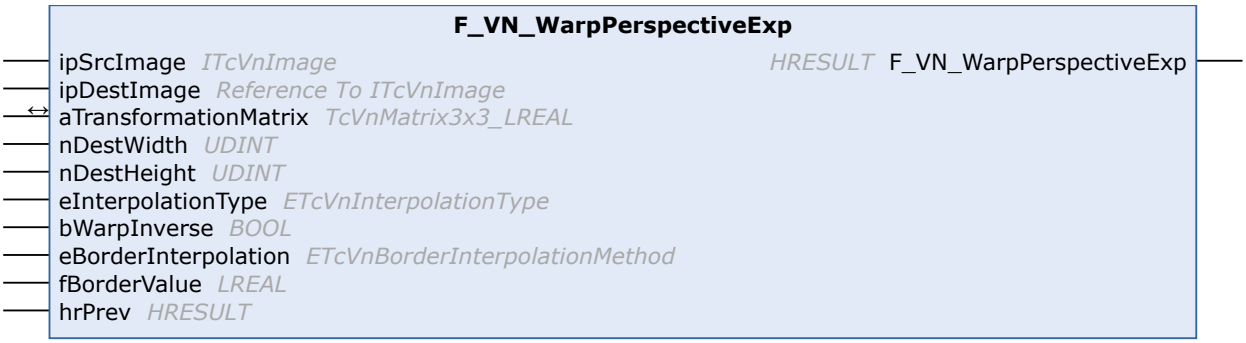
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.14.67 F_VN_WarpPerspectiveExp



对图像进行透视变换 （专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被看门狗取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_WarpPerspectiveExp : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
    ipDestImage           : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aTransformationMatrix : TcVnMatrix3x3_LREAL;
END_VAR
VAR_INPUT
    nDestWidth            : UDINT;
    nDestHeight           : UDINT;
    eInterpolationType    : ETcVnInterpolationType;
    bWarpInverse          : BOOL;
    eBorderInterpolation  : ETcVnBorderInterpolationMethod;
    fBorderValue          : LREAL;
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像

名称	类型	描述
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
nDestWidth	UDINT	目标图像的宽度
nDestHeight	UDINT	目标图像的高度
eInterpolationType	ETcVnInterpolationType [▶ 198]	插值法（不支持 BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）
bWarpInverse	BOOL	将 aTransformationMatrix 作为逆变换处理
eBorderInterpolation	ETcVnBorderInterpolationMethod [▶ 160]	边界插值法（不支持 ISOLATED）
fBorderValue	LREAL	边界值（如果使用 CONSTANT）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	透视变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.15 图像分析

6.1.4.15.1 目标检测

该子组包含用于物体识别的函数。

函数

目标检测

- F_VN_DetectBlobs(Exp) [▶ 1171]
- F_VN_FindContours(Exp) [▶ 1181]
- F_VN_FindContourHierarchy(Exp [▶ 1175])
- F_VN_FindTemplateWalsh(Exp) [▶ 1184]
- F_VN_FindTemplateWalsh_AngleRange(Exp) [▶ 1188]
- F_VN_HoughCircles(Exp) [▶ 1195]
- F_VN_HoughLines(Exp) [▶ 1197]
- F_VN_HoughLinesP(Exp) [▶ 1199]

匹配

- F_VN_MatchImageHuMoments [▶ 1200]
- F_VN_MatchTemplate(Exp) [▶ 1202]
- F_VN_MatchTemplateWalsh(Exp) [▶ 1205]

- F_VN_MatchTemplateAndEvaluate(Exp) [▶ 1204]

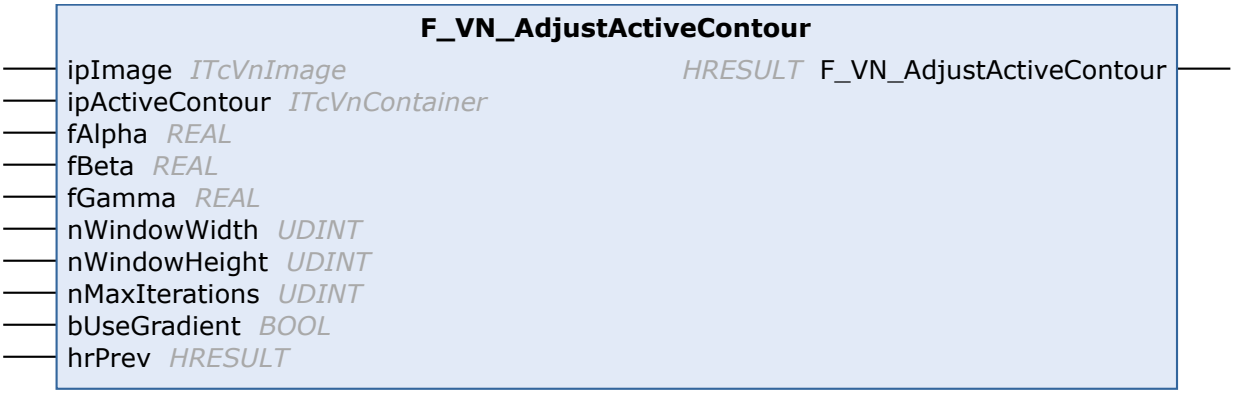
其他

- F_VN_AdjustActiveContour [▶ 1170]

样本

对象检测 [▶ 2755]

6.1.4.15.1.1 F_VN_AdjustActiveContour



Adjust active contour (snake) in order to minimize its cummulative (internal and external) energy.

语法

定义：

```
FUNCTION F_VN_AdjustActiveContour : HRESULT
VAR_INPUT
    ipImage          : ITcVnImage;
    ipActiveContour  : ITcVnContainer;
    fAlpha           : REAL;
    fBeta            : REAL;
    fGamma           : REAL;
    nWindowWidth     : UDINT;
    nWindowHeight    : UDINT;
    nMaxIterations   : UDINT;
    bUseGradient     : BOOL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	源图像（1 通道，USINT 元素）
ipActiveContour	ITcVnContainer [▶ 365]	初始轮廓，将进行调整
fAlpha	REAL	连续能量系数
fBeta	REAL	曲率能量系数
fGamma	REAL	图像能量系数
nWindowWidth	UDINT	窗口宽度（3、5、7……）
nWindowHeight	UDINT	窗口高度（3、5、7……）
nMaxIterations	UDINT	最大迭代
bUseGradient	BOOL	如果为 true，则使用梯度幅值作为图像能量（否则：像素亮度）

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

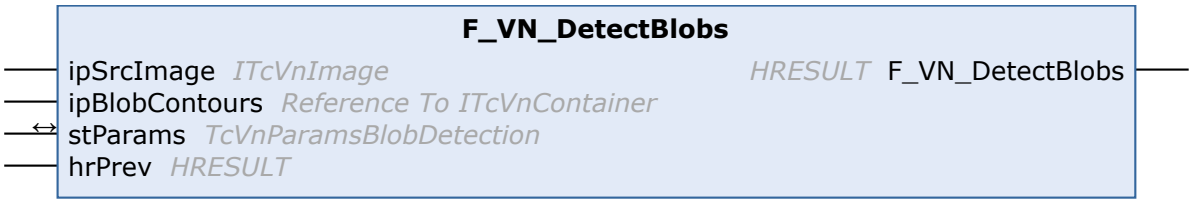
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.2 F_VN_DetectBlobs



Detects blob-contours. Applies a threshold, a contour search and offers several options for filtering the found contours. Provides easy setup for multiple thresholds and combination of results.
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_DetectBlobs : HRESULT
VAR_INPUT
    ipSrcImage     : ITcVnImage;
    ipBlobContours : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    stParams       : TcVnParamsBlobDetection;
END_VAR
VAR_INPUT
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 或 REAL，1 通道或 3 通道（3 通道输入应为 RGB 并内部转换为灰色））
ipBlobContours	引用 ITcVnContainer [▶ 365]	返回一个以所找到的轮廓填充的容器（ContainerType_Vector_Vector_TcVnPoint2_DINT；此容器的元素为单一轮廓。重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stParams	TcVnParamsBlobDetection [▶ 229]	用于筛选检测到的轮廓的参数。

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_DetectBlobs根据输入图像中可定义的特征检测斑点轮廓。在内部，这是通过类似于F_VN_FindContours [▶ 1179]的轮廓查找和随后根据轮廓特征对找到的轮廓进行过滤来进行的。

参数

输入图像

输入图像ipSrcImage必须是类型为USINT或REAL的单通道或三通道图像。三通道图像被解释为 RGB 并在内部转换为灰度图像。

找到的轮廓列表 (返回值)

在输入图像中发现的所有轮廓都会在容器ipBlobContours中返回。

斑点检测的参数

TcVnParamsBlobDetection [▶ 229]结构的参数stParams定义了输入图像中的轮廓必须具备哪些特征才能被找到并返回。

专家参数

专家版F_VN_DetectBlobsExp [▶ 1174]包含额外的参数。

斑点参数

阈值

在第一步，在内部使用一个或多个二进制图像进行轮廓搜索。为此，使用一个或多个阈值和阈值类型将输入图像转换为二进制图像。

设置一个单一的阈值对应于调用F_VN_Threshold [▶ 1322]。为此，fThresholdStep必须设置为 0。仅使用fMinThreshold的值。

另外，也可以考虑几个阈值。第一个阈值是fMinThreshold。从这个值开始，对于下一个阈值增加fThresholdStep，直到阈值大于fMaxThreshold。例如，如果需要考虑三个阈值 100、150 和 200，则参数化如下所示：

```
stBlobParams.fMinThreshold     := 100;
stBlobParams.fMaxThreshold     := 200;
stBlobParams.fThresholdStep    := 50;
```

相同轮廓，不同的阈值

可能发现相同的轮廓而阈值不同。相同轮廓通过中心点比较进行识别。在这种情况下，fMinBlobDistance指定不同轮廓的最小中心距离。如果没有达到这个距离，则根据eBlobCombination（ETcVnBlobCombination [▶ 159]）选择轮廓。

此外，在不同的阈值下多次找到一个轮廓可能为一个过滤标准。一个轮廓必须被找到多少次才能被考虑，这一点通过nMinRepeatability指定。

过滤器

在第二步，根据定义的参数对找到的轮廓进行过滤。各个过滤标准可以通过布尔变量激活。

面积（单位：像素）

- 参数：bFilterByArea、fMinArea、fMaxArea
- 或者，可以通过F_VN_ContourArea [▶ 965]确定轮廓的面积。

圆度

- 参数：bFilterByCircularity、fMinCicularity、fMaxCircularity
- 值范围从 0 到 1；一个完美的圆的圆度为 1。
- 另外，轮廓的圆度可以通过F_VN_ContourCircularity [▶ 967]确定。

凸性

- 参数：bFilterByConvexity、fMinConvexity、fMaxConvexity
- 其数值范围为 0 至 1。完美凸型的凸度为 1。

偏心度

- 参数：bFilterByEccentricity、fMinEccentricity、fMaxEccentricity
- 值范围从 0 到 1；完美圆形的偏心度为 0。
- 另外，轮廓的偏心度可以通过F_VN_ContourEccentricity [▶ 968]确定。

惯性比

- 参数：bFilterByInertiaRatio、fMinInertiaRatio、fMaxInertiaRatio
- 数值范围从 0 到 1；具有相同高度和宽度的形状的惯性比率为 1。

应用

```
VAR
    stBlobParams    :   TcVnParamsBlobDetection;
END_VAR

stBlobParams.bFilterByArea  :=  TRUE;
stBlobParams.fMinArea       :=  100;
stBlobParams.fMaxArea       :=  100_000;

hr := F_VN_DetectBlobs(
    ipSrcImage      :=  ipImageIn,
    ipBlobContours  :=  ipContours,
    stParams        :=  stBlobsParams,
    hrPrev          :=  hr
);
```

样本

- 带看门狗监控的斑点检测 [▶ 2755]

相关函数

- F_VN_FindContours(Exp) [▶ 1179]用于寻找一般的轮廓
- F_VN_FindContourHierarchyExp [▶ 1175]返回层次结构
- F_VN_DetectBlobs [▶ 1171]对轮廓进行综合过滤

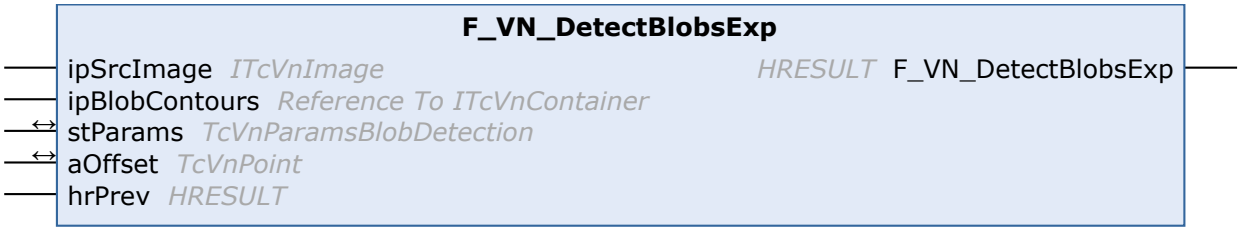
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.3 F_VN_DetectBlobsExp



Detects blob-contours. Applies a threshold, a contour search and offers several options for filtering the found contours. Provides easy setup for multiple thresholds and combination of results. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_DetectBlobsExp : HRESULT
VAR_INPUT
    ipSrcImage     : ITcVnImage;
    ipBlobContours : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    stParams       : TcVnParamsBlobDetection;
    aOffset        : TcVnPoint;
END_VAR
VAR_INPUT
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 或 REAL，1 通道或 3 通道（3 通道输入应为 RGB 并内部转换为灰色））
ipBlobContours	引用 ITcVnContainer [▶ 365]	返回一个以所找到的轮廓填充的容器（ContainerType_Vector_Vector_TcVnPoint2_DINT；此容器的元素为单一轮廓。重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stParams	TcVnParamsBlobDetection [▶ 229]	用于筛选检测到的轮廓的参数。
aOffset	TcVnPoint [▶ 152]	每个轮廓点移动的偏移量

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_DetectBlobsExp是F_VN_DetectBlobs [▶ 1171]的专家版本。它包含额外的参数。

参数

输入图像

输入图像ipSrcImage必须是类型为USINT或REAL的单通道或三通道图像。三通道图像被解释为 RGB 并在内部转换为灰度图像。

找到的轮廓列表（返回值）

在输入图像中发现的所有轮廓都会在容器ipBlobContours中返回。

斑点检测的参数

TcVnParamsBlobDetection [▶ 229]结构的参数stParams定义了输入图像中的轮廓必须具备哪些特征才能被找到并返回。

偏移量（专家）

偏移量aOffset定义了常量 X/Y 值，通过数值所有的轮廓点都被移位。当在一个 ROI 中寻找轮廓且它们的位置应该为原始图像提供参考时，这可能有所帮助。

相关函数

- F_VN_FindContours(Exp) [▶ 1179]用于寻找一般的轮廓
- F_VN_FindContourHierarchyExp [▶ 1175]返回层次结构
- F_VN_DetectBlobs [▶ 1171]对轮廓进行综合过滤

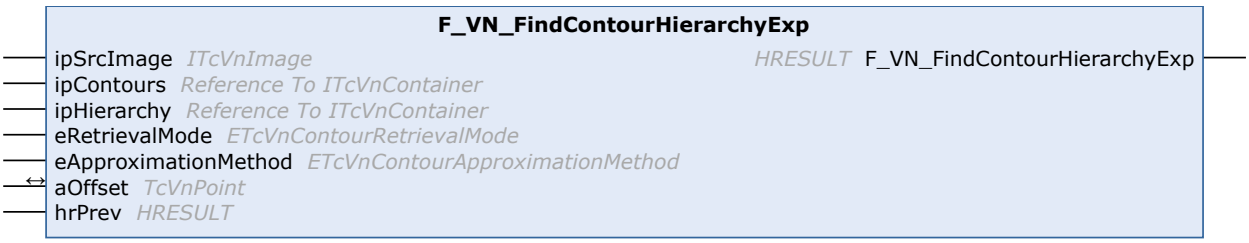
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.4 F_VN_FindContourHierarchyExp



Search for object contours in a binary image and determine their hierarchical relationship. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_FindContourHierarchyExp : HRESULT
VAR_INPUT
    ipSrcImage           : ITcVnImage;
    ipContours           : Reference To ITcVnContainer;
    ipHierarchy          : Reference To ITcVnContainer;
    eRetrievalMode       : ETcVnContourRetrievalMode;
    eApproximationMethod : ETcVnContourApproximationMethod;
END_VAR
VAR_IN_OUT
    aOffset              : TcVnPoint;
END_VAR
VAR_INPUT
    hrPrev               : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipSrcImage	ITcVnImage [▶ 407]	Source image (1 channel, binary, DINT if eRetrievalMode = CRM_FLOODFILL, otherwise USINT)
ipContours	Reference To ITcVnContainer [▶ 365]	Returns a container which is filled with the found contours (ContainerType_Vector_Vector_TcVnPoint2_DINT; The elements of this container are single contours. Non-zero interface pointers are reused.)
ipHierarchy	Reference To ITcVnContainer [▶ 365]	Returns a container which is filled with information on the found contours' hierarchy (ContainerType_Vector_TcVnVector4_DINT; The elements of this container are four-dimensional vectors containing the 0-based indices of the next [0] and the previous contour [1] at the same level, the first child [2], and the parent [3]. Non-zero interface pointers are reused.)
eRetrievalMode	ETcVnContourRetrievalMode [▶ 181]	Specifies which contours are retrieved and how their relationship is encoded
eApproximationMethod	ETcVnContourApproximationMethod [▶ 179]	Contour encoding
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

输入/输出

名称	类型	描述
aOffset	TcVnPoint [▶ 152]	每个轮廓点移动的偏移量

返回值

HRESULT [▶ 135]

更多信息

通过返回轮廓层次结构，函数`F_VN_FindContourHierarchyExp`对函数F_VN_FindContoursExp [▶ 1181]进行了扩展。

参数

输入图像

输入图像`ipSrcImage`必须正好有 1 个通道，而且应该是二进制图像。

如果输入图像不是二进制图像，它将通过 1 的阈值在内部转换成二进制图像。因此，输入的图像应该通过阈值或二进制分割进行手动预处理。

找到的轮廓列表（返回值）

在输入图像中发现的所有轮廓都会在容器ipContours中返回。容器是一个二维的点阵列，因此，其类型为ContainerType_Vector_Vector_TcVnPoint2_DINT。

找到的轮廓会以ipContours返回。每个轮廓由一个点的总和组成，这些点被组合成一个容器 [▶ 146]。然后，另一个更高层次的容器将所有的轮廓合并。

层次结构（专家，返回值）

参数ipHierarchy返回在 TcVnVector4_DINT 类型的容器 [▶ 146]中发现的轮廓的层次结构。容器的向量有以下元素：

[0] - NEXT - 同层下一个轮廓的索引
[1] - PREVIOUS - 同层上一个轮廓的索引
[2] - CHILD - 第一个子轮廓的索引
[3] - PARENT - 父轮廓的索引

搜索模式（专家）

ETcVnContourRetrievalMode [▶ 181]类型的搜索模式eRetrievalMode定义了要寻找哪些轮廓：

- TCVN_CRM_LIST：所有轮廓
- TCVN_CRM_EXTERNAL：仅外部轮廓（如同F_VN_FindContours [▶ 1179]）
- TCVN_CRM_CONNECTED_COMPONENTS：具有两层层次结构的轮廓
- TCVN_CRM_TREE：所有轮廓；与函数F_VN_FindContourHierarchyExp [▶ 1175]交互使用，以确定轮廓的层次结构
- TCVN_CRM_FLOODFILL：返回在泛洪填充算法帮助下发现的轮廓（仅适用于 DINT 图像）

近似法（专家）

ETcVnContourApproximationMethod [▶ 179]类型的近似法eApproximationMethod指定是否在所有周围点的基础上描述轮廓（TCVN_CAM_NONE），或者是否在近似法的帮助下减少描述点的数量。

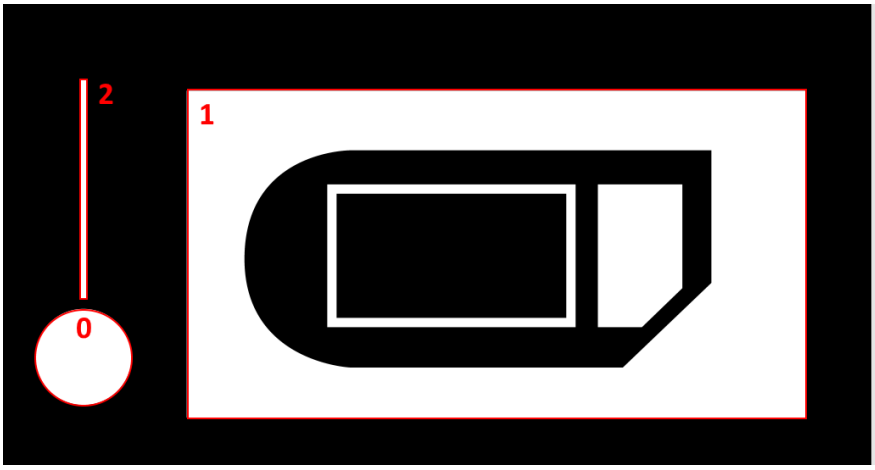
偏移量（专家）

偏移量aOffset定义了常量 X/Y 值，通过数值所有的轮廓点都被移位。当在一个 ROI 中寻找轮廓且它们的位置应该为原始图像提供参考时，这可能有所帮助。

轮廓层次的示例

通过 eRetrievalMode，不仅可以指定是寻找所有的轮廓还是只寻找外轮廓，还可以指定如何对轮廓进行排序以及它们之间的层次关系。这可以从参数ipHierarchy读取：

- **TCVN_CRM_EXTERNAL**
  仅外部轮廓以排序列表形式返回。**子**[2]和**父**[3]关系不确定，且处于无效状态-1。



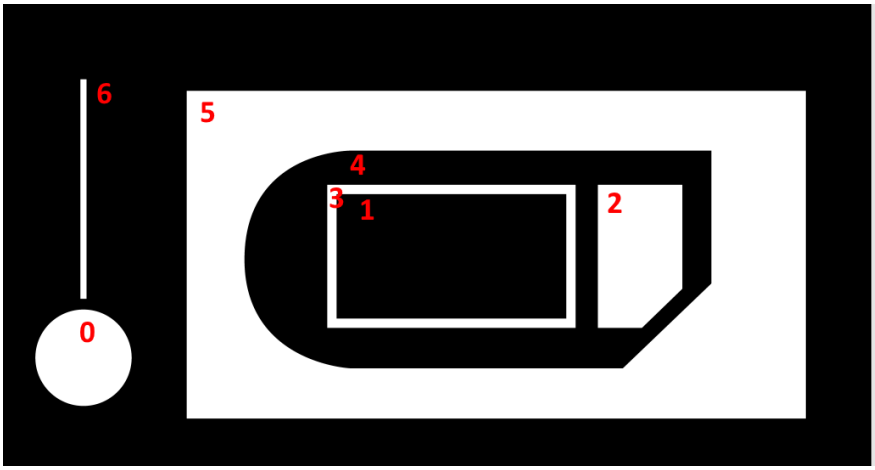
--- Hierarchy ---

[I] [N, P, C, P]

[0] [1, -1, -1, -1]
[1] [2, 0, -1, -1]
[2] [-1, 1, -1, -1]

- **TCVN_CRM_LIST**

  所有轮廓都以排序列表形式返回。**子**[2]和**父**[3]关系不确定，且处于无效状态-1。



--- Hierarchy ---

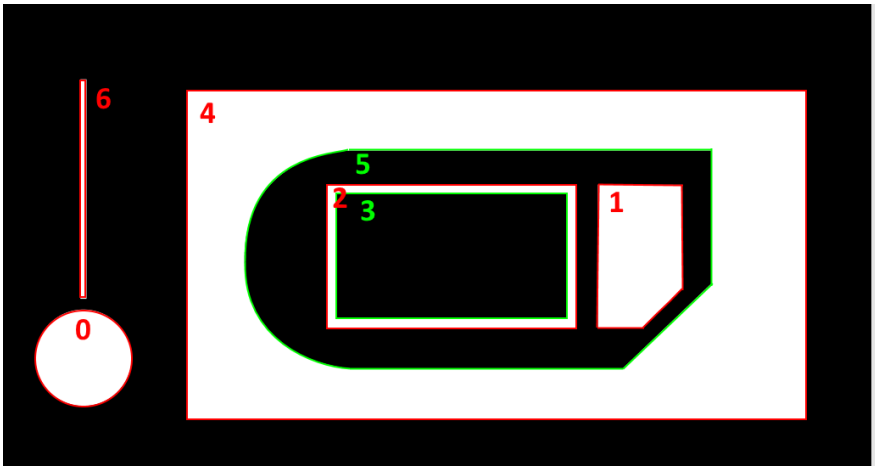
[I] [N, P, C, P]

[0] [1, -1, -1, -1]
[1] [2, 0, -1, -1]
[2] [3, 1, -1, -1]
[3] [4, 2, -1, -1]
[4] [5, 3, -1, -1]
[5] [6, 4, -1, -1]
[6] [-1, 4, -1, -1]

- **TCVN_CRM_CONNECTED_COMPONENTS**

  所有轮廓都是按照 2 个步骤的层次结构进行排序。首先考虑外部轮廓，然后在第二步考虑直接位于其内的轮廓。如果在第二步内存在其他轮廓，它们会像外轮廓一样被设置在第一步中

  在这个样本中，这意味着内部的**子**[2]轮廓（图像中的绿色）不能同时是**父**[3]轮廓（图像中的红色）。因此，将位于第 3 级树状结构中的轮廓再次被视为外部轮廓，并设置为第 1 级（**父**[3] = -1），在这种情况下，轮廓 1 和 2。



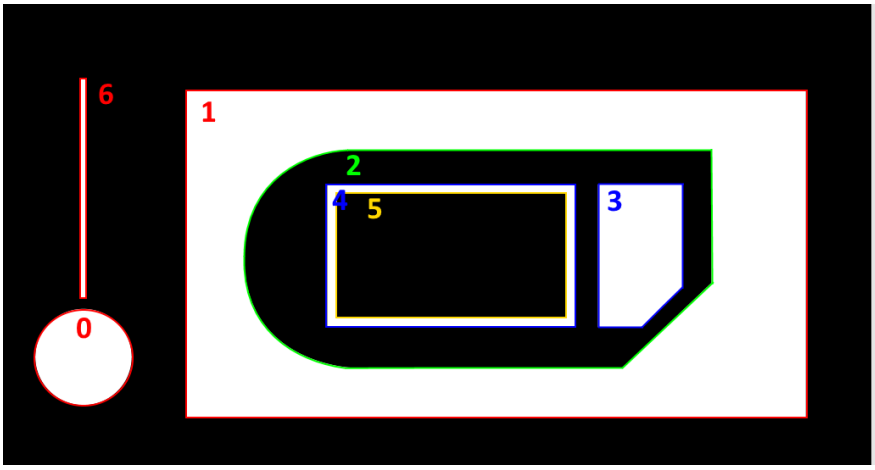
--- Hierarchy ---

[I] [N, P, C, P]

[0] [1, -1, -1, -1]
[1] [2, 0, -1, -1]
[2] [4, 1, 3, -1]
[3] [-1, -1, -1, 2]
[4] [6, 2, 5, -1]
[5] [-1, -1, -1, 4]
[6] [-1, 4, -1, -1]

- **TCVN_CRM_TREE**

  所有轮廓均按照树状结构设置。轮廓有可能既是**子**[2]，又是**父**[3]，具体取决于它们如何相互嵌套。



--- Hierarchy ---

[I] [N, P, C, P]

[0] [1, -1, -1, -1]
[1] [6, 0, 2, -1]
[2] [-1, -1, 3, 1]
[3] [4, -1, -1, 2]
[4] [-1, 3, 5, 2]
[5] [-1, -1, -1, 4]
[6] [-1, 1, -1, -1]

样本

- 查找轮廓 - 层次结构和检索模式 [▶ 2765]用于参数ETcVnContourRetrievalMode [▶ 181]
- 查找轮廓 - 近似法 [▶ 2763]用于参数ETcVnContourApproximationMethod [▶ 179]
- 查找轮廓而不是斑点检测 [▶ 2761]用于与函数F_VN_DetectBlobs [▶ 1171]进行比较

相关函数

- F_VN_FindContours(Exp) [▶ 1179]用于寻找一般的轮廓
- F_VN_FindContourHierarchyExp [▶ 1175]返回层次结构
- F_VN_DetectBlobs [▶ 1171]对轮廓进行综合过滤

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.5 F_VN_FindContours



Search for object contours in a binary image. Returns only external contours.
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_FindContours : HRESULT
VAR_INPUT
    ipSrcImage : ITcVnImage;
    ipContours : Reference To ITcVnContainer;
    hrPrev     : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipSrcImage	ITcVnImage [▶ 407]	Source image (USINT, 1 channel, binary)
ipContours	Reference To ITcVnContainer [▶ 365]	Returns a container which is filled with the found contours (ContainerType_Vector_Vector_TcVnPoint2_DINT; The elements of this container are single contours. Non-zero interface pointers are reused.)
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_FindContours检测输入图像中的外轮廓。

参数

输入图像

输入图像ipSrcImage必须正好有 1 个通道，而且应该是二进制图像。

如果输入图像不是二进制图像，它将通过 1 的阈值在内部转换成二进制图像。因此，输入的图像应该通过阈值或二进制分割进行手动预处理。

找到的轮廓列表 (返回值)

在输入图像中发现的所有轮廓都会在容器ipContours中返回。容器是一个二维的点阵列，因此，其类型为ContainerType_Vector_Vector_TcVnPoint2_DINT。

找到的轮廓会以ipContours返回。每个轮廓由一个点的总和组成，这些点被组合成一个容器 [▶ 146]。然后，另一个更高层次的容器将所有的轮廓合并。

专家参数

专家版F_VN_FindContoursExp [▶ 1181]包含额外的参数。例如，这也使得内部轮廓能够被发现。

应用

例如，寻找和处理轮廓如下所示：

预处理

由于要使用二进制图像作为输入图像ipSrcImage，因此在调用函数之前应将阈值或二进制分割应用于图像，例如使用函数F_VN_Threshold [▶ 1322]：

```
hr := F_VN_Threshold(ipImageIn, ipImageWork, 128, 255, TCVN_TT_BINARY, hr);
```

函数调用

关于函数调用，没有什么需要注意。参数设置仅在专家版F_VN_FindContoursExp [▶ 1181]下需要。

```
hr := F_VN_FindContours(
    ipSrcImage  :=  ipImageIn,
    ipContours  :=  ipContours,
    hrPrev      :=  hr
);
```

进一步处理

F_VN_GetForwardIterator [▶ 764]可用于将迭代器应用于上层容器，从而可以通过F_VN_GetContainer [▶ 762]一个接一个地检索各个轮廓的容器，以便使用轮廓分析函数来分析它们。记住通过F_VN_IncrementIterator [▶ 766]递增迭代器。

```
hr := F_VN_GetForwardIterator(ipContours, ipIterator, hr);
WHILE SUCCEEDED(hr) AND_THEN ipIterator.CheckIfEnd() <> S_OK DO
    hr := F_VN_GetContainer( ipIterator, ipContour, hr);
    hr := F_VN_IncrementIterator(ipIterator, hr);

    // Analyse the single contour in ipContours with the contour analysis functions

END_WHILE
```

另外，也可以用F_VN_GetAt_ITcVnContainer [▶ 644]从所有轮廓的容器中检索出单个轮廓。

找到的轮廓数量可以通过F_VN_GetNumberOfElements [▶ 764]查询。

可视化

使用函数F_VN_DrawContours(Exp) [▶ 1012]绘制所发现的轮廓。

```
hr := F_VN_DrawContours(
    ipContours      :=  ipContours,
    nContourIndex   :=  -1,
    ipDestImage     :=  ipImageRes,
    aColor          :=  aColorRed,
    nThickness      :=  5,
    hrPrev          :=  hr
);
```

轮廓列表`ipContours`中的所有轮廓都是通过设置`nContoursIndex`到`-1`进行绘制。

样本

- 查找轮廓而不是斑点检测 [▶ 2761]

相关函数

- F_VN_FindContours(Exp) [▶ 1179]用于寻找一般的轮廓
- F_VN_FindContourHierarchyExp [▶ 1175]返回层次结构
- F_VN_DetectBlobs [▶ 1171]对轮廓进行综合过滤

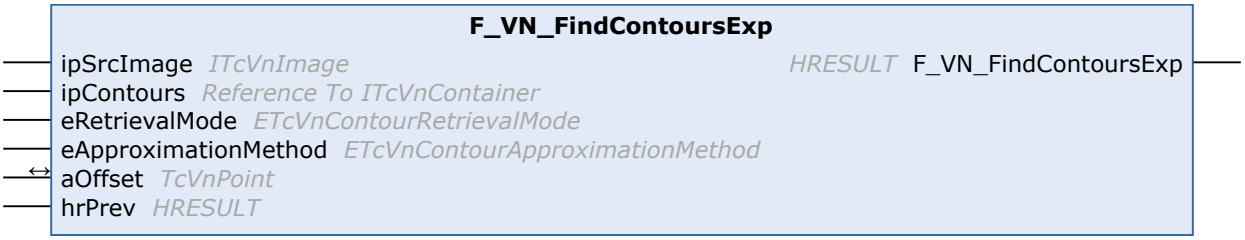
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.6 F_VN_FindContoursExp



Search for object contours in a binary image. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_FindContoursExp : HRESULT
VAR_INPUT
    ipSrcImage           : ITcVnImage;
    ipContours           : Reference To ITcVnContainer;
    eRetrievalMode       : ETcVnContourRetrievalMode;
    eApproximationMethod : ETcVnContourApproximationMethod;
END_VAR
VAR_IN_OUT
    aOffset              : TcVnPoint;
END_VAR
VAR_INPUT
    hrPrev               : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipSrcImage	ITcVnImage [▶ 407]	Source image (1 channel, binary, DINT if eRetrievalMode = CRM_FLOODFILL, otherwise USINT)
ipContours	Reference To ITcVnContainer [▶ 365]	Returns a container which is filled with the found contours (ContainerType_Vector_Vector_TcVnPoint2_DINT; The elements of this container are single contours. Non-zero interface pointers are reused.)
eRetrievalMode	ETcVnContourRetrievalMode [▶ 181]	Specifies which contours are retrieved and how their relationship is encoded
eApproximationMethod	ETcVnContourApproximationMethod [▶ 179]	Contour encoding
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

输入/输出

名称	类型	描述
aOffset	TcVnPoint [▶ 152]	每个轮廓点移动的偏移量

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_FindContoursExp是F_VN_FindContours [▶ 1179]的专家版本。它包含额外的参数。

参数

输入图像

输入图像ipSrcImage必须正好有 1 个通道，而且应该是二进制图像。

如果输入图像不是二进制图像，它将通过 1 的阈值在内部转换成二进制图像。因此，输入的图像应该通过阈值或二进制分割进行手动预处理。

找到的轮廓列表 (返回值)

在输入图像中发现的所有轮廓都会在容器ipContours中返回。容器是一个二维的点阵列，因此，其类型为ContainerType_Vector_Vector_TcVnPoint2_DINT。

找到的轮廓会以ipContours返回。每个轮廓由一个点的总和组成，这些点被组合成一个容器 [▶ 146]。然后，另一个更高层次的容器将所有的轮廓合并。

搜索模式（专家）

ETcVnContourRetrievalMode [▶ 181]类型的搜索模式eRetrievalMode定义了要寻找哪些轮廓：

- TCVN_CRM_LIST：所有轮廓
- TCVN_CRM_EXTERNAL：仅外部轮廓（如同F_VN_FindContours [▶ 1179]）
- TCVN_CRM_CONNECTED_COMPONENTS：具有两层层次结构的轮廓
- TCVN_CRM_TREE：所有轮廓；与函数F_VN_FindContourHierarchyExp [▶ 1175]交互使用，以确定轮廓的层次结构
- TCVN_CRM_FLOODFILL：返回在泛洪填充算法帮助下发现的轮廓（仅适用于 DINT 图像）

近似法（专家）

ETcVnContourApproximationMethod [▶ 179]类型的近似法eApproximationMethod指定是否在所有周围点的基础上描述轮廓（TCVN_CAM_NONE），或者是否在近似法的帮助下减少描述点的数量。

偏移量（专家）

偏移量aOffset定义了常量 X/Y 值，通过数值所有的轮廓点都被移位。当在一个 ROI 中寻找轮廓且它们的位置应该为原始图像提供参考时，这可能有所帮助。

应用

例如，查找输入图像中的所有轮廓如下所示，其中的轮廓点不是近似的，而是在 x 方向上移动了 100px，在 y 方向上移动了 -50px：

```
VAR
    aOffset :   TcVnPoint   :=  [100, -50];
END_VAR

hr := F_VN_FindContours(
    ipSrcImage              :=  ipImageIn,
    ipContours              :=  ipContours,
  eRetrievalMode      := TCVN_CRM_LIST,
  eApproximationMethod  := TCVN_CAM_NONE,
  aOffset         := aOffset,
    hrPrev                  :=  hr
);
```

样本

- 查找轮廓 - 层次结构和检索模式 [▶ 2765]用于参数ETcVnContourRetrievalMode [▶ 181]
- 查找轮廓 - 近似法 [▶ 2763]用于参数ETcVnContourApproximationMethod [▶ 179]
- 查找轮廓而不是斑点检测 [▶ 2761]用于与函数F_VN_DetectBlobs [▶ 1171]进行比较

相关函数

- F_VN_FindContours(Exp) [▶ 1179]用于寻找一般的轮廓
- F_VN_FindContourHierarchyExp [▶ 1175]返回层次结构
- F_VN_DetectBlobs [▶ 1171]对轮廓进行综合过滤

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.7 F_VN_FindTemplateWalsh



Match a template image with every location in the source image (using the Walsh transform) and evaluate the comparison results. Returns a sorted list of possible matches (best match first).
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_FindTemplateWalsh : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipTemplateImage : ITcVnImage;
    ipMatches       : Reference To ITcVnContainer;
    fMatchThreshold : REAL;
    nProjections    : UDINT;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 或 SINT，1 或 3 通道）
ipTemplateImage	ITcVnImage [▶ 407]	模板图像（与 ipSrcImage 类型相同，宽度和高度较小）
ipMatches	引用 ITcVnContainer [▶ 365]	返回匹配位置（ContainerType_Vector_TcVnVector4_REAL，其中每个元素包含中心点 (x, y)、以度为单位的旋转角以及 ipTemplateImage 在 ipSrcImage 中的匹配值；容器元素按相关性排序（最佳匹配项优先））
fMatchThreshold	REAL	用于区分相关匹配和不相关匹配的阈值（0...1，1.0 表示完全匹配。）
nProjections	UDINT	要使用的Walsh投影数量。nProjections 值越高，匹配时考虑的细节就越多。由于最相关的信息由前几个Walsh投影所捕获，因此较少的数量（例如 5、10、25 或 50）即可满足许多使用场景。每个投影都需要额外的计算时间。因此，只要满足匹配要求，选择较少的Walsh投影数量是有利的。允许的最大 nProjections 值由模板图像的宽度和高度决定。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

`F_VN_FindTemplateWalsh` 函数使用Walsh变换来定位参考图像（模板）在搜索图像中的所有实例。

参数

输入图像

输入图像 ipSrcImage 必须是 USINT 或 SINT 类型的单通道或三通道图像。对于三通道图像，匹配将在所有三个通道中联合执行，这也会导致处理时间增加。

模板图像

模板图像 ipTemplateImage 必须与 ipSrcImage 具有相同的类型，并且尺寸（高度和宽度）小于输入图像。

找到的匹配位置的列表

在输入图像中找到的所有匹配项按其匹配值排序（最佳匹配项优先），并在类型为 ContainerType_Vector_TcVnVector4_REAL 的容器 ipMatches 中返回。每个 TcVnVector4_REAL 均包含中心坐标、旋转角度（以度为单位）以及匹配值 (fX, fY, fAngleDeg, fMatchValue).

匹配阈值

阈值 fMatchThreshold （0 至 1.0）可用于区分不相关匹配和相关匹配，其中 1.0 表示完全匹配。该值取决于输入图像的性质以及所选参数，必须进行相应调整。

Walsh投影数量

nProjections 参数指定了用于匹配的WAlsh投影的数量。较低的投影数量主要捕获粗略的图像结构，而较高的投影数量则会逐步捕获更精细的细节，但会增加处理时间。因此，nProjections 应设置得尽可能低，但又要高到足以满足需求。在许多应用中，5-50 个投影已足够。nProjections 的允许上限受模板图像的宽度和高度限制。

示例

- 带旋转的Walsh模板匹配 [▶ 2771]

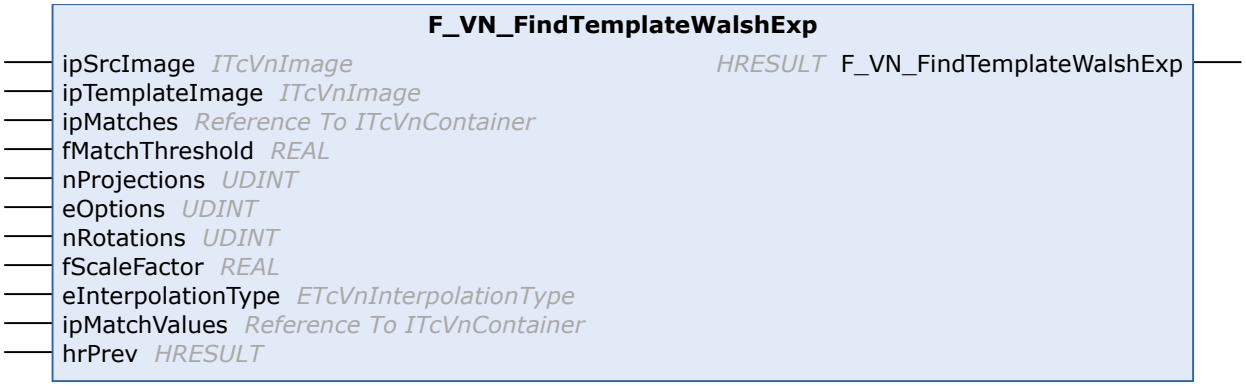
相关函数

- F_VN_FindTemplateWalsh(Exp) [▶ 1184]：在输入图像中搜索模板，并返回找到的匹配项的排序列表。
- F_VN_FindTemplateWalsh(Exp)_AngleRange [▶ 1188]：与 F_VN_FindTemplateWalsh(Exp) 类似，但具有可自由定义的角度范围和任意数量的角度步长，以便有针对性地优化运行时和精度。
- F_VN_MatchTemplateWalsh(Exp) [▶ 1205]：将模板与输入图像中的每个位置进行比较，并返回原始结果图像，以进行特定应用的后处理。

所需的许可证

TC3 Vision Matching

6.1.4.15.1.8 F_VN_FindTemplateWalshExp



Match a template image with every location in the source image (using the Walsh transform) and evaluate the comparison results. Returns a sorted list of possible matches (best match first). (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_FindTemplateWalshExp : HRESULT
VAR_INPUT
    ipSrcImage          : ITcVnImage;
    ipTemplateImage     : ITcVnImage;
    ipMatches           : Reference To ITcVnContainer;
    fMatchThreshold     : REAL;
    nProjections        : UDINT;
    eOptions            : UDINT;
    nRotations          : UDINT;
    fScaleFactor        : REAL;
    eInterpolationType  : ETcVnInterpolationType;
    ipMatchValues       : Reference To ITcVnContainer;
    hrPrev              : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 或 SINT，1 或 3 通道）
ipTemplateImage	ITcVnImage [▶ 407]	模板图像（与 ipSrcImage 类型相同，宽度和高度较小）
ipMatches	引用 ITcVnContainer [▶ 365]	返回匹配位置（容器类型取决于 eOptions；ContainerType_Vector_TcVnRotatedRectangle 或 ContainerType_Vector_TcVnVector4_REAL，其中每个元素包含中心点 (x, y)、以度为单位的旋转角以及 ipTemplateImage 在 ipSrcImage 中的匹配值；容器元素按相关性排序（最佳匹配项优先））
fMatchThreshold	REAL	用于区分相关匹配和不相关匹配的阈值（0...1，1.0 表示完全匹配。）
nProjections	UDINT	要使用的Walsh投影数量。nProjections 值越高，匹配时考虑的细节就越多。由于最相关的信息由前几个Walsh投影所捕获，因此较少的数量（例如 5、10、25 或 50）即可满足许多使用场景。每个投影都需要额外的计算时间。因此，只要满足匹配要求，选择较少的Walsh投影数量是有利的。允许的最大 nProjections 值由模板图像的宽度和高度决定。
eOptions	UDINT	控制匹配过程的标志。（参见 ETcVnFindTemplateWalshOptions；默认值：TCVN_FTWO_NONE）
nRotations	UDINT	要检查的旋转数量。（有效值：1（0°）、2（0° 和 180°）、4（0°、90°、180° 和 270°）、8（0°、45°、……、315°）、16（0°、22.5°、……、337.5°）和 32（0°、11.25°、……、348.75°）；默认值：1）
fScaleFactor	REAL	系数（0..1]，用于减小源图像和模板图像的宽度和高度，以提高性能（但精度降低！）
eInterpolationType	ETcVnInterpolationType [▶ 198]	图像大小调整插值类型（仅在 fScaleFactor != 1 时使用，建议大多数情况下使用 TCVN_IT_BILINEAR）
ipMatchValues	引用 ITcVnContainer [▶ 365]	可选择返回匹配值（ContainerType_Vector_REAL，大小和排序与 ipMatches 相同。如果不需要，则设为 0。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

F_VN_FindTemplateWalshExp 函数是 F_VN_FindTemplateWalsh [▶ 1184] 的专家变体，包含附加参数。它使用Walsh变换来定位参考图像（模板）在搜索图像中的所有实例。通过固定的步长可以检查所找到对象的方向，该步长由 0° 到 360° 范围内要检查的旋转数量决定。

参数

输入图像

输入图像 ipSrcImage 必须是 USINT 或 SINT 类型的单通道或三通道图像。对于三通道图像，匹配将在所有三个通道中联合执行，这也会导致处理时间增加。

模板图像

模板图像 ipTemplateImage 必须与 ipSrcImage 具有相同的类型，并且尺寸（高度和宽度）小于输入图像。

找到的匹配位置的列表

在输入图像中找到的所有匹配项按其匹配值排序（最佳匹配项优先），并在类型为 ContainerType_Vector_TcVnVector4_REAL 的容器 ipMatches 中返回。每个 TcVnVector4_REAL 均包含中心坐标、旋转角度（以度为单位）以及匹配值 (fX, fY, fAngleDeg, fMatchValue).

匹配阈值

阈值 fMatchThreshold （0 至 1.0）可用于区分不相关匹配和相关匹配，其中 1.0 表示完全匹配。该值取决于输入图像的性质以及所选参数，必须进行相应调整。

Walsh投影数量

nProjections 参数指定了用于匹配的WAlsh投影的数量。较低的投影数量主要捕获粗略的图像结构，而较高的投影数量则会逐步捕获更精细的细节，但会增加处理时间。因此，nProjections 应设置得尽可能低，但又要高到足以满足需求。在许多应用中，5-50 个投影已足够。nProjections 的允许上限受模板图像的宽度和高度限制。

Options

通过 eOptions 可以指定影响功能和结果输出的选项（ETcVnFindTemplateWalshOptions [▶ 195]）。多个选项可以使用 OR 运算符进行组合。提供以下选项：

- TCVN_FTWO_NONE：无特殊选项。
- TCVN_FTWO_FUSE_MATCHES：合并位置和旋转相近的命中结果。建议设置此选项。停用功能可能有助于开发和参数设置，以便查看所有原始检测结果。
- TCVN_FTWO_ROTATED_RECTANGLE：直接将结果以 ContainerType_Vector_TcVnRotatedRectangle 形式返回。相应的匹配值可以使用 ipMatchValues 单独读取。
- TCVN_FTWO_SKIP_DC：在相似度计算中跳过 DC 部分，这可以提高对光照变化的适应性。通常需要更多的投影数量。
- TCVN_FTWO_WD_STOP_CORRECTION：如果处理过程因 Watchdog（看门狗）中止而中断，则会启发式地调整匹配阈值。

要检查的旋转数量

要检查的旋转步数 nNumRotations 决定了匹配过程中的角度分辨率。有效值为：

- 1（0°）
- 2（0° 和 180°）
- 4（0°、90°、180° 和 270°）
- 8（0°、45°、……、315°）
- 16（0°、22.5°、……、337.5°）

- 32（0°、11.25°、……、348.75°）

最终匹配结果的输出旋转角度也可能位于所定义的旋转步长之间。如果多个角度步长均找到了匹配结果，并且启用了 TCVN_FTWO_FUSE_MATCHES 选项，则由插值确定最终角度。

脉冲当量

系数 fScaleFactor（0 至 1.0）用于降低输入图像和模板图像的分辨率。缩小图像尺寸可以提高性能，但会降低匹配的准确性。

插值类型

eInterpolationType 参数确定了图像缩放的方法（仅在 fScaleFactor 不等于 1 时才会进行评估）。对于大多数应用，推荐使用 TCVN_IT_BILINEAR。

匹配值的返回

ipMatchValues 参数是一个可选的返回容器（ContainerType_Vector_REAL），用于存储纯匹配值。它的长度和排序与 ipMatches 相同。如果未使用 TCVN_FTWO_ROTATED_RECTANGLE 选项，则这些值是冗余的，因为它们已经包含在 ipMatches 的返回结构中。如果不需要单独输出，则应将该参数设置为 0。

示例

- 带旋转的Walsh模板匹配 [▶ 2771]

相关函数

- F_VN_FindTemplateWalsh(Exp) [▶ 1184]：在输入图像中搜索模板，并返回找到的匹配项的排序列表。
- F_VN_FindTemplateWalsh(Exp)_AngleRange [▶ 1188]：与 F_VN_FindTemplateWalsh(Exp) 类似，但具有可自由定义的角度范围和任意数量的角度步长，以便有针对性地优化运行时和精度。
- F_VN_MatchTemplateWalsh(Exp) [▶ 1205]：将模板与输入图像中的每个位置进行比较，并返回原始结果图像，以进行特定应用的后处理。

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.9 F_VN_FindTemplateWalsh_AngleRange



Match a template image with every location in the source image (using the Walsh transform) and evaluate the comparison results. Returns a sorted list of possible matches (best match first).
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_FindTemplateWalsh_AngleRange : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipTemplateImage : ITcVnImage;
    ipMatches       : Reference To ITcVnContainer;
    fMatchThreshold : REAL;
    nProjections    : UDINT;
    fStartAngle     : LREAL;
    fStopAngle      : LREAL;
    nAngles         : UDINT;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 或 SINT，1 或 3 通道）
ipTemplateImage	ITcVnImage [▶ 407]	模板图像（与 ipSrcImage 类型相同，宽度和高度较小）
ipMatches	引用 ITcVnContainer [▶ 365]	返回匹配位置（ContainerType_Vector_TcVnVector4_REAL，其中每个元素包含中心点 (x, y)、以度为单位的旋转角以及 ipTemplateImage 在 ipSrcImage 中的匹配值；容器元素按相关性排序（最佳匹配项优先））
fMatchThreshold	REAL	用于区分相关匹配和不相关匹配的阈值（0...1，1.0 表示完全匹配。）
nProjections	UDINT	要使用的Walsh投影数量。nProjections 值越高，匹配时考虑的细节就越多。由于最相关的信息由前几个Walsh投影所捕获，因此较少的数量（例如 5、10、25 或 50）即可满足许多使用场景。每个投影都需要额外的计算时间。因此，只要满足匹配要求，选择较少的Walsh投影数量是有利的。允许的最大 nProjections 值由模板图像的宽度和高度决定。
fStartAngle	LREAL	起始搜索角度
fStopAngle	LREAL	终止搜索角度
nAngles	UDINT	要使用的不同搜索角度的数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

`F_VN_FindTemplateWalsh_AngleRange` 函数使用Walsh变换来定位参考图像（模板）在搜索图像中的所有实例。它将对象旋转的搜索范围限制在一个可自由定义的角度区间内，并采用可变步长。这样可以有针对性地优化角度精度和运行时。如果所搜索对象的角度范围未知或无法限制，则应使用 F_VN_FindTemplateWalsh [▶ 1184]。

参数

输入图像

输入图像 `ipSrcImage` 必须是 `USINT` 或 `SINT` 类型的单通道或三通道图像。对于三通道图像，匹配将在所有三个通道中联合执行，这也会导致处理时间增加。

模板图像

模板图像 `ipTemplateImage` 必须与 `ipSrcImage` 具有相同的类型，并且尺寸（高度和宽度）小于输入图像。

找到的匹配位置的列表

在输入图像中找到的所有匹配项按其匹配值排序（最佳匹配项优先），并在类型为 `ContainerType_Vector_TcVnVector4_REAL` 的容器 `ipMatches` 中返回。每个 `TcVnVector4_REAL` 均包含中心坐标、旋转角度（以度为单位）以及匹配值 `(fX, fY, fAngleDeg, fMatchValue)`.

匹配阈值

阈值 `fMatchThreshold` （0 至 1.0）可用于区分不相关匹配和相关匹配，其中 1.0 表示完全匹配。该值取决于输入图像的性质以及所选参数，必须进行相应调整。

Walsh投影数量

`nProjections` 参数指定了用于匹配的WAlsh投影的数量。较低的投影数量主要捕获粗略的图像结构，而较高的投影数量则会逐步捕获更精细的细节，但会增加处理时间。因此，`nProjections` 应设置得尽可能低，但又要高到足以满足需求。在许多应用中，5-50 个投影已足够。`nProjections` 的允许上限受模板图像的宽度和高度限制。

角度范围

对象旋转的搜索范围由 `fStartAngle` 和 `fStopAngle` 定义，并以度为单位作为 `LREAL` 值。这两个参数指定了所检查角度范围的外部限值。如果搜索范围设置得过窄，则存在无法识别该范围之外实际存在的对象的风险。建议预留足够的公差缓冲。

角度步数

`nAngles` 参数定义了在指定角度范围内均匀分布的搜索角度的数量。与 F_VN_FindTemplateWalsh [▶ 1184] 相比，此处可以选择任意数量。这样可以在指定范围内实现更精细的角度分辨率。

示例：当 `fStartAngle` = -30°，`fStopAngle` = 30° 且 `nAngles` = 13 时，将检查角度 -30°、-25°、-20°、……、+25°、+30°。

示例

- 带旋转的Walsh模板匹配 [▶ 2771]

相关函数

- F_VN_FindTemplateWalsh(Exp) [▶ 1184]：在输入图像中搜索模板，并返回找到的匹配项的排序列表。
- F_VN_FindTemplateWalsh(Exp)_AngleRange [▶ 1188]：与 `F_VN_FindTemplateWalsh(Exp)` 类似，但具有可自由定义的角度范围和任意数量的角度步长，以便有针对性地优化运行时和精度。
- F_VN_MatchTemplateWalsh(Exp) [▶ 1205]：将模板与输入图像中的每个位置进行比较，并返回原始结果图像，以进行特定应用的后处理。

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.10 F_VN_FindTemplateWalshExp_AngleRange



Match a template image with every location in the source image (using the Walsh transform) and evaluate the comparison results. Returns a sorted list of possible matches (best match first). (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_FindTemplateWalshExp_AngleRange : HRESULT
VAR_INPUT
    ipSrcImage          : ITcVnImage;
    ipTemplateImage     : ITcVnImage;
    ipMatches           : Reference To ITcVnContainer;
    fMatchThreshold     : REAL;
    nProjections        : UDINT;
    fStartAngle         : LREAL;
    fStopAngle          : LREAL;
    nAngles             : UDINT;
    eOptions            : UDINT;
    fScaleFactor        : REAL;
    eInterpolationType  : ETcVnInterpolationType;
    ipMatchValues       : Reference To ITcVnContainer;
    hrPrev              : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 或 SINT，1 或 3 通道）
ipTemplateImage	ITcVnImage [▶ 407]	模板图像（与 ipSrcImage 类型相同，宽度和高度较小）
ipMatches	引用 ITcVnContainer [▶ 365]	返回匹配位置（容器类型取决于 eOptions；ContainerType_Vector_TcVnRotatedRectangle 或 ContainerType_Vector_TcVnVector4_REAL，其中每个元素包含中心点 (x, y)、以度为单位的旋转角以及 ipTemplateImage 在 ipSrcImage 中的匹配值；容器元素按相关性排序（最佳匹配项优先））
fMatchThreshold	REAL	用于区分相关匹配和不相关匹配的阈值（0...1，1.0 表示完全匹配。）
nProjections	UDINT	要使用的Walsh投影数量。nProjections 值越高，匹配时考虑的细节就越多。由于最相关的信息由前几个Walsh投影所捕获，因此较少的数量（例如 5、10、25 或 50）即可满足许多使用场景。每个投影都需要额外的计算时间。因此，只要满足匹配要求，选择较少的Walsh投影数量是有利的。允许的最大 nProjections 值由模板图像的宽度和高度决定。
fStartAngle	LREAL	起始搜索角度
fStopAngle	LREAL	终止搜索角度

名称	类型	描述
nAngles	UDINT	要使用的不同搜索角度的数量
eOptions	UDINT	控制匹配过程的标志。（参见 ETcVnFindTemplateWalshOptions；默认值：TCVN_FTWO_NONE）
fScaleFactor	REAL	系数（0..1]，用于减小源图像和模板图像的宽度和高度，以提高性能（但精度降低！）
eInterpolationType	ETcVnInterpolationType [▶ 198]	图像大小调整插值类型（仅在 fScaleFactor != 1 时使用，建议大多数情况下使用 TCVN_IT_BILINEAR）
ipMatchValues	引用 ITcVnContainer [▶ 365]	可选择返回匹配值（ContainerType_Vector_REAL，大小和排序与 ipMatches 相同。如果不需要，则设为 0。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_FindTemplateWalshExp_AngleRange` 是 F_VN_FindTemplateWalsh_AngleRange [▶ 1188] 的专家变体，包含附加参数。它使用Walsh变换来定位参考图像（模板）在搜索图像中的所有实例。它将对象旋转的搜索范围限制在一个可自由定义的角度区间内，并采用可变步长。这样可以有针对性地优化角度精度和运行时。如果所搜索对象的角度范围未知或无法限制，则应使用 F_VN_FindTemplateWalsh [▶ 1184]。

参数

输入图像

输入图像 `ipSrcImage` 必须是 `USINT` 或 `SINT` 类型的单通道或三通道图像。对于三通道图像，匹配将在所有三个通道中联合执行，这也会导致处理时间增加。

模板图像

模板图像 `ipTemplateImage` 必须与 `ipSrcImage` 具有相同的类型，并且尺寸（高度和宽度）小于输入图像。

找到的匹配位置的列表

在输入图像中找到的所有匹配项按其匹配值排序（最佳匹配项优先），并在类型为 `ContainerType_Vector_TcVnVector4_REAL` 的容器 `ipMatches` 中返回。每个 `TcVnVector4_REAL` 均包含中心坐标、旋转角度（以度为单位）以及匹配值 `(fX, fY, fAngleDeg, fMatchValue)`.

匹配阈值

阈值 `fMatchThreshold` （0 至 1.0）可用于区分不相关匹配和相关匹配，其中 1.0 表示完全匹配。该值取决于输入图像的性质以及所选参数，必须进行相应调整。

Walsh投影数量

`nProjections` 参数指定了用于匹配的WAlsh投影的数量。较低的投影数量主要捕获粗略的图像结构，而较高的投影数量则会逐步捕获更精细的细节，但会增加处理时间。因此，`nProjections` 应设置得尽可能低，但又要高到足以满足需求。在许多应用中，5-50 个投影已足够。`nProjections` 的允许上限受模板图像的宽度和高度限制。

角度范围

对象旋转的搜索范围由 `fStartAngle` 和 `fStopAngle` 定义，并以度为单位作为 `LREAL` 值。这两个参数指定了所检查角度范围的外部限值。如果搜索范围设置得过窄，则存在无法识别该范围之外实际存在的对象的风险。建议预留足够的公差缓冲。

角度步数

nAngles 参数定义了在指定角度范围内均匀分布的搜索角度的数量。与 F_VN_FindTemplateWalsh [▶ 1184] 相比，此处可以选择任意数量。这样可以在指定范围内实现更精细的角度分辨率。

示例：当 fStartAngle = -30°，fStopAngle = 30° 且 nAngles = 13 时，将检查角度 -30°、-25°、-20°、……、+25°、+30°。

Options

通过 eOptions 可以指定影响功能和结果输出的选项（ETcVnFindTemplateWalshOptions [▶ 195]）。多个选项可以使用 OR 运算符进行组合。提供以下选项：

- TCVN_FTWO_NONE：无特殊选项。
- TCVN_FTWO_FUSE_MATCHES：合并位置和旋转相近的命中结果。建议设置此选项。停用功能可能有助于开发和参数设置，以便查看所有原始检测结果。
- TCVN_FTWO_ROTATED_RECTANGLE：直接将结果以 ContainerType_Vector_TcVnRotatedRectangle 形式返回。相应的匹配值可以使用 ipMatchValues 单独读取。
- TCVN_FTWO_SKIP_DC：在相似度计算中跳过 DC 部分，这可以提高对光照变化的适应性。通常需要更多的投影数量。
- TCVN_FTWO_WD_STOP_CORRECTION：如果处理过程因 Watchdog（看门狗）中止而中断，则会启发式地调整匹配阈值。
- TCVN_FTWO_MATCH_OPPOSITE_RANGE：此外，还会使用相反的角度（旋转 180°）执行匹配。当不确定对象是否可能旋转 180° 时，此选项很有用。它可以有效地将所检查的角度范围扩大一倍，而无需搜索整个 360° 范围。

脉冲当量

系数 fScaleFactor（0 至 1.0）用于降低输入图像和模板图像的分辨率。缩小图像尺寸可以提高性能，但会降低匹配的准确性。

插值类型

eInterpolationType 参数确定了图像缩放的方法（仅在 fScaleFactor 不等于 1 时才会进行评估）。对于大多数应用，推荐使用 TCVN_IT_BILINEAR。

匹配值的返回

ipMatchValues 参数是一个可选的返回容器（ContainerType_Vector_REAL），用于存储纯匹配值。它的长度和排序与 ipMatches 相同。如果未使用 TCVN_FTWO_ROTATED_RECTANGLE 选项，则这些值是冗余的，因为它们已经包含在 ipMatches 的返回结构中。如果不需要单独输出，则应将该参数设置为 0。

示例

- 带旋转的Walsh模板匹配 [▶ 2771]

相关函数

- F_VN_FindTemplateWalsh(Exp) [▶ 1184]：在输入图像中搜索模板，并返回找到的匹配项的排序列表。
- F_VN_FindTemplateWalsh(Exp)_AngleRange [▶ 1188]：与 F_VN_FindTemplateWalsh(Exp) 类似，但具有可自由定义的角度范围和任意数量的角度步长，以便有针对性地优化运行时和精度。
- F_VN_MatchTemplateWalsh(Exp) [▶ 1205]：将模板与输入图像中的每个位置进行比较，并返回原始结果图像，以进行特定应用的后处理。

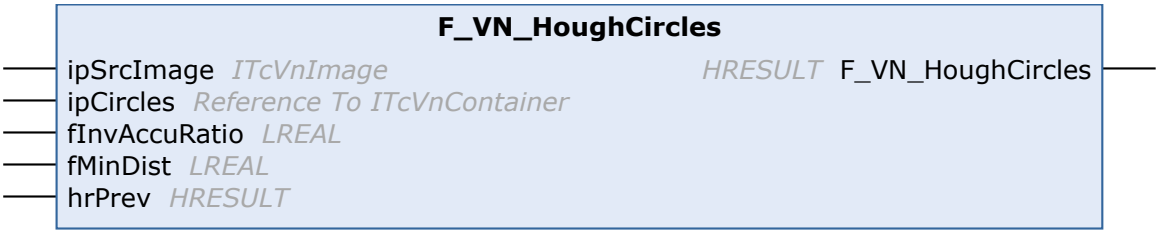
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.11 F_VN_HoughCircles



Search for circles using the Hough transform.
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_HoughCircles : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipCircles     : Reference To ITcVnContainer;
    fInvAccuRatio : LREAL;
    fMinDist      : LREAL;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道，灰度等级）
ipCircles	引用 ITcVnContainer [▶ 365]	返回一个以找到的圆填充的容器（ContainerType_Vector_TcVnVector3_REAL；每个容器元素分别包含圆心的 x 坐标 [0]、圆心的 y 坐标 [1] 和半径 [2]。重复使用非零接口指针。）
fInvAccuRatio	LREAL	累加器大小与源图像大小的反比（必须大于 0。值为 2 表示大小双向减半）。
fMinDist	LREAL	两个圆的最小允许距离（必须大于 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

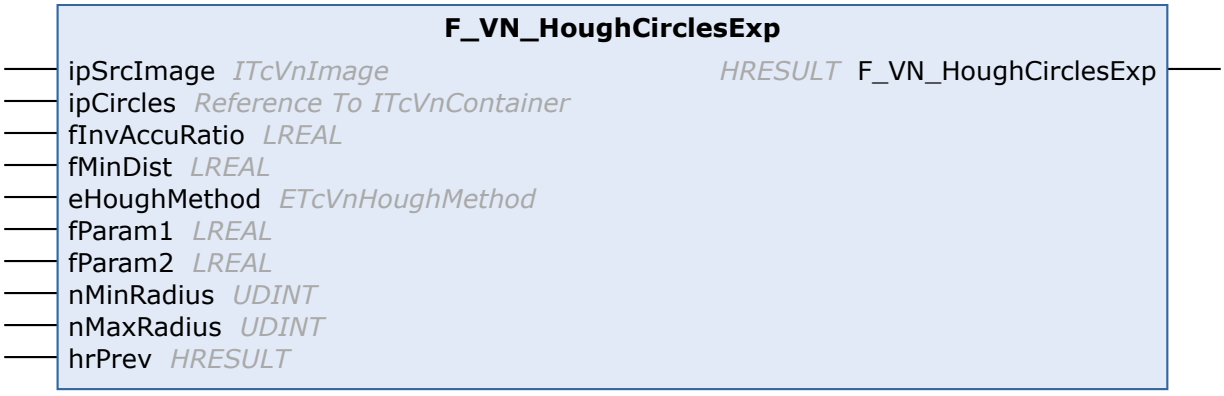
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.12 F_VN_HoughCirclesExp



Search for circles using the Hough transform. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_HoughCirclesExp : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipCircles     : Reference To ITcVnContainer;
    fInvAccuRatio : LREAL;
    fMinDist      : LREAL;
    eHoughMethod  : ETcVnHoughMethod;
    fParam1       : LREAL;
    fParam2       : LREAL;
    nMinRadius    : UDINT;
    nMaxRadius    : UDINT;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道，灰度等级）
ipCircles	引用 ITcVnContainer [▶ 365]	返回一个以找到的圆填充的容器（ContainerType_Vector_TcVnVector3_REAL；每个容器元素分别包含圆心的 x 坐标 [0]、圆心的 y 坐标 [1] 和半径 [2]。重复使用非零接口指针。）
fInvAccuRatio	LREAL	累加器大小与源图像大小的反比（必须大于 0。值为 2 表示大小双向减半）。
fMinDist	LREAL	两个圆的最小允许距离（必须大于 0）
eHoughMethod	ETcVnHoughMethod [▶ 197]	要使用的霍夫（Hough）变换方法（GRADIENT 或 GRADIENT_ALT）
fParam1	LREAL	第一个方法特定参数（GRADIENT、GRADIENT_ALT：Canny 边缘检测的阈值上限，必须大于 0）
fParam2	LREAL	第二个方法特定参数（GRADIENT：用于检测圆心的累加器阈值，必须大于 0。GRADIENT_ALT：所需的圆完美度，必须大于 0 且小于 1，1 表示一个完美圆）
nMinRadius	UDINT	允许的最小圆半径
nMaxRadius	UDINT	允许的最大圆半径（如果为 0，则该值被内设为最大值（图像行，图像列））
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

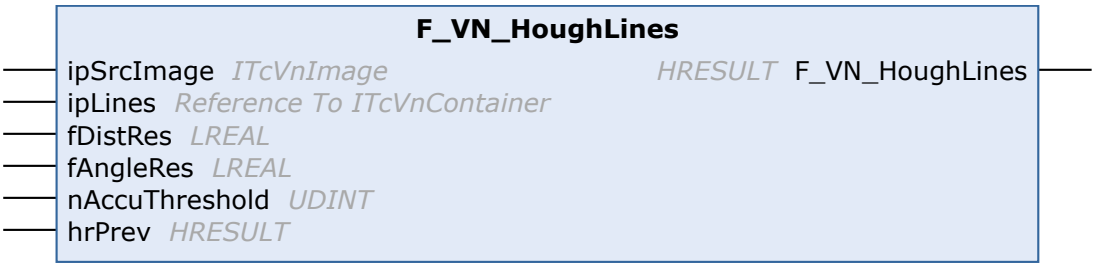
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.13 F_VN_HoughLines



Search for lines using the standard Hough transform.

语法

定义：

```
FUNCTION F_VN_HoughLines : HRESULT
VAR_INPUT
    ipSrcImage     : ITcVnImage;
    ipLines        : Reference To ITcVnContainer;
    fDistRes       : LREAL;
    fAngleRes      : LREAL;
    nAccuThreshold : UDINT;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道，二值，可由函数修改）
ipLines	引用 ITcVnContainer [▶ 365]	返回一个以找到的线条填充的容器（ContainerType_Vector_TcVnVector2_REAL；每个容器元素分别包含与原点的距离 [0]（以像素为单位）和旋转角 [1]（以弧度为单位）。重复使用非零接口指针。）
fDistRes	LREAL	累加器的距离分辨率（以像素为单位，必须大于 0）
fAngleRes	LREAL	累加器的角度分辨率（以弧度为单位，必须大于 0）
nAccuThreshold	UDINT	累加器阈值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

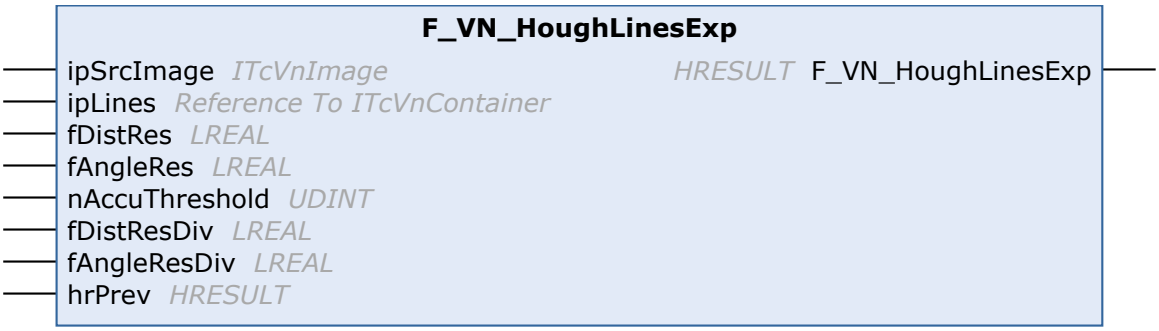
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.14 F_VN_HoughLinesExp



Search for lines using the standard Hough transform. (expert function)

语法

定义：

```
FUNCTION F_VN_HoughLinesExp : HRESULT
VAR_INPUT
    ipSrcImage     : ITcVnImage;
    ipLines        : Reference To ITcVnContainer;
    fDistRes       : LREAL;
    fAngleRes      : LREAL;
    nAccuThreshold : UDINT;
    fDistResDiv    : LREAL;
    fAngleResDiv   : LREAL;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道，二值，可由函数修改）
ipLines	引用 ITcVnContainer [▶ 365]	返回一个以找到的线条填充的容器（ContainerType_Vector_TcVnVector2_REAL；每个容器元素分别包含与原点的距离 [0]（以像素为单位）和旋转角 [1]（以弧度为单位）。重复使用非零接口指针。）
fDistRes	LREAL	累加器的距离分辨率（以像素为单位，必须大于 0）
fAngleRes	LREAL	累加器的角度分辨率（以弧度为单位，必须大于 0）
nAccuThreshold	UDINT	累加器阈值
fDistResDiv	LREAL	多尺度霍夫（Hough）变换的距离分辨率除数（>= 0，如果 fAngleResDiv > 0，则必须 > 0）
fAngleResDiv	LREAL	多尺度霍夫（Hough）变换的角度分辨率除数（>= 0，如果 fDistResDiv > 0，则必须 > 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

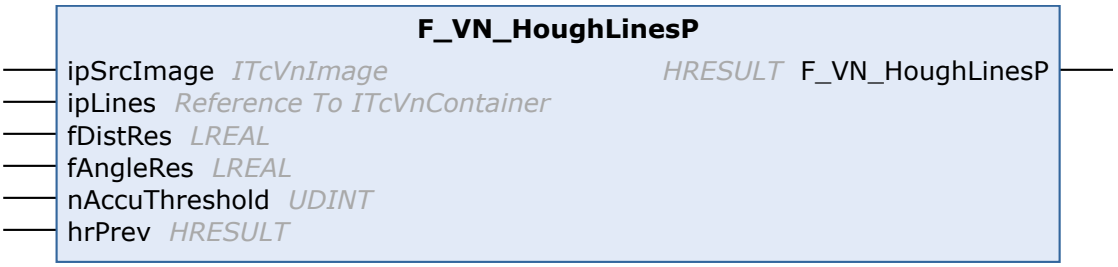
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.15 F_VN_HoughLinesP



Search for line segments using the probabilistic Hough transform.

语法

定义：

```
FUNCTION F_VN_HoughLinesP : HRESULT
VAR_INPUT
    ipSrcImage     : ITcVnImage;
    ipLines        : Reference To ITcVnContainer;
    fDistRes       : LREAL;
    fAngleRes      : LREAL;
    nAccuThreshold : UDINT;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道，二值，可由函数修改）
ipLines	引用 ITcVnContainer [▶ 365]	返回一个以找到的线段填充的容器（ContainerType_Vector_TcVnVector4_DINT；每个容器元素包含两个终点。重复使用非零接口指针。）
fDistRes	LREAL	累加器的距离分辨率（以像素为单位，必须大于 0）
fAngleRes	LREAL	累加器的角度分辨率（以弧度为单位，必须大于 0）
nAccuThreshold	UDINT	累加器阈值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

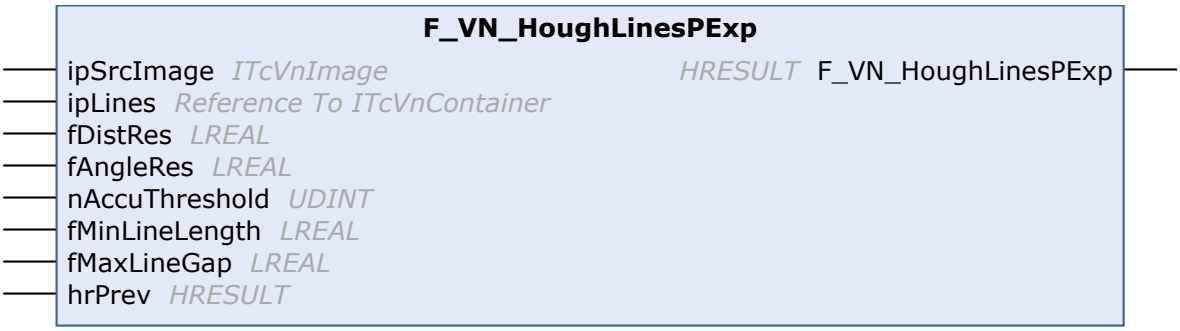
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.16 F_VN_HoughLinesPExp



Search for line segments using the probabilistic Hough transform. (expert function)

语法

定义：

```
FUNCTION F_VN_HoughLinesPExp : HRESULT
VAR_INPUT
    ipSrcImage     : ITcVnImage;
    ipLines        : Reference To ITcVnContainer;
    fDistRes       : LREAL;
    fAngleRes      : LREAL;
    nAccuThreshold : UDINT;
    fMinLineLength : LREAL;
    fMaxLineGap    : LREAL;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道，二值，可由函数修改）
ipLines	引用 ITcVnContainer [▶ 365]	返回一个以找到的线段填充的容器（ContainerType_Vector_TcVnVector4_DINT；每个容器元素包含两个终点。重复使用非零接口指针。）
fDistRes	LREAL	累加器的距离分辨率（以像素为单位，必须大于 0）
fAngleRes	LREAL	累加器的角度分辨率（以弧度为单位，必须大于 0）
nAccuThreshold	UDINT	累加器阈值
fMinLineLength	LREAL	搜索的最小直线长度
fMaxLineGap	LREAL	同一直线上各点之间的最大间距
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

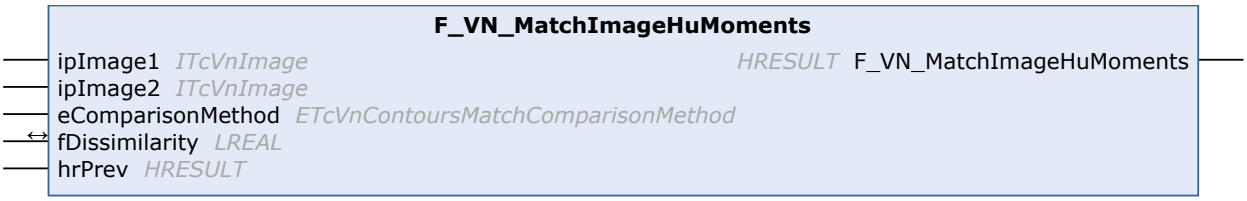
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.17 F_VN_MatchImageHuMoments



Compare two images using the Hu moment invariants.

语法

定义：

```
FUNCTION F_VN_MatchImageHuMoments : HRESULT
VAR_INPUT
    ipImage1          : ITcVnImage;
    ipImage2          : ITcVnImage;
    eComparisonMethod : ETcVnContoursMatchComparisonMethod;
END_VAR
VAR_IN_OUT
    fDissimilarity    : LREAL;
END_VAR
VAR_INPUT
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage1	ITcVnImage [▶ 407]	第一个图像（1 通道）
ipImage2	ITcVnImage [▶ 407]	第二个图像（1 通道）
eComparisonMethod	ETcVnContoursMatchComparisonMethod [▶ 184]	用于比较图像 Hu 不变矩的方法
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fDissimilarity	LREAL	根据所选的比较方法，返回图像 Hu 不变矩的相异度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.18 F_VN_MatchTemplate



Match a template image with every location in the source image (using the TCVN_TMM_CCORR_NORMED method) and save the comparison results.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_MatchTemplate : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipTemplateImage : ITcVnImage;
    ipResultImage   : Reference To ITcVnImage;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 或 REAL，1 或 3 通道）
ipTemplateImage	ITcVnImage [▶ 407]	模板图像（与 ipSrcImage 类型相同，宽度和高度较小）
ipResultImage	引用 ITcVnImage [▶ 407]	返回结果图像（REAL，归一化为 [0..1]，1 通道，尺寸：（ipSrcImage.width - ipTemplateImage.width + 1）x（ipSrcImage.height - ipTemplateImage.height + 1）。最佳匹配是全局最大值。ipResultImage 中的位置是 ipSrcImage 中 ipTemplateImage 位置的左上角）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

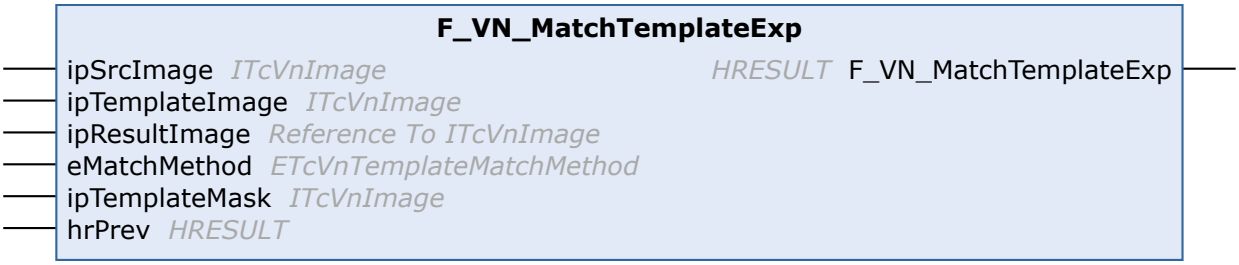
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.19 F_VN_MatchTemplateExp



Match a template image with every location in the source image and save the comparison results. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_MatchTemplateExp : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipTemplateImage : ITcVnImage;
    ipResultImage   : Reference To ITcVnImage;
    eMatchMethod    : ETcVnTemplateMatchMethod;
    ipTemplateMask  : ITcVnImage;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 或 REAL，1 或 3 通道）
ipTemplateImage	ITcVnImage [▶ 407]	模板图像（与 ipSrcImage 类型相同，宽度和高度较小）
ipResultImage	引用 ITcVnImage [▶ 407]	返回结果图像（REAL，1 通道，尺寸：（ipSrcImage.width - ipTemplateImage.width + 1）x（ipSrcImage.height - ipTemplateImage.height + 1）。最佳匹配是全局最小值（SQDIFF（_NORMED））或最大值（CCORR（_NORMED），CCOEFF（_NORMED））。ipResultImage 中的位置是 ipSrcImage 中 ipTemplateImage 位置的左上角）
eMatchMethod	ETcVnTemplateMatchMethod [▶ 213]	指定模板匹配方法
ipTemplateMask	ITcVnImage [▶ 407]	ipTemplateImage 的可选掩膜（类型和大小与 ipTemplateImage 相同）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

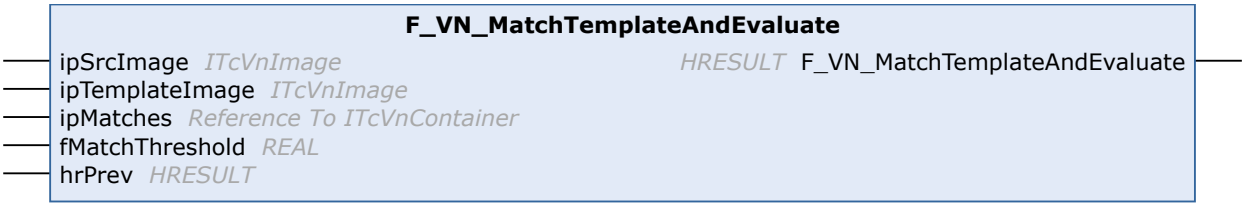
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.20 F_VN_MatchTemplateAndEvaluate



Match a template image with every location in the source image (using the TCVN_TMM_CCORR_NORMED method) and evaluate the comparison results. Returns a sorted list of possible matches (best match first).
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_MatchTemplateAndEvaluate : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipTemplateImage : ITcVnImage;
    ipMatches       : Reference To ITcVnContainer;
    fMatchThreshold : REAL;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 或 REAL，1 或 3 通道）
ipTemplateImage	ITcVnImage [▶ 407]	模板图像（与 ipSrcImage 类型相同，宽度和高度较小）
ipMatches	引用 ITcVnContainer [▶ 365]	返回 ipSrcImage 中的匹配位置（ContainerType_Vector_TcVnPoint2_DINT，其中每个元素代表 ipTemplateImage 的左上角），按相关性排序（最佳匹配项优先）
fMatchThreshold	REAL	用于区分相关匹配和不相关匹配的阈值（0...1，1.0 表示完全匹配。为了找到合适的值，可以对 F_VN_MatchTemplate 的一些示例结果图像进行评估。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

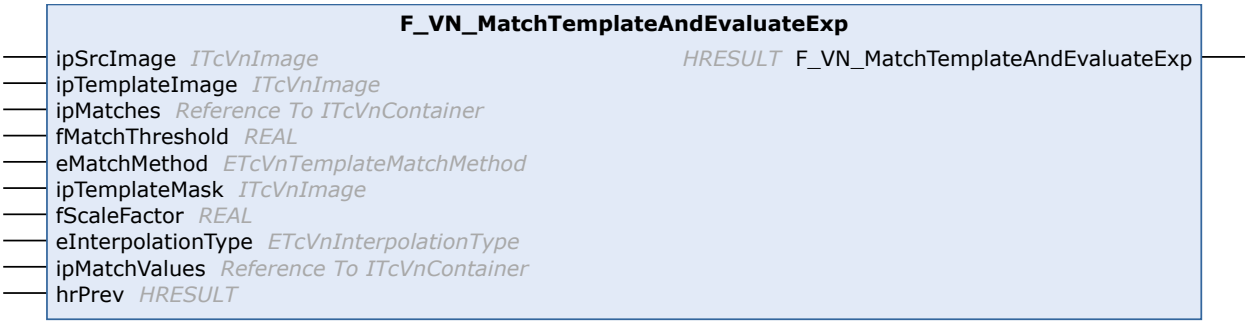
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.21 F_VN_MatchTemplateAndEvaluateExp



Match a template image with every location in the source image and evaluate the comparison results. Returns a sorted list of possible matches (best match first). (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_MatchTemplateAndEvaluateExp : HRESULT
VAR_INPUT
    ipSrcImage          : ITcVnImage;
    ipTemplateImage     : ITcVnImage;
    ipMatches           : Reference To ITcVnContainer;
    fMatchThreshold     : REAL;
    eMatchMethod        : ETcVnTemplateMatchMethod;
    ipTemplateMask      : ITcVnImage;
    fScaleFactor        : REAL;
    eInterpolationType  : ETcVnInterpolationType;
    ipMatchValues       : Reference To ITcVnContainer;
    hrPrev              : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 或 REAL，1 或 3 通道）
ipTemplateImage	ITcVnImage [▶ 407]	模板图像（与 ipSrcImage 类型相同，宽度和高度较小）
ipMatches	引用 ITcVnContainer [▶ 365]	返回 ipSrcImage 中的匹配位置（ContainerType_Vector_TcVnPoint2_DINT，其中每个元素代表 ipTemplateImage 的左上角），按相关性排序（最佳匹配项优先）
fMatchThreshold	REAL	用于区分相关匹配和不相关匹配的阈值（NORMED 方法为 0...1，其它取决于模板大小和内容。为了找到合适的值，可以对 F_VN_MatchTemplateExp 的一些示例结果图像进行评估。）
eMatchMethod	ETcVnTemplateMatchMethod [▶ 213]	指定模板匹配方法
ipTemplateMask	ITcVnImage [▶ 407]	ipTemplateImage 的可选掩膜（类型和大小与 ipTemplateImage 相同）
fScaleFactor	REAL	系数（0..1），用于减小源图像和模板图像的宽度和高度，以提高性能（但精度降低！）

名称	类型	描述
eInterpolationType	ETcVnInterpolationType [▶ 198]	图像大小调整插值类型（仅在 fScaleFactor != 1 时使用，建议大多数情况下使用 TCVN_IT_BILINEAR）
ipMatchValues	引用 ITcVnContainer [▶ 365]	可选择返回匹配值（ContainerType_Vector_REAL，大小和排序与 ipMatches 相同。如果不需要，则设为 0。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

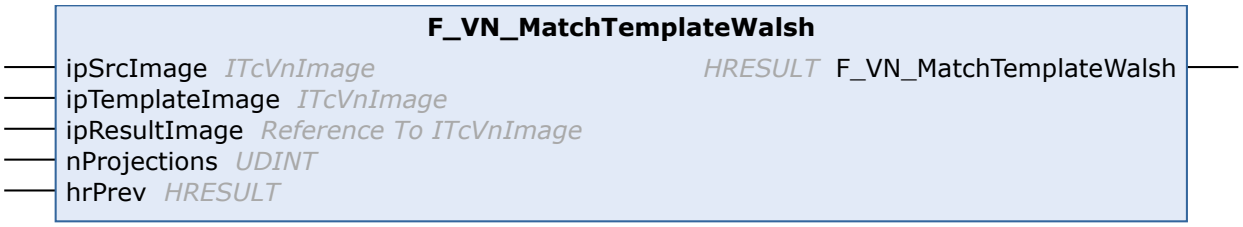
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.22 F_VN_MatchTemplateWalsh



Match a template image with every location in the source image (using the Walsh transform) and return the comparison results.
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_MatchTemplateWalsh : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipTemplateImage : ITcVnImage;
    ipResultImage   : Reference To ITcVnImage;
    nProjections    : UDINT;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 或 SINT，1 或 3 通道）
ipTemplateImage	ITcVnImage [▶ 407]	模板图像（与 ipSrcImage 类型相同，宽度和高度较小）

名称	类型	描述
ipResultImage	引用 ITcVnImage [▶ 407]	返回结果图像（REAL，归一化为 [0..1]，1 通道，尺寸：（ipSrcImage.width - ipTemplateImage.width + 1）x（ipSrcImage.height - ipTemplateImage.height + 1）。最佳匹配是全局最小值。ipResultImage 中的位置是 ipSrcImage 中 ipTemplateImage 位置的左上角。）
nProjections	UDINT	要使用的Walsh投影数量。nProjections 值越高，匹配时考虑的细节就越多。由于最相关的信息由前几个Walsh投影所捕获，因此较少的数量（例如 5、10、25 或 50）即可满足许多使用场景。每个投影都需要额外的计算时间。因此，只要满足匹配要求，选择较少的Walsh投影数量是有利的。允许的最大 nProjections 值由模板图像的宽度和高度决定。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

`F_VN_MatchTemplateWalsh` 函数使用Walsh变换来定位参考图像（模板）在搜索图像中的所有实例，并将相似度值作为结果图像返回。模板仅以其原始方向与输入图像中的每个可能位置进行比较。

参数

输入图像

输入图像 `ipSrcImage` 必须是 `USINT` 或 `SINT` 类型的单通道或三通道图像。对于三通道图像，匹配将在所有三个通道中联合执行，这也会导致处理时间增加。

模板图像

模板图像 `ipTemplateImage` 必须与 `ipSrcImage` 具有相同的类型，并且尺寸（高度和宽度）小于输入图像。

结果图像

生成的图像 `ipResultImage` 是一个具有以下尺寸的单通道 `REAL` 图像：

- 宽度：`ipSrcImage.Width - ipTemplateImage.Width + 1`
- 高度：`ipSrcImage.Height - ipTemplateImage.Height + 1`
- 通道：每个测试的旋转步长对应一个通道（默认值：1）

每个像素值代表模板与输入图像中相应图像部分之间的相似度。数值越小，匹配度越高。所有通道中的全局最小值即为最佳匹配。结果图像中最小值的位置对应于输入图像中模板位置的左上角。

例如，F_VN_MinPixelValue [▶ 1233] 可以用于确定最小像素值以及相关的位置和通道。

Walsh投影数量

`nProjections` 参数指定了用于匹配的WAlsh投影的数量。较低的投影数量主要捕获粗略的图像结构，而较高的投影数量则会逐步捕获更精细的细节，但会增加处理时间。因此，`nProjections` 应设置得尽可能低，但又要高到足以满足需求。在许多应用中，5-50 个投影已足够。`nProjections` 的允许上限受模板图像的宽度和高度限制。

相关函数

- F_VN_FindTemplateWalsh(Exp) [▶ 1184]：在输入图像中搜索模板，并返回找到的匹配项的排序列表。
- F_VN_FindTemplateWalsh(Exp)_AngleRange [▶ 1188]：与 `F_VN_FindTemplateWalsh(Exp)` 类似，但具有可自由定义的角度范围和任意数量的角度步长，以便有针对性地优化运行时和精度。

- F_VN_MatchTemplateWalsh(Exp) [▶ 1205]：将模板与输入图像中的每个位置进行比较，并返回原始结果图像，以进行特定应用的后处理。

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.1.23 F_VN_MatchTemplateWalshExp



Match a template image with every location in the source image (using the Walsh transform) and return the comparison results. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_MatchTemplateWalshExp : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipTemplateImage : ITcVnImage;
    ipResultImage   : Reference To ITcVnImage;
    nProjections    : UDINT;
    eOptions        : UDINT;
    nRotations      : UDINT;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 或 SINT，1 或 3 通道）
ipTemplateImage	ITcVnImage [▶ 407]	模板图像（与 ipSrcImage 类型相同，宽度和高度较小）
ipResultImage	引用 ITcVnImage [▶ 407]	返回结果图像（REAL，归一化为 [0...1]，nRotations 通道，尺寸：（ipSrcImage.width - ipTemplateImage.width + 1）x（ipSrcImage.height - ipTemplateImage.height + 1）。最佳匹配是所有通道中的全局最小值。ipResultImage 中的位置是 ipSrcImage 中 ipTemplateImage 位置的左上角。旋转可以通过最小值所在通道来推断。）
nProjections	UDINT	要使用的Walsh投影数量。nProjections 值越高，匹配时考虑的细节就越多。由于最相关的信息由前几个Walsh投影所捕获，因此较少的数量（例如 5、10、25 或 50）即可满足许多使用场景。每个投影都需要额外的计算时间。因此，只要满足匹配要求，选择较少的Walsh投影数量是有利的。允许的最大 nProjections 值由模板图像的宽度和高度决定。

名称	类型	描述
eOptions	UDINT	控制匹配过程的标志。（请参见 ETcVnMatchTemplateWalshOptions）
nRotations	UDINT	要检查的旋转数量。（有效值：1（0°）、2（0° 和 180°）和 4（0°、90°、180° 和 270°；仅适用于正方形图像））
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

`F_VN_MatchTemplateWalshExp` 函数是 F_VN_MatchTemplateWalsh [▶ 1205] 的专家变体，包含附加参数。它使用Walsh变换来定位参考图像（模板）在搜索图像中的所有实例，并将相似度值作为结果图像返回。与标准变体相比，模板可以通过多个旋转步长进行检查。

参数

输入图像

输入图像 `ipSrcImage` 必须是 `USINT` 或 `SINT` 类型的单通道或三通道图像。对于三通道图像，匹配将在所有三个通道中联合执行，这也会导致处理时间增加。

模板图像

模板图像 `ipTemplateImage` 必须与 `ipSrcImage` 具有相同的类型，并且尺寸（高度和宽度）小于输入图像。

结果图像

生成的图像 `ipResultImage` 是一个具有以下尺寸的单通道 `REAL` 图像：

- 宽度：`ipSrcImage.Width - ipTemplateImage.Width + 1`
- 高度：`ipSrcImage.Height - ipTemplateImage.Height + 1`
- 通道：每个测试的旋转步长对应一个通道（默认值：1）

每个像素值代表模板与输入图像中相应图像部分之间的相似度。数值越小，匹配度越高。所有通道中的全局最小值即为最佳匹配。结果图像中最小值的位置对应于输入图像中模板位置的左上角。

例如，F_VN_MinPixelValue [▶ 1233] 可以用于确定最小像素值以及相关的位置和通道。

Walsh投影数量

`nProjections` 参数指定了用于匹配的WAlsh投影的数量。较低的投影数量主要捕获粗略的图像结构，而较高的投影数量则会逐步捕获更精细的细节，但会增加处理时间。因此，`nProjections` 应设置得尽可能低，但又要高到足以满足需求。在许多应用中，5-50 个投影已足够。`nProjections` 的允许上限受模板图像的宽度和高度限制。

Options

通过 `eOptions` 可以指定影响功能和结果输出的选项（ETcVnMatchTemplateWalshOptions [▶ 202]）。多个选项可以使用 OR 运算符进行组合。提供以下选项：

- `TCVN_FTWO_NONE`：无特殊选项。
- `TCVN_MTWO_NORMED`：将输出图像的元素归一化到值范围 `[0, 1.0]`。
- `TCVN_FTWO_SKIP_DC`：在相似度计算中跳过 DC 部分，这可以提高对光照变化的适应性。通常需要更多的投影数量。

要检查的旋转数量

`nRotations` 参数指定了要检查的旋转步数，从而既定义了所检测的角度位置，也决定了结果图像中对应的通道数量。找到最小值所在的通道与匹配项的旋转角度相对应。有效值为：

- 1（0°）
- 2（0° 和 180°）
- 4（0°、90°、180° 和 270°）- 仅适用于正方形模板图像

相关函数

- F_VN_FindTemplateWalsh(Exp) [▶ 1184]：在输入图像中搜索模板，并返回找到的匹配项的排序列表。
- F_VN_FindTemplateWalsh(Exp)_AngleRange [▶ 1188]：与 `F_VN_FindTemplateWalsh(Exp)` 类似，但具有可自由定义的角度范围和任意数量的角度步长，以便有针对性地优化运行时和精度。
- F_VN_MatchTemplateWalsh(Exp) [▶ 1205]：将模板与输入图像中的每个位置进行比较，并返回原始结果图像，以进行特定应用的后处理。

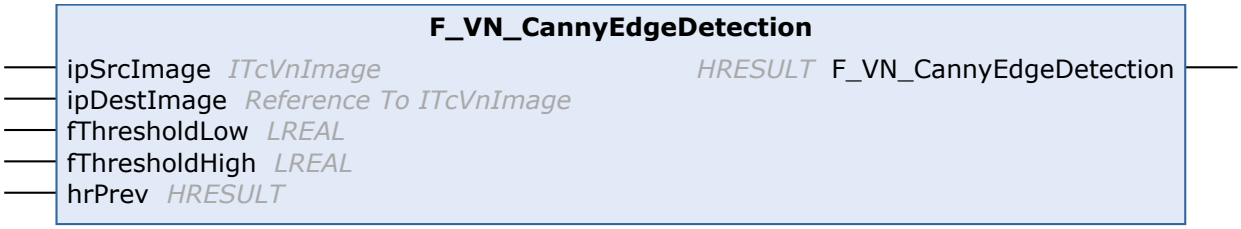
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.2 F_VN_CannyEdgeDetection



Find edges using the Canny edge detection algorithm.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_CannyEdgeDetection : HRESULT
VAR_INPUT
    ipSrcImage     : ITcVnImage;
    ipDestImage    : Reference To ITcVnImage;
    fThresholdLow  : LREAL;
    fThresholdHigh : LREAL;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 类型元素）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（USINT.ID 类型元素。如有需要，将创建适当的目标图像。）
fThresholdLow	LREAL	低阈值

名称	类型	描述
fThresholdHigh	LREAL	高阈值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

这个函数在图像中寻找边缘，并在黑色背景上通过白线进行展示说明。此外，这个专家版 F_VN_CannyEdgeDetectionExp [▶ 1211]存在高级参数化选项。

算法

Canny Edge 边缘检测是一种多阶段的算法，用于实现最稳健的边缘检测。它包括下列步骤：

1. 通过高斯过滤器减少噪音。
2. 通过索贝尔算子寻找边缘。
3. 通过非最大限度的抑制来进行线条瘦身，即所产生的线条为 1px 厚。
4. 滞后，所以只留下粗线（见下面的阈值参数）。

参数

原始图像

预计原始图像`ipSrcImage`是一个单通道 8 位强度图像（像素格式`TCVN_ET_USINT`）。因此，可以使用灰度图像或彩色图像的各个通道。

结果图像

结果图像`ipDestImage`格式相同。

阈值

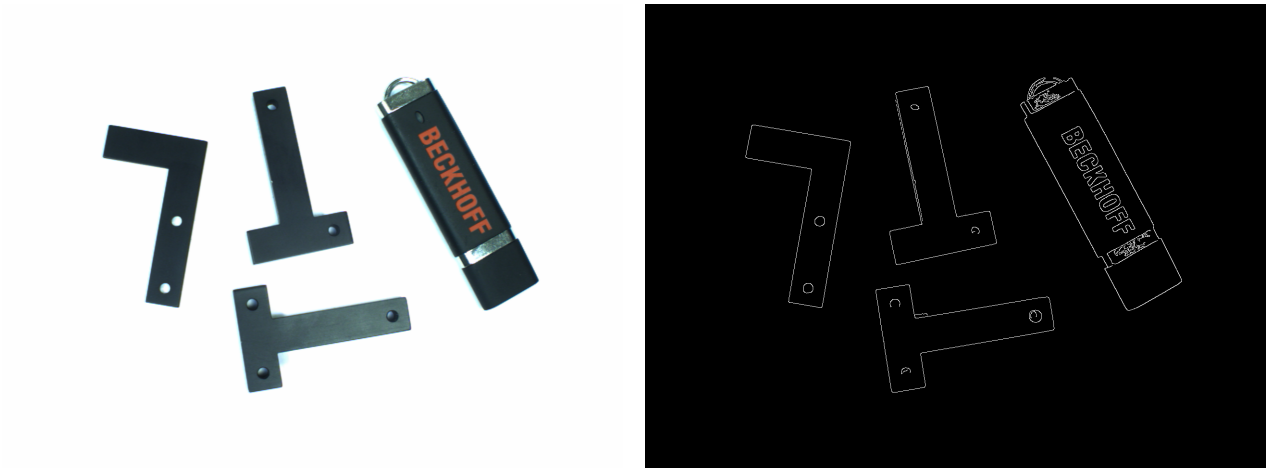
两个阈值`fThresholdLow`和`fThresholdHigh`决定接受哪些发现的边缘：接受梯度高于`fThresholdHigh`的像素。丢弃颜色梯度低于`fThresholdLow`的像素。中间梯度的像素仅在与已经接受的像素相连接时才会被接受。

通过这两个阈值，可实现灵活设置。通过设置两个相同数值，也可以实现硬阈值。

如果阈值设置过高，可能会丢失重要的边缘信息。如果阈值过低，噪声和其他不相关的信息可能会解释为边缘。

应用

```
hr := F_VN_CannyEdgeDetection(
    ipSrcImage      :=  ipImageIn,
    ipDestImage     :=  ipImageWork,
    fThresholdLow   :=  0,
    fThresholdHigh  :=  200,
    hr
);
```



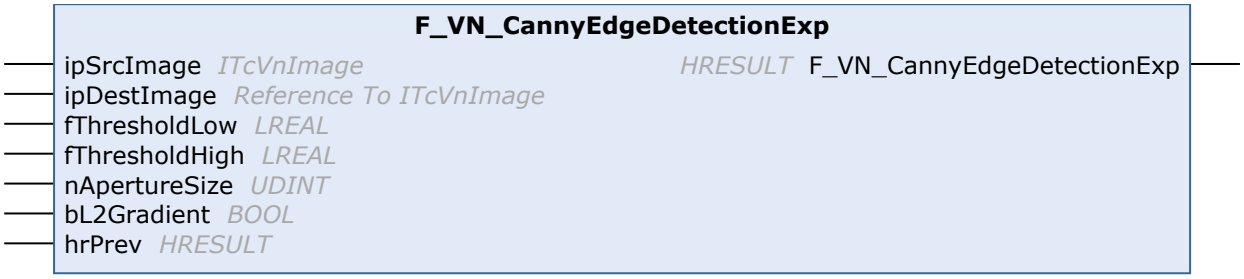
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.3 F_VN_CannyEdgeDetectionExp



Find edges using the Canny edge detection algorithm. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_CannyEdgeDetectionExp : HRESULT
VAR_INPUT
    ipSrcImage     : ITcVnImage;
    ipDestImage    : Reference To ITcVnImage;
    fThresholdLow  : LREAL;
    fThresholdHigh : LREAL;
    nApertureSize  : UDINT;
    bL2Gradient    : BOOL;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 类型元素）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（USINT.ID 类型元素。如有需要，将创建适当的目标图像。）

名称	类型	描述
fThresholdLow	LREAL	低阈值
fThresholdHigh	LREAL	高阈值
nApertureSize	UDINT	索贝尔（Sobel）算子的孔径大小（3、5、7）
bL2Gradient	BOOL	如果为 true，则使用更精确（也更慢）的 L2 范数，而不是 L1 范数
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

该函数是F_VN_CannyEdgeDetection [▶ 1209]的专家版。它包含高级参数化选项。

参数

在该专家版中额外提供了以下参数：

光圈大小

仅奇数值3或更大对于nApertureSize有效。

L1/L2 标准

两个标准可用于计算梯度。L1 标准更快。反之，L2 标准则更准确。

$$L_1 = \left|\frac{dI}{dx}\right| + \left|\frac{dI}{dy}\right| \qquad L_2 = \sqrt{\left(\frac{dI}{dx}\right)^2 + \left(\frac{dI}{dy}\right)^2}$$

将参数bL2Gradient设为FALSE，以使用 L1 标准，设为TRUE，以使用 L2 标准。

应用

例如，这个函数的应用可以如下所示：

```
hr := F_VN_CannyEdgeDetectionExp(
    ipSrcImage      :=  ipImageIn,
    ipDestImage     :=  ipImageWork,
    fThresholdLow   :=  0,
    fThresholdHigh  :=  200,
  nApertureSize  := 5,
  bL2Gradient    := TRUE,
    hr
);
```

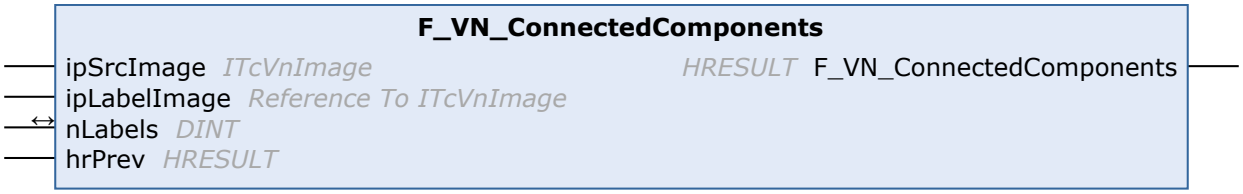
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.4 F_VN_ConnectedComponents



Computes the connected components of an image.

语法

定义：

```
FUNCTION F_VN_ConnectedComponents : HRESULT
VAR_INPUT
    ipSrcImage   : ITcVnImage;
    ipLabelImage : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    nLabels      : DINT;
END_VAR
VAR_INPUT
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道，二值）
ipLabelImage	引用 ITcVnImage [▶ 407]	返回每个源图像像素的标签（1 通道，DINT. 如有需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nLabels	DINT	返回标签的数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.5 F_VN_ConnectedComponentsExp



Computes the connected components of an image. (expert function)

语法

定义：

```
FUNCTION F_VN_ConnectedComponentsExp : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipLabelImage  : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    nLabels       : DINT;
END_VAR
VAR_INPUT
    eConnectivity : ETcVnPixelConnectivity;
    eLabelType    : ETcVnElementType;
    eAlgorithm    : ETcVnConnectedComponentsAlgorithm;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道，二值）
ipLabelImage	引用 ITcVnImage [▶ 407]	返回每个源图像像素的标签（1 通道，类型取决于 eLabelType。如有需要，将创建适当的目标图像。）
eConnectivity	ETcVnPixelConnectivity [▶ 206]	选择使用像素 4 连通还是 8 连通
eLabelType	ETcVnElementType [▶ 191]	选择 ipLabelImage 的类型（仅支持 UINT 或 DINT）
eAlgorithm	ETcVnConnectedComponentsAlgorithm [▶ 178]	选择应用的算法
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nLabels	DINT	返回标签的数量

返回值

HRESULT [▶ 135]

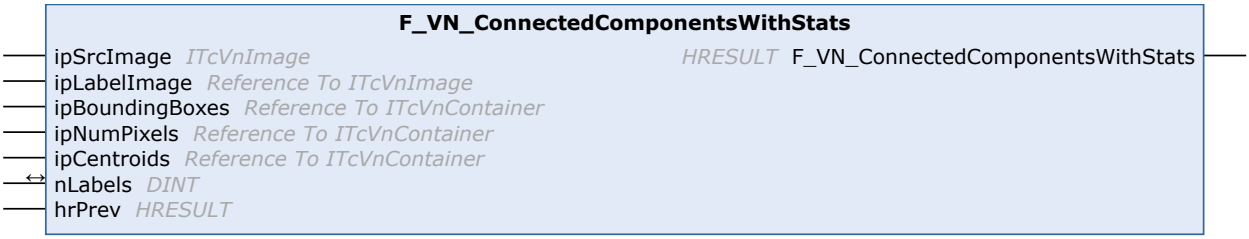
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.6 F_VN_ConnectedComponentsWithStats



Computes the connected components and corresponding statistics of an image.

语法

定义：

```
FUNCTION F_VN_ConnectedComponentsWithStats : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipLabelImage    : Reference To ITcVnImage;
    ipBoundingBoxes : Reference To ITcVnContainer;
    ipNumPixels     : Reference To ITcVnContainer;
    ipCentroids     : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    nLabels         : DINT;
END_VAR
VAR_INPUT
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道，二值）
ipLabelImage	引用 ITcVnImage [▶ 407]	返回每个源图像像素的标签（1 通道，DINT. 如有需要，将创建适当的目标图像。）
ipBoundingBoxes	引用 ITcVnContainer [▶ 365]	返回每个标记区域的边界框。（ContainerType_Vector_TcVnRectangle_DINT；重复使用非零接口指针。）
ipNumPixels	引用 ITcVnContainer [▶ 365]	返回每个标记区域的像素数量。（ContainerType_Vector_DINT；重复使用非零接口指针。）
ipCentroids	引用 ITcVnContainer [▶ 365]	返回每个标记区域的重心。（ContainerType_Vector_TcVnPoint2_LREAL；重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nLabels	DINT	返回标签的数量

返回值

HRESULT [▶ 135]

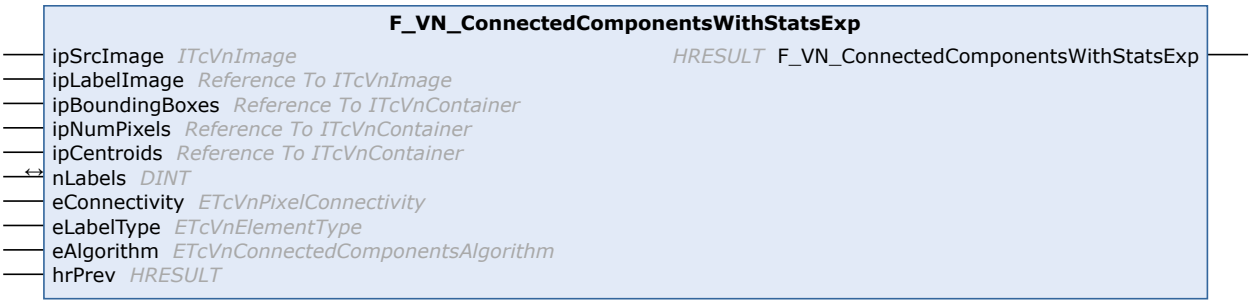
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.7 F_VN_ConnectedComponentsWithStatsExp



Computes the connected components and corresponding statistics of an image. (expert function)

语法

定义：

```
FUNCTION F_VN_ConnectedComponentsWithStatsExp : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipLabelImage    : Reference To ITcVnImage;
    ipBoundingBoxes : Reference To ITcVnContainer;
    ipNumPixels     : Reference To ITcVnContainer;
    ipCentroids     : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    nLabels         : DINT;
END_VAR
VAR_INPUT
    eConnectivity   : ETcVnPixelConnectivity;
    eLabelType      : ETcVnElementType;
    eAlgorithm      : ETcVnConnectedComponentsAlgorithm;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道，二值）
ipLabelImage	引用 ITcVnImage [▶ 407]	返回每个源图像像素的标签（1 通道，DINT. 如有需要，将创建适当的目标图像。）
ipBoundingBoxes	引用 ITcVnContainer [▶ 365]	返回每个标记区域的边界框。（ContainerType_Vector_TcVnRectangle_DINT；重复使用非零接口指针。）
ipNumPixels	引用 ITcVnContainer [▶ 365]	返回每个标记区域的像素数量。（ContainerType_Vector_DINT；重复使用非零接口指针。）

名称	类型	描述
ipCentroids	引用 ITcVnContainer [▶ 365]	返回每个标记区域的重心。（ContainerType_Vector_TcVnPoint2_LREAL；重复使用非零接口指针。）
eConnectivity	ETcVnPixelConnectivity [▶ 206]	选择使用像素 4 连通还是 8 连通
eLabelType	ETcVnElementType [▶ 191]	选择 ipLabelImage 的类型（仅支持 UINT 或 DINT）
eAlgorithm	ETcVnConnectedComponentsAlgorithm [▶ 178]	选择应用的算法
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nLabels	DINT	返回标签的数量

返回值

HRESULT [▶ 135]

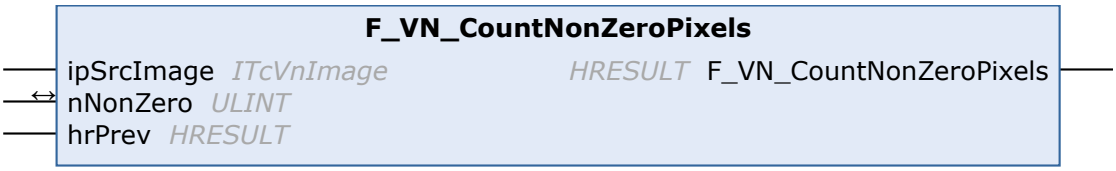
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.8 F_VN_CountNonZeroPixels



Counts the non-zero pixels in a single-channel image (e.g. useful to analyze threshold results).

语法

定义：

```
FUNCTION F_VN_CountNonZeroPixels : HRESULT
VAR_INPUT
    ipSrcImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    nNonZero   : ULINT;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nNonZero	ULINT	返回 ipSrcImage 中非零像素的数量

返回值

HRESULT [▶ 135]

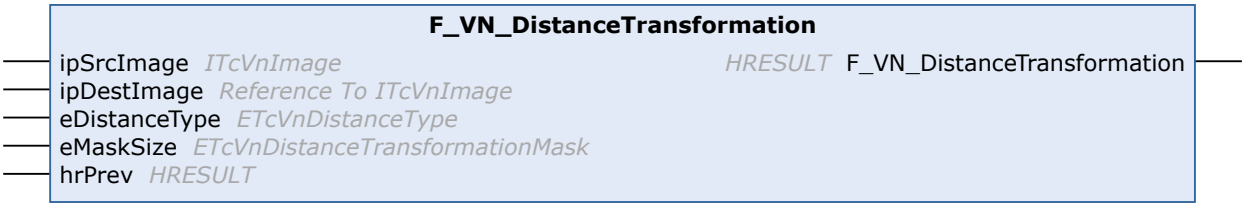
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.9 F_VN_DistanceTransformation



Calculates the distance transformation, which is the distance to the closest zero pixel in a binary image.

语法

定义：

```
FUNCTION F_VN_DistanceTransformation : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDestImage   : Reference To ITcVnImage;
    eDistanceType : ETcVnDistanceType;
    eMaskSize     : ETcVnDistanceTransformationMask;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（REAL，1 通道。如有需要，将创建适当的目标图像。）
eDistanceType	ETcVnDistanceType [▶ 188]	距离计算方法（支持：L1、L2、C）

名称	类型	描述
eMaskSize	ETcVnDistanceTransformationMask [▶ 187]	距离变换掩膜的大小
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

这个函数执行图像的距离变换。此外，还有一个专家版F_VN_DistanceTransformationExp [▶ 1220]，配有高级参数化选项。

参数

下列参数在基本版中可用：

原始图像

预计原始图像`ipSrcImage`为 1 通道的 8 位图像（元素类型`TCVN_ET_USINT`）。其他图像格式必须首先进行转换。

结果图像

结果图像`ipDestImage`以元素类型`TCVN_ET_REAL`返回。每个像素的值描述了各自像素与原始图像中最近的 0 值像素之间的距离。

在 ADS 图像查看中显示

为了能够在 ADS 图像查看中显示结果图像，必须首先将其转换为兼容的图像格式。根据使用情况，对像素值进行缩放可能有用。

距离类型

距离类型`eDistanceType`确定计算像素和最近 0 值像素之间距离的方程式。支持枚举ETcVnDistanceType [▶ 188]的下列距离类型：

- TCVN_DT_L1
- TCVN_DT_L2
- TCVN_DT_C

掩码尺寸

掩码大小`eMaskSize`描述了，在选择 L2 标准作为距离类型的情况下，欧几里得距离的近似值。

- TCVN_DTM_PRECISE：精确计算欧几里得距离。
- TCVN_DTM_3：欧几里得距离被近似为 3x3 掩码（水平、垂直和对角线）内距离元素的总和。
- TCVN_DTM_5：欧几里得距离被近似为 5x5 掩码（水平、垂直、对角线和骑士的移动（如国际象棋））内的距离元素之和。

掩码大小对距离类型 L1 和 C 没有影响。相反，内部总是使用 3x3 掩码。

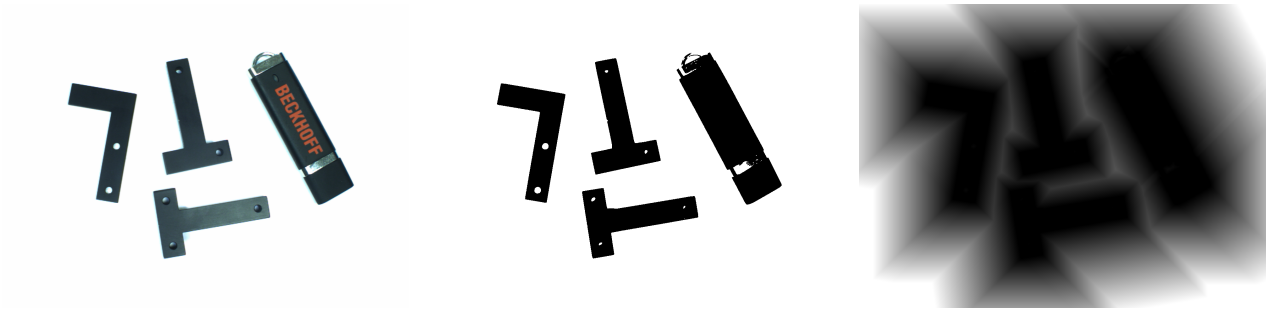
应用

例如，这个函数的应用可以如下所示：

```
// convert to binary image
hr := F_VN_Threshold(ipImageIn, ipImageWork, 128, 255, TCVN_TT_BINARY, hr);
// apply distance transformation
hr := F_VN_DistanceTransformation(
```

```
  ipSrcImage    := ipImageWork,
  ipDestImage   := ipImageWork,
  eDistanceType := TCVN_DT_C,
  eMaskSize     := TCVN_DTM_3,
  hr
);
// convert image to display it in ADS Image Watch
hr := F_VN_ConvertElementType(ipImageWork, ipImageWork, TCVN_ET_USINT, hr);
```

原始图像 **二进制图像** **结果图像**



所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.10 F_VN_DistanceTransformationExp



Calculates the distance transformation, which is the distance to the closest zero pixel in a binary image. Additionally, an image with component labels is created (discrete Voronoi diagram). (expert function)

语法

定义：

```
FUNCTION F_VN_DistanceTransformationExp : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDestImage   : Reference To ITcVnImage;
    eDistanceType : ETcVnDistanceType;
    eMaskSize     : ETcVnDistanceTransformationMask;
    ipDestLabels  : Reference To ITcVnImage;
    eLabelType    : ETcVnDistanceTransformationLabel;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道）

名称	类型	描述
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（REAL，1 通道。如有需要，将创建适当的目标图像。）
eDistanceType	ETcVnDistanceType [▶ 188]	距离计算方法（支持：L1、L2、C）
eMaskSize	ETcVnDistanceTransformationMask [▶ 187]	距离变换掩膜的大小（不支持 PRECISE 用于标签计算）
ipDestLabels	引用 ITcVnImage [▶ 407]	返回分量标签（离散维诺图（Voronoi）；DINT，1 通道。如有需要，将创建适当的图像）。
eLabelType	ETcVnDistanceTransformationLabel [▶ 187]	标签类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

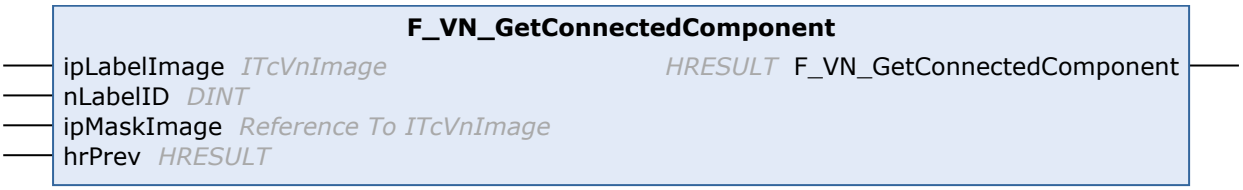
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.11 F_VN_GetConnectedComponent



Get the mask image of a connected component from a label image based on the label ID.

语法

定义：

```
FUNCTION F_VN_GetConnectedComponent : HRESULT
VAR_INPUT
    ipLabelImage : ITcVnImage;
    nLabelID     : DINT;
    ipMaskImage  : Reference To ITcVnImage;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipLabelImage	ITcVnImage [▶ 407]	包含每个像素标签的图像（1 通道 DINT 或 UINT）。
nLabelID	DINT	请求标签的值

名称	类型	描述
ipMaskImage	引用 ITcVnImage [▶ 407]	返回掩膜图像，该图像中标记属于分量的像素（1 通道 USINT。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

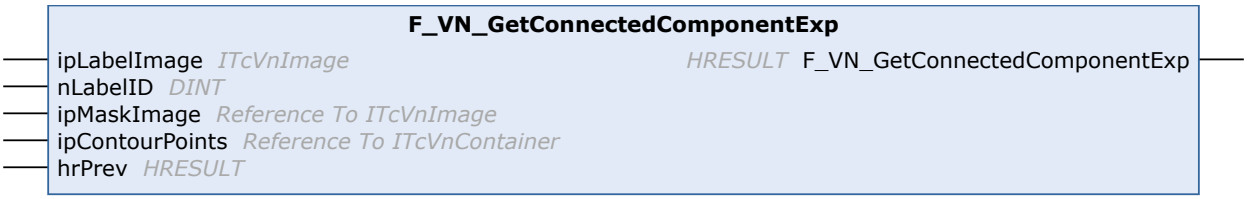
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.12 F_VN_GetConnectedComponentExp



Get the mask image (and optionally the outer contour) of a connected component from a label image based on the label ID. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_GetConnectedComponentExp : HRESULT
VAR_INPUT
    ipLabelImage    : ITcVnImage;
    nLabelID        : DINT;
    ipMaskImage     : Reference To ITcVnImage;
    ipContourPoints : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipLabelImage	ITcVnImage [▶ 407]	包含每个像素标签的图像（1 通道 DINT 或 UINT）。
nLabelID	DINT	请求标签的值
ipMaskImage	引用 ITcVnImage [▶ 407]	返回掩膜图像，该图像中标记属于分量的像素（1 通道 USINT。掩膜为可选项，如果不需要，则设为 0。）
ipContourPoints	引用 ITcVnContainer [▶ 365]	返回分量的外轮廓（ContainerType_Vector_TcVnPoint2_DINT。轮廓是可选项，如果不需要，则设为 0。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

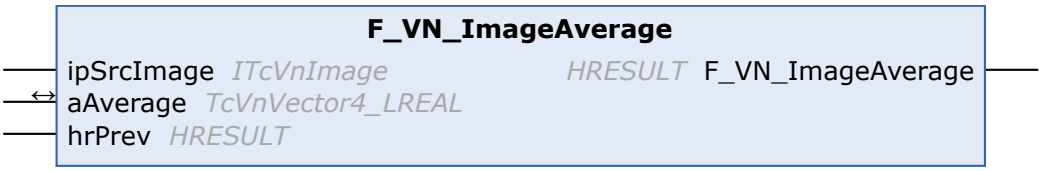
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.13 F_VN_ImageAverage



Computes the (channel-wise) average pixel value of an image.

语法

定义：

```
FUNCTION F_VN_ImageAverage : HRESULT
VAR_INPUT
    ipSrcImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aAverage   : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 - 4 通道）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aAverage	TcVnVector4_LREAL [▶ 153]	返回 ipSrcImage 的（通道）平均像素值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.14 F_VN_ImageAverageExp



Computes the (channel-wise) average pixel value of an image. (expert function)

语法

定义：

```
FUNCTION F_VN_ImageAverageExp : HRESULT
VAR_INPUT
    ipSrcImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aAverage   : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipMask     : ITcVnImage;
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 - 4 通道）
ipMask	ITcVnImage [▶ 407]	USINT 类型的掩膜（1 通道）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aAverage	TcVnVector4_LREAL [▶ 153]	返回 ipSrcImage 的（通道）平均像素值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.15 F_VN_ImageAverageStdDev



Computes the (channel-wise) average pixel value and the corresponding standard deviation of an image.

语法

定义：

```
FUNCTION F_VN_ImageAverageStdDev : HRESULT
VAR_INPUT
    ipSrcImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aAverage   : TcVnVector4_LREAL;
    aStdDev    : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 - 4 通道）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aAverage	TcVnVector4_LREAL [▶ 153]	返回 ipSrcImage 的（通道）平均像素值
aStdDev	TcVnVector4_LREAL [▶ 153]	返回 ipSrcImage 的（通道）像素值标准偏差

返回值

HRESULT [▶ 135]

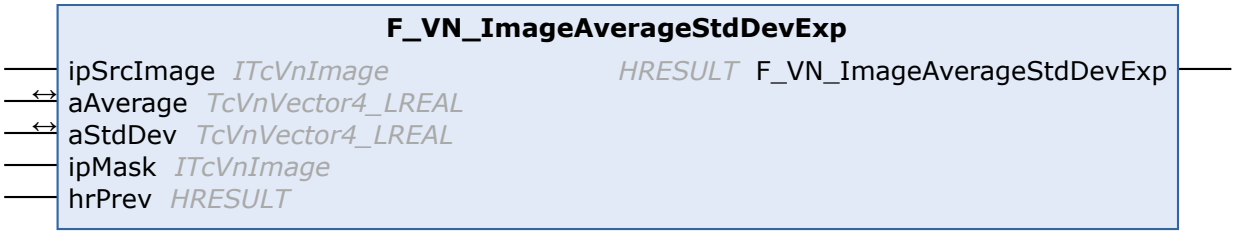
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.16 F_VN_ImageAverageStdDevExp



Computes the (channel-wise) average pixel value and the corresponding standard deviation of an image. (expert function)

语法

定义:

```
FUNCTION F_VN_ImageAverageStdDevExp : HRESULT
VAR_INPUT
    ipSrcImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aAverage   : TcVnVector4_LREAL;
    aStdDev    : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipMask     : ITcVnImage;
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 - 4 通道）
ipMask	ITcVnImage [▶ 407]	USINT 类型的掩膜（1 通道）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aAverage	TcVnVector4_LREAL [▶ 153]	返回 ipSrcImage 的（通道）平均像素值
aStdDev	TcVnVector4_LREAL [▶ 153]	返回 ipSrcImage 的（通道）像素值标准偏差

返回值

HRESULT [▶ 135]

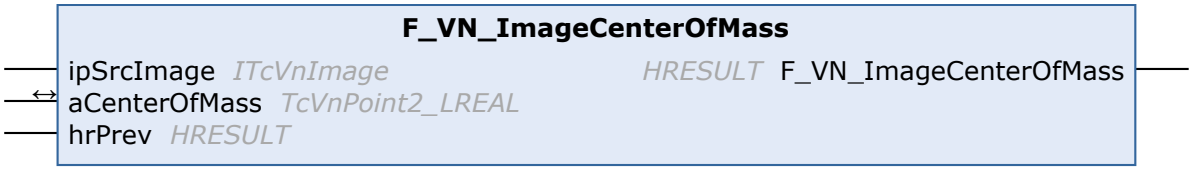
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.17 F_VN_ImageCenterOfMass



Computes the center of mass of an image.

语法

定义：

```
FUNCTION F_VN_ImageCenterOfMass : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
END_VAR
VAR_IN_OUT
    aCenterOfMass : TcVnPoint2_LREAL;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCenterOfMass	TcVnPoint2_LREAL [▶ 152]	返回图像的质心

返回值

HRESULT [▶ 135]

更多信息

这个函数计算输入图像的质量中心。专家版F_VN_ImageCenterOfMassExp [▶ 1228]存在高级参数化选项。

参数

输入图像

输入图像`ipSrcImage`必须为单通道图像。如果想要计算多通道图像的质量中心，你有以下选项：

- 可以使用F_VN_SplitImageChannels [▶ 817]将多通道图像分割成各个通道，并为每个通道计算出一个质心。
- 可以使用F_VN_ConvertColorSpace [▶ 1248]将多通道图像转换成单通道图像，并为其计算出质心。

质量中心 (返回值)

计算的质心通过参考`aCenterOfMass`返回。

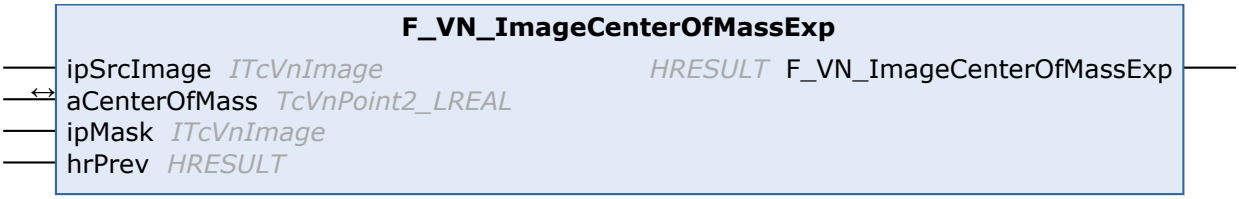
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.18 F_VN_ImageCenterOfMassExp



Computes the center of mass of an image.

语法

定义：

```
FUNCTION F_VN_ImageCenterOfMassExp : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
END_VAR
VAR_IN_OUT
    aCenterOfMass : TcVnPoint2_LREAL;
END_VAR
VAR_INPUT
    ipMask        : ITcVnImage;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
ipMask	ITcVnImage [▶ 407]	可选掩膜（1 通道，USINT 类型，宽度和高度与 ipSrcImage 相同）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aCenterOfMass	TcVnPoint2_LREAL [▶ 152]	返回图像的质心

返回值

HRESULT [▶ 135]

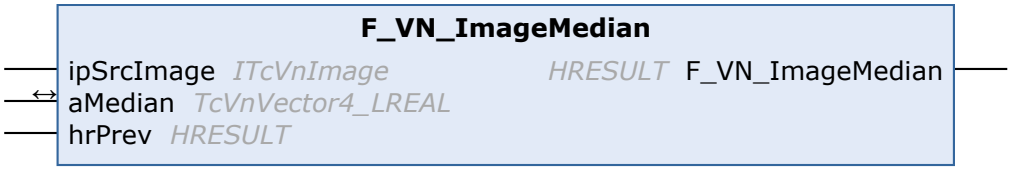
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.19 F_VN_ImageMedian



Computes the (approximated, channel-wise) median pixel value of an image.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_ImageMedian : HRESULT
VAR_INPUT
    ipSrcImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aMedian    : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 - 4 通道）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector4_LREAL [▶ 153]	返回 ipSrcImage 的（近似、通道）像素中值

返回值

HRESULT [▶ 135]

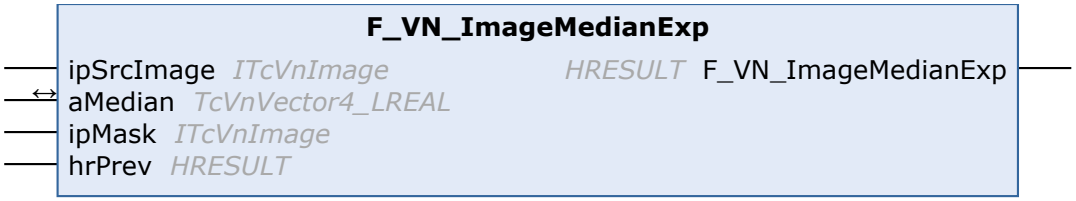
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.20 F_VN_ImageMedianExp



Computes the (approximated, channel-wise) median pixel value of an image. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_ImageMedianExp : HRESULT
VAR_INPUT
    ipSrcImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    aMedian    : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipMask     : ITcVnImage;
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 - 4 通道）
ipMask	ITcVnImage [▶ 407]	USINT 类型的掩膜（1 通道）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMedian	TcVnVector4_LREAL [▶ 153]	返回 ipSrcImage 的（近似、通道）像素中值

返回值

HRESULT [▶ 135]

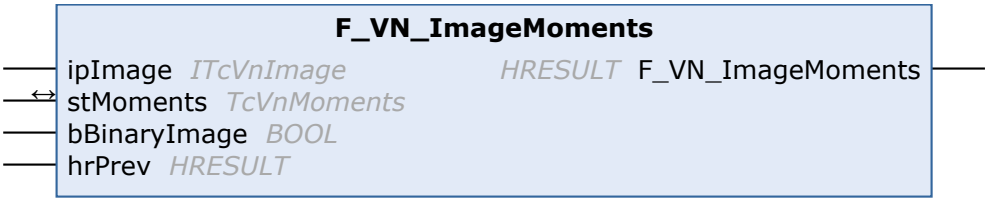
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.21 F_VN_ImageMoments



Computes the spatial moments, the central moments, and the central normalized moments of an image up to the third order.

语法

定义：

```
FUNCTION F_VN_ImageMoments : HRESULT
VAR_INPUT
    ipImage      : ITcVnImage;
END_VAR
VAR_IN_OUT
    stMoments    : TcVnMoments;
END_VAR
VAR_INPUT
    bBinaryImage : BOOL;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	源图像（1 通道，USINT 或 REAL 类型元素）
bBinaryImage	BOOL	如果为 true，所有非零像素都被视为 1
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stMoments	TcVnMoments [▶ 226]	返回包含矩的结构体

返回值

HRESULT [▶ 135]

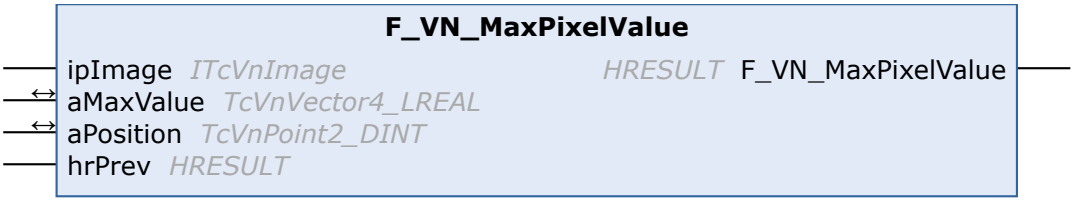
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.22 F_VN_MaxPixelValue



Finds the maximum pixel value in an image (1 - 4 channels supported).

语法

定义：

```
FUNCTION F_VN_MaxPixelValue : HRESULT
VAR_INPUT
    ipImage    : ITcVnImage;
END_VAR
VAR_IN_OUT
    aMaxValue : TcVnVector4_LREAL;
```

```
    aPosition : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	源图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMaxValue	TcVnVector4_LREAL [▶ 153]	返回最大像素值
aPosition	TcVnPoint2_DINT [▶ 152]	返回第一个找到的 aMaxValue 位置

返回值

HRESULT [▶ 135]

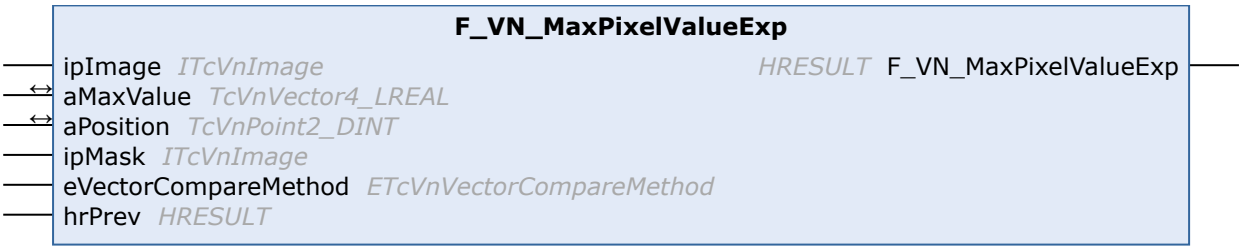
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.23 F_VN_MaxPixelValueExp



Finds the maximum pixel value in an image (1 - 4 channels supported). (expert function)

语法

定义：

```
FUNCTION F_VN_MaxPixelValueExp : HRESULT
VAR_INPUT
    ipImage               : ITcVnImage;
END_VAR
VAR_IN_OUT
    aMaxValue             : TcVnVector4_LREAL;
    aPosition             : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    ipMask                : ITcVnImage;
    eVectorCompareMethod  : ETcVnVectorCompareMethod;
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	源图像
ipMask	ITcVnImage [▶ 407]	可选掩膜，用于指定考虑哪些像素位置（USINT，如果不需要，将参数设为 0）
eVectorCompareMethod	ETcVnVectorCompareMethod [▶ 215]	为多通道图像选择向量比较方法
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMaxValue	TcVnVector4_LREAL [▶ 153]	返回最大像素值
aPosition	TcVnPoint2_DINT [▶ 152]	返回第一个找到的 aMaxValue 位置（不支持使用 ELEMENTWISE 的多通道图像）

返回值

HRESULT [▶ 135]

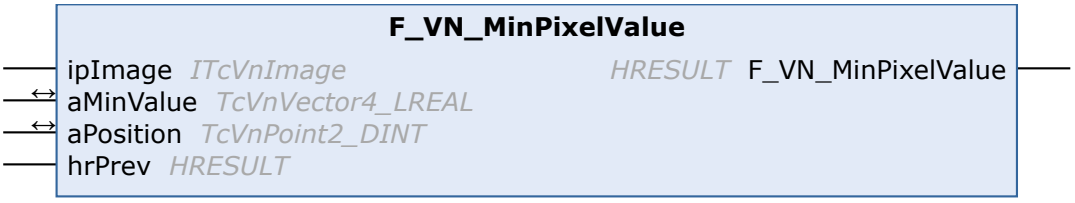
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.24 F_VN_MinPixelValue



Finds the minimum pixel value in an image (1 - 4 channels supported).

语法

定义：

```
FUNCTION F_VN_MinPixelValue : HRESULT
VAR_INPUT
    ipImage    : ITcVnImage;
END_VAR
VAR_IN_OUT
    aMinValue : TcVnVector4_LREAL;
    aPosition : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	源图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMinValue	TcVnVector4_LREAL [▶ 153]	返回最小像素值
aPosition	TcVnPoint2_DINT [▶ 152]	返回第一个找到的 aMinValue 位置

返回值

HRESULT [▶ 135]

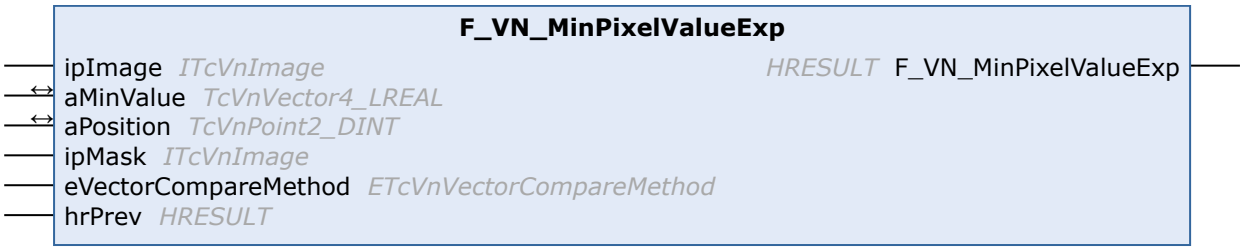
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.25 F_VN_MinPixelValueExp



Finds the minimum pixel value in an image (1 - 4 channels supported). (expert function)

语法

定义：

```
FUNCTION F_VN_MinPixelValueExp : HRESULT
VAR_INPUT
    ipImage               : ITcVnImage;
END_VAR
VAR_IN_OUT
    aMinValue             : TcVnVector4_LREAL;
    aPosition             : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    ipMask                : ITcVnImage;
    eVectorCompareMethod  : ETcVnVectorCompareMethod;
    hrPrev                : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	源图像
ipMask	ITcVnImage [▶ 407]	可选掩膜，用于指定考虑哪些像素位置（USINT，如果不需要，将参数设为 0）
eVectorCompareMethod	ETcVnVectorCompareMethod [▶ 215]	为多通道图像选择向量比较方法
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aMinValue	TcVnVector4_LREAL [▶ 153]	返回最小像素值
aPosition	TcVnPoint2_DINT [▶ 152]	返回第一个找到的 aMinValue 位置（不支持使用 ELEMENTWISE 的多通道图像）

返回值

HRESULT [▶ 135]

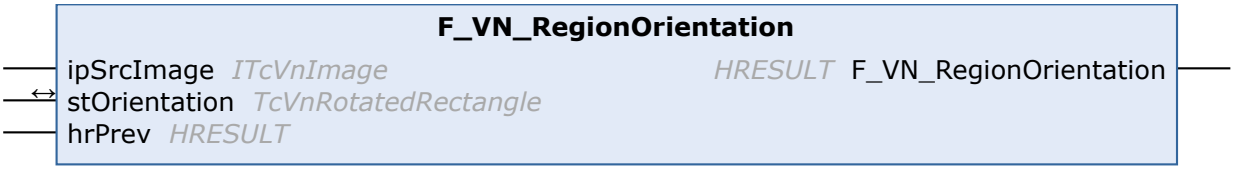
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.26 F_VN_RegionOrientation



Calculate the orientation of a region based on a binary image that contains a non zero value for every pixel in that region.

语法

定义：

```
FUNCTION F_VN_RegionOrientation : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
END_VAR
VAR_IN_OUT
    stOrientation : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道，二值）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stOrientation	TcVnRotatedRectangle [▶ 237]	旋转后的矩形，包含中心点、轴长和顺时针方向的区域旋转角。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.27 F_VN_RegionOrientationExp



Calculate the orientation of a region based on a binary image that contains a non zero value for every pixel in that region. (expert function)

语法

定义：

```
FUNCTION F_VN_RegionOrientationExp : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
END_VAR
VAR_IN_OUT
    stOrientation : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    eMethod       : ETcVnOrientationMethod;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道，二值）

名称	类型	描述
eMethod	ETcVnOrientationMethod [▶ 205]	计算方向的方法。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stOrientation	TcVnRotatedRectangle [▶ 237]	旋转后的矩形，包含中心点、轴长和顺时针方向的区域旋转角。

返回值

HRESULT [▶ 135]

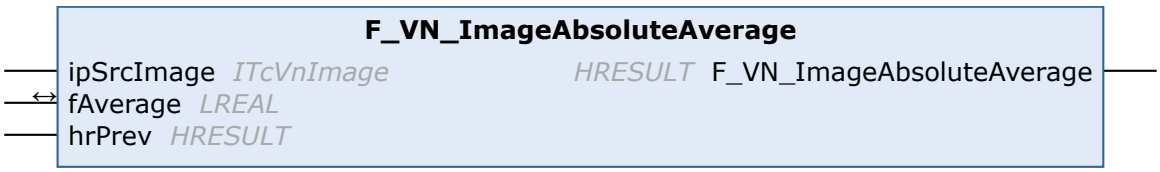
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.28 F_VN_ImageAbsoluteAverage



Computes the average absolute pixel value of an image.
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_ImageAbsoluteAverage : HRESULT
VAR_INPUT
    ipSrcImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    fAverage   : LREAL;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fAverage	LREAL	返回平均绝对像素值

返回值

HRESULT [▶ 135]

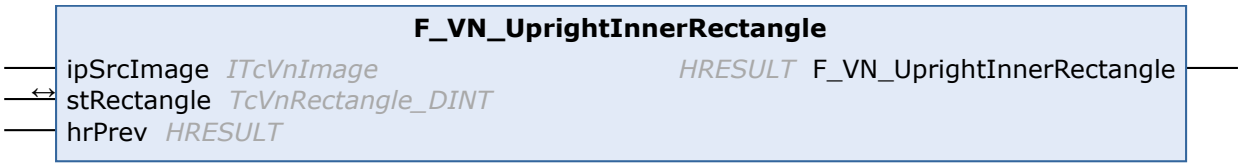
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.29 F_VN_UprightInnerRectangle



Computes the maximum area upright inner rectangle inside an area. If the valid area contains very small invalid areas or the performance is too slow, use the expert variant of this function and manually choose a suitable nDownScaleFactor.
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_UprightInnerRectangle : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
END_VAR
VAR_IN_OUT
    stRectangle : TcVnRectangle_DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道，二进制，值 > 0 标记有效区域）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stRectangle	TcVnRectangle_DINT [▶ 236]	返回直立内矩形的最大面积

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.30 F_VN_UprightInnerRectangleExp



Computes the maximum area upright inner rectangle inside an area. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_UprightInnerRectangleExp : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
END_VAR
VAR_IN_OUT
    stRectangle      : TcVnRectangle_DINT;
END_VAR
VAR_INPUT
    nDownScaleFactor : DINT;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道，二进制，值 > 0 标记有效区域）
nDownScaleFactor	DINT	为了权衡性能与精度，可以提供一个系数来降低内部使用的图像分辨率。只要有效区域中的无效区域大于 nDownScaleFactor，结果通常仍然是最优的。使用 -1 可根据 ipSrcImage 的分辨率自动选择，使用 0 可始终使用完整分辨率（速度较慢，但始终是最优结果）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stRectangle	TcVnRectangle_DINT [▶ 236]	返回直立内矩形的最大面积

返回值

HRESULT [▶ 135]

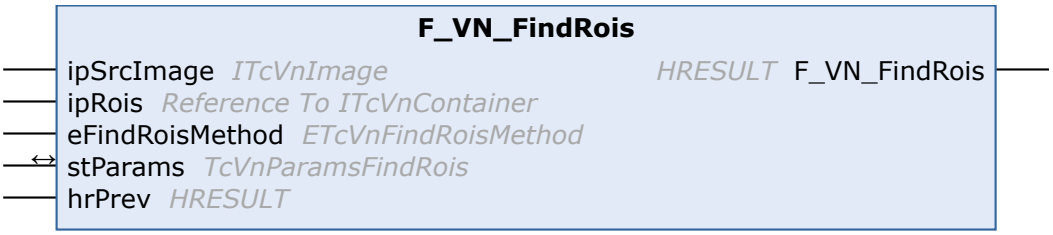
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.31 F_VN_FindRois



Extracts rectangular regions of interest (ROIs) from an image by partitioning it into tiles using the selected method and parameters.

Syntax

Definition:

```
FUNCTION F_VN_FindRois : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipRois          : Reference To ITcVnContainer;
    eFindRoisMethod : ETcVnFindRoisMethod;
END_VAR
VAR_IN_OUT
    stParams        : TcVnParamsFindRois;
END_VAR
VAR_INPUT
    hrPrev          : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipSrcImage	ITcVnImage [▶ 407]	Source image (1 channel, non-negative values).
ipRois	Reference To ITcVnContainer [▶ 365]	Returns the ROIs (ContainerType_Vector_TcVnRectangle_DINT; Non-zero interface pointers are reused.)
eFindRoisMethod	ETcVnFindRoisMethod [▶ 155]	Method to create ROIs
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

In/Outputs

Name	Type	Description
stParams	TcVnParamsFindRois [▶ 217]	Parameters for extracting ROIs

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Base

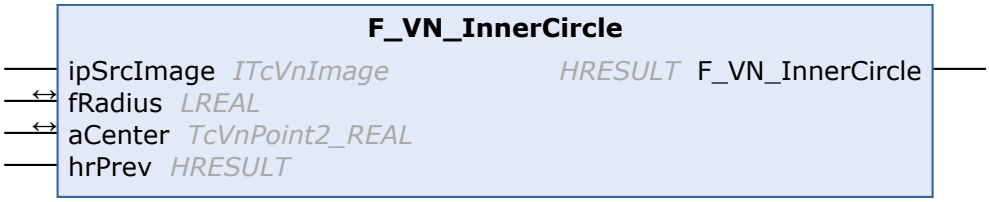
系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

还请参阅有关此

- F_VN_SalientFeatures [▶ 1243]
- ETcVnFindRoisMethod [▶ 155]
- TcVnParamsFindRois [▶ 217]

6.1.4.15.32 F_VN_InnerCircle



Calculates the position and radius of the largest inscribed circle within the white region of a binary image.

Syntax

Definition:

```
FUNCTION F_VN_InnerCircle : HRESULT
VAR_INPUT
    ipSrcImage : ITcVnImage;
END_VAR
VAR_IN_OUT
    fRadius    : LREAL;
    aCenter    : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipSrcImage	ITcVnImage [▶ 407]	Source image (USINT, 1 channel)
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

In/Outputs

Name	Type	Description
fRadius	LREAL	Returns the radius of the inner circle
aCenter	TcVnPoint2_REAL [▶ 152]	Returns the center of the inner circle

Return value

HRESULT [▶ 135]

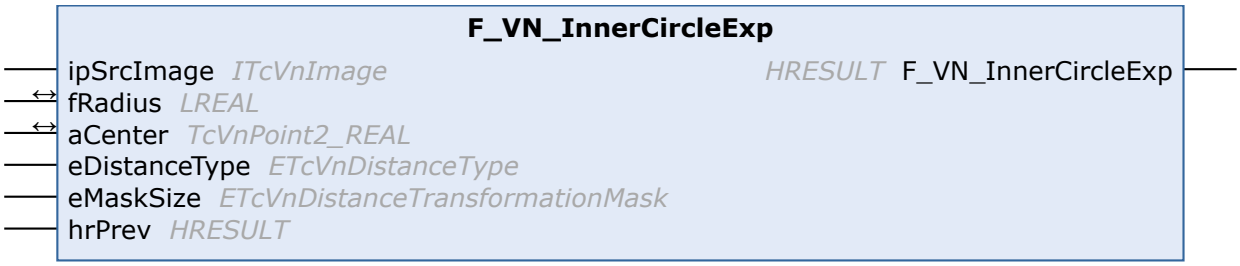
Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.33 F_VN_InnerCircleExp



Calculates the position and radius of the largest inscribed circle within the white region of a binary image. (expert function)

Syntax

Definition:

```
FUNCTION F_VN_InnerCircleExp : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
END_VAR
VAR_IN_OUT
    fRadius       : LREAL;
    aCenter       : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    eDistanceType : ETcVnDistanceType;
    eMaskSize     : ETcVnDistanceTransformationMask;
    hrPrev        : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipSrcImage	ITcVnImage [▶ 407]	Source image (USINT, 1 channel)
eDistanceType	ETcVnDistanceType [▶ 188]	Distance computation method (supported: L1, L2, C)
eMaskSize	ETcVnDistanceTransformationMask [▶ 187]	Size of the distance transformation mask
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

In/Outputs

Name	Type	Description
fRadius	LREAL	Returns the radius of the inner circle
aCenter	TcVnPoint2_REAL [▶ 152]	Returns the center of the inner circle

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.15.34 F_VN_SalientFeatures



Compute salient features from the input image using the specified method.

Syntax

Definition:

```
FUNCTION F_VN_SalientFeatures : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    eMethod     : ETcVnSalientFeaturesMethod;
    hrPrev      : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipSrcImage	ITcVnImage [▶ 407]	Source image (USINT elements, 1 or 3 channels; 3 channel input is expected to be RGB and internally converted to Gray).
ipDestImage	Reference To ITcVnImage [▶ 407]	Destination image (1 channel image of type TCVN_ET_UINT, an appropriate destination image will be created if required.)
eMethod	ETcVnSalientFeaturesMethod [▶ 156]	Method to compute salient features.
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Base

系统要求

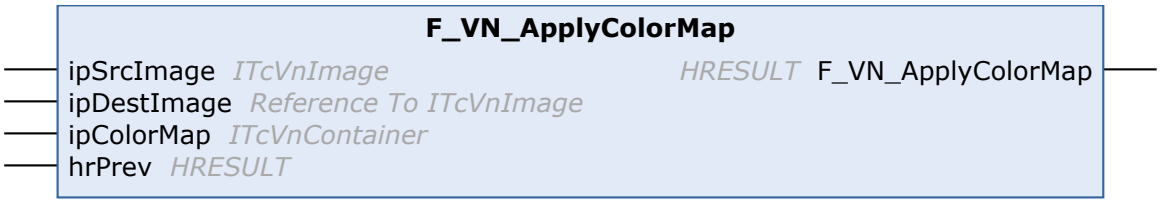
开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

还请参阅有关此

- F_VN_FindRois [▶ 1240]
- ETcVnSalientFeaturesMethod [▶ 156]
- F_VN_DetectCodesNeuralNetwork [▶ 839]

6.1.4.16 图像颜色和对比度处理

6.1.4.16.1 F_VN_ApplyColorMap



Apply a color map to a gray-level image.

语法

定义：

```
FUNCTION F_VN_ApplyColorMap : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    ipColorMap  : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（TCVN_ET_USINT 或 TCVN_ET_UINT，1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	返回彩色图像（TCVN_ET_USINT，3 通道。如有需要，将创建适当的图像）。
ipColorMap	ITcVnContainer [▶ 365]	应用于 ipSrcImage 的色图（ContainerType_Vector_TcVnVector3_REAL，包含 256 或 65536 个元素，视 ipSrcImage 类型而定）。可进行自定义，也可以通过 F_VN_GenerateColorMap 创建。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_ApplyColorMap将一个色表应用于输入图像。

参数

输入图像

输入图像ipSrcImage必须为单通道灰度图像，且其元素类型为USINT（8 位）或UINT（16 位）。

结果图像

结果图像ipDestImage返回通过色表转换的 3 通道 RGB 彩色图像，其元素类型为USINT（8 位）。

色表

色表ipColorMap定义了输入图像的转换方式。色表是一个容器，可以手动创建，也可以通过以下两种函数之一创建：

- F_VN_GenerateColorMap [▶ 1249]
- F_VN_GenerateCustomColorMap [▶ 1250]

容器必须包含 256 个元素（对于 8 位）或 65,536 个元素（对于 16 位）（取决于输入图像的元素类型），并且必须是类型ContainerType_Vector_TcVnVector3_REAL。

应用

例如，一个典型的热图在 8 位图像上的应用如下所示：

```
hr := F_VN_GenerateColorMap(ipColorMap, TCVN_CMM_HOT, TCVN_CMS_256, hr);
hr := F_VN_ApplyColorMap(
  ipSrcImage    := ipImageIn,
  ipDestImage   := ipImageRes,
  ipColorMap    := ipColorMap,
  hrPrev        := hr
);
```

样本

- 通过颜色表显示 [▶ 2784]

相关函数

- F_VN_GenerateColorMap [▶ 1249]
- F_VN_GenerateCustomColorMap [▶ 1250]
- F_VN_ApplyColorMap [▶ 1244]

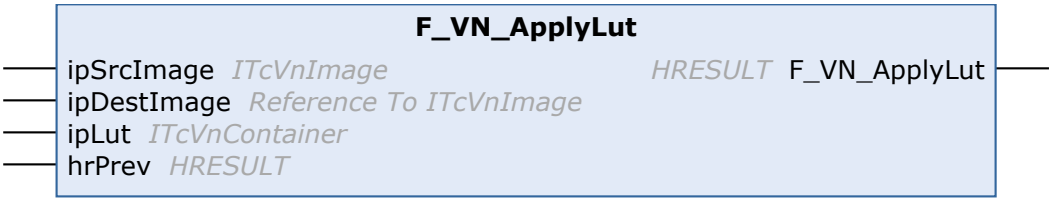
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.2 F_VN_ApplyLut



Apply a lookup table to an image to manipulate its colors.

语法

定义：

```
FUNCTION F_VN_ApplyLut : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    ipLut       : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1-4 个通道，8 或 16 位，SINT、USINT、INT、UINT 类型）
ipDestImage	引用 ITcVnImage [▶ 407]	返回结果图像（大小、类型和通道与 ipSrcImage 相同。如有需要，将创建适当的图像）。
ipLut	ITcVnContainer [▶ 365]	查找表，包含与 ipSrcImage 类型相匹配的元素，256（8 位图像）或 65536（16 位图像）个元素。例如，用于 1-4 通道 USINT 图像的 ContainerType_Vector_USINT（在这种情况下，每个通道使用相同的查找值）或用于 3 通道 USINT 图像的 ContainerType_TcVnVector3_USINT（在这种情况下，每个通道使用单独的查找值）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.3 F_VN_Clahe



应用限制对比度自适应性直方图均衡（CLAHE）。对单个平铺图应用局部直方图均衡。
可以使用可用的 TwinCAT 任务并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_Clahe : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（TCVN_ET_USINT 或 TCVN_ET_UINT，1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	返回结果图像（如果需要，将创建合适的图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.4 F_VN_ClaheExp



应用限制对比度自适应性直方图均衡（CLAHE）。对单个平铺图应用局部直方图均衡。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_ClaheExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    fClipLimit  : LREAL;
    nTilesX     : UDINT;
    nTilesY     : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（TCVN_ET_USINT 或 TCVN_ET_UINT，1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	返回结果图像（如果需要，将创建合适的图像。）
fClipLimit	LREAL	对比度限制阈值（设置为 <= 0 时禁用裁剪）
nTilesX	UDINT	x 方向的平铺图数量
nTilesY	UDINT	y 方向的平铺图数量

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

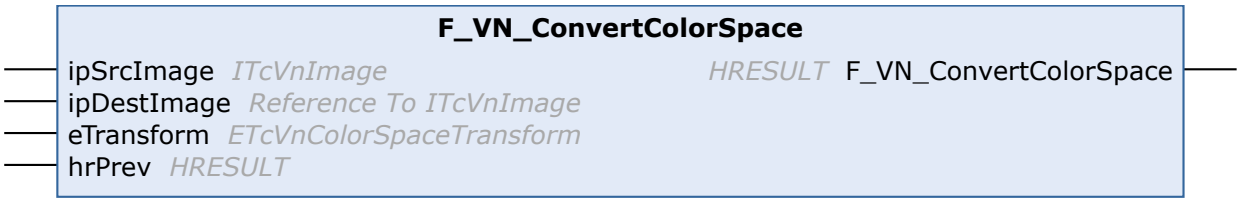
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.5 F_VN_ConvertColorSpace



Convert image from one color space to another.
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_ConvertColorSpace : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    eTransform  : ETcVnColorSpaceTransform;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eTransform	ETcVnColorSpaceTransform [▶ 167]	要应用的变换
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

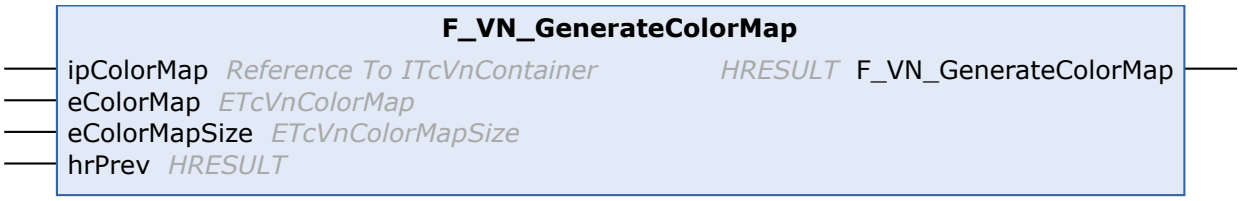
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.6 F_VN_GenerateColorMap



Generates a pre-defined color map.

语法

定义：

```
FUNCTION F_VN_GenerateColorMap : HRESULT
VAR_INPUT
    ipColorMap    : Reference To ITcVnContainer;
    eColorMap     : ETcVnColorMap;
    eColorMapSize : ETcVnColorMapSize;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipColorMap	引用 ITcVnContainer [▶ 365]	返回色图（ContainerType_Vector_TcVnVector3_REAL，包含 256 或 65536 个元素）
eColorMap	ETcVnColorMap [▶ 165]	选择色图（类似于 GNU Octave/MATLAB 型）
eColorMapSize	ETcVnColorMapSize [▶ 166]	定义生成的色图应包含多少个元素
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数`F_VN_GenerateColorMap`创建一个可以通过函数F_VN_ApplyColorMap [▶ 1244]应用到图像上色表。

参数

色图

参数`ipColorMap`返回创建的色表。

颜色梯度

色表的颜色梯度`eColorMap`通过枚举ETcVnColorMap [▶ 165]定义。如果想要手动定义颜色梯度，请使用函数F_VN_GenerateCustomColorMap [▶ 1250]。

色表尺寸

大小`eColorMapSize`通过枚举ETcVnColorMapSize [▶ 166]指定，并定义是否为 8 位图像（`TCVN_CMS_256`）或 16 位图像（`TCVN_CMS_65536`）创建色表。

应用

```
VAR
    ipColorMap      :   ITcVnColorMap;
END_VAR

hr := F_VN_GenerateColorMap(
    ipColorMap      :=  ipColorMap,
    eColorMap       :=  TCVN_CM_HOT,
    eColorMapSize   :=  TCVN_CMS_256,
    hrPrev          :=  hr
);
```

样本

- 通过颜色表显示 [▶ 2784]

相关函数

- F_VN_GenerateColorMap [▶ 1249]
- F_VN_GenerateCustomColorMap [▶ 1250]
- F_VN_ApplyColorMap [▶ 1244]

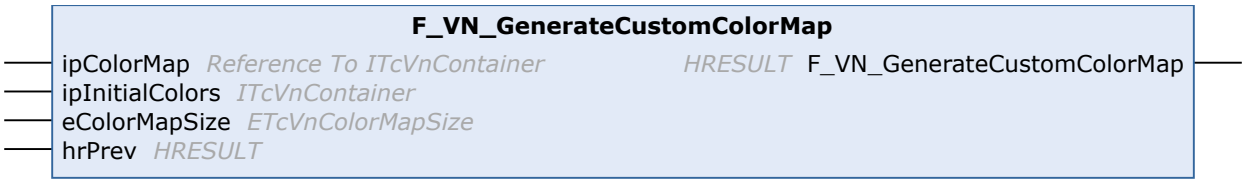
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.7 F_VN_GenerateCustomColorMap



Generate a custom color map by interpolating between user defined colors (equally distributed, linear for each channel).

语法

定义：

```
FUNCTION F_VN_GenerateCustomColorMap : HRESULT
VAR_INPUT
    ipColorMap      : Reference To ITcVnContainer;
    ipInitialColors : ITcVnContainer;
    eColorMapSize   : ETcVnColorMapSize;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipColorMap	引用 ITcVnContainer [▶ 365]	返回色图（ContainerType_Vector_TcVnVector3_REAL，包含 256 或 65536 个元素）
ipInitialColors	ITcVnContainer [▶ 365]	用户自定义颜色（ContainerType_Vector_TcVnVector3_REAL，至少 2 个元素）
eColorMapSize	ETcVnColorMapSize [▶ 166]	定义生成的色图应包含多少个元素
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

样本

- 通过颜色表显示 [▶ 2784]

相关函数

- F_VN_GenerateColorMap [▶ 1249]
- F_VN_GenerateCustomColorMap [▶ 1250]
- F_VN_ApplyColorMap [▶ 1244]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.8 F_VN_Histogram



计算图像的（多通道）直方图。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_Histogram : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipDestHistogram : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestHistogram	引用 ITcVnContainer [▶ 365]	返回包含多通道直方图的容器，其中每个通道都以 UDINT 向量表示（ContainerType_Vector_Vector_UDINT。重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

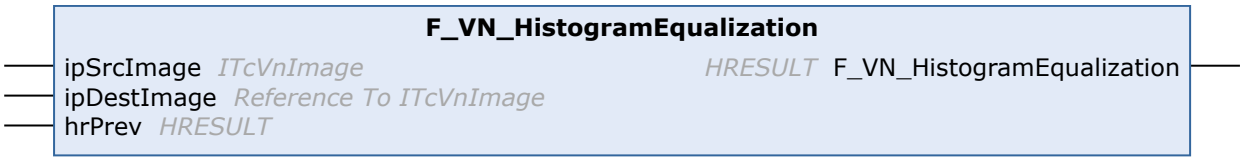
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.16.9 F_VN_HistogramEqualization



Equalize the histogram of a grayscale or rgb image, which normalizes the brightness and improves the contrast.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_HistogramEqualization : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

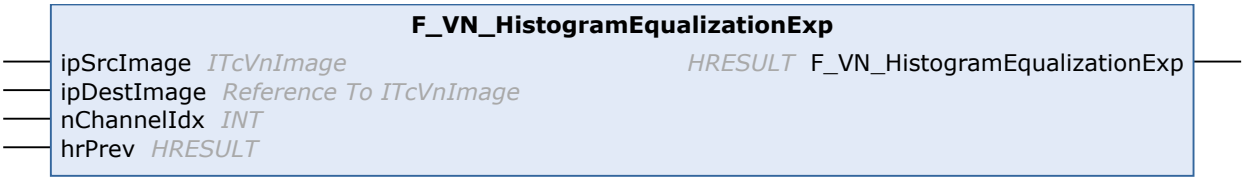
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.10 F_VN_HistogramEqualizationExp



Equalize the histogram of a grayscale or color image, which normalizes the brightness and improves the contrast. (expert function) The channel index that should be equalized has to be specified (-1 expects a RGB image, converts it to YCbCr, equalizes the Y channel and converts the image back to RGB).
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_HistogramEqualizationExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    nChannelIdx : INT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
nChannelIdx	INT	应均衡化的图像通道的索引（-1 应为 RGB 图像，将其转换为 YCbCr，均衡 Y 通道并将图像转换回 RGB）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.11 F_VN_HistogramExp



Calculate the (multi-channel) histogram of an image. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义:

```
FUNCTION F_VN_HistogramExp : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipDestHistogram : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    nBins           : UDINT;
    fLowerBound     : LREAL;
    fUpperBound     : LREAL;
END_VAR
VAR_INPUT
    ipMask          : ITcVnImage;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestHistogram	引用 ITcVnContainer [▶ 365]	返回包含多通道直方图的容器，其中每个通道都以 UDINT 向量表示（ContainerType_Vector_Vector_UDINT。重复使用非零接口指针。）
ipMask	ITcVnImage [▶ 407]	源图像掩膜（TCVN_ET_USINT，1 通道。掩膜为可选项，如果不需要，则设为 0。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nBins	UDINT	所需的分组数量，或用 0 保持相应图像格式的默认值（输入）和默认分组数量（输出）
fLowerBound	LREAL	第 0 个直方图分组的下边界（包含）（输入），如果 fLowerBound 与 fUpperBound 都设置为 0，则接收默认值（输出）
fUpperBound	LREAL	最后一个直方图分组 nBins-1 的上边界（不包含）（输入），如果 fLowerBound 与 fUpperBound 设置为 0，则接收默认值（输出）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_HistogramExp是F_VN_Histogram [▶ 1251]的专家版本。它包含额外的参数。

参数

输入图像

输入图像ipSrcImage可以有任何格式。

直方图

计算出的多通道直方图将作为类型ContainerType_Vector_Vector_UDINT的容器ipDestHistogram返回。每个子元素对应于一个通道的直方图。直方图的每个容器均以UDINT表示，其中指定了属于该容器的像素数量。

容器

数字nBins定义了观察到的像素光谱要被分成多少个容器。

限制

下限fLowerBound和上限fUpperBound定义了在创建直方图时哪些像素值被考虑在内。

掩码

掩码ipMask定义了在计算直方图时输入图像ipSrcImage的哪些像素被考虑在内。它由一个元素类型为USINT的单通道图像描述，其大小与输入图像ipSrcImage相同。在计算直方图时，掩码图像中所有数值为<> 0的像素都被考虑在内。

应用

例如，一个有 10 个容器的直方图，数值范围为 100-200，有一个圆形掩码，其计算如下：

```
VAR
    ipImageMask :   ITcVnImage;
END_VAR

// Create mask image with a circle at image center.
hr := F_VN_CopyImage(ipImageIn, ipImageMask, hr);
hr := F_VN_SetPixels(ipImageMask, aBlack, hr);
hr := F_VN_DrawCircle(400, 300, 300, ipImageMask, aWhite, -1, hr);

hr := F_VN_HistogramExp(
    ipSrcImage      :=  ipImageIn,
    ipDestHistogram :=  ipHistograms,
  nBins      := 10,
  fLowerBound   := 100,
  fUpperBound   := 201,
  ipMask      := ipImageMask,
    hrPrev          :=  hr
);
```

样本

- 处理直方图

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.12 F_VN_InvertImageColor



Invert the color of an image. For signed integer or negative floating point values, only the signs will be switched. If the image contains only positive floating point values, each pixel value is subtracted from the maximum available pixel value (or 1.0, whatever is higher).

语法

定义：

```
FUNCTION F_VN_InvertImageColor : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

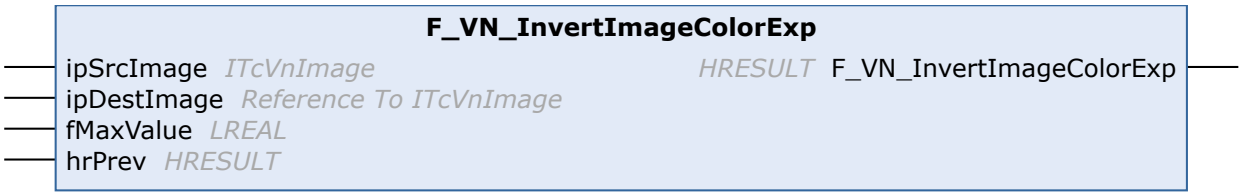
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.13 F_VN_InvertImageColorExp



Invert the color of an image. (expert function) For signed integer or negative floating point values, only the signs will be switched. If the image contains only positive floating point values, each pixel value is subtracted from the maximum available pixel value (or 1.0, whatever is higher).

语法

定义：

```
FUNCTION F_VN_InvertImageColorExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    fMaxValue   : LREAL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的图像。）
fMaxValue	LREAL	最大像素值（例如，如果 16 位图像包含 12 位值）。-1 表示使用默认值。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

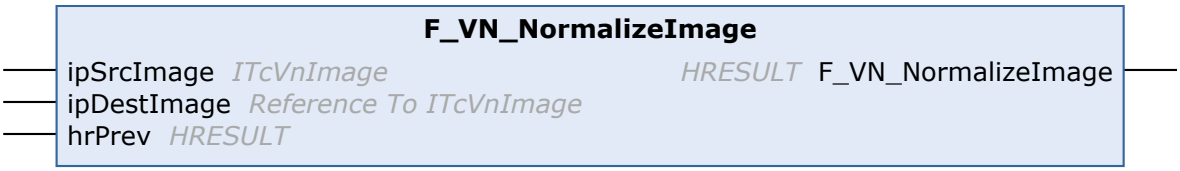
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.14 F_VN_NormalizeImage



Normalize an image regarding its value range (e.g. stretch pixel values [50..150] to full range [0..255]).

语法

定义：

```
FUNCTION F_VN_NormalizeImage : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

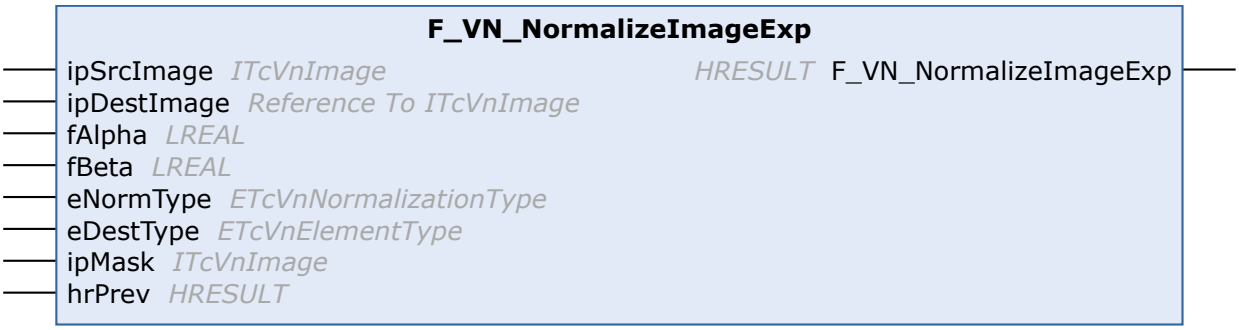
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.15 F_VN_NormalizeImageExp



Normalize an image regarding its value range (e.g. stretch pixel values [50..150] to full range [0..255]) or scale the values regarding a specific normalization (e.g. L2-norm). (expert function)

语法

定义：

```
FUNCTION F_VN_NormalizeImageExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    fAlpha      : LREAL;
    fBeta       : LREAL;
    eNormType   : ETcVnNormalizationType;
    eDestType   : ETcVnElementType;
    ipMask      : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
fAlpha	LREAL	范围的下边界（在对值范围进行归一化时）或要归一化为的数值（||ipDestImage|| = fAlpha）
fBeta	LREAL	范围的上边界（在对值范围进行归一化时）
eNormType	ETcVnNormalizationType [▶ 203]	归一化类型（仅限 INF、L1、L2 或 MINMAX）
eDestType	ETcVnElementType [▶ 191]	目标图像深度（通常为 SAME_AS_SOURCE）
ipMask	ITcVnImage [▶ 407]	用于将归一化限制在特定像素位置的掩膜（设为 0 则对整个图像进行归一化）

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

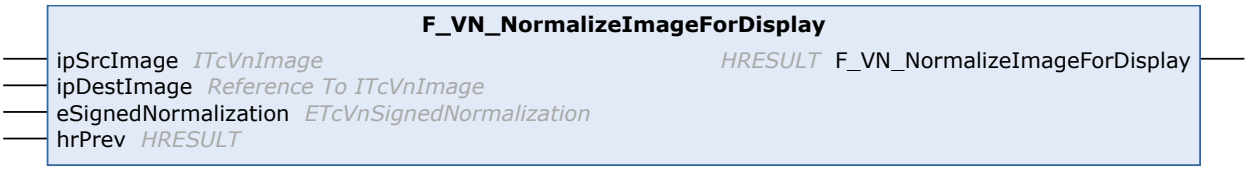
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.16 F_VN_NormalizeImageForDisplay



Normalize an image for display, i.e. scale it to the full value range of the underlying data type (-1 to 1 for floating point).
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_NormalizeImageForDisplay : HRESULT
VAR_INPUT
    ipSrcImage           : ITcVnImage;
    ipDestImage          : Reference To ITcVnImage;
    eSignedNormalization : ETcVnSignedNormalization;
    hrPrev               : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eSignedNormalization	ETcVnSignedNormalization [▶ 209]	可对带符号的值进行归一化
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

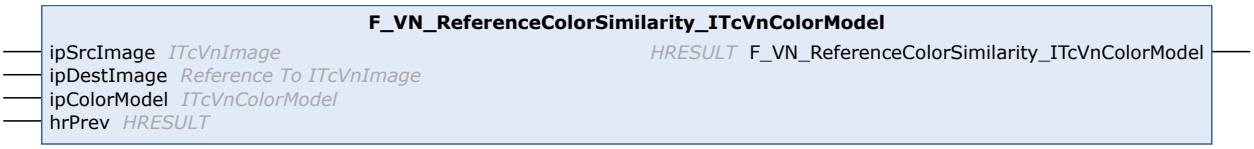
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.17 F_VN_ReferenceColorSimilarity_ITcVnColorModel



Computes the similarity to a reference color model for each pixel in the source image.
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_ReferenceColorSimilarity_ITcVnColorModel : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDestImage   : Reference To ITcVnImage;
    ipColorModel  : ITcVnColorModel;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	RGB 源图像（USINT，3 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	返回 ipSrcImage 中每个像素与 ipColorModel 的相似度（USINT，1 通道。如有需要，将创建适当的目标图像。）
ipColorModel	ITcVnColorModel [▶ 415]	颜色模型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

有三种方法可以将参考颜色传递给相应的函数：

- 作为一个经过训练的颜色模型，类型`ITcVnColorModel`
- 作为一个经过训练的颜色模型，类型`ITcVnMlModel`
- 直接通过`TcVnVector3_LREAL`中的数值， 见样本

如需训练一个颜色模型，必须使用相应的F_VN_TrainImageColor [▶ 1265]函数。这两种颜色模型仅在数据类型上有所区别，而在功能上没有区别。`ITcVnMlModel`是 ML 模型的一般数据类型，对其也有加载和保存的功能块。因此，如果需要将训练好的颜色模型保存到磁盘上，应该使用这种数据类型和相应的函数。

样本

- 与 RGB 参考颜色的颜色相似度 [▶ 2781]

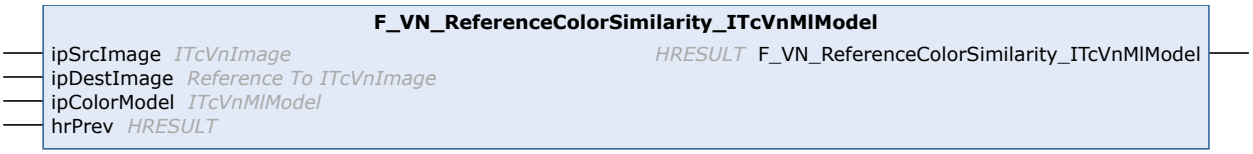
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.18 F_VN_ReferenceColorSimilarity_ITcVnMlModel



计算源图像中每个像素与参考颜色模型的相似度。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_ReferenceColorSimilarity_ITcVnMlModel : HRESULT
VAR_INPUT
    ipSrcImage   : ITcVnImage;
    ipDestImage  : Reference To ITcVnImage;
    ipColorModel : ITcVnMlModel;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	RGB 源图像（USINT，3 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	返回 ipSrcImage 中每个像素与 ipColorModel 的相似度（USINT，1 通道。如有需要，将创建适当的目标图像。）
ipColorModel	ITcVnMlModel [▶ 417]	颜色模型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

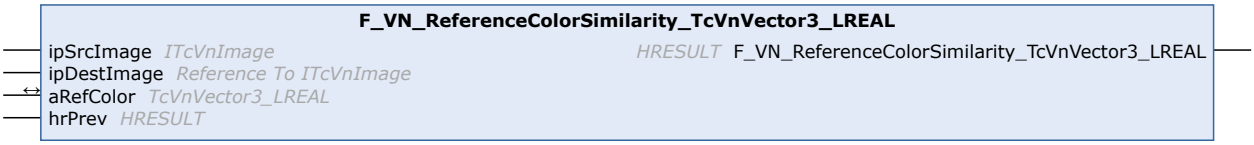
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.16.19 F_VN_ReferenceColorSimilarity_TcVnVector3_LREAL



计算源图像中每个像素与参考颜色的相似度。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_ReferenceColorSimilarity_TcVnVector3_LREAL : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aRefColor   : TcVnVector3_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	RGB 源图像（USINT，3 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	返回 ipSrcImage 中每个像素与 aRefColor 的相似度（USINT，1 通道。如有需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aRefColor	TcVnVector3_LREAL [▶ 153]	参考颜色（RGB，[0...255]）

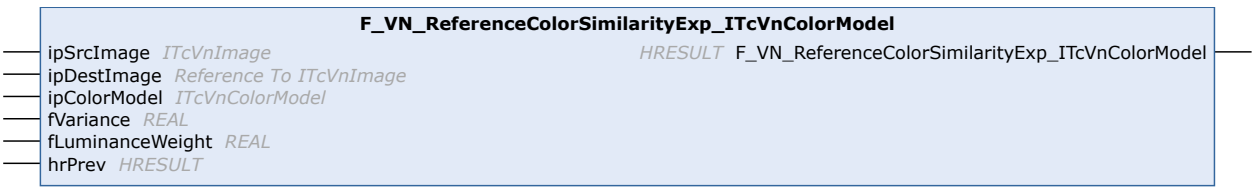
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.16.20 F_VN_ReferenceColorSimilarityExp_ITcVnColorModel



计算源图像中每个像素与参考颜色模型的相似度。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_ReferenceColorSimilarityExp_ITcVnColorModel : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
    ipDestImage      : Reference To ITcVnImage;
    ipColorModel     : ITcVnColorModel;
    fVariance        : REAL;
    fLuminanceWeight : REAL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	RGB 源图像（USINT，3 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	返回 ipSrcImage 中每个像素与 ipColorModel 的相似度（USINT，1 通道。如有需要，将创建适当的目标图像。）
ipColorModel	ITcVnColorModel [▶ 415]	颜色模型
fVariance	REAL	允许的颜色方差（0.1 - 0.3 可能是一个不错的尝试起点）
fLuminanceWeight	REAL	对亮度的影响进行加权（[0...1]，例如，设置为 0 更能抵御不均匀光照的影响，但可能需要用于区分某些颜色。如果 ipColorModel 为 RGB 类型，则忽略。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.16.21 F_VN_ReferenceColorSimilarityExp_ITcVnMlModel



计算源图像中每个像素与参考颜色模型的相似度。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_ReferenceColorSimilarityExp_ITcVnMlModel : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
    ipDestImage      : Reference To ITcVnImage;
    ipColorModel     : ITcVnMlModel;
    fVariance        : REAL;
    fLuminanceWeight : REAL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	RGB 源图像（USINT，3 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	返回 ipSrcImage 中每个像素与 ipColorModel 的相似度（USINT，1 通道。如有需要，将创建适当的目标图像。）
ipColorModel	ITcVnMlModel [▶ 417]	颜色模型
fVariance	REAL	允许的颜色方差（0.1 - 0.3 可能是一个不错的尝试起点）
fLuminanceWeight	REAL	对亮度的影响进行加权（[0...1]，例如，设置为 0 更能抵御不均匀光照的影响，但可能需要用于区分某些颜色。如果 ipColorModel 为 RGB 类型，则忽略。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.16.22 F_VN_ReferenceColorSimilarityExp_TcVnVector3_LREAL



计算源图像中每个像素与参考颜色的相似度。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_ReferenceColorSimilarityExp_TcVnVector3_LREAL : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
    ipDestImage      : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aRefColor        : TcVnVector3_LREAL;
END_VAR
VAR_INPUT
    fVariance        : REAL;
    fLuminanceWeight : REAL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	RGB 源图像（USINT，3 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	返回 ipSrcImage 中每个像素与 aRefColor 的相似度（USINT，1 通道。如有需要，将创建适当的目标图像。）
fVariance	REAL	允许的颜色方差（0.1 - 0.3 可能是一个不错的尝试起点）

名称	类型	描述
fLuminanceWeight	REAL	对亮度的影响进行加权（[0...1]，例如，设置为 0 更能抵御不均匀光照的影响，但可能需要用于区分某些颜色）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aRefColor	TcVnVector3_LREAL [▶ 153]	参考颜色（RGB，[0...255]）

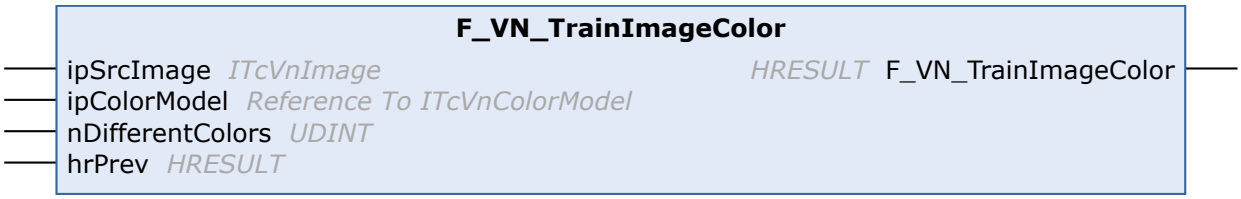
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.16.23 F_VN_TrainImageColor



Create a new color model, describing the image color.
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_TrainImageColor : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
    ipColorModel     : Reference To ITcVnColorModel;
    nDifferentColors : UDINT;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（3 通道（RGB），USINT 类型）
ipColorModel	引用 ITcVnColorModel [▶ 415]	返回颜色模型
nDifferentColors	UDINT	要区分的不同颜色数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_TrainImageColor根据参考图像训练颜色模型。如果要将训练好的颜色模型保存到磁盘，必须使用类型为ITcVnMlModel的F_VN_TrainImageColor_ITcVnMlModel [▶ 1267]。

参数

参考图像

参考图像ipSrcImage必须有 3 个 RGB 格式的通道，且元素类型为USINT（8 位）。

颜色模型

参数ipColorModel返回训练的颜色模型。这是一个类型ITcVnColorModel [▶ 415]的接口指针。

需要区分的颜色数量

数字nDifferentColors定义了需要训练的颜色数量。请注意，图像背景也可以有自己的颜色。

专家参数

关于更多参数，可查看专家版F_VN_TrainImageColorExp [▶ 1268]。

应用

例如，对于包括 4 个不同颜色物体以及深色背景的参考图像，进行颜色模型的训练如下所示：

```
VAR
    ipColorModel    :   ITcVnColorModel;
END_VAR

hr := F_VN_TrainImageColor(
            ipSrcImage      := ipImageRef,
            ipColorModel    := ipColorModel,
            nDifferentColors:= 5, // 4 colors + 1 background = 5
            hrPrev          := hr);
```

相关函数

有三种方法可以将参考颜色传递给相应的F_VN_ReferenceColorSimilarity函数：

- 作为一个经过训练的颜色模型，类型ITcVnColorModel
- 作为一个经过训练的颜色模型，类型ITcVnMlModel
- 直接使用TcVnVector3_LREAL中的数值

对于颜色模型，有相应的函数用于训练，而对于所有三种数据类型，各有一个函数用于执行：

- F_VN_TrainImageColor [▶ 1265]用于训练一个颜色模型
- F_VN_ReferenceColorSimilarity [▶ 1260]通过颜色模型进行分割

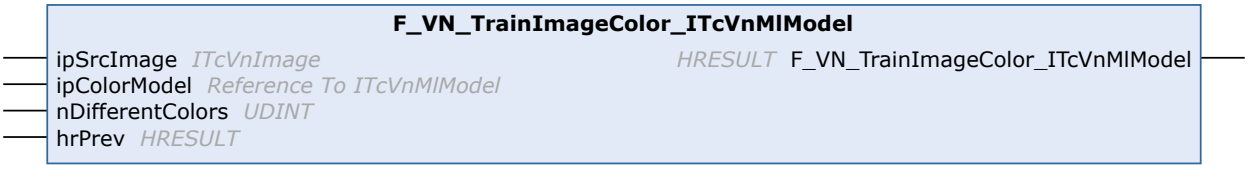
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.24 F_VN_TrainImageColor_ITcVnMlModel



Create a new color model, describing the image color.
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_TrainImageColor_ITcVnMlModel : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
    ipColorModel     : Reference To ITcVnMlModel;
    nDifferentColors : UDINT;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（3 通道（RGB），USINT 类型）
ipColorModel	引用 ITcVnMlModel [▶ 417]	返回颜色模型
nDifferentColors	UDINT	要区分的不同颜色数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_TrainImageColor_ITcVnMlModel根据参考图像训练颜色模型。通过相应的功能块，颜色模型可以在硬盘上保存和加载。

参数

参考图像

参考图像ipSrcImage必须有 3 个 RGB 格式的通道，且元素类型为USINT（8 位）。

颜色模型

参数ipColorModel返回训练的颜色模型。这是一个接口指针，类型为ITcVnMlModel [▶ 417]。

需要区分的颜色数量

数字nDifferentColors定义了需要训练的颜色数量。请注意，图像背景也可以有自己的颜色。

专家参数

关于更多参数，可查看专家版F_VN_TrainImageColorExp_ITcVnMlModel [▶ 1270]。

应用

例如，对于包括 4 个不同颜色物体以及深色背景的参考图像，进行颜色模型的训练如下所示：

```
VAR
    ipColorModel    :   ITcVnMlModel;
END_VAR

hr := F_VN_TrainImageColor_ITcVnMlModel(
            ipSrcImage      := ipImageRef,
            ipColorModel    := ipColorModel,
            nDifferentColors:= 5, // 4 colors + 1 background = 5
            hrPrev          := hr);
```

相关函数

有三种方法可以将参考颜色传递给相应的F_VN_ReferenceColorSimilarity函数：

- 作为一个经过训练的颜色模型，类型ITcVnColorModel
- 作为一个经过训练的颜色模型，类型ITcVnMlModel
- 直接使用TcVnVector3_LREAL中的数值

对于颜色模型，有相应的函数用于训练，而对于所有三种数据类型，各有一个函数用于执行：

- F_VN_TrainImageColor [▶ 1265]用于训练一个颜色模型
- F_VN_ReferenceColorSimilarity [▶ 1260]通过颜色模型进行分割

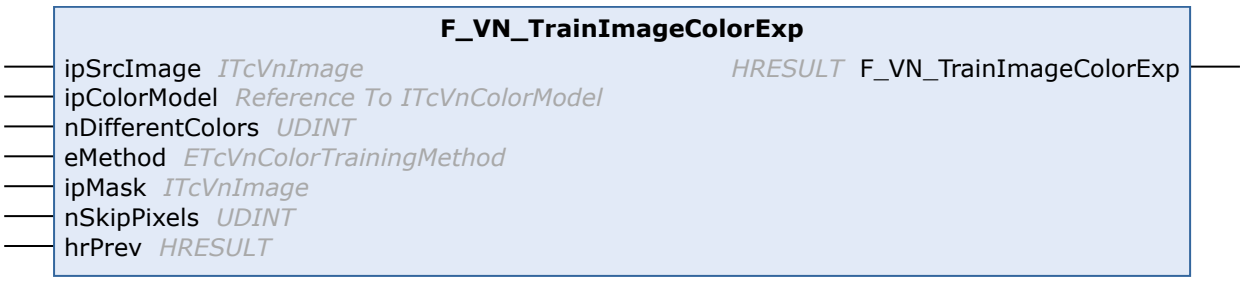
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.25 F_VN_TrainImageColorExp



Create a new color model, describing the image color. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_TrainImageColorExp : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
    ipColorModel     : Reference To ITcVnColorModel;
    nDifferentColors : UDINT;
    eMethod          : ETcVnColorTrainingMethod;
    ipMask           : ITcVnImage;
    nSkipPixels      : UDINT;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（3 通道（RGB），USINT 类型）
ipColorModel	引用 ITcVnColorModel [▶ 415]	返回颜色模型
nDifferentColors	UDINT	要区分的不同颜色数量
eMethod	ETcVnColorTrainingMethod [▶ 177]	色彩训练方法
ipMask	ITcVnImage [▶ 407]	可选的图像掩膜（1 通道，USINT 类型，如果不需要，则设为 0）
nSkipPixels	UDINT	在每个评估的颜色样本之间跳过的像素数量（以获得更好的性能）。0 将每个像素都考虑在内，往往更准确。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

该函数F_VN_TrainImageColorExp是F_VN_TrainImageColor [▶ 1265]的专家版本。它包含进一步的参数。如果需要将训练好的颜色模型保存到磁盘，必须使用类型ITcVnMlModel的F_VN_TrainImageColorExp_ITcVnMlModel [▶ 1270]。

参数

参考图像

参考图像ipSrcImage必须有 3 个 RGB 格式的通道，且元素类型为USINT（8 位）。

颜色模型

参数ipColorModel返回训练的颜色模型。这是一个类型ITcVnColorModel [▶ 415]的接口指针。

需要区分的颜色数量

数字nDifferentColors定义了需要训练的颜色数量。请注意，图像背景也可以有自己的颜色。

专家参数

关于其他参数，可查看另一专家版本F_VN_TrainImageColorExp2 [▶ 1272]。

培训方法

在选择训练方法时，可以选择 LAB 和 RGB 颜色空间。

掩码

掩码ipMask定义了输入图像ipSrcImage的哪些像素在计算中被考虑进去。掩码由一个元素类型为 USINT 的单通道图像描述，其大小与输入图像ipSrcImage相同。在计算集群时，掩码图像中所有数值 <>0 的像素都被考虑在内。

像素差距

为了优化性能，可以跳过一些像素，这样在划分各个集群时需要考虑的像素就会减少。

应用

```
VAR
    ipColorModel    :   ITcVnColorModel;
END_VAR

hr := F_VN_TrainImageColor(
            ipSrcImage      :=  ipImageRef,
            ipColorModel    :=  ipColorModel,
            nDifferentColors:=  5, // 4 colors + 1 background = 5
          eMethod     := TCVN_CTM_LAB,
          ipMask      := 0,
          nSkipPixels   := 3,
            hrPrev          :=  hr);
```

相关函数

有三种方法可以将参考颜色传递给相应的F_VN_ReferenceColorSimilarity函数：

- 作为一个经过训练的颜色模型，类型ITcVnColorModel
- 作为一个经过训练的颜色模型，类型ITcVnMlModel
- 直接使用TcVnVector3_LREAL中的数值

对于颜色模型，有相应的函数用于训练，而对于所有三种数据类型，各有一个函数用于执行：

- F_VN_TrainImageColor [▶ 1265]用于训练一个颜色模型
- F_VN_ReferenceColorSimilarity [▶ 1260]通过颜色模型进行分割

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.26 F_VN_TrainImageColorExp_ITcVnMlModel



Create a new color model, describing the image color. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_TrainImageColorExp_ITcVnMlModel : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
    ipColorModel     : Reference To ITcVnMlModel;
    nDifferentColors : UDINT;
    eMethod          : ETcVnColorTrainingMethod;
    ipMask           : ITcVnImage;
```

```
    nSkipPixels      : UDINT;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（3 通道（RGB），USINT 类型）
ipColorModel	引用 ITcVnMlModel [▶ 417]	返回颜色模型
nDifferentColors	UDINT	要区分的不同颜色数量
eMethod	ETcVnColorTrainingMethod [▶ 177]	色彩训练方法
ipMask	ITcVnImage [▶ 407]	可选的图像掩膜（1 通道，USINT 类型，如果不需要，则设为 0）
nSkipPixels	UDINT	在每个评估的颜色样本之间跳过的像素数量（以获得更好的性能）。0 将每个像素都考虑在内，往往更准确。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_TrainImageColorExp_ITcVnMlModel是F_VN_TrainImageColor_ITcVnMlModel [▶ 1267]的专家版。它包含进一步的参数。通过相应的功能块，颜色模型可以在硬盘上保存和加载。

参数

参考图像

参考图像ipSrcImage必须有 3 个 RGB 格式的通道，且元素类型为USINT（8 位）。

颜色模型

参数ipColorModel返回训练的颜色模型。这是一个接口指针，类型为ITcVnMlModel [▶ 417]。

需要区分的颜色数量

数字nDifferentColors定义了需要训练的颜色数量。请注意，图像背景也可以有自己的颜色。

专家参数

关于其他参数，可查看另一专家版本F_VN_TrainImageColorExp2_ITcVnMlModel [▶ 1275]。

培训方法

在选择训练方法时，可以选择 LAB 和 RGB 颜色空间。

掩码

掩码ipMask定义了输入图像ipSrcImage的哪些像素在计算中被考虑进去。掩码由一个元素类型为 USINT 的单通道图像描述，其大小与输入图像ipSrcImage相同。在计算集群时，掩码图像中所有数值 <>0 的像素都被考虑在内。

像素差距

为了优化性能，可以跳过一些像素，这样在划分各个集群时需要考虑的像素就会减少。

应用

```
VAR
    ipColorModel    :   ITcVnMlModel;
END_VAR

hr := F_VN_TrainImageColorExp_ITcVnMlModel (
            ipSrcImage      := ipImageRef,
            ipColorModel    := ipColorModel,
            nDifferentColors:= 5, // 4 colors + 1 background = 5
          eMethod     := TCVN_CTM_LAB,
          ipMask      := 0,
          nSkipPixels   := 3,
            hrPrev          := hr);
```

相关函数

有三种方法可以将参考颜色传递给相应的F_VN_ReferenceColorSimilarity函数：

- 作为一个经过训练的颜色模型，类型ITcVnColorModel
- 作为一个经过训练的颜色模型，类型ITcVnMlModel
- 直接使用TcVnVector3_LREAL中的数值

对于颜色模型，有相应的函数用于训练，而对于所有三种数据类型，各有一个函数用于执行：

- F_VN_TrainImageColor [▶ 1265]用于训练一个颜色模型
- F_VN_ReferenceColorSimilarity [▶ 1260]通过颜色模型进行分割

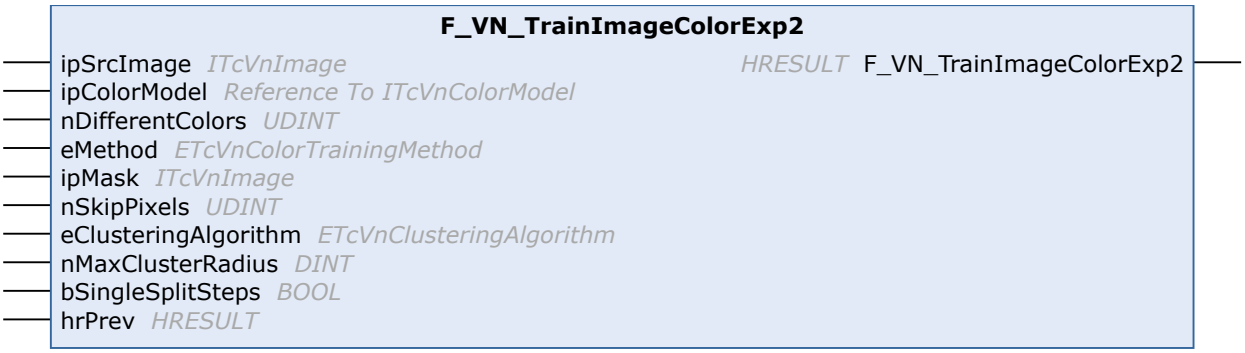
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.27 F_VN_TrainImageColorExp2



Create a new color model, describing the image color. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_TrainImageColorExp2 : HRESULT
VAR_INPUT
    ipSrcImage           : ITcVnImage;
    ipColorModel         : Reference To ITcVnColorModel;
    nDifferentColors     : UDINT;
    eMethod              : ETcVnColorTrainingMethod;
    ipMask               : ITcVnImage;
    nSkipPixels          : UDINT;
    eClusteringAlgorithm : ETcVnClusteringAlgorithm;
    nMaxClusterRadius    : DINT;
    bSingleSplitSteps    : BOOL;
    hrPrev               : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（3 通道（RGB），USINT 类型）
ipColorModel	引用 ITcVnColorModel [▶ 415]	返回颜色模型
nDifferentColors	UDINT	要区分的不同颜色的最大数量（如果使用 LBG 作为聚类算法，则结果中的不同颜色数量可能较少，具体取决于 nMaxClusterRadius）
eMethod	ETcVnColorTrainingMethod [▶ 177]	色彩训练方法
ipMask	ITcVnImage [▶ 407]	可选的图像掩膜（1 通道，USINT 类型，如果不需要，则设为 0）
nSkipPixels	UDINT	在每个评估的颜色样本之间跳过的像素数量（以获得更好的性能）。0 将每个像素都考虑在内，往往更准确。
eClusteringAlgorithm	ETcVnClusteringAlgorithm [▶ 163]	聚类算法
nMaxClusterRadius	DINT	仅用于 LBG 聚类算法。单个聚类允许的最大半径（> 0），即聚类半径越大，被分割成的聚类越小，直到达到 nDifferentColors 的全局数量。
bSingleSplitSteps	BOOL	仅用于 LBG 聚类算法。如果为 true，则总是在拆分单个聚类后运行全局优化。如果为 false，则会在应用全局优化之前，在同一步骤中拆分多个聚类。减少全局优化的应用次数会加快速度，但可能会导致结果不理想，尤其是有 2 个可由 1 表示的邻近聚类时
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

该函数`F_VN_TrainImageColorExp2`是F_VN_TrainImageColorExp [▶ 1268]的扩展专家版本。它包含进一步的参数。如果需要将训练好的颜色模型保存到磁盘，必须使用类型`ITcVnMlModel`的F_VN_TrainImageColorExp2_ITcVnMlModel [▶ 1275]。

参数

参考图像

参考图像`ipSrcImage`必须有 3 个 RGB 格式的通道，且元素类型为`USINT`（8 位）。

颜色模型

参数ipColorModel返回训练的颜色模型。这是一个类型ITcVnColorModel [▶ 415]的接口指针。

需要区分的颜色数量

数字nDifferentColors定义了需要训练的颜色数量。请注意，图像背景也可以有自己的颜色。

集群算法

在选择集群算法时，有 KMeans++（固定集群数量）和 LBG（动态集群数量）可供选择。

最大集群半径

仅与 LBG 集群算法一起使用。这可以用来影响各个集群的分布。

全球优化

仅与 LBG 集群算法一起使用。这允许在更短的执行时间（bSingleSplitSteps = FALSE）和更好的结果（bSingleSplitSteps = TRUE）之间做出选择。

更少使用全局优化会更快，但可能导致不太理想的结果。

应用

```
VAR
    ipColorModel    :   ITcVnColorModel;
END_VAR

hr := F_VN_TrainImageColorExp2(
            ipSrcImage          := ipImageRef,
            ipColorModel        := ipColorModel,
            nDifferentColors    := 5, // 4 colors + 1 background = 5
            eMethod             := TCVN_CTM_LAB,
            ipMask              := 0,
            nSkipPixels         := 3,
    eClusteringAlgorithm:= TCVN_CA_KMEANSPP,
    nMaxClusterRadius  := 0,
    bSingleSplitSteps  := FALSE,
            hrPrev              := hr);
```

相关函数

有三种方法可以将参考颜色传递给相应的F_VN_ReferenceColorSimilarity函数：

- 作为一个经过训练的颜色模型，类型ITcVnColorModel
- 作为一个经过训练的颜色模型，类型ITcVnMlModel
- 直接使用TcVnVector3_LREAL中的数值

对于颜色模型，有相应的函数用于训练，而对于所有三种数据类型，各有一个函数用于执行：

- F_VN_TrainImageColor [▶ 1265]用于训练一个颜色模型
- F_VN_ReferenceColorSimilarity [▶ 1260]通过颜色模型进行分割

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.28 F_VN_TrainImageColorExp2_ITcVnMlModel



Create a new color model, describing the image color. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_TrainImageColorExp2_ITcVnMlModel : HRESULT
VAR_INPUT
    ipSrcImage           : ITcVnImage;
    ipColorModel         : Reference To ITcVnMlModel;
    nDifferentColors     : UDINT;
    eMethod              : ETcVnColorTrainingMethod;
    ipMask               : ITcVnImage;
    nSkipPixels          : UDINT;
    eClusteringAlgorithm : ETcVnClusteringAlgorithm;
    nMaxClusterRadius    : DINT;
    bSingleSplitSteps    : BOOL;
    hrPrev               : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（3 通道（RGB），USINT 类型）
ipColorModel	引用 ITcVnMlModel [▶ 417]	返回颜色模型
nDifferentColors	UDINT	要区分的不同颜色的最大数量（如果使用 LBG 作为聚类算法，则结果中的不同颜色数量可能较少，具体取决于 nMaxClusterRadius）
eMethod	ETcVnColorTrainingMethod [▶ 177]	色彩训练方法
ipMask	ITcVnImage [▶ 407]	可选的图像掩膜（1 通道，USINT 类型，如果不需要，则设为 0）
nSkipPixels	UDINT	在每个评估的颜色样本之间跳过的像素数量（以获得更好的性能）。0 将每个像素都考虑在内，往往更准确。
eClusteringAlgorithm	ETcVnClusteringAlgorithm [▶ 163]	聚类算法
nMaxClusterRadius	DINT	仅用于 LBG 聚类算法。单个聚类允许的最大半径（> 0），即聚类半径越大，被分割成的聚类越小，直到达到 nDifferentColors 的全局数量。
bSingleSplitSteps	BOOL	仅用于 LBG 聚类算法。如果为 true，则总是在拆分单个聚类后运行全局优化。如果为 false，则会在应用全局优化之前，在同一步骤中拆分多个聚类。减少全局优化的应用次数会加快速度，但可能会导致结果不理想，尤其是有 2 个可由 1 表示的邻近聚类时
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

该函数F_VN_TrainImageColorExp2_ITcVnMlModel 是F_VN_TrainImageColorExp_ITcVnMlModel [▶ 1270]的扩展专家版。它包含进一步的参数。通过相应的功能块，颜色模型可以在硬盘上保存和加载。

参数

参考图像

参考图像ipSrcImage必须有 3 个 RGB 格式的通道，且元素类型为USINT（8 位）。

颜色模型

参数ipColorModel返回训练的颜色模型。这是一个接口指针，类型为ITcVnMlModel [▶ 417]。

需要区分的颜色数量

数字nDifferentColors定义了需要训练的颜色数量。请注意，图像背景也可以有自己的颜色。

集群算法

在选择集群算法时，有 KMeans++（固定集群数量）和 LBG（动态集群数量）可供选择。

最大集群半径

仅与 LBG 集群算法一起使用。这可以用来影响各个集群的分布。

全球优化

仅与 LBG 集群算法一起使用。这允许在更短的执行时间（bSingleSplitSteps = FALSE）和更好的结果（bSingleSplitSteps = TRUE）之间做出选择。

更少使用全局优化会更快，但可能导致不太理想的结果。

应用

```
VAR
    ipColorModel    :   ITcVnColorModel;
END_VAR

hr := F_VN_TrainImageColorExp2_ITcVnMlModel(
            ipSrcImage          := ipImageRef,
       ipColorModel      := ipColorModel,
       nDifferentColors  := 5, // 4 colors + 1 background = 5
            eMethod             := TCVN_CTM_LAB,
            ipMask              := 0,
            nSkipPixels         := 3,
     eClusteringAlgorithm:= TCVN_CA_KMEANSPP,
     nMaxClusterRadius  := 0,
     bSingleSplitSteps  := FALSE,
       hrPrev            := hr);
```

相关函数

有三种方法可以将参考颜色传递给相应的F_VN_ReferenceColorSimilarity函数：

- 作为一个经过训练的颜色模型，类型ITcVnColorModel
- 作为一个经过训练的颜色模型，类型ITcVnMlModel
- 直接使用TcVnVector3_LREAL中的数值

对于颜色模型，有相应的函数用于训练，而对于所有三种数据类型，各有一个函数用于执行：

- F_VN_TrainImageColor [▶ 1265]用于训练一个颜色模型
- F_VN_ReferenceColorSimilarity [▶ 1260]通过颜色模型进行分割

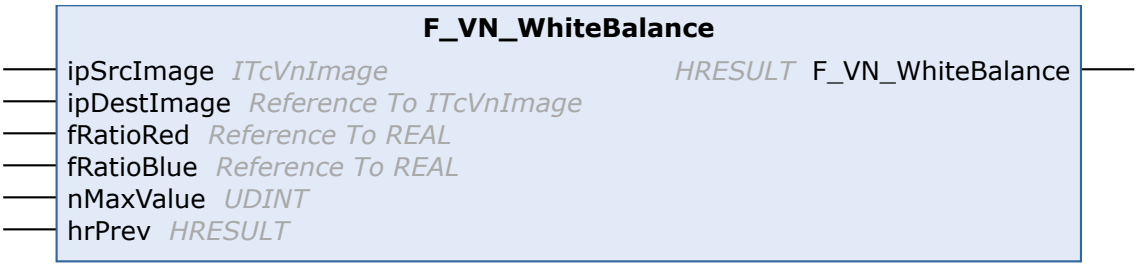
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.16.29 F_VN_WhiteBalance



Calculates the white balance ratios for a RGB image. Uses the green channel as a reference and computes the relative ratios for the red and blue channels. Requires a non-reflecting, not overexposed white colored object (e.g. a sheet of paper) in at least 100 pixels of the image to provide reasonable results. The resulting ratios can then be set on the camera (multiply with existing ratios), or applied in the PLC via MultiplyImageWithVector.

语法

定义：

```
FUNCTION F_VN_WhiteBalance : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    fRatioRed   : Reference To REAL;
    fRatioBlue  : Reference To REAL;
    nMaxValue   : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（3 通道（RGB），USINT 或 UINT）
ipDestImage	引用 ITcVnImage [▶ 407]	应用白平衡的目标图像（可选项，如果不需要，则设为 0。如有需要，将创建适当的图像）。
fRatioRed	引用 REAL	返回红色通道的比率
fRatioBlue	引用 REAL	返回蓝色通道的比率
nMaxValue	UDINT	指定曝光过度所能达到的最大像素值（8 位数据通常为 255，12 位数据通常为 4095）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17 图像滤波

6.1.4.17.1 F_VN_BilateralFilter



Apply a Bilateral filter to smooth the image but preserve edges.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_BilateralFilter : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    nDiameter   : DINT;
    fSigmaColor : LREAL;
    fSigmaSpace : LREAL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（必须与 ipSrcImage 不同！如有需要，将创建适当的目标图像。）
nDiameter	DINT	用于滤波的像素邻域直径（如 5、7、9）。如果 <= 0，则根据 fSigmaSpace 自动选择。
fSigmaColor	LREAL	用于颜色空间滤波的西格玛值（> 0）。数值越大，表示混合的颜色越多。
fSigmaSpace	LREAL	用于坐标空间滤波的西格玛值（> 0）。数值越大，表示能相互影响的像素越多。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_BilateralFilter对原始图像进行双边过滤。在保留图像中的边缘的同时，该过滤器使图像变得平滑并减少噪音。

参数

原始图像

原始图像ipSrcImage必须是元素类型TCVN_ET_USINT或浮点类型，且有 1 或 3 个通道。

结果图像

结果图像ipDestImage必须是与原始图像ipSrcImage不同的接口指针！

过滤器直径

过滤器直径nDiameter决定了一个过滤区域的大小。预计为奇数。如果nDiameter = 0，那么过滤器直径将根据fSigmaSpace自动计算。

颜色区域

这个值可能不是0。

房间面积

这个值可能不是0。

专家参数

关于更多参数，可查看专家版F_VN_BilateralFilterExp [▶ 1280]。

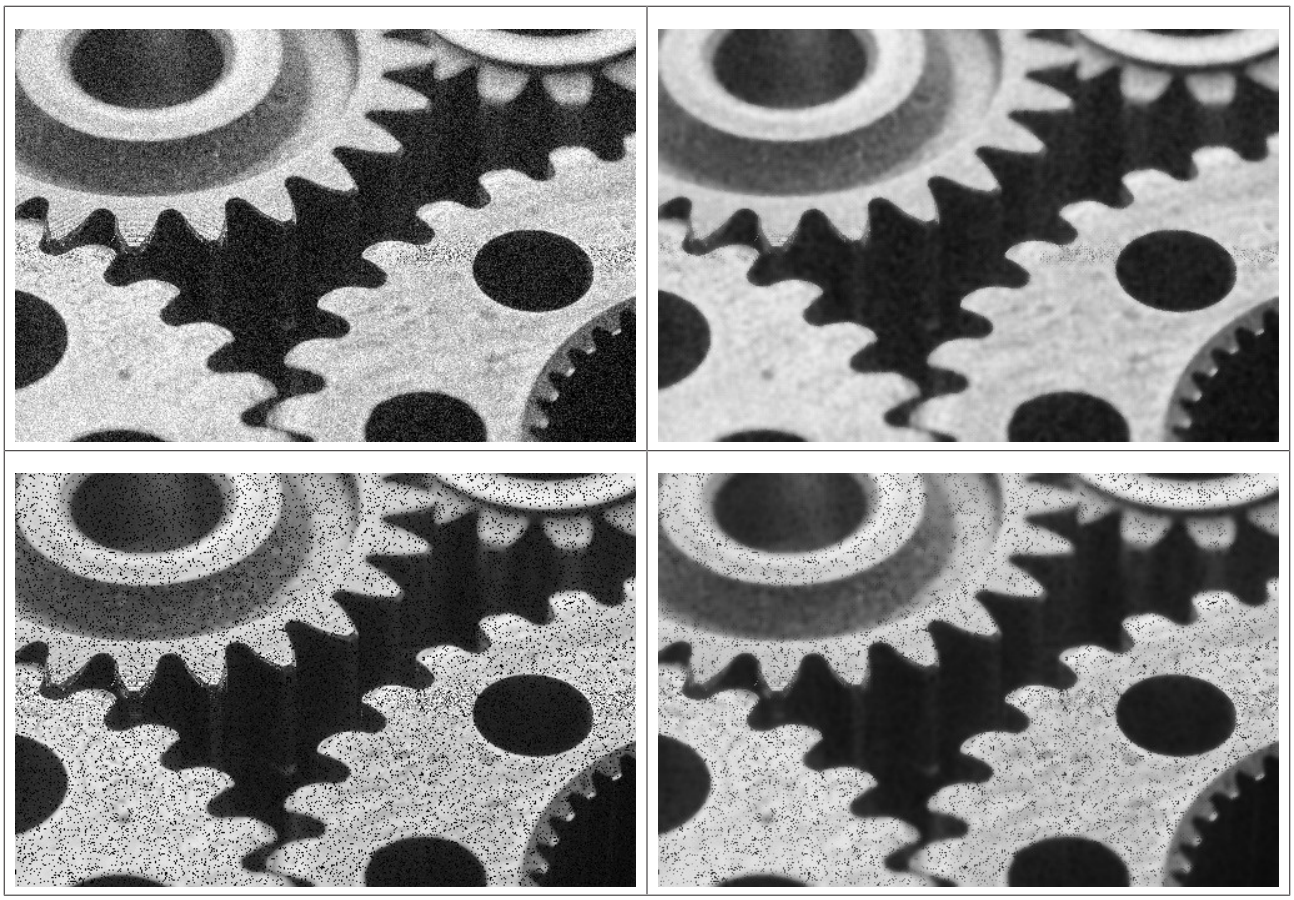
应用

例如，这个函数的应用可以如下所示：

```
hr := F_VN_BilateralFilter(
    ipSrcImage      :=  ipImage,
    ipDestImage     :=  ipImageRes,
    nDiameter       :=  nDiameter,
    fSigmaColor     :=  fSigmaColor,
    fSigmaSpace     :=  fSigmaSpace,
    hrPrev          :=  hr
);
```

未经处理的原始图像（第[1]行）在细节上已经表现出齿轮表面的精细结构。为了说明过滤器的效果，在原始图像中增加了高斯噪声（第[2]行）和椒盐噪声（第[3]行）的额外干扰。

原始图像	结果图像： 直径 = 7 Sigma = 100
	



样本

- 模糊过滤器 [▶ 2785]

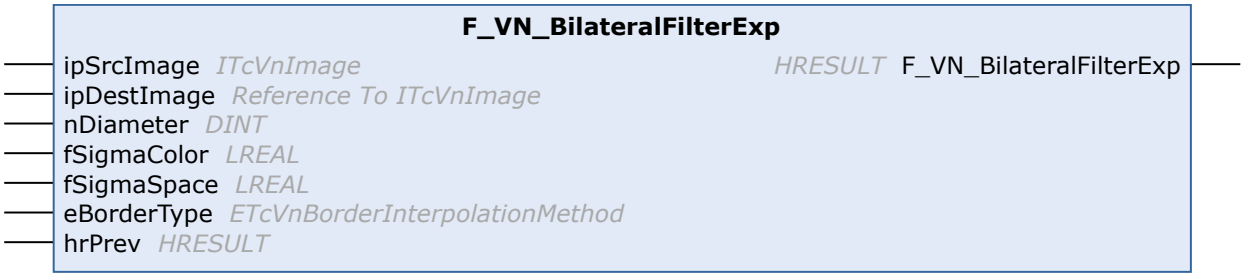
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.2 F_VN_BilateralFilterExp



Apply a Bilateral filter to smooth the image but preserve edges. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_BilateralFilterExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    nDiameter   : DINT;
    fSigmaColor : LREAL;
    fSigmaSpace : LREAL;
    eBorderType : ETcVnBorderInterpolationMethod;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（必须与 ipSrcImage 不同！如有需要，将创建适当的目标图像。）
nDiameter	DINT	用于滤波的像素邻域直径（如 5、7、9）。如果 <= 0，则根据 fSigmaSpace 自动选择。
fSigmaColor	LREAL	用于颜色空间滤波的西格玛值（> 0）。数值越大，表示混合的颜色越多。
fSigmaSpace	LREAL	用于坐标空间滤波的西格玛值（> 0）。数值越大，表示能相互影响的像素越多。
eBorderType	ETcVnBorderInterpolationMethod [▶ 160]	图像边框处理
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

该函数`F_VN_BilateralFilterExp`是F_VN_BilateralFilter [▶ 1278]的专家版本。它包含额外的参数。

参数

原始图像

原始图像`ipSrcImage`必须是元素类型`TCVN_ET_USINT`或浮点类型，且有 1 或 3 个通道。

结果图像

结果图像`ipDestImage`必须是与原始图像`ipSrcImage`不同的接口指针！

过滤器直径

过滤器直径`nDiameter`决定了一个过滤区域的大小。预计为奇数。如果`nDiameter = 0`，那么过滤器直径将根据`fSigmaSpace`自动计算。

颜色区域

这个值可能不是`0`。

房间面积

这个值可能不是`0`。

边界推断

边界推断eBorderType的方法定义了如何将不存在的像素外推到图像边界之外，以计算图像边界的过滤值。更多细节，请参见ETcVnBorderInterpolationMethod [▶ 160]。

应用

例如，这个函数的应用可以如下所示：

```
hr := F_VN_BilateralFilterExp(
    ipSrcImage      :=  ipImage,
    ipDestImage     :=  ipImageRes,
    nDiameter       :=  nDiameter,
    fSigmaColor     :=  fSigmaColor,
    fSigmaSpace     :=  fSigmaSpace,
  eBorderType   := eBorderType,
    hrPrev          :=  hr
);
```

样本

- 模糊过滤器 [▶ 2785]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.3 F_VN_BoxFilter



Apply a box filter to an image.

语法

定义：

```
FUNCTION F_VN_BoxFilter : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDestImage   : Reference To ITcVnImage;
    nFilterWidth  : UDINT;
    nFilterHeight : UDINT;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
nFilterWidth	UDINT	滤波宽度，单位：像素
nFilterHeight	UDINT	滤波高度，单位：像素
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

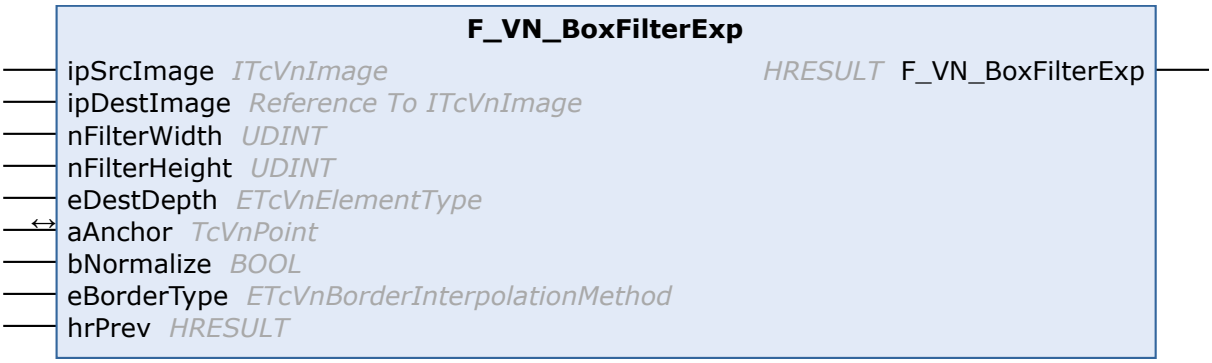
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.4 F_VN_BoxFilterExp



Apply a box filter to an image. (expert function)

语法

定义：

```
FUNCTION F_VN_BoxFilterExp : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDestImage   : Reference To ITcVnImage;
    nFilterWidth  : UDINT;
    nFilterHeight : UDINT;
    eDestDepth    : ETcVnElementType;
END_VAR
VAR_IN_OUT
    aAnchor       : TcVnPoint;
END_VAR
VAR_INPUT
    bNormalize    : BOOL;
    eBorderType   : ETcVnBorderInterpolationMethod;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
nFilterWidth	UDINT	滤波宽度，单位：像素
nFilterHeight	UDINT	滤波高度，单位：像素
eDestDepth	ETcVnElementType [▶ 191]	目标图像深度
bNormalize	BOOL	如果为 true，内核将根据 nFilterWidth * nFilterHeight 进行归一化处理

名称	类型	描述
eBorderType	ETcVnBorderInterpolationMethod [▶ 160]	图像边框处理
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aAnchor	TcVnPoint [▶ 152]	内核的锚点（[-1，-1] 为中心点）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.5 F_VN_BrightBorderObjects



Find bright objects connected to the image border within a gray-scale single-channel image.

语法

定义：

```
FUNCTION F_VN_BrightBorderObjects : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT、UINT、INT、REAL 或 LREAL，1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

这个函数将输入图像中的所有明亮的边框对象转移到结果图像中。

算法

图像中所有比周围像素更亮的物体都被移除。

参数

输入图像

输入图像ipSrcImage只能有一个通道，并且必须为以下元素类型之一：USINT、UINT、INT、REAL或LREAL。

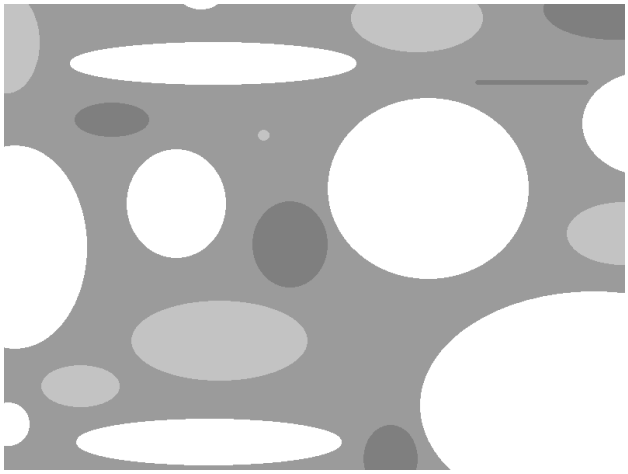
结果图像

结果图像ipDestImage返回明亮的边界对象。

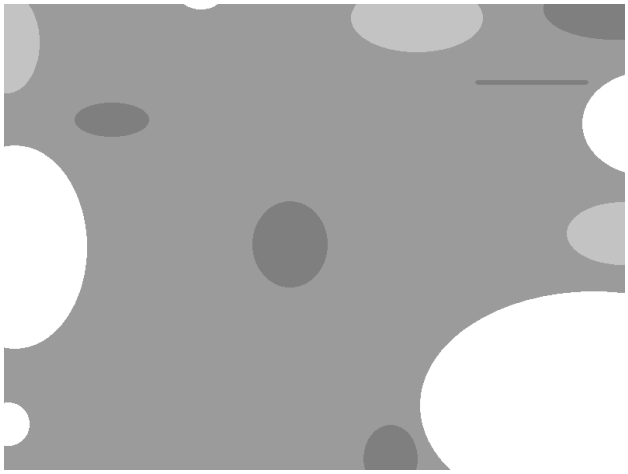
应用

```
hr := F_VN_BrightBorderObjects(
    ipSrcImage      :=  ipImageIn,
    ipDestImage     :=  ipImageRes,
    hrPrev          :=  hr
);
```

输入图像



结果图像



相关函数

- F_VN_DarkBorderObjects [▶ 1292]
- F_VN_FillHoles [▶ 1293]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.6 F_VN_CreateStructuringElement



Creates a structuring element for the usage with morphological operators and allocate an appropriate data buffer. The initial reference count is set to one if a new image interface is created and kept, otherwise.

语法

定义：

```
FUNCTION F_VN_CreateStructuringElement : HRESULT
VAR_INPUT
    ipStructuringElement : Reference To ITcVnImage;
    eShape               : ETcVnStructuringElementShape;
    nWidth               : UDINT;
    nHeight              : UDINT;
    hrPrev               : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipStructuringElement	引用 ITcVnImage [▶ 407]	返回创建的结构元素（重复使用非零接口指针。）
eShape	ETcVnStructuringElementShape [▶ 211]	结构元素的形状（矩形、十字形或椭圆形）
nWidth	UDINT	Width（宽度）
nHeight	UDINT	高度
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

该函数创建了一个结构元素，可通过F_VN_MorphologicalOperator [▶ 1303]用于形态学操作。

参数

结构元素

创建的结构元素`ipStructuringElement`通过使用数值`1`以像素表示结构元素返回为ITcVnImage [▶ 407]。

形状

结构元素的形状`eShape`由枚举ETcVnStructuringElementShape [▶ 211]定义。以下形状可用：

TCVN_SES_RECTANGLE **TCVN_SES_CROSS** **TCVN_SES_ELLIPSE**





由于结构元素的像素值为 1，为了更好地显示，这里将它们按比例放大。

大小

结构元素的大小通过待创建图像`ipStructuringElement`的宽度`nWidth`和高度`nHeight`定义。

应用

例如，创建大小为`9`的方形结构元素如下所示：

```
hr := F_VN_CreateStructuringElement(
    ipStructuringElement    :=  ipElement,
    eShape                  :=  TCVN_SES_RECTANGLE,
    nWidth                  :=  9,
    nHeight                 :=  9,
    hrPrev                  :=  hr
);
```

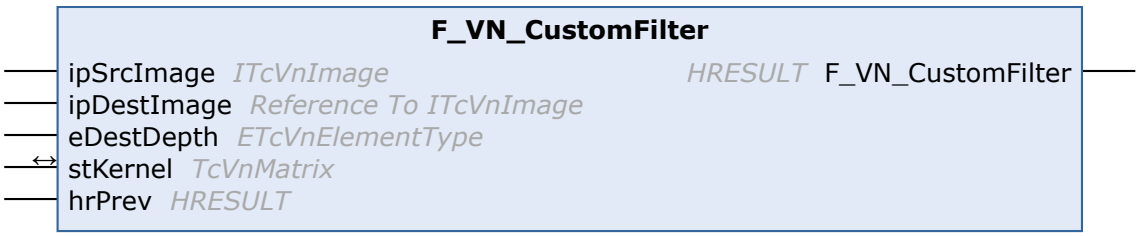
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.7 F_VN_CustomFilter



Apply a custom filter to the image.

语法

定义：

```
FUNCTION F_VN_CustomFilter : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    eDestDepth  : ETcVnElementType;
END_VAR
VAR_IN_OUT
    stKernel    : TcVnMatrix;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eDestDepth	ETcVnElementType [▶ 191]	目标图像深度
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stKernel	TcVnMatrix [▶ 226]	包含 REAL 或 LREAL 类型值的自定义滤波核

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_CustomFilter对输入图像应用用户定义的过滤器。过滤器内核由一个矩阵定义。这个矩阵与图像一起被折叠起来。

参数

输入图像

输入图像ipSrcImage可以有任何可用的格式。

结果图像

结果图像ipDestImage的格式与输入图像ipSrcImage相同。

结果深度

结果深度eDestDepth定义了结果图像ipDestImage的元素类型。一个更大的元素类型可以显示更多的信息。

过滤器内核

用户定义的过滤器内核stKernel以矩阵形式传输。允许的矩阵元素类型为TCVN_ET_REAL和TCVN_ET_LREAL。通过函数F_VN_InitMatrixStruct [▶ 1524]，可以创建类型为TcVnMatrix [▶ 226]的相应矩阵。

专家参数

专家版F_VN_CustomFilterExp [▶ 1290]包含额外的参数。

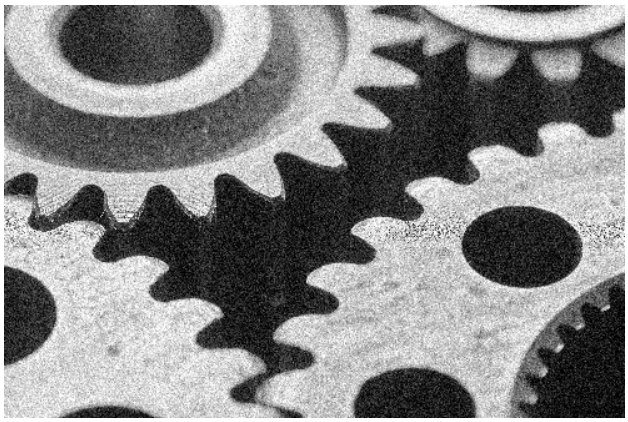
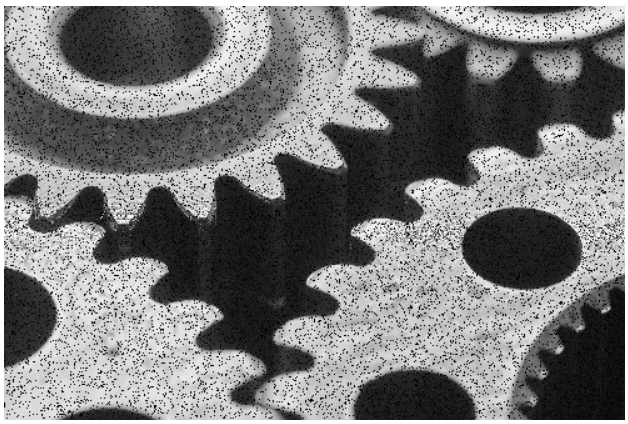
应用

```
hr := F_VN_InitMatrixStruct(
    pSrcBuffer      := ADR(aMatrixArray7x7),
    stDestMatrix    := stKernelMatrix,
    nRows           := 7,
    nCols           := 7,
    eElementType    := TCVN_ET_REAL,
    hrPrev          := hr
);

hr := F_VN_CustomFilter(
    ipSrcImage  :=  ipImageIn,
    ipDestImage :=  ipImageRes,
    eDestDepth  :=  TCVN_ET_USINT,
```

```
    stKernel    :=  stKernelMatrix,
    hrPrev      :=  hr
);
```

- 未经处理的原始图像（第[1]行）在细节上已经表现出齿轮表面的精细结构。为了说明过滤器的效果，在原始图像中增加了高斯噪声（第[2]行）和椒盐噪声（第[3]行）的额外干扰。作为一个例子，实现了一个平均过滤器；关于更详细的描述，可查看模糊过滤器 [▶ 2785]的样本：

原始图像	结果图像： 平均数过滤器
	
	
	

样本

- 模糊过滤器 [▶ 2785]
-

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.8 F_VN_CustomFilterExp



Apply a custom filter to the image. (expert function)

语法

定义：

```
FUNCTION F_VN_CustomFilterExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    eDestDepth  : ETcVnElementType;
END_VAR
VAR_IN_OUT
    stKernel    : TcVnMatrix;
    aAnchor     : TcVnPoint;
END_VAR
VAR_INPUT
    fDelta      : LREAL;
    eBorderType : ETcVnBorderInterpolationMethod;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eDestDepth	ETcVnElementType [▶ 191]	目标图像深度
fDelta	LREAL	滤波后每个像素的附加值
eBorderType	ETcVnBorderInterpolationMethod [▶ 160]	图像边框处理
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stKernel	TcVnMatrix [▶ 226]	包含 REAL 或 LREAL 类型值的自定义滤波核
aAnchor	TcVnPoint [▶ 152]	内核的锚点

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_CustomFilterExp是F_VN_CustomFilter [▶ 1287]的专家版本。它包含额外的参数。

参数

输入图像

输入图像ipSrcImage可以有任何可用的格式。

结果图像

结果图像ipDestImage的格式与输入图像ipSrcImage相同。

结果深度

结果深度eDestDepth定义了结果图像ipDestImage的元素类型。一个更大的元素类型可以显示更多的信息。

过滤器内核

用户定义的过滤器内核stKernel以矩阵形式传输。允许的矩阵元素类型为TCVN_ET_REAL和TCVN_ET_LREAL。通过函数F_VN_InitMatrixStruct [▶ 1524]，可以创建类型为TcVnMatrix [▶ 226]的相应矩阵。

锚点

锚点aAnchor定义了在输入图像ipSrcImage的每个像素上移动过滤器内核的哪个点。

Delta

常量 deltafDelta被添加到过滤操作的结果中。

边界推断

边界推断eBorderType的方法定义了如何将不存在的像素外推到图像边界之外，以计算图像边界的过滤值。更多细节，请参见ETcVnBorderInterpolationMethod [▶ 160]。

应用

```
hr := F_VN_CustomFilterExp(
    ipSrcImage   :=  ipImageIn,
    ipDestImage  :=  ipImageRes,
    eDestDepth   :=  TCVN_ET_USINT,
    stKernel     :=  stKernelMatrix,
  aAnchor    := aAnchorPoint,
  fDelta     := 0,
  eBorderType := TCVN_BIM_DEFAULT,
    hrPrev       :=  hr
);
```

样本

-

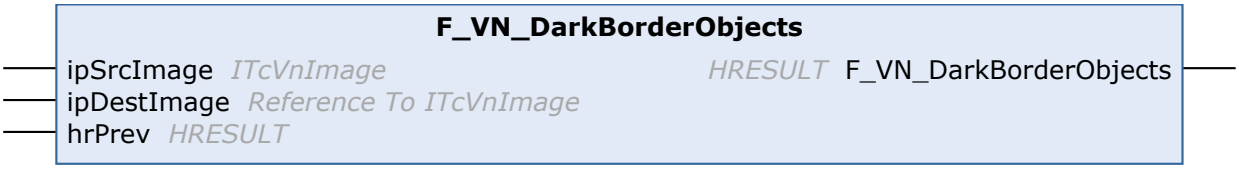
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.9 F_VN_DarkBorderObjects



Find dark objects connected to the image border within a gray-scale single-channel image. (equivalent to filling holes)

语法

定义：

```
FUNCTION F_VN_DarkBorderObjects : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT、UINT、INT、REAL 或 LREAL，1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

这个函数将输入图像中的所有暗色边界对象转移到结果图像中。

算法

图像中所有比周围像素更暗的对象都被删除。

参数

输入图像

输入图像`ipSrcImage`只能有一个通道，并且必须为以下元素类型之一：`USINT`、`UINT`、`INT`、`REAL`或`LREAL`。

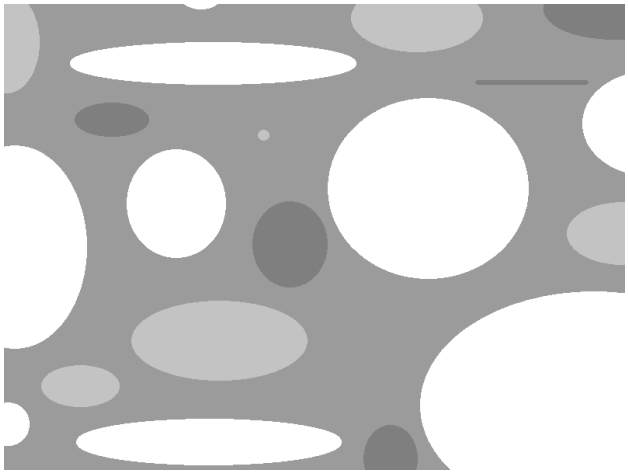
结果图像

结果图像`ipDestImage`返回深色边界对象。

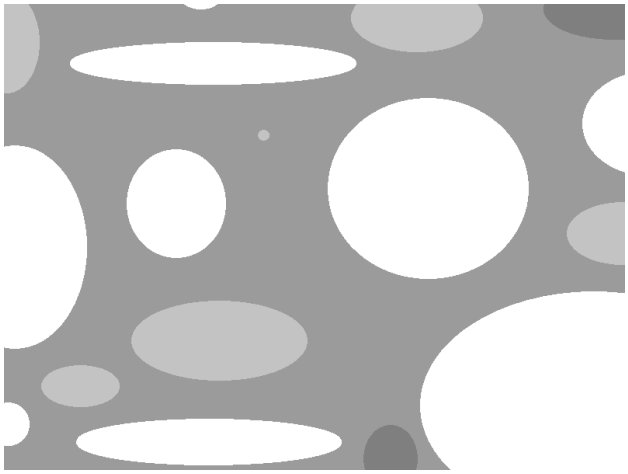
应用

```
hr := F_VN_DarkBorderObjects(
    ipSrcImage      :=  ipImageIn,
    ipDestImage     :=  ipImageRes,
    hrPrev          :=  hr
);
```

输入图像



结果图像



相关函数

- F_VN_FillHoles [▶ 1293] als Alias
- F_VN_BrightBorderObjects [▶ 1284]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.10 F_VN_FillHoles



Fill holes within a gray-scale single-channel image. (equivalent to finding dark border objects)

语法

定义：

```
FUNCTION F_VN_FillHoles : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT、UINT、INT、REAL 或 LREAL，1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

这个函数填补了输入图像中的孔。

算法

图像中所有比周围像素更暗的对象都被删除。

参数

输入图像

输入图像`ipSrcImage`只能有一个通道，并且必须为以下元素类型之一：`USINT`、`UINT`、`INT`、`REAL`或`LREAL`。

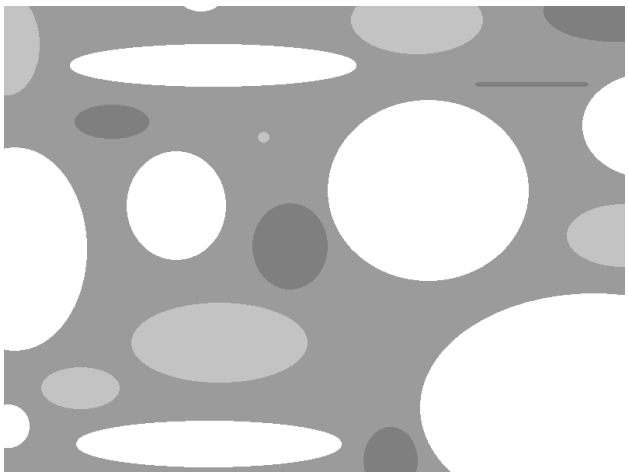
结果图像

结果图像`ipDestImage`返回填充了孔的图像。

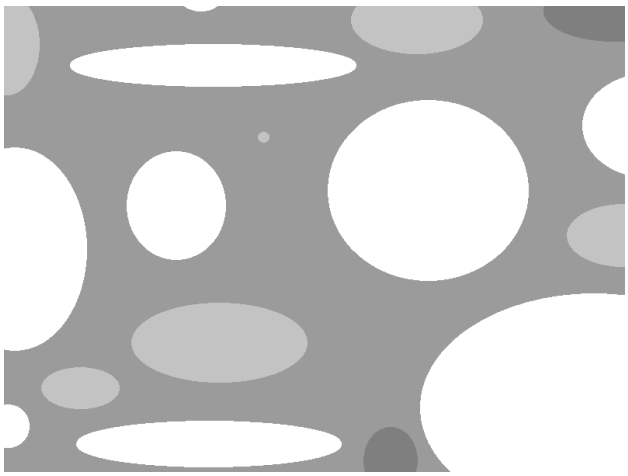
应用

```
hr := F_VN_FillHoles(
    ipSrcImage      :=  ipImageIn,
    ipDestImage     :=  ipImageRes,
    hrPrev          :=  hr
);
```

输入图像



结果图像



相关函数

- F_VN_DarkBorderObjects [▶ 1292] as alias
- F_VN_BrightBorderObjects [▶ 1284]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.11 F_VN_GaussianFilter



应用高斯（Gaussian）滤波器来平滑图像。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_GaussianFilter : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDestImage   : Reference To ITcVnImage;
    nFilterWidth  : UDINT;
    nFilterHeight : UDINT;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
nFilterWidth	UDINT	滤波宽度，单位：像素（1、3、5、7...）
nFilterHeight	UDINT	滤波高度，单位：像素（1、3、5、7...）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.17.12 F_VN_GaussianFilterExp



Apply a Gaussian filter to smooth the image. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_GaussianFilterExp : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDestImage   : Reference To ITcVnImage;
    nFilterWidth  : UDINT;
    nFilterHeight : UDINT;
    fSigmaX       : LREAL;
    fSigmaY       : LREAL;
    eBorderType   : ETcVnBorderInterpolationMethod;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
nFilterWidth	UDINT	滤波宽度，单位：像素（1、3、5、7...）
nFilterHeight	UDINT	滤波高度，单位：像素（1、3、5、7...）
fSigmaX	LREAL	高斯核在 X 方向上的标准偏差（>= 0，如果为 0 则自动选择）
fSigmaY	LREAL	高斯核在 Y 方向上的标准偏差（>= 0，如果为 0 则自动选择）
eBorderType	ETcVnBorderInterpolationMethod [▶ 160]	图像边框处理
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_GaussianFilterExp是F_VN_GaussianFilter [▶ 1295]的专家版本。它包含额外的参数。

参数

输入图像

输入图像ipSrcImage可以有任何可用的格式。

结果图像

结果图像ipDestImage格式与输入图像ipSrcImage相同。

过滤器尺寸

高斯过滤器的大小在 X 方向由nFilterWidth且在 Y 方向由nFilterHeight描述。尺寸规格必须是奇数，因为必须始终有一个中心像素。

标准偏差

X 方向的标准偏差 fSigmaX 和 Y 方向的标准偏差 fSigmaY 不仅定义了过滤器的大小，而且还定义了过滤器中使用的高斯曲线的形状。

边界推断

边界推断eBorderType的方法定义了如何将不存在的像素外推到图像边界之外，以计算图像边界的过滤值。更多细节，请参见ETcVnBorderInterpolationMethod [▶ 160]。

应用

```
hr := F_VN_GaussianFilterExp(
    ipSrcImage      :=  ipImageIn,
    ipDestImage     :=  ipImageRes,
    nFilterWidth    :=  3,
    nFilterHeight   :=  3,
    fSigmaX         :=  5,
    fSigmaY         :=  5,
    eBorderType     :=  TCVN_BIM_DEFAULT,
    hrPrev          :=  hr
);
```

样本

- 模糊过滤器 [▶ 2785]

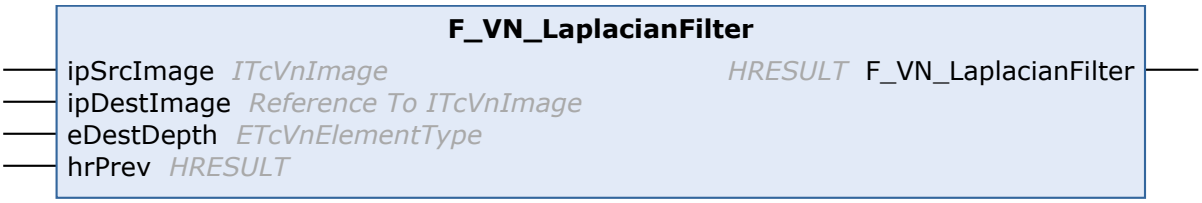
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.13 F_VN_LaplacianFilter



对图像应用拉普拉斯（Laplacian）滤波。

语法

定义：

```
FUNCTION F_VN_LaplacianFilter : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    eDestDepth  : ETcVnElementType;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eDestDepth	ETcVnElementType [▶ 191]	目标图像深度
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

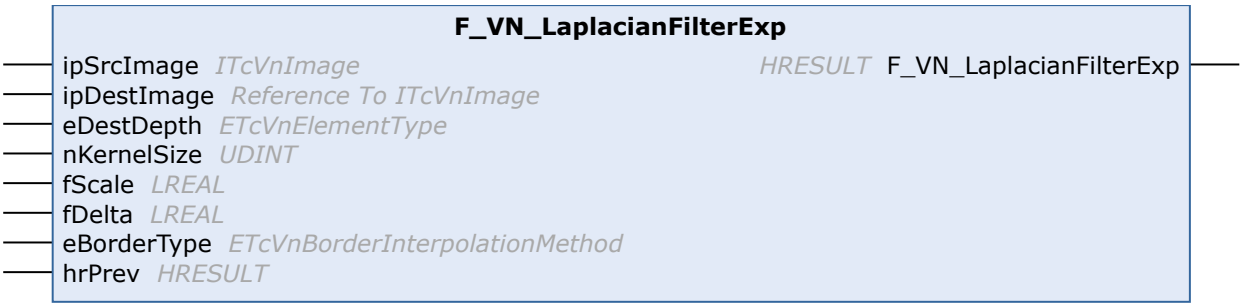
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.17.14 F_VN_LaplacianFilterExp



Apply a Laplacian filter to an image. (expert function)

语法

定义：

```
FUNCTION F_VN_LaplacianFilterExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    eDestDepth  : ETcVnElementType;
    nKernelSize : UDINT;
    fScale      : LREAL;
    fDelta      : LREAL;
    eBorderType : ETcVnBorderInterpolationMethod;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eDestDepth	ETcVnElementType [▶ 191]	目标图像深度
nKernelSize	UDINT	用于计算二阶导数滤波的孔径大小（1、3、5...31）
fScale	LREAL	导数计算值的比例系数
fDelta	LREAL	在存储结果前加入这些结果的 Delta 值

名称	类型	描述
eBorderType	ETcVnBorderInterpolationMethod [▶ 160]	图像边框处理
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_LaplacianFilterExp是F_VN_LaplacianFilter [▶ 1297]的专家版本。它包含额外的参数。

参数

输入图像

输入图像 ipSrcImage 可以是任何可用的格式，除了一般不支持的数据类型 SINT 和 DINT。

结果图像

结果图像 ipDestImage 的大小与输入图像 ipSrcImage 相同，但像素格式由结果深度eDestDepth决定。

结果深度

结果深度eDestDepth定义了结果图像ipDestImage的元素类型。一个更大的元素类型可以显示更多的信息。

注意

输入图像和结果深度的组合

只有与输入图像相同或更大的数据类型才能作为结果深度，否则数据就会丢失。例如，一个 UINT 不能被转换为 USINT。此外，还有两个限制，所以 UINT 不能被转换成 INT，且 REAL 不能被转换成 LREAL。

内核大小

参数nKernelSize定义了用于近似二阶导数的过滤器核的大小。只允许使用1和31之间的奇数。

缩放

拉普拉斯运算的结果乘以缩放系数fScale。

Delta

在通过fScale缩放后，常量 deltafDelta被添加到夏尔运算的结果中。

边界推断

边界推断eBorderType的方法定义了如何将不存在的像素外推到图像边界之外，以计算图像边界的过滤值。更多细节，请参见ETcVnBorderInterpolationMethod [▶ 160]。

应用

例如，内核大小为3×3的拉普拉斯过滤器的应用如下所示：

```
hr := F_VN_LaplacianFilterExp(
    ipSrcImage  :=  ipImageIn,
    ipDestImage :=  ipImageRes,
    eDestDepth  :=  TCVN_ET_USINT,
 nKernelSize := 3,
 fScale    := 1,
 fDelta    := 0,
```

```
  eBorderType := TCVN_BIM_DEFAULT,
    hrPrev      :=  hr
);
```

样本

- 边缘检测过滤器 [▶ 2787]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.15 F_VN_LocalMaxima



Find local maxima in a gray-scale single-channel image. The found maxima are marked by a value of 1 in the destination image.

语法

定义：

```
FUNCTION F_VN_LocalMaxima : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 或 UINT，1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（必须与 ipSrcImage 不同！如有需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_LocalMaxima找到输入图像中的局部极大值。

参数

输入图像

输入图像ipSrcImage必须是位深为 8 或 16 位的单通道灰度图像（USINT 或 UINT）。

结果图像

结果图像ipDestImage与输入图像ipSrcImage具有相同的类型。在这里，输入图像的每个局部最大值都以像素值 1 标记。

应用

寻找局部最大值如下所示：

```
hr := F_VN_LocalMaxima(
    ipSrcImage  :=  ipImageIn,
    ipDestImage :=  ipImageRes,
    hrPrev      :=  hr
);
```

样本

- 消除局部极值

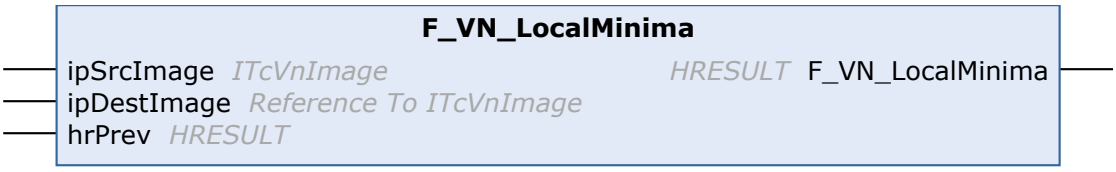
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.16 F_VN_LocalMinima



Find local minima in a gray-scale single-channel image. The found minima are marked by a value of 1 in the destination image.

语法

定义：

```
FUNCTION F_VN_LocalMinima : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT 或 UINT，1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（必须与 ipSrcImage 不同！如有需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_LocalMinima在输入图像中寻找局部极小值。

参数

输入图像

输入图像ipSrcImage必须是位深为 8 或 16 位的单通道灰度图像（USINT 或 UINT）。

结果图像

结果图像ipDestImage与输入图像ipSrcImage具有相同的类型。在这里，输入图像的每个局部最小值都以像素值 1 标记。

应用

寻找局部最小值如下所示：

```
hr := F_VN_LocalMinima(
    ipSrcImage  :=  ipImageIn,
    ipDestImage :=  ipImageRes,
    hrPrev      :=  hr
);
```

样本

- 消除局部极值

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.17 F_VN_MedianFilter



对图像应用中值滤波。

语法

定义：

```
FUNCTION F_VN_MedianFilter : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    nFilterSize : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（对于 nFilterSize 3 或 5：USINT、UINT、REAL。对于较大的滤波，只支持 USINT。
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
nFilterSize	UDINT	滤波大小（宽度和高度）（3、5、7...）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

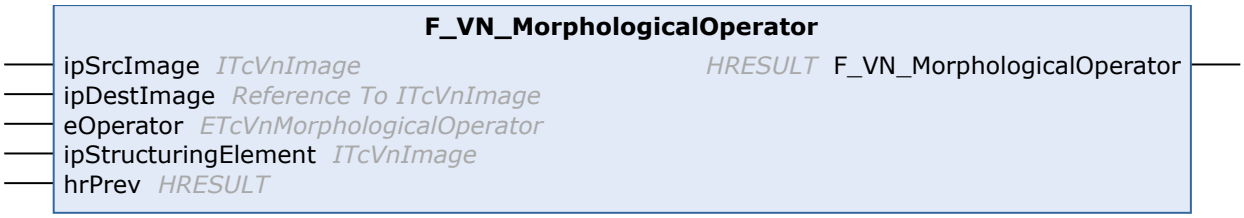
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.17.18 F_VN_MorphologicalOperator



应用形态学算子。

语法

定义：

```
FUNCTION F_VN_MorphologicalOperator : HRESULT
VAR_INPUT
    ipSrcImage           : ITcVnImage;
    ipDestImage          : Reference To ITcVnImage;
    eOperator            : ETcVnMorphologicalOperator;
    ipStructuringElement : ITcVnImage;
    hrPrev               : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（重建算子仅支持 1 个通道）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eOperator	ETcVnMorphologicalOperator [▶ 202]	算子类型
ipStructuringElement	ITcVnImage [▶ 407]	要应用的结构元素（通常通过 F_VN_CreateStructuringElement 创建。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

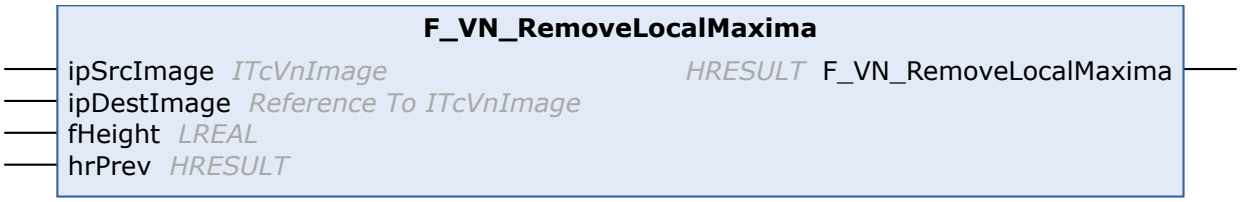
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.17.19 F_VN_RemoveLocalMaxima



Remove local maxima up to a given height from a gray-scale single-channel image.

语法

定义：

```
FUNCTION F_VN_RemoveLocalMaxima : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    fHeight     : LREAL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT、UINT、INT、REAL 或 LREAL，1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
fHeight	LREAL	要删除的最大值的最大高度（必须大于 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_RemoveLocalMaxima从输入图像中删除局部极大值。

参数

输入图像

输入图像ipSrcImage只能有一个通道，并且必须为以下元素类型之一：USINT、UINT、INT、REAL或LREAL。

结果图像

结果图像ipDestImage返回已经去除局部极大值的图像。

最大高度

最大高度fHeight定义了输入图像中的特定高度，如果不超过该高度，极大值将被移除。高度描述的是最大值和有关像素被设定的值之间的强度差异。fHeight必须大于0。

应用

例如，去除不超过高度10的局部极大值如下所示：

```
hr := F_VN_RemoveLocalMaxima(
    ipSrcImage      :=  ipImageIn,
    ipDestImage     :=  ipImageRes,
    fHeight         :=  10,
    hrPrev          :=  hr
);
```

样本

- 消除局部极值

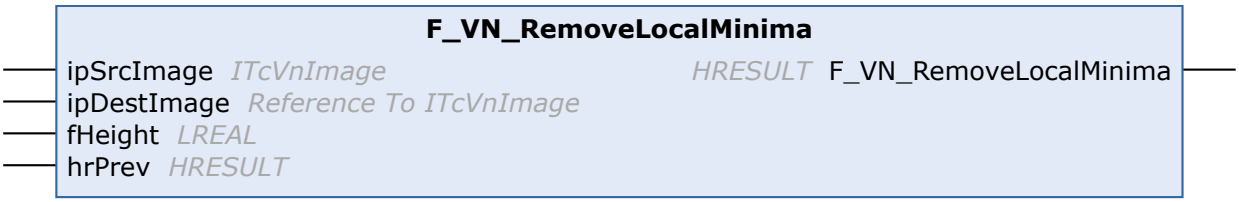
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.20 F_VN_RemoveLocalMinima



Remove local minima up to a given height from a gray-scale single-channel image.

语法

定义：

```
FUNCTION F_VN_RemoveLocalMinima : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    fHeight     : LREAL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT、UINT、INT、REAL 或 LREAL，1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
fHeight	LREAL	要删除的最小值的最大高度（必须大于 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_RemoveLocalMinima从输入图像中删除局部极小值。

参数

输入图像

输入图像ipSrcImage只能有一个通道，并且必须为以下元素类型之一：USINT、UINT、INT、REAL或LREAL。

结果图像

结果图像ipDestImage返回已经去除局部极小值的图像。

最大高度

最大高度fHeight定义了输入图像中的特定高度，如果不超过该高度，极小值将被移除。高度描述的是最小值和有关像素被设定的值之间的强度差异。fHeight必须大于0。

应用

例如，去除不超过高度10的局部极小值如下所示：

```
hr := F_VN_RemoveLocalMinima(
    ipSrcImage      :=  ipImageIn,
    ipDestImage     :=  ipImageRes,
    fHeight         :=  10,
    hrPrev          :=  hr
);
```

样本

- 消除局部极值

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.21 F_VN_ScharrFilter



使用沙尔（Scharr）滤波计算 x 或 y 方向的一阶导数。

语法

定义：

```
FUNCTION F_VN_ScharrFilter : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
    ipDestImage      : Reference To ITcVnImage;
    eDestDepth       : ETcVnElementType;
    eFilterDirection : ETcVnFilterDirection;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eDestDepth	ETcVnElementType [▶ 191]	目标图像深度
eFilterDirection	ETcVnFilterDirection [▶ 195]	滤波方向
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.17.22 F_VN_ScharrFilterExp



Calculates the first order derivative in x or y direction using a Scharr filter. (expert function)

语法

定义：

```
FUNCTION F_VN_ScharrFilterExp : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
    ipDestImage      : Reference To ITcVnImage;
    eDestDepth       : ETcVnElementType;
    eFilterDirection : ETcVnFilterDirection;
    fScale           : LREAL;
    fDelta           : LREAL;
    eBorderType      : ETcVnBorderInterpolationMethod;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eDestDepth	ETcVnElementType [▶ 191]	目标图像深度
eFilterDirection	ETcVnFilterDirection [▶ 195]	滤波方向

名称	类型	描述
fScale	LREAL	导数计算值的比例系数
fDelta	LREAL	将结果存储到目标位置之前，加入这些结果的 Delta 值
eBorderType	ETcVnBorderInterpolationMethod [▶ 160]	图像边框处理
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_ScharrFilterExp是F_VN_ScharrFilter [▶ 1306]的专家版本。它包含额外的参数。

参数

原始图像

原始图像ipSrcImage可以有任何格式。

结果图像

结果图像ipDestImage返回过滤结果，且其格式与原始图像ipSrcImage相同。

结果深度

结果深度eDestDepth定义了结果图像ipDestImage的元素类型。一个更大的元素类型可以显示更多的信息。

过滤方向

过滤方向eFilterDirection定义了夏尔过滤器是在 X 方向（TCVN_FD_X）还是在 Y 方向（TCVN_FD_Y）应用于图像。

缩放

夏尔运算的结果乘以比例系数fScale。

Delta

在通过fScale缩放后，常量 deltafDelta被添加到夏尔运算的结果中。

边界推断

边界推断eBorderType的方法定义了如何将不存在的像素外推到图像边界之外，以计算图像边界的过滤值。更多细节，请参见ETcVnBorderInterpolationMethod [▶ 160]。

应用

```
hr := F_VN_ScharrFilterExp(
    ipSrcImage      :=  ipImageIn,
    ipDestImage     :=  ipImageRes,
    eDestDepth      :=  TCVN_ET_USINT,
    eFilterDirection:=  TCVN_FD_X,
  fScale     := 1,
  fDelta     := 0,
  eBorderType   := TCVN_BIM_DEFAULT,
    hrPrev          :=  hr,
);
```

样本

- 边缘检测过滤器 [▶ 2787]

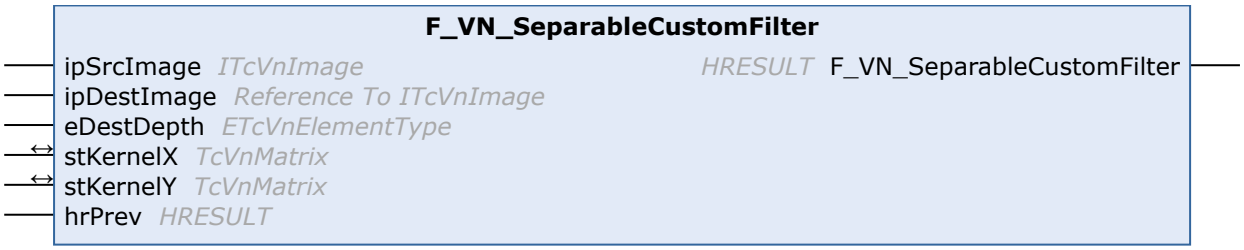
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.23 F_VN_SeparableCustomFilter



对图像应用可分离的自定义滤波。

语法

定义：

```
FUNCTION F_VN_SeparableCustomFilter : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    eDestDepth  : ETcVnElementType;
END_VAR
VAR_IN_OUT
    stKernelX   : TcVnMatrix;
    stKernelY   : TcVnMatrix;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eDestDepth	ETcVnElementType [▶ 191]	目标图像深度
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stKernelX	TcVnMatrix [▶ 226]	包含 REAL 或 LREAL 类型值的一维自定义行滤波核
stKernelY	TcVnMatrix [▶ 226]	包含 REAL 或 LREAL 类型值的一维自定义列滤波核

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.17.24 F_VN_SeparableCustomFilterExp



Apply a separable custom filter to the image. (expert function)

语法

定义：

```
FUNCTION F_VN_SeparableCustomFilterExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    eDestDepth  : ETcVnElementType;
END_VAR
VAR_IN_OUT
    stKernelX   : TcVnMatrix;
    stKernelY   : TcVnMatrix;
    aAnchor     : TcVnPoint;
END_VAR
VAR_INPUT
    fDelta      : LREAL;
    eBorderType : ETcVnBorderInterpolationMethod;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eDestDepth	ETcVnElementType [▶ 191]	目标图像深度
fDelta	LREAL	滤波后每个像素的附加值
eBorderType	ETcVnBorderInterpolationMethod [▶ 160]	图像边框处理
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stKernelX	TcVnMatrix [▶ 226]	包含 REAL 或 LREAL 类型值的一维自定义行滤波核
stKernelY	TcVnMatrix [▶ 226]	包含 REAL 或 LREAL 类型值的一维自定义列滤波核
aAnchor	TcVnPoint [▶ 152]	内核的锚点

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_SeparableCustomFilterExp 是 F_VN_SeparableCustomFilter [▶ 1309] 的 Expert（专家）版本。它包含额外的参数。

参数

输入图像

输入图像ipSrcImage可以有任何可用的格式。

结果图像

结果图像ipDestImage大小与输入图像ipSrcImage相同，但像素格式由结果深度eDestDepth决定。

X 和 Y 内核

两个一维过滤器内核stKernelX和stKernelY作为一个矩阵被转移。允许的矩阵元素类型为TCVN_ET_REAL和TCVN_ET_LREAL。相应类型的矩阵TcVnMatrix [▶ 226]可以通过函数F_VN_InitMatrixStruct [▶ 1524]进行创建。

锚点

锚点 aAnchor 定义了在输入图像 ipSrcImage 的每个像素上移动滤波核的点。

Delta

会将常数 delta fDelta 添加到滤波操作的结果中。

边界推断

边界推断eBorderType的方法定义了如何将不存在的像素外推到图像边界之外，以计算图像边界的过滤值。更多细节，请参见ETcVnBorderInterpolationMethod [▶ 160]。

应用

```
hr := F_VN_SeparableCustomFilterExp(
    ipSrcImage  :=  ipImageIn,
    ipDestImage :=  ipImageRes,
    eDestDepth  :=  TCVN_ET_USINT,
    stKernelX   :=  stKernelMatrixX,
    stKernelY   :=  stKernelMatrixY,
  aAnchor     :=  aAnchorPoint,
    fDelta      :=  0,
    eBorderType :=  TCVN_BIM_DEFAULT,
    hrPrev      :=  hr
);
```

例程

-

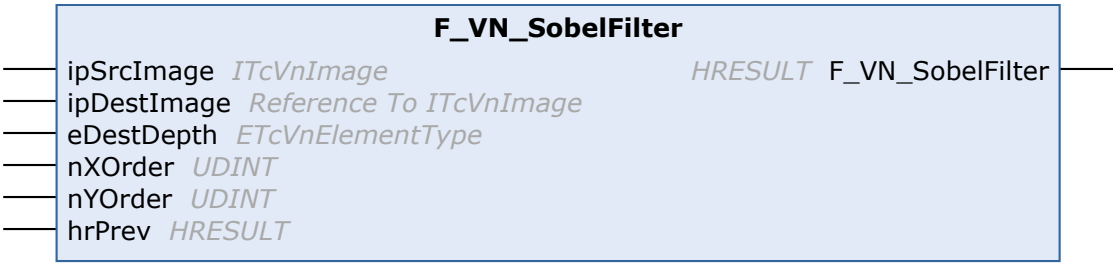
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.25 F_VN_SobelFilter



使用扩展索贝尔（Sobel）滤波计算一阶、二阶或混合图像导数。

语法

定义：

```
FUNCTION F_VN_SobelFilter : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    eDestDepth  : ETcVnElementType;
    nXOrder     : UDINT;
    nYOrder     : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eDestDepth	ETcVnElementType [▶ 191]	目标图像深度
nXOrder	UDINT	x 导数的阶数（0、1、2）
nYOrder	UDINT	y 导数的阶数（0、1、2）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

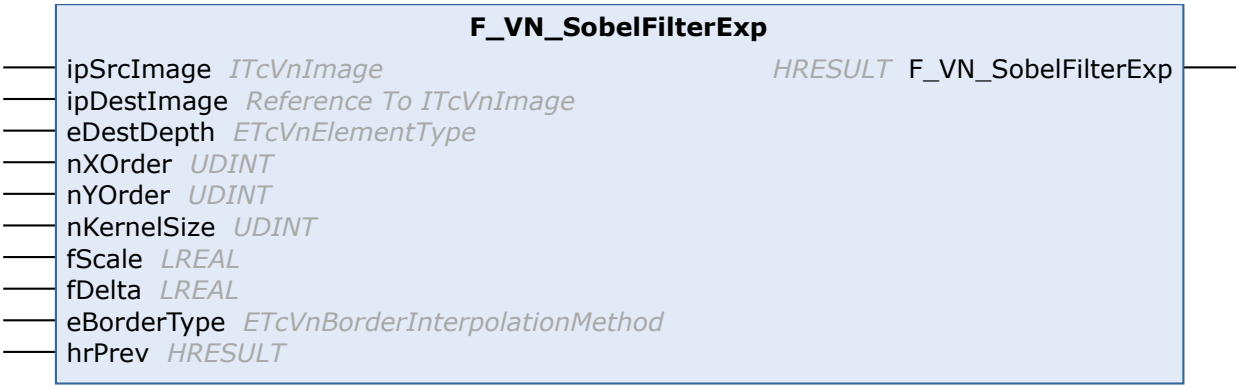
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.17.26 F_VN_SobelFilterExp



Calculates the first, second, third, or mixed image derivatives using an extended Sobel filter. (expert function)

语法

定义：

```
FUNCTION F_VN_SobelFilterExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    eDestDepth  : ETcVnElementType;
    nXOrder     : UDINT;
    nYOrder     : UDINT;
    nKernelSize : UDINT;
    fScale      : LREAL;
    fDelta      : LREAL;
    eBorderType : ETcVnBorderInterpolationMethod;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
eDestDepth	ETcVnElementType [▶ 191]	目标图像深度
nXOrder	UDINT	x 导数的阶数（必须 < nKernelSize）
nYOrder	UDINT	y 导数的阶数（必须 < nKernelSize）
nKernelSize	UDINT	扩展索贝尔（Sobel）内核的大小（3、5、7...31）
fScale	LREAL	导数计算值的比例系数
fDelta	LREAL	将结果存储到目标位置之前，加入这些结果的 Delta 值
eBorderType	ETcVnBorderInterpolationMethod [▶ 160]	图像边框处理
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数`F_VN_SobelFilterExp`是F_VN_SobelFilter [▶ 1312]的专家版本。它包含额外的参数。

参数

原始图像

原始图像`ipSrcImage`可以有任何格式。

结果图像

结果图像`ipDestImage`返回过滤结果，且其格式与原始图像`ipSrcImage`相同。

结果深度

结果深度`eDestDepth`定义了结果图像`ipDestImage`的元素类型。一个更大的元素类型可以显示更多的信息。

导数阶数

导数阶数nXOrder和nYOrder定义在 X 和 Y 方向上哪个导数用于创建过滤器。

内核大小

索贝尔内核的大小 nKernelSize 决定了影响一个像素结果值的区域有多大。

缩放

索贝尔运算的结果乘以缩放系数fScale。

Delta

在通过fScale缩放后，常量 deltafDelta被添加到索贝尔运算的结果中。

边界推断

边界推断eBorderType的方法定义了如何将不存在的像素外推到图像边界之外，以计算图像边界的过滤值。更多细节，请参见ETcVnBorderInterpolationMethod [▶ 160]。

应用

```
hr := F_VN_SobelFilterExp(
    ipSrcImage   :=  ipImageIn,
    ipDestImage  :=  ipImageRes,
    eDestDepth   :=  TCVN_ET_USINT,
    nXOrder      :=  1,
    nYOrder      :=  1,
  nKernelSize := 3,
  fScale    := 1,
  fDelta    := 0,
  eBorderType := TCVN_BIM_DEFAULT,
    hrPrev       :=  hr,
);
```

样本

- 边缘检测过滤器 [▶ 2787]

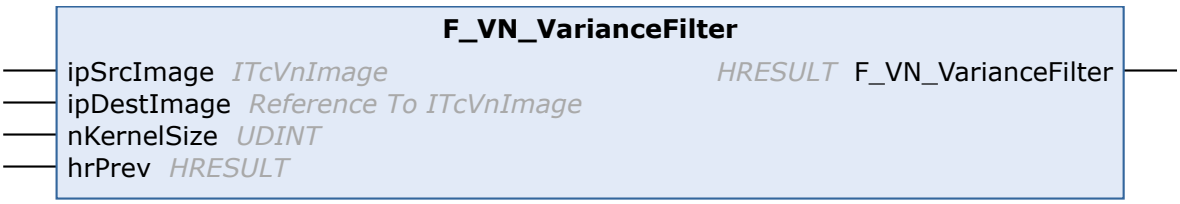
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.27 F_VN_VarianceFilter



Calculates the local variance of an image.

语法

定义：

```
FUNCTION F_VN_VarianceFilter : HRESULT
VAR_INPUT
    ipSrcImage   : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
```

```
    nKernelSize : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像。仅支持 TCVN_ET_USINT 类型。
ipDestImage	引用 ITcVnImage [▶ 407]	TCVN_ET_UINT 类型的目标图像（如果需要，将创建适当的目标图像）。
nKernelSize	UDINT	内核大小
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

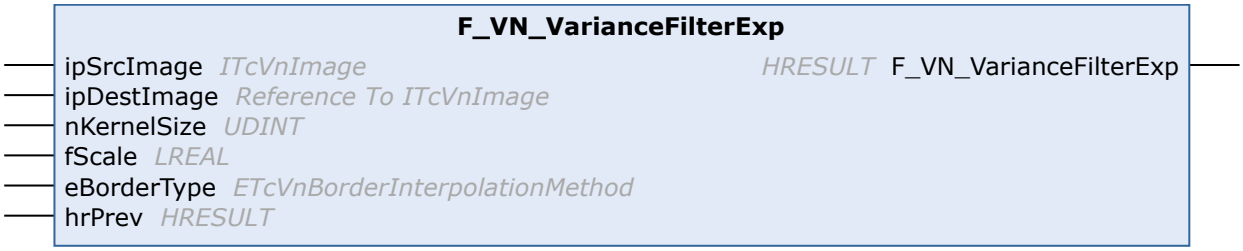
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.17.28 F_VN_VarianceFilterExp



计算图像的局部方差。（专家函数）

语法

定义：

```
FUNCTION F_VN_VarianceFilterExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    nKernelSize : UDINT;
    fScale      : LREAL;
    eBorderType : ETcVnBorderInterpolationMethod;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像。仅支持 TCVN_ET_USINT 类型。
ipDestImage	引用 ITcVnImage [▶ 407]	TCVN_ET_UINT 类型的目标图像（如果需要，将创建适当的目标图像）。

名称	类型	描述
nKernelSize	UDINT	内核大小
fScale	LREAL	方差计算值的比例系数
eBorderType	ETcVnBorderInterpolationMethod [▶ 160]	图像边框处理
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.18 图像分割

6.1.4.18.1 F_VN_AdaptiveThreshold



对灰度图像应用自适应阈值。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_AdaptiveThreshold : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    fMaxValue   : LREAL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道，USINT 类型元素）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
fMaxValue	LREAL	为阈值条件为 true 的像素分配的值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

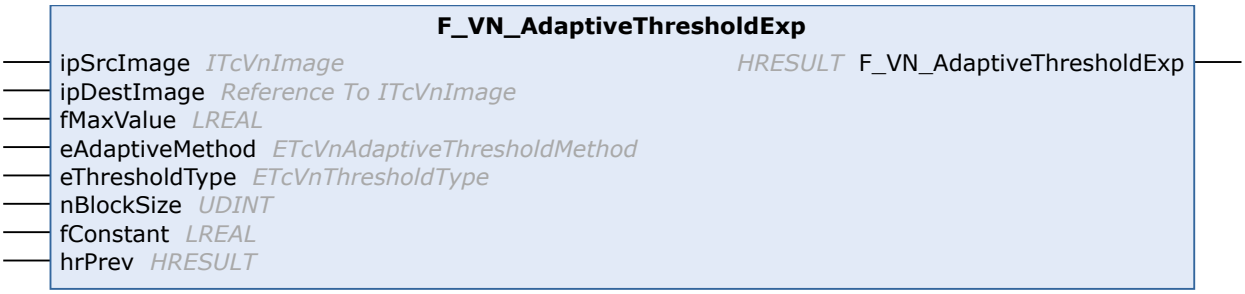
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.18.2 F_VN_AdaptiveThresholdExp



Apply an adaptive threshold to a gray level image. (expert function)
Can return partial results when canceled by Watchdog.

语法

定义：

```
FUNCTION F_VN_AdaptiveThresholdExp : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipDestImage     : Reference To ITcVnImage;
    fMaxValue       : LREAL;
    eAdaptiveMethod : ETcVnAdaptiveThresholdMethod;
    eThresholdType  : ETcVnThresholdType;
    nBlockSize      : UDINT;
    fConstant       : LREAL;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道，USINT 类型元素）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
fMaxValue	LREAL	为阈值条件为 true 的像素分配的值
eAdaptiveMethod	ETcVnAdaptiveThresholdMethod [▶ 158]	要应用的自适应阈值方法（MEAN：阈值将以（x，y）的 nBlockSize x nBlockSize 邻域的平均值减去 fConstant 来计算。GAUSSIAN：阈值是（x,y）的 nBlockSize x nBlockSize 邻域的加权和再减去 fConstant（与高斯窗口互相关）。
eThresholdType	ETcVnThresholdType [▶ 213]	要应用的阈值类型（仅支持 BINARY 和 BINARY_INV）
nBlockSize	UDINT	用于计算局部阈值的像素邻域大小（3、5、7...）
fConstant	LREAL	从像素邻域的加权平均值中减去的常数，从而得出局部阈值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数F_VN_AdaptiveThresholdExp是F_VN_AdaptiveThreshold [▶ 1316]的专家版本。它包含额外的参数。

参数

输入图像

输入图像ipSrcImage必须是位深为 8 位的单通道灰度图像（USINT）。

结果图像

二进制结果图像ipDestImage的格式与输入图像相同。像素的值是0或目标值fMaxValue。

目标值

目标值fMaxValue定义了应该为满足阈值标准的像素赋哪个值。如果结果图像只由黑白像素组成，那么必须在这里设置输入图像的元素类型的最大值。这对应于(2^d)-1，其中 d 代表元素类型的位深；例如，(2^8)-1=255，8 位图像。

阈值法

阈值方法 eAdaptiveMethod 定义了自适应阈值是作为平均数还是作为高斯加权和进行计算。在枚举ETcVnAdaptiveThresholdMethod [▶ 158]中，有两个值TCVN_ATM_MEAN和TCVN_ATM_GAUSSIAN可供使用。

阈值反转

参数 eThresholdType 指定结果图像是否应该被反转。因此，目前只支持来自枚举ETcVnThresholdType [▶ 213]的两个值TCVN_TT_BINARY和TCVN_TT_BINARY_INV。

阈值邻域大小

参数nBlockSize定义了计算局部阈值时考虑的邻域的大小。参数必须是一个奇数（3、5、7 等）。

阈值调整

从每个局部平均值中减去常量值fConstant。其结果是用于计算结果图像的局部阈值。

应用

```
hr := F_VN_AdaptiveThreshold(
    ipSrcImage      :=  ipImageIn,
    ipDestImage     :=  ipImageRes,
    fMaxValue       :=  255, // max value of 8-bit image: (2^8)-1=255
    eAdaptiveMethod :=  TCVN_ATM_MEAN,
    eThresholdType  :=  TCVN_TT_BINARY_INV,
    nBlockSize      :=  5,
    fConstant       :=  10,
    hrPrev          :=  hr
);
```

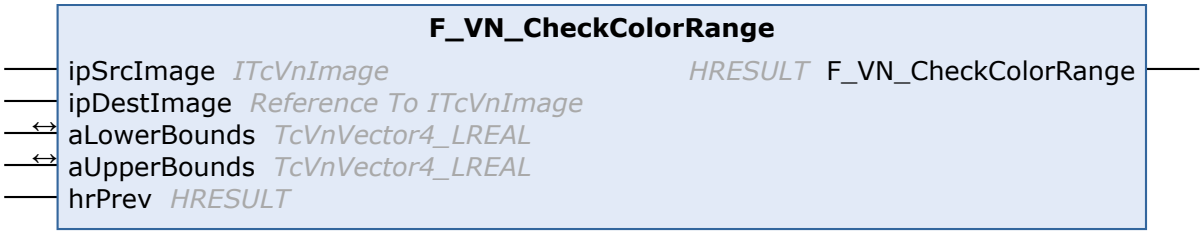
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.18.3 F_VN_CheckColorRange



检查图像的像素值是否在给定范围内。目标图像的大小与源图像相同，但只有一个通道包含 8 位像素元素。如果源图像的相应像素在检查范围内，则其元素设置为 255，否则设置为 0。

语法

定义：

```
FUNCTION F_VN_CheckColorRange : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipDestImage   : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aLowerBounds : TcVnVector4_LREAL;
    aUpperBounds : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aLowerBounds	TcVnVector4_LREAL [▶ 153]	通道下界（未使用的通道将被忽略。）
aUpperBounds	TcVnVector4_LREAL [▶ 153]	通道上界（未使用的通道将被忽略。）

返回值

HRESULT [▶ 135]

更多信息

F_VN_CheckColorRange 函数根据特定值范围对输入图像进行分割。这意味着使用哪种颜色空间并不重要，因为分割过程纯粹基于数值。

- 处于该范围内的图像像素被赋值为最大值 255（白色）。
- 处于该色谱之外的图像像素被赋值为 0（黑色）。

参数

输入图像

输入图像 `ipSrcImage` 可以是任何格式。

结果图像

结果图像 `ipDestImage` 返回分割后的单通道 8 位二值图像。

值范围

分割的数值范围由下限 `aLowerBounds` 和上限 `aUpperBounds` 定义。

对于循环值范围（例如 HSV 空间中的色调通道），下限值可以大于上限值，以便跨越零点对颜色范围进行分割。从下限 [250, ...] 到上限 [5, ...] 的范围对应于函数调用中 [250, ...] - [MAX, ...] 与 [0, ...] - [5, ...] 的组合。这样可以通过函数调用直接选择位于值范围过渡区域的色调，例如色调通道中的红色范围（0° 附近的 +-20°）。

初始化边界向量

务必确保边界向量初始化的元素个数至少与输入图像的通道数相同。例如，如果您对一个 4 通道图像（例如带 α 通道的图像）进行分割，但只使用三个元素来初始化颜色向量，那么这些向量的第四个元素为 0，结果图像变为黑色。

应用

为了实现可靠的彩色分割，通常建议在加法性 RGB 空间以外的颜色空间中进行处理。例如，在 HSV 空间中，色调（H）与明度（V）和饱和度（S）相分离，这意味着颜色区分通常对光照波动不太敏感。

例如，对 HSV 图像（`HSV_FULL`）进行红色光谱分割的操作如下所示：

```
VAR
    aLower      :   TcVnVector4_LREAL := [250, 150, 100];
    aUpper      :   TcVnVector4_LREAL := [5, 255, 255];
END_VAR

hr := F_VN_CheckColorRange(
    ipSrcImage      := ipImageIn,
    ipDestImage     := ipImageRes,
    aLowerBounds    := aLower,
    aUpperBounds    := aUpper,
    hrPrev          := hr);
```

示例

- 检查颜色范围 [▶ 2790]

相关函数

- F_VN_DoubleThreshold [▶ 1320] 基于双阈值利用形态学重建进行二进制分割
- F_VN_Threshold [▶ 1322] 基于阈值进行二进制分割
- F_VN_AdaptiveThreshold [▶ 1316] 用于二进制自适应分割
- F_VN_CheckColorRange [▶ 1319] 基于色谱进行二进制分割
- F_VN_ReferenceColorSimilarity [▶ 1260] 基于颜色模型进行二进制分割

所需的许可证

TC3 Vision Base

6.1.4.18.4 F_VN_DoubleThreshold



Apply a double threshold (also called hysteresis threshold) using morphological reconstruction.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_DoubleThreshold : HRESULT
VAR_INPUT
    ipSrcImage     : ITcVnImage;
    ipDestImage    : Reference To ITcVnImage;
    fLowThreshold  : LREAL;
    fHighThreshold : LREAL;
    fMaxValue      : LREAL;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT、UINT、INT、REAL 或 LREAL，1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
fLowThreshol	LREAL	决定已找到区域形状的低阈值
fHighThreshold	LREAL	选择已找到区域的高阈值（如果应用高阈值，则每个区域必须至少包含一个像素。）
fMaxValue	LREAL	最大像素值
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_DoubleThreshold` 利用形态重建计算双阈值。该函数希望搜索到的图像分割比背景更亮，与函数 `F_VN_Threshold` 相比，可以对确定的图像分割进行额外的滤波。

参数

输入图像

输入图像 `ipSrcImage` 必须是单通道灰度图像。

结果图像

二进制结果图像 `ipDestImage` 与输入图像格式相同。像素值为 `0` 或目标值为 `fMaxValue`。

阈值下限

较低的阈值决定了所发现线段的形状，例如函数 `F_VN_Threshold`。

阈值上限

阈值上限用于标记图像分割，然后根据阈值下限计算其形状。这样，灰度值大于阈值下限而小于阈值上限的不需要的片段就可以被滤波掉。

目标值

目标值 `fMaxValue` 定义了要分配给满足相应阈值标准的像素的值。如果结果图像仅由黑白像素组成，则必须在此处设置输入图像元素类型的最大值。这相当于 `(2^d)-1`，其中 d 是元素类型的位深度；例如 `(2^8)-1=255` 表示 8 位图像。

应用

```
hr := F_VN_DoubleThreshold(
    ipSrcImage      := ipSrcImage,
    ipDestImage     := ipDestImage,
    fLowThreshold   := fLowThreshold,
    fHighThreshold  := fHighThreshold,
    fMaxValue       := 255,
    hrPrev          := hr);
```

相关函数

- F_VN_DoubleThreshold [▶ 1320] 基于双阈值利用形态学重建进行二进制分割
- F_VN_Threshold [▶ 1322] 基于阈值进行二进制分割
- F_VN_AdaptiveThreshold [▶ 1316] 用于二进制自适应分割
- F_VN_CheckColorRange [▶ 1319] 基于色谱进行二进制分割
- F_VN_ReferenceColorSimilarity [▶ 1260] 基于颜色模型进行二进制分割

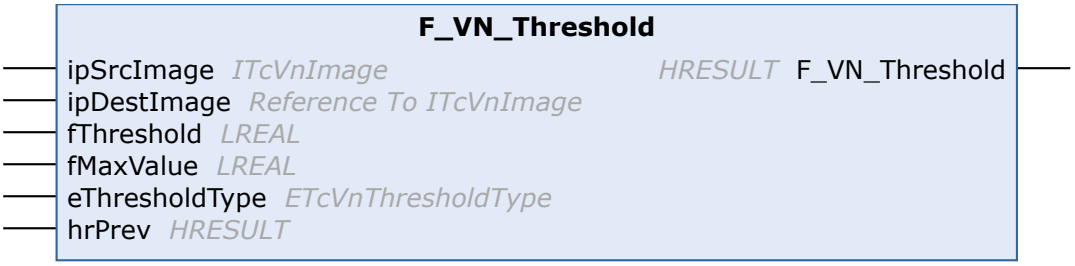
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.18.5 F_VN_Threshold



根据大津（Otsu）算法应用固定阈值或动态阈值。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_Threshold : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipDestImage     : Reference To ITcVnImage;
    fThreshold      : LREAL;
    fMaxValue       : LREAL;
    eThresholdType  : ETcVnThresholdType;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（对于大津（Otsu）和三角阈值类型，仅支持 1 通道 TCVN_ET_USINT 类型）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
fThreshold	LREAL	固定阈值（选择动态阈值时未使用）
fMaxValue	LREAL	最大像素值
eThresholdType	ETcVnThresholdType [▶ 213]	要应用的阈值类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

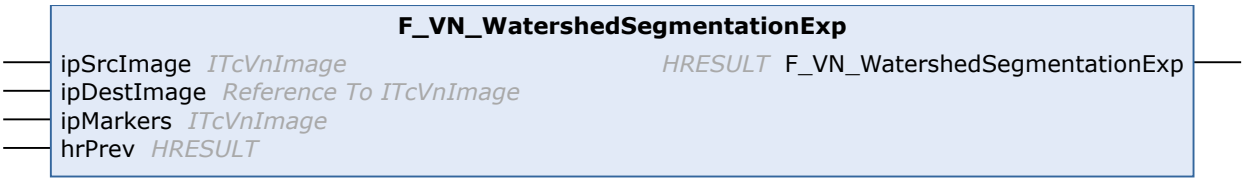
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.18.6 F_VN_WatershedSegmentationExp



Apply a marker-based watershed segmentation. (expert function)

语法

定义：

```
FUNCTION F_VN_WatershedSegmentationExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    ipMarkers   : ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，3 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（DINT，1 通道。如有需要，将创建适当的图像）。
ipMarkers	ITcVnImage [▶ 407]	标记图像（DINT，1 通道）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.18.7 F_VN_SauvolaThreshold



Apply Sauvola local threshold to source image.

语法

定义：

```
FUNCTION F_VN_SauvolaThreshold : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    pipDestImage    : Reference To ITcVnImage;
    nBlockSize      : UDINT;
    fParamK         : LREAL;
    fMaxSigma       : LREAL;
    eThresholdType  : ETcVnThresholdType;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT、UINT、INT、REAL 或 LREAL，1 通道）
pipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的单通道目标图像。））
nBlockSize	UDINT	用于计算局部阈值的像素邻域大小（3、5、7...）
fParamK	LREAL	在范围（0，1）内取正值的参数 K。建议 K 在范围 [0.2，0.5] 内
fMaxSigm	LREAL	标准偏差的最大值，必须严格为正值。（对于灰度文档，fMaxSigma = 128）。
eThresholdType	ETcVnThresholdType [▶ 213]	要应用的阈值类型。支持 TCVN_TT_BINARY 和 TCVN_TT_BINARY_INV。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

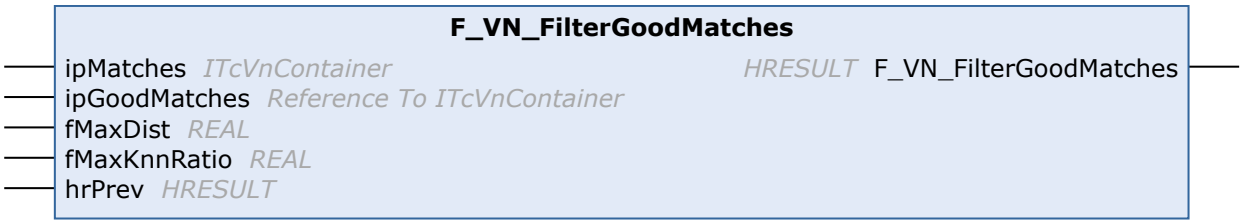
TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19 关键点特征

6.1.4.19.1 F_VN_FilterGoodMatches



Filter the descriptor matches and return only good ones.

语法

定义：

```
FUNCTION F_VN_FilterGoodMatches : HRESULT
VAR_INPUT
    ipMatches     : ITcVnContainer;
    ipGoodMatches : Reference To ITcVnContainer;
    fMaxDist      : REAL;
    fMaxKnnRatio  : REAL;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMatches	ITcVnContainer [▶ 365]	包含描述符匹配项的容器（ContainerType_Vector_TcVnDMatch 或 ContainerType_Vector_Vector_TcVnDMatch）
ipGoodMatches	引用 ITcVnContainer [▶ 365]	返回包含良好匹配项的容器（ContainerType_Vector_TcVnDMatch）
fMaxDist	REAL	允许的最大描述符距离（-1 禁用此过滤标准）
fMaxKnnRatio	REAL	第一和第二最佳匹配结果之间允许的最大距离比（[0...1]，仅用于 knn 匹配结果，-1 则禁用此过滤标准）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

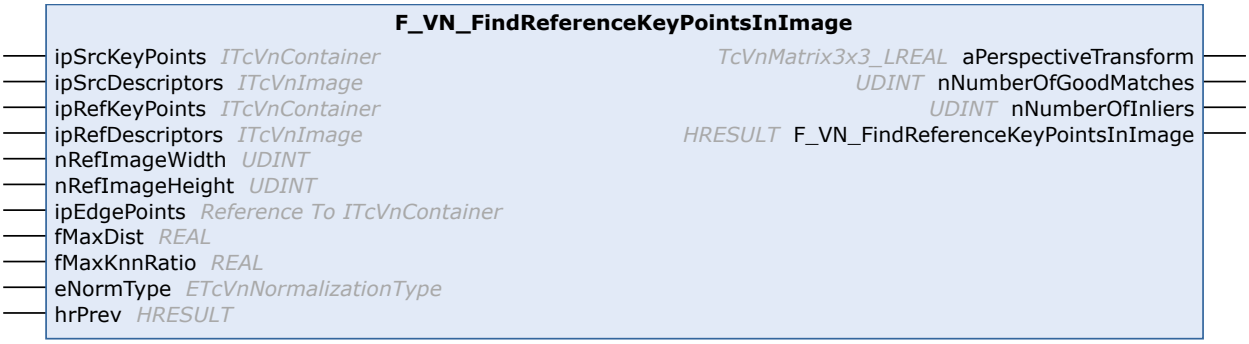
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.2 F_VN_FindReferenceKeyPointsInImage



Searches a reference image, represented by its keypoints and corresponding descriptors, in a given source image.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_FindReferenceKeyPointsInImage : HRESULT
VAR_INPUT
    ipSrcKeyPoints        : ITcVnContainer;
    ipSrcDescriptors      : ITcVnImage;
    ipRefKeyPoints        : ITcVnContainer;
    ipRefDescriptors      : ITcVnImage;
    nRefImageWidth        : UDINT;
    nRefImageHeight       : UDINT;
    ipEdgePoints          : Reference To ITcVnContainer;
    fMaxDist              : REAL;
    fMaxKnnRatio          : REAL;
    eNormType             : ETcVnNormalizationType;
    hrPrev                : HRESULT;
END_VAR
VAR_OUTPUT
    aPerspectiveTransform : TcVnMatrix3x3_LREAL;
    nNumberOfGoodMatches  : UDINT;
    nNumberOfInliers      : UDINT;
END_VAR
```

输入

名称	类型	描述
ipSrcKeyPoints	ITcVnContainer [▶ 365]	源图像的关键点，例如（ContainerType_Vector_TcVnKeyPoint）
ipSrcDescriptors	ITcVnImage [▶ 407]	源图像关键点的描述符
ipRefKeyPoints	ITcVnContainer [▶ 365]	参考图像的关键点，例如（ContainerType_Vector_TcVnKeyPoint）
ipRefDescriptors	ITcVnImage [▶ 407]	参考图像关键点的描述符
nRefImageWidth	UDINT	参考图像的宽度，单位：像素
nRefImageHeight	UDINT	参考图像的高度，单位：像素
ipEdgePoints	引用 ITcVnContainer [▶ 365]	返回一个容器，其中包含转换为源图像坐标的 4 个参考图像边缘点（ContainerType_Vector_TcVnPoint2_REAL）

名称	类型	描述
fMaxDist	REAL	允许的最大描述符距离（-1 禁用此过滤标准）
fMaxKnnRatio	REAL	第一和第二最佳匹配结果之间允许的最大距离比（[0...1]，-1 则禁用此过滤标准）
eNormType	ETcVnNormalizationType [▶ 203]	用于描述符匹配的归一化类型（AKAZE、ORB 和 BRISK 建议使用 HAMMING。如果 ORB nBriefPoints 参数为 3 或 4，建议 ORB 使用 HAMMING2。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输出

名称	类型	描述
aPerspectiveTransform	TcVnMatrix3x3_LREAL [▶ 153]	返回透视变换矩阵，该矩阵将参考点变换为源点
nNumberOfGoodMatches	UDINT	返回良好匹配项的数量
nNumberOfInliers	UDINT	返回内群匹配项的数量

返回值

HRESULT [▶ 135]

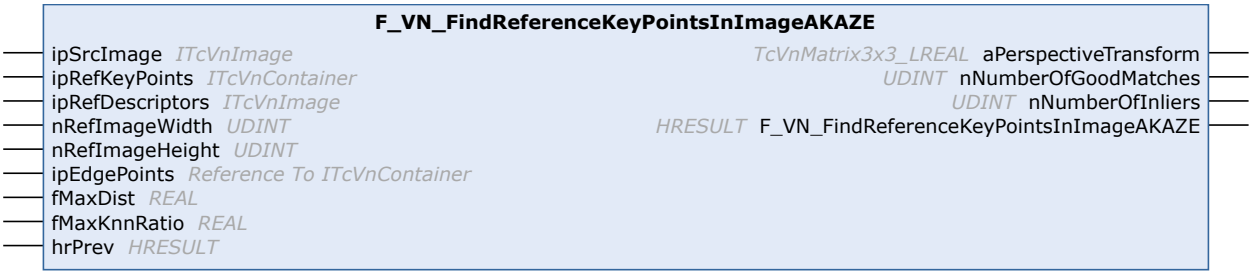
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.3 F_VN_FindReferenceKeyPointsInImageAKAZE



Searches a reference image, represented by its AKAZE keypoints and corresponding descriptors, in a given source image (uses the default TcVnParamsAKAZE parameters).
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_FindReferenceKeyPointsInImageAKAZE : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
    ipRefKeyPoints        : ITcVnContainer;
    ipRefDescriptors      : ITcVnImage;
```

```
    nRefImageWidth         : UDINT;
    nRefImageHeight        : UDINT;
    ipEdgePoints           : Reference To ITcVnContainer;
    fMaxDist               : REAL;
    fMaxKnnRatio           : REAL;
    hrPrev                 : HRESULT;
END_VAR
VAR_OUTPUT
    aPerspectiveTransform : TcVnMatrix3x3_LREAL;
    nNumberOfGoodMatches  : UDINT;
    nNumberOfInliers      : UDINT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipRefKeyPoints	ITcVnContainer [▶ 365]	参考图像的关键点，例如使用 F_VN_KeyPointsAndDescriptorsAKAZE （ContainerType_Vector_TcVnKeyPoint）计算得出的关键点
ipRefDescriptors	ITcVnImage [▶ 407]	参考图像关键点的描述符，例如使用 F_VN_KeyPointsAndDescriptorsAKAZE 计算得出的描述符
nRefImageWidth	UDINT	参考图像的宽度，单位：像素
nRefImageHeight	UDINT	参考图像的高度，单位：像素
ipEdgePoints	引用 ITcVnContainer [▶ 365]	返回一个容器，其中包含转换为 ipSrcImage 坐标的 4 个参考图像边缘点（ContainerType_Vector_TcVnPoint2_REAL）
fMaxDist	REAL	允许的最大描述符距离（-1 禁用此过滤标准）
fMaxKnnRatio	REAL	第一和第二最佳匹配结果之间允许的最大距离比（[0...1]，-1 则禁用此过滤标准）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输出

名称	类型	描述
aPerspectiveTransform	TcVnMatrix3x3_LREAL [▶ 153]	返回透视变换矩阵，该矩阵将参考点变换为源点
nNumberOfGoodMatches	UDINT	返回良好匹配项的数量
nNumberOfInliers	UDINT	返回内群匹配项的数量

返回值

HRESULT [▶ 135]

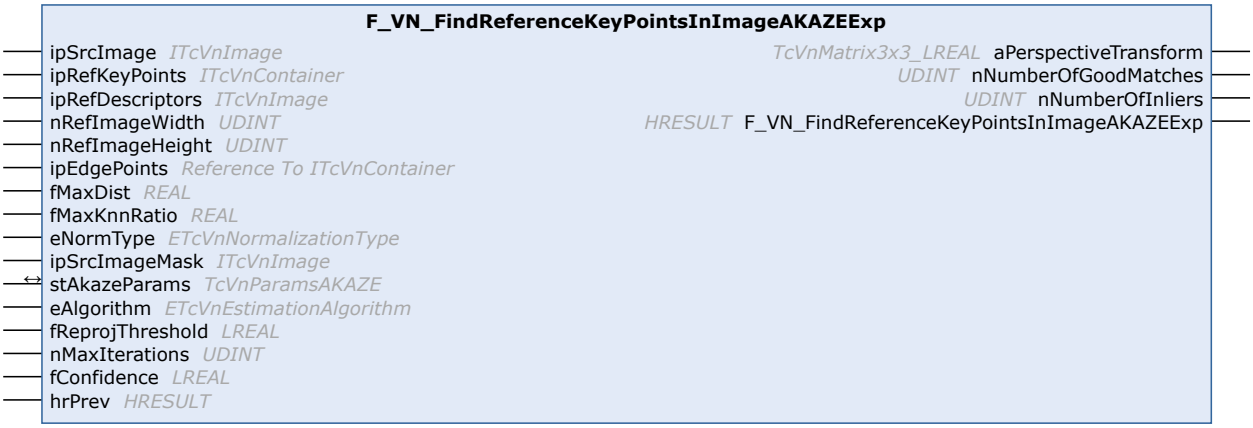
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.4 F_VN_FindReferenceKeyPointsInImageAKAZEExp



Searches a reference image, represented by its AKAZE keypoints and corresponding descriptors, in a given source image. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_FindReferenceKeyPointsInImageAKAZEExp : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
    ipRefKeyPoints        : ITcVnContainer;
    ipRefDescriptors      : ITcVnImage;
    nRefImageWidth        : UDINT;
    nRefImageHeight       : UDINT;
    ipEdgePoints          : Reference To ITcVnContainer;
    fMaxDist              : REAL;
    fMaxKnnRatio          : REAL;
    eNormType             : ETcVnNormalizationType;
    ipSrcImageMask        : ITcVnImage;
END_VAR
VAR_IN_OUT
    stAkazeParams         : TcVnParamsAKAZE;
END_VAR
VAR_INPUT
    eAlgorithm            : ETcVnEstimationAlgorithm;
    fReprojThreshold      : LREAL;
    nMaxIterations        : UDINT;
    fConfidence           : LREAL;
    hrPrev                : HRESULT;
END_VAR
VAR_OUTPUT
    aPerspectiveTransform : TcVnMatrix3x3_LREAL;
    nNumberOfGoodMatches  : UDINT;
    nNumberOfInliers      : UDINT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipRefKeyPoints	ITcVnContainer [▶ 365]	参考图像的关键点，例如使用 F_VN_KeyPointsAndDescriptorsAKAZE （ContainerType_Vector_TcVnKeyPoint）计算得出的关键点
ipRefDescriptors	ITcVnImage [▶ 407]	参考图像关键点的描述符，例如使用 F_VN_KeyPointsAndDescriptorsAKAZE 计算得出的描述符
nRefImageWidth	UDINT	参考图像的宽度，单位：像素
nRefImageHeight	UDINT	参考图像的高度，单位：像素
ipEdgePoints	引用 ITcVnContainer [▶ 365]	返回一个容器，其中包含转换为 ipSrcImage 坐标的 4 个参考图像边缘点（ContainerType_Vector_TcVnPoint2_REAL）

名称	类型	描述
fMaxDist	REAL	允许的最大描述符距离（-1 禁用此过滤标准）
fMaxKnnRatio	REAL	第一和第二最佳匹配结果之间允许的最大距离比（[0...1]，-1 则禁用此过滤标准）
eNormType	ETcVnNormalizationType [▶ 203]	用于描述符匹配的归一化类型
ipSrcImageMask	ITcVnImage [▶ 407]	该掩膜用于指定在哪里查找 ipSrcImage 的关键点（如果不需要掩膜，则设为 0）
eAlgorithm	ETcVnEstimationAlgorithm [▶ 192]	用于计算两个点集之间透视变换的估计算法
fReprojThreshold	LREAL	将点对视为内群时允许的最大重投影误差（仅在 eAlgorithm 为 RANSAC、RHO 时使用）
nMaxIterations	UDINT	最大 RANSAC 迭代次数
fConfidence	LREAL	置信度（0..1）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stAkazeParams	TcVnParamsAKAZE [▶ 228]	用于为 ipSrcImage 配置关键点和描述符计算的参数（产生的描述符必须与 ipRefDescriptors 兼容！）

输出

名称	类型	描述
aPerspectiveTransform	TcVnMatrix3x3_LREAL [▶ 153]	返回透视变换矩阵，该矩阵将参考点变换为源点
nNumberOfGoodMatches	UDINT	返回良好匹配项的数量
nNumberOfInliers	UDINT	返回内群匹配项的数量

返回值

HRESULT [▶ 135]

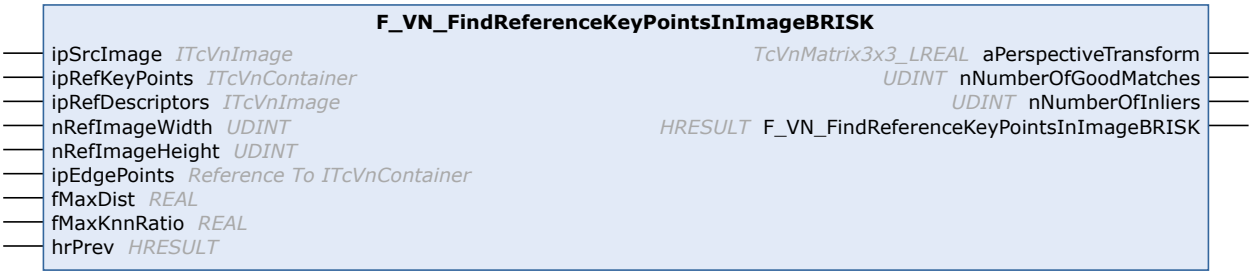
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.5 F_VN_FindReferenceKeyPointsInImageBRISK



Searches a reference image, represented by its BRISK keypoints and corresponding descriptors, in a given source image (uses the default TcVnParamsBRISK parameters).
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_FindReferenceKeyPointsInImageBRISK : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
    ipRefKeyPoints        : ITcVnContainer;
    ipRefDescriptors      : ITcVnImage;
    nRefImageWidth        : UDINT;
    nRefImageHeight       : UDINT;
    ipEdgePoints          : Reference To ITcVnContainer;
    fMaxDist              : REAL;
    fMaxKnnRatio          : REAL;
    hrPrev                : HRESULT;
END_VAR
VAR_OUTPUT
    aPerspectiveTransform : TcVnMatrix3x3_LREAL;
    nNumberOfGoodMatches  : UDINT;
    nNumberOfInliers      : UDINT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipRefKeyPoints	ITcVnContainer [▶ 365]	参考图像的关键点，例如使用 F_VN_KeyPointsAndDescriptorsBRISK（ContainerType_Vector_TcVnKeyPoint）计算得出的关键点
ipRefDescriptors	ITcVnImage [▶ 407]	参考图像关键点的描述符，例如使用 F_VN_KeyPointsAndDescriptorsBRISK 计算得出的描述符
nRefImageWidth	UDINT	参考图像的宽度，单位：像素
nRefImageHeight	UDINT	参考图像的高度，单位：像素
ipEdgePoints	引用 ITcVnContainer [▶ 365]	返回一个容器，其中包含转换为 ipSrcImage 坐标的 4 个参考图像边缘点（ContainerType_Vector_TcVnPoint2_REAL）
fMaxDist	REAL	允许的最大描述符距离（-1 禁用此过滤标准）
fMaxKnnRatio	REAL	第一和第二最佳匹配结果之间允许的最大距离比（[0...1]，-1 则禁用此过滤标准）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输出

名称	类型	描述
aPerspectiveTransform	TcVnMatrix3x3_LREAL [▶ 153]	返回透视变换矩阵，该矩阵将参考点变换为源点
nNumberOfGoodMatches	UDINT	返回良好匹配项的数量
nNumberOfInliers	UDINT	返回内群匹配项的数量

返回值

HRESULT [▶ 135]

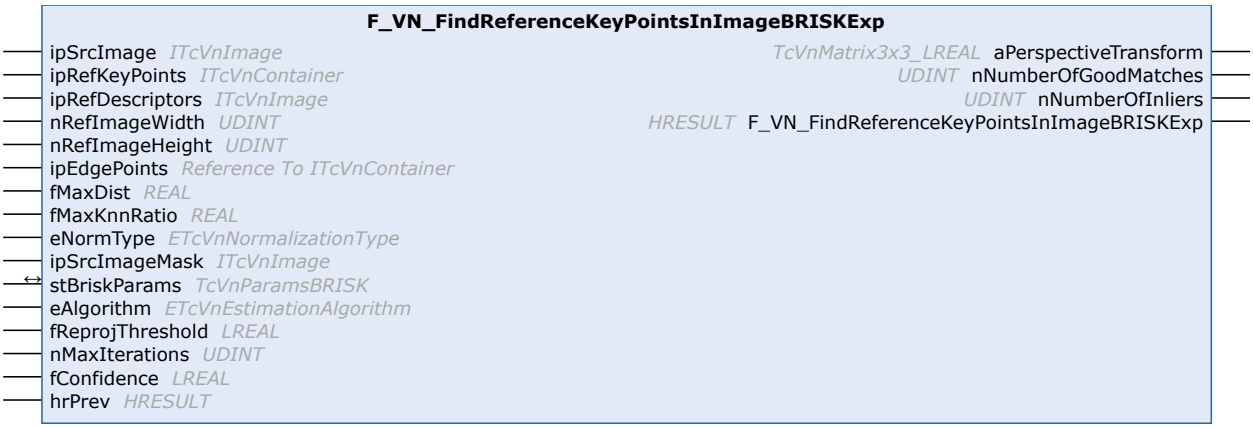
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.6 F_VN_FindReferenceKeyPointsInImageBRISKExp



Searches a reference image, represented by its BRISK keypoints and corresponding descriptors, in a given source image. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_FindReferenceKeyPointsInImageBRISKExp : HRESULT
VAR_INPUT
    ipSrcImage              : ITcVnImage;
    ipRefKeyPoints          : ITcVnContainer;
    ipRefDescriptors        : ITcVnImage;
    nRefImageWidth          : UDINT;
    nRefImageHeight         : UDINT;
    ipEdgePoints            : Reference To ITcVnContainer;
    fMaxDist                : REAL;
    fMaxKnnRatio            : REAL;
    eNormType               : ETcVnNormalizationType;
    ipSrcImageMask          : ITcVnImage;
END_VAR
VAR_IN_OUT
    stBriskParams           : TcVnParamsBRISK;
END_VAR
```

```
VAR_INPUT
    eAlgorithm            : ETcVnEstimationAlgorithm;
    fReprojThreshold      : LREAL;
    nMaxIterations        : UDINT;
    fConfidence           : LREAL;
    hrPrev                : HRESULT;
END_VAR
VAR_OUTPUT
    aPerspectiveTransform : TcVnMatrix3x3_LREAL;
    nNumberOfGoodMatches  : UDINT;
    nNumberOfInliers      : UDINT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipRefKeyPoints	ITcVnContainer [▶ 365]	参考图像的关键点，例如使用 F_VN_KeyPointsAndDescriptorsBRISK（ContainerType_Vector_TcVnKeyPoint）计算得出的关键点
ipRefDescriptors	ITcVnImage [▶ 407]	参考图像关键点的描述符，例如使用 F_VN_KeyPointsAndDescriptorsBRISK 计算得出的描述符
nRefImageWidth	UDINT	参考图像的宽度，单位：像素
nRefImageHeight	UDINT	参考图像的高度，单位：像素
ipEdgePoints	引用 ITcVnContainer [▶ 365]	返回一个容器，其中包含转换为 ipSrcImage 坐标的 4 个参考图像边缘点（ContainerType_Vector_TcVnPoint2_REAL）
fMaxDist	REAL	允许的最大描述符距离（-1 禁用此过滤标准）
fMaxKnnRatio	REAL	第一和第二最佳匹配结果之间允许的最大距离比（[0...1]，-1 则禁用此过滤标准）
eNormType	ETcVnNormalizationType [▶ 203]	用于描述符匹配的归一化类型
ipSrcImageMask	ITcVnImage [▶ 407]	该掩膜用于指定在哪里查找 ipSrcImage 的关键点（如果不需要掩膜，则设为 0）
eAlgorithm	ETcVnEstimationAlgorithm [▶ 192]	用于计算两个点集之间透视变换的估计算法
fReprojThreshold	LREAL	将点对视为内群时允许的最大重投影误差（仅在 eAlgorithm 为 RANSAC、RHO 时使用）
nMaxIterations	UDINT	最大 RANSAC 迭代次数
fConfidence	LREAL	置信度（0..1）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stBriskParams	TcVnParamsBRISK [▶ 231]	用于为 ipSrcImage 配置关键点和描述符计算的参数（产生的描述符必须与 ipRefDescriptors 兼容！）

输出

名称	类型	描述
aPerspectiveTransform	TcVnMatrix3x3_LREAL [▶ 153]	返回透视变换矩阵，该矩阵将参考点变换为源点
nNumberOfGoodMatches	UDINT	返回良好匹配项的数量
nNumberOfInliers	UDINT	返回内群匹配项的数量

返回值

HRESULT [▶ 135]

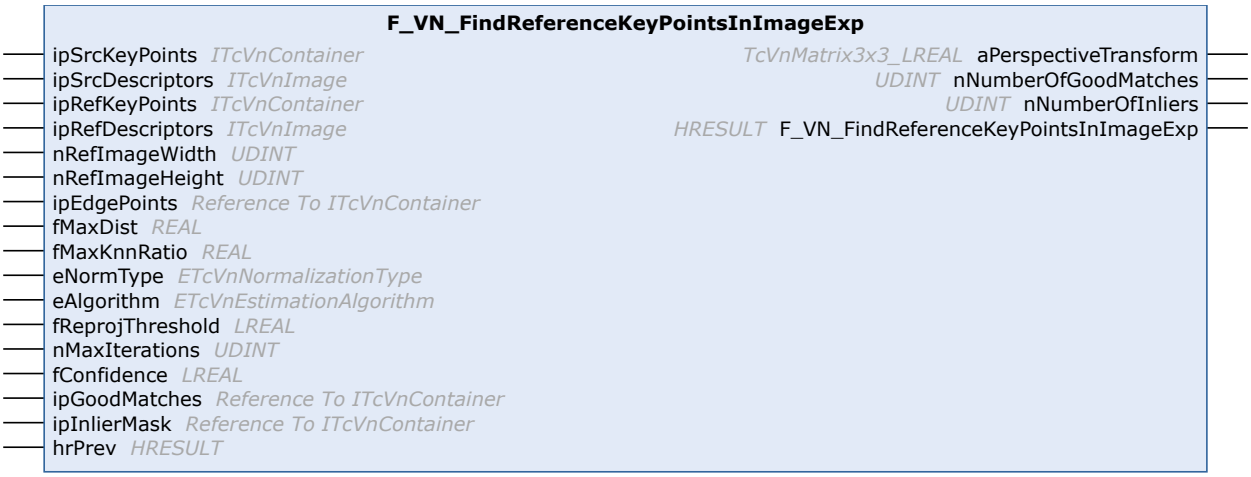
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.7 F_VN_FindReferenceKeyPointsInImageExp



Searches a reference image, represented by its keypoints and corresponding descriptors, in a given source image. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_FindReferenceKeyPointsInImageExp : HRESULT
VAR_INPUT
    ipSrcKeyPoints        : ITcVnContainer;
    ipSrcDescriptors      : ITcVnImage;
    ipRefKeyPoints        : ITcVnContainer;
    ipRefDescriptors      : ITcVnImage;
    nRefImageWidth        : UDINT;
    nRefImageHeight       : UDINT;
    ipEdgePoints          : Reference To ITcVnContainer;
    fMaxDist              : REAL;
    fMaxKnnRatio          : REAL;
    eNormType             : ETcVnNormalizationType;
    eAlgorithm            : ETcVnEstimationAlgorithm;
    fReprojThreshold      : LREAL;
```

```
    nMaxIterations        : UDINT;
    fConfidence           : LREAL;
    ipGoodMatches         : Reference To ITcVnContainer;
    ipInlierMask          : Reference To ITcVnContainer;
    hrPrev                : HRESULT;
END_VAR
VAR_OUTPUT
    aPerspectiveTransform : TcVnMatrix3x3_LREAL;
    nNumberOfGoodMatches  : UDINT;
    nNumberOfInliers      : UDINT;
END_VAR
```

输入

名称	类型	描述
ipSrcKeyPoints	ITcVnContainer [▶ 365]	源图像的关键点，例如（ContainerType_Vector_TcVnKeyPoint）
ipSrcDescriptors	ITcVnImage [▶ 407]	源图像关键点的描述符
ipRefKeyPoints	ITcVnContainer [▶ 365]	参考图像的关键点（ContainerType_Vector_TcVnKeyPoint）
ipRefDescriptors	ITcVnImage [▶ 407]	参考图像关键点的描述符
nRefImageWidth	UDINT	参考图像的宽度，单位：像素
nRefImageHeight	UDINT	参考图像的高度，单位：像素
ipEdgePoints	引用 ITcVnContainer [▶ 365]	返回一个容器，其中包含转换为源图像坐标的 4 个参考图像边缘点（ContainerType_Vector_TcVnPoint2_REAL）
fMaxDist	REAL	允许的最大描述符距离（-1 禁用此过滤标准）
fMaxKnnRatio	REAL	第一和第二最佳匹配结果之间允许的最大距离比（[0...1]，-1 则禁用此过滤标准）
eNormType	ETcVnNormalizationType [▶ 203]	用于描述符匹配的归一化类型（AKAZE、ORB 和 BRISK 建议使用 HAMMING。如果 ORB nBriefPoints 参数为 3 或 4，建议 ORB 使用 HAMMING2。）
eAlgorithm	ETcVnEstimationAlgorithm [▶ 192]	用于计算两个点集之间透视变换的估计算法
fReprojThreshold	LREAL	将点对视为内群时允许的最大重投影误差（仅在 eAlgorithm 为 RANSAC、RHO 时使用）
nMaxIterations	UDINT	最大 RANSAC 迭代次数
fConfidence	LREAL	置信度（0..1）
ipGoodMatches	引用 ITcVnContainer [▶ 365]	返回包含良好匹配项的容器（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnDMatch）
ipInlierMask	引用 ITcVnContainer [▶ 365]	返回标记内群值的掩码（可选，如果不需要，则设为 0；ContainerType_Vector_SINT；仅适用于 RANSAC 和 LMEDS）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输出

名称	类型	描述
aPerspectiveTransform	TcVnMatrix3x3_LREAL [▶ 153]	返回透视变换矩阵，该矩阵将参考点变换为源点
nNumberOfGoodMatches	UDINT	返回良好匹配项的数量

名称	类型	描述
nNumberOfInliers	UDINT	返回内群匹配项的数量

返回值

HRESULT [▶ 135]

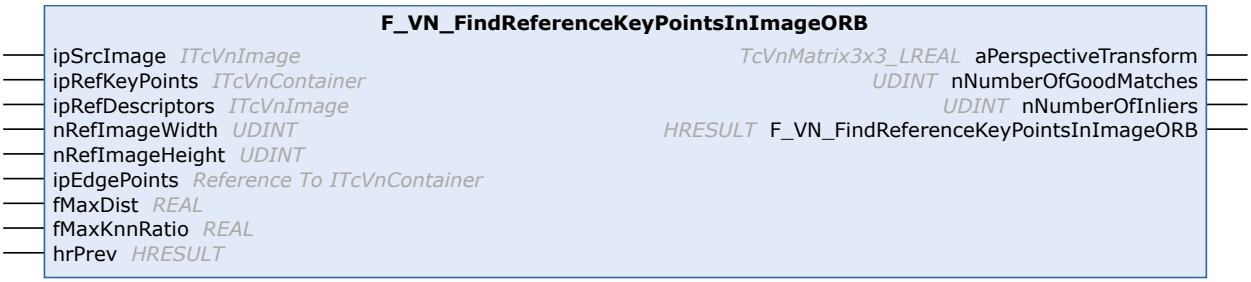
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.8 F_VN_FindReferenceKeyPointsInImageORB



Searches a reference image, represented by its ORB keypoints and corresponding descriptors, in a given source image (uses the default TcVnParamsORB parameters).
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_FindReferenceKeyPointsInImageORB : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
    ipRefKeyPoints        : ITcVnContainer;
    ipRefDescriptors      : ITcVnImage;
    nRefImageWidth        : UDINT;
    nRefImageHeight       : UDINT;
    ipEdgePoints          : Reference To ITcVnContainer;
    fMaxDist              : REAL;
    fMaxKnnRatio          : REAL;
    hrPrev                : HRESULT;
END_VAR
VAR_OUTPUT
    aPerspectiveTransform : TcVnMatrix3x3_LREAL;
    nNumberOfGoodMatches  : UDINT;
    nNumberOfInliers      : UDINT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipRefKeyPoints	ITcVnContainer [▶ 365]	参考图像的关键点，例如使用 F_VN_KeyPointsAndDescriptorsORB （ContainerType_Vector_TcVnKeyPoint）计算得出的关键点

名称	类型	描述
ipRefDescriptors	ITcVnImage [▶ 407]	参考图像关键点的描述符，例如使用 F_VN_KeyPointsAndDescriptorsORB 计算得出的描述符
nRefImageWidth	UDINT	参考图像的宽度，单位：像素
nRefImageHeight	UDINT	参考图像的高度，单位：像素
ipEdgePoints	引用 ITcVnContainer [▶ 365]	返回一个容器，其中包含转换为 ipSrcImage 坐标的 4 个参考图像边缘点（ContainerType_Vector_TcVnPoint2_REAL）
fMaxDist	REAL	允许的最大描述符距离（-1 禁用此过滤标准）
fMaxKnnRatio	REAL	第一和第二最佳匹配结果之间允许的最大距离比（[0...1]，-1 则禁用此过滤标准）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输出

名称	类型	描述
aPerspectiveTransform	TcVnMatrix3x3_LREAL [▶ 153]	返回透视变换矩阵，该矩阵将参考点变换为源点
nNumberOfGoodMatches	UDINT	返回良好匹配项的数量
nNumberOfInliers	UDINT	返回内群匹配项的数量

返回值

HRESULT [▶ 135]

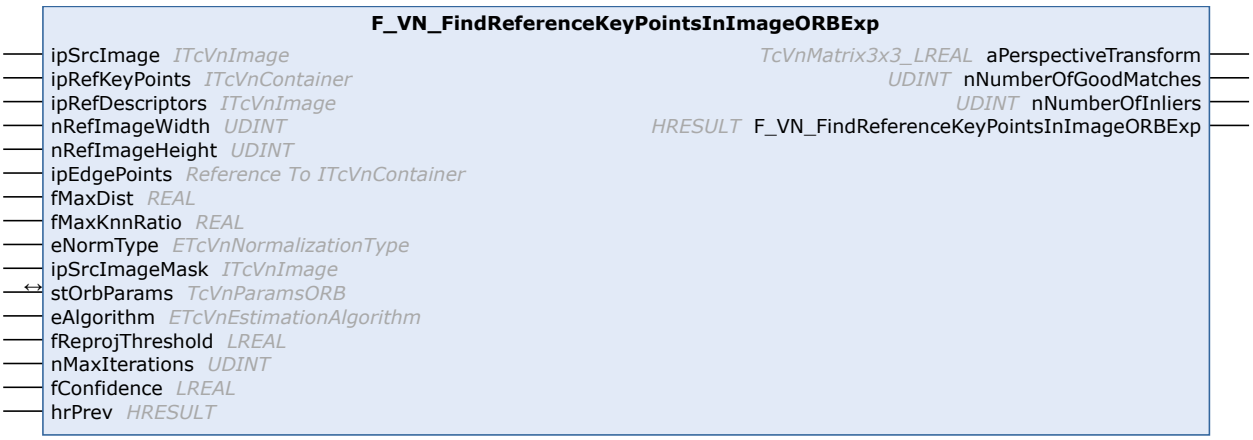
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.9 F_VN_FindReferenceKeyPointsInImageORBExp



Searches a reference image, represented by its ORB keypoints and corresponding descriptors, in a given source image. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_FindReferenceKeyPointsInImageORBExp : HRESULT
VAR_INPUT
    ipSrcImage            : ITcVnImage;
    ipRefKeyPoints        : ITcVnContainer;
    ipRefDescriptors      : ITcVnImage;
    nRefImageWidth        : UDINT;
    nRefImageHeight       : UDINT;
    ipEdgePoints          : Reference To ITcVnContainer;
    fMaxDist              : REAL;
    fMaxKnnRatio          : REAL;
    eNormType             : ETcVnNormalizationType;
    ipSrcImageMask        : ITcVnImage;
END_VAR
VAR_IN_OUT
    stOrbParams           : TcVnParamsORB;
END_VAR
VAR_INPUT
    eAlgorithm            : ETcVnEstimationAlgorithm;
    fReprojThreshold      : LREAL;
    nMaxIterations        : UDINT;
    fConfidence           : LREAL;
    hrPrev                : HRESULT;
END_VAR
VAR_OUTPUT
    aPerspectiveTransform : TcVnMatrix3x3_LREAL;
    nNumberOfGoodMatches  : UDINT;
    nNumberOfInliers      : UDINT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipRefKeyPoints	ITcVnContainer [▶ 365]	参考图像的关键点，例如使用 F_VN_KeyPointsAndDescriptorsORB （ContainerType_Vector_TcVnKeyPoint）计算得出的关键点
ipRefDescriptors	ITcVnImage [▶ 407]	参考图像关键点的描述符，例如使用 F_VN_KeyPointsAndDescriptorsORB 计算得出的描述符
nRefImageWidth	UDINT	参考图像的宽度，单位：像素
nRefImageHeight	UDINT	参考图像的高度，单位：像素
ipEdgePoints	引用 ITcVnContainer [▶ 365]	返回一个容器，其中包含转换为 ipSrcImage 坐标的 4 个参考图像边缘点（ContainerType_Vector_TcVnPoint2_REAL）
fMaxDist	REAL	允许的最大描述符距离（-1 禁用此过滤标准）
fMaxKnnRatio	REAL	第一和第二最佳匹配结果之间允许的最大距离比（[0...1]，-1 则禁用此过滤标准）
eNormType	ETcVnNormalizationType [▶ 203]	用于描述符匹配的归一化类型（建议 ORB 使用 HAMMING，如果 ORB nBriefPoints 参数为 3 或 4，则使用 HAMMING2）
ipSrcImageMask	ITcVnImage [▶ 407]	该掩膜用于指定在哪里查找 ipSrcImage 的关键点（如果不需要掩膜，则设为 0）
eAlgorithm	ETcVnEstimationAlgorithm [▶ 192]	用于计算两个点集之间透视变换的估计算法
fReprojThreshold	LREAL	将点对视为内群时允许的最大重投影误差（仅在 eAlgorithm 为 RANSAC、RHO 时使用）

名称	类型	描述
nMaxIterations	UDINT	最大 RANSAC 迭代次数
fConfidence	LREAL	置信度（0..1）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stOrbParams	TcVnParamsORB [▶ 233]	用于为 ipSrcImage 配置关键点和描述符计算的参数（产生的描述符必须与 ipRefDescriptors 兼容！）

输出

名称	类型	描述
aPerspectiveTransform	TcVnMatrix3x3_LREAL [▶ 153]	返回透视变换矩阵，该矩阵将参考点变换为源点
nNumberOfGoodMatches	UDINT	返回良好匹配项的数量
nNumberOfInliers	UDINT	返回内群匹配项的数量

返回值

HRESULT [▶ 135]

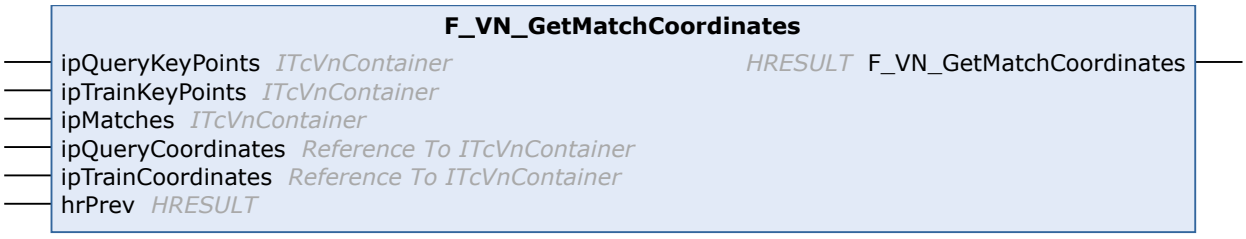
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.10 F_VN_GetMatchCoordinates



Return the coordinates of keypoints that match each other.

语法

定义：

```
FUNCTION F_VN_GetMatchCoordinates : HRESULT
VAR_INPUT
    ipQueryKeyPoints    : ITcVnContainer;
    ipTrainKeyPoints    : ITcVnContainer;
```

```
    ipMatches          : ITcVnContainer;
    ipQueryCoordinates : Reference To ITcVnContainer;
    ipTrainCoordinates : Reference To ITcVnContainer;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipQueryKeyPoints	ITcVnContainer [▶ 365]	包含 CTcVnContainer_Vector_TcVnKeyPoint 类型的查询关键点（即参考图像/模板图像的关键点）的容器
ipTrainKeyPoints	ITcVnContainer [▶ 365]	包含 CTcVnContainer_Vector_TcVnKeyPoint 类型的训练关键点（即源图像/输入图像的关键点）的容器
ipMatches	ITcVnContainer [▶ 365]	查询关键点与训练关键点匹配的容器（CTcVnContainer_Vector_TcVnDMatch）。
ipQueryCoordinates	引用 ITcVnContainer [▶ 365]	返回一个包含关键点坐标的容器，这些关键点同时存在于查询关键点和 ipMatches 中（CTcVnContainer_Vector_TcVnPoint2_REAL）
ipTrainCoordinates	引用 ITcVnContainer [▶ 365]	返回一个包含关键点坐标的容器，这些关键点同时存在于训练关键点和 ipMatches 中（CTcVnContainer_Vector_TcVnPoint2_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

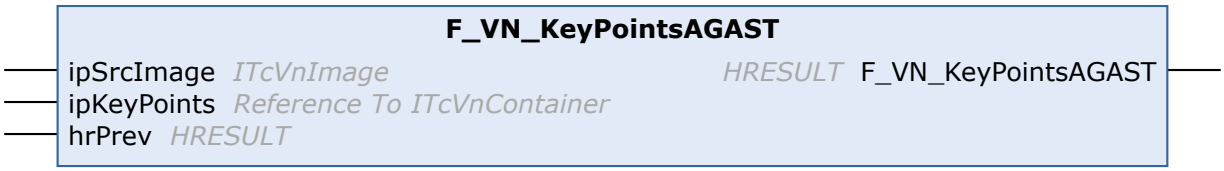
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.11 F_VN_KeyPointsAGAST



Detects keypoints using the AGAST method.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsAGAST : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipKeyPoints : Reference To ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

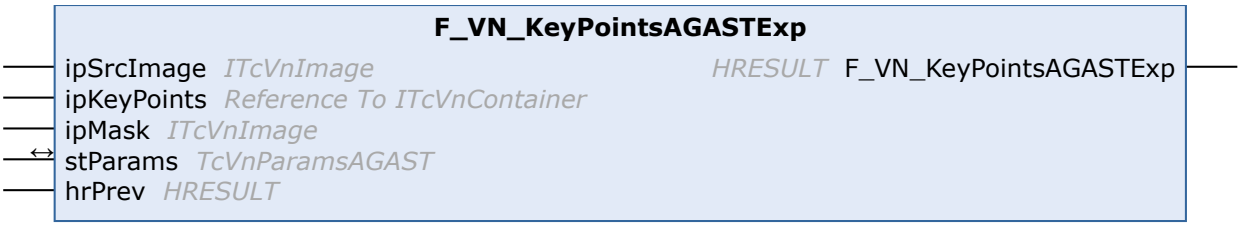
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.12 F_VN_KeyPointsAGASTExp



Detects keypoints using the AGAST method. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsAGASTExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipKeyPoints : Reference To ITcVnContainer;
    ipMask      : ITcVnImage;
END_VAR
VAR_IN_OUT
    stParams    : TcVnParamsAGAST;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）

名称	类型	描述
ipMask	ITcVnImage [▶ 407]	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stParams	TcVnParamsAGAST [▶ 228]	其它专家参数

返回值

HRESULT [▶ 135]

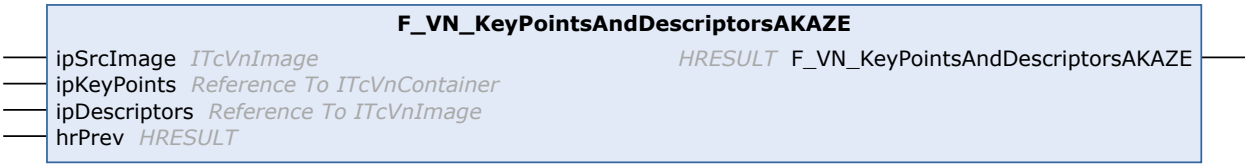
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.13 F_VN_KeyPointsAndDescriptorsAKAZE



Detects keypoints and compute descriptors using the AKAZE method.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsAndDescriptorsAKAZE : HRESULT
VAR_INPUT
    ipSrcImage     : ITcVnImage;
    ipKeyPoints    : Reference To ITcVnContainer;
    ipDescriptors : Reference To ITcVnImage;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipDescriptors	引用 ITcVnImage [▶ 407]	描述符图像（如果不需要，则设为 0；每行 1 个描述符；如有需要，将创建适当的图像。）

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

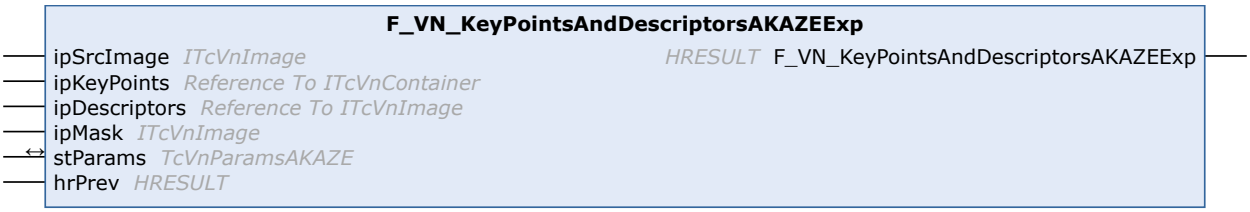
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.14 F_VN_KeyPointsAndDescriptorsAKAZEExp



Detects keypoints and compute descriptors using the AKAZE method. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsAndDescriptorsAKAZEExp : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipKeyPoints   : Reference To ITcVnContainer;
    ipDescriptors : Reference To ITcVnImage;
    ipMask        : ITcVnImage;
END_VAR
VAR_IN_OUT
    stParams      : TcVnParamsAKAZE;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipDescriptors	引用 ITcVnImage [▶ 407]	描述符图像（如果不需要，则设为 0；每行 1 个描述符；如有需要，将创建适当的图像。）
ipMask	ITcVnImage [▶ 407]	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stParams	TcVnParamsAKAZE [▶ 228]	其它专家参数

返回值

HRESULT [▶ 135]

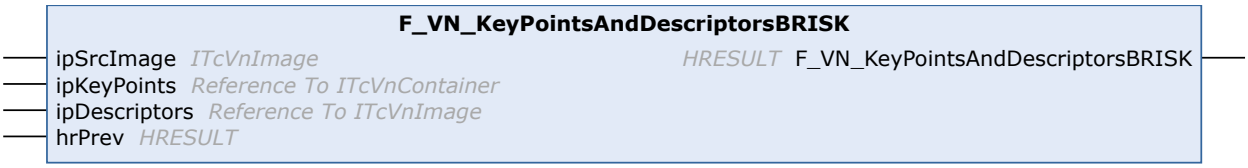
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.15 F_VN_KeyPointsAndDescriptorsBRISK



Detects keypoints and compute descriptors using the BRISK method.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsAndDescriptorsBRISK : HRESULT
VAR_INPUT
    ipSrcImage     : ITcVnImage;
    ipKeyPoints    : Reference To ITcVnContainer;
    ipDescriptors : Reference To ITcVnImage;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipDescriptors	引用 ITcVnImage [▶ 407]	描述符图像（如果不需要，则设为 0；每行 1 个描述符；如有需要，将创建适当的图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.16 F_VN_KeyPointsAndDescriptorsBRISKExp



Detects keypoints and compute descriptors using the BRISK method. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsAndDescriptorsBRISKExp : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipKeyPoints   : Reference To ITcVnContainer;
    ipDescriptors : Reference To ITcVnImage;
    ipMask        : ITcVnImage;
END_VAR
VAR_IN_OUT
    stParams      : TcVnParamsBRISK;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipDescriptors	引用 ITcVnImage [▶ 407]	描述符图像（如果不需要，则设为 0；每行 1 个描述符；如有需要，将创建适当的图像。）
ipMask	ITcVnImage [▶ 407]	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stParams	TcVnParamsBRISK [▶ 231]	其它专家参数

返回值

HRESULT [▶ 135]

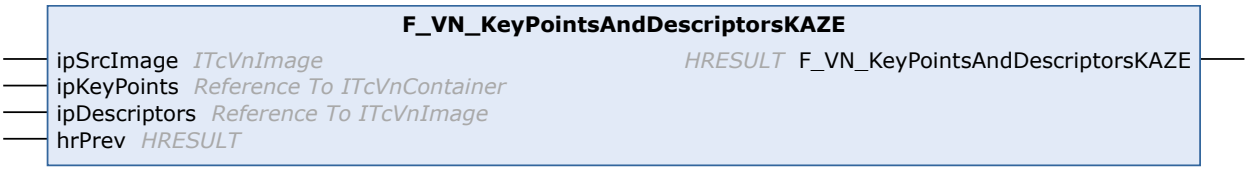
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.17 F_VN_KeyPointsAndDescriptorsKAZE



Detects keypoints and compute descriptors using the KAZE method.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsAndDescriptorsKAZE : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipKeyPoints   : Reference To ITcVnContainer;
    ipDescriptors : Reference To ITcVnImage;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipDescriptors	引用 ITcVnImage [▶ 407]	描述符图像（如果不需要，则设为 0；每行 1 个描述符；如有需要，将创建适当的图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

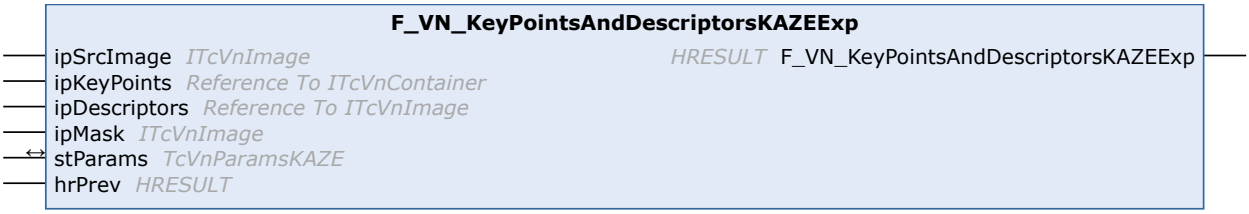
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.18 F_VN_KeyPointsAndDescriptorsKAZEExp



Detects keypoints and compute descriptors using the KAZE method. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsAndDescriptorsKAZEExp : HRESULT
VAR_INPUT
    ipSrcImage     : ITcVnImage;
    ipKeyPoints    : Reference To ITcVnContainer;
    ipDescriptors : Reference To ITcVnImage;
    ipMask         : ITcVnImage;
END_VAR
VAR_IN_OUT
    stParams       : TcVnParamsKAZE;
END_VAR
VAR_INPUT
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipDescriptors	引用 ITcVnImage [▶ 407]	描述符图像（如果不需要，则设为 0；每行 1 个描述符；如有需要，将创建适当的图像。）
ipMask	ITcVnImage [▶ 407]	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stParams	TcVnParamsKAZE [▶ 232]	其它专家参数

返回值

HRESULT [▶ 135]

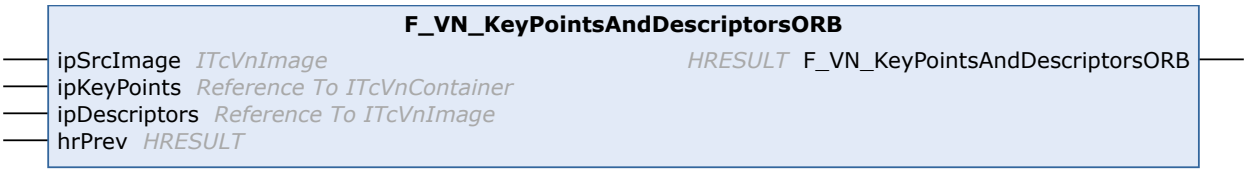
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.19 F_VN_KeyPointsAndDescriptorsORB



Detects keypoints and compute descriptors using the ORB method.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsAndDescriptorsORB : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipKeyPoints   : Reference To ITcVnContainer;
    ipDescriptors : Reference To ITcVnImage;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipDescriptors	引用 ITcVnImage [▶ 407]	描述符图像（如果不需要，则设为 0；每行 1 个描述符；如有需要，将创建适当的图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.20 F_VN_KeyPointsAndDescriptorsORBExp



Detect keypoints and compute descriptors using the ORB method. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsAndDescriptorsORBExp : HRESULT
VAR_INPUT
    ipSrcImage    : ITcVnImage;
    ipKeyPoints   : Reference To ITcVnContainer;
    ipDescriptors : Reference To ITcVnImage;
    ipMask        : ITcVnImage;
END_VAR
VAR_IN_OUT
    stParams      : TcVnParamsORB;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipDescriptors	引用 ITcVnImage [▶ 407]	描述符图像（如果不需要，则设为 0；每行 1 个描述符；如有需要，将创建适当的图像。）
ipMask	ITcVnImage [▶ 407]	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stParams	TcVnParamsORB [▶ 233]	其它专家参数

返回值

HRESULT [▶ 135]

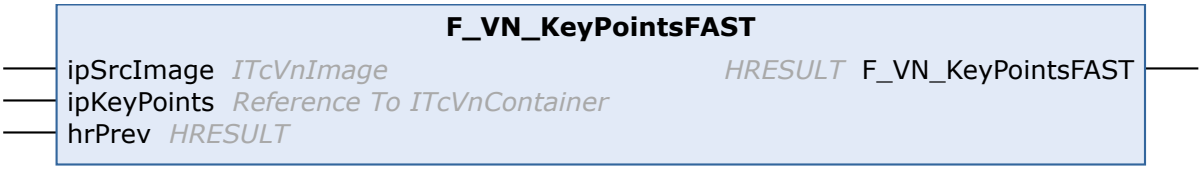
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.21 F_VN_KeyPointsFAST



Detects keypoints using the FAST method.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsFAST : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipKeyPoints : Reference To ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（单通道）。
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

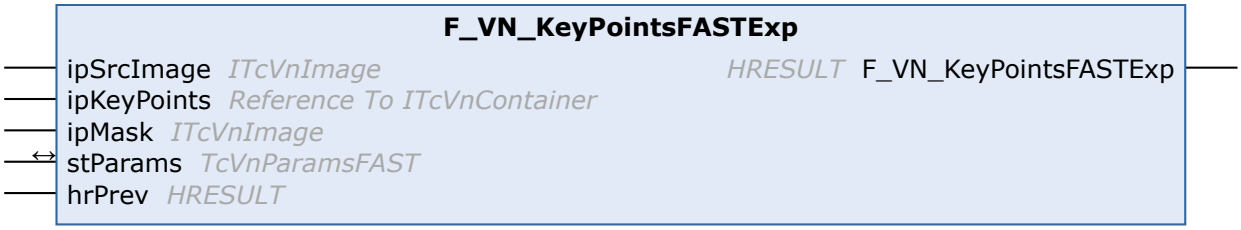
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.22 F_VN_KeyPointsFASTExp



Detects keypoints using the FAST method. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsFASTExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipKeyPoints : Reference To ITcVnContainer;
    ipMask      : ITcVnImage;
END_VAR
VAR_IN_OUT
    stParams    : TcVnParamsFAST;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipMask	ITcVnImage [▶ 407]	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stParams	TcVnParamsFAST [▶ 231]	其它专家参数

返回值

HRESULT [▶ 135]

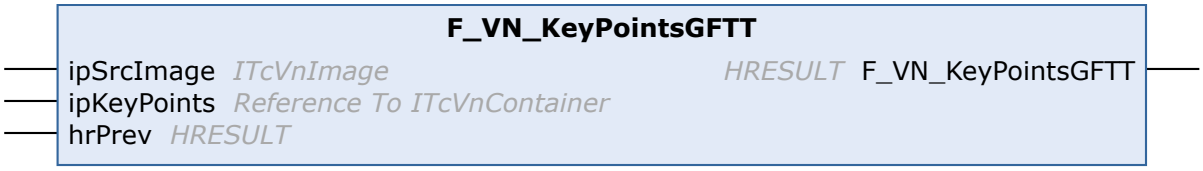
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.23 F_VN_KeyPointsGFTT



Detects keypoints using the GFTT method, which detects strong corners.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsGFTT : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipKeyPoints : Reference To ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

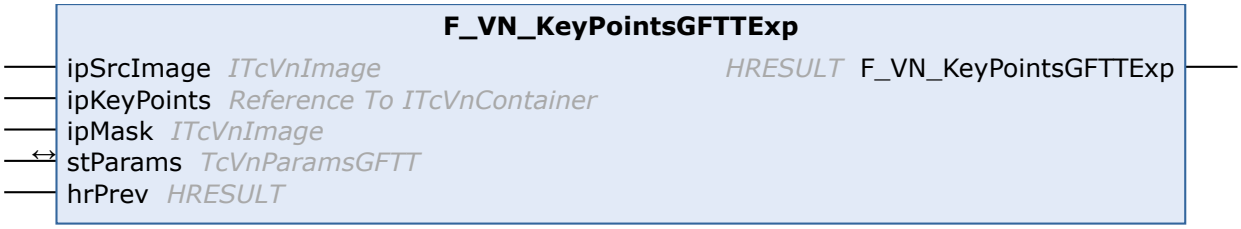
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.24 F_VN_KeyPointsGFTTExp



Detects keypoints using the GFTT method, which detects strong corners. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsGFTTExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipKeyPoints : Reference To ITcVnContainer;
    ipMask      : ITcVnImage;
END_VAR
VAR_IN_OUT
    stParams    : TcVnParamsGFTT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipMask	ITcVnImage [▶ 407]	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stParams	TcVnParamsGFTT [▶ 231]	其它专家参数

返回值

HRESULT [▶ 135]

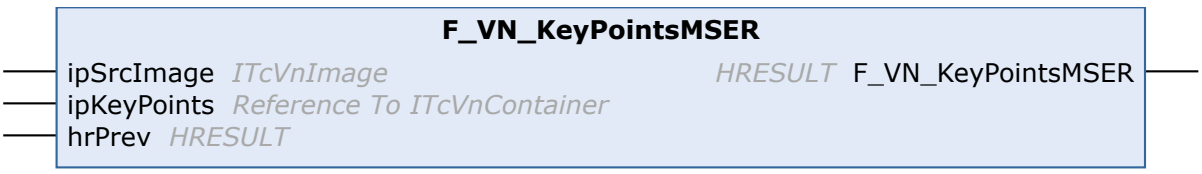
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.25 F_VN_KeyPointsMSER



Detects keypoints using the MSER method.

语法

定义：

```
FUNCTION F_VN_KeyPointsMSER : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipKeyPoints : Reference To ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

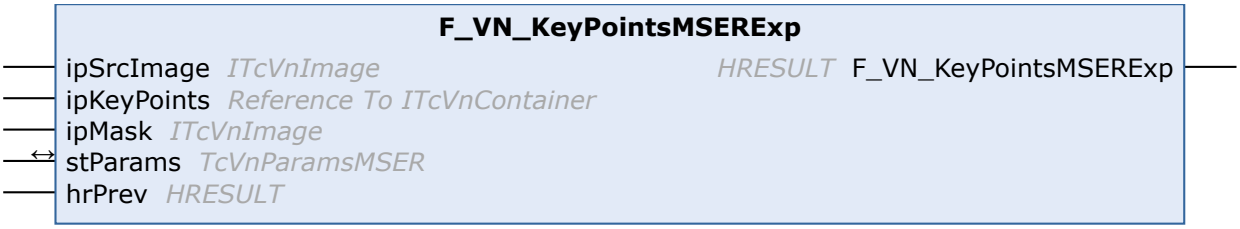
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.26 F_VN_KeyPointsMSERExp



Detects keypoints using the MSER method. (expert function)

语法

定义：

```
FUNCTION F_VN_KeyPointsMSERExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipKeyPoints : Reference To ITcVnContainer;
    ipMask      : ITcVnImage;
END_VAR
VAR_IN_OUT
    stParams    : TcVnParamsMSER;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipMask	ITcVnImage [▶ 407]	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stParams	TcVnParamsMSER [▶ 232]	其它专家参数

返回值

HRESULT [▶ 135]

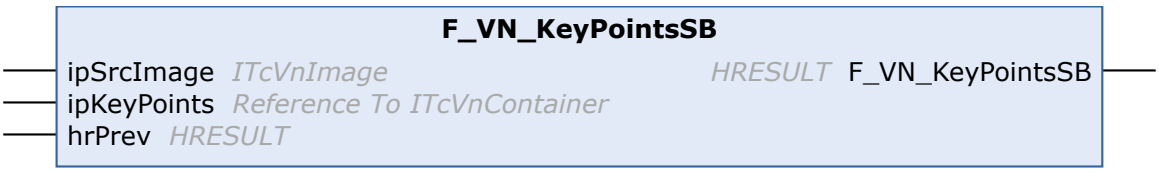
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.27 F_VN_KeyPointsSB



Detects keypoints using a Simple Blob method - several iterations apply different thresholds to source image - connected components (blobs) are detected - the center and radius of the blobs are returned as keypoints
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsSB : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipKeyPoints : Reference To ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

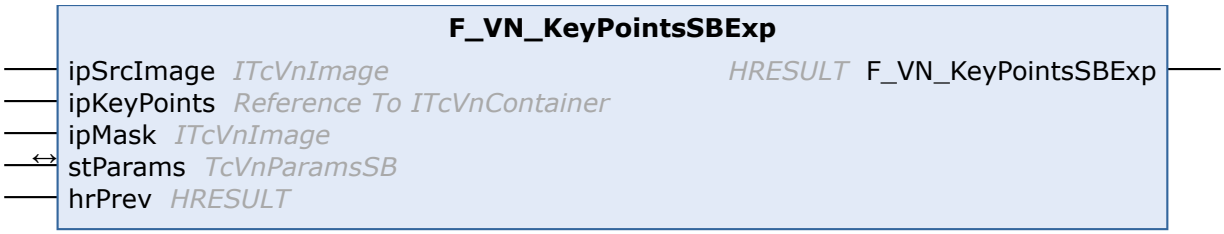
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.28 F_VN_KeyPointsSBExp



Detects keypoints using a Simple Blob method (expert function) - several iterations apply different thresholds to source image - connected components (blobs) are detected - the center and radius of the blobs are returned as keypoints
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_KeyPointsSBExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipKeyPoints : Reference To ITcVnContainer;
    ipMask      : ITcVnImage;
END_VAR
VAR_IN_OUT
    stParams    : TcVnParamsSB;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像

名称	类型	描述
ipKeyPoints	引用 ITcVnContainer [▶ 365]	返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipMask	ITcVnImage [▶ 407]	该掩膜用于指定在哪里查找关键点（可选，如果不需要，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stParams	TcVnParamsSB [▶ 234]	用于过滤检测到的斑点的几个参数

返回值

HRESULT [▶ 135]

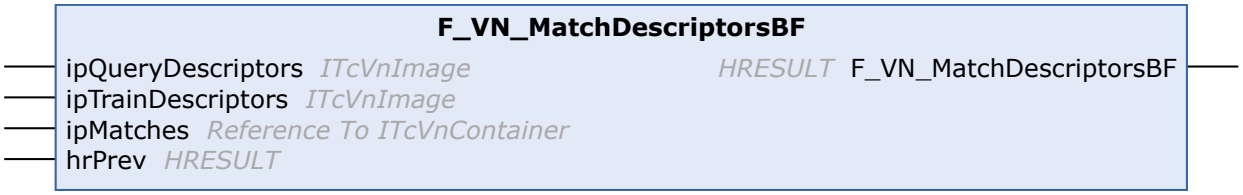
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.29 F_VN_MatchDescriptorsBF



Match descriptors using a brute force approach.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_MatchDescriptorsBF : HRESULT
VAR_INPUT
    ipQueryDescriptors : ITcVnImage;
    ipTrainDescriptors : ITcVnImage;
    ipMatches          : Reference To ITcVnContainer;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipQueryDescriptors	ITcVnImage [▶ 407]	查询描述符（即参考图像/模板图像的描述符）
ipTrainDescriptors	ITcVnImage [▶ 407]	训练描述符（即源图像/输入图像的描述符）

名称	类型	描述
ipMatches	引用 ITcVnContainer [▶ 365]	返回一个以描述符匹配项填充的容器（ContainerType_Vector_TcVnDMatch；重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

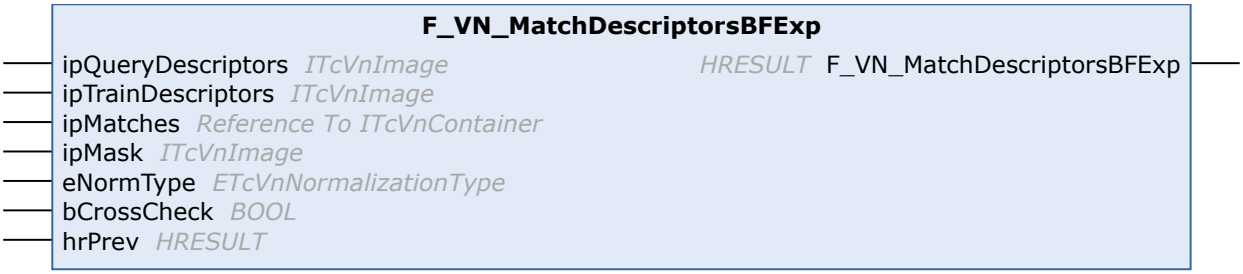
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.30 F_VN_MatchDescriptorsBFExp



Match descriptors using a brute force approach. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_MatchDescriptorsBFExp : HRESULT
VAR_INPUT
    ipQueryDescriptors : ITcVnImage;
    ipTrainDescriptors : ITcVnImage;
    ipMatches          : Reference To ITcVnContainer;
    ipMask             : ITcVnImage;
    eNormType          : ETcVnNormalizationType;
    bCrossCheck        : BOOL;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipQueryDescriptors	ITcVnImage [▶ 407]	查询描述符（即参考图像/模板图像的描述符）
ipTrainDescriptors	ITcVnImage [▶ 407]	训练描述符（即源图像/输入图像的描述符）
ipMatches	引用 ITcVnContainer [▶ 365]	返回一个以描述符匹配项填充的容器（ContainerType_Vector_TcVnDMatch；重复使用非零接口指针。）

名称	类型	描述
ipMask	ITcVnImage [▶ 407]	用于指定允许匹配的掩膜，即仅当 mask[i][j] != 0 时，query[i] 才能与 train[j] 匹配（如果不需要掩膜，则设为 0）
eNormType	ETcVnNormalizationType [▶ 203]	归一化类型（仅支持 L1、L2、L2SQR、HAMMING、HAMMING2）。如果 ORB nBriefPoints 参数为 3 或 4，则 ORB 描述符应使用 HAMMING2
bCrossCheck	BOOL	如果为 true，则只返回一致的匹配结果，即 query->train 和 train->query 检测到的匹配结果相同
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

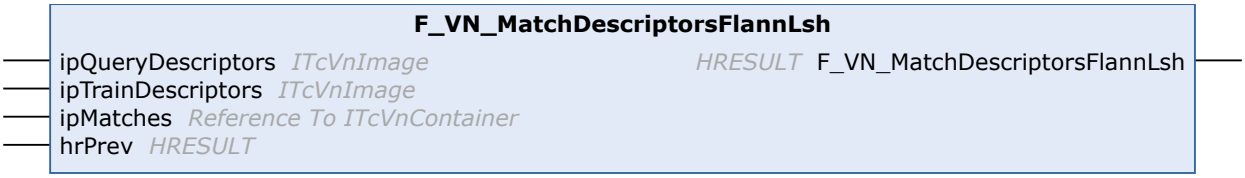
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.31 F_VN_MatchDescriptorsFlannLsh



Match descriptors using a FLANN based approach with LSH index.

语法

定义：

```
FUNCTION F_VN_MatchDescriptorsFlannLsh : HRESULT
VAR_INPUT
    ipQueryDescriptors : ITcVnImage;
    ipTrainDescriptors : ITcVnImage;
    ipMatches          : Reference To ITcVnContainer;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipQueryDescriptors	ITcVnImage [▶ 407]	查询描述符（即参考图像/模板图像的描述符）
ipTrainDescriptors	ITcVnImage [▶ 407]	训练描述符（即源图像/输入图像的描述符）
ipMatches	引用 ITcVnContainer [▶ 365]	返回一个以描述符匹配项填充的容器（ContainerType_Vector_TcVnDMatch；重复使用非零接口指针。）

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

匹配的数量

由于与方法有关的原因，在每次调用此函数时，匹配的数量可能会有所不同。

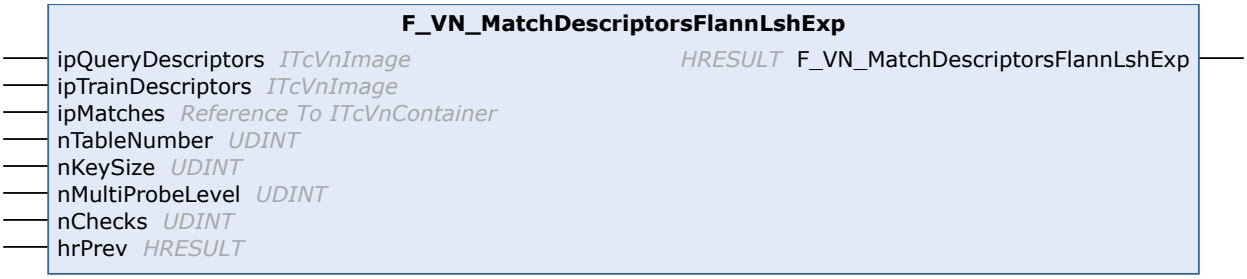
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.32 F_VN_MatchDescriptorsFlannLshExp



Match descriptors using a FLANN based approach with LSH index. (expert function)

语法

定义：

```
FUNCTION F_VN_MatchDescriptorsFlannLshExp : HRESULT
VAR_INPUT
    ipQueryDescriptors : ITcVnImage;
    ipTrainDescriptors : ITcVnImage;
    ipMatches          : Reference To ITcVnContainer;
    nTableNumber       : UDINT;
    nKeySize           : UDINT;
    nMultiProbeLevel   : UDINT;
    nChecks            : UDINT;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipQueryDescriptors	ITcVnImage [▶ 407]	查询描述符（即参考图像/模板图像的描述符）
ipTrainDescriptors	ITcVnImage [▶ 407]	训练描述符（即源图像/输入图像的描述符）

名称	类型	描述
ipMatches	引用 ITcVnContainer [▶ 365]	返回一个以描述符匹配项填充的容器（ContainerType_Vector_TcVnDMatch；重复使用非零接口指针。）
nTableNumber	UDINT	表格数量
nKeySize	UDINT	关键点大小
nMultiProbeLevel	UDINT	多探针级别
nChecks	UDINT	搜索邻区时访问的最大叶节点数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

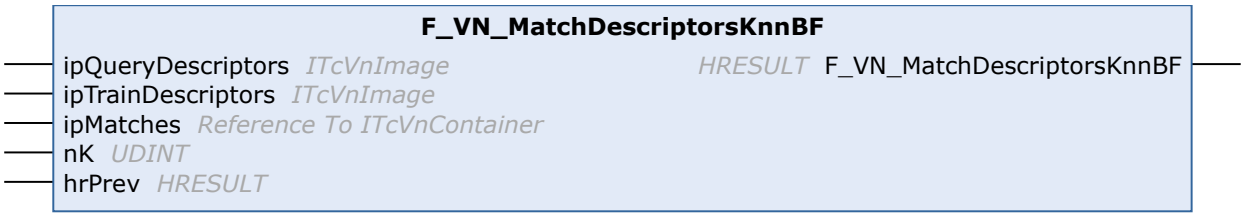
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.33 F_VN_MatchDescriptorsKnnBF



Match descriptors (k nearest neighbors) using a brute force approach.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_MatchDescriptorsKnnBF : HRESULT
VAR_INPUT
    ipQueryDescriptors : ITcVnImage;
    ipTrainDescriptors : ITcVnImage;
    ipMatches          : Reference To ITcVnContainer;
    nK                 : UDINT;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipQueryDescriptors	ITcVnImage [▶ 407]	查询描述符（即参考图像/模板图像的描述符）
ipTrainDescriptors	ITcVnImage [▶ 407]	训练描述符（即源图像/输入图像的描述符）

名称	类型	描述
ipMatches	引用 ITcVnContainer [▶ 365]	返回一个以描述符匹配项填充的容器（ContainerType_Vector_Vector_TcVnDMatch；重复使用非零接口指针。）
nK	UDINT	每个查询描述符所需的最佳训练匹配项数量（即 nk := 2）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

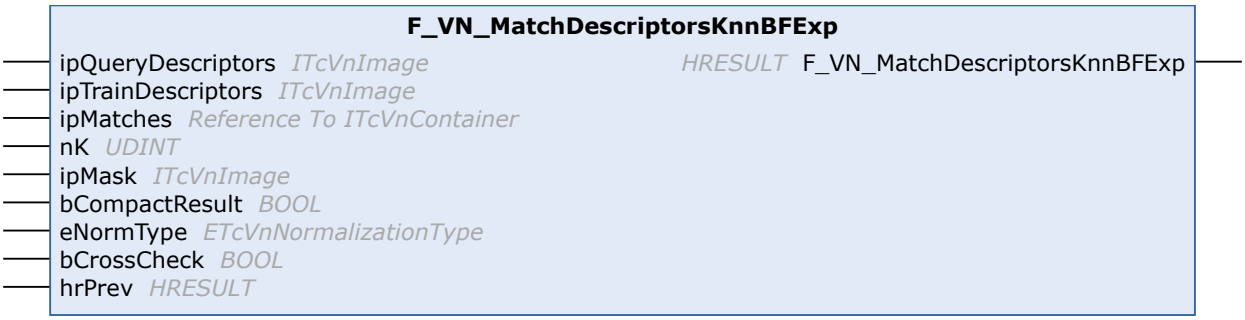
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.34 F_VN_MatchDescriptorsKnnBFExp



Match descriptors (k nearest neighbors) using a brute force approach. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_MatchDescriptorsKnnBFExp : HRESULT
VAR_INPUT
    ipQueryDescriptors : ITcVnImage;
    ipTrainDescriptors : ITcVnImage;
    ipMatches          : Reference To ITcVnContainer;
    nK                 : UDINT;
    ipMask             : ITcVnImage;
    bCompactResult     : BOOL;
    eNormType          : ETcVnNormalizationType;
    bCrossCheck        : BOOL;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipQueryDescriptors	ITcVnImage [▶ 407]	查询描述符（即参考图像/模板图像的描述符）
ipTrainDescriptors	ITcVnImage [▶ 407]	训练描述符（即源图像/输入图像的描述符）

名称	类型	描述
ipMatches	引用 ITcVnContainer [▶ 365]	返回一个以描述符匹配项填充的容器（ContainerType_Vector_Vector_TcVnDMatch；重复使用非零接口指针。）
nK	UDINT	每个查询描述符所需的最佳训练匹配项数量（即 nk := 2）
ipMask	ITcVnImage [▶ 407]	用于指定允许匹配的掩膜，即仅当 mask[i][j] != 0 时，query[i] 才能与 train[j] 匹配（如果不需要掩膜，则设为 0）
bCompactResult	BOOL	如果为 true，则匹配向量不包含完全屏蔽的查询描述符条目
eNormType	ETcVnNormalizationType [▶ 203]	归一化类型（仅支持 L1、L2、L2SQR、HAMMING、HAMMING2）。如果 ORB nBriefPoints 参数为 3 或 4，则 ORB 描述符应使用 HAMMING2
bCrossCheck	BOOL	如果为 true，则只返回一致的匹配结果，即 query->train 和 train->query 检测到的匹配结果相同（仅在 nK = 1 时使用）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

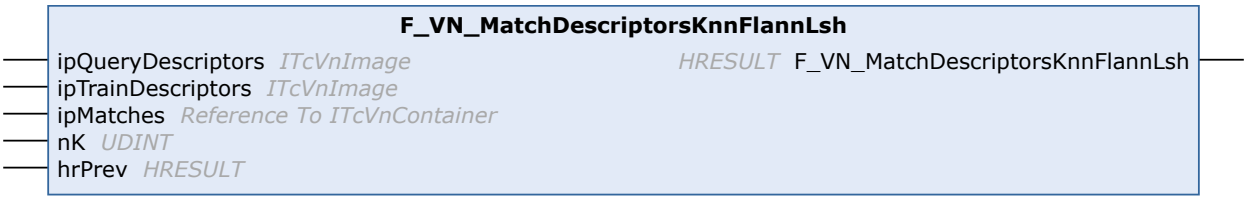
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.35 F_VN_MatchDescriptorsKnnFlannLsh



Match descriptors (k nearest neighbors) using a FLANN based approach with LSH index.

语法

定义：

```
FUNCTION F_VN_MatchDescriptorsKnnFlannLsh : HRESULT
VAR_INPUT
    ipQueryDescriptors : ITcVnImage;
    ipTrainDescriptors : ITcVnImage;
    ipMatches          : Reference To ITcVnContainer;
    nK                 : UDINT;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipQueryDescriptors	ITcVnImage [▶ 407]	查询描述符（即参考图像/模板图像的描述符）
ipTrainDescriptors	ITcVnImage [▶ 407]	训练描述符（即源图像/输入图像的描述符）
ipMatches	引用 ITcVnContainer [▶ 365]	返回一个以描述符匹配项填充的容器（ContainerType_Vector_Vector_TcVnDMatch；重复使用非零接口指针。）
nK	UDINT	每个查询描述符所需的最佳训练匹配项数量（即 nk := 2）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

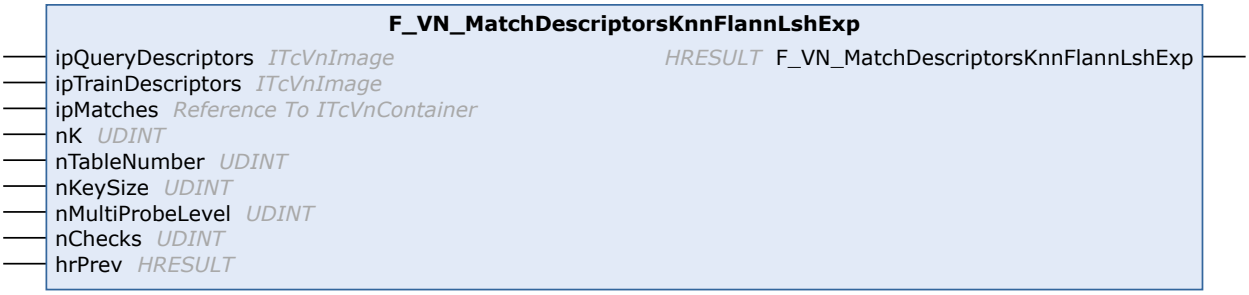
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.36 F_VN_MatchDescriptorsKnnFlannLshExp



Match descriptors (k nearest neighbors) using a FLANN based approach with LSH index. (expert function)

语法

定义：

```
FUNCTION F_VN_MatchDescriptorsKnnFlannLshExp : HRESULT
VAR_INPUT
    ipQueryDescriptors : ITcVnImage;
    ipTrainDescriptors : ITcVnImage;
    ipMatches          : Reference To ITcVnContainer;
    nK                 : UDINT;
    nTableNumber       : UDINT;
    nKeySize           : UDINT;
    nMultiProbeLevel   : UDINT;
    nChecks            : UDINT;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipQueryDescriptors	ITcVnImage [▶ 407]	查询描述符（即参考图像/模板图像的描述符）
ipTrainDescriptors	ITcVnImage [▶ 407]	训练描述符（即源图像/输入图像的描述符）
ipMatches	引用 ITcVnContainer [▶ 365]	返回一个以描述符匹配项填充的容器（ContainerType_Vector_Vector_TcVnDMatch；重复使用非零接口指针。）
nK	UDINT	每个查询描述符所需的最佳训练匹配项数量（即 nk := 2）
nTableNumber	UDINT	表格数量
nKeySize	UDINT	关键点大小
nMultiProbeLevel	UDINT	多探针级别
nChecks	UDINT	搜索邻区时访问的最大叶节点数量
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

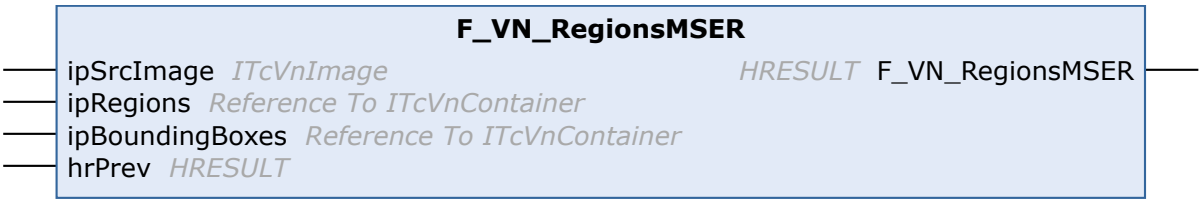
所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.37 F_VN_RegionsMSER



Detects regions using the MSER method.

语法

定义：

```
FUNCTION F_VN_RegionsMSER : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipRegions       : Reference To ITcVnContainer;
    ipBoundingBoxes : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipRegions	引用 ITcVnContainer [▶ 365]	返回一个以区域点填充的容器（ContainerType_Vector_Vector_TcVnPoint2_DINT；重复使用非零接口指针。）
ipBoundingBoxes	引用 ITcVnContainer [▶ 365]	返回一个以区域边界框填充的容器（ContainerType_Vector_TcVnRectangle_DINT；重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.19.38 F_VN_RegionsMSERExp



Detects regions using the MSER method. (expert function)

语法

定义：

```
FUNCTION F_VN_RegionsMSERExp : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipRegions       : Reference To ITcVnContainer;
    ipBoundingBoxes : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    stParams        : TcVnParamsMSER;
END_VAR
VAR_INPUT
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像

名称	类型	描述
ipRegions	引用 ITcVnContainer [▶ 365]	返回一个以区域点填充的容器（ContainerType_Vector_Vector_TcVnPoint2_DINT；重复使用非零接口指针。）
ipBoundingBoxes	引用 ITcVnContainer [▶ 365]	返回一个以区域边界框填充的容器（ContainerType_Vector_TcVnRectangle_DINT；重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stParams	TcVnParamsMSER [▶ 232]	其它专家参数

返回值

HRESULT [▶ 135]

所需的许可证

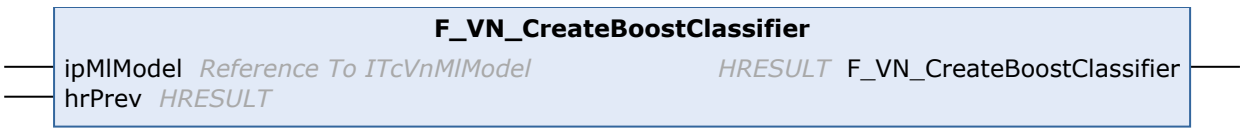
TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20 机器学习.

6.1.4.20.1 F_VN_CreateBoostClassifier



Create a Boost classifier using the default parameters. The initial reference count is set to one if a new model is created and kept, otherwise. The Boost classifier is only applicable to binary classification problems. It learns to distinguish between samples labelled with two user-defined class labels by incrementally adding weak classifiers to improve the classification results. Models of this type neither support on-line training (sample by sample) nor retraining.

语法

定义：

```
FUNCTION F_VN_CreateBoostClassifier : HRESULT
VAR_INPUT
    ipMlModel : Reference To ITcVnMlModel;
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CreateBoostClassifier` 可创建boost分类器模型。

提升分类器模型（Boost classifier models）

提升分类器模型只适用于二元分类问题。该模型通过逐步增加弱分类器来改进分类结果，从而学会区分标有用户定义的两个类别标签的样本。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

专家参数

专家变体 F_VN_CreateBoostClassifierExp [▶ 1368] 和 F_VN_CreateBoostClassifierExp2 [▶ 1370] 包含附加参数。

应用

例如，可以这样创建一个提升分类器模型：

```
hr := F_VN_CreateBoostClassifier(
    ipMlModel    := ipMlModel,
    hrPrev       := hr);
```

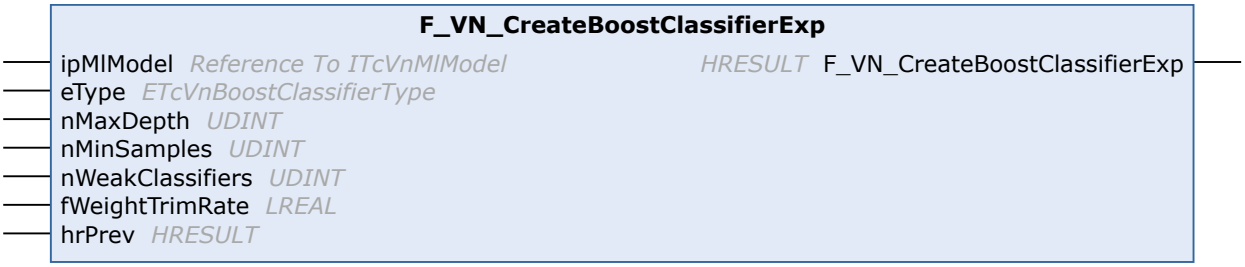
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.2 F_VN_CreateBoostClassifierExp



Create a Boost classifier. The initial reference count is set to one if a new model is created and kept, otherwise. The Boost classifier is only applicable to binary classification problems. It learns to distinguish between samples labelled with two user-defined class labels by incrementally adding weak classifiers to improve the classification results. Models of this type neither support on-line training (sample by sample) nor retraining. (expert function)

语法

定义：

```
FUNCTION F_VN_CreateBoostClassifierExp : HRESULT
VAR_INPUT
    ipMlModel        : Reference To ITcVnMlModel;
    eType            : ETcVnBoostClassifierType;
    nMaxDepth        : UDINT;
    nMinSamples      : UDINT;
    nWeakClassifiers : UDINT;
    fWeightTrimRate  : LREAL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eType	ETcVnBoostClassifierType [▶ 160]	学习算法类型（默认类型：TCVN_BCT_REAL）
nMaxDepth	UDINT	树模型最大深度（默认值：1）
nMinSamples	UDINT	节点内需要分割的最小样本数（默认值：10）
nWeakClassifiers	UDINT	弱分类器数量（默认：100）
fWeightTrimRate	LREAL	训练时使用的权重阈值（关闭：0；默认：0.95）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CreateBoostClassifierExp` 是 F_VN_CreateBoostClassifier [▶ 1367] 的专家变体。它包含额外的参数。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

学习技巧

可通过 `eType` 选择学习算法。支持 Discrete AdaBoost、Real AdaBoost、LogitBoost 和 Gentle AdaBoost。

最大树深度

`nMaxDepth` 表示树中决策层的最大数量。提升分类器通常使用树深度为 1 的决策树。不过，也有可能出现更高的数值。

节点内样本的最少数量

`nMinSamples` 是树形成过程中节点内必须保留的最小样本数。低于这一数字的拆分不会进行。

决策树数量

`nWeakClassifiers` 表示通过助推法组合成分类器的决策树（弱学习者）的数量。

运行时优化

通过 `fWeightTrimRate`，可以利用自动确定的权重对各个学习步骤所使用的样本数量进行限制。允许取值范围为 0 至 1，只有权重大于 1 - `fWeightTrimRate` 的样本才会在接下来的学习步骤中得到考虑。值 0 则禁用此优化。

应用

例如，可以这样创建一个提升分类器模型：

```
hr := F_VN_CreateBoostClassifierExp(
    ipMlModel           := ipMlModel,
    eType               := TCVN_BCT_REAL,
    nMaxDepth           := 1,
    nMinSamples         := 10,
    nWeakClassifiers    := 100,
    fWeightTrimRate     := 0.95,
    hrPrev              := hr);
```

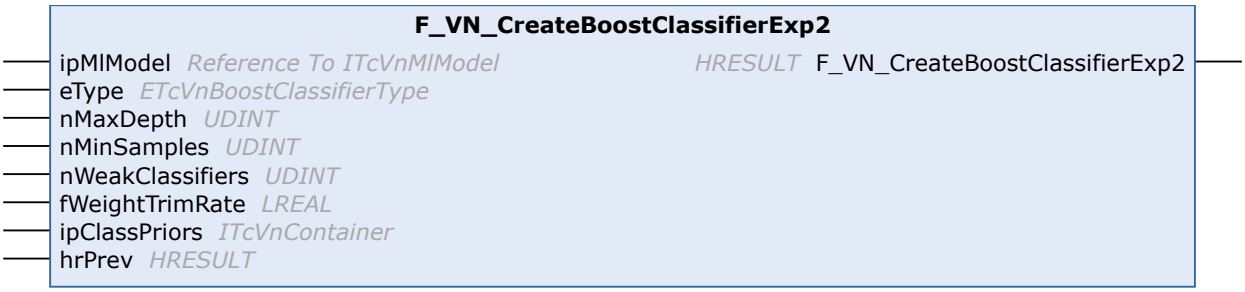
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.3 F_VN_CreateBoostClassifierExp2



Create a Boost classifier. The initial reference count is set to one if a new model is created and kept, otherwise. The Boost classifier is only applicable to binary classification problems. It learns to distinguish between samples labelled with two user-defined class labels by incrementally adding weak classifiers to improve the classification results. Models of this type neither support on-line training (sample by sample) nor retraining. (expert function)

语法

定义：

```
FUNCTION F_VN_CreateBoostClassifierExp2 : HRESULT
VAR_INPUT
    ipMlModel        : Reference To ITcVnMlModel;
    eType            : ETcVnBoostClassifierType;
    nMaxDepth        : UDINT;
    nMinSamples      : UDINT;
    nWeakClassifiers : UDINT;
    fWeightTrimRate  : LREAL;
```

```
    ipClassPriors    : ITcVnContainer;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eType	ETcVnBoostClassifierType [▶ 160]	学习算法类型（默认类型：TCVN_BCT_REAL）
nMaxDepth	UDINT	树模型最大深度（默认值：1）
nMinSamples	UDINT	节点内需要分割的最小样本数（默认值：10）
nWeakClassifiers	UDINT	弱分类器数量（默认：100）
fWeightTrimRate	LREAL	训练时使用的权重阈值（关闭：0；默认：0.95）。
ipClassPriors	ITcVnContainer [▶ 365]	类优先级（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL；仅适用于分类器；可选，如果不需要或不允许，则设为 0；默认值：0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CreateBoostClassifierExp2` 是 F_VN_CreateBoostClassifier [▶ 1367] 的专家变体。它包含额外的参数。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

学习技巧

可通过 `eType` 选择学习算法。支持 Discrete AdaBoost、Real AdaBoost、LogitBoost 和 Gentle AdaBoost。

最大树深度

`nMaxDepth` 表示树中决策层的最大数量。提升分类器通常使用树深度为 1 的决策树。不过，也有可能出现更高的数值。

节点内样本的最少数量

`nMinSamples` 是树形成过程中节点内必须保留的最小样本数。低于这一数字的拆分不会进行。

决策树数量

`nWeakClassifiers` 表示通过助推法组合成分类器的决策树（弱学习者）的数量。

运行时优化

通过 `fWeightTrimRate`，可以利用自动确定的权重对各个学习步骤所使用的样本数量进行限制。允许取值范围为 0 至 1，只有权重大于 1 - `fWeightTrimRate` 的样本才会在接下来的学习步骤中得到考虑。值 0 则禁用此优化。

类别先验

`ipClassPriors` 是一个容器，可用来指定单个类别的先验概率。

应用

例如，可以这样创建一个提升分类器模型：

```
hr := F_VN_CreateBoostClassifierExp2(
    ipMlModel           := ipMlModel,
    eType               := TCVN_BCT_REAL,
    nMaxDepth           := 1,
    nMinSamples         := 10,
    nWeakClassifiers    := 100,
    fWeightTrimRate     := 0.95,
    ipClassPriors       := 0,
    hrPrev              := hr);
```

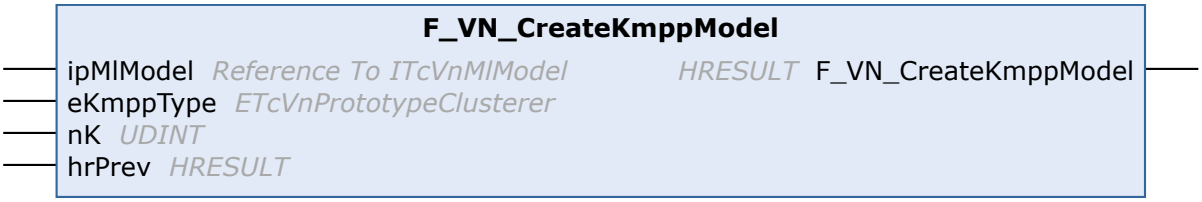
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.4 F_VN_CreateKmppModel



创建指定类型的 k-means++ 模型。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。此类模型既不支持在线训练（逐样本训练），也不支持再训练。可选择提供预测的异常程度。

语法

定义：

```
FUNCTION F_VN_CreateKmppModel : HRESULT
VAR_INPUT
    ipMlModel : Reference To ITcVnMlModel;
    eKmppType : ETcVnPrototypeClusterer;
    nK        : UDINT;
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eKmppType	ETcVnPrototypeClusterer [▶ 207]	k-means++ 模型类型
nK	UDINT	参数 k（聚类数）

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_CreateKmppModel 可创建 K-Means++（KMPP）模型。

K-Means++ 模型

K-Means++ 可用于聚类或异常检测。它能发现数据点的聚类。需要同时对所有样本进行训练（批量训练），无法进行后期训练。

在不同训练中，集群名称的分配（如 DINT）可能会有所不同。如果该分配是确定性的，则必须事先使用 F_VN_SetRngSeed [▶ 1541]设置一个固定的随机种子。

参数

模型

创建的模型将在接口指针 ipMlModel 中返回。

模型类型

eKmppType 指定 K-Means++ 是用于聚类（TCVN_PC_CLUSTERER）还是异常检测（TCVN_PC_NOVELTY_DETECTOR）。

聚类数

K-Means++ 模型中的聚类数由 nK 指定，一旦创建就不能更改。在训练过程中，样本数必须大于或等于聚类数。

专家参数

专家变体 F_VN_CreateKmppModelExp [▶ 1374] 包含附加参数。

应用

例如，一个包含 5 个聚类的 K-Means++ 模型可如下创建：

```
hr := F_VN_CreateKmppModel(
    ipMlModel    := ipMlModel,
    eKmppType    := TCVN_PC_CLUSTERER,
    nK           := 5,
    hrPrev       := hr);
```

所需的许可证

TC3 Machine Learning Realtime Inference

6.1.4.20.5 F_VN_CreateKmppModelExp



创建指定类型的 k-means++ 模型。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。此类模型既不支持在线训练（逐样本训练），也不支持再训练。可选择提供预测的异常程度。（专家函数）

语法

定义：

```
FUNCTION F_VN_CreateKmppModelExp : HRESULT
VAR_INPUT
    ipMlModel         : Reference To ITcVnMlModel;
    eKmppType         : ETcVnPrototypeClusterer;
    nK                : UDINT;
    bDoublePrecision  : BOOL;
    nMaxIterations    : UDINT;
    fEpsilon          : LREAL;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eKmppType	ETcVnPrototypeClusterer [▶ 207]	k-means++ 模型类型
nK	UDINT	参数 k（聚类数）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成模型。（默认设置：FALSE）
nMaxIterations	UDINT	最大迭代次数（如果等于 0，则使用默认值 10）
fEpsilon	LREAL	两次连续迭代之间允许的最大误差差值（如果差值等于 0，则使用默认值 0.001）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CreateKmppModelExp` 是F_VN_CreateKmppModel [▶ 1372] 的专家变体。它包含额外的参数。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

模型类型

eKmppType 指定 K-Means++ 是用于聚类（TCVN_PC_CLUSTERER）还是异常检测（TCVN_PC_NOVELTY_DETECTOR）。

聚类数

K-Means++ 模型中的聚类数由 nK 指定，一旦创建就不能更改。在训练过程中，样本数必须大于或等于聚类数。

更高精度

如果 bDoublePrecision 是 TRUE，则使用 LREAL 作为模型内部计算的数据类型；如果是 FALSE，则使用 REAL。

最大迭代

优化最多使用 nMaxIterations 中指定的迭代次数。如果值为 0，则使用相应的默认值。

终止限制

只要两次迭代之间的误差变化不超过 fEpsilon 中的规定，优化就会终止。如果值为 0，则使用相应的默认值。

应用

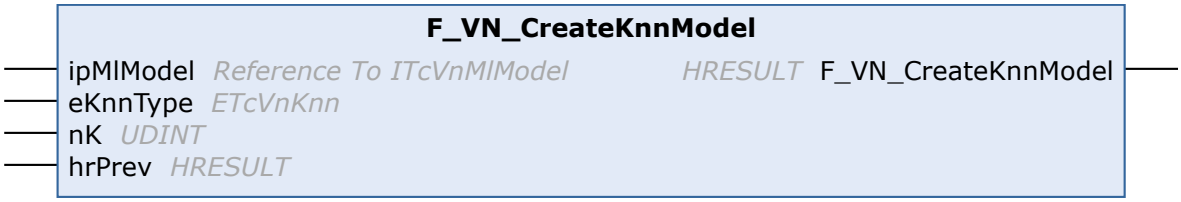
例如，一个包含 5 个聚类的 K-Means++ 模型可如下创建：

```
hr := F_VN_CreateKmppModelExp(
    ipMlModel           := ipMlModel,
    eKmppType           := TCVN_PC_CLUSTERER,
    nK                  := 5,
    bDoublePrecision    := FALSE,
    nMaxIterations      := 0,
    fEpsilon            := 0,
    hrPrev              := hr);
```

所需的许可证

TC3 Machine Learning Realtime Inference

6.1.4.20.6 F_VN_CreateKnnModel



创建指定类型的 k-NN 模型。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。这类模型支持在线训练（逐样本）和再训练。预测值只能是标量。可选择 k 近邻模型提供预测的异常程度。

语法

定义：

```
FUNCTION F_VN_CreateKnnModel : HRESULT
VAR_INPUT
    ipMlModel : Reference To ITcVnMlModel;
    eKnnType  : ETcVnKnn;
    nK        : UDINT;
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eKnnType	ETcVnKnn [▶ 200]	k-NN 模型类型
nK	UDINT	用于预测的参数 k（考虑的邻区数量）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

6.1.4.20.7 F_VN_CreateLbgModel



创建指定类型的 LBG 模型。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。此类模型既不支持在线训练（逐样本训练），也不支持再训练。可选择提供预测的异常程度。

语法

定义：

```
FUNCTION F_VN_CreateLbgModel : HRESULT
VAR_INPUT
    ipMlModel         : Reference To ITcVnMlModel;
    eLbgType          : ETcVnPrototypeClusterer;
    nMaxClusters      : UDINT;
    fMaxClusterRadius : LREAL;
    bSingleSplitSteps : BOOL;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eLbgType	ETcVnPrototypeClusterer [▶ 207]	LBG 模型类型
nMaxClusters	UDINT	最大聚类数
fMaxClusterRadius	LREAL	单个聚类允许的最大半径（L2 范数），即聚类半径越大，被分割成的聚类越小，直到达到 nMaxClusters 的全局数量。

名称	类型	描述
bSingleSplitSteps	BOOL	如果为 true，则总是在拆分单个聚类后运行全局优化。如果为 false，则会在应用全局优化之前，在同一步骤中拆分多个聚类。减少全局优化的应用次数会加快速度，但可能会导致结果不理想，尤其是有 2 个可由 1 表示的邻近聚类时。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CreateLbgModel` 创建 Linde-Buzo-Gray（LBG）模型。

Linde-Buzo-Gray 模型

LBG 模型的工作原理与 K-Means++ 模型类似，都是根据样本迭代调整聚类。LBG 的主要特征是，在迭代过程中，只要新聚类大于 `fMaxClusterRadius`，就会创建新聚类（直到达到最大数 `nMaxClusters`）。另一方面，对于 K-Means++，聚类数为初始设置值。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

模型类型

`eLbgType` 指定 LBG 是用于聚类（`TCVN_PC_CLUSTERER`）还是异常检测（`TCVN_PC_NOVELTY_DETECTOR`）。

最大聚类数

如果由于拆分过大聚类而导致总计存在 `nMaxClusters`，则不会再创建新聚类。

最大聚类半径

在培训过程中，如果某个聚类的半径大于 `fMaxClusterRadius`，该聚类就会被分成两个较小聚类。

优化质量

迭代培训过程的一部分是全局优化步骤。如果 `bSingleSplitSteps = TRUE`，则在每个聚类拆分后执行该优化。这将提高聚类质量，但也会延长训练时间。如果 `bSingleSplitSteps = FALSE`，则只有在多个聚类被拆分后才会执行优化。

专家参数

专家变体 F_VN_CreateLbgModelExp [▶ 1378] 包含附加参数。

应用

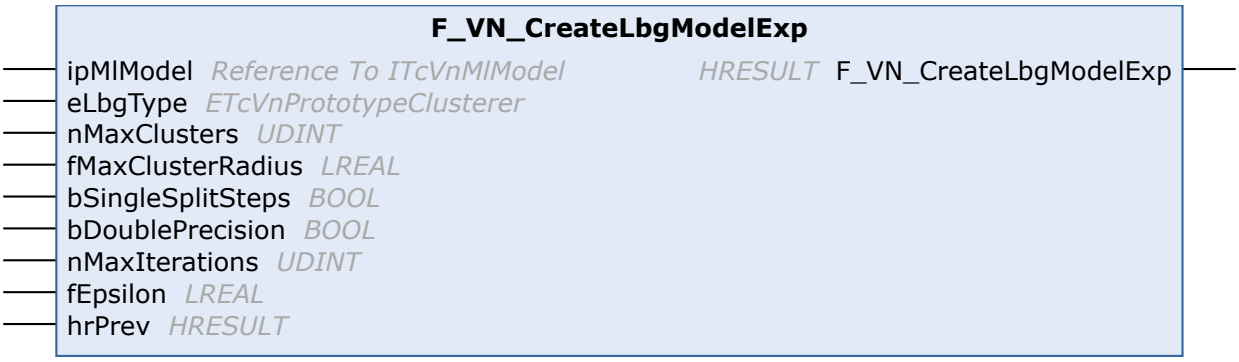
例如，可通过以下方法创建一个具有 `5` 个聚类、最大聚类大小为 `3.6` 以及拥有较高优化质量的LBG 聚类模型：

```
hr := F_VN_CreateLbgModel(
    ipMlModel           := ipMlModel,
    eLbgType            := TCVN_PC_CLUSTERER,
    nMaxClusters        := 5,
    fMaxClusterRadius   := 3.6,
    bSingleSplitSteps   := TRUE,
    hrPrev              := hr);
```

所需的许可证

TC3 Machine Learning Realtime Inference

6.1.4.20.8 F_VN_CreateLbgModelExp



创建指定类型的 LBG 模型。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。此类模型既不支持在线训练（逐样本训练），也不支持再训练。可选择提供预测的异常程度。（专家函数）

语法

定义：

```
FUNCTION F_VN_CreateLbgModelExp : HRESULT
VAR_INPUT
    ipMlModel          : Reference To ITcVnMlModel;
    eLbgType           : ETcVnPrototypeClusterer;
    nMaxClusters       : UDINT;
    fMaxClusterRadius  : LREAL;
    bSingleSplitSteps  : BOOL;
    bDoublePrecision   : BOOL;
    nMaxIterations     : UDINT;
    fEpsilon           : LREAL;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eLbgType	ETcVnPrototypeClusterer [▶ 207]	LBG 模型类型
nMaxClusters	UDINT	最大聚类数
fMaxClusterRadius	LREAL	单个聚类允许的最大半径（L2 范数），即聚类半径越大，被分割成的聚类越小，直到达到 nMaxClusters 的全局数量。
bSingleSplitSteps	BOOL	如果为 true，则总是在拆分单个聚类后运行全局优化。如果为 false，则会在应用全局优化之前，在同一步骤中拆分多个聚类。减少全局优化的应用次数会加快速度，但可能会导致结果不理想，尤其是有 2 个可由 1 表示的邻近聚类时。
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成模型。（默认设置：FALSE）
nMaxIterations	UDINT	最大迭代次数（如果等于 0，则使用默认值 10）
fEpsilon	LREAL	两次连续迭代之间允许的最大误差差值（如果差值等于 0，则使用默认值 0.001）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_CreateLbgModelExp 是 F_VN_CreateLbgModel [▶ 1376] 的专家变体。它包含额外的参数。

参数

模型

创建的模型将在接口指针 ipMlModel 中返回。

模型类型

eLbgType 指定 LBG 是用于聚类（TCVN_PC_CLUSTERER）还是异常检测（TCVN_PC_NOVELTY_DETECTOR）。

最大聚类数

如果由于拆分过大聚类而导致总计存在 nMaxClusters，则不会再创建新聚类。

最大聚类半径

在培训过程中，如果某个聚类的半径大于 fMaxClusterRadius，该聚类就会被分成两个较小聚类。

优化质量

迭代培训过程的一部分是全局优化步骤。如果 bSingleSplitSteps = TRUE，则在每个聚类拆分后执行该优化。这将提高聚类质量，但也会延长训练时间。如果 bSingleSplitSteps = FALSE，则只有在多个聚类被拆分后才会执行优化。

更高精度

如果 bDoublePrecision 是 TRUE，则使用 LREAL 作为模型内部计算的数据类型；如果是 FALSE，则使用 REAL。

最大迭代

优化最多使用 nMaxIterations 中指定的迭代次数。如果值为 0，则使用相应的默认值。

终止限制

只要两次迭代之间的误差变化不超过 fEpsilon 中的规定，优化就会终止。如果值为 0，则使用相应的默认值。

应用

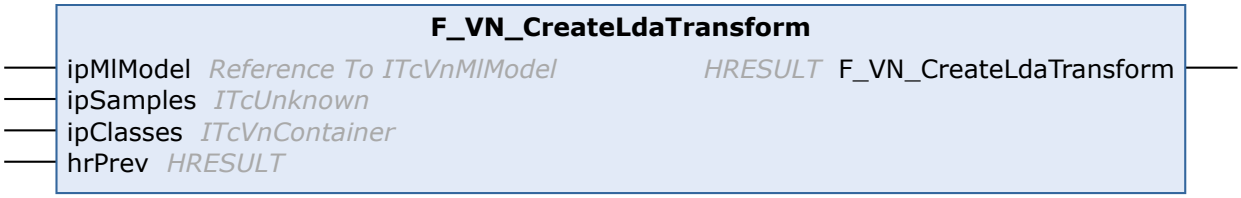
例如，可通过以下方法创建一个具有 5 个聚类、最大聚类大小为 3.6 以及拥有较高优化质量的LBG 聚类模型：

```
hr := F_VN_CreateLbgModelExp(
    ipMlModel           := ipMlModel,
    eLbgType            := TCVN_PC_CLUSTERER,
    nMaxClusters        := 5,
    fMaxClusterRadius   := 3.6,
    bSingleSplitSteps   := TRUE,
    bDoublePrecision    := TRUE,
    nMaxIterations      := 0,
    fEpsilon            := 0,
    hrPrev              := hr);
```

所需的许可证

TC3 Machine Learning Realtime Inference

6.1.4.20.9 F_VN_CreateLdaTransform



Create an LDA-based feature transform from the provided data. The number of samples must be >= the number of features and the number of classes must be >= 2. The initial reference count is set to one if a new model is created and kept, otherwise.

语法

定义：

```
FUNCTION F_VN_CreateLdaTransform : HRESULT
VAR_INPUT
    ipMlModel : Reference To ITcVnMlModel;
    ipSamples : ITcUnknown;
    ipClasses : ITcVnContainer;
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回已创建的特征变换（重复使用非零接口指针。）
ipSamples	ITcUnknown [▶ 424]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
ipClasses	ITcVnContainer [▶ 365]	与输入样本相对应的类标签（ContainerType_Vector_DINT）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CreateLdaTransform` 可创建一个用于特征转换的线性判别分析（LDA）模型。

线性判别分析模型

线性判别分析创建一种特征转换，目的是利用转换后的特征尽可能容易地将类别区分开来。该模型可用于有效压缩样本。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

样本

所有样本均在一个容器中通过 `ipSamples` 发送至模型。

分类

容器中所有样本的类别分配通过 `ipClasses` 提供给模型。

应用

例如，一个用于特征降维的 LDA 模型可如下创建：

```
hr := F_VN_CreateLdaTransform(
    ipMlModel    := ipMlModel,
    ipSamples    := ipSamples,
    ipClasses    := ipClasses,
    hrPrev       := hr);
```

相关函数

- F_VN_CreateLdaTransformViaComponentNum [▶ 1381]
- F_VN_CreatePcaTransform [▶ 1384]
- F_VN_CreatePcaTransformViaComponentNum [▶ 1385]
- F_VN_CreatePcaTransformViaVariance [▶ 1386]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.10 F_VN_CreateLdaTransformViaComponentNum



Create an LDA-based feature transform from the provided data where the number of components to be used is predefined. The number of samples must be >= the number of features and the number of classes must be >= 2. The initial reference count is set to one if a new model is created and kept, otherwise. (expert function)

语法

定义：

```
FUNCTION F_VN_CreateLdaTransformViaComponentNum : HRESULT
VAR_INPUT
    ipMlModel     : Reference To ITcVnMlModel;
    ipSamples     : ITcUnknown;
    ipClasses     : ITcVnContainer;
    nComponentNum : UDINT;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回已创建的特征变换（重复使用非零接口指针。）
ipSamples	ITcUnknown [▶ 424]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
ipClasses	ITcVnContainer [▶ 365]	与输入样本相对应的类标签（ContainerType_Vector_DINT）

名称	类型	描述
nComponentNum	UDINT	分量数（nComponentNum 必须小于类的数量。如果数值为 0，则自动计算最大分量数。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数`F_VN_CreateLdaTransformViaComponentNum`是 F_VN_CreateLdaTransform [▶ 1380] 的变体。它包含一个附加参数`nComponentNum`，用于预先指定特征降维的组件数。这样就可以设置数据压缩级别。

参数

模型

创建的模型将在接口指针`ipMlModel`中返回。

样本

所有样本均在一个容器中通过`ipSamples`发送至模型。

分类

容器中所有样本的类别分配通过`ipClasses`提供给模型。

组件数量

`nComponentNum`组件用于特征降维。如为 0，将自动计算组件的最大数量。

应用

例如，一个包含 25 个组件的 LDA 模型可以如下创建：

```
hr := F_VN_CreateLdaTransformViaComponentNum(
    ipMlModel       := ipMlModel,
    ipSamples       := ipSamples,
    ipClasses       := ipClasses,
    nComponentNum   := 25,
    hrPrev          := hr);
```

相关函数

- F_VN_CreateLdaTransform [▶ 1380]
- F_VN_CreatePcaTransform [▶ 1384]
- F_VN_CreatePcaTransformViaComponentNum [▶ 1385]
- F_VN_CreatePcaTransformViaVariance [▶ 1386]

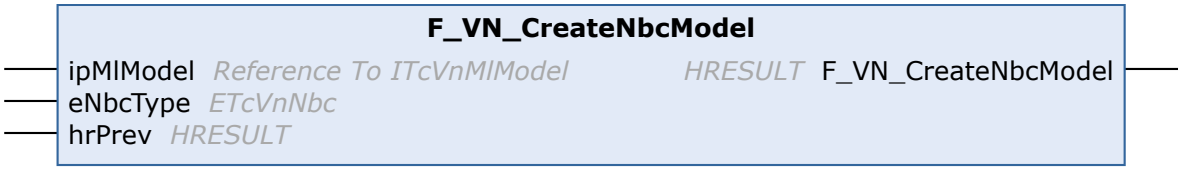
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.11 F_VN_CreateNbcModel



创建指定类型的普通贝叶斯（Bayes）分类器。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。如要训练普通贝叶斯（Bayes）分类器，每个类别都需要足够多的样本。这取决于特征的数量和数据的分布。因此，需要对每种应用进行测试。此类模型不支持在线训练（逐样本）。要对此类分类器模型进行再训练，显示的类集必须与之前的学习步骤完全相同， 否则会出现异常。可选择普通贝叶斯分类器提供预测的异常程度。

语法

定义：

```
FUNCTION F_VN_CreateNbcModel : HRESULT
VAR_INPUT
    ipMlModel : Reference To ITcVnMlModel;
    eNbcType  : ETcVnNbc;
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eNbcType	ETcVnNbc [▶ 203]	普通贝叶斯（Bayes）分类器类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CreateNbcModel` 可创建一个一般贝叶斯分类器 (NBC) 模型。

一般贝叶斯分类器模型

一般贝叶斯分类器创建一个混合模型，其中每个类别均由高斯分布表示。这些高斯函数的参数在训练过程中确定。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

模型类型

`eNbcType` 指定 NBC 是用于分类（`TCVN_NBC_CLASSIFIER`）还是异常检测（`TCVN_NBC_NOVELTY_DETECTOR`）。

应用

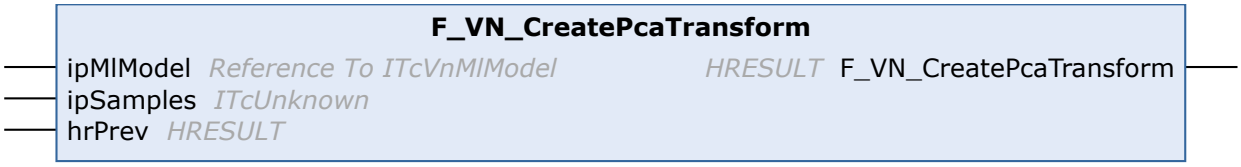
例如，用于分类的 NBC 模型可如下创建：

```
hr := F_VN_CreateNbcModel(
    ipMlModel    := ipMlModel,
    eNbcType     := TCVN_NBC_CLASSIFIER,
    hrPrev       := hr);
```

所需的许可证

TC3 Machine Learning Realtime Inference

6.1.4.20.12 F_VN_CreatePcaTransform



Create a PCA-based feature transform from the provided data. The maximum number of principal components that can be computed equals the minimum of the number of samples and the number of features. The initial reference count is set to one if a new model is created and kept, otherwise.

语法

定义：

```
FUNCTION F_VN_CreatePcaTransform : HRESULT
VAR_INPUT
    ipMlModel : Reference To ITcVnMlModel;
    ipSamples : ITcUnknown;
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回已创建的特征变换（重复使用非零接口指针。）
ipSamples	ITcUnknown [▶ 424]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CreatePcaTransform` 可创建主组件分析 (PCA) 模型。

主成分分析模型

在主成分分析中，要计算样本分布的主要组成成分。主成分是特征空间中显示样本方差最大的方向。通过将数据投影到选定主成分上，特征空间的维度会降低，其中尽可能保留了数据的方差。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

样本

所有样本均在一个容器中通过 ipSamples 发送至模型。

应用

例如，用于特征降维的 PCA 模型可如下创建：

```
hr := F_VN_CreatePcaTransform(
    ipMlModel    := ipMlModel,
    ipSamples    := ipSamples,
    hrPrev       := hr);
```

相关函数

- F_VN_CreatePcaTransformViaComponentNum [▶ 1385]
- F_VN_CreatePcaTransformViaVariance [▶ 1386]
- F_VN_CreateLdaTransform [▶ 1380]
- F_VN_CreateLdaTransformViaComponentNum [▶ 1381]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.13 F_VN_CreatePcaTransformViaComponentNum



Create a PCA-based feature transform from the provided data where the number of prinicpal components to be used is predefined. The maximum number of principal components that can be computed equals the minimum of the number of samples and the number of features. The initial reference count is set to one if a new model is created and kept, otherwise. (expert function)

语法

定义：

```
FUNCTION F_VN_CreatePcaTransformViaComponentNum : HRESULT
VAR_INPUT
    ipMlModel     : Reference To ITcVnMlModel;
    ipSamples     : ITcUnknown;
    nComponentNum : UDINT;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回已创建的特征变换（重复使用非零接口指针。）
ipSamples	ITcUnknown [▶ 424]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
nComponentNum	UDINT	主分量数（如果数值为 0，则自动计算最大分量数。）

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_CreatePcaTransformViaComponentNum 是F_VN_CreatePcaTransform [▶ 1384] 的变体。它包含一个附加参数 nComponentNum，用于提前定义特征降维的主成分数量。

参数

模型

创建的模型将在接口指针 ipMlModel 中返回。

样本

所有样本均在一个容器中通过 ipSamples 发送至模型。

组件数量

PCA 计算 nComponentNum 主成分。如为 0，将自动计算成分的最大数量。

应用

例如，一个有 5 个主成分的 PCA 模型可如下创建：

```
hr := F_VN_CreatePcaTransformViaComponentNum(
    ipMlModel       := ipMlModel,
    ipSamples       := ipSamples,
    nComponentNum   := 5,
    hrPrev          := hr);
```

相关函数

- F_VN_CreatePcaTransform [▶ 1384]
- F_VN_CreatePcaTransformViaVariance [▶ 1386]
- F_VN_CreateLdaTransform [▶ 1380]
- F_VN_CreateLdaTransformViaComponentNum [▶ 1381]

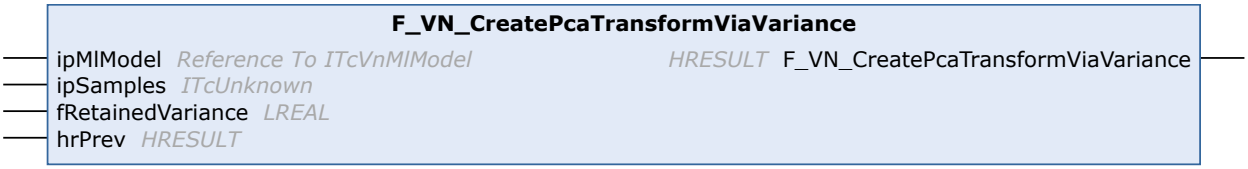
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.14 F_VN_CreatePcaTransformViaVariance



Create a PCA-based feature transform from the provided data based on a given fraction of variance to be retained. The maximum number of principal components that can be computed equals the minimum of the number of samples and the number of features. The initial reference count is set to one if a new model is created and kept, otherwise. (expert function)

语法

定义：

```
FUNCTION F_VN_CreatePcaTransformViaVariance : HRESULT
VAR_INPUT
    ipMlModel         : Reference To ITcVnMlModel;
    ipSamples         : ITcUnknown;
    fRetainedVariance : LREAL;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回已创建的特征变换（重复使用非零接口指针。）
ipSamples	ITcUnknown [▶ 424]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
fRetainedVariance	LREAL	PCA 保留的方差比例（1.0 表示 100%。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CreatePcaTransformViaVariance` 是F_VN_CreatePcaTransform [▶ 1384] 的变体。它包含用于定义保留方差的附加参数 `fRetainedVariance`。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

样本

所有样本均在一个容器中通过 `ipSamples` 发送至模型。

保留方差

在 PCA 中，选择主成分以使方差比例 `fRetainedVariance` 保留在样本中。如为 `1`，保留整个方差，因此特征数量不会减少。

应用

例如，保留方差为 65% 的 PCA 模型可如下创建：

```
hr := F_VN_CreatePcaTransformViaComponentNum(
    ipMlModel           := ipMlModel,
    ipSamples           := ipSamples,
    fRetainedVariance   := 0.65,
    hrPrev              := hr);
```

相关函数

- F_VN_CreatePcaTransform [▶ 1384]
- F_VN_CreatePcaTransformViaComponentNum [▶ 1385]
- F_VN_CreateLdaTransform [▶ 1380]
- F_VN_CreateLdaTransformViaComponentNum [▶ 1381]

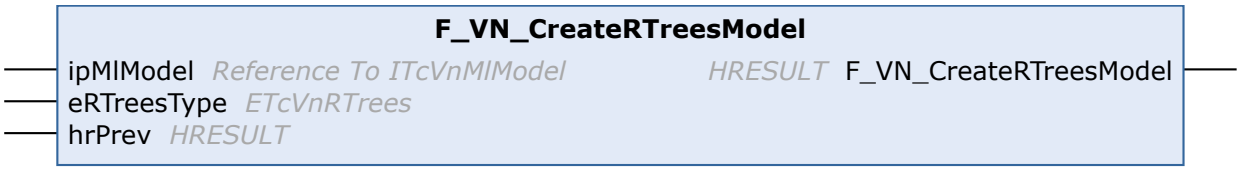
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.15 F_VN_CreateRTreesModel



Create a random trees model of the specified type. The initial reference count is set to one if a new model is created and kept, otherwise. Models of this type neither support on-line training (sample by sample) nor retraining. Predictions can only be scalar.

语法

定义：

```
FUNCTION F_VN_CreateRTreesModel : HRESULT
VAR_INPUT
    ipMlModel   : Reference To ITcVnMlModel;
    eRTreesType : ETcVnRTrees;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eRTreesType	ETcVnRTrees [▶ 209]	随机树模型类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CreateRTreesModel` 可创建随机森林（RTrees）模型。

随机森林模型

随机森林模型使用决策树，其分支由随机过程决定。RTrees 模型的预测根据单个决策树的结果确定。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

模型类型

`eRTreesType` 指定随机森林模型是用于分类（`TCVN_RT_CLASSIFIER`）还是回归（`TCVN_RT_REGRESSOR`）。

专家参数

专家变体 F_VN_CreateRTreesModelExp [▶ 1389] 和 F_VN_CreateRTreesModelExp2 [▶ 1391] 包含附加参数。

应用

例如，用于回归的 RTrees 模型可如下创建：

```
hr := F_VN_CreateRTreesModel(
    ipMlModel   := ipMlModel,
    eRTreesType := TCVN_RT_REGRESSOR,
    hrPrev      := hr);
```

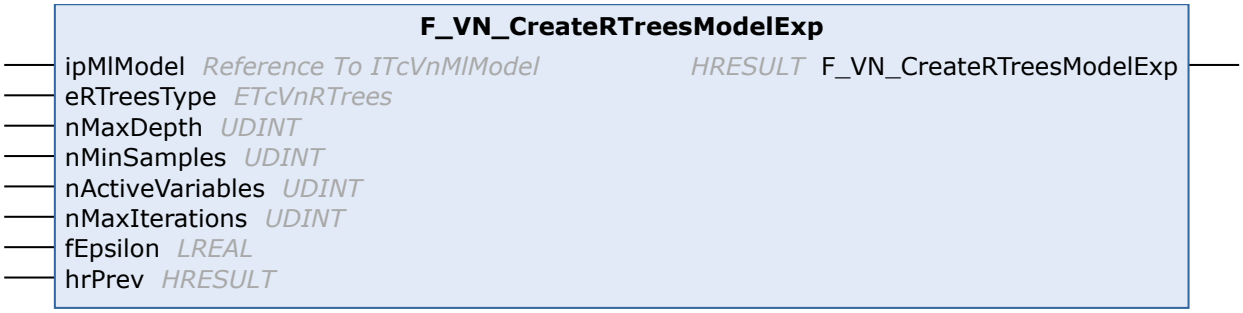
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.16 F_VN_CreateRTreesModelExp



Create a random trees model of the specified type. The initial reference count is set to one if a new model is created and kept, otherwise. Models of this type neither support on-line training (sample by sample) nor retraining. Predictions can only be scalar. (expert function)

语法

定义：

```
FUNCTION F_VN_CreateRTreesModelExp : HRESULT
VAR_INPUT
    ipMlModel        : Reference To ITcVnMlModel;
    eRTreesType      : ETcVnRTrees;
    nMaxDepth        : UDINT;
    nMinSamples      : UDINT;
    nActiveVariables : UDINT;
    nMaxIterations   : UDINT;
    fEpsilon         : LREAL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eRTreesType	ETcVnRTrees [▶ 209]	随机树模型类型
nMaxDepth	UDINT	树模型最大深度（默认值：5）
nMinSamples	UDINT	节点内需要分割的最小样本数（默认值：10）
nActiveVariables	UDINT	分割时考虑的变量数（0 表示 sqrt（变量总数）；默认值：0）
nMaxIterations	UDINT	最大迭代次数（如果等于 0 且 fEpsilon 不为 0.0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 50）
fEpsilon	LREAL	两次连续迭代之间允许的最大误差差值（如果差值等于 0.0 且 nMaxIterations 不为 0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 0.1）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CreateRTReesModelExp` 是F_VN_CreateRTReesModel [▶ 1388] 的专家变体。它包含额外的参数。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

模型类型

`eRTreesType` 指定随机森林模型是用于分类（`TCVN_RT_CLASSIFIER`）还是回归（`TCVN_RT_REGRESSOR`）。

最大树深度

`nMaxDepth` 表示树中决策层的最大数量。

节点内样本的最少数量

`nMinSamples` 是树形成过程中节点内必须保留的最小样本数。低于这一数字的拆分不会进行。

拆分特征

`nActiveVariables` 是用于拆分树的特征数量。如为 `0`，`sqrt(N)` 特征与所有可用特征数 `N` 一起使用。

最大迭代

优化最多使用 `nMaxIterations` 中指定的迭代次数。如果值为 `0`，则使用相应的默认值。

终止限制

只要两次迭代之间的误差变化不超过 `fEpsilon` 中的规定，优化就会终止。如果值为 `0`，则使用相应的默认值。

应用

例如，用于回归的 RTrees 模型可如下创建：

```
hr := F_VN_CreateRTreesModelExp(
    ipMlModel           := ipMlModel,
    eRTreesType         := TCVN_RT_REGRESSOR,
    nMaxDepth           := 8,
    nMinSamples         := 4,
    nActiveVariables    := 0,
    nMaxIterations      := 0,
    fEpsilon            := 0,
    hrPrev              := hr);
```

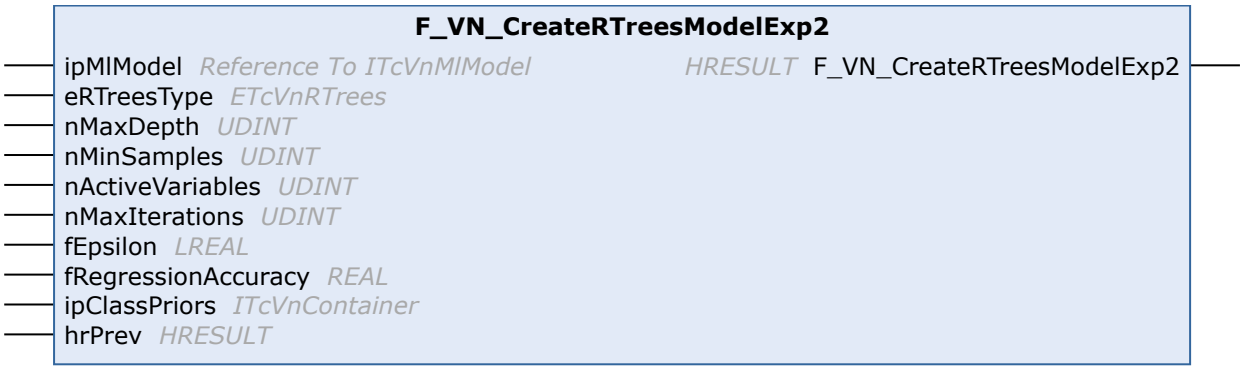
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.17 F_VN_CreateRTreesModelExp2



Create a random trees model of the specified type. The initial reference count is set to one if a new model is created and kept, otherwise. Models of this type neither support on-line training (sample by sample) nor retraining. Predictions can only be scalar. (expert function)

语法

定义：

```
FUNCTION F_VN_CreateRTreesModelExp2 : HRESULT
VAR_INPUT
    ipMlModel           : Reference To ITcVnMlModel;
    eRTreesType         : ETcVnRTrees;
    nMaxDepth           : UDINT;
    nMinSamples         : UDINT;
    nActiveVariables    : UDINT;
    nMaxIterations      : UDINT;
    fEpsilon            : LREAL;
    fRegressionAccuracy : REAL;
    ipClassPriors       : ITcVnContainer;
    hrPrev              : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eRTreesType	ETcVnRTrees [▶ 209]	随机树模型类型
nMaxDepth	UDINT	树模型最大深度（默认值：5）

名称	类型	描述
nMinSamples	UDINT	节点内需要分割的最小样本数（默认值：10）
nActiveVariables	UDINT	分割时考虑的变量数（0 表示 sqrt（变量总数）；默认值：0）
nMaxIterations	UDINT	最大迭代次数（如果等于 0 且 fEpsilon 不为 0.0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 50）
fEpsilon	LREAL	两次连续迭代之间允许的最大误差差值（如果差值等于 0.0 且 nMaxIterations 不为 0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 0.1）
fRegressionAccuracy	REAL	回归子的终止标准（仅适用于回归子；如果不允许，则设为默认值；默认值：0.0）
ipClassPriors	ITcVnContainer [▶ 365]	类优先级（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL；仅适用于分类器；可选，如果不需要或不允许，则设为 0；默认值：0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CreateRTReesModelExp2` 是F_VN_CreateRTreesModel [▶ 1388] 的专家变体。它包含额外的参数。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

模型类型

`eRTreesType` 指定随机森林模型是用于分类（`TCVN_RT_CLASSIFIER`）还是回归（`TCVN_RT_REGRESSOR`）。

最大树深度

`nMaxDepth` 表示树中决策层的最大数量。

节点内样本的最少数量

`nMinSamples` 是树形成过程中节点内必须保留的最小样本数。低于这一数字的拆分不会进行。

拆分特征

`nActiveVariables` 是用于拆分树的特征数量。如为 `0`，`sqrt(N)` 特征与所有可用特征数 `N` 一起使用。

最大迭代

优化最多使用 `nMaxIterations` 中指定的迭代次数。如果值为 `0`，则使用相应的默认值。

终止限制

只要两次迭代之间的误差变化不超过 `fEpsilon` 中的规定，优化就会终止。如果值为 `0`，则使用相应的默认值。

回归精度

`fRegressionAccuracy` 是回归优化的终止准则。对于分类，该值必须设置为 `0`。

类别先验

`ipClassPriors` 是一个容器，可用来指定单个类别的先验概率。对于回归，该值应设置为 `0`。

应用

例如，用于回归的 RTrees 模型可如下创建：

```
hr := F_VN_CreateRTreesExp2(
    ipMlModel           := ipMlModel,
    eRTreesType         := TCVN_RT_REGRESSOR,
    nMaxDepth           := 8,
    nMinSamples         := 4,
    nActiveVariables    := 0,
    nMaxIterations      := 0,
    fEpsilon            := 0,
    fRegressionAccuracy := 0.3,
    ipClassPriors       := 0,
    hrPrev              := hr);
```

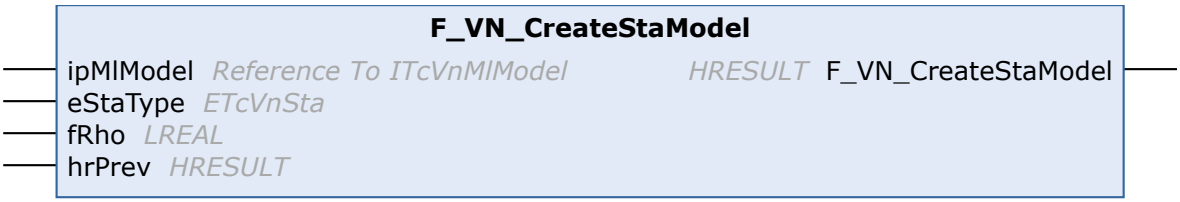
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.18 F_VN_CreateStaModel



创建指定类型的简化 TopoART 神经网络。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。此类模型支持在线训练（逐样本）、再训练以及标量和向量预测。它要求除类标签外的所有输入（即样本和训练输出/预测）都位于区间 [0.0，1.0]。在使用前，需要将回归子的预测值从区间 [0.0，1.0] 重新调整至其各自的值范围。可选择简化 TopoART 神经网络提供预测的异常程度。根据参数设置和可用训练样本的数量，使用相同数据进行重复训练可以改善结果。与其它基于自适应共振理论（ART）的神经网络一样，简化 TopoART 神经网络不易发生灾难性推理，尤其非常适合增量学习任务。

语法

定义：

```
FUNCTION F_VN_CreateStaModel : HRESULT
VAR_INPUT
    ipMlModel : Reference To ITcVnMlModel;
    eStaType  : ETcVnSta;
    fRho      : LREAL;
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eStaType	ETcVnSta [▶ 210]	简化 TopoART 模型类型
fRho	LREAL	警戒参数（控制插入的神经元数量和所形成类别的大小上限；有效范围：[0.0，1.0]；建议范围：[0.8，1.0]）

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_CreateStaModel 可创建简化 TopoART（STA）模型。

简化 TopoART 模型

简化 TopoART 网络学习特征空间的子区域。根据不同的应用，它们会被分组为不同的聚类、分配到不同的类别或用于预测。由于增量结构，可进行后期培训，从而以最佳可能方式保留已学习信息。只接受 0.0 和 1.0 之间的数值作为输入（类别名称除外）。因此，建议使用特征归一化。

模型

创建的模型将在接口指针 ipMlModel 中返回。

模型类型

eStaType 指定 STA 模型是用于分类（TCVN_STA_CLASSIFIER）、回归（TCVN_STA_REGRESSOR）、聚类（TCVN_STA_CLUSTERER）还是异常检测（TCVN_STA_NOVELTY_DETECTOR）。

警戒参数

该参数对调整模型以适应相应任务至关重要，因为它控制着要学习的子区域的最大尺寸。

专家参数

专家变体 F_VN_CreateStaModelExp [▶ 1394] 和 F_VN_CreateStaModelExp2 [▶ 1396] 包含附加参数。

应用

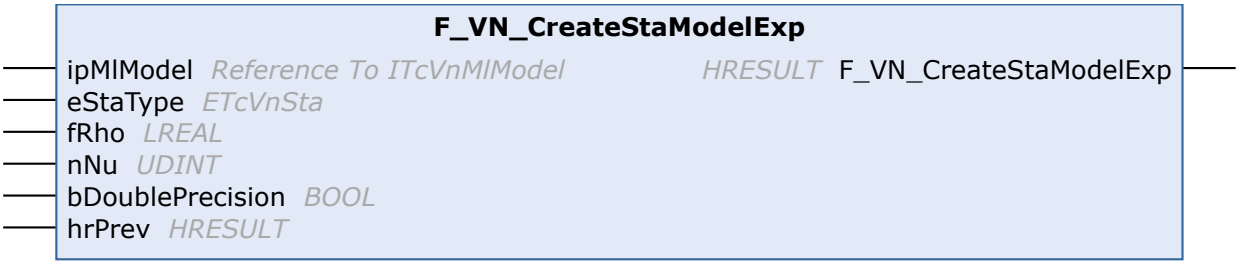
例如，用于分类的 STA 模型可如下创建：

```
hr := F_VN_CreateStaModel(
    ipMlModel    := ipMlModel,
    eStaType     := TCVN_STA_CLASSIFIER,
    fRho         := 0.9,
    hrPrev       := hr);
```

所需的许可证

TC3 Machine Learning Realtime Inference

6.1.4.20.19 F_VN_CreateStaModelExp



创建指定类型的简化 TopoART 神经网络。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。此类模型支持在线训练（逐样本）、再训练以及标量和向量预测。它要求除类标签外的所有输入（即样本和训练输出/预测）都位于区间 [0.0，1.0]。在使用前，需要将回归子的预测值从区间 [0.0，1.0] 重新

调整至其各自的值范围。可选择简化 TopoART 神经网络提供预测的异常程度。根据参数设置和可用训练样本的数量，使用相同数据进行重复训练可以改善结果。与其它基于自适应共振理论（ART）的神经网络一样，简化 TopoART 神经网络不易发生灾难性推理，尤其非常适合增量学习任务。（专家函数）

语法

定义：

```
FUNCTION F_VN_CreateStaModelExp : HRESULT
VAR_INPUT
    ipMlModel         : Reference To ITcVnMlModel;
    eStaType          : ETcVnSta;
    fRho              : LREAL;
    nNu               : UDINT;
    bDoublePrecision  : BOOL;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eStaType	ETcVnSta [▶ 210]	简化 TopoART 模型类型
fRho	LREAL	警戒参数（控制插入的神经元数量和所形成类别的大小上限；有效范围：[0.0，1.0]；建议范围：[0.8，1.0]）
nNu	UDINT	用于分类和预测的神经元数量（必须大于或等于 1；建议范围：[1，10]；默认值：3）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成模型。（默认设置：FALSE）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_CreateStaModelExp 是F_VN_CreateStaModel [▶ 1393] 的专家变体。它包含额外的参数。

参数

模型

创建的模型将在接口指针 ipMlModel 中返回。

模型类型

eStaType 指定 STA 模型是用于分类（TCVN_STA_CLASSIFIER）、回归（TCVN_STA_REGRESSOR）、聚类（TCVN_STA_CLUSTERER）还是异常检测（TCVN_STA_NOVELTY_DETECTOR）。

警戒参数

该参数对调整模型以适应相应任务至关重要，因为它控制着要学习的子区域的最大尺寸。

使用的邻算法数量

nNu 指定用于分类和预测的相邻子区域的数量，且必须大于等于 1。

更高精度

如果 bDoublePrecision 是 TRUE，则使用 LREAL 作为模型内部计算的数据类型；如果是 FALSE，则使用 REAL。

应用

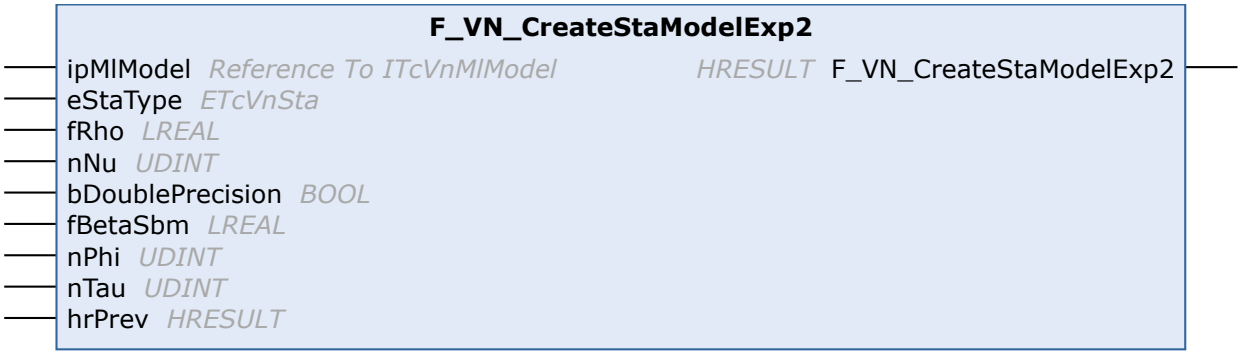
例如，用于分类的 STA 模型可如下创建：

```
hr := F_VN_CreateStaModel(
    ipMlModel         := ipMlModel,
    eStaType          := TCVN_STA_CLASSIFIER,
    fRho              := 0.9,
    nNu               := 3,
    bDoublePrecision  := FALSE,
    hrPrev            := hr);
```

所需的许可证

TC3 Machine Learning Realtime Inference

6.1.4.20.20 F_VN_CreateStaModelExp2



创建指定类型的简化 TopoART 神经网络。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。此类模型支持在线训练（逐样本）、再训练以及标量和向量预测。它要求除类标签外的所有输入（即样本和训练输出/预测）都位于区间 [0.0，1.0]。在使用前，需要将回归子的预测值从区间 [0.0，1.0] 重新调整至其各自的值范围。可选择简化 TopoART 神经网络提供预测的异常程度。根据参数设置和可用训练样本的数量，使用相同数据进行重复训练可以改善结果。与其它基于自适应共振理论（ART）的神经网络一样，简化 TopoART 神经网络不易发生灾难性推理，尤其非常适合增量学习任务。（附加专家函数，提供微调和降噪参数）

语法

定义：

```
FUNCTION F_VN_CreateStaModelExp2 : HRESULT
VAR_INPUT
    ipMlModel         : Reference To ITcVnMlModel;
    eStaType          : ETcVnSta;
    fRho              : LREAL;
    nNu               : UDINT;
    bDoublePrecision  : BOOL;
    fBetaSbm          : LREAL;
    nPhi              : UDINT;
    nTau              : UDINT;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eStaType	ETcVnSta [▶ 210]	简化 TopoART 模型类型
fRho	LREAL	警戒参数（控制插入的神经元数量和所形成类别的大小上限；有效范围：[0.0，1.0]；建议范围：[0.8，1.0]）

名称	类型	描述
nNu	UDINT	用于分类和预测的神经元数量（必须大于或等于 1；建议范围：[1，10]；默认值：3）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成模型。（默认设置：FALSE）
fBetaSbm	LREAL	第二最佳匹配神经元的学习率（学习第二最佳匹配神经元可使相关类别更接近，并可能改善结果，但可能需要更多神经元并运行更多训练；值为 0.0 时禁用第二最佳匹配神经元的学习；有效范围：[0.0，1.0]；建议范围：[0.0，0.5]）；默认值：0.0
nPhi	UDINT	一个神经元为了成为永久性神经元而必须学习的样本数量（降噪所需；与 nTau 配合使用；必须大于或等于 1；数值越大，降噪效果越强；数值为 1 时，禁止删除候选神经元；默认值：1）
nTau	UDINT	移除节点后执行的学习步骤数量（降噪所需；与 nPhi 配合使用；必须足够大，以便在移除节点之前，让代表非噪声样本的候选神经元成为最佳匹配神经元至少 nPhi 次；默认值：100）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_CreateStaModelExp2 是F_VN_CreateStaModel [▶ 1393] 的专家变体。它包含额外的参数。

参数

模型

创建的模型将在接口指针 ipMlModel 中返回。

模型类型

eStaType 指定 STA 模型是用于分类（TCVN_STA_CLASSIFIER）、回归（TCVN_STA_REGRESSOR）、聚类（TCVN_STA_CLUSTERER）还是异常检测（TCVN_STA_NOVELTY_DETECTOR）。

警戒参数

该参数对调整模型以适应相应任务至关重要，因为它控制着要学习的子区域的最大尺寸。

使用的邻算法数量

nNu 指定用于分类和预测的相邻子区域的数量，且必须大于等于 1。

更高精度

如果 bDoublePrecision 是 TRUE，则使用 LREAL 作为模型内部计算的数据类型；如果是 FALSE，则使用 REAL。

调整分配

fBetaSbm 补充根据基本样本分配调整子区域的控制。

噪音抑制

nPhi 和 nTau 用于控制学习过程中对极少数样本（噪声、异常值）的抑制。

nPhi 指定在一个子区域内必须学习的样本数，以便永久考虑该子区域。

nTau 指定在多少个学习步骤后检查 nPhi 的条件，并删除不满足该条件的子区域

增加 nPhi 和减少 nTau 可以增加噪音抑制，因此 nPhi 与 nTau 的比值可用于估算强度。

应用

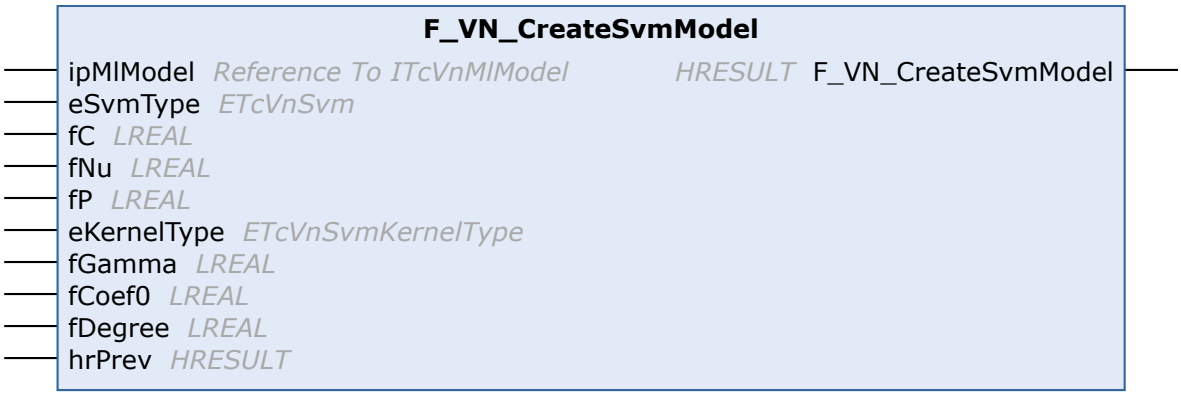
例如，用于分类的 STA 模型可如下创建：

```
hr := F_VN_CreateStaModel(
    ipMlModel         := ipMlModel,
    eStaType          := TCVN_STA_CLASSIFIER,
    fRho              := 0.9,
    nNu               := 3,
    bDoublePrecision  := FALSE,
    fBetaSbm          := 0.0,
    nPhi              := 1,
    nTau              := 100,
    hrPrev            := hr);
```

所需的许可证

TC3 Machine Learning Realtime Inference

6.1.4.20.21 F_VN_CreateSvmModel



Create an SVM model of the specified type. The initial reference count is set to one if a new model is created and kept, otherwise. Models of this type neither support on-line training (sample by sample) nor retraining. Predictions can only be scalar.

语法

定义：

```
FUNCTION F_VN_CreateSvmModel : HRESULT
VAR_INPUT
    ipMlModel    : Reference To ITcVnMlModel;
    eSvmType     : ETcVnSvm;
    fC           : LREAL;
    fNu          : LREAL;
    fP           : LREAL;
    eKernelType  : ETcVnSvmKernelType;
    fGamma       : LREAL;
    fCoef0       : LREAL;
    fDegree      : LREAL;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eSvmType	ETcVnSvm [▶ 211]	SVM 模型类型
fC	LREAL	参数 C（对于 TCVN_SVM_C_CLASSIFIER、TCVN_SVM_EPS_REGRESSOR 和 TCVN_SVM_NU_REGRESSOR 为必需参数；其它情况可忽略）

名称	类型	描述
fNu	LREAL	参数 nu（对于 TCVN_SVM_NU_CLASSIFIER、TCVN_SVM_NOVELTY_DETECTOR 和 TCVN_SVM_NU_REGRESSOR 为必需参数；其它情况可忽略）
fP	LREAL	参数 p（对于 TCVN_SVM_EPS_REGRESSOR 为必需参数，其它情况可忽略）
eKernelType	ETcVnSvmKernelType [▶ 212]	内核类型
fGamma	LREAL	参数 gamma（多项式核、RBF 核、S 形核和卡方核使用；其它情况下忽略不计）
fCoef0	LREAL	参数 coef0（多项式核和 S 形核使用，其它情况下忽略不计）
fDegree	LREAL	度数（多项式核使用，其它情况下可忽略）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CreateSvmModel` 可创建支持向量机（SVM）模型。

支持向量机模型

支持向量机还可用于复杂的非线性问题。预测完全基于所谓的支持向量，这就是 SVM 模型内存效率相对较高的原因。需要同时对所有样本进行训练（批量训练），无法进行后期训练。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

模型类型

`eSvmType` 指定 SVM 模型是用于分类、回归还是异常检测：

- TCVN_SVM_C_CLASSIFIER
- TCVN_SVM_NU_CLASSIFIER
- TCVN_SVM_NOVELTY_DETECTOR
- TCVN_SVM_EPS_REGRESSOR
- TCVN_SVM_NU_REGRESSOR

Model parameters

参数 `fC`、`fNu` 和 `fP` 的使用和含义取决于所选的 `eSvmType`。

内核类型

用于计算的内核类型通过 `eKernelType` 进行定义。内核类型取决于任务/数据分布，且必须进行相应调整。

内核参数

参数 `fGamma`、`fCoef0` 和 `fDegree` 的使用和含义取决于所选的 `eKernelType`。

专家参数

专家变体 F_VN_CreateSvmModelExp [▶ 1400] 和 F_VN_CreateSvmModelExp2 [▶ 1402] 包含附加参数。

应用

例如，用于分类的 SVM 模型可如下创建：

```
hr := F_VN_CreateSvmModel(
    ipMlModel    := ipMlModel,
    eSvmType     := TCVN_SVM_NU_CLASSIFIER,
    fC           := 0,
    fNu          := 0.1,
    fP           := 0,
    eKernelType  := TCVN_SKT_RBF,
    fGamma       := 1,
    fCoef0       := 0,
    fDegree      := 0,
    hrPrev       := hr);
```

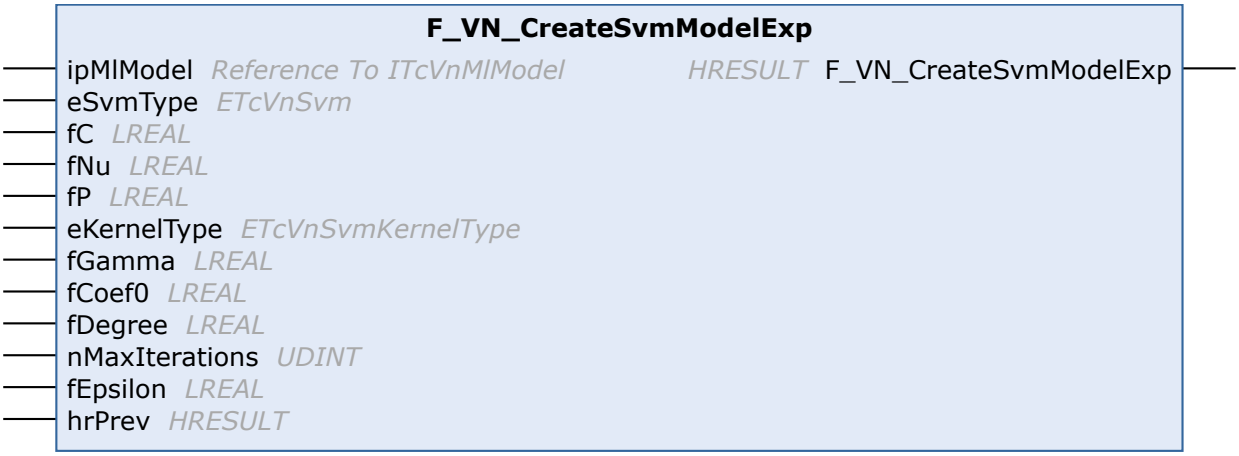
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.22 F_VN_CreateSvmModelExp



Create an SVM model of the specified type. The initial reference count is set to one if a new model is created and kept, otherwise. Models of this type neither support on-line training (sample by sample) nor retraining. Predictions can only be scalar. (expert function)

语法

定义：

```
FUNCTION F_VN_CreateSvmModelExp : HRESULT
VAR_INPUT
    ipMlModel      : Reference To ITcVnMlModel;
    eSvmType       : ETcVnSvm;
    fC             : LREAL;
    fNu            : LREAL;
    fP             : LREAL;
    eKernelType    : ETcVnSvmKernelType;
    fGamma         : LREAL;
    fCoef0         : LREAL;
    fDegree        : LREAL;
    nMaxIterations : UDINT;
    fEpsilon       : LREAL;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eSvmType	ETcVnSvm [▶ 211]	SVM 模型类型
fC	LREAL	参数 C（对于 TCVN_SVM_C_CLASSIFIER、TCVN_SVM_EPS_REGRESSOR 和 TCVN_SVM_NU_REGRESSOR 为必需参数；其它情况可忽略）
fNu	LREAL	参数 nu（对于 TCVN_SVM_NU_CLASSIFIER、TCVN_SVM_NOVELTY_DETECTOR 和 TCVN_SVM_NU_REGRESSOR 为必需参数；其它情况可忽略）
fP	LREAL	参数 p（对于 TCVN_SVM_EPS_REGRESSOR 为必需参数，其它情况可忽略）
eKernelType	ETcVnSvmKernelType [▶ 212]	内核类型
fGamma	LREAL	参数 gamma（多项式核、RBF 核、S 形核和卡方核使用；其它情况下忽略不计）
fCoef0	LREAL	参数 coef0（多项式核和 S 形核使用；其它情况下忽略不计）
fDegree	LREAL	度数（多项式核使用，其它情况下可忽略）
nMaxIterations	UDINT	最大迭代次数（如果等于 0 且 fEpsilon 不为 0.0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 100000）
fEpsilon	LREAL	两次连续迭代之间允许的最大误差差值（如果差值等于 0.0 且 nMaxIterations 不为 0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 0.00001）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CreateSvmModelExp` 是F_VN_CreateSvmModel [▶ 1398] 的专家变体。它包含额外的参数。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

模型类型

`eSvmType` 指定 SVM 模型是用于分类、回归还是异常检测：

- TCVN_SVM_C_CLASSIFIER
- TCVN_SVM_NU_CLASSIFIER
- TCVN_SVM_NOVELTY_DETECTOR
- TCVN_SVM_EPS_REGRESSOR
- TCVN_SVM_NU_REGRESSOR

Model parameters

参数 `fC`、`fNu` 和 `fP` 的使用和含义取决于所选的 `eSvmType`。

内核类型

用于计算的内核类型通过 `eKernelType` 进行定义。内核类型取决于任务/数据分布，且必须进行相应调整。

内核参数

参数 `fGamma`、`fCoef0` 和 `fDegree` 的使用和含义取决于所选的 `eKernelType`。

最大迭代

优化最多使用 `nMaxIterations` 中指定的迭代次数。如果值为 `0`，则使用相应的默认值。

终止限制

只要两次迭代之间的误差变化不超过 `fEpsilon` 中的规定，优化就会终止。如果值为 `0`，则使用相应的默认值。

应用

例如，用于分类的 SVM 模型可如下创建：

```
hr := F_VN_CreateSvmModelExp(
    ipMlModel       := ipMlModel,
    eSvmType        := TCVN_SVM_NU_CLASSIFIER,
    fC              := 0,
    fNu             := 0.1,
    fP              := 0,
    eKernelType     := TCVN_SKT_RBF,
    fGamma          := 1,
    fCoef0          := 0,
    fDegree         := 0,
    nMaxIterations  := 0,
    fEpsilon        := 0,
    hrPrev          := hr);
```

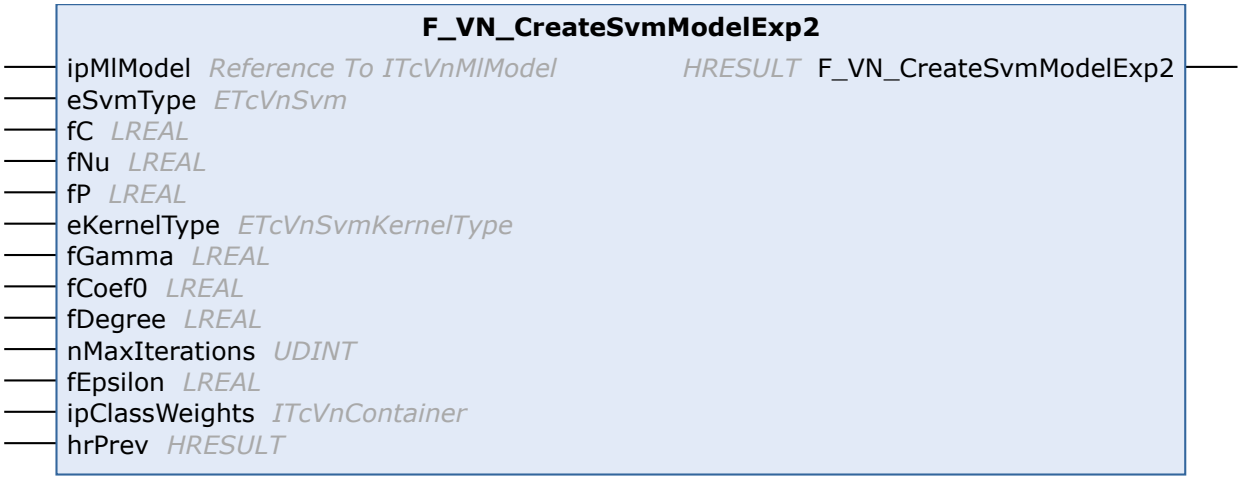
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.23 F_VN_CreateSvmModelExp2



Create an SVM model of the specified type. The initial reference count is set to one if a new model is created and kept, otherwise. Models of this type neither support on-line training (sample by sample) nor retraining. Predictions can only be scalar. (additional expert function for C support vector classifiers)

语法

定义：

```
FUNCTION F_VN_CreateSvmModelExp2 : HRESULT
VAR_INPUT
    ipMlModel       : Reference To ITcVnMlModel;
    eSvmType        : ETcVnSvm;
    fC              : LREAL;
    fNu             : LREAL;
    fP              : LREAL;
    eKernelType     : ETcVnSvmKernelType;
    fGamma          : LREAL;
    fCoef0          : LREAL;
    fDegree         : LREAL;
    nMaxIterations  : UDINT;
    fEpsilon        : LREAL;
    ipClassWeights  : ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eSvmType	ETcVnSvm [▶ 211]	SVM 模型类型
fC	LREAL	参数 C（对于 TCVN_SVM_C_CLASSIFIER、TCVN_SVM_EPS_REGRESSOR 和 TCVN_SVM_NU_REGRESSOR 为必需参数；其它情况可忽略）
fNu	LREAL	参数 nu（对于 TCVN_SVM_NU_CLASSIFIER、TCVN_SVM_NOVELTY_DETECTOR 和 TCVN_SVM_NU_REGRESSOR 为必需参数；其它情况可忽略）
fP	LREAL	参数 p（对于 TCVN_SVM_EPS_REGRESSOR 为必需参数，其它情况可忽略）
eKernelType	ETcVnSvmKernelType [▶ 212]	内核类型
fGamma	LREAL	参数 gamma（多项式核、RBF 核、S 形核和卡方核使用；其它情况下忽略不计）
fCoef0	LREAL	参数 coef0（多项式核和 S 形核使用；其它情况下忽略不计）
fDegree	LREAL	度数（多项式核使用，其它情况下可忽略）
nMaxIterations	UDINT	最大迭代次数（如果等于 0 且 fEpsilon 不为 0.0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 100000）
fEpsilon	LREAL	两次连续迭代之间允许的最大误差差值（如果差值等于 0.0 且 nMaxIterations 不为 0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 0.00001）
ipClassWeights	ITcVnContainer [▶ 365]	类权重（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL；仅当 eSvmType 等于 TCVN_SVM_C_CLASSIFIER 时有效；可选，如果不需要或不允许，则设为 0；默认值：0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_CreateSvmModelExp2 是F_VN_CreateSvmModel [▶ 1398] 的专家变体。它包含只与 TCVN_SVM_C_CLASSIFIER 类型模型相关的附加参数。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

模型类型

`eSvmType` 指定 SVM 模型是用于分类、回归还是异常检测：

- TCVN_SVM_C_CLASSIFIER
- TCVN_SVM_NU_CLASSIFIER
- TCVN_SVM_NOVELTY_DETECTOR
- TCVN_SVM_EPS_REGRESSOR
- TCVN_SVM_NU_REGRESSOR

Model parameters

参数 `fC`、`fNu` 和 `fP` 的使用和含义取决于所选的 `eSvmType`。

内核类型

用于计算的内核类型通过 `eKernelType` 进行定义。内核类型取决于任务/数据分布，且必须进行相应调整。

内核参数

参数 `fGamma`、`fCoef0` 和 `fDegree` 的使用和含义取决于所选的 `eKernelType`。

最大迭代

优化最多使用 `nMaxIterations` 中指定的迭代次数。如果值为 `0`，则使用相应的默认值。

终止限制

只要两次迭代之间的误差变化不超过 `fEpsilon` 中的规定，优化就会终止。如果值为 `0`，则使用相应的默认值。

分类权重

`ipClassWeights` 是一个容器，可用于指定单个类别的权重。

应用

例如，使用 Nu 进行参数设置的 SVM 分类模型可如下创建：

```
hr := F_VN_CreateSvmModelExp2(
    ipMlModel        := ipMlModel,
    eSvmType         := TCVN_SVM_C_CLASSIFIER,
    fC               := 100,
    fNu              := 0,
    fP               := 0,
    eKernelType      := TCVN_SKT_RBF,
    fGamma           := 1,
    fCoef0           := 0,
    fDegree          := 0,
    nMaxIterations   := 0,
    fEpsilon         := 0,
    ipClassWeights   := ipClassWeights,
    hrPrev           := hr);
```

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.24 F_VN_CreateSvmSgdClassifier



Create a linear SVM classifier using stochastic gradient descent for training. The initial reference count is set to one if a new model is created and kept, otherwise. This SVM classifier is only applicable to binary classification problems. It learns a separating hyperplane between a class with label -1 and a class with label 1. These class labels are predefined. For training, any positive class labels are mapped to 1 and any negative class labels are mapped to -1. Models of this type neither support on-line training (sample by sample) nor retraining.

语法

定义：

```
FUNCTION F_VN_CreateSvmSgdClassifier : HRESULT
VAR_INPUT
    ipMlModel : Reference To ITcVnMlModel;
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_CreateSvmSgdClassifier 可创建一个带有可用于随机梯度下降（SVM-SGD）训练的线性内核的支持向量机模型。

SVM-SGD 模型

这种 SVM 分类器模型只适用于二元分类问题。它在标签为 -1 的类和标签为 1 的类之间学习一个分离超平面。所有样本均需要同时进行训练（批量训练），不可能进行后期训练。

参数

模型

创建的模型将在接口指针 ipMlModel 中返回。

专家参数

专家变体 F_VN_CreateSvmSgdClassifierExp [▶ 1406] 包含附加参数。

应用

例如，用于分类的 SVM-SGD 模型可如下创建：

```
hr := F_VN_CreateSvmSgdClassifier(
    ipMlModel   := ipMlModel,
    hrPrev      := hr);
```

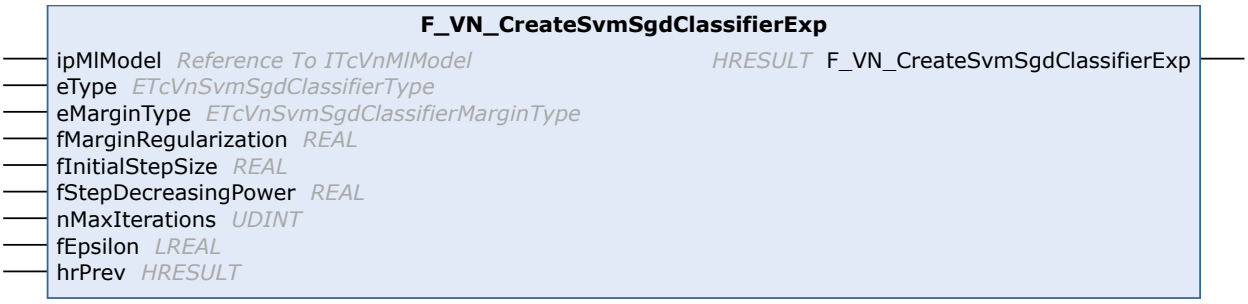
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.25 F_VN_CreateSvmSgdClassifierExp



Create a linear SVM classifier using stochastic gradient descent for training. The initial reference count is set to one if a new model is created and kept, otherwise. This SVM classifier is only applicable to binary classification problems. It learns a separating hyperplane between a class with label -1 and a class with label 1. These class labels are predefined. For training, any positive class labels are mapped to 1 and any negative class labels are mapped to -1. Models of this type neither support on-line training (sample by sample) nor retraining. (expert function)

语法

定义：

```
FUNCTION F_VN_CreateSvmSgdClassifierExp : HRESULT
VAR_INPUT
    ipMlModel              : Reference To ITcVnMlModel;
    eType                  : ETcVnSvmSgdClassifierType;
    eMarginType            : ETcVnSvmSgdClassifierMarginType;
    fMarginRegularization  : REAL;
    fInitialStepSize       : REAL;
    fStepDecreasingPower   : REAL;
    nMaxIterations         : UDINT;
    fEpsilon               : LREAL;
    hrPrev                 : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]	返回创建的模型（重复使用非零接口指针。）
eType	ETcVnSvmSgdClassifierType [▶ 212]	学习算法类型（默认类型：TCVN_SSCT_ASGD）
eMarginType	ETcVnSvmSgdClassifierMarginType [▶ 212]	边际类型（默认类型：TCVN_SSCMT_SOFT_MARGIN）
fMarginRegularization	REAL	边际正则化参数（默认值：0.00001）
fInitialStepSize	REAL	初始步长（默认值：0.05）

名称	类型	描述
fStepDecreasingPower	REAL	幂参数（默认值：0.75）
nMaxIterations	UDINT	最大迭代次数（如果等于 0 且 fEpsilon 不为 0.0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 100000）
fEpsilon	LREAL	两次连续迭代之间允许的最大误差差值（如果差值等于 0.0 且 nMaxIterations 不为 0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 0.00001）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_CreateSvmSgdClassifierExp` 是 F_VN_CreateSvmSgdClassifier [▶ 1405] 的专家变体。它包含额外的参数。

参数

模型

创建的模型将在接口指针 `ipMlModel` 中返回。

算法

`eType` 指定训练中使用的优化方法：

- TCVN_SSCT_SGD 随机梯度下降
- TCVN_SSCT_ASGD 平均随机梯度下降

分类界定

`eMarginType` 指定是严格（`TCVN_SSCMT_HARD_MARGIN`）还是使用离群值（`TCVN_SSCMT_SOFT_MARGIN`）进行类别界定。

正则化

`fMarginRegularization` 负责每一步的权重减少，并控制异常值影响的限制程度。参数越小，异常值被忽略的概率就越低。

步长

`fInitialStepSize` 表示培训开始时的步长。

减少步长

`fStepDecreasingPower` 指定用于减小步长的指数。

最大迭代

优化最多使用 `nMaxIterations` 中指定的迭代次数。如果值为 `0`，则使用相应的默认值。

终止限制

只要两次迭代之间的误差变化不超过 `fEpsilon` 中的规定，优化就会终止。如果值为 `0`，则使用相应的默认值。

应用

例如，用于分类的 SVM-SGD 模型可如下创建：

```
hr := F_VN_CreateSvmSgdClassifierExp(
    ipMlModel               := ipMlModel,
    eType                   := TCVN_SSCT_ASGD,
    eMarginType             := TCVN_SSCMT_SOFT_MARGIN,
    fMarginRegularization   := 0.00001,
    fInitialStepSize        := 0.05,
    fStepDecreasingPower    := 0.75,
    nMaxIterations          := 0,
    fEpsilon                := 0,
    hrPrev                  := hr);
```

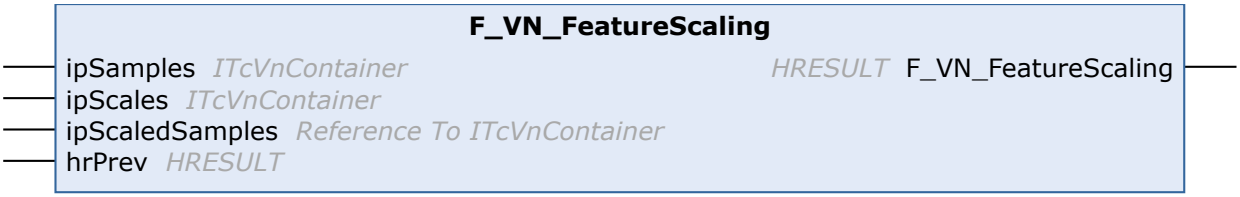
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.26 F_VN_FeatureScaling



Apply a feature scaling to one or more sample(s).

语法

定义：

```
FUNCTION F_VN_FeatureScaling : HRESULT
VAR_INPUT
    ipSamples       : ITcVnContainer;
    ipScales        : ITcVnContainer;
    ipScaledSamples : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSamples	ITcVnContainer [▶ 365]	包含一个或多个输入样本的容器（ContainerType_Vector_REAL、ContainerType_Vector_LREAL、ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_LREAL）
ipScales	ITcVnContainer [▶ 365]	包含每个特征的缩放参数的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）。
ipScaledSamples	引用 ITcVnContainer [▶ 365]	使用与 ipSamples 相同的类型返回缩放后的样本。如果使用同一个容器 ipSamples，源数据将被替换。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_FeatureScaling 对来自多个样本的特征进行缩放。函数 F_VN_InverseFeatureScaling [▶ 1425] 可用于反向缩放。

参数

样本

必须将一个或多个样本作为一个容器传输到 ipSamples。

缩放

必须将带有缩放类型和相应缩放参数的容器传输到 ipScales。容器可以使用函数 F_VN_GetFeatureScales [▶ 1419] 计算，也可以使用其中描述的结构创建。

缩放样本

缩放样本通过引用 ipScaledSamples 返回。

专家参数

专家变体 F_VN_FeatureScalingExp [▶ 1410] 包含附加参数。

应用

例如，F_VN_GetFeatureScales [▶ 1419] 使用MINMAX 方法事先计算缩放参数的特征缩放过程如下所示：

```
hr := F_VN_GetFeatureScales(ipSamples, ipScales, TCVN_FST_MINMAX, hr);

hr := F_VN_FeatureScaling(
    ipSamples       := ipSamples,
    ipScales        := ipScales,
    ipScaledSamples := ipScaledSamples,
    hrPrev          := hr);
```

相关函数

- F_VN_GetFeatureScales [▶ 1419]
- F_VN_InverseFeatureScaling [▶ 1425]
- F_VN_InverseFeatureScaling_REAL [▶ 1427]

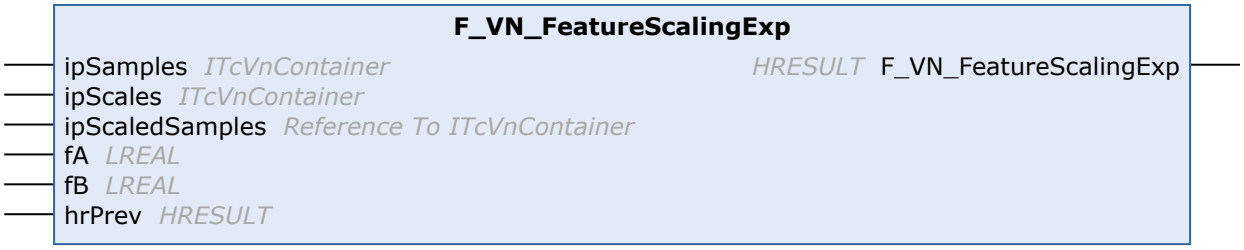
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.27 F_VN_FeatureScalingExp



Apply a feature scaling to one or more sample(s) and get output range between [fA, fB] in the case of TCVN_FST1_MINMAX. (expert function)

语法

定义：

```
FUNCTION F_VN_FeatureScalingExp : HRESULT
VAR_INPUT
    ipSamples       : ITcVnContainer;
    ipScales        : ITcVnContainer;
    ipScaledSamples : Reference To ITcVnContainer;
    fA              : LREAL;
    fB              : LREAL;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSamples	ITcVnContainer [▶ 365]	包含一个或多个输入样本的容器（ContainerType_Vector_REAL、ContainerType_Vector_LREAL、ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_LREAL）
ipScales	ITcVnContainer [▶ 365]	包含每个特征的缩放参数的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）。
ipScaledSamples	引用 ITcVnContainer [▶ 365]	使用与 ipSamples 相同的类型返回缩放后的样本。如果使用同一个容器 ipSamples，源数据将被替换。
fA	LREAL	表示 TCVN_FST1_MINMAX 范围的下限
fB	LREAL	表示 TCVN_FST1_MINMAX 范围的上限
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_FeatureScalingExp` 是F_VN_FeatureScaling [▶ 1408] 的专家变体。它包含额外的参数。

参数

样本

必须将一个或多个样本作为一个容器传输到 `ipSamples`。

缩放

必须将带有缩放类型和相应缩放参数的容器传输到 `ipScales`。容器可以使用函数 F_VN_GetFeatureScales [▶ 1419] 计算，也可以使用其中描述的结构创建。

缩放样本

缩放样本通过引用 `ipScaledSamples` 返回。

定义输出范围

如果使用缩放类型 `TCVN_FST1_MINMAX`，还可以通过 `fA` 和 `fB` 指定输出值的值范围。

应用

例如，输出值范围为 0.1 至 0.9 的特征缩放如下所示：

```
hr := F_VN_FeatureScalingExp(
    ipSamples       := ipSamples,
    ipScales        := ipScales,
    ipScaledSamples := ipScaledSamples,
    fA              := 0.1,
    fB              := 0.9,
    hrPrev          := hr);
```

相关函数

- F_VN_GetFeatureScales [▶ 1419]
- F_VN_InverseFeatureScaling [▶ 1425]
- F_VN_InverseFeatureScaling_REAL [▶ 1427]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.28 F_VN_FeatureTransform



Apply a feature transform to one or more sample(s).

语法

定义：

```
FUNCTION F_VN_FeatureTransform : HRESULT
VAR_INPUT
    ipFeatureTransform : ITcVnMlModel;
    ipSamples          : ITcUnknown;
    pipTransformed     : Pointer To ITcUnknown;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipFeatureTransform	ITcVnMlModel [▶ 417]	要使用的特征变换实例

名称	类型	描述
ipSamples	ITcUnknown [▶ 424]	包含一个或多个输入样本的容器（ContainerType_Vector_REAL、ContainerType_Vector_LREAL、ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_LREAL）
pipTransformed	指针指向 ITcUnknown [▶ 424]	使用与 ipSamples 相同的类型返回变换后的样本
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_FeatureTransform` 使用转换模型对一个或多个样本的特征进行转换。可以使用 F_VN_InverseFeatureTransform [▶ 1432] 函数进行逆转换。

参数

转换模型

先前创建模型必须传输到 `ipFeatureTransform` 进行特征转换。

样本

具有待转换样本的容器会被传输到 `ipSamples`。允许使用以下容器类型：

- `ContainerType_Vector_REAL` 或 `ContainerType_Vector_LREAL`，用于单个样本
- `ContainerType_Vector_Vector_REAL` 或 `ContainerType_Vector_Vector_LREAL`，用于具有多个样本的容器。

转换样本

通过指针 `pipTransformed` 返回具有转换样本的容器。容器类型取自 `ipSamples`。

应用

例如，特征转换如下所示进行：

```
hr := F_VN_FeatureTransform(
    ipFeatureTransform  := ipMlModel,
    ipSamples           := ipSamples,
    pipTransformed      := ADR(ipTransformed),
    hrPrev              := hr);
```

相关函数

- F_VN_InverseFeatureTransform [▶ 1432]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.29 F_VN_GetBatchClusters



Get the IDs of the best-matching clusters of a batch of samples.

语法

定义：

```
FUNCTION F_VN_GetBatchClusters : HRESULT
VAR_INPUT
    ipClusterer : ITcVnMlModel;
    ipSamples   : ITcUnknown;
    ipClusters  : Reference To ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipClusterer	ITcVnMlModel [▶ 417]	要使用的聚类器
ipSamples	ITcUnknown [▶ 424]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
ipClusters	引用 ITcVnContainer [▶ 365]	返回聚类 ID（ContainerType_Vector_DINT）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_GetBatchClusters` 可确定容器中所有样本的最佳匹配聚类，并返回其 ID。

参数

聚类器模型

先前培训模型必须传输到 `ipClusterer`，以便分配至聚类。

样本

具有样本的容器被传输到 `ipSamples`。容器类型必须是 `ContainerType_Vector_Vector_REAL` 或 `ContainerType_Vector_Vector_LREAL`。

聚类 ID

将通过引用 `ipClusters` 返回具有已确定 ID 的容器。

专家参数

专家变体 F_VN_GetBatchClustersExp [▶ 1414] 包含附加参数。

应用

通过以下方法可以计算出样本至聚类的分配：

```
hr := F_VN_GetBatchClusters(
    ipClusterer := ipClusterer,
    ipSamples   := ipSamples,
    ipClusters  := ipClusters,
    hrPrev      := hr);
```

相关函数

- F_VN_GetSampleCluster [▶ 1421]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.30 F_VN_GetBatchClustersExp



Get the IDs of the best-matching clusters of a batch of samples.

语法

定义：

```
FUNCTION F_VN_GetBatchClustersExp : HRESULT
VAR_INPUT
    ipClusterer : ITcVnMlModel;
    ipSamples   : ITcUnknown;
    ipClusters  : Reference To ITcVnContainer;
    ipNovelties : Reference To ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipClusterer	ITcVnMlModel [▶ 417]	要使用的聚类器
ipSamples	ITcUnknown [▶ 424]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
ipClusters	引用 ITcVnContainer [▶ 365]	返回聚类 ID（ContainerType_Vector_DINT）
ipNovelties	引用 ITcVnContainer [▶ 365]	返回每个样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）（ContainerType_Vector_REAL；可选，如果不需要，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_GetBatchClustersExp 是F_VN_GetBatchClusters [▶ 1413] 的专家变体。它包含额外的参数。

参数

聚类器模型

先前培训模型必须传输到 ipClusterer，以便分配至聚类。

样本

具有样本的容器被传输到 ipSamples。容器类型必须是 ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL。

聚类 ID

将通过引用 ipClusters 返回具有已确定 ID 的容器。

新颖性

具有每个样本 新颖性的容器将通过引用 ipNovelties 返回。新颖性表示样本与聚类的差异程度。0 的 新颖性表示样本正好位于聚类的中心。

应用

样本至聚类的分配（包括新颖性）可如下计算：

```
hr := F_VN_GetBatchClustersExp(
    ipClusterer := ipClusterer,
    ipSamples   := ipSamples,
    ipClusters  := ipClusters,
    ipNovelties := ipNovelties,
    hrPrev      := hr);
```

相关函数

- F_VN_GetSampleCluster [▶ 1421]
- F_VN_GetSampleNovelty [▶ 1424]
- F_VN_GetBatchNovelties [▶ 1415]

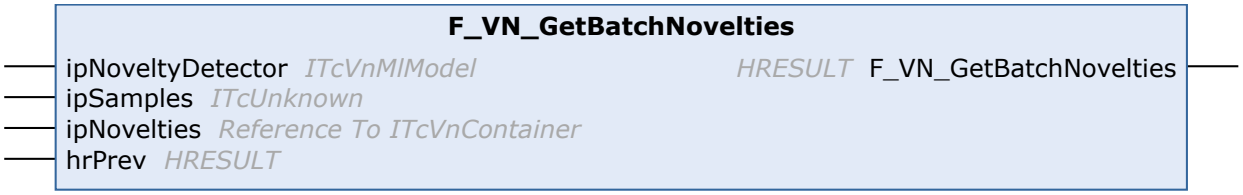
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.31 F_VN_GetBatchNovelties



Get the degree of novelty of multiple samples.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_GetBatchNovelties : HRESULT
VAR_INPUT
    ipNoveltyDetector : ITcVnMlModel;
    ipSamples         : ITcUnknown;
    ipNovelties       : Reference To ITcVnContainer;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipNoveltyDetector	ITcVnMlModel [▶ 417]	要使用异常探测器
ipSamples	ITcUnknown [▶ 424]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
ipNovelties	引用 ITcVnContainer [▶ 365]	返回每个样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）（ContainerType_Vector_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_GetBatchNovelties` 可确定容器中所有样本的新颖性。

参数

异常模型

要计算新颖性，必须将先前训练模型传输到 `ipNoveltyDetector`。

样本

具有样本的容器被传输到 `ipSamples`。

新颖性

具有计算新颖性的容器将通过引用 `ipNovelties` 返回。

应用

多个样本的新颖性可如下计算：

```
hr := F_VN_GetBatchNovelties(
    ipNoveltyDetector   := ipNoveltyDetector,
    ipSamples           := ipSamples,
    ipNovelties         := ipNovelties,
    hrPrev              := hr);
```

相关函数

- F_VN_GetSampleNovelty [▶ 1424]

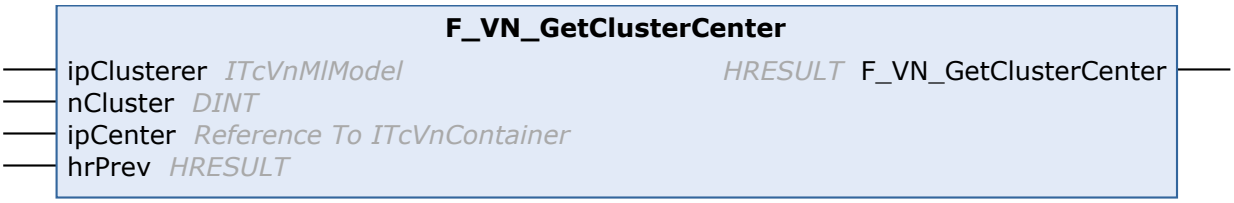
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.32 F_VN_GetClusterCenter



Get the center of a cluster.

语法

定义：

```
FUNCTION F_VN_GetClusterCenter : HRESULT
VAR_INPUT
    ipClusterer : ITcVnMlModel;
    nCluster    : DINT;
    ipCenter    : Reference To ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipClusterer	ITcVnMlModel [▶ 417]	要使用的聚类器
nCluster	DINT	聚类 ID，需要该聚类的中心
ipCenter	引用 ITcVnContainer [▶ 365]	返回聚类中心（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL，取决于聚类器内部使用的浮点类型）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_GetClusterCenter` 返回聚类的中心。

参数

聚类器模型

要计算其中心的聚类必须包含在先前训练模型 `ipClusterer` 中。

聚类 ID

相应聚类的 ID 将传输到 `nCluster`。

聚类中心

聚类中心通过引用 `ipCenter` 返回。根据 `ipClusterer` 内部使用的数据类型，返回的容器类型是 `ContainerType_Vector_REAL` 或 `ContainerType_Vector_LREAL`。

应用

聚类模型中第三个聚类的中心可如下确定：

```
hr := F_VN_GetClusterCenter(
    ipClusterer := ipClusterer,
    nCluster    := 2,
    ipCenter    := ipCenter,
    hrPrev      := hr);
```

相关函数

- F_VN_GetClusterNum [▶ 1418]

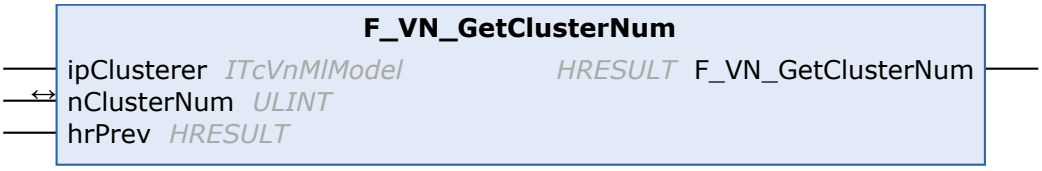
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.33 F_VN_GetClusterNum



Get the number of clusters used by this clusterer.

语法

定义：

```
FUNCTION F_VN_GetClusterNum : HRESULT
VAR_INPUT
    ipClusterer : ITcVnMlModel;
END_VAR
VAR_IN_OUT
    nClusterNum : ULINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipClusterer	ITcVnMlModel [▶ 417]	要使用的聚类器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nClusterNum	ULINT	返回聚类数目

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_GetClusterNum 返回 聚类器模型中的聚类数。

参数

聚类器模型

先前培训聚类器模型必须传输到 ipClusterer。

聚类数

聚类数通过 nClusterNum 返回。

应用

例如，聚类数可如下确定：

```
hr := F_VN_GetClusterNum(
    ipClusterer    := ipClusterer,
    nClusterNum    := nClusterNum,
    hrPrev         := hr);
```

相关函数

- F_VN_GetClusterCenter [▶ 1417]

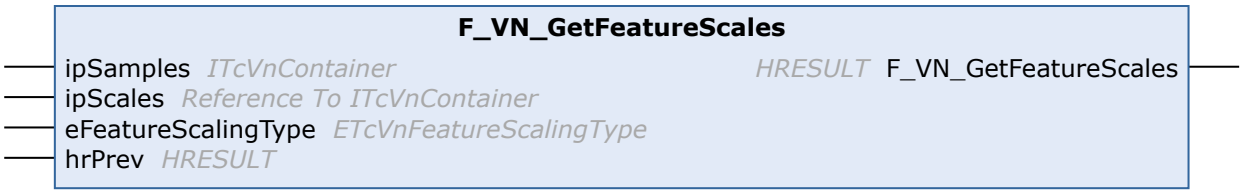
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.34 F_VN_GetFeatureScales



Calculate the scaling parameters for each feature in the input samples based on the scaling type.

语法

定义：

```
FUNCTION F_VN_GetFeatureScales : HRESULT
VAR_INPUT
    ipSamples           : ITcVnContainer;
```

```
    ipScales            : Reference To ITcVnContainer;
    eFeatureScalingType : ETcVnFeatureScalingType;
    hrPrev              : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSamples	ITcVnContainer [▶ 365]	包含输入样本的容器（ContainerType_Vector_REAL、ContainerType_Vector_LREAL、ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_LREAL）
ipScales	引用 ITcVnContainer [▶ 365]	返回一个容器，其中包含每个特征的缩放参数（Vector_REAL 或 Vector_LREAL，取决于 ipSamples 的类型）。
eFeatureScalingType	ETcVnFeatureScalingType [▶ 194]	特征缩放类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_GetFeatureScales` 可为传输样本的每个特征值计算缩放参数。然后可以使用 F_VN_FeatureScaling [▶ 1408] 函数进行相应的缩放，以实现特征归一化。同样的缩放比例也可以与 F_VN_InverseFeatureScaling [▶ 1425] 配合使用，以反转回归预测结果的缩放。

参数

样本

用于计算缩放参数的样本作为容器传输到 `ipSamples`。每个样本的特征值数必须相同。

缩放

计算缩放参数将通过引用 `ipScales` 作为一个容器返回。容器中元素的数量和顺序取决于缩放类型，始终存在于第一个元素中。如为 `MAXABS`，每个特征始终有一个缩放值。对于其他类型，每个特征始终有两个缩放值。

缩放类型

所需缩放类型将作为枚举 ETcVnFeatureScalingType [▶ 194] 传输到`eFeatureScalingType`。选项有：

- `TCVN_FST1_MAXABS`：使用特征绝对值的最大值从 -1 到 1 的范围内获得归一化。如果所有值都是正值，则从 0 到 1 进行归一化。
- `TCVN_FST1_MINMAX`：使用特征值的最小值和最大值从 0 到 1 的范围内获得归一化。
- `TCVN_FST1_STANDARDIZATION`：使用特征值的平均值和标准偏差将向量平均值归一化为 0，将向量标准偏差归一化为 1。

应用

例如，使用 `MINMAX` 方法计算缩放参数的过程如下所示：

```
hr := F_VN_GetFeatureScales(
    ipSamples           := ipSamples,
    ipScales            := ipScales,
    eFeatureScalingType := TCVN_FST1_MINMAX,
    hrPrev              := hr);
```

例如，如果 ipSamples 中每个样本的特征值数为 12，则ipScales 针对此缩放类型将正好有1 + 2 * 12 = 25 个元素。

相关函数

- F_VN_FeatureScaling [▶ 1408]
- F_VN_InverseFeatureScaling [▶ 1425]

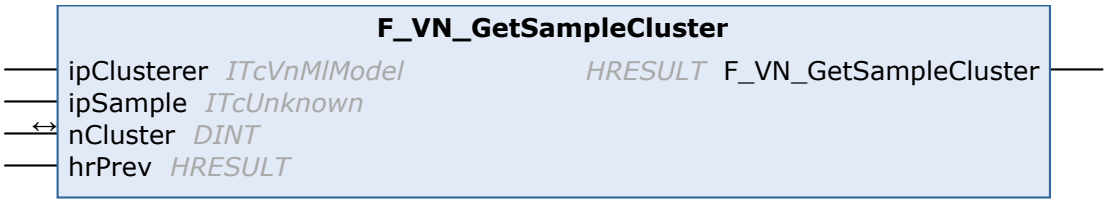
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.35 F_VN_GetSampleCluster



Get the ID of the best-matching cluster of a single sample.

语法

定义：

```
FUNCTION F_VN_GetSampleCluster : HRESULT
VAR_INPUT
    ipClusterer : ITcVnMlModel;
    ipSample    : ITcUnknown;
END_VAR
VAR_IN_OUT
    nCluster    : DINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipClusterer	ITcVnMlModel [▶ 417]	要使用的聚类器
ipSample	ITcUnknown [▶ 424]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nCluster	DINT	返回样本所属聚类的 ID

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_GetSampleCluster 可确定样本的最佳匹配聚类，并返回其 ID。

参数

聚类器模型

先前训练模型必须传输到 ipClusterer，以便分配至聚类。

样本

带有样本的容器被传输到 ipSample。容器类型必须是 ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL。

聚类 ID

确定的 ID 通过 nCluster 返回。

专家参数

专家变体 F_VN_GetSampleClusterExp [▶ 1422] 包含附加参数。

应用

样本到聚类的分配可如下计算：

```
hr := F_VN_GetSampleCluster(
    ipClusterer := ipClusterer,
    ipSample    := ipSample,
    nCluster    := nCluster,
    hrPrev      := hr);
```

相关函数

- F_VN_GetBatchClusters [▶ 1413]

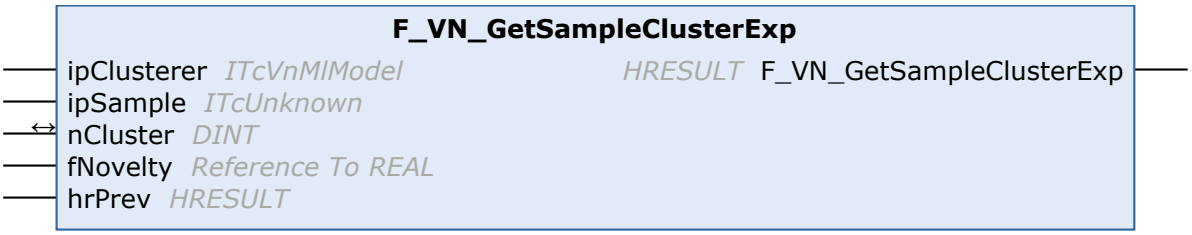
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.36 F_VN_GetSampleClusterExp



Get the ID of the best-matching cluster of a single sample. (expert function)

语法

定义：

```
FUNCTION F_VN_GetSampleClusterExp : HRESULT
VAR_INPUT
    ipClusterer : ITcVnMlModel;
    ipSample    : ITcUnknown;
END_VAR
VAR_IN_OUT
    nCluster    : DINT;
END_VAR
VAR_INPUT
    fNovelty    : Reference To REAL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipClusterer	ITcVnMlModel [▶ 417]	要使用的聚类器
ipSample	ITcUnknown [▶ 424]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
fNovelty	引用 REAL	返回所显示样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）（可选，如果不需要，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nCluster	DINT	返回样本所属聚类的 ID

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_GetSampleClusterExp` 是F_VN_GetSampleCluster [▶ 1421] 的专家变体。它包含额外的参数。

参数

聚类器模型

先前训练模型必须传输到 `ipClusterer`，以便分配至聚类。

样本

带有样本的容器被传输到 `ipSample`。容器类型必须是 `ContainerType_Vector_REAL` 或 `ContainerType_Vector_LREAL`。

聚类 ID

确定的 ID 通过 `nCluster` 返回。

新颖性

样本的新颖性通过引用 `fNovelty` 返回。新颖性表示样本与聚类的差异程度。新颖性为 0 表示模型已知样本。

应用

样本的聚类和新颖性可如下计算：

```
hr := F_VN_GetSampleClusterExp(
    ipClusterer := ipClusterer,
    ipSample    := ipSample,
    nCluster    := nCluster,
    fNovelty    := fNovelty,
    hrPrev      := hr);
```

相关函数

- F_VN_GetSampleNovelty [▶ 1424]
- F_VN_GetBatchClusters [▶ 1413]
- F_VN_GetBatchNovelties [▶ 1415]

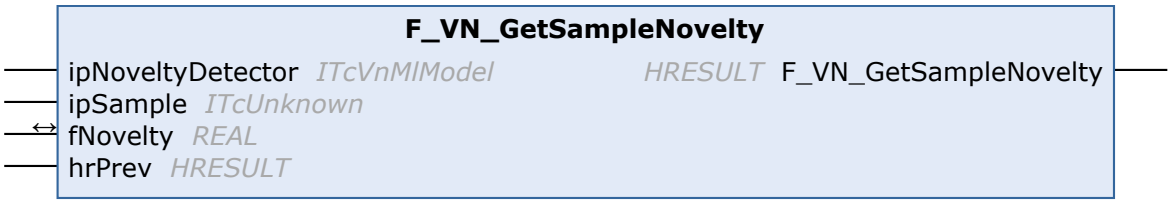
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.37 F_VN_GetSampleNovelty



Get the degree of novelty of a single sample.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_GetSampleNovelty : HRESULT
VAR_INPUT
    ipNoveltyDetector : ITcVnMlModel;
    ipSample          : ITcUnknown;
END_VAR
VAR_IN_OUT
    fNovelty          : REAL;
END_VAR
VAR_INPUT
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipNoveltyDetector	ITcVnMlModel [▶ 417]	要使用异常探测器
ipSample	ITcUnknown [▶ 424]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fNovelty	REAL	返回所显示样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_GetSampleNovelty 可确定容器中样本的新颖性。

参数

异常模型

要计算新颖性，必须将先前训练模型传输到 ipNoveltyDetector。

样本

带有样本的容器被传输到 ipSample

新颖性

新颖性通过引用 fNovelty 返回。新颖性表示样本与已学习数据的差异程度。新颖性为 0 表示模型已知样本。

应用

样本的新颖性可如下计算：

```
hr := F_VN_GetSampleNovelty(
    ipNoveltyDetector   := ipNoveltyDetector,
    ipSample            := ipSample,
    fNovelty            := fNovelty,
    hrPrev              := hr);
```

相关函数

- F_VN_GetBatchNovelties [▶ 1415]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.38 F_VN_InverseFeatureScaling



Apply the inverse of feature scaling to one or more sample(s).

语法

定义：

```
FUNCTION F_VN_InverseFeatureScaling : HRESULT
VAR_INPUT
    ipSamples        : ITcVnContainer;
    ipScales         : ITcVnContainer;
    ipInverseSamples : Reference To ITcVnContainer;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSamples	ITcVnContainer [▶ 365]	包含一个或多个输入样本的容器（ContainerType_Vector_REAL、ContainerType_Vector_LREAL、ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_LREAL）
ipScales	ITcVnContainer [▶ 365]	包含每个特征的缩放参数的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）。
ipInverseSamples	引用 ITcVnContainer [▶ 365]	使用 ipScales 的缩放类型和参数返回逆样本。它与 ipSamples 类型相同。如果使用同一个容器 ipSamples，源数据将被替换。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_InverseFeatureScaling` 反转一个或多个样本的缩放。函数 F_VN_FeatureScaling [▶ 1408] 可用于缩放。

参数

样本

必须将一个或多个样本作为一个容器传输到 `ipSamples`。

缩放

必须将带有缩放类型和相应缩放参数的容器传输到 `ipScales`。容器可以使用函数 F_VN_GetFeatureScales [▶ 1419] 计算，也可以使用其中描述的结构创建。

缩放样本

反缩放样本通过引用 `ipInverseSamples` 返回。

专家参数

专家变体 F_VN_InverseFeatureScalingExp [▶ 1428] 包含附加参数。

应用

例如，缩放比例之前使用 F_VN_FeatureScaling [▶ 1408] 计算的反特征缩放如下所示：

```
hr := F_VN_FeatureScaling(ipSamples, ipScales, ipSamplesScaled, hr);

hr := F_VN_InverseFeatureScaling(
    ipSamples        := ipSamplesScaled,
    ipScales         := ipScales,
    ipInverseSamples:= ipSamplesInvertedScaling,
    hrPrev           := hr);
```

相关函数

- F_VN_GetFeatureScales [▶ 1419]
- F_VN_FeatureScaling [▶ 1408]
- F_VN_InverseFeatureScaling_REAL [▶ 1427]

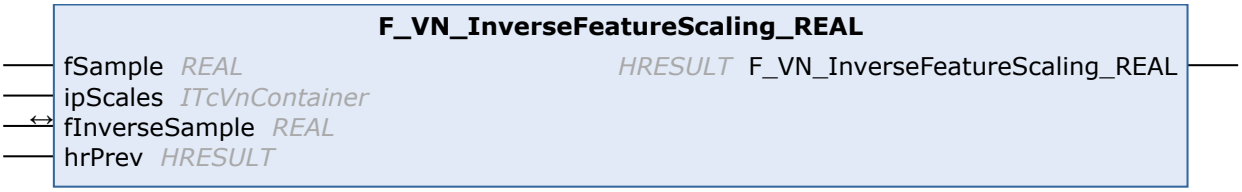
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.39 F_VN_InverseFeatureScaling_REAL



Apply the inverse of feature scaling to a single value.

语法

定义：

```
FUNCTION F_VN_InverseFeatureScaling_REAL : HRESULT
VAR_INPUT
    fSample         : REAL;
    ipScales        : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fInverseSample : REAL;
END_VAR
VAR_INPUT
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
fSample	REAL	样本值
ipScales	ITcVnContainer [▶ 365]	包含每个特征的缩放参数的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fInverseSample	REAL	返回逆样本。

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_InverseFeatureScaling_REAL 可逆缩放只有一个特征的样本或标量回归子的预测值。函数 F_VN_FeatureScaling [▶ 1408] 可用于缩放。

参数

样本

REAL 类型的样本必须传输到 fSample。

缩放

必须将带有缩放类型和相应缩放参数的容器传输到 ipScales。容器可以使用函数 F_VN_GetFeatureScales [▶ 1419] 计算，也可以使用其中描述的结构创建。

缩放样本

逆缩放特征通过 fInverseSample 返回。

专家参数

专家变体 F_VN_InverseFeatureScalingExp_REAL [▶ 1430] 包含附加参数。

应用

例如，一个样本的逆特征缩放如下所示：

```
hr := F_VN_InverseFeatureScaling_REAL(
    fSample        := fSample,
    ipScales       := ipScales,
    fInverseSample := fSampleInvertedScaling,
    hrPrev         := hr);
```

相关函数

- F_VN_GetFeatureScales [▶ 1419]
- F_VN_FeatureScaling [▶ 1408]
- F_VN_InverseFeatureScaling [▶ 1425]

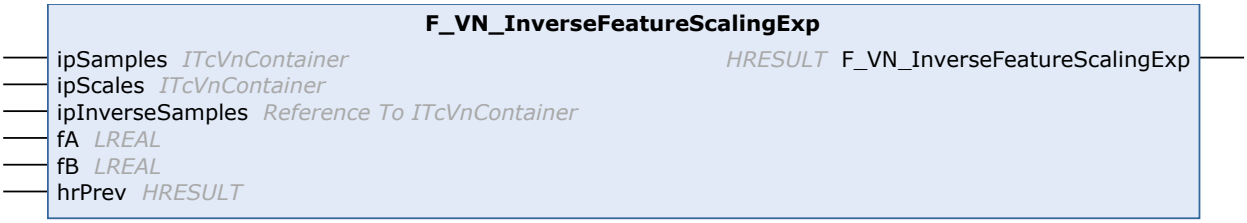
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.40 F_VN_InverseFeatureScalingExp



Apply the inverse of feature scaling to one or more sample(s). [fA, fB] should be used if the scaling has been performed using TCVN_FST1_MINMAX with a predetermined output range.. (expert function)

语法

定义：

```
FUNCTION F_VN_InverseFeatureScalingExp : HRESULT
VAR_INPUT
    ipSamples        : ITcVnContainer;
    ipScales         : ITcVnContainer;
    ipInverseSamples : Reference To ITcVnContainer;
    fA               : LREAL;
    fB               : LREAL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSamples	ITcVnContainer [▶ 365]	包含一个或多个输入样本的容器（ContainerType_Vector_REAL、ContainerType_Vector_LREAL、ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_LREAL）
ipScales	ITcVnContainer [▶ 365]	包含每个特征的缩放参数的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）。
ipInverseSamples	引用 ITcVnContainer [▶ 365]	使用 ipScales 的缩放类型和参数返回逆样本。它与 ipSamples 类型相同。如果使用同一个容器 ipSamples，源数据将被替换。
fA	LREAL	表示 TCVN_FST1_MINMAX 范围的下限
fB	LREAL	表示 TCVN_FST1_MINMAX 范围的上限
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_InverseFeatureScalingExp` 是F_VN_InverseFeatureScaling [▶ 1425] 的专家变体。它包含额外的参数。

参数

样本

必须将一个或多个样本作为一个容器传输到 `ipSamples`。

缩放

必须将带有缩放类型和相应缩放参数的容器传输到 `ipScales`。容器可以使用函数 F_VN_GetFeatureScales [▶ 1419] 计算，也可以使用其中描述的结构创建。

缩放样本

反缩放样本通过引用 `ipInverseSamples` 返回。

定义输出范围

如果使用缩放类型 `TCVN_FST1_MINMAX`，还可以通过 `fA` 和 `fB` 指定输出值的值范围。

应用

例如，先前由 F_VN_FeatureScalingExp [▶ 1410] 计算的逆特征缩放的输出值范围为 0.1 至 0.9，具体如下所示：

```
hr := F_VN_FeatureScalingExp(ipSamples, ipScales, ipSamplesScaled, 0.1, 0.9, hr);

hr := F_VN_InverseFeatureScalingExp(
    ipSamples       := ipSamplesScaled,
    ipScales        := ipScales,
    ipInverseSamples:= ipSamplesInvertedScaling,
    fA              := 0.1,
    fB              := 0.9,
    hrPrev          := hr);
```

相关函数

- F_VN_GetFeatureScales [▶ 1419]
- F_VN_FeatureScaling [▶ 1408]
- F_VN_InverseFeatureScaling_REAL [▶ 1427]

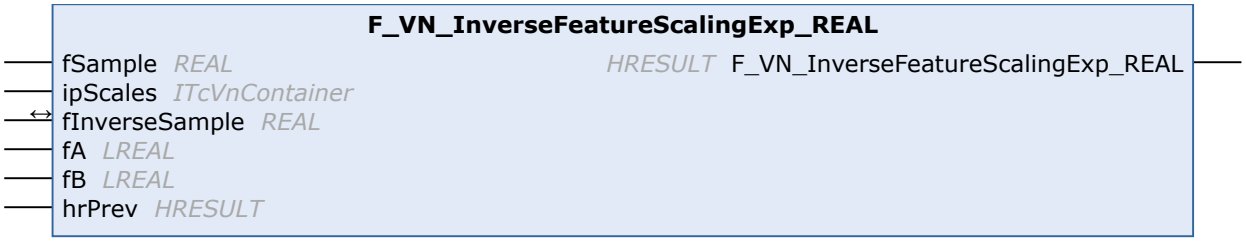
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.41 F_VN_InverseFeatureScalingExp_REAL



Apply the inverse of feature scaling to a single value. [fA, fB] should be used if the scaling has been performed using TCVN_FST1_MINMAX with a predetermined output range.

语法

定义：

```
FUNCTION F_VN_InverseFeatureScalingExp_REAL : HRESULT
VAR_INPUT
    fSample         : REAL;
    ipScales        : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fInverseSample : REAL;
END_VAR
VAR_INPUT
    fA              : LREAL;
    fB              : LREAL;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
fSample	REAL	样本值
ipScales	ITcVnContainer [▶ 365]	包含每个特征的缩放参数的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）。
fA	LREAL	表示 TCVN_FST1_MINMAX 范围的下限
fB	LREAL	表示 TCVN_FST1_MINMAX 范围的上限

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fInverseSample	REAL	返回逆样本。

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_InverseFeatureScalingExp_REAL` 是F_VN_InverseFeatureScaling_REAL [▶ 1427] 的专家变体。它包含额外的参数。

参数

样本

`REAL` 类型的样本必须传输到 `fSample`。

缩放

必须将带有缩放类型和相应缩放参数的容器传输到 `ipScales`。容器可以使用函数 F_VN_GetFeatureScales [▶ 1419] 计算，也可以使用其中描述的结构创建。

缩放样本

逆缩放特征通过 `fInverseSample` 返回。

定义输出范围

如果使用缩放类型 `TCVN_FST1_MINMAX`，还可以通过 `fA` 和 `fB` 指定输出值的值范围。

应用

例如，一个样本的逆特征缩放如下所示：

```
hr := F_VN_InverseFeatureScalingExp_REAL(
    fSample        := fSample,
    ipScales       := ipScales,
    fInverseSample := fSampleInvertedScaling,
    fA             := 0.1,
    fB             := 0.9,
    hrPrev         := hr);
```

相关函数

- F_VN_GetFeatureScales [▶ 1419]
- F_VN_FeatureScaling [▶ 1408]
- F_VN_InverseFeatureScaling_REAL [▶ 1427]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.42 F_VN_InverseFeatureTransform



Apply an inverse feature transform to one or more sample(s).

语法

定义：

```
FUNCTION F_VN_InverseFeatureTransform : HRESULT
VAR_INPUT
    ipFeatureTransform : ITcVnMlModel;
    ipSamples          : ITcUnknown;
    pipTransformed     : Pointer To ITcUnknown;
    hrPrev             : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipFeatureTransform	ITcVnMlModel [▶ 417]	要使用的特征变换实例
ipSamples	ITcUnknown [▶ 424]	包含一个或多个输入样本的容器（ContainerType_Vector_REAL、ContainerType_Vector_LREAL、ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_LREAL）
pipTransformed	指针指向 ITcUnknown [▶ 424]	使用与 ipSamples 相同的类型返回变换后的样本
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_InverseFeatureTransform` 利用转换模型逆转换一个或多个样本的特征。使用函数 F_VN_FeatureTransform [▶ 1411] 可以对同一模型进行直接转换。

参数

转换模型

先前创建模型必须传输到 `ipFeatureTransform` 进行特征转换。

样本

具有待转换样本的容器会被传输到 `ipSamples`。允许使用以下容器类型：

- `ContainerType_Vector_REAL` 或 `ContainerType_Vector_LREAL`，用于单个样本
- `ContainerType_Vector_Vector_REAL` 或 `ContainerType_Vector_Vector_LREAL`，用于具有多个样本的容器。

转换样本

通过指针 `pipTransformed` 返回具有转换样本的容器。容器类型取自 ipSamples。

应用

例如，逆特征转换如下所示进行：

```
hr := F_VN_InverseFeatureTransform(
    ipFeatureTransform  := ipMlModel,
    ipSamples           := ipSamplesTransformed,
    pipTransformed      := ADR(ipTransformed),
    hrPrev              := hr);
```

相关函数

- F_VN_FeatureTransform [▶ 1411]

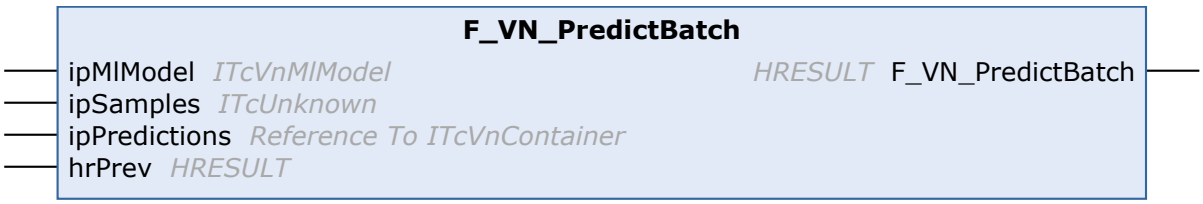
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.43 F_VN_PredictBatch



Compute predictions for a batch of samples.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_PredictBatch : HRESULT
VAR_INPUT
    ipMlModel     : ITcVnMlModel;
    ipSamples     : ITcUnknown;
    ipPredictions : Reference To ITcVnContainer;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	ITcVnMlModel [▶ 417]	要使用的分类器或回归子
ipSamples	ITcUnknown [▶ 424]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）

名称	类型	描述
ipPredictions	引用 ITcVnContainer [▶ 365]	返回预测输出（取决于 ipSamples；类标签（用于分类为 ContainerType_Vector_DINT）或实值预测（用于标量输出回归为 ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL；用于矢量输出回归为 ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL））
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_PredictBatch 针对多个样本计算模型预测。根据模型类型的不同，预测的数据类型也不同（请参见 ipPredictions）。

参数

模型

要进行预测，必须将先前训练模型传输到 ipMlModel。

样本

样本在一个容器中被传输到 ipSamples。容器类型必须是 ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL。

预测

预测在一个容器中通过引用 ipPredictions 返回。根据模型的不同，容器的类型也不同：

- 分类的类别标签：ContainerType_Vector_DINT
- 标量回归中的数值预测值：ContainerType_Vector_REAL 或ContainerType_Vector_LREAL,，具体取决于 ipSamples
- 向量回归中的数值预测值：ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL,，具体取决于 ipSamples

专家参数

专家变体 F_VN_PredictBatchExp [▶ 1435] 包含附加参数。

应用

例如，可以同时对多个样本进行分类：

```
hr := F_VN_PredictBatch(
    ipMlModel       :=  ipMlModel,
    ipSamples       :=  ipSamples,
    ipPredictions   :=  ipPredictions,
    hrPrev          :=  hr);

hr := F_VN_GetAt_DINT(ipPredictions, nClassOfThirdSample, 2, hr);
```

F_VN_GetAt_DINT 的调用可显示如何从预测中获得第 3 个样本的类别标签。

相关函数

- F_VN_PredictSampleClass [▶ 1437]
- F_VN_PredictSampleScalar [▶ 1440]
- F_VN_PredictSampleVector [▶ 1443]

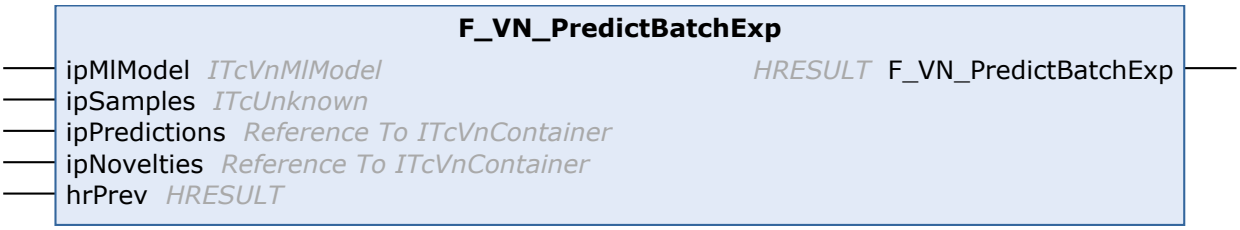
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.44 F_VN_PredictBatchExp



Compute predictions for a batch of samples. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_PredictBatchExp : HRESULT
VAR_INPUT
    ipMlModel     : ITcVnMlModel;
    ipSamples     : ITcUnknown;
    ipPredictions : Reference To ITcVnContainer;
    ipNovelties   : Reference To ITcVnContainer;
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	ITcVnMlModel [▶ 417]	要使用的分类器或回归子
ipSamples	ITcUnknown [▶ 424]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
ipPredictions	引用 ITcVnContainer [▶ 365]	返回预测输出（取决于 ipSamples；类标签（用于分类为 ContainerType_Vector_DINT）或实值预测（用于标量输出回归为 ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL；用于矢量输出回归为 ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL））
ipNovelties	引用 ITcVnContainer [▶ 365]	返回每个样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）（ContainerType_Vector_REAL；可选，如果不需要，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_PredictBatchExp 是F_VN_PredictBatch [▶ 1433] 的专家变体。它包含额外的参数。

参数

模型

要进行预测，必须将先前训练模型传输到 ipMlModel。

样本

样本在一个容器中被传输到 ipSamples。容器类型必须是 ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL。

预测

预测在一个容器中通过引用 ipPredictions 返回。根据模型的不同，容器的类型也不同：

- 分类的类别标签：ContainerType_Vector_DINT
- 标量回归中的数值预测值：ContainerType_Vector_REAL 或ContainerType_Vector_LREAL,，具体取决于 ipSamples
- 向量回归中的数值预测值：ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL,，具体取决于 ipSamples

新颖性

样本的新颖性通过引用 ipNovelties 作为容器返回。

应用

例如，可以同时对多个样本进行分类：

```
hr := F_VN_PredictBatchExp(
    ipMlModel       := ipMlModel,
    ipSamples       := ipSamples,
    ipPredictions   := ipPredictions,
    ipNovelties     := ipNovelties,
    hrPrev          := hr);

hr := F_VN_GetAt_DINT(ipPredictions, nClassOfThirdSample, 2, hr);
```

F_VN_GetAt_DINT 的调用可显示如何从预测中获得第 3 个样本的类别标签。

相关函数

- F_VN_PredictSampleClass [▶ 1437]
- F_VN_PredictSampleScalar [▶ 1440]
- F_VN_PredictSampleVector [▶ 1443]
- F_VN_GetBatchNovelties [▶ 1424]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.45 F_VN_PredictSampleClass



Classify a single sample.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_PredictSampleClass : HRESULT
VAR_INPUT
    ipClassifier : ITcVnMlModel;
    ipSample     : ITcUnknown;
END_VAR
VAR_IN_OUT
    nClass       : DINT;
END_VAR
VAR_INPUT
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipClassifier	ITcVnMlModel [▶ 417]	要使用的分类器
ipSample	ITcUnknown [▶ 424]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nClass	DINT	返回分类结果

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_PredictSampleClass` 根据分类模型对样本进行分类。

参数

分类模型

先前训练分类模型必须传输到 `ipClassifier`。

样本

样本在一个容器中被传输到 `ipSample`。容器类型必须是 `ContainerType_Vector_REAL` 或 `ContainerType_Vector_LREAL`。

类别

样本的类别作为分类结果通过 nClass 返回。

专家参数

专家变体 F_VN_PredictSampleClassExp [▶ 1438] 包含附加参数。

应用

例如，一个样本可如下分类：

```
hr := F_VN_PredictSampleClass(
    ipClassifier    := ipMlModel,
    ipSample        := ipSample,
    nClass          := nClassResult,
    hrPrev          := hr);
```

相关函数

- F_VN_PredictBatch [▶ 1433]

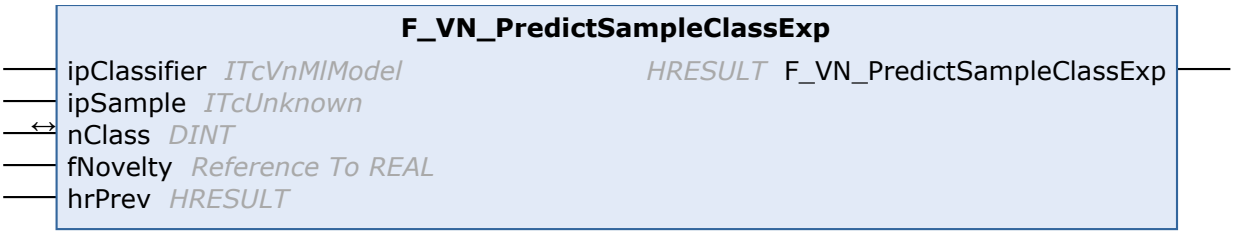
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.46 F_VN_PredictSampleClassExp



Classify a single sample. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_PredictSampleClassExp : HRESULT
VAR_INPUT
    ipClassifier : ITcVnMlModel;
    ipSample     : ITcUnknown;
END_VAR
VAR_IN_OUT
    nClass       : DINT;
END_VAR
VAR_INPUT
    fNovelty     : Reference To REAL;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipClassifier	ITcVnMlModel [▶ 417]	要使用的分类器
ipSample	ITcUnknown [▶ 424]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）

名称	类型	描述
fNovelty	引用 REAL	返回所显示样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）（可选，如果不需要，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nClass	DINT	返回分类结果

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_PredictSampleClassExp` 是F_VN_PredictSampleClass [▶ 1437] 的专家变体。它包含额外的参数。

参数

分类模型

先前训练分类模型必须传输到 `ipClassifier`。

样本

样本在一个容器中被传输到 `ipSample`。容器类型必须是 `ContainerType_Vector_REAL` 或 `ContainerType_Vector_LREAL`。

类别

样本的类别作为分类结果通过 `nClass` 返回。

新颖性

样本的新颖性通过 `fNovelty` 返回。

应用

例如，一个样本可如下分类：

```
hr := F_VN_PredictSampleClassExp(
    ipClassifier    := ipMlModel,
    ipSample        := ipSample,
    nClass          := nClassResult,
    fNovelty        := fNovelty,
    hrPrev          := hr);
```

相关函数

- F_VN_GetSampleNovelty [▶ 1424]
- F_VN_PredictBatch [▶ 1433]

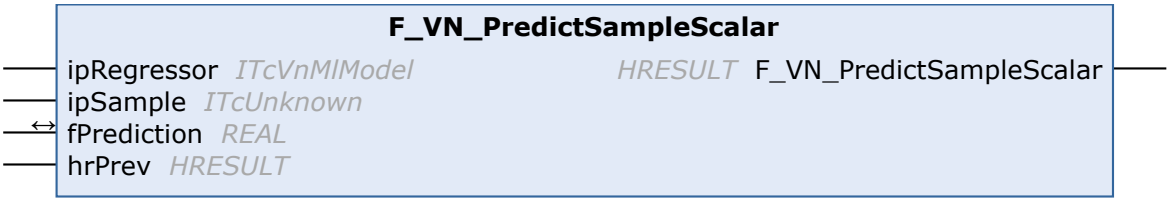
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.47 F_VN_PredictSampleScalar



Compute a scalar prediction for a single sample.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_PredictSampleScalar : HRESULT
VAR_INPUT
    ipRegressor : ITcVnMlModel;
    ipSample    : ITcUnknown;
END_VAR
VAR_IN_OUT
    fPrediction : REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipRegressor	ITcVnMlModel [▶ 417]	要使用的回归子
ipSample	ITcUnknown [▶ 424]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fPrediction	REAL	返回预测的输出

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_PredictSampleScalar` 根据回归模型计算样本的标量预测值。

参数

回归模型

先前培训回归模型必须传输到 `ipRegressor`。

样本

样本容器作为 ipSample 传输。容器类型必须是 ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL。

预测

计算预测值通过 fPrediction 返回。

专家参数

专家变体 F_VN_PredictSampleScalarExp [▶ 1441] 包含附加参数。

应用

例如，一个样本的预测值可如下计算：

```
hr := F_VN_PredictSampleScalar(
    ipRegressor := ipRegressor,
    ipSample    := ipSample,
    fPrediction := fPrediction,
    hrPrev      := hr);
```

相关函数

- F_VN_PredictBatch [▶ 1433]

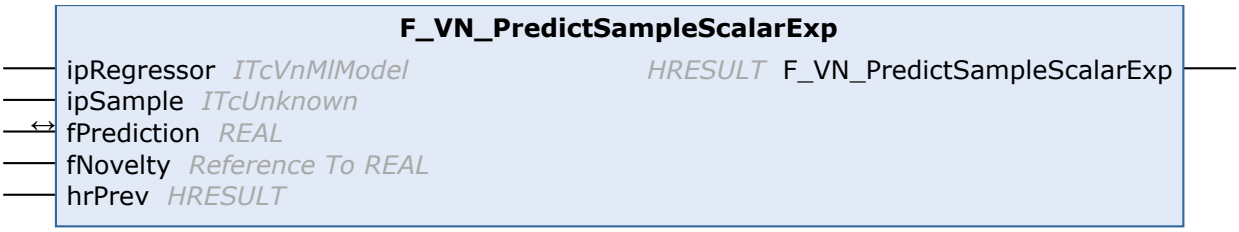
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.48 F_VN_PredictSampleScalarExp



Compute a scalar prediction for a single sample. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_PredictSampleScalarExp : HRESULT
VAR_INPUT
    ipRegressor : ITcVnMlModel;
    ipSample    : ITcUnknown;
END_VAR
VAR_IN_OUT
    fPrediction : REAL;
END_VAR
VAR_INPUT
    fNovelty    : Reference To REAL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipRegressor	ITcVnMlModel [▶ 417]	要使用的回归子
ipSample	ITcUnknown [▶ 424]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
fNovelty	引用 REAL	返回所显示样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）（可选，如果不需要，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fPrediction	REAL	返回预测的输出

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_PredictSampleScalarExp` 是 F_VN_PredictSampleScalar [▶ 1440] 的专家变体。它包含额外的参数。

参数

回归模型

先前培训回归模型必须传输到 `ipRegressor`。

样本

样本容器作为 `ipSample` 传输。容器类型必须是 `ContainerType_Vector_REAL` 或 `ContainerType_Vector_LREAL`。

预测

计算预测值通过 `fPrediction` 返回。

新颖性

样本的新颖性通过 `fNovelty` 返回。

应用

例如，一个样本的回归值可如下计算：

```
hr := F_VN_PredictSampleScalarExp(
    ipRegressor := ipRegressor,
    ipSample    := ipSample,
    fPrediction := fPrediction,
    fNovelty    := fNovelty,
    hrPrev      := hr);
```

相关函数

- F_VN_PredictBatch [▶ 1433]
- F_VN_GetBatchNovelties [▶ 1424]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.49 F_VN_PredictSampleVector



Compute a vectorial prediction for a single sample.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_PredictSampleVector : HRESULT
VAR_INPUT
    ipRegressor  : ITcVnMlModel;
    ipSample     : ITcUnknown;
    ipPrediction : Reference To ITcVnContainer;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipRegressor	ITcVnMlModel [▶ 417]	要使用的回归子
ipSample	ITcUnknown [▶ 424]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
ipPrediction	引用 ITcVnContainer [▶ 365]	返回预测输出（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL，取决于 ipSample）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_PredictSampleVector` 根据回归模型计算样本的预测向量。

参数

回归模型

先前培训回归模型必须传输到 `ipRegressor`。

样本

样本容器作为 ipSample 传输。容器类型必须是 ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL。

预测

计算出的预测向量作为一个容器通过引用 ipPrediction 返回。容器类型取自 ipSamples。

专家参数

专家变体 F_VN_PredictSampleVectorExp [▶ 1444] 包含附加参数。

应用

例如，一个样本的预测向量可如下计算：

```
hr := F_VN_PredictSampleVector(
    ipRegressor := ipRegressor,
    ipSample    := ipSample,
    ipPrediction:= ipPrediction,
    hrPrev      := hr);
```

相关函数

- F_VN_PredictBatch [▶ 1433]

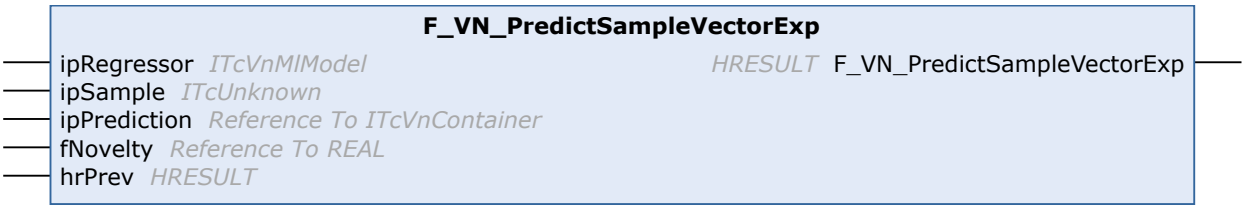
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.50 F_VN_PredictSampleVectorExp



Compute a vectorial prediction for a single sample. (expert function)
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_PredictSampleVectorExp : HRESULT
VAR_INPUT
    ipRegressor  : ITcVnMlModel;
    ipSample     : ITcUnknown;
    ipPrediction : Reference To ITcVnContainer;
    fNovelty     : Reference To REAL;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipRegressor	ITcVnMlModel [▶ 417]	要使用的回归子

名称	类型	描述
ipSample	ITcUnknown [▶ 424]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
ipPrediction	引用 ITcVnContainer [▶ 365]	返回预测输出（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL，取决于 ipSample）
fNovelty	引用 REAL	返回所显示样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）（可选，如果不需要，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_PredictSampleVectorExp` 是F_VN_PredictSampleVector [▶ 1443] 的专家变体。它包含额外的参数。

参数

回归模型

先前培训回归模型必须传输到 `ipRegressor`。

样本

样本容器作为 `ipSample` 传输。容器类型必须是 `ContainerType_Vector_REAL` 或 `ContainerType_Vector_LREAL`。

预测

计算出的预测向量作为一个容器通过引用 `ipPrediction` 返回。容器类型取自 `ipSamples`。

新颖性

样本的新颖性通过 `fNovelty` 返回。

应用

例如，一个样本的预测向量可如下计算：

```
hr := F_VN_PredictSampleVectorExp(
    ipRegressor := ipRegressor,
    ipSample    := ipSample,
    ipPrediction:= ipPrediction,
    fNovelty    := fNovelty,
    hrPrev      := hr);
```

相关函数

- F_VN_PredictBatch [▶ 1433]
- F_VN_GetBatchNovelties [▶ 1424]

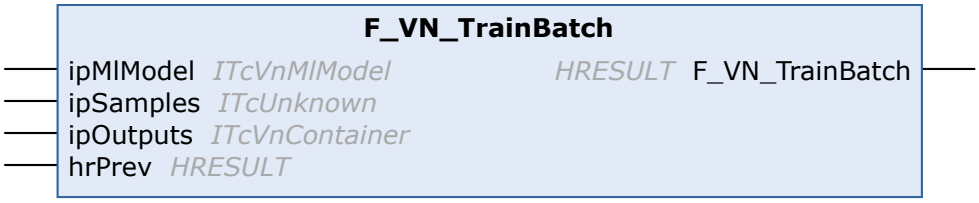
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.51 F_VN_TrainBatch



Train a classifier, a clusterer, a novelty detector, or a regressor with a batch of samples. On-line trainable models are trained once with each sample. Depending on the application and the number of available training samples, repeated training of such models with the same data may improve the results.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_TrainBatch : HRESULT
VAR_INPUT
    ipMlModel : ITcVnMlModel;
    ipSamples : ITcUnknown;
    ipOutputs : ITcVnContainer;
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	ITcVnMlModel [▶ 417]	要使用的分类器、聚类器、异常检测器或回归子
ipSamples	ITcUnknown [▶ 424]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
ipOutputs	ITcVnContainer [▶ 365]	要学习的类标签（用于分类为 ContainerType_Vector_DINT）或目标输出（用于标量输出回归为 ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL；用于向量输出回归为 ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）（对于聚类器和异常检测器，必须为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_TrainBatch` 使用多个样本培训模型。根据模型类型，结果值必须具有不同的类型（请参见 `ipOutputs`）。分类器、聚类器、新颖性检测器和回归子作为模型类型获得支持。

参数

模型

先前创建模型必须传输到 `ipMlModel`。

样本

样本在一个容器中被传输到 `ipSamples`。容器类型必须是 `ContainerType_Vector_Vector_REAL` 或 `ContainerType_Vector_Vector_LREAL`。

结果

样本的已知结果在一个容器中被传输到 `ipOutputs`。根据模型的类型，容器必须有不同的类型：

- 分类的类别标签：`ContainerType_Vector_DINT`
- 标量回归的数值结果值：`ContainerType_Vector_REAL` 或 `ContainerType_Vector_LREAL`
- 向量回归的数值结果值：`ContainerType_Vector_Vector_REAL` 或 `ContainerType_Vector_Vector_LREAL`

如果是聚类器模型或新颖性检测器，则必须是 `ipOutputs := 0`。

应用

例如，用于分类的 ML 模型可如下训练：

```
hr := F_VN_TrainBatch(
    ipMlModel   := ipMlModel,
    ipSamples   := ipSamples,
    ipOutputs   := ipClassLabels,
    hrPrev      := hr);
```

相关函数

- F_VN_TrainBatchClusters [▶ 1447]
- F_VN_TrainSample [▶ 1449]
- F_VN_TrainSampleClass [▶ 1450]
- F_VN_TrainSampleCluster [▶ 1451]
- F_VN_TrainSampleScalar [▶ 1453]
- F_VN_TrainSampleVector [▶ 1454]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.52 F_VN_TrainBatchClusters



Train a clusterer with a batch of samples and return the IDs of the clusters the samples have been assigned to, if requested. On-line trainable clusterers are trained once with each sample. Depending on the application and the number of available training samples, repeated training of such models with the same data may improve the results.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_TrainBatchClusters : HRESULT
VAR_INPUT
    ipClusterer : ITcVnMlModel;
    ipSamples   : ITcUnknown;
```

```
    ipClusters  : Reference To ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipClusterer	ITcVnMlModel [▶ 417]	要使用的聚类器
ipSamples	ITcUnknown [▶ 424]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
ipClusters	引用 ITcVnContainer [▶ 365]	返回样本所属聚类的 ID（ContainerType_Vector_DINT；可选，如果不需要，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_TrainBatchClusters` 使用多个样本训练聚类器模型。与 F_VN_TrainBatch [▶ 1446] 相比，还可以为每个样本返回分配的聚类。

参数

聚类器模型

先前创建聚类器模型必须传输到 `ipClusterer`。

样本

样本在一个容器中被传输到 `ipSamples`。容器类型必须是 `ContainerType_Vector_Vector_REAL` 或 `ContainerType_Vector_Vector_LREAL`。

聚类

分配聚类的 ID 通过引用 `ipClusters` 在一个 `ContainerType_Vector_DINT` 类型的容器中返回。

应用

例如，聚类器模型可如下训练：

```
hr := F_VN_TrainBatchClusters(
    ipClusterer := ipClusterModel,
    ipSamples   := ipSamples,
    ipClusters  := ipClusters,
    hrPrev      := hr);

hr := F_VN_GetAt_DINT(ipClusters, nClusterOfThirdSample, 2, hr);
```

`F_VN_GetAt_DINT` 的调用可显示如何在训练后直接查询第 3 个样本的聚类分配。

相关函数

- F_VN_TrainBatch [▶ 1446]
- F_VN_TrainSample [▶ 1449]
- F_VN_TrainSampleClass [▶ 1450]
- F_VN_TrainSampleCluster [▶ 1451]
- F_VN_TrainSampleScalar [▶ 1453]
- F_VN_TrainSampleVector [▶ 1454]

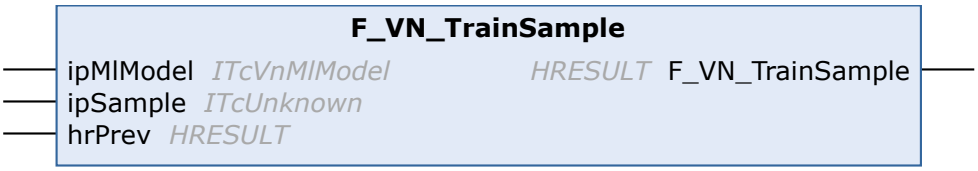
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.53 F_VN_TrainSample



Train a clusterer or a novelty detector with a single sample.

语法

定义：

```
FUNCTION F_VN_TrainSample : HRESULT
VAR_INPUT
    ipMlModel : ITcVnMlModel;
    ipSample  : ITcUnknown;
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipMlModel	ITcVnMlModel [▶ 417]	要使用聚类器或异常检测器
ipSample	ITcUnknown [▶ 424]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_TrainSample` 使用样本训练聚类或新颖性检测模型。

参数

模型

先前创建模型必须传输到 `ipMlModel`。

样本

样本容器传输到 `ipSample`。容器类型必须是 `ContainerType_Vector_REAL` 或 `ContainerType_Vector_LREAL`。

应用

例如，可以如下使用样本训练ML 模型：

```
hr := F_VN_TrainSample(
    ipMlModel   := ipMlModel,
    ipSample    := ipSample,
    hrPrev      := hr);
```

相关函数

- F_VN_TrainBatch [▶ 1446]
- F_VN_TrainBatchClusters [▶ 1447]
- F_VN_TrainSampleClass [▶ 1450]
- F_VN_TrainSampleCluster [▶ 1451]
- F_VN_TrainSampleScalar [▶ 1453]
- F_VN_TrainSampleVector [▶ 1454]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.54 F_VN_TrainSampleClass



Train a classifier with a single sample.

语法

定义：

```
FUNCTION F_VN_TrainSampleClass : HRESULT
VAR_INPUT
    ipClassifier : ITcVnMlModel;
    ipSample     : ITcUnknown;
    nClass       : DINT;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipClassifier	ITcVnMlModel [▶ 417]	要使用的分类器
ipSample	ITcUnknown [▶ 424]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
nClass	DINT	要学习的类标签
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 F_VN_TrainSampleClass 使用样本训练分类器模型。

参数

分类器模型

先前创建分类器模型必须传输到 ipClassifier。

样本

样本容器传输到 ipSample。容器类型必须是 ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL。

类别

样本的目标类别作为 nClass 转移。

应用

例如，分类器模型可以使用单个样本如下进行训练：

```
hr := F_VN_TrainSampleClass(
    ipClassifier    := ipMlModel,
    ipSample        := ipSample,
    nClass          := nClass,
    hrPrev          := hr);
```

相关函数

- F_VN_TrainBatch [▶ 1446]
- F_VN_TrainBatchClusters [▶ 1447]
- F_VN_TrainSample [▶ 1449]
- F_VN_TrainSampleCluster [▶ 1451]
- F_VN_TrainSampleScalar [▶ 1453]
- F_VN_TrainSampleVector [▶ 1454]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.55 F_VN_TrainSampleCluster



Train a clusterer with a single sample and return the ID of the cluster the sample has been assigned to, if requested. The clusterer must be online trainable.

语法

定义：

```
FUNCTION F_VN_TrainSampleCluster : HRESULT
VAR_INPUT
    ipClusterer : ITcVnMlModel;
    ipSample    : ITcUnknown;
    nCluster    : Reference To DINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipClusterer	ITcVnMlModel [▶ 417]	要使用的聚类器
ipSample	ITcUnknown [▶ 424]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
nCluster	引用 DINT	返回样本所属聚类的 ID（如果指针为 0，则不需要聚类 ID。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_TrainSampleCluster` 使用样本训练聚类器模型，并返回样本的对应聚类 ID。

参数

聚类器模型

先前创建聚类器模型必须传输到 `ipClusterer`。

样本

样本容器作为 `ipSample` 传输。容器类型必须是 `ContainerType_Vector_REAL` 或 `ContainerType_Vector_LREAL`。

聚类

与样本相关的聚类 ID 通过 `nCluster` 返回。

应用

例如，一个聚类器模型可以使用单个样本如下进行训练：

```
hr := F_VN_TrainSampleCluster(
    ipClusterer     := ipClusterModel,
    ipSample        := ipSample,
    nCluster        := nCluster,
    hrPrev          := hr);
```

相关函数

- F_VN_TrainBatch [▶ 1446]
- F_VN_TrainBatchClusters [▶ 1447]
- F_VN_TrainSample [▶ 1449]
- F_VN_TrainSampleClass [▶ 1450]

- F_VN_TrainSampleScalar [▶ 1453]
- F_VN_TrainSampleVector [▶ 1454]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.56 F_VN_TrainSampleScalar



Train a regressor with a single sample and scalar output.

语法

定义：

```
FUNCTION F_VN_TrainSampleScalar : HRESULT
VAR_INPUT
    ipRegressor : ITcVnMlModel;
    ipSample    : ITcUnknown;
    fOutput     : REAL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipRegressor	ITcVnMlModel [▶ 417]	要使用的回归子
ipSample	ITcUnknown [▶ 424]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
fOutput	REAL	要学习的标量输出
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_TrainSampleScalar` 使用样本训练标量回归子模型。

参数

回归子模型

先前创建回归子模型必须传输到 `ipRegressor`。

样本

样本容器作为 ipSample 传输。容器类型必须是 ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL。

训练值

样本的结果值被传输到 fOutput。

应用

例如，一个标量回归子模型可以使用单个样本如下进行训练：

```
hr := F_VN_TrainSampleScalar(
    ipRegressor     := ipRegressorModel,
    ipSample        := ipSample,
    fOutput         := fOutput,
    hrPrev          := hr);
```

相关函数

- F_VN_TrainBatch [▶ 1446]
- F_VN_TrainBatchClusters [▶ 1447]
- F_VN_TrainSample [▶ 1449]
- F_VN_TrainSampleClass [▶ 1450]
- F_VN_TrainSampleCluster [▶ 1451]
- F_VN_TrainSampleVector [▶ 1454]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.57 F_VN_TrainSampleVector



Train a regressor with a single sample and vectorial output.

语法

定义：

```
FUNCTION F_VN_TrainSampleVector : HRESULT
VAR_INPUT
    ipRegressor : ITcVnMlModel;
    ipSample    : ITcUnknown;
    ipOutput    : ITcVnContainer;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipRegressor	ITcVnMlModel [▶ 417]	要使用的回归子

名称	类型	描述
ipSample	ITcUnknown [▶ 424]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
ipOutput	ITcVnContainer [▶ 365]	要学习的矢量输出（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_TrainSampleVector` 使用样本训练向量回归子模型。

参数

回归子模型

先前创建回归子模型必须传输到 `ipRegressor`。

样本

样本容器作为 `ipSample` 传输。容器类型必须是 `ContainerType_Vector_REAL` 或 `ContainerType_Vector_LREAL`。

训练向量

样本的结果向量被传输到 `ipOutput`。

应用

例如，向量回归子模型可以使用单个样本如下进行训练：

```
hr := F_VN_TrainSampleVector(
    ipRegressor     := ipRegressorModel,
    ipSample        := ipSample,
    ipOutput        := ipVector,
    hrPrev          := hr);
```

相关函数

- F_VN_TrainBatch [▶ 1446]
- F_VN_TrainBatchClusters [▶ 1447]
- F_VN_TrainSample [▶ 1449]
- F_VN_TrainSampleClass [▶ 1450]
- F_VN_TrainSampleCluster [▶ 1451]
- F_VN_TrainSampleScalar [▶ 1453]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.20.58 F_VN_GetTransformParameters



Get the learned parameters of a feature transform model (e.g., PCA or LDA). Returns the component scores, the projection matrix, and optionally the mean vector. For PCA, the scores are eigenvalues representing the variance captured by each component. For LDA, the scores are eigenvalues representing the discriminative power of each component. Components are ordered by decreasing importance.

Syntax

Definition:

```
FUNCTION F_VN_GetTransformParameters : HRESULT
VAR_INPUT
    ipModel             : ITcVnMlModel;
    ipScores            : Reference To ITcVnContainer;
    ipProjectionVectors : Reference To ITcVnContainer;
    ipMean              : Reference To ITcVnContainer;
    hrPrev              : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipModel	ITcVnMlModel [▶ 417]	Trained feature transform model (e.g., PCA or LDA).
ipScores	Reference To ITcVnContainer [▶ 365]	Container holding the component scores, sorted in descending order of importance. For PCA and LDA, these represent the eigenvalues. The container type depends on the model (ContainerType_Vector_REAL for PCA, ContainerType_Vector_LREAL for LDA).
ipProjectionVectors	Reference To ITcVnContainer [▶ 365]	Container holding the projection vectors, where each inner vector represents one component direction, ordered consistently with the scores. For PCA and LDA, these represent the eigenvectors. The container type depends on the model (ContainerType_Vector_Vector_REAL for PCA, ContainerType_Vector_Vector_LREAL for LDA).
ipMean	Reference To ITcVnContainer [▶ 365]	Container holding the mean vector used for centering. The container type depends on the model (ContainerType_Vector_REAL for PCA). For LDA, the mean is not applicable and the container will be set to 0. Optional, set to 0 if not required.
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

Return value

HRESULT [▶ 135]

Required License

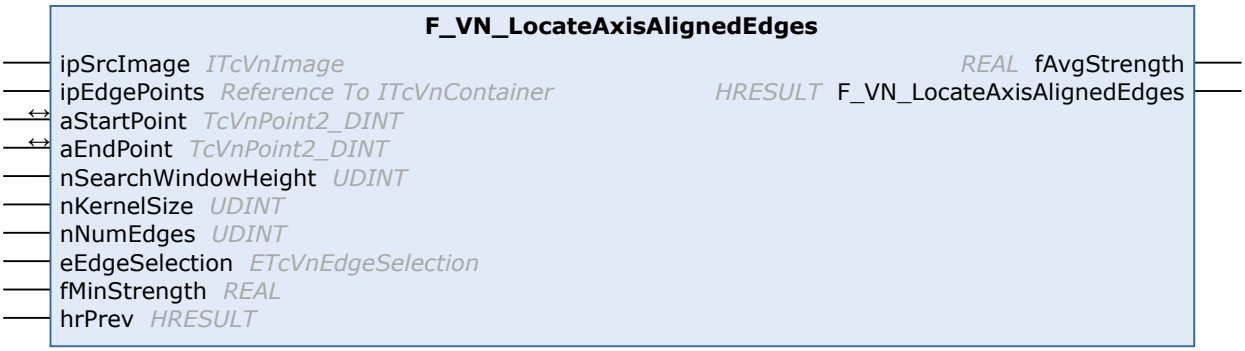
TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.21 测量

6.1.4.21.1 F_VN_LocateAxisAlignedEdges



Locate the position of multiple occurring edges inside a specified search window. The edges inside the search window must be straight and aligned to one of the image axes. They are located by interpolating the peaks of the first Gaussian derivative of the pixel intensities.

Syntax

Definition:

```
FUNCTION F_VN_LocateAxisAlignedEdges : HRESULT
VAR_INPUT
    ipSrcImage          : ITcVnImage;
    ipEdgePoints        : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aStartPoint         : TcVnPoint2_DINT;
    aEndPoint           : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    nSearchWindowHeight : UDINT;
    nKernelSize         : UDINT;
    nNumEdges           : UDINT;
    eEdgeSelection      : ETcVnEdgeSelection;
    fMinStrength        : REAL;
    hrPrev              : HRESULT;
END_VAR
VAR_OUTPUT
    fAvgStrength        : REAL;
END_VAR
```

Inputs

Name	Type	Description
ipSrcImage	ITcVnImage [▶ 407]	Source image (1 channel of type ET_USINT, ET_UINT, ET_INT or ET_REAL)
ipEdgePoints	Reference To ITcVnContainer [▶ 365]	Returns the detected edge points (ContainerType_Vector_TcVnPoint2_REAL)
nSearchWindowHeight	UDINT	Specifies the height of the search window, i.e. the number of rows for horizontal or columns for vertical search directions, centered around the line between aStartPoint and aEndPoint. Depending on the orientation, the rows or columns inside that window are averaged and the edges are located on that averaged row or column. Therefore it is important that the edges inside the defined search window are straight and perpendicular to the search direction.

Name	Type	Description
nKernelSize	UDINT	The size of the Gaussian derivative kernel, that is used to locate the edges. The size must be odd and >= 3 (5 or 7 are usually a good choice for sharp edges). A suitable sigma parameter for the Gaussian derivative function is automatically chosen.
nNumEdges	UDINT	The (maximum) number of edges to search for
eEdgeSelection	ETcVnEdgeSelection [▶ 154]	Selects the edge directions to search for
fMinStrength	REAL	Specification of the minimum strength (absolute peak height of the Gaussian derivative of the pixel intensities) of the edge to search for. This value depends on the edge sharpness and nKernelSize, but is generally smaller or equal to the difference of the pixel intensities of the edge. Sharp edges have a higher peak than unsharp ones. Initially, you might choose a relatively small value (e.g. 30) and then increase it depending on the fAvgStrength output.
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

In/Outputs

Name	Type	Description
aStartPoint	TcVnPoint2_DINT [▶ 152]	Position from which to start the search process (in the direction of aEndPoint)
aEndPoint	TcVnPoint2_DINT [▶ 152]	Position where the search process ends. As the search direction must be parallel to one of the images axes, aEndPoint must have the same x- or y-coordinate as aStartPoint.

Outputs

Name	Type	Description
fAvgStrength	REAL	Returns the average strength of the detected edges

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Metrology 2D

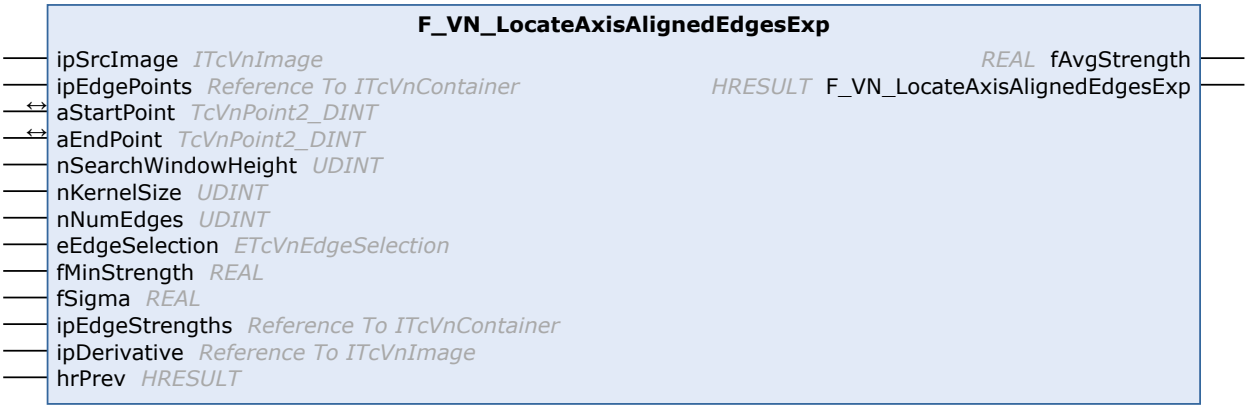
系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

还请参阅有关此

- F_VN_LocateAxisAlignedEdgesExp [▶ 1459]
- F_VN_LocateEdges [▶ 1476]

6.1.4.21.2 F_VN_LocateAxisAlignedEdgesExp



Locate the position of multiple occurring edges inside a specified search window. The edges inside the search window must be straight and aligned to one of the image axes. They are located by interpolating the peaks of the first Gaussian derivative of the pixel intensities. (expert function)

Syntax

Definition:

```
FUNCTION F_VN_LocateAxisAlignedEdgesExp : HRESULT
VAR_INPUT
    ipSrcImage          : ITcVnImage;
    ipEdgePoints        : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aStartPoint         : TcVnPoint2_DINT;
    aEndPoint           : TcVnPoint2_DINT;
END_VAR
VAR_INPUT
    nSearchWindowHeight : UDINT;
    nKernelSize         : UDINT;
    nNumEdges           : UDINT;
    eEdgeSelection      : ETcVnEdgeSelection;
    fMinStrength        : REAL;
    fSigma              : REAL;
    ipEdgeStrengths     : Reference To ITcVnContainer;
    ipDerivative        : Reference To ITcVnImage;
    hrPrev              : HRESULT;
END_VAR
VAR_OUTPUT
    fAvgStrength        : REAL;
END_VAR
```

Inputs

Name	Type	Description
ipSrcImage	ITcVnImage [▶ 407]	Source image (1 channel of type ET_USINT, ET_UINT, ET_INT or ET_REAL)
ipEdgePoints	Reference To ITcVnContainer [▶ 365]	Returns the detected edge points (ContainerType_Vector_TcVnPoint2_REAL)
nSearchWindowHeight	UDINT	Specifies the height of the search window, i.e. the number of rows for horizontal or columns for vertical search directions, centered around the line between aStartPoint and aEndPoint. Depending on the orientation, the rows or columns inside that window are averaged and the edges are located on that averaged row or column. Therefore it is important that the edges inside the defined search window are straight and perpendicular to the search direction.
nKernelSize	UDINT	The size of the Gaussian derivative kernel, that is used to locate the edges. The size must be odd and >= 3 (5 or 7 are usually a good choice for sharp edges).
nNumEdges	UDINT	The (maximum) number of edges to search for

Name	Type	Description
eEdgeSelection	ETcVnEdgeSelection [▶ 154]	Selects the edge directions to search for
fMinStrength	REAL	Specification of the minimum strength (absolute peak height of the Gaussian derivative of the pixel intensities) of the edge to search for. This value depends on the edge sharpness and nKernelSize, but is generally smaller or equal to the difference of the pixel intensities of the edge. Sharp edges have a higher peak than unsharp ones. Initially, you might choose a relatively small value (e.g. 30) and then increase it depending on the fAvgStrength and ipEdgeStrengths output.
fSigma	REAL	The sigma parameter for the Gaussian derivative function. If set to 0, a suitable value is automatically chosen depending on nKernelSize. For manually defined sigma, the value should always be chosen depending on nKernelSize (e.g. fSigma = (nKernelSize - 1) / factor, with factor in a range of 4 to 8).
ipEdgeStrengths	Reference To ITcVnContainer [▶ 365]	Returns the individual edge strengths, where dark to light edge strengths are positive and light to dark edge strengths are negative (optional, set to 0 if not required; ContainerType_Vector_REAL).
ipDerivative	Reference To ITcVnImage [▶ 407]	Returns the Gaussian derivative of the pixel intensities on the search line between aStartPoint and aEndPoint as a single image row, where aStartPoint would be on the left and aEndPoint on the right (optional, set to 0 if not required; 1 channel, ET_REAL).
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

In/Outputs

Name	Type	Description
aStartPoint	TcVnPoint2_DINT [▶ 152]	Position from which to start the search process (in the direction of aEndPoint)
aEndPoint	TcVnPoint2_DINT [▶ 152]	Position where the search process ends. As the search direction must be parallel to one of the images axes, aEndPoint must have the same x- or y-coordinate as aStartPoint.

Outputs

Name	Type	Description
fAvgStrength	REAL	Returns the average strength of the detected edges

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Metrology 2D

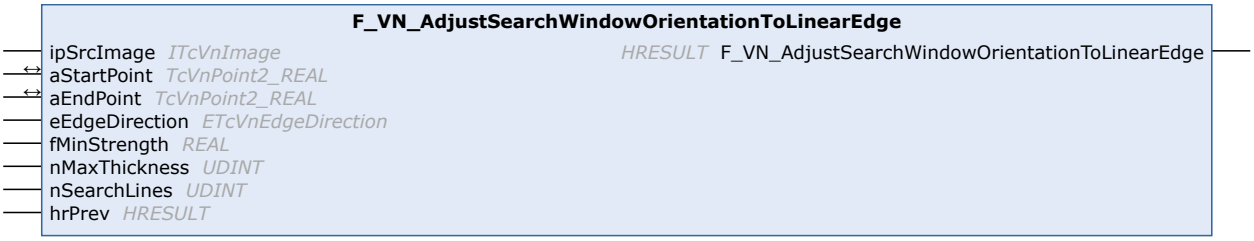
系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

还请参阅有关此

- F_VN_LocateAxisAlignedEdges [▶ 1457]
- F_VN_PlotIntensityProfileExp [▶ 1051]

6.1.4.21.3 F_VN_AdjustSearchWindowOrientationToLinearEdge



调整搜索窗口，使其与线性边缘正交。在调用依赖搜索窗口的边缘定位和距离测量函数之前可能会有所帮助。可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_AdjustSearchWindowOrientationToLinearEdge : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
END_VAR
VAR_IN_OUT
    aStartPoint     : TcVnPoint2_REAL;
    aEndPoint       : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    eEdgeDirection : ETcVnEdgeDirection;
    fMinStrength   : REAL;
    nMaxThickness  : UDINT;
    nSearchLines   : UDINT;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aStartPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的起始位置（沿 aEndPoint 方向），该位置不变

名称	类型	描述
aEndPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的结束位置，由该函数调整

返回值

HRESULT [▶ 135]

更多信息

F_VN_AdjustSearchWindowOrientationToLinearEdge 函数会旋转测量函数的搜索区域，使其与线性边缘正交。

参数

输入图像

ipSrcImage 输入图像必须包含符合边缘检测参数的边缘。

搜索窗口 (返回值)

要调整的搜索窗口作为引用进行传输，以便函数可以对其进行更改。搜索窗口由中心搜索线（由 aStartPoint 起点和 aEndPoint 终点标识）以及 nSearchLines 搜索线的数量组成。

函数调用之后，aStartPoint 和 aEndPoint 包含调整后的中心搜索线。现在，它与 ipSrcImage 输入图像上的线性边缘正交。

边缘定位的参数

关于其余参数的详细解释，请参见边缘定位一章。

应用

例如，将搜索窗口适配到边缘的效果是这样的：

```
hr := F_VN_AdjustSearchWindowOrientationToLinearEdge(
    ipSrcImage      := ipImageIn,
    aStartPoint     := aStartPoint,
    aEndPoint       := aEndPoint,
    eEdgeDirection  := eDirection,
    fMinStrength    := fMinStrength,
    nMaxThickness   := nMaxThickness,
    nSearchLines    := nSearchLines,
    hrPrev          := hr);
```

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.4 F_VN_ClosestPointsBF



用蛮力法找出两个相同类型的二维点集之间的最近距离。

语法

定义：

```
FUNCTION F_VN_ClosestPointsBF : HRESULT
VAR_INPUT
    ipPointSet1 : ITcVnContainer;
    ipPointSet2 : ITcVnContainer;
END_VAR
VAR_IN_OUT
    fMinDist    : LREAL;
    aPoint1     : TcVnPoint2_LREAL;
    aPoint2     : TcVnPoint2_LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipPointSet1	ITcVnContainer [▶ 365]	包含第一组二维点的容器
ipPointSet2	ITcVnContainer [▶ 365]	包含第二组二维点的容器
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fMinDist	LREAL	返回不同点集的两点之间的最小距离
aPoint1	TcVnPoint2_LREAL [▶ 152]	返回 ipPointSet1 中达到 fMinDist 的点
aPoint2	TcVnPoint2_LREAL [▶ 152]	返回 ipPointSet2 中达到 fMinDist 的点

返回值

HRESULT [▶ 135]

更多信息

F_VN_ClosestPointsBF 函数采用穷举法确定两个二维点云中哪些点的距离最小。

参数

点云（二维点云）

在 ipPointSet1 和 ipPointSet2 二维点云中寻找最近的两个点。

最小距离 (返回值)

fMinDist 参数返回彼此距离最近的两个点之间的距离。这相当于两个二维点云之间的最小距离。

最近点 (返回值)

aPoint1 和 aPoint2 参数返回二维点云中彼此距离最近的两个点。

应用

例如，确定两条线之间的最小距离是这样的：

```
hr := F_VN_ClosestPointsBF(
    ipPointSet1 := ipPointSet1,
    ipPointSet2 := ipPointSet2,
    fMinDist    := fMinDist,
    aPoint1     := aPoint1,
    aPoint2     := aPoint2,
    hrPrev      := hr);
```

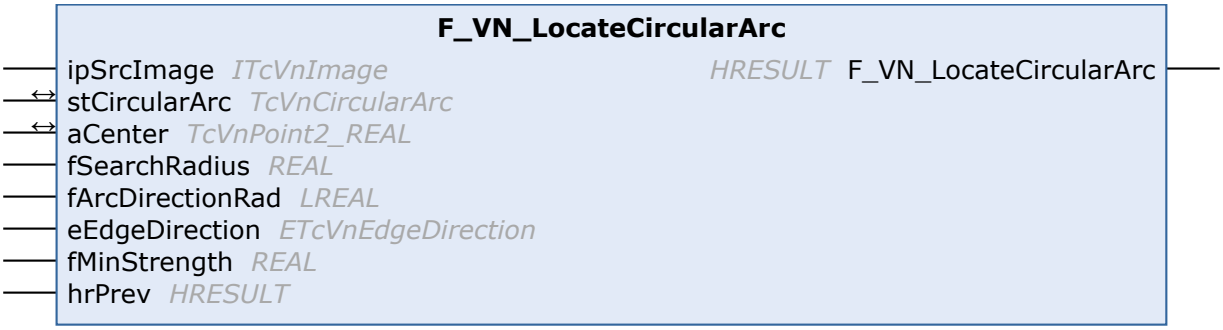
相关函数

- ClosestPointsBF [▶ 1462]，用于计算两个点云中最近的点
- MeasureAngleBetweenEdges [▶ 1490]，用于测量两条边之间的角度
- MeasureEdgeDistance [▶ 1496]，用于测量两条边之间的平均距离
- MeasureMinEdgeDistance [▶ 1505]，用于测量两条边之间的最小距离

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.5 F_VN_LocateCircularArc



使用插值法定位一个圆弧。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_LocateCircularArc : HRESULT
VAR_INPUT
    ipSrcImage       : ITcVnImage;
END_VAR
VAR_IN_OUT
    stCircularArc    : TcVnCircularArc;
    aCenter          : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    fSearchRadius    : REAL;
    fArcDirectionRad : LREAL;
    eEdgeDirection   : ETcVnEdgeDirection;
    fMinStrength     : REAL;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
fSearchRadius	REAL	输入搜索半径（从 aCenter 开始，应大于实际圆半径，但 aCenter + fSearchRadius 应在图像边界范围内）
fArcDirectionRad	LREAL	输入以弧度为单位的搜索起始方向。有效范围为 [-pi，+pi]，其中右侧为 0 rad，顶部为 -pi/2 rad，底部为 +pi/2 rad。圆弧至少应在 fArcDirectionRad +- 0.4 rad 范围内有效。
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stCircularArc	TcVnCircularArc [▶ 223]	返回圆弧参数
aCenter	TcVnPoint2_REAL [▶ 152]	输入估算的圆心（仅用作搜索圆轮廓的起点，不用于圆心估算）

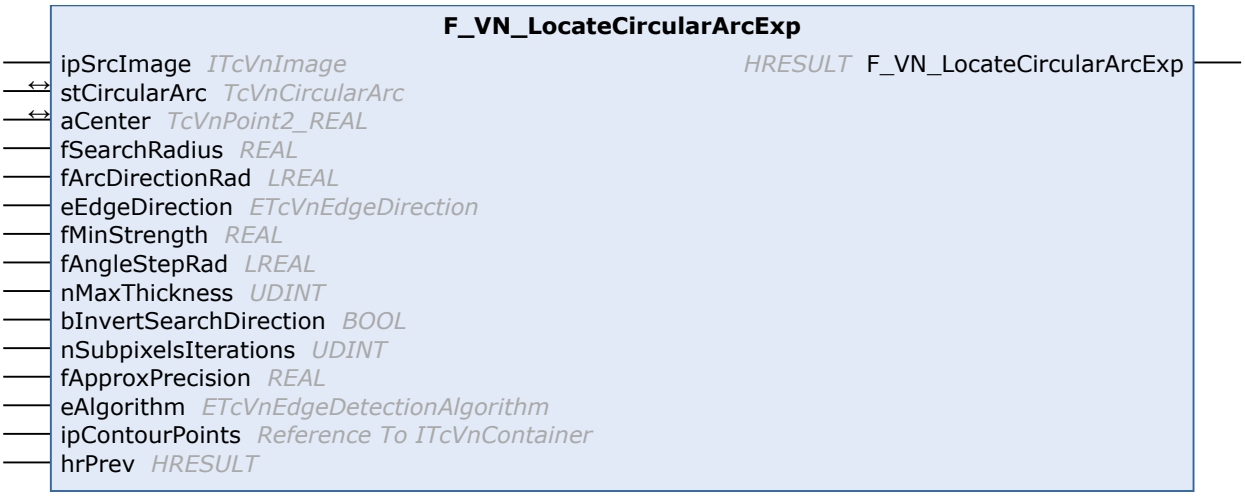
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.6 F_VN_LocateCircularArcExp



定位圆弧。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_LocateCircularArcExp : HRESULT
VAR_INPUT
    ipSrcImage              : ITcVnImage;
END_VAR
VAR_IN_OUT
    stCircularArc           : TcVnCircularArc;
    aCenter                 : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    fSearchRadius           : REAL;
    fArcDirectionRad        : LREAL;
    eEdgeDirection          : ETcVnEdgeDirection;
    fMinStrength            : REAL;
    fAngleStepRad           : LREAL;
    nMaxThickness           : UDINT;
    bInvertSearchDirection  : BOOL;
    nSubpixelsIterations    : UDINT;
    fApproxPrecision        : REAL;
    eAlgorithm              : ETcVnEdgeDetectionAlgorithm;
    ipContourPoints         : Reference To ITcVnContainer;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
fSearchRadius	REAL	输入搜索半径（从 aCenter 开始，应大于实际圆半径，但 aCenter + fSearchRadius 应在图像边界范围内）
fArcDirectionRad	LREAL	输入以弧度为单位的搜索起始方向。有效范围为 [-pi，+pi]，其中右侧为 0 rad，顶部为 -pi/2 rad，底部为 +pi/2 rad。圆弧至少应在 fArcDirectionRad +- 4 * fAngleStepRad 范围内有效
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
fAngleStepRad	LREAL	搜索步长，以弧度为单位（选择时应确保整个弧线有大约 20 - 60 个步长。在大多数情况下，0.1 弧度（5.7 度）是良好值）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
bInvertSearchDirection	BOOL	如果为 true，则搜索从圆弧外沿圆心方向开始
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
ipContourPoints	引用 ITcVnContainer [▶ 365]	返回亚像素精确轮廓（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL；重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stCircularArc	TcVnCircularArc [▶ 223]	返回圆弧参数
aCenter	TcVnPoint2_REAL [▶ 152]	输入估算的圆心（仅用作搜索圆轮廓的起点，不用于圆心估算）

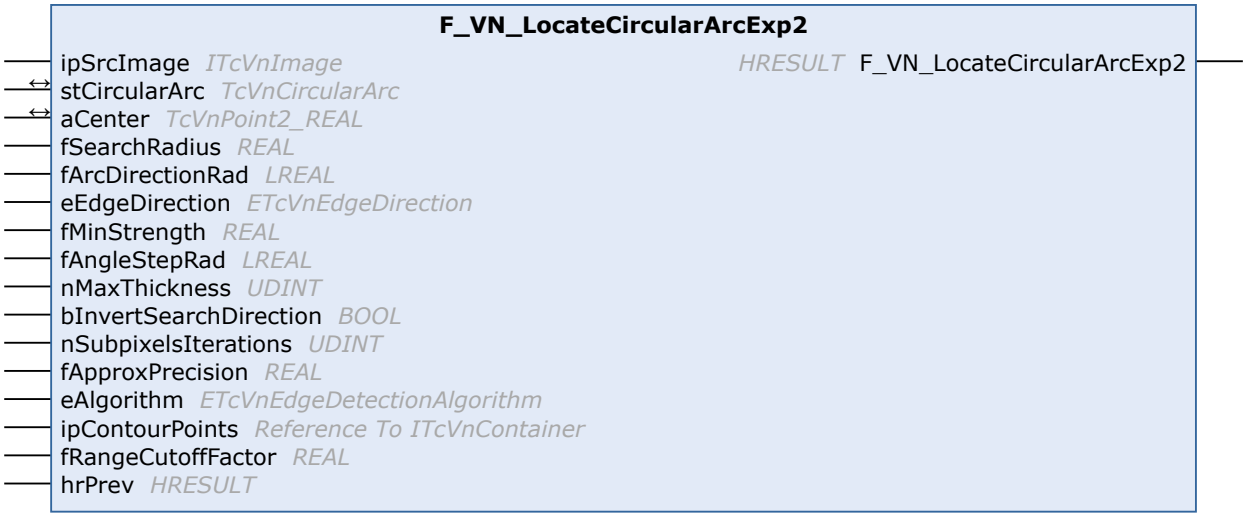
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.7 F_VN_LocateCircularArcExp2



定位圆弧。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_LocateCircularArcExp2 : HRESULT
VAR_INPUT
    ipSrcImage              : ITcVnImage;
END_VAR
VAR_IN_OUT
    stCircularArc           : TcVnCircularArc;
    aCenter                 : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    fSearchRadius           : REAL;
    fArcDirectionRad        : LREAL;
    eEdgeDirection          : ETcVnEdgeDirection;
    fMinStrength            : REAL;
    fAngleStepRad           : LREAL;
    nMaxThickness           : UDINT;
    bInvertSearchDirection  : BOOL;
    nSubpixelsIterations    : UDINT;
    fApproxPrecision        : REAL;
    eAlgorithm              : ETcVnEdgeDetectionAlgorithm;
    ipContourPoints         : Reference To ITcVnContainer;
    fRangeCutoffFactor      : REAL;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
fSearchRadius	REAL	输入搜索半径（从 aCenter 开始，应大于实际圆半径，但 aCenter + fSearchRadius 应在图像边界范围内）
fArcDirectionRad	LREAL	输入以弧度为单位的搜索起始方向。有效范围为 [-pi，+pi]，其中右侧为 0 rad，顶部为 -pi/2 rad，底部为 +pi/2 rad。圆弧至少应在 fArcDirectionRad +- 4 * fAngleStepRad 范围内有效
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
fAngleStepRad	LREAL	搜索步长，以弧度为单位（选择时应确保整个弧线有大约 20 - 60 个步长。在大多数情况下，0.1 弧度（5.7 度）是良好值）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength

名称	类型	描述
bInvertSearchDirection	BOOL	如果为 true，则搜索从圆弧外沿圆心方向开始
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
ipContourPoints	引用 ITcVnContainer [▶ 365]	返回亚像素精确轮廓（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL；重复使用非零接口指针。）
fRangeCutoffFactor	REAL	用于剔除参与拟合的像素范围的相对边缘强度系数。负数表示包含噪声，即另一个方向上的强度差异。每个方向最多使用 nMaxThickness 个像素。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stCircularArc	TcVnCircularArc [▶ 223]	返回圆弧参数
aCenter	TcVnPoint2_REAL [▶ 152]	输入估算的圆心（仅用作搜索圆轮廓的起点，不用于圆心估算）

返回值

HRESULT [▶ 135]

更多信息

`F_VN_LocateCircularArcExp2` 函数是 F_VN_LocateCircularArc [▶ 1464] 的专家版本。它包含额外的参数。

参数

输入图像

`ipSrcImage` 输入图像必须是单通道灰度图像。

圆弧 (返回值)

`stCircularArc` 参数以 TcVnCircularArc [▶ 223] 类型的结构返回定位到的圆弧。

估计的中心点

务必指定圆弧估计的 `aCenter` 中心点，以确定搜索线的位置。

如果圆的中心点位置未知，则可能需要先通过目标检测 [▶ 1169]的方法来确定它。

搜索半径

`fSearchRadius` 搜索半径定义了从圆的中心点出发的搜索线长度（以像素为单位）。搜索半径应定义得比圆的实际半径略大一些，以便能够明确无误地进行边缘定位。

搜索方向

fArcDirectionRad 搜索方向指定了边缘搜索应开始的圆周方向（从圆的中心点出发）。该值以弧度为单位，取值范围为 [-π, π]。右侧对应 0，上方对应 -π/2，下方对应 π/2。从 fArcDirectionRad 开始，圆弧必须至少出现在 +- 0.4 的范围内才能被检测到。建议将目标对准圆弧的中心。

径向搜索线距离 (专家)

fAngleStepRad 径向搜索线距离定义了搜索线之间的角度步长大小。它以弧度为单位进行指定。选择的距离应使得完整的圆弧被划分为大约 20 – 60 个步长。在大多数情况下，0,1 rad （5,7°）是一个合适的值。

搜索方向反转 (专家)

通过 bInvertSearchDirection 布尔变量可以反转搜索方向。在这种情况下，其含义如下：

- FALSE：从圆的中心点向圆弧外部搜索。
- TRUE：从圆弧外部向圆的中心点搜索

轮廓点 (专家，返回值)

ipContourPoints 容器返回在圆弧上找到的边缘点，因此属于 ContainerType_Vector_TcVnPoint2_REAL 类型。每条成功找到的搜索线对应容器中的一个点。

如果不需要返回边缘点，可将该参数设置为 0。

圆弧 stCircularArc

圆弧 stCircularArc 是根据这些点的集合拟合得到的。

边缘定位的参数

关于其余参数的详细解释，请参见边缘定位一章。

应用

例如，使用专家参数对圆弧进行定位的过程是这样的：

```
hr := F_VN_LocateCircularArcExp2(
    ipSrcImage              := ipImageIn,
    stCircularArc           := stArc,
    aCenter                 := aCenter,
    fSearchRadius           := 270,
    fArcDirectionRad        := 2.1,
    eEdgeDirection          := TCVN_ED_LIGHT_TO_DARK,
    fMinStrength            := 100,
    fAngleStepRad           := 0.1,
    nMaxThickness           := 10,
    bInvertSearchDirection  := FALSE,
    nSubpixelsIterations    := 10,
    fApproxPrecision        := 0.001,
    eAlgorithm              := TCVN_EDA_INTERPOLATION,
    ipContourPoints         := ipEllipseContour,
 fRangeCutoffFactor   := 0.135,
    hrPrev                  := hr);
```

使用 F_VN_DrawContours [▶ 1012] 函数可以将定位到的点可视化。

```
hr := F_VN_DrawContours(ipEllipseContour, -1, ipImageRes, aColorGreen, 5, hr);
```

示例

- 定位圆弧 [▶ 2793]

相关函数

- LocateCircularArc [▶ 1464]，用于定位比例圆弧
- LocateEdge [▶ 1470]，用于定位线性边缘
- LocateEdges [▶ 1476]，用于定位几个线性边缘
- LocateEllipse [▶ 1485]，用于定位完整椭圆

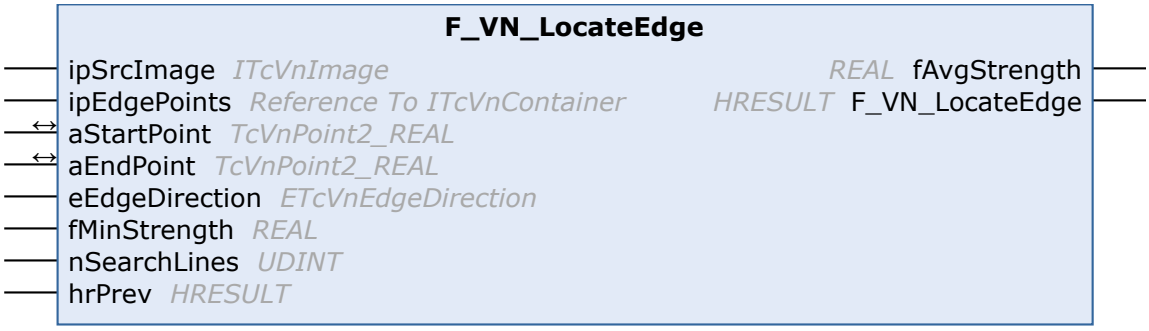
所需的许可证

TC3 Vision Metrology 2D

还请参阅有关此

F_VN_LocateAxisAlignedEdges [▶ 1457]

6.1.4.21.8 F_VN_LocateEdge



使用插值法定位指定搜索窗口内首次出现的边缘的点。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_LocateEdge : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipEdgePoints    : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aStartPoint     : TcVnPoint2_REAL;
    aEndPoint       : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    eEdgeDirection : ETcVnEdgeDirection;
    fMinStrength    : REAL;
    nSearchLines    : UDINT;
    hrPrev          : HRESULT;
END_VAR
VAR_OUTPUT
    fAvgStrength    : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
ipEdgePoints	引用 ITcVnContainer [▶ 365]	返回检测到的边缘点（ContainerType_Vector_TcVnPoint2_REAL）
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aStartPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的起始位置（沿 aEndPoint 方向）
aEndPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的结束位置

输出

名称	类型	描述
fAvgStrength	REAL	返回检测到的边缘的平均强度

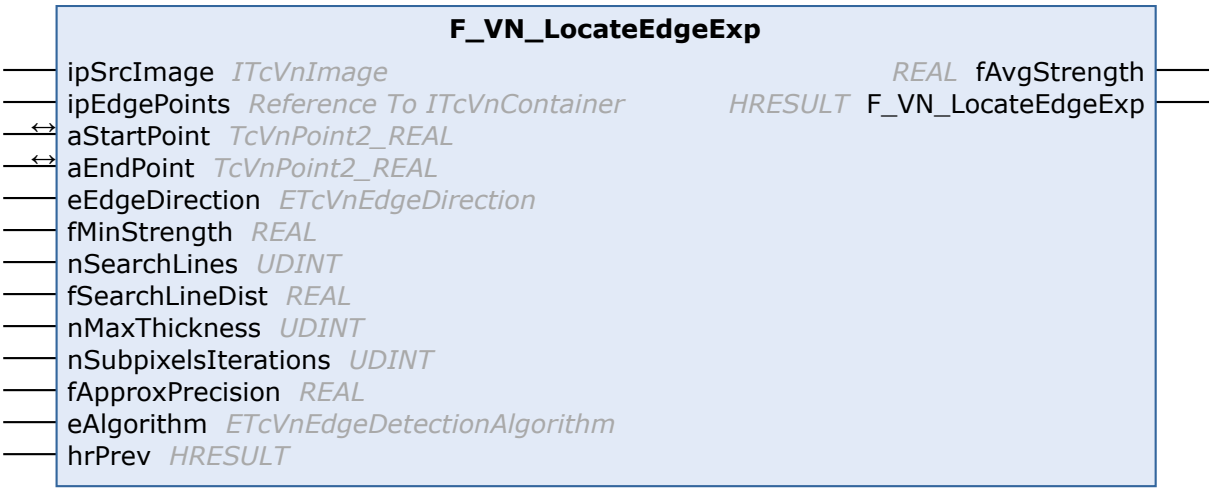
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.9 F_VN_LocateEdgeExp



以亚像素精度定位指定搜索窗口内首次出现的边缘的点。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_LocateEdgeExp : HRESULT
VAR_INPUT
    ipSrcImage           : ITcVnImage;
    ipEdgePoints         : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aStartPoint          : TcVnPoint2_REAL;
    aEndPoint            : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    eEdgeDirection       : ETcVnEdgeDirection;
    fMinStrength         : REAL;
    nSearchLines         : UDINT;
    fSearchLineDist      : REAL;
    nMaxThickness        : UDINT;
    nSubpixelsIterations : UDINT;
    fApproxPrecision     : REAL;
    eAlgorithm           : ETcVnEdgeDetectionAlgorithm;
    hrPrev               : HRESULT;
END_VAR
```

```
VAR_OUTPUT
    fAvgStrength         : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
ipEdgePoints	引用 ITcVnContainer [▶ 365]	返回检测到的边缘点（ContainerType_Vector_TcVnPoint2_REAL）
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
fSearchLineDist	REAL	搜索线之间的距离，以像素为单位（> 0）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aStartPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的起始位置（沿 aEndPoint 方向）
aEndPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的结束位置

输出

名称	类型	描述
fAvgStrength	REAL	返回检测到的边缘的平均强度

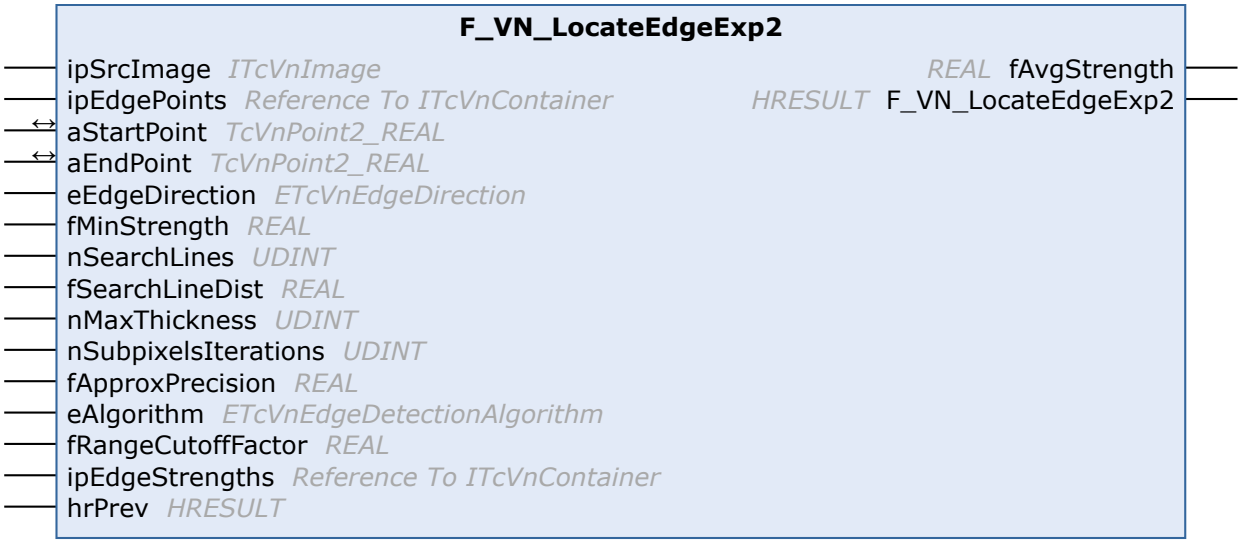
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.10 F_VN_LocateEdgeExp2



以亚像素精度定位指定搜索窗口内首次出现的边缘的点。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_LocateEdgeExp2 : HRESULT
VAR_INPUT
    ipSrcImage           : ITcVnImage;
    ipEdgePoints         : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aStartPoint          : TcVnPoint2_REAL;
    aEndPoint            : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    eEdgeDirection       : ETcVnEdgeDirection;
    fMinStrength         : REAL;
    nSearchLines         : UDINT;
    fSearchLineDist      : REAL;
    nMaxThickness        : UDINT;
    nSubpixelsIterations : UDINT;
    fApproxPrecision     : REAL;
    eAlgorithm           : ETcVnEdgeDetectionAlgorithm;
    fRangeCutoffFactor   : REAL;
    ipEdgeStrengths      : Reference To ITcVnContainer;
    hrPrev               : HRESULT;
END_VAR
VAR_OUTPUT
    fAvgStrength         : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
ipEdgePoints	引用 ITcVnContainer [▶ 365]	返回检测到的边缘点（ContainerType_Vector_TcVnPoint2_REAL）
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
fSearchLineDist	REAL	搜索线之间的距离，以像素为单位（> 0）

名称	类型	描述
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
fRangeCutoffFactor	REAL	用于剔除参与拟合的像素范围的相对边缘强度系数。负数表示包含噪声，即另一个方向上的强度差异。每个方向最多使用 nMaxThickness 个像素。
ipEdgeStrengths	引用 ITcVnContainer [▶ 365]	返回检测到的边缘强度（可选，如果不需要，则设为 0；ContainerType_Vector_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aStartPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的起始位置（沿 aEndPoint 方向）
aEndPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的结束位置

输出

名称	类型	描述
fAvgStrength	REAL	返回检测到的边缘的平均强度

返回值

HRESULT [▶ 135]

更多信息

`F_VN_LocateEdgeExp2` 函数是 F_VN_LocateEdge [▶ 1470] 的专家版本。它包含额外的参数。

参数

输入图像

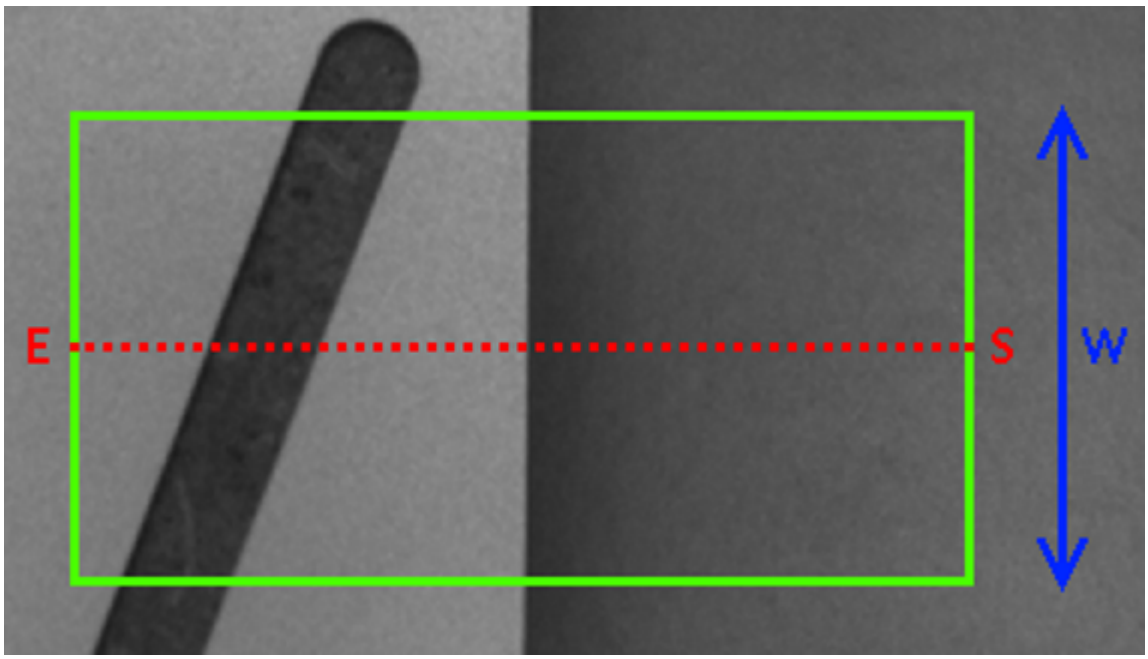
`ipSrcImage` 输入图像必须是单通道灰度图像。

定位点 (返回值)

`ipEdgePoints` 容器返回线性边缘上定位到的点。

搜索窗口

首先，必须定义一个搜索窗口，以确定在哪个位置以及沿哪个方向搜索边缘。这是通过 `aStartPoint`、`aEndPoint` 和 `nSearchLines` 参数实现的。搜索窗口由从 `aStartPoint` 到 `aEndPoint` 的搜索线及其两侧平行线构成，每个方向各有（`nSearchLines` - 1）/2 条搜索线，相邻搜索线的距离为 1 个像素。因此，`nSearchLines` 定义了搜索线的数量，且该值必须始终为奇数。



附图 15: S：起点，E：终点，W：窗口宽度

边缘强度 (返回值)

`fAvgStrength` 返回值显示定位到的边缘的实际对比度（对所有搜索线取平均值）。

边缘定位的参数

关于其余参数的详细解释，请参见边缘定位一章。

边缘厚度 (返回值)

各条搜索线检测到的边缘厚度通过 `ipEdgeStrengths` 返回到类型为 `ContainerType_Vector_REAL` 的容器中。该参数为可选参数，如果不需要，可设置为 0。

应用

例如，使用专家参数对线性边缘进行定位的过程是这样的：

```
hr := F_VN_LocateEdgeExp2(
    ipSrcImage              := ipImageIn,
    ipEdgePoints            := ipEdgePoints,
    aStartPoint             := aStartPoint,
    aEndPoint               := aEndPoint,
    eEdgeDirection          := eDirection,
    fMinStrength            := fMinStrength,
    nSearchLines            := nSearchLines,
    fSearchLineDist         := fSearchLineDist,
    nMaxThickness           := nMaxThickness,
    nSubpixelsIterations    := nSubpixIter,
    fApproxPrecision        := 0.001,
    eAlgorithm              := eAlgorithm,
  fRangeCutoffFactor    := 0.135,
  ipEdgeStrengths       := ipEdgeStrengths,
    hrPrev                  := hr,
    fAvgStrength            => fAvgStrength);
```

示例

- 定位边缘 [▶ 2797]

相关函数

- LocateCircularArc [▶ 1464]，用于定位比例圆弧
- LocateEdge [▶ 1470]，用于定位线性边缘
- LocateEdges [▶ 1476]，用于定位几个线性边缘

- LocateEllipse [▶ 1485]，用于定位完整椭圆

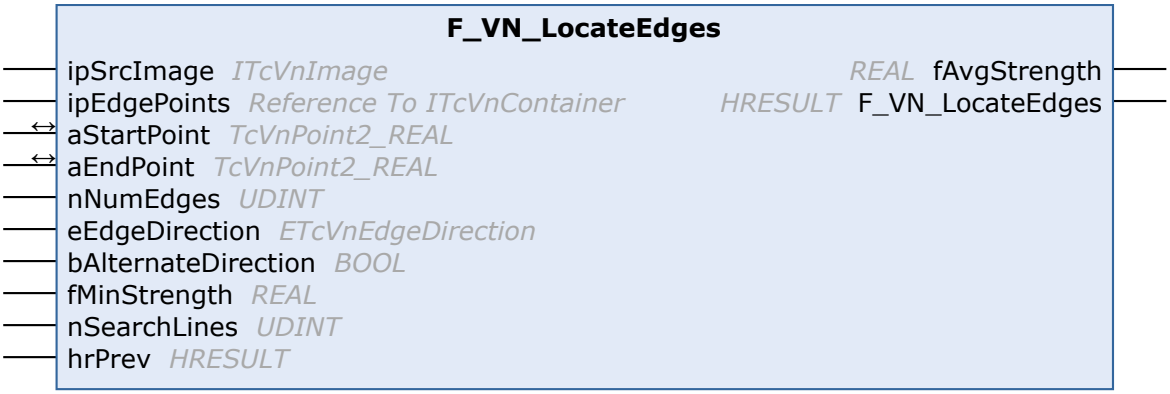
所需的许可证

TC3 Vision Metrology 2D

还请参阅有关此

F_VN_LocateAxisAlignedEdges [▶ 1457]

6.1.4.21.11 F_VN_LocateEdges



使用插值法定位指定搜索窗口内多次出现的边缘的点。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被看门狗取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_LocateEdges : HRESULT
VAR_INPUT
    ipSrcImage          : ITcVnImage;
    ipEdgePoints        : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aStartPoint         : TcVnPoint2_REAL;
    aEndPoint           : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    nNumEdges           : UDINT;
    eEdgeDirection      : ETcVnEdgeDirection;
    bAlternateDirection : BOOL;
    fMinStrength        : REAL;
    nSearchLines        : UDINT;
    hrPrev              : HRESULT;
END_VAR
VAR_OUTPUT
    fAvgStrength        : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
ipEdgePoints	引用 ITcVnContainer [▶ 365]	返回检测到的边缘点（ContainerType_Vector_Vector_TcVnPoint2_REAL）
nNumEdges	UDINT	要搜索的边缘的（最大）数量
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向
bAlternateDirection	BOOL	如果为 true，eEdgeDirection 会在每次检测到边缘后交替出现。否则，仅搜索具有 eEdgeDirection 的边缘。
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）

名称	类型	描述
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aStartPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的起始位置（沿 aEndPoint 方向）
aEndPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的结束位置

输出

名称	类型	描述
fAvgStrength	REAL	返回检测到的边缘的平均强度

返回值

HRESULT [▶ 135]

更多信息

`F_VN_LocateEdges` 函数在定义的搜索窗口中定位多个边缘。

参数

输入图像

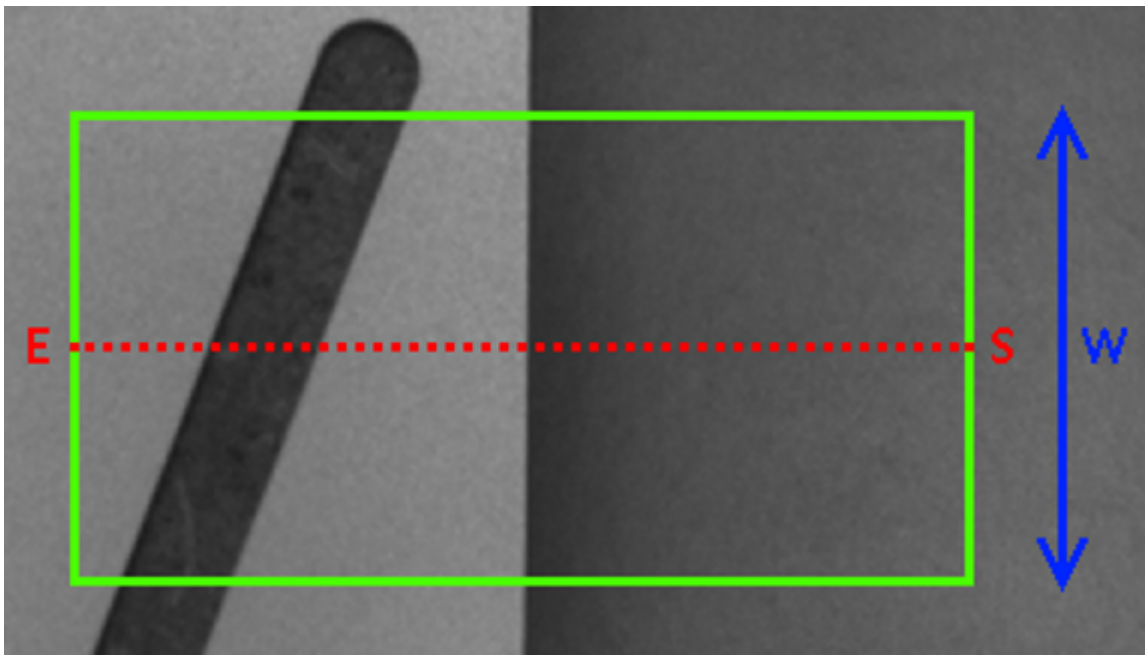
`ipSrcImage` 输入图像必须是单通道灰度图像。

定位点 (返回值)

`ipEdgePoints` 容器返回线性边缘上定位到的点。

搜索窗口

首先，必须定义一个搜索窗口，以确定在哪个位置以及沿哪个方向搜索边缘。这是通过 `aStartPoint`、`aEndPoint` 和 `nSearchLines` 参数实现的。搜索窗口由从 `aStartPoint` 到 `aEndPoint` 的搜索线及其两侧平行线构成，每个方向各有（`nSearchLines` - 1）/2 条搜索线，相邻搜索线的距离为 1 个像素。因此，`nSearchLines` 定义了搜索线的数量，且该值必须始终为奇数。



附图 16: S：起点，E：终点，W：窗口宽度

边缘强度 (返回值)

`fAvgStrength` 返回值显示定位到的边缘的实际对比度（对所有搜索线取平均值）。

边缘定位的参数

关于其余参数的详细解释，请参见边缘定位一章。

专家参数

F_VN_LocateEdgeExp [▶ 1479] 专家版本包含附加参数。

应用

例如，对线性边缘进行定位的过程是这样的：

```
hr := F_VN_LocateEdges(
    ipSrcImage          := ipImageIn,
    ipEdgePoints        := ipEdgePoints,
    aStartPoint         := aStartPoint,
    aEndPoint           := aEndPoint,
    nNumEdges           := nNumEdges,
    eEdgeDirection      := eDirection,
    bAlternateDirection := bAlternateDirection,
    fMinStrength        := fMinStrength,
    nSearchLines        := nSearchLines,
    hrPrev              := hr,
    fAvgStrength        => fAvgStrength);
```

可以使用函数 F_VN_DrawContours [▶ 1013] 将局部边缘点可视化：

```
F_VN_DrawContours(ipEdgePoints, -1, ipImageRes, aColorGreen, 5, hr);
```

相关函数

- LocateCircularArc [▶ 1464]，用于定位比例圆弧
- LocateEdge [▶ 1470]，用于定位线性边缘
- LocateEdges [▶ 1476]，用于定位几个线性边缘
- LocateEllipse [▶ 1485]，用于定位完整椭圆

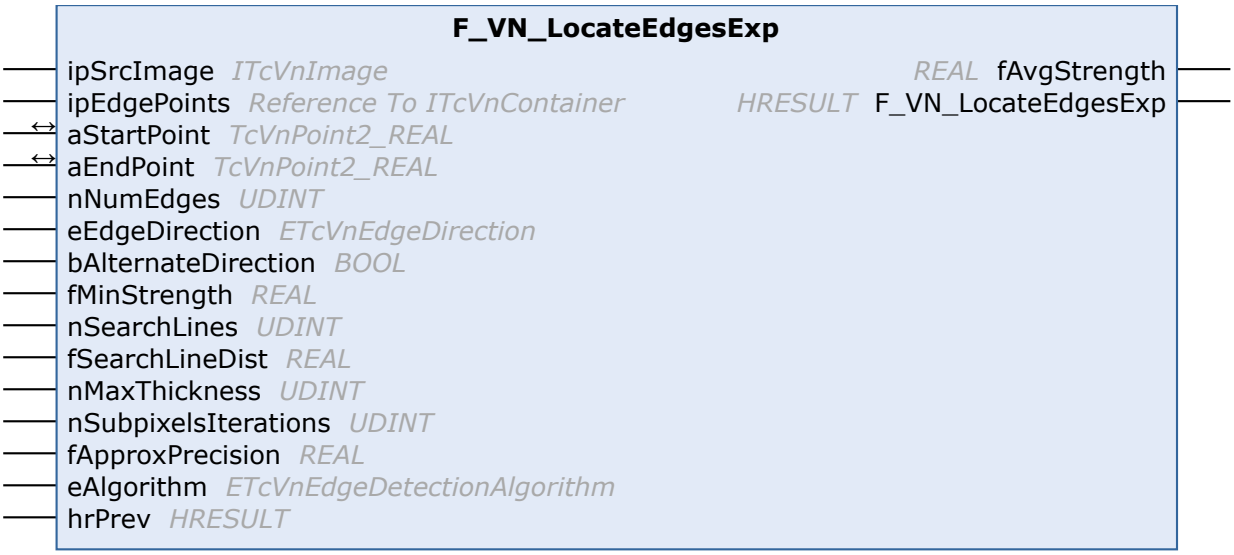
所需的许可证

TC3 Vision Metrology 2D

还请参阅有关此

F_VN_LocateAxisAlignedEdges [▶ 1457]

6.1.4.21.12 F_VN_LocateEdgesExp



以亚像素精度在指定搜索窗口内定位多次出现的边缘的点。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被看门狗取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_LocateEdgesExp : HRESULT
VAR_INPUT
    ipSrcImage           : ITcVnImage;
    ipEdgePoints         : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aStartPoint          : TcVnPoint2_REAL;
    aEndPoint            : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    nNumEdges            : UDINT;
    eEdgeDirection       : ETcVnEdgeDirection;
    bAlternateDirection  : BOOL;
    fMinStrength         : REAL;
    nSearchLines         : UDINT;
    fSearchLineDist      : REAL;
    nMaxThickness        : UDINT;
    nSubpixelsIterations : UDINT;
    fApproxPrecision     : REAL;
    eAlgorithm           : ETcVnEdgeDetectionAlgorithm;
    hrPrev               : HRESULT;
END_VAR
VAR_OUTPUT
    fAvgStrength         : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
ipEdgePoints	引用 ITcVnContainer [▶ 365]	返回检测到的边缘点（ContainerType_Vector_Vector_TcVnPoint2_REAL）
nNumEdges	UDINT	要搜索的边缘的（最大）数量
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向

名称	类型	描述
bAlternateDirection	BOOL	如果为 true，eEdgeDirection 会在每次检测到边缘后交替出现。否则，仅搜索具有 eEdgeDirection 的边缘。
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
fSearchLineDist	REAL	搜索线之间的距离，以像素为单位（> 0）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aStartPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的起始位置（沿 aEndPoint 方向）
aEndPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的结束位置

输出

名称	类型	描述
fAvgStrength	REAL	返回检测到的边缘的平均强度

返回值

HRESULT [▶ 135]

更多信息

F_VN_LocateEdgesExp 函数是 F_VN_LocateEdges [▶ 1476] 的专家版本。它包含额外的参数。

参数

输入图像

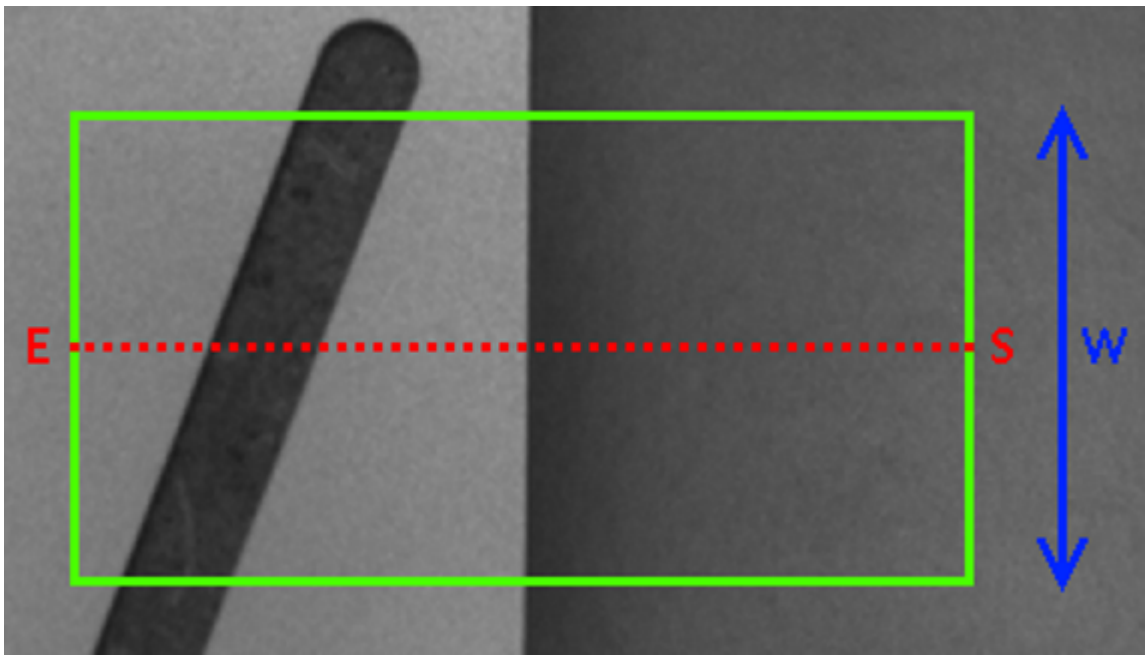
`ipSrcImage` 输入图像必须是单通道灰度图像。

定位点 (返回值)

`ipEdgePoints` 容器返回线性边缘上定位到的点。

搜索窗口

首先，必须定义一个搜索窗口，以确定在哪个位置以及沿哪个方向搜索边缘。这是通过 `aStartPoint`、`aEndPoint` 和 `nSearchLines` 参数实现的。搜索窗口由从 `aStartPoint` 到 `aEndPoint` 的搜索线及其两侧平行线构成，每个方向各有（`nSearchLines` - 1）/2 条搜索线，相邻搜索线的距离为 1 个像素。因此，`nSearchLines` 定义了搜索线的数量，且该值必须始终为奇数。



附图 17: S：起点，E：终点，W：窗口宽度

边缘强度 (返回值)

`fAvgStrength` 返回值显示定位到的边缘的实际对比度（对所有搜索线取平均值）。

边缘定位的参数

关于其余参数的详细解释，请参见边缘定位一章。

应用

例如，使用专家参数对线性边缘进行定位的过程是这样的：

```
hr := F_VN_LocateEdgesExp(
    ipSrcImage              := ipImageIn,
    ipEdgePoints            := ipEdgePoints,
    aStartPoint             := aStartPoint,
    aEndPoint               := aEndPoint,
    nNumEdges               := nNumEdges,
    eEdgeDirection          := eDirection,
    bAlternateDirection     := bAlternateDirection,
    fMinStrength            := fMinStrength,
    nSearchLines            := nSearchLines,
    fSearchLineDist     := fSearchLineDist,
    nMaxThickness       := nMaxThickness,
    nSubpixelsIterations   := nSubpixIter,
    fApproxPrecision     := 0.001,
    eAlgorithm         := eAlgorithm,
    hrPrev                  := hr,
    fAvgStrength            => fAvgStrength);
```

相关函数

- LocateCircularArc [▶ 1464]，用于定位比例圆弧
- LocateEdge [▶ 1470]，用于定位线性边缘
- LocateEdges [▶ 1476]，用于定位几个线性边缘
- LocateEllipse [▶ 1485]，用于定位完整椭圆

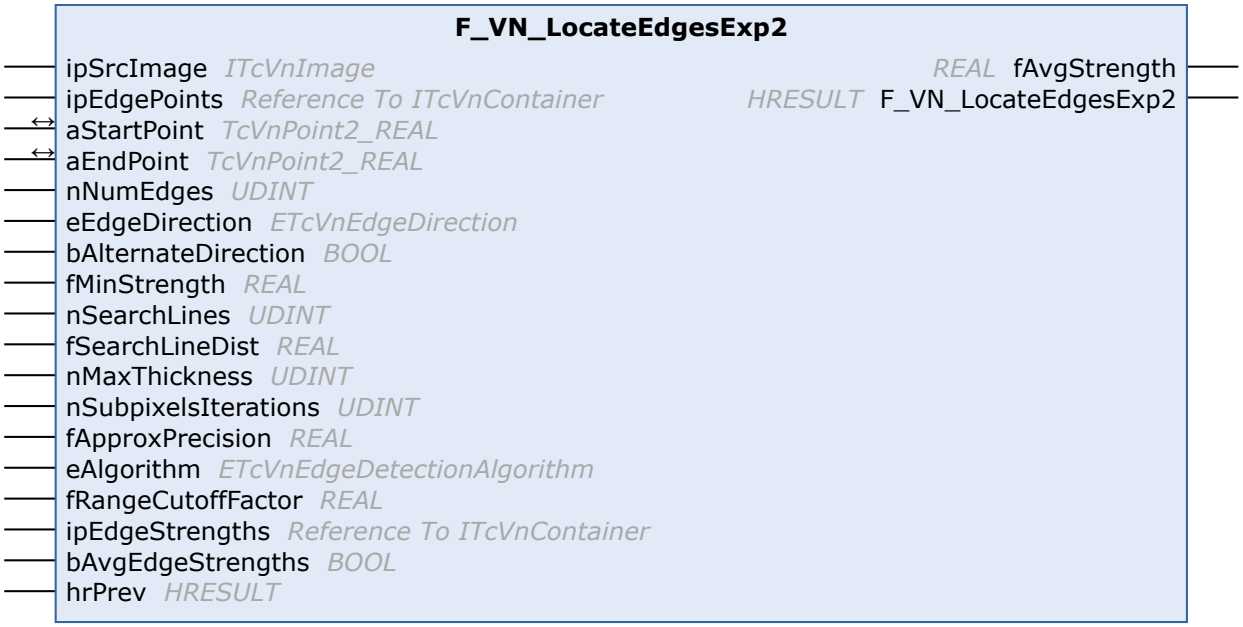
所需的许可证

TC3 Vision Metrology 2D

还请参阅有关此

- F_VN_LocateAxisAlignedEdges [▶ 1457]

6.1.4.21.13 F_VN_LocateEdgesExp2



以亚像素精度在指定搜索窗口内定位多次出现的边缘的点。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_LocateEdgesExp2 : HRESULT
VAR_INPUT
    ipSrcImage           : ITcVnImage;
    ipEdgePoints         : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aStartPoint          : TcVnPoint2_REAL;
    aEndPoint            : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    nNumEdges            : UDINT;
    eEdgeDirection       : ETcVnEdgeDirection;
    bAlternateDirection  : BOOL;
    fMinStrength         : REAL;
    nSearchLines         : UDINT;
    fSearchLineDist      : REAL;
    nMaxThickness        : UDINT;
    nSubpixelsIterations : UDINT;
    fApproxPrecision     : REAL;
    eAlgorithm           : ETcVnEdgeDetectionAlgorithm;
    fRangeCutoffFactor   : REAL;
    ipEdgeStrengths      : Reference To ITcVnContainer;
    bAvgEdgeStrengths    : BOOL;
    hrPrev               : HRESULT;
END_VAR
VAR_OUTPUT
    fAvgStrength         : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
ipEdgePoints	引用 ITcVnContainer [▶ 365]	返回检测到的边缘点（ContainerType_Vector_Vector_TcVnPoint2_REAL）
nNumEdges	UDINT	要搜索的边缘的（最大）数量
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向

名称	类型	描述
bAlternateDirection	BOOL	如果为 true，eEdgeDirection 会在每次检测到边缘后交替出现。否则，仅搜索具有 eEdgeDirection 的边缘。
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
fSearchLineDist	REAL	搜索线之间的距离，以像素为单位（> 0）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
fRangeCutoffFactor	REAL	用于剔除参与拟合的像素范围的相对边缘强度系数。负数表示包含噪声，即另一个方向上的强度差异。每个方向最多使用 nMaxThickness 个像素。
ipEdgeStrengths	引用 ITcVnContainer [▶ 365]	返回检测到的边缘强度（可选，如果不需要，则设为 0；ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_REAL，取决于 bAvgEdgeStrengths）
bAvgEdgeStrengths	BOOL	如果为 true，ipEdgeStrengths 将返回每条边缘的平均强度，而不是每条搜索线的单独强度
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aStartPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的起始位置（沿 aEndPoint 方向）
aEndPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的结束位置

输出

名称	类型	描述
fAvgStrength	REAL	返回检测到的边缘的平均强度

返回值

HRESULT [▶ 135]

更多信息

F_VN_LocateEdgesExp2 函数是 F_VN_LocateEdges [▶ 1476] 的专家版本。它包含额外的参数。

参数

输入图像

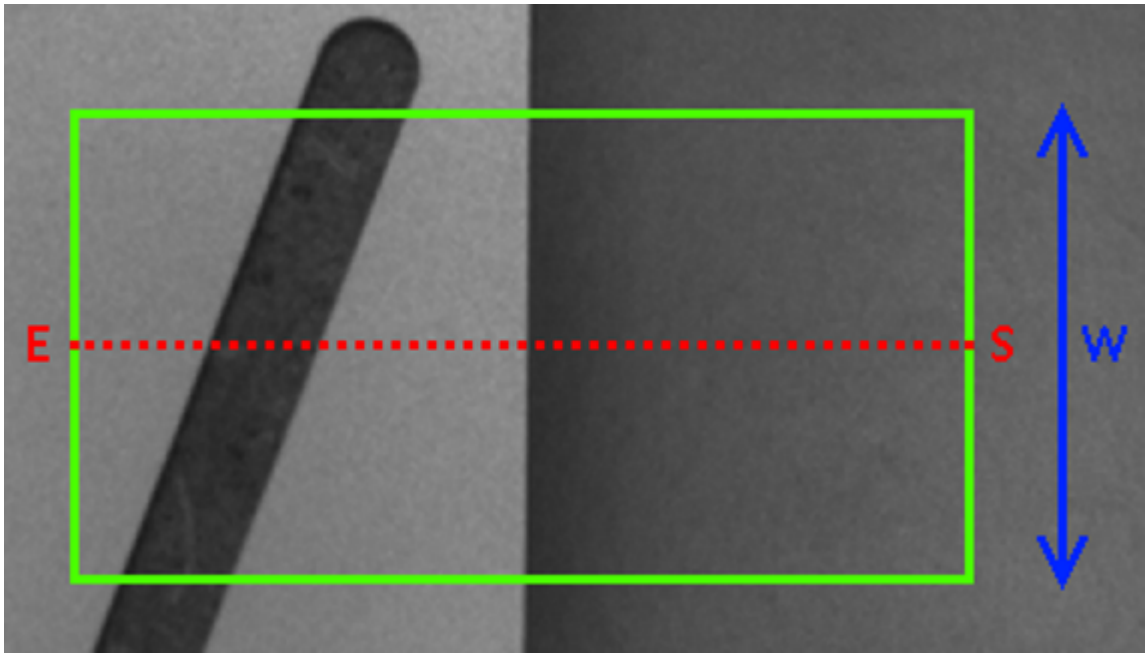
`ipSrcImage` 输入图像必须是单通道灰度图像。

定位点 (返回值)

ipEdgePoints 容器返回线性边缘上定位到的点。

搜索窗口

首先，必须定义一个搜索窗口，以确定在哪个位置以及沿哪个方向搜索边缘。这是通过 aStartPoint、aEndPoint 和 nSearchLines 参数实现的。搜索窗口由从 aStartPoint 到 aEndPoint 的搜索线及其两侧平行线构成，每个方向各有（nSearchLines - 1）/2 条搜索线，相邻搜索线的距离为 1 个像素。因此，nSearchLines 定义了搜索线的数量，且该值必须始终为奇数。



附图 18: S：起点，E：终点，W：窗口宽度

边缘强度 (返回值)

fAvgStrength 返回值显示定位到的边缘的实际对比度（对所有搜索线取平均值）。

边缘定位的参数

关于其余参数的详细解释，请参见边缘定位一章。

边缘厚度 (返回值)

检测到的边缘厚度通过 ipEdgeStrengths 返回。根据 bAvgEdgeStrengths，要么返回每条边缘的平均厚度（容器类型为 ContainerType_Vector_REAL），要么返回每条搜索线的单独厚度（容器类型为 ContainerType_Vector_Vector_REAL）。该参数为可选参数，如果不需要，可设置为 0。

应用

例如，使用专家参数对线性边缘进行定位的过程是这样的：

```
hr := F_VN_LocateEdgesExp2(
    ipSrcImage              := ipImageIn,
    ipEdgePoints            := ipEdgePoints,
    aStartPoint             := aStartPoint,
    aEndPoint               := aEndPoint,
    nNumEdges               := nNumEdges,
    eEdgeDirection          := eDirection,
    bAlternateDirection     := bAlternateDirection,
    fMinStrength            := fMinStrength,
    nSearchLines            := nSearchLines,
    fSearchLineDist         := fSearchLineDist,
    nMaxThickness           := nMaxThickness,
    nSubpixelsIterations    := nSubpixIter,
    fApproxPrecision        := 0.001,
    eAlgorithm              := eAlgorithm,
 fRangeCutoffFactor    := 0.135,
 ipEdgeStrengths       := ipEdgeStrengths,
```

```
bAvgEdgeStrengths    := bAvgEdgeStrengths,
   hrPrev                  := hr,
   fAvgStrength            => fAvgStrength);
```

相关函数

- LocateCircularArc [▶ 1464]，用于定位比例圆弧
- LocateEdge [▶ 1470]，用于定位线性边缘
- LocateEdges [▶ 1476]，用于定位几个线性边缘
- LocateEllipse [▶ 1485]，用于定位完整椭圆

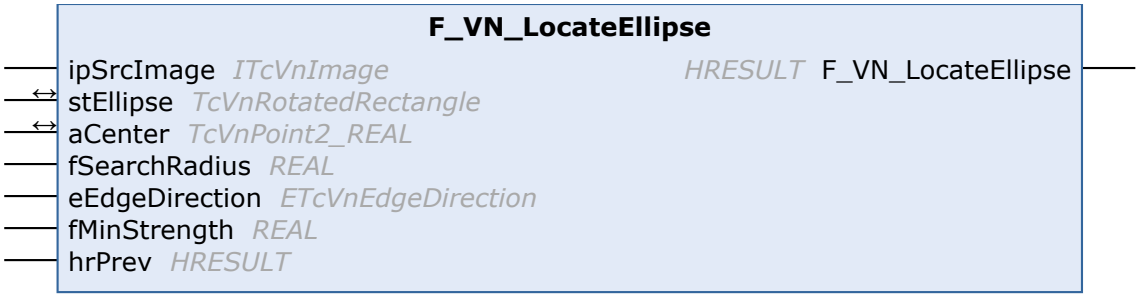
所需的许可证

TC3 Vision Metrology 2D

还请参阅有关此

F_VN_LocateAxisAlignedEdges [▶ 1457]

6.1.4.21.14 F_VN_LocateEllipse



使用定位边缘的插值法定位一个椭圆。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_LocateEllipse : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
END_VAR
VAR_IN_OUT
    stEllipse       : TcVnRotatedRectangle;
    aCenter         : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    fSearchRadius   : REAL;
    eEdgeDirection  : ETcVnEdgeDirection;
    fMinStrength    : REAL;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
fSearchRadius	REAL	围绕 aCenter 搜索边缘的半径（aCenter + fSearchRadius 的所有方向都必须在图像边界内！）
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向（从中心到椭圆外侧）
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stEllipse	TcVnRotatedRectangle [▶ 237]	返回检测到的椭圆
aCenter	TcVnPoint2_REAL [▶ 152]	预期的椭圆中心

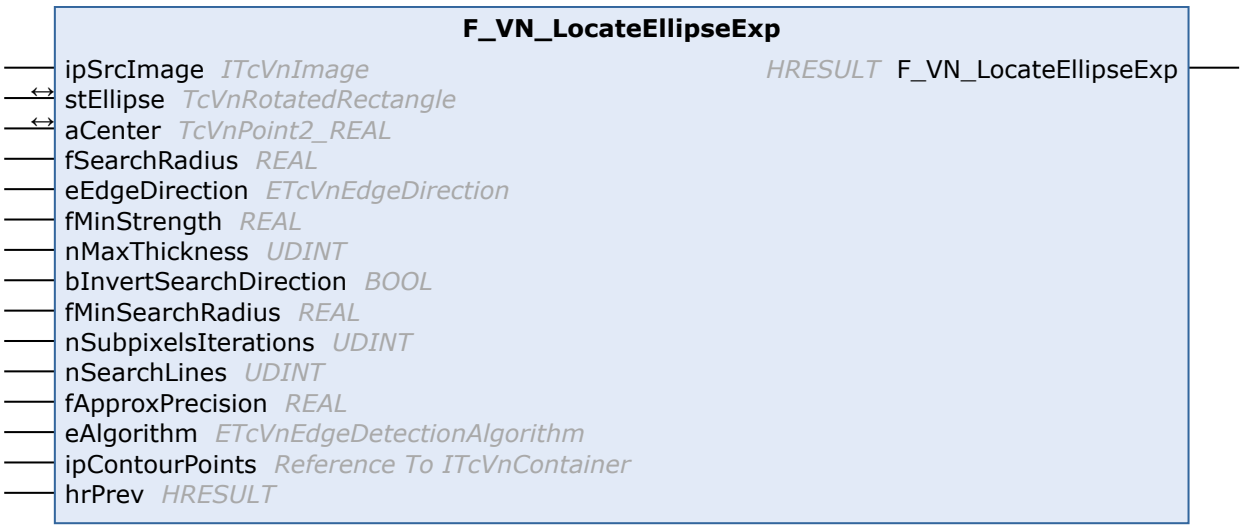
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.15 F_VN_LocateEllipseExp



定位一个椭圆。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_LocateEllipseExp : HRESULT
VAR_INPUT
    ipSrcImage              : ITcVnImage;
END_VAR
VAR_IN_OUT
    stEllipse               : TcVnRotatedRectangle;
    aCenter                 : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    fSearchRadius           : REAL;
    eEdgeDirection          : ETcVnEdgeDirection;
    fMinStrength            : REAL;
    nMaxThickness           : UDINT;
    bInvertSearchDirection  : BOOL;
    fMinSearchRadius        : REAL;
    nSubpixelsIterations    : UDINT;
    nSearchLines            : UDINT;
    fApproxPrecision        : REAL;
    eAlgorithm              : ETcVnEdgeDetectionAlgorithm;
```

```
    ipContourPoints         : Reference To ITcVnContainer;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
fSearchRadius	REAL	围绕 aCenter 搜索边缘的半径
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向（从中心到椭圆外侧，或如果 bInvertSearchDirection 为 true，则从椭圆外侧到中心）
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
bInvertSearchDirection	BOOL	如果为 true，则搜索从椭圆外沿中心方向开始
fMinSearchRadius	REAL	在开始搜索边缘之前要跳过的 aCenter 周围半径（例如，为了节省时间或中心包含应忽略的边缘）
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
nSearchLines	UDINT	指定搜索线的数量，在所有方向上平均分配（必须 >= 8 且是 4 的倍数）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
ipContourPoints	引用 ITcVnContainer [▶ 365]	返回亚像素精确椭圆轮廓（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL；重复使用非零接口指针。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stEllipse	TcVnRotatedRectangle [▶ 237]	返回检测到的椭圆
aCenter	TcVnPoint2_REAL [▶ 152]	预期的椭圆中心

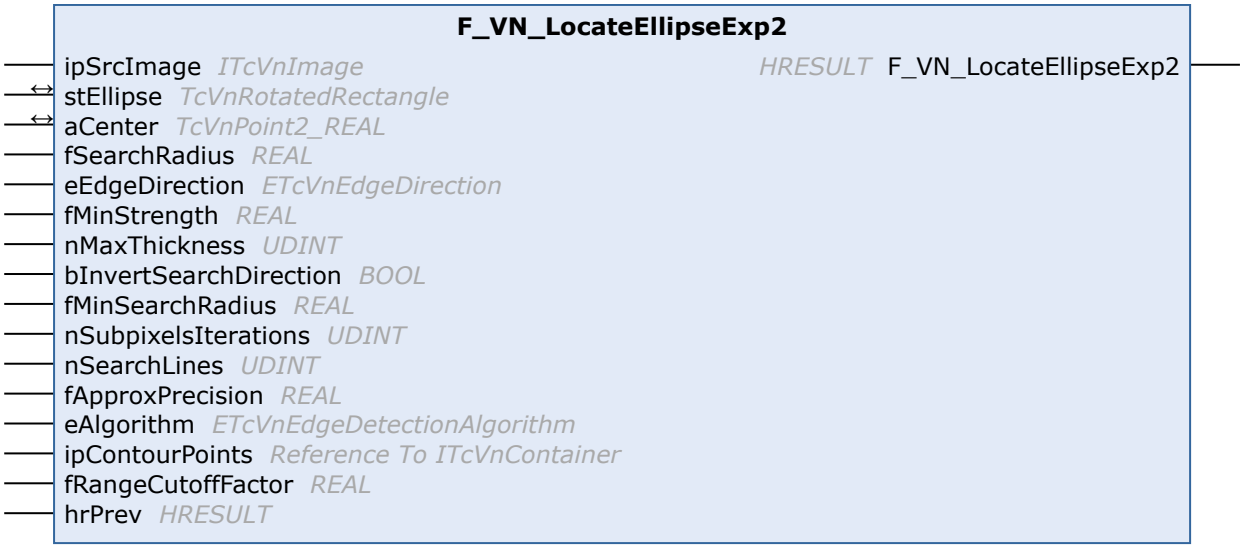
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.16 F_VN_LocateEllipseExp2



定位一个椭圆。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_LocateEllipseExp2 : HRESULT
VAR_INPUT
    ipSrcImage              : ITcVnImage;
END_VAR
VAR_IN_OUT
    stEllipse               : TcVnRotatedRectangle;
    aCenter                 : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    fSearchRadius           : REAL;
    eEdgeDirection          : ETcVnEdgeDirection;
    fMinStrength            : REAL;
    nMaxThickness           : UDINT;
    bInvertSearchDirection  : BOOL;
    fMinSearchRadius        : REAL;
    nSubpixelsIterations    : UDINT;
    nSearchLines            : UDINT;
    fApproxPrecision        : REAL;
    eAlgorithm              : ETcVnEdgeDetectionAlgorithm;
    ipContourPoints         : Reference To ITcVnContainer;
    fRangeCutoffFactor      : REAL;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
fSearchRadius	REAL	围绕 aCenter 搜索边缘的半径
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的边缘方向（从中心到椭圆外侧，或如果 bInvertSearchDirection 为 true，则从椭圆外侧到中心）
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
bInvertSearchDirection	BOOL	如果为 true，则搜索从椭圆外沿中心方向开始
fMinSearchRadius	REAL	在开始搜索边缘之前要跳过的 aCenter 周围半径（例如，为了节省时间或中心包含应忽略的边缘）

名称	类型	描述
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
nSearchLines	UDINT	指定搜索线的数量，在所有方向上平均分配（必须 >= 8 且是 4 的倍数）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
ipContourPoints	引用 ITcVnContainer [▶ 365]	返回亚像素精确椭圆轮廓（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL；重复使用非零接口指针。）
fRangeCutoffFactor	REAL	用于剔除参与拟合的像素范围的相对边缘强度系数。负数表示包含噪声，即另一个方向上的强度差异。每个方向最多使用 nMaxThickness 个像素。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stEllipse	TcVnRotatedRectangle [▶ 237]	返回检测到的椭圆
aCenter	TcVnPoint2_REAL [▶ 152]	预期的椭圆中心

返回值

HRESULT [▶ 135]

更多信息

F_VN_LocateEllipseExp2 函数是 F_VN_LocateEllipse [▶ 1485] 的专家版本。它包含额外的参数。

参数

输入图像

`ipSrcImage` 输入图像必须是单通道灰度图像。

椭圆 (返回值)

`stEllipse` 参数以 TcVnRotatedRectangle [▶ 237]（中心点、尺寸和角度）类型的结构返回定位到的椭圆。

搜索圆形

圆形搜索区域通过预期的 `aCenter` 中心点和 `fSearchRadius` 搜索半径来定义。搜索半径应设置得比预期的椭圆半径略大一些，以确保能够可靠地定位到椭圆。

定位点 (Expert（专家），返回值)

`ipContourPoints` 容器可返回定位到的点。

如果不需要返回边缘点，可将该参数设置为 0。

边缘定位的参数

关于其余参数的详细解释，请参见边缘定位一章。

应用

例如，使用专家参数对椭圆进行定位和测量的过程是这样的：

```
hr := F_VN_LocateEllipseExp2(
    ipSrcImage              := ipImageIn,
    stEllipse               := stEllipse,
    aCenter                 := aGuessedCenter,
    fSearchRadius           := (fGuessedRadius + 15), // increase search radius to make sure ellipse
is found!
    eEdgeDirection          := TCVN_ED_LIGHT_TO_DARK,
    fMinStrength            := fMinStrength,
    nMaxThickness           := 7,
    bInvertSearchDirection  := TRUE,
    fMinSearchRadius        := fMinRadius,
    nSubpixelsIterations    := nSubpixIter,
    nSearchLines            := nSearchLines,
    fApproxPrecision        := 0.001,
    eAlgorithm              := eAlgorithm,
    ipContourPoints         := ipContourPoints,
  fRangeCutoffFactor    := 0.135,
    hrPrev                  := hr);
```

使用 F_VN_DrawContours [▶ 1012] 函数可以将椭圆上定位到的点可视化：

```
hr := F_VN_DrawContours(ipContourPoints, -1, ipImageRes, aColorGreen, 5, hr);
```

示例

- 定位椭圆 [▶ 2802]

相关函数

- LocateCircularArc [▶ 1464]，用于定位比例圆弧
- LocateEdge [▶ 1470]，用于定位线性边缘
- LocateEdges [▶ 1476]，用于定位几个线性边缘
- LocateEllipse [▶ 1485]，用于定位完整椭圆

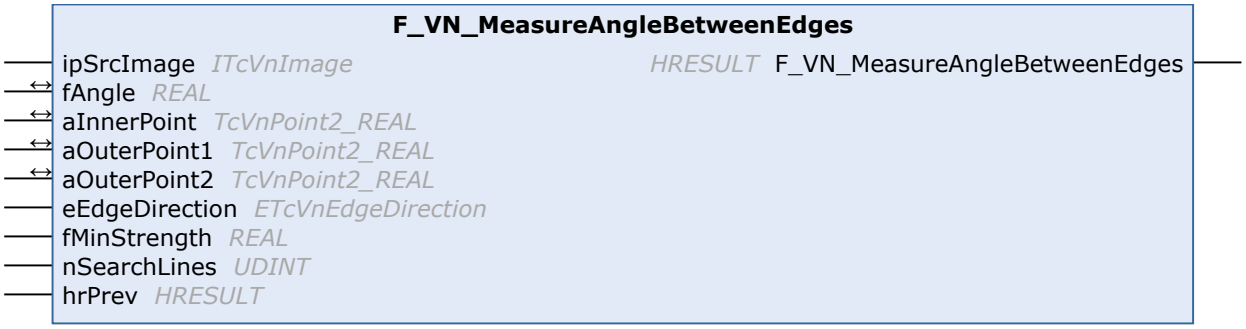
所需的许可证

TC3 Vision Metrology 2D

还请参阅有关此

F_VN_LocateAxisAlignedEdges [▶ 1457]

6.1.4.21.17 F_VN_MeasureAngleBetweenEdges



使用插值法测量 2 个边缘之间的角度。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_MeasureAngleBetweenEdges : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
```

```
END_VAR
VAR_IN_OUT
    fAngle          : REAL;
    aInnerPoint     : TcVnPoint2_REAL;
    aOuterPoint1    : TcVnPoint2_REAL;
    aOuterPoint2    : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    eEdgeDirection : ETcVnEdgeDirection;
    fMinStrength   : REAL;
    nSearchLines   : UDINT;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的从 aInnerPoint 到 aOuterPoint1 和 aOuterPoint2 的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（3、5、7.....），以 aInnerPoint 和 aOuterPoint1、aOuterPoint2 指定的行为中心
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fAngle	REAL	返回检测到的边缘之间的角度（弧度）
aInnerPoint	TcVnPoint2_REAL [▶ 152]	要测量的角内位置，从该位置开始边缘搜索过程（沿 aOuterPoint1 和 aOuterPoint2 方向）
aOuterPoint1	TcVnPoint2_REAL [▶ 152]	边缘 1 搜索过程的结束位置
aOuterPoint2	TcVnPoint2_REAL [▶ 152]	边缘 2 搜索过程的结束位置

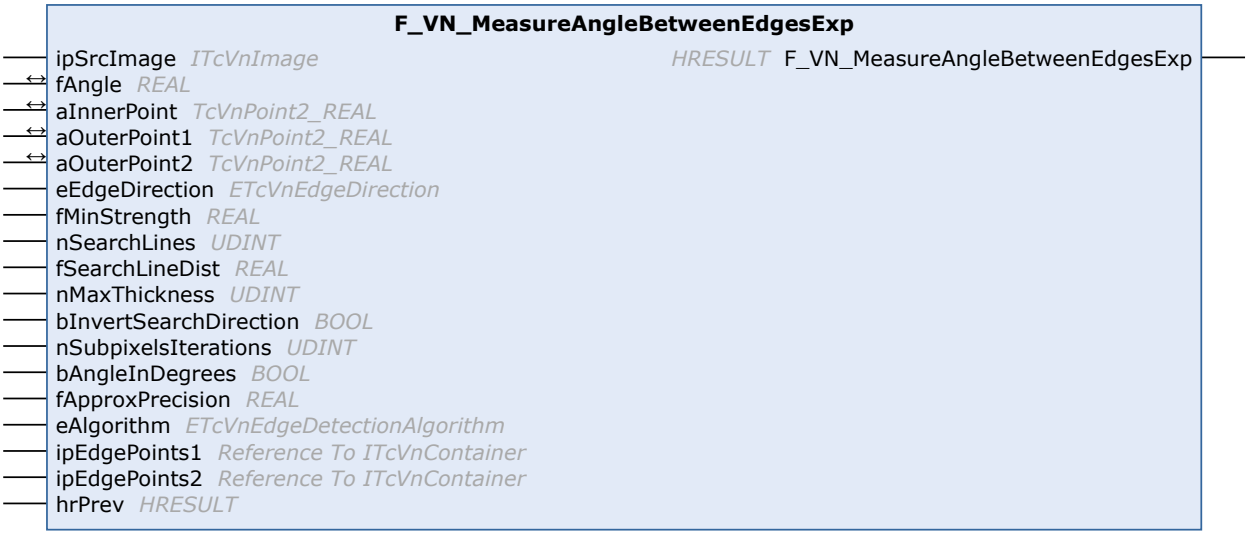
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.18 F_VN_MeasureAngleBetweenEdgesExp



测量 2 个边缘之间的角度。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_MeasureAngleBetweenEdgesExp : HRESULT
VAR_INPUT
    ipSrcImage              : ITcVnImage;
END_VAR
VAR_IN_OUT
    fAngle                  : REAL;
    aInnerPoint             : TcVnPoint2_REAL;
    aOuterPoint1            : TcVnPoint2_REAL;
    aOuterPoint2            : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    eEdgeDirection          : ETcVnEdgeDirection;
    fMinStrength            : REAL;
    nSearchLines            : UDINT;
    fSearchLineDist         : REAL;
    nMaxThickness           : UDINT;
    bInvertSearchDirection  : BOOL;
    nSubpixelsIterations    : UDINT;
    bAngleInDegrees         : BOOL;
    fApproxPrecision        : REAL;
    eAlgorithm              : ETcVnEdgeDetectionAlgorithm;
    ipEdgePoints1           : Reference To ITcVnContainer;
    ipEdgePoints2           : Reference To ITcVnContainer;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的从 aInnerPoint 到 aOuterPoint1 和 aOuterPoint2 的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（3、5、7.....），以 aInnerPoint 和 aOuterPoint1、aOuterPoint2 指定的行为中心
fSearchLineDist	REAL	搜索线之间的距离，以像素为单位（> 0）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength

名称	类型	描述
bInvertSearchDirection	BOOL	如果为 true，则从每个 aOuterPoint 开始沿 aInnerPoint 方向搜索
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
bAngleInDegrees	BOOL	如果为 true，fAngle 以度为单位
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
ipEdgePoints1	引用 ITcVnContainer [▶ 365]	返回在 aInnerPoint 和 aOuterPoint1 之间检测到的边缘点（ContainerType_Vector_TcVnPoint2_REAL）
ipEdgePoints2	引用 ITcVnContainer [▶ 365]	返回在 aInnerPoint 和 aOuterPoint2 之间检测到的边缘点（ContainerType_Vector_TcVnPoint2_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fAngle	REAL	返回检测到的边缘之间的角度（弧度或度，取决于 bAngleInDegrees）
aInnerPoint	TcVnPoint2_REAL [▶ 152]	要测量的角内位置，从该位置开始边缘搜索过程（沿 aOuterPoint1 和 aOuterPoint2 方向）
aOuterPoint1	TcVnPoint2_REAL [▶ 152]	边缘 1 搜索过程的结束位置
aOuterPoint2	TcVnPoint2_REAL [▶ 152]	边缘 2 搜索过程的结束位置

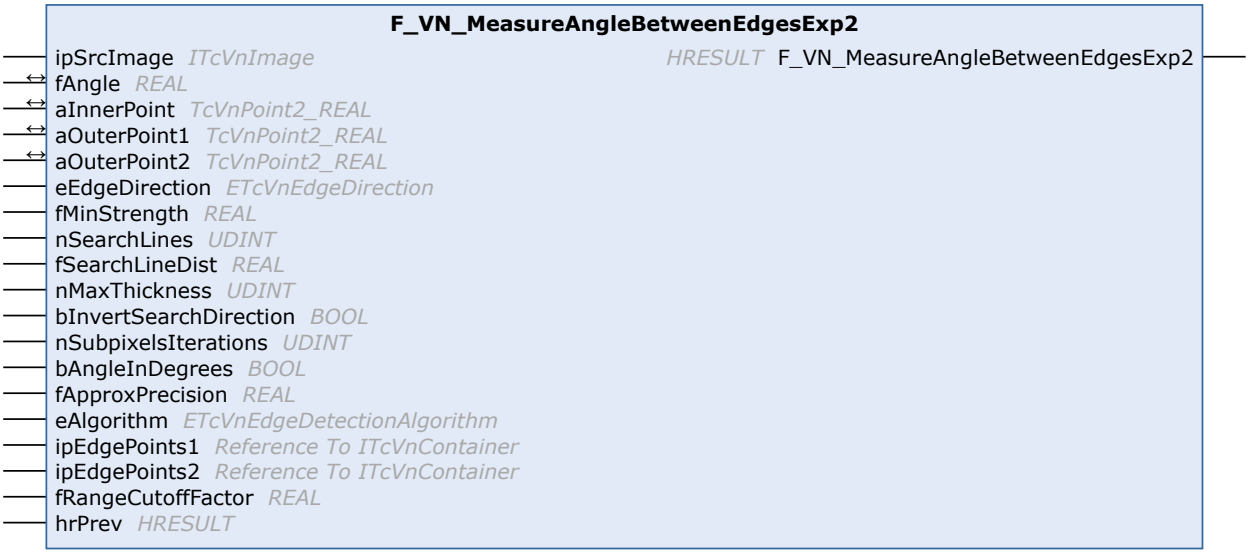
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.19 F_VN_MeasureAngleBetweenEdgesExp2



测量 2 个边缘之间的角度。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_MeasureAngleBetweenEdgesExp2 : HRESULT
VAR_INPUT
    ipSrcImage              : ITcVnImage;
END_VAR
VAR_IN_OUT
    fAngle                  : REAL;
    aInnerPoint             : TcVnPoint2_REAL;
    aOuterPoint1            : TcVnPoint2_REAL;
    aOuterPoint2            : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    eEdgeDirection          : ETcVnEdgeDirection;
    fMinStrength            : REAL;
    nSearchLines            : UDINT;
    fSearchLineDist         : REAL;
    nMaxThickness           : UDINT;
    bInvertSearchDirection  : BOOL;
    nSubpixelsIterations    : UDINT;
    bAngleInDegrees         : BOOL;
    fApproxPrecision        : REAL;
    eAlgorithm              : ETcVnEdgeDetectionAlgorithm;
    ipEdgePoints1           : Reference To ITcVnContainer;
    ipEdgePoints2           : Reference To ITcVnContainer;
    fRangeCutoffFactor      : REAL;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的从 aInnerPoint 到 aOuterPoint1 和 aOuterPoint2 的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（3、5、7.....），以 aInnerPoint 和 aOuterPoint1、aOuterPoint2 指定的行为中心
fSearchLineDist	REAL	搜索线之间的距离，以像素为单位（> 0）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength

名称	类型	描述
bInvertSearchDirection	BOOL	如果为 true，则从每个 aOuterPoint 开始沿 aInnerPoint 方向搜索
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
bAngleInDegrees	BOOL	如果为 true，fAngle 以度为单位
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
ipEdgePoints1	引用 ITcVnContainer [▶ 365]	返回在 aInnerPoint 和 aOuterPoint1 之间检测到的边缘点（ContainerType_Vector_TcVnPoint2_REAL）
ipEdgePoints2	引用 ITcVnContainer [▶ 365]	返回在 aInnerPoint 和 aOuterPoint2 之间检测到的边缘点（ContainerType_Vector_TcVnPoint2_REAL）
fRangeCutoffFactor	REAL	用于剔除参与拟合的像素范围的相对边缘强度系数。负数表示包含噪声，即另一个方向上的强度差异。每个方向最多使用 nMaxThickness 个像素。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fAngle	REAL	返回检测到的边缘之间的角度（弧度或度，取决于 bAngleInDegrees）
aInnerPoint	TcVnPoint2_REAL [▶ 152]	要测量的角内位置，从该位置开始边缘搜索过程（沿 aOuterPoint1 和 aOuterPoint2 方向）
aOuterPoint1	TcVnPoint2_REAL [▶ 152]	边缘 1 搜索过程的结束位置
aOuterPoint2	TcVnPoint2_REAL [▶ 152]	边缘 2 搜索过程的结束位置

返回值

HRESULT [▶ 135]

更多信息

F_VN_MeasureAngleBetweenEdgesExp2 函数是 F_VN_MeasureAngleBetweenEdges [▶ 1490] 的专家版本。

参数

输入图像

`ipSrcImage` 输入图像必须是单通道灰度图像。

角度 (返回值)

`fAngle` 参数可返回定位到的边缘之间的角度。

根据 `bAngleInDegrees` 参数，`fAngle` 角度以弧度（`FALSE`）或度（`TRUE`）为单位进行指定。

位置信息

为了定位边缘，必须首先指定共同起点（aInnerPoint）和两个目标点（aOuterPoint1，aOuterPoint2）。起点的位置必须位于要测量的角度范围内，边缘搜索过程将从该点开始。搜索过程在相应的目标点处结束。

边缘定位的参数

关于其余参数的详细解释，请参见边缘定位一章。

定位点 （Expert（专家），返回值）

ipEdgePoints1 和 ipEdgePoints2 容器会返回根据每条搜索线定位的精确点。如果不需要，可以传输值 0，而不是接口指针

应用

例如，使用专家参数测量角度的过程是这样的：

```
hr := F_VN_MeasureAngleBetweenEdgesExp2(
    ipSrcImage              := ipImageIn,
    fAngle                  := fAngle,
    aInnerPoint             := aInnerPoint,
    aOuterPoint1            := aOuterPoint1,
    aOuterPoint2            := aOuterPoint2,
    eEdgeDirection          := TCVN_ED_LIGHT_TO_DARK,
    fMinStrength            := fMinStrength,
    nSearchLines            := nSearchLines,
    fSearchLineDist         := fSearchLineDist,
    nMaxThickness           := 7,
    bInvertSearchDirection  := False,
    nSubpixelsIterations    := 10,
    bAngleInDegrees         := TRUE,
    fApproxPrecision        := 0.001,
    eAlgorithm              := TCVN_EDA_INTERPOLATION,
    ipEdgePoints1           := ipEdgePoints1,
    ipEdgePoints2           := ipEdgePoints2,
 fRangeCutoffFactor   := 0.135,
    hrPrev                  := hr);
```

示例

- 测量边缘之间的角度 [▶ 2807]

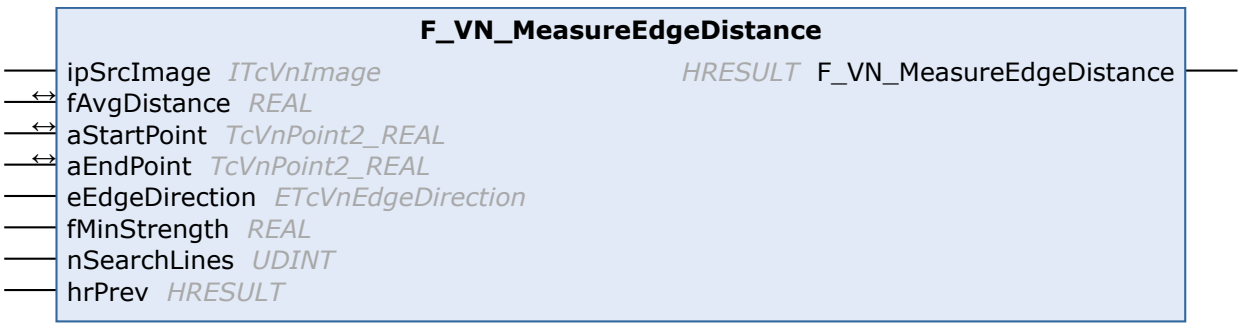
相关函数

- ClosestPointsBF [▶ 1462]，用于计算两个点云中最近的点
- MeasureAngleBetweenEdges [▶ 1490]，用于测量两条边之间的角度
- MeasureEdgeDistance [▶ 1496]，用于测量两条边之间的平均距离
- MeasureMinEdgeDistance [▶ 1505]，用于测量两条边之间的最小距离

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.20 F_VN_MeasureEdgeDistance



使用插值法测量两条平行边缘之间的距离。aStartPoint 和 aEndPoint 之间的中点必须位于两条边缘之间。
可以使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_MeasureEdgeDistance : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
END_VAR
VAR_IN_OUT
    fAvgDistance    : REAL;
    aStartPoint     : TcVnPoint2_REAL;
    aEndPoint       : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    eEdgeDirection : ETcVnEdgeDirection;
    fMinStrength    : REAL;
    nSearchLines    : UDINT;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的从 aStartPoint 到 aEndPoint 的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fAvgDistance	REAL	返回检测到的边缘之间的平均距离
aStartPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的起始位置（沿 aEndPoint 方向）
aEndPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的结束位置

返回值

HRESULT [▶ 135]

更多信息

`F_VN_MeasureEdgeDistance` 函数可定位两条线性边缘，并测量它们之间的平均距离。

参数

输入图像

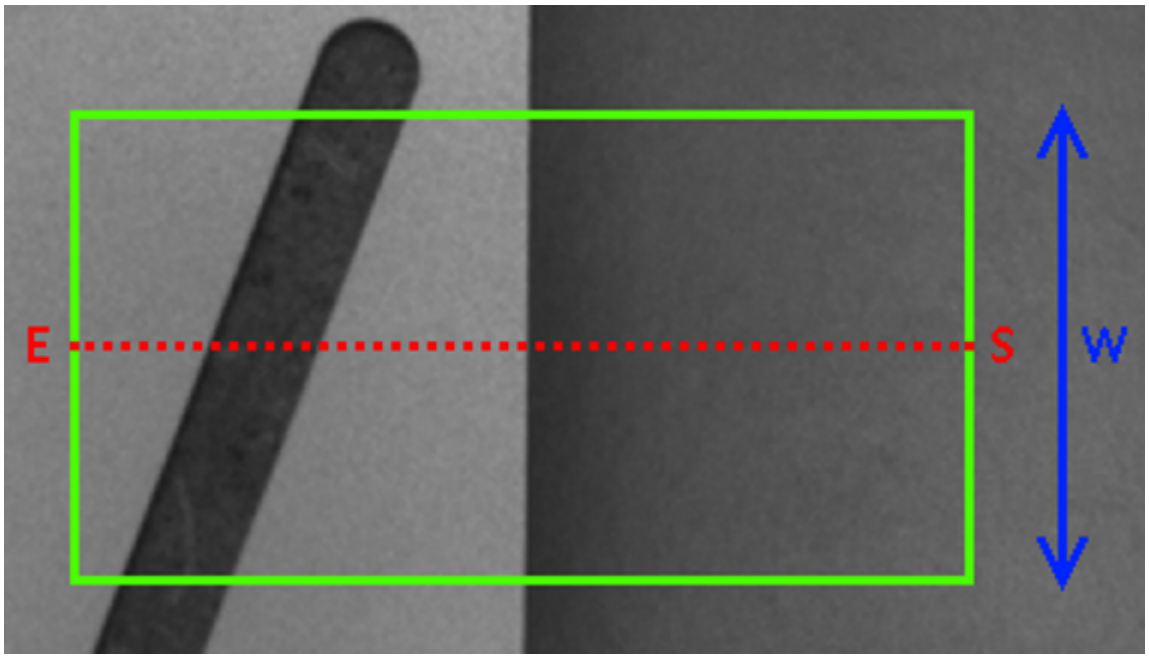
`ipSrcImage` 输入图像必须是单通道灰度图像。

平均距离 (返回值)

`fAvgDistance` 参数可返回两条定位到的边缘之间的平均距离。

搜索窗口

首先，必须定义一个搜索窗口，以确定在哪个位置以及沿哪个方向搜索边缘。这是通过 aStartPoint、aEndPoint 和 nSearchLines 参数实现的。搜索窗口由从 aStartPoint 到 aEndPoint 的搜索线及其两侧平行线构成，每个方向各有（nSearchLines - 1）/2 条搜索线，相邻搜索线的距离为 1 个像素。因此，nSearchLines 定义了搜索线的数量，且该值必须始终为奇数。



附图 19: S：起点，E：终点，W：窗口宽度

边缘定位的参数

关于其余参数的详细解释，请参见边缘定位一章。

专家参数

F_VN_MeasureEdgeDistanceExp [▶ 1499] 专家版本包含附加参数。

应用

例如，测量边缘距离的过程是这样的：

```
hr := F_VN_MeasureEdgeDistance(
    ipSrcImage          := ipImageIn,
    fAvgDistance        := fAvgDistance,
    aStartPoint         := aStartPoint,
    aEndPoint           := aEndPoint,
    eEdgeDirection      := TCVN_ED_LIGHT_TO_DARK,
    fMinStrength        := fMinStrength,
    nSearchLines        := nSearchLines,
    hrPrev              := hr);
```

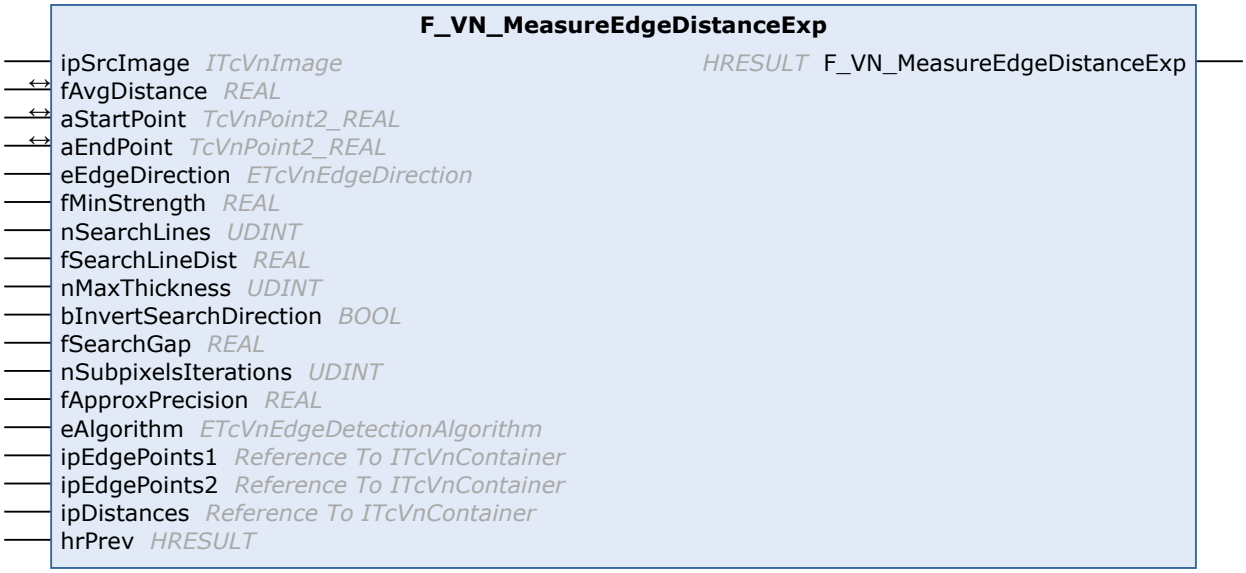
相关函数

- ClosestPointsBF [▶ 1462]，用于计算两个点云中最近的点
- MeasureAngleBetweenEdges [▶ 1490]，用于测量两条边之间的角度
- MeasureEdgeDistance [▶ 1496]，用于测量两条边之间的平均距离
- MeasureMinEdgeDistance [▶ 1505]，用于测量两条边之间的最小距离

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.21 F_VN_MeasureEdgeDistanceExp



测量两条平行边缘之间的距离。aStartPoint 和 aEndPoint 之间的中点必须位于两条边缘之间。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_MeasureEdgeDistanceExp : HRESULT
VAR_INPUT
    ipSrcImage             : ITcVnImage;
END_VAR
VAR_IN_OUT
    fAvgDistance           : REAL;
    aStartPoint            : TcVnPoint2_REAL;
    aEndPoint              : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    eEdgeDirection         : ETcVnEdgeDirection;
    fMinStrength           : REAL;
    nSearchLines           : UDINT;
    fSearchLineDist        : REAL;
    nMaxThickness          : UDINT;
    bInvertSearchDirection : BOOL;
    fSearchGap             : REAL;
    nSubpixelsIterations   : UDINT;
    fApproxPrecision       : REAL;
    eAlgorithm             : ETcVnEdgeDetectionAlgorithm;
    ipEdgePoints1          : Reference To ITcVnContainer;
    ipEdgePoints2          : Reference To ITcVnContainer;
    ipDistances            : Reference To ITcVnContainer;
    hrPrev                 : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的从 aStartPoint 到 aEndPoint 的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
fSearchLineDist	REAL	搜索线之间的距离，以像素为单位（> 0）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength

名称	类型	描述
bInvertSearchDirection	BOOL	如果为 true，则搜索从 aStartPoint 和 aEndPoint 之间的中心点沿两个方向开始
fSearchGap	REAL	可选的间隙宽度（>= 0，以 aStartPoint 和 aEndPoint 之间为中心），搜索边缘时忽略该间隙（可缩短执行时间）。要搜索的 2 个边缘必须位于间隙的不同侧。
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
ipEdgePoints1	引用 ITcVnContainer [▶ 365]	返回检测到的靠近 aStartPoint 的边缘点（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL）
ipEdgePoints2	引用 ITcVnContainer [▶ 365]	返回检测到的靠近 aEndPoint 的边缘点（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL）
ipDistances	引用 ITcVnContainer [▶ 365]	返回检测到的边缘点之间的距离（可选，如果不需要，则设为 0；ContainerType_Vector_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fAvgDistance	REAL	返回检测到的边缘之间的平均距离
aStartPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的起始位置（沿 aEndPoint 方向）
aEndPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的结束位置

返回值

HRESULT [▶ 135]

更多信息

`F_VN_MeasureEdgeDistanceExp` 函数是 F_VN_MeasureEdgeDistance [▶ 1496] 的专家版本。它包含额外的参数。

参数

输入图像

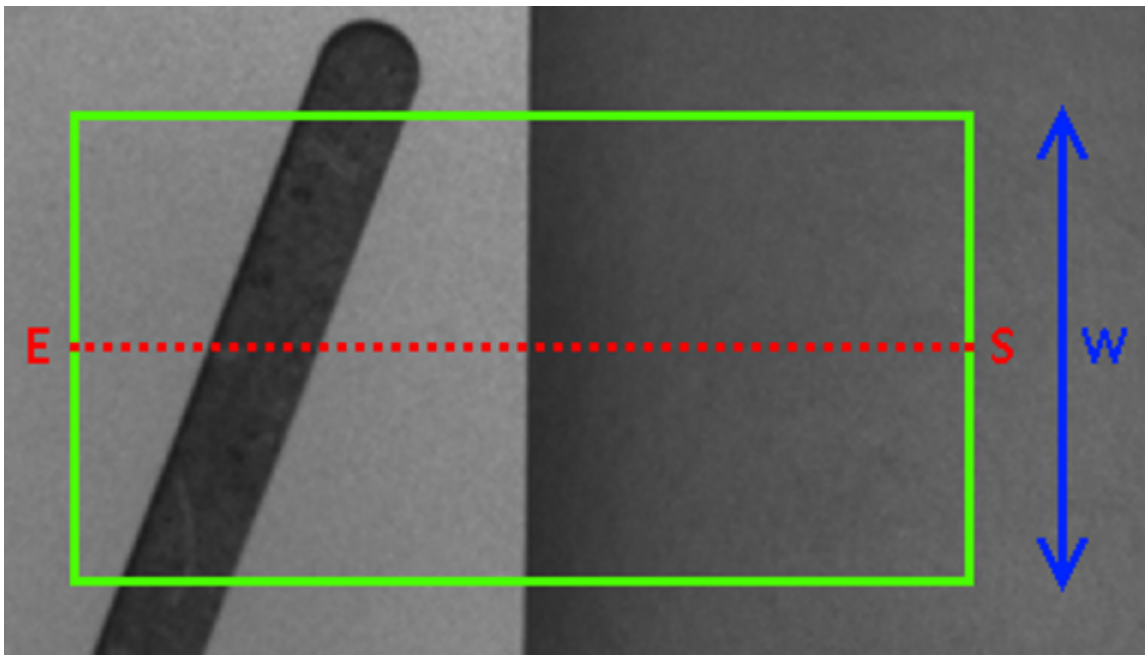
`ipSrcImage` 输入图像必须是单通道灰度图像。

平均距离 (返回值)

`fAvgDistance` 参数可返回两条定位到的边缘之间的平均距离。

搜索窗口

首先，必须定义一个搜索窗口，以确定在哪个位置以及沿哪个方向搜索边缘。这是通过 `aStartPoint`、`aEndPoint` 和 `nSearchLines` 参数实现的。搜索窗口由从 `aStartPoint` 到 `aEndPoint` 的搜索线及其两侧平行线构成，每个方向各有（`nSearchLines` - 1）/2 条搜索线，相邻搜索线的距离为 1 个像素。因此，`nSearchLines` 定义了搜索线的数量，且该值必须始终为奇数。



附图 20: S：起点，E：终点，W：窗口宽度

定位边缘点 (Expert（专家），返回值)

`ipEdgePoints1` 和 `ipEdgePoints2` 容器会返回根据每条搜索线定位的精确点。如果不需要，可以传输值 `0`，而不是接口指针。

测量距离 (Expert（专家），返回值)

`ipDistances` 容器可返回所有相对点之间的测得距离。如果不需要，可以传输值 `0`，而不是接口指针。

搜索差距 (Expert（专家）)

通过 `fSearchGap`，您可以指定在已定义搜索线的中间区域有一个在搜索过程中被忽略或跳过的区域。这可以缩短执行时间。

边缘定位的参数

关于其余参数的详细解释，请参见边缘定位一章。

应用

例如，使用专家参数测量边缘距离的过程是这样的：

```
hr := F_VN_MeasureEdgeDistanceExp(
    ipSrcImage              := ipImageIn,
    fAvgDistance            := fAvgDistance,
    aStartPoint             := aStartPoint,
    aEndPoint               := aEndPoint,
    eEdgeDirection          := TCVN_ED_LIGHT_TO_DARK,
    fMinStrength            := fMinStrength,
    nSearchLines            := nSearchLines,
  fSearchLineDist     := 3,
  nMaxThickness       := 7,
  bInvertSearchDirection := TRUE,
  fSearchGap          := 100,
  nSubpixelsIterations   := nSubpixIter,
  fApproxPrecision     := 0.001,
  eAlgorithm          := eAlgorithm,
  ipEdgePoints1       := ipEdgePoints1,
  ipEdgePoints2       := ipEdgePoints2,
  ipDistances         := ipDistances,
    hrPrev                  := hr);
```

可以使用函数 F_VN_DrawContours [▶ 1012] 将局部边缘点可视化：

```
hr := F_VN_DrawContours(ipEdgePoints1, -1, ipImageRes, aColorGreen, 5, hr);
hr := F_VN_DrawContours(ipEdgePoints2, -1, ipImageRes, aColorGreen, 5, hr);
```

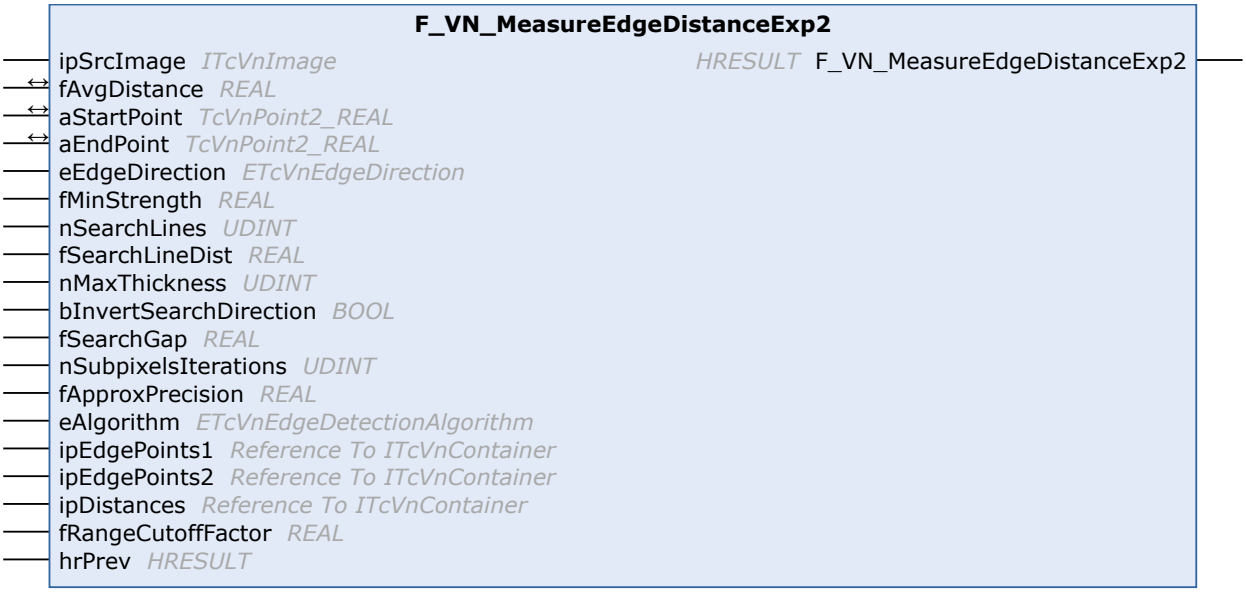
相关函数

- ClosestPointsBF [▶ 1462]，用于计算两个点云中最近的点
- MeasureAngleBetweenEdges [▶ 1490]，用于测量两条边之间的角度
- MeasureEdgeDistance [▶ 1496]，用于测量两条边之间的平均距离
- MeasureMinEdgeDistance [▶ 1505]，用于测量两条边之间的最小距离

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.22 F_VN_MeasureEdgeDistanceExp2



测量两条平行边缘之间的距离。aStartPoint 和 aEndPoint 之间的中点必须位于两条边缘之间。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_MeasureEdgeDistanceExp2 : HRESULT
VAR_INPUT
    ipSrcImage              : ITcVnImage;
END_VAR
VAR_IN_OUT
    fAvgDistance            : REAL;
    aStartPoint             : TcVnPoint2_REAL;
    aEndPoint               : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    eEdgeDirection          : ETcVnEdgeDirection;
    fMinStrength            : REAL;
    nSearchLines            : UDINT;
    fSearchLineDist         : REAL;
    nMaxThickness           : UDINT;
    bInvertSearchDirection  : BOOL;
    fSearchGap              : REAL;
    nSubpixelsIterations    : UDINT;
    fApproxPrecision        : REAL;
    eAlgorithm              : ETcVnEdgeDetectionAlgorithm;
    ipEdgePoints1           : Reference To ITcVnContainer;
    ipEdgePoints2           : Reference To ITcVnContainer;
    ipDistances             : Reference To ITcVnContainer;
    fRangeCutoffFactor      : REAL;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的从 aStartPoint 到 aEndPoint 的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
fSearchLineDist	REAL	搜索线之间的距离，以像素为单位（> 0）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
bInvertSearchDirection	BOOL	如果为 true，则搜索从 aStartPoint 和 aEndPoint 之间的中心点沿两个方向开始
fSearchGap	REAL	可选的间隙宽度（>= 0，以 aStartPoint 和 aEndPoint 之间为中心），搜索边缘时忽略该间隙（可缩短执行时间）。要搜索的 2 个边缘必须位于间隙的不同侧。
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
ipEdgePoints1	引用 ITcVnContainer [▶ 365]	返回检测到的靠近 aStartPoint 的边缘点（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL）
ipEdgePoints2	引用 ITcVnContainer [▶ 365]	返回检测到的靠近 aEndPoint 的边缘点（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL）
ipDistances	引用 ITcVnContainer [▶ 365]	返回检测到的边缘点之间的距离（可选，如果不需要，则设为 0；ContainerType_Vector_REAL）
fRangeCutoffFactor	REAL	用于剔除参与拟合的像素范围的相对边缘强度系数。负数表示包含噪声，即另一个方向上的强度差异。每个方向最多使用 nMaxThickness 个像素。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fAvgDistance	REAL	返回检测到的边缘之间的平均距离
aStartPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的起始位置（沿 aEndPoint 方向）
aEndPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的结束位置

返回值

HRESULT [▶ 135]

更多信息

F_VN_MeasureEdgeDistanceExp2 函数是 F_VN_MeasureEdgeDistance [▶ 1496] 的专家版本。它包含额外的参数。

参数

输入图像

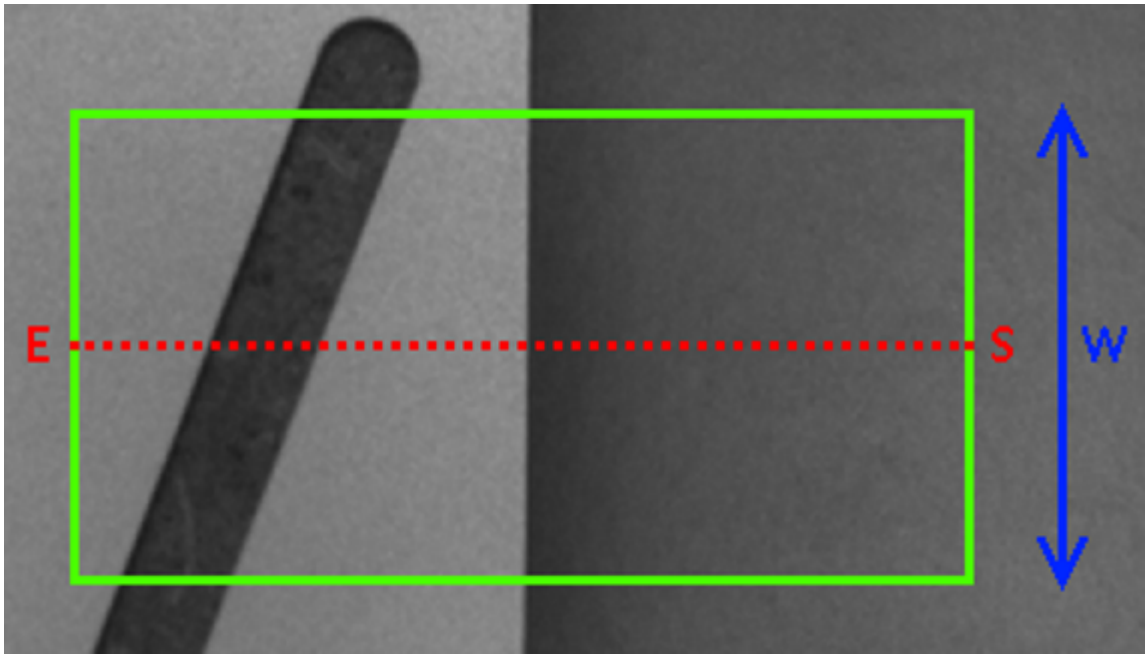
`ipSrcImage` 输入图像必须是单通道灰度图像。

平均距离 (返回值)

`fAvgDistance` 参数可返回两条定位到的边缘之间的平均距离。

搜索窗口

首先，必须定义一个搜索窗口，以确定在哪个位置以及沿哪个方向搜索边缘。这是通过 `aStartPoint`、`aEndPoint` 和 `nSearchLines` 参数实现的。搜索窗口由从 `aStartPoint` 到 `aEndPoint` 的搜索线及其两侧平行线构成，每个方向各有（`nSearchLines` - 1）/2 条搜索线，相邻搜索线的距离为 1 个像素。因此，`nSearchLines` 定义了搜索线的数量，且该值必须始终为奇数。



附图 21: S：起点，E：终点，W：窗口宽度

定位边缘点 (Expert（专家），返回值)

`ipEdgePoints1` 和 `ipEdgePoints2` 容器会返回根据每条搜索线定位的精确点。如果不需要，可以传输值 `0`，而不是接口指针。

测量距离 (Expert（专家），返回值)

`ipDistances` 容器可返回所有相对点之间的测得距离。如果不需要，可以传输值 `0`，而不是接口指针。

搜索差距 (Expert（专家）)

通过 `fSearchGap`，您可以指定在已定义搜索线的中间区域有一个在搜索过程中被忽略或跳过的区域。这可以缩短执行时间。

边缘定位的参数

关于其余参数的详细解释，请参见边缘定位一章。

应用

例如，使用专家参数测量边缘距离的过程是这样的：

```
hr := F_VN_MeasureEdgeDistanceExp2(
    ipSrcImage              := ipImageIn,
    fAvgDistance            := fAvgDistance,
    aStartPoint             := aStartPoint,
```

```
    aEndPoint               := aEndPoint,
    eEdgeDirection          := TCVN_ED_LIGHT_TO_DARK,
    fMinStrength            := fMinStrength,
    nSearchLines            := nSearchLines,
    fSearchLineDist         := 3,
    nMaxThickness           := 7,
    bInvertSearchDirection  := TRUE,
    fSearchGap              := 100,
    nSubpixelsIterations    := nSubpixIter,
    fApproxPrecision        := 0.001,
    eAlgorithm              := eAlgorithm,
    ipEdgePoints1           := ipEdgePoints1,
    ipEdgePoints2           := ipEdgePoints2,
    ipDistances             := ipDistances,
fRangeCutoffFactor    := 0.135,
    hrPrev                  := hr);
```

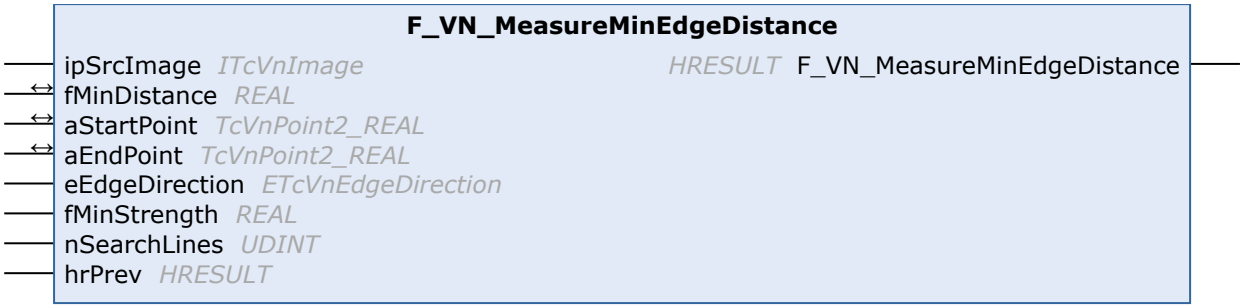
相关函数

- ClosestPointsBF [▶ 1462]，用于计算两个点云中最近的点
- MeasureAngleBetweenEdges [▶ 1490]，用于测量两条边之间的角度
- MeasureEdgeDistance [▶ 1496]，用于测量两条边之间的平均距离
- MeasureMinEdgeDistance [▶ 1505]，用于测量两条边之间的最小距离

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.23 F_VN_MeasureMinEdgeDistance



使用插值法，在指定搜索窗口内测量两条边缘之间的最小距离。aStartPoint 和 aEndPoint 之间的中点必须位于两条边缘之间。
可以使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_MeasureMinEdgeDistance : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
END_VAR
VAR_IN_OUT
    fMinDistance    : REAL;
    aStartPoint     : TcVnPoint2_REAL;
    aEndPoint       : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    eEdgeDirection : ETcVnEdgeDirection;
    fMinStrength   : REAL;
    nSearchLines   : UDINT;
    hrPrev         : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）

名称	类型	描述
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的从 aStartPoint 到 aEndPoint 的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fMinDistance	REAL	返回检测到的边缘之间的最小距离
aStartPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的起始位置（沿 aEndPoint 方向）
aEndPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的结束位置

返回值

HRESULT [▶ 135]

更多信息

`F_VN_MeasureMinEdgeDistance` 函数可定位两条线性边缘，并测量它们之间的最小距离。

参数

输入图像

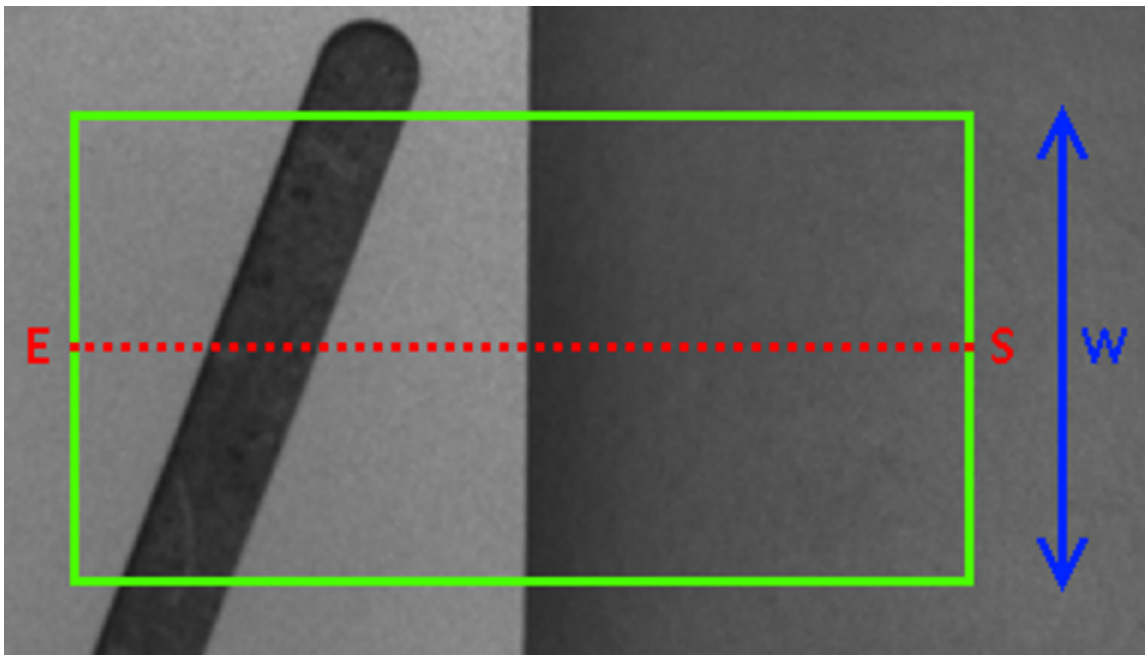
`ipSrcImage` 输入图像必须是单通道灰度图像。

最小距离 (返回值)

`fMinDistance` 参数可返回两条定位到的边缘之间的最小距离。

搜索窗口

首先，必须定义一个搜索窗口，以确定在哪个位置以及沿哪个方向搜索边缘。这是通过 `aStartPoint`、`aEndPoint` 和 `nSearchLines` 参数实现的。搜索窗口由从 `aStartPoint` 到 `aEndPoint` 的搜索线及其两侧平行线构成，每个方向各有（`nSearchLines` - 1）/2 条搜索线，相邻搜索线的距离为 1 个像素。因此，`nSearchLines` 定义了搜索线的数量，且该值必须始终为奇数。



附图 22: S：起点，E：终点，W：窗口宽度

边缘定位的参数

关于其余参数的详细解释，请参见边缘定位一章。

专家参数

F_VN_MeasureMinEdgeDistanceExp [▶ 1508] 专家版本包含附加参数。

应用

例如，测量边缘距离的过程是这样的：

```
hr := F_VN_MeasureMinEdgeDistance(
    ipSrcImage          := ipImageIn,
    fMinDistance        := fMinDistance,
    aStartPoint         := aStartPoint,
    aEndPoint           := aEndPoint,
    eEdgeDirection      := TCVN_ED_LIGHT_TO_DARK,
    fMinStrength        := fMinStrength,
    nSearchLines        := nSearchLines,
    hrPrev              := hr);
```

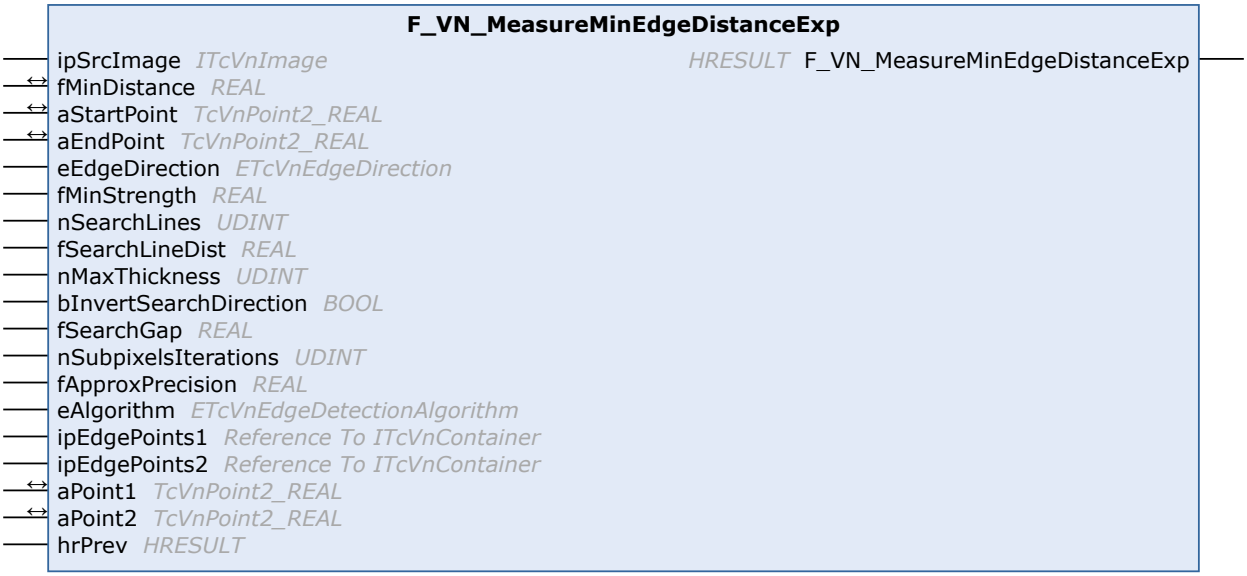
相关函数

- ClosestPointsBF [▶ 1462]，用于计算两个点云中最近的点
- MeasureAngleBetweenEdges [▶ 1490]，用于测量两条边之间的角度
- MeasureEdgeDistance [▶ 1496]，用于测量两条边之间的平均距离
- MeasureMinEdgeDistance [▶ 1505]，用于测量两条边之间的最小距离

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.24 F_VN_MeasureMinEdgeDistanceExp



在指定搜索窗口内测量两条边缘之间的最小距离。aStartPoint 和 aEndPoint 之间的中点必须位于两条边缘之间。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_MeasureMinEdgeDistanceExp : HRESULT
VAR_INPUT
    ipSrcImage              : ITcVnImage;
END_VAR
VAR_IN_OUT
    fMinDistance            : REAL;
    aStartPoint             : TcVnPoint2_REAL;
    aEndPoint               : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    eEdgeDirection          : ETcVnEdgeDirection;
    fMinStrength            : REAL;
    nSearchLines            : UDINT;
    fSearchLineDist         : REAL;
    nMaxThickness           : UDINT;
    bInvertSearchDirection  : BOOL;
    fSearchGap              : REAL;
    nSubpixelsIterations    : UDINT;
    fApproxPrecision        : REAL;
    eAlgorithm              : ETcVnEdgeDetectionAlgorithm;
    ipEdgePoints1           : Reference To ITcVnContainer;
    ipEdgePoints2           : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aPoint1                 : TcVnPoint2_REAL;
    aPoint2                 : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的从 aStartPoint 到 aEndPoint 的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心

名称	类型	描述
fSearchLineDist	REAL	搜索线之间的距离，以像素为单位（> 0）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
bInvertSearchDirection	BOOL	如果为 true，则搜索从 aStartPoint 和 aEndPoint 之间的中心点沿两个方向开始
fSearchGap	REAL	可选的间隙宽度（>= 0，以 aStartPoint 和 aEndPoint 之间为中心），搜索边缘时忽略该间隙（可缩短执行时间）。要搜索的 2 个边缘必须位于间隙的不同侧。
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
ipEdgePoints1	引用 ITcVnContainer [▶ 365]	返回检测到的靠近 aStartPoint 的边缘点（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL）
ipEdgePoints2	引用 ITcVnContainer [▶ 365]	返回检测到的靠近 aEndPoint 的边缘点（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fMinDistance	REAL	返回检测到的边缘之间的最小距离
aStartPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的起始位置（沿 aEndPoint 方向）
aEndPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的结束位置
aPoint1	TcVnPoint2_REAL [▶ 152]	返回靠近 aStartPoint 的边缘上达到最小距离的点
aPoint2	TcVnPoint2_REAL [▶ 152]	返回靠近 aEndPoint 的边缘上达到最小距离的点

返回值

HRESULT [▶ 135]

更多信息

`F_VN_MeasureMinEdgeDistanceExp` 函数是 F_VN_MeasureMinEdgeDistance [▶ 1505] 的专家版本。它包含额外的参数。

参数

输入图像

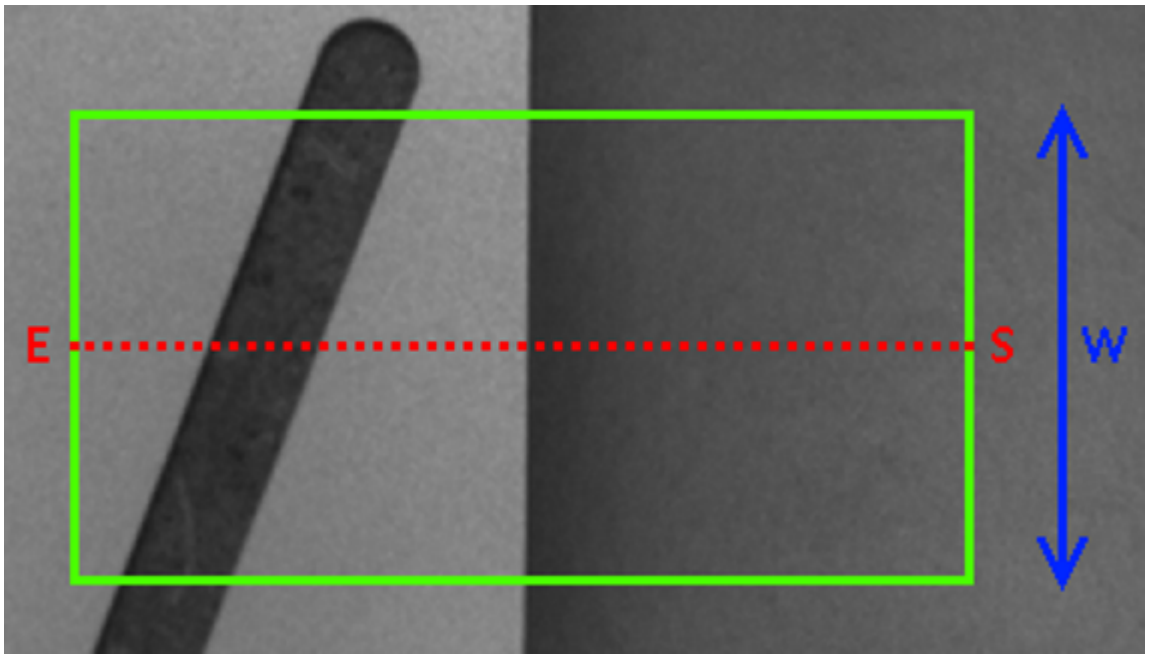
`ipSrcImage` 输入图像必须是单通道灰度图像。

最小距离 (返回值)

`fMinDistance` 参数可返回两条定位到的边缘之间的最小距离。

搜索窗口

首先，必须定义一个搜索窗口，以确定在哪个位置以及沿哪个方向搜索边缘。这是通过 aStartPoint、aEndPoint 和 nSearchLines 参数实现的。搜索窗口由从 aStartPoint 到 aEndPoint 的搜索线及其两侧平行线构成，每个方向各有（nSearchLines - 1）/2 条搜索线，相邻搜索线的距离为 1 个像素。因此，nSearchLines 定义了搜索线的数量，且该值必须始终为奇数。



附图 23: S：起点，E：终点，W：窗口宽度

局部边缘点 （返回值）

ipEdgePoints1 和 ipEdgePoints2 容器会返回根据每条搜索线定位的精确点。如果不需要，可以传输值 0，而不是接口指针。

此外，aPoint1 和 aPoint2 点可返回两条边缘之间距离最小的点。

搜索差距 （Expert（专家））

通过 fSearchGap，您可以指定在已定义搜索线的中间区域有一个在搜索过程中被忽略的区域。这可以缩短执行时间。

边缘定位的参数

关于其余参数的详细解释，请参见边缘定位一章。

应用

例如，使用 Expert（专家）参数测量最小边缘距离的结果是这样的：

```
hr := F_VN_MeasureMinEdgeDistanceExp(
    ipSrcImage              := ipImageIn,
    fMinDistance            := fMinDistance,
    aStartPoint             := aStartPoint,
    aEndPoint               := aEndPoint,
    eEdgeDirection          := TCVN_ED_LIGHT_TO_DARK,
    fMinStrength            := fMinStrength,
    nSearchLines            := nSearchLines,
  fSearchLineDist     := 3,
  nMaxThickness       := 7,
  bInvertSearchDirection := TRUE,
  fSearchGap          := 100,
  nSubpixelsIterations  := 10,
  fApproxPrecision      := 0.001,
  eAlgorithm          := TCVN_EDA_INTERPOLATION,
  ipEdgePoints1       := ipEdgePoints1,
  ipEdgePoints2       := ipEdgePoints2,
  aPoint1           := aPoint1,
  aPoint2           := aPoint2,
    hrPrev                  := hr);
```

使用 F_VN_DrawContours [▶ 1012] 和 F_VN_DrawPoint [▶ 1033] 函数可以将定位到的边缘点可视化：

```
hr := F_VN_DrawContours(ipEdgePoints1, -1, ipImageRes, aColorGreen, 5, hr);
hr := F_VN_DrawContours(ipEdgePoints2, -1, ipImageRes, aColorGreen, 5, hr);

hr := F_VN_DrawPoint(REAL_TO_UDINT(aPoint1[0]), REAL_TO_UDINT(aPoint1[1]), ipImageRes, TCVN_DS_X,
aColorGreen, hr);
hr := F_VN_DrawPoint(REAL_TO_UDINT(aPoint2[0]), REAL_TO_UDINT(aPoint2[1]), ipImageRes, TCVN_DS_X,
aColorGreen, hr);
```

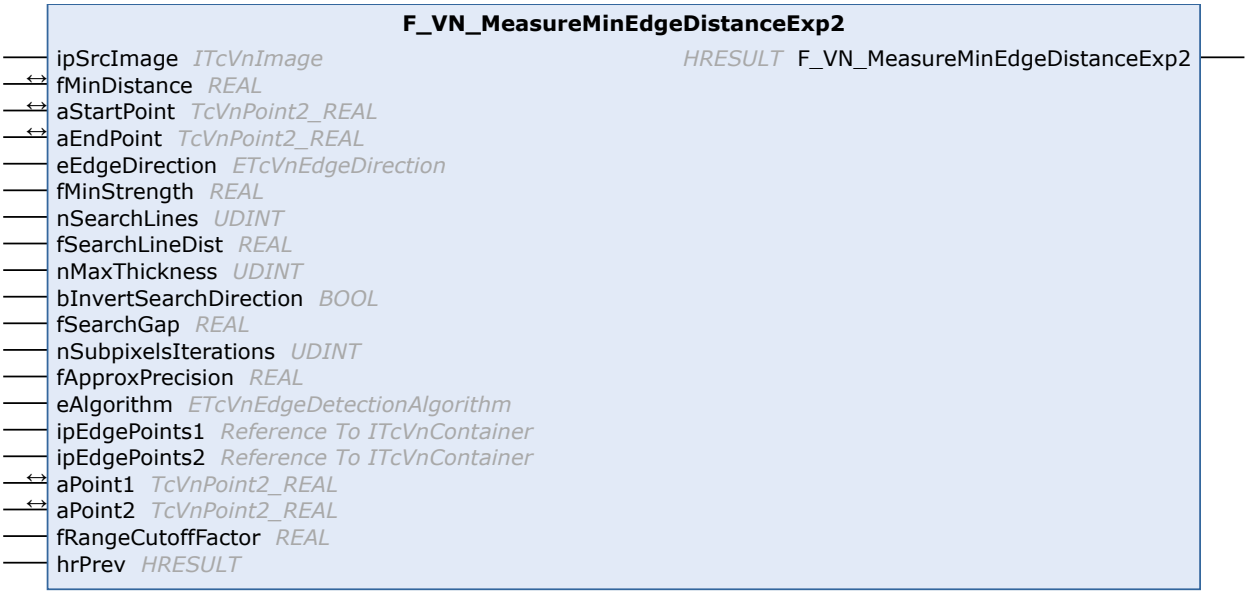
相关函数

- ClosestPointsBF [▶ 1462]，用于计算两个点云中最近的点
- MeasureAngleBetweenEdges [▶ 1490]，用于测量两条边之间的角度
- MeasureEdgeDistance [▶ 1496]，用于测量两条边之间的平均距离
- MeasureMinEdgeDistance [▶ 1505]，用于测量两条边之间的最小距离

所需的许可证

TC3 Vision Metrology 2D

6.1.4.21.25 F_VN_MeasureMinEdgeDistanceExp2



在指定搜索窗口内测量两条边缘之间的最小距离。aStartPoint 和 aEndPoint 之间的中点必须位于两条边缘之间。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
FUNCTION F_VN_MeasureMinEdgeDistanceExp2 : HRESULT
VAR_INPUT
    ipSrcImage             : ITcVnImage;
END_VAR
VAR_IN_OUT
    fMinDistance           : REAL;
    aStartPoint            : TcVnPoint2_REAL;
    aEndPoint              : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    eEdgeDirection         : ETcVnEdgeDirection;
    fMinStrength           : REAL;
    nSearchLines           : UDINT;
    fSearchLineDist        : REAL;
    nMaxThickness          : UDINT;
    bInvertSearchDirection : BOOL;
    fSearchGap             : REAL;
    nSubpixelsIterations   : UDINT;
```

```
    fApproxPrecision        : REAL;
    eAlgorithm              : ETcVnEdgeDetectionAlgorithm;
    ipEdgePoints1           : Reference To ITcVnContainer;
    ipEdgePoints2           : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    aPoint1                 : TcVnPoint2_REAL;
    aPoint2                 : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    fRangeCutoffFactor      : REAL;
    hrPrev                  : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
eEdgeDirection	ETcVnEdgeDirection [▶ 191]	指定要搜索的从 aStartPoint 到 aEndPoint 的边缘方向
fMinStrength	REAL	指定要搜索的边缘的最小强度（强度差）
nSearchLines	UDINT	搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
fSearchLineDist	REAL	搜索线之间的距离，以像素为单位（> 0）
nMaxThickness	UDINT	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
bInvertSearchDirection	BOOL	如果为 true，则搜索从 aStartPoint 和 aEndPoint 之间的中心点沿两个方向开始
fSearchGap	REAL	可选的间隙宽度（>= 0，以 aStartPoint 和 aEndPoint 之间为中心），搜索边缘时忽略该间隙（可缩短执行时间）。要搜索的 2 个边缘必须位于间隙的不同侧。
nSubpixelsIterations	UDINT	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	REAL	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 190]	选择边缘检测算法
ipEdgePoints1	引用 ITcVnContainer [▶ 365]	返回检测到的靠近 aStartPoint 的边缘点（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL）
ipEdgePoints2	引用 ITcVnContainer [▶ 365]	返回检测到的靠近 aEndPoint 的边缘点（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL）
fRangeCutoffFactor	REAL	用于剔除参与拟合的像素范围的相对边缘强度系数。负数表示包含噪声，即另一个方向上的强度差异。每个方向最多使用 nMaxThickness 个像素。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
fMinDistance	REAL	返回检测到的边缘之间的最小距离
aStartPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的起始位置（沿 aEndPoint 方向）
aEndPoint	TcVnPoint2_REAL [▶ 152]	搜索过程的结束位置
aPoint1	TcVnPoint2_REAL [▶ 152]	返回靠近 aStartPoint 的边缘上达到最小距离的点
aPoint2	TcVnPoint2_REAL [▶ 152]	返回靠近 aEndPoint 的边缘上达到最小距离的点

返回值

HRESULT [▶ 135]

更多信息

F_VN_MeasureMinEdgeDistanceExp2 函数是 F_VN_MeasureMinEdgeDistance [▶ 1505] 的专家版本。它包含额外的参数。

参数

输入图像

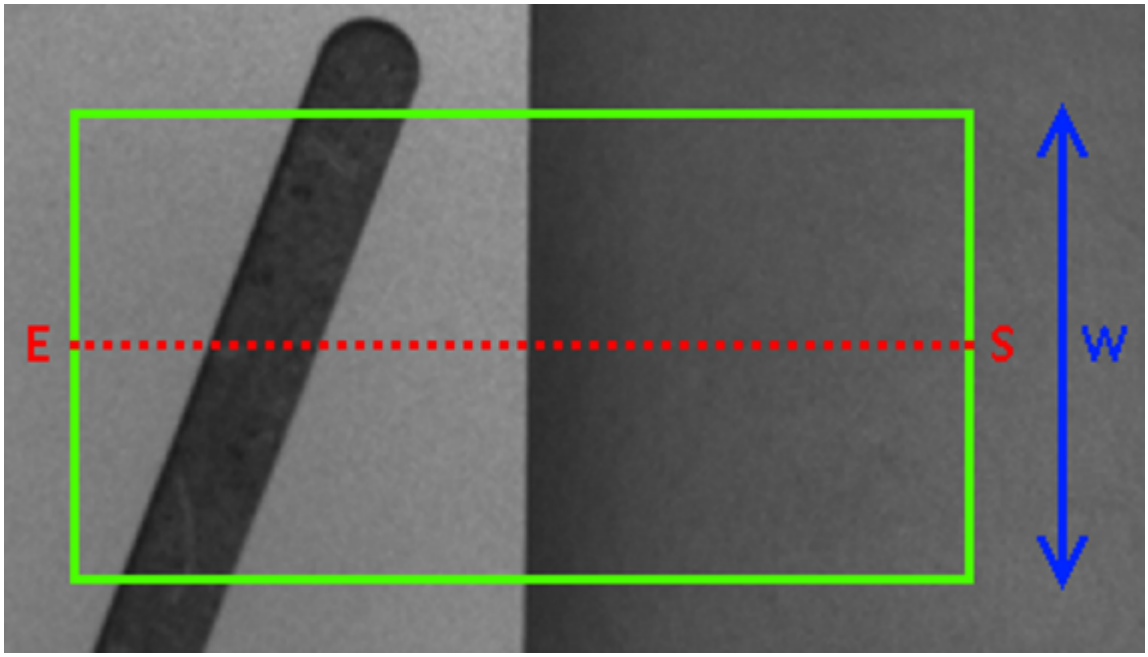
`ipSrcImage` 输入图像必须是单通道灰度图像。

最小距离 (返回值)

`fMinDistance` 参数可返回两条定位到的边缘之间的最小距离。

搜索窗口

首先，必须定义一个搜索窗口，以确定在哪个位置以及沿哪个方向搜索边缘。这是通过 `aStartPoint`、`aEndPoint` 和 `nSearchLines` 参数实现的。搜索窗口由从 `aStartPoint` 到 `aEndPoint` 的搜索线及其两侧平行线构成，每个方向各有（`nSearchLines` - 1）/2 条搜索线，相邻搜索线的距离为 1 个像素。因此，`nSearchLines` 定义了搜索线的数量，且该值必须始终为奇数。



附图 24: S：起点，E：终点，W：窗口宽度

局部边缘点 (返回值)

`ipEdgePoints1` 和 `ipEdgePoints2` 容器会返回根据每条搜索线定位的精确点。如果不需要，可以传输值 0，而不是接口指针。

此外，`aPoint1` 和 `aPoint2` 点可返回两条边缘之间距离最小的点。

搜索差距 (Expert（专家）)

通过 `fSearchGap`，您可以指定在已定义搜索线的中间区域有一个在搜索过程中被忽略的区域。这可以缩短执行时间。

边缘定位的参数

关于其余参数的详细解释，请参见边缘定位一章。

应用

例如，使用 Expert（专家）参数测量最小边缘距离的结果是这样的：

```
hr := F_VN_MeasureMinEdgeDistanceExp2(
    ipSrcImage              := ipImageIn,
    fMinDistance            := fMinDistance,
    aStartPoint             := aStartPoint,
    aEndPoint               := aEndPoint,
    eEdgeDirection          := TCVN_ED_LIGHT_TO_DARK,
    fMinStrength            := fMinStrength,
    nSearchLines            := nSearchLines,
    fSearchLineDist         := 3,
    nMaxThickness           := 7,
    bInvertSearchDirection  := TRUE,
    fSearchGap              := 100,
    nSubpixelsIterations    := 10,
    fApproxPrecision        := 0.001,
    eAlgorithm              := TCVN_EDA_INTERPOLATION,
    ipEdgePoints1           := ipEdgePoints1,
    ipEdgePoints2           := ipEdgePoints2,
    aPoint1                 := aPoint1,
    aPoint2                 := aPoint2,
 fRangeCutoffFactor    := 0.135,
    hrPrev                  := hr);
```

相关函数

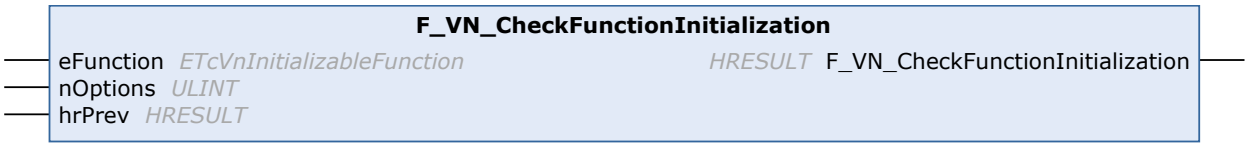
- ClosestPointsBF [▶ 1462]，用于计算两个点云中最近的点
- MeasureAngleBetweenEdges [▶ 1490]，用于测量两条边之间的角度
- MeasureEdgeDistance [▶ 1496]，用于测量两条边之间的平均距离
- MeasureMinEdgeDistance [▶ 1505]，用于测量两条边之间的最小距离

所需的许可证

TC3 Vision Metrology 2D

6.1.4.22 其它

6.1.4.22.1 F_VN_CheckFunctionInitialization



Check if a function is initialized with the specified options.

语法

定义：

```
FUNCTION F_VN_CheckFunctionInitialization : HRESULT
VAR_INPUT
    eFunction : ETcVnInitializableFunction;
    nOptions  : ULINT;
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
eFunction	ETcVnInitializableFunction [▶ 198]	可初始化函数
nOptions	ULINT	函数的初始化选项

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

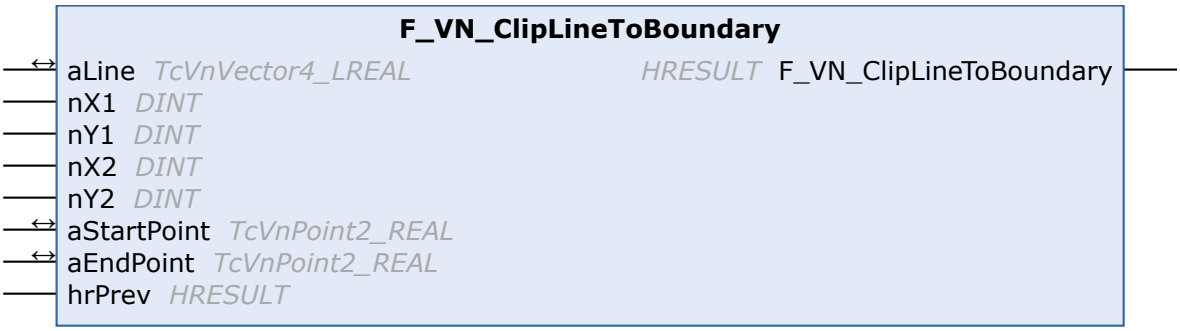
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.2 F_VN_ClipLineToBoundary



Clips a line to a rectangular boundary.

语法

定义：

```
FUNCTION F_VN_ClipLineToBoundary : HRESULT
VAR_IN_OUT
    aLine       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    nX1         : DINT;
    nY1         : DINT;
    nX2         : DINT;
    nY2         : DINT;
END_VAR
VAR_IN_OUT
    aStartPoint : TcVnPoint2_REAL;
    aEndPoint   : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
nX1	DINT	左上角矩形点的 x
nY1	DINT	左上角矩形点的 y
nX2	DINT	右下角矩形点的 x
nY2	DINT	右下角矩形点的 Y

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aLine	TcVnVector4_LREAL [▶ 153]	第一个和第二个元素描述了方向向量的 x 和 y 分量。第三个和第四个元素描述位置向量的 x 和 y 分量。
aStartPoint	TcVnPoint2_REAL [▶ 152]	返回直线的剪切起点。
aEndPoint	TcVnPoint2_REAL [▶ 152]	返回直线的剪切终点。

返回值

HRESULT [▶ 135]

更多信息

该函数根据指定区域调整线条。

对于这一区域的各种定义形式，存在不同的函数：

矩形区域的定义	待使用的函数
由单个 X/Y 坐标指定	F_VN_ClipLineToBoundary [▶ 1515]
按图像大小	F_VN_ClipLineToBoundary_ITcVnImage [▶ 1516]
按矩形结构	F_VN_ClipLineToBoundary_TcVnRectangle_DINT [▶ 1517]

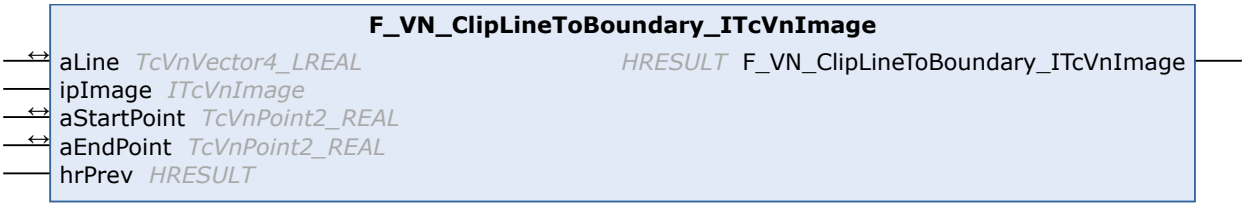
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.3 F_VN_ClipLineToBoundary_ITcVnImage



Clips a line to the boundary of an image.

语法

定义：

```
FUNCTION F_VN_ClipLineToBoundary_ITcVnImage : HRESULT
VAR_IN_OUT
    aLine       : TcVnVector4_LREAL;
END_VAR
VAR_INPUT
    ipImage     : ITcVnImage;
```

```
END_VAR
VAR_IN_OUT
    aStartPoint : TcVnPoint2_REAL;
    aEndPoint   : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	从中提取边界的图像
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aLine	TcVnVector4_LREAL [▶ 153]	第一个和第二个元素描述了方向向量的 x 和 y 分量。第三个和第四个元素描述位置向量的 x 和 y 分量。
aStartPoint	TcVnPoint2_REAL [▶ 152]	返回直线的剪切起点。
aEndPoint	TcVnPoint2_REAL [▶ 152]	返回直线的剪切终点。

返回值

HRESULT [▶ 135]

更多信息

该函数根据指定区域调整线条。

对于这一区域的各种定义形式，存在不同的函数：

矩形区域的定义	待使用的函数
由单个 X/Y 坐标指定	F_VN_ClipLineToBoundary [▶ 1515]
按图像大小	F_VN_ClipLineToBoundary_ITcVnImage [▶ 1516]
按矩形结构	F_VN_ClipLineToBoundary_TcVnRectangle_DINT [▶ 1517]

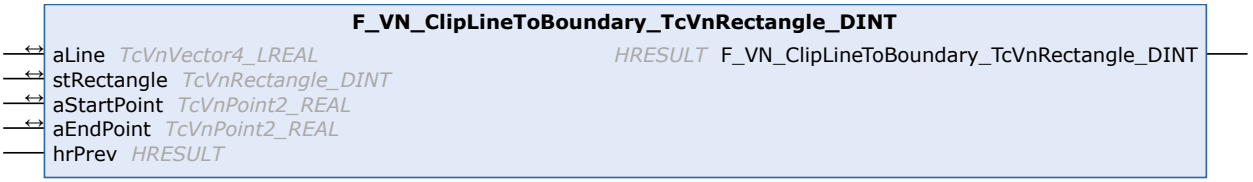
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.4 F_VN_ClipLineToBoundary_TcVnRectangle_DINT



Clips a line to a rectangular boundary.

语法

定义：

```
FUNCTION F_VN_ClipLineToBoundary_TcVnRectangle_DINT : HRESULT
VAR_IN_OUT
    aLine       : TcVnVector4_LREAL;
    stRectangle : TcVnRectangle_DINT;
    aStartPoint : TcVnPoint2_REAL;
    aEndPoint   : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aLine	TcVnVector4_LREAL [▶ 153]	第一个和第二个元素描述了方向向量的 x 和 y 分量。第三个和第四个元素描述位置向量的 x 和 y 分量。
stRectangle	TcVnRectangle_DINT [▶ 236]	矩形边界
aStartPoint	TcVnPoint2_REAL [▶ 152]	返回直线的剪切起点。
aEndPoint	TcVnPoint2_REAL [▶ 152]	返回直线的剪切终点。

返回值

HRESULT [▶ 135]

更多信息

该函数根据指定区域调整线条。

对于这一区域的各种定义形式，存在不同的函数：

矩形区域的定义	待使用的函数
由单个 X/Y 坐标指定	F_VN_ClipLineToBoundary [▶ 1515]
按图像大小	F_VN_ClipLineToBoundary_ITcVnImage [▶ 1516]
按矩形结构	F_VN_ClipLineToBoundary_TcVnRectangle_DINT [▶ 1517]

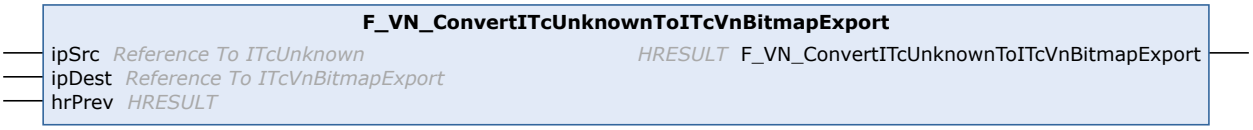
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.5 F_VN_ConvertITcUnknownToITcVnBitmapExport



Convert an ITcUnknown interface pointer to an ITcVnBitmapExport interface pointer. The destination pointer will be released (if existing) and set to the source pointer. The source pointer will be set to 0.

语法

定义:

```
FUNCTION F_VN_ConvertITcUnknownToITcVnBitmapExport : HRESULT
VAR_INPUT
    ipSrc  : Reference To ITcUnknown;
    ipDest : Reference To ITcVnBitmapExport;
    hrPrev : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrc	引用 ITcUnknown [▶ 424]	源指针
ipDest	引用 ITcVnBitmapExport [▶ 388]	目标指针
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

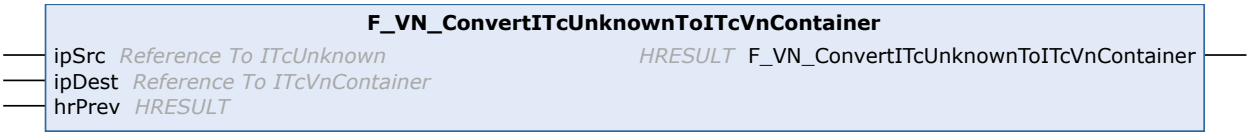
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.6 F_VN_ConvertITcUnknownToITcVnContainer



Convert an ITcUnknown interface pointer to an ITcVnContainer interface pointer. The destination pointer will be released (if existing) and set to the source pointer. The source pointer will be set to 0.

语法

定义:

```
FUNCTION F_VN_ConvertITcUnknownToITcVnContainer : HRESULT
VAR_INPUT
    ipSrc  : Reference To ITcUnknown;
```

```
    ipDest : Reference To ITcVnContainer;
    hrPrev : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrc	引用 ITcUnknown [▶ 424]	源指针
ipDest	引用 ITcVnContainer [▶ 365]	目标指针
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

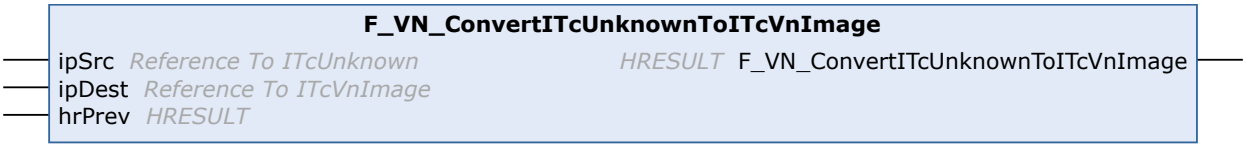
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.7 F_VN_ConvertITcUnknownToITcVnImage



Convert an ITcUnknown interface pointer to an ITcVnImage interface pointer. The destination pointer will be released (if existing) and set to the source pointer. The source pointer will be set to 0.

语法

定义：

```
FUNCTION F_VN_ConvertITcUnknownToITcVnImage : HRESULT
VAR_INPUT
    ipSrc  : Reference To ITcUnknown;
    ipDest : Reference To ITcVnImage;
    hrPrev : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrc	引用 ITcUnknown [▶ 424]	源指针
ipDest	引用 ITcVnImage [▶ 407]	目标指针
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

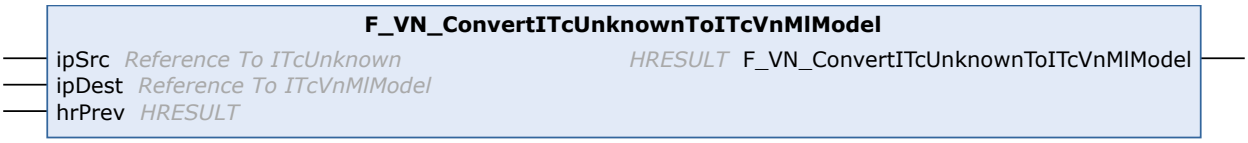
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.8 F_VN_ConvertITcUnknownToITcVnMlModel



Convert an ITcUnknown interface pointer to an ITcVnMlModel interface pointer. The destination pointer will be released (if existing) and set to the source pointer. The source pointer will be set to 0.

语法

定义：

```
FUNCTION F_VN_ConvertITcUnknownToITcVnMlModel : HRESULT
VAR_INPUT
    ipSrc  : Reference To ITcUnknown;
    ipDest : Reference To ITcVnMlModel;
    hrPrev : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrc	引用 ITcUnknown [▶ 424]	源指针
ipDest	引用 ITcVnMlModel [▶ 417]	目标指针
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

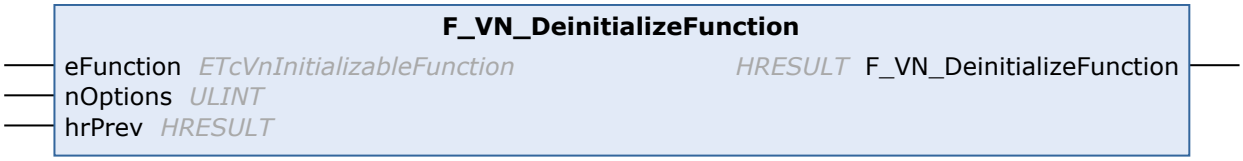
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.9 F_VN_DeinitializeFunction



Deinitialize the specified options for the function.

语法

定义：

```
FUNCTION F_VN_DeinitializeFunction : HRESULT
VAR_INPUT
    eFunction : ETcVnInitializableFunction;
    nOptions  : ULINT;
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
eFunction	ETcVnInitializableFunction [▶ 198]	可初始化函数
nOptions	ULINT	函数的初始化选项
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

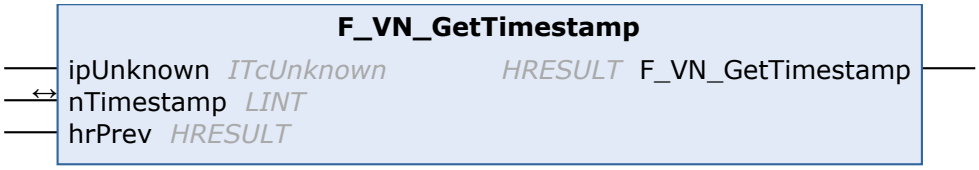
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.10 F_VN_GetTimestamp



Get the timestamp (actual DC time) from the last change of an object (e.g. ITcVnImage or ITcVnContainer)

语法

定义：

```
FUNCTION F_VN_GetTimestamp : HRESULT
VAR_INPUT
    ipUnknown  : ITcUnknown;
END_VAR
VAR_IN_OUT
    nTimestamp : LINT;
END_VAR
VAR_INPUT
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipUnknown	ITcUnknown [▶ 424]	实现 ITcUnknown 和 ITcVnTimestamp 的对象（例如 ITcVnImage 或 ITcVnContainer）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
nTimestamp	LINT	返回时间戳

返回值

HRESULT [▶ 135]

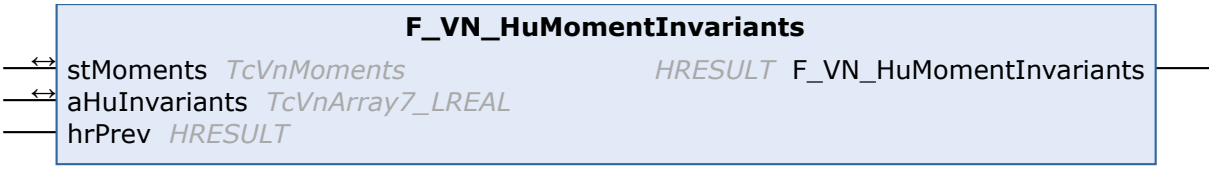
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.11 F_VN_HuMomentInvariants



Computes the Hu moment invariants.

语法

定义：

```
FUNCTION F_VN_HuMomentInvariants : HRESULT
VAR_IN_OUT
    stMoments     : TcVnMoments;
    aHuInvariants : TcVnArray7_LREAL;
END_VAR
VAR_INPUT
    hrPrev        : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stMoments	TcVnMoments [▶ 226]	基于此矩计算 Hu 不变矩
aHuInvariants	TcVnArray7_LREAL [▶ 153]	返回 Hu 不变矩

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.12 F_VN_InitMatrixStruct



Initialize a struct of the type TcVnMatrix extending a buffer with metainformation so that it can be used as a matrix.

语法

定义：

```
FUNCTION F_VN_InitMatrixStruct : HRESULT
VAR_INPUT
    pSrcBuffer   : PVOID;
END_VAR
VAR_IN_OUT
    stDestMatrix : TcVnMatrix;
END_VAR
VAR_INPUT
    nRows        : UDINT;
    nCols        : UDINT;
    eElementType : ETcVnElementType;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
pSrcBuffer	PVOID	源缓冲区
nRows	UDINT	矩阵行
nCols	UDINT	矩阵列
eElementType	ETcVnElementType [▶ 191]	矩阵元素类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stDestMatrix	TcVnMatrix [▶ 226]	返回完成的矩阵结构体

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_InitMatrixStruct` 可从任意内存区域创建 TcVnMatrix [▶ 226] 类型的矩阵结构。

参数

数据缓冲区

指针`pSrcBuffer`必须指向包含矩阵数据的内存区域。例如，这可以是一个数组。

矩阵描述

矩阵必须由行数`nRows`、列数`nCols`、以及元素的数据类型`eElementType`进行描述。

结果矩阵

TcVnMatrix [▶ 226]类型的矩阵结构`stDestMatrix`作为结果返回。

应用

例如，`3×3` 矩阵的创建过程是这样的：

```
VAR
    aMatrixArray    :   ARRAY [0..2, 0..2] OF USINT :=
                                                [-1, 0, 1,
                                                 -2, 0, 2,
                                                 -1, 0, 1];
END_VAR

hr := F_VN_InitMatrixStruct(
    pSrcBuffer      := ADR(aMatrixArray),
    stDestMatrix    := stMatrix,
    nRows           := 3,               // must match with array dimensions
    nCols           := 3,               // must match with array dimensions
    eElementType    := TCVN_ET_USINT, // must match with array type
    hrPrev          := hr);
```

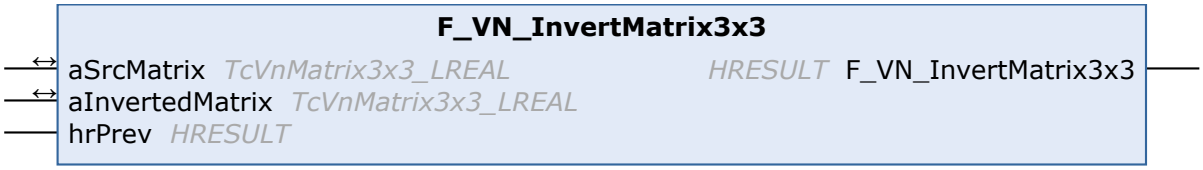
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.13 F_VN_InvertMatrix3x3



Invert a 3x3 matrix.

语法

定义：

```
FUNCTION F_VN_InvertMatrix3x3 : HRESULT
VAR_IN_OUT
    aSrcMatrix      : TcVnMatrix3x3_LREAL;
    aInvertedMatrix : TcVnMatrix3x3_LREAL;
END_VAR
VAR_INPUT
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aSrcMatrix	TcVnMatrix3x3_LREAL [▶ 153]	源矩阵
aInvertedMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回反转的矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.14 F_VN_LineIntersectionPoint



Computes the intersection point between two lines (returns S_FALSE if the provided lines are parallel).

语法

定义：

```
FUNCTION F_VN_LineIntersectionPoint : HRESULT
VAR_IN_OUT
    aLine1          : TcVnVector4_LREAL;
    aLine2          : TcVnVector4_LREAL;
    aIntersecPoint : TcVnPoint2_REAL;
END_VAR
VAR_INPUT
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aLine1	TcVnVector4_LREAL [▶ 153]	第一行。第一个和第二个元素描述的是该直线共线向量的 x 和 y 分量。第三个和第四个元素描述该直线上一个点的 x 和 y 分量。
aLine2	TcVnVector4_LREAL [▶ 153]	第二行。第一个和第二个元素描述的是该直线共线向量的 x 和 y 分量。第三个和第四个元素描述该直线上一个点的 x 和 y 分量。
aIntersecPoint	TcVnPoint2_REAL [▶ 152]	返回 aLine1 和 aLine2 的交点。

返回值

HRESULT [▶ 135]

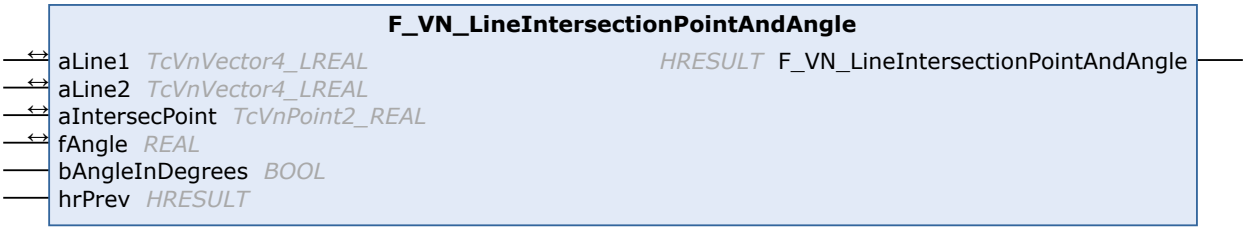
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.15 F_VN_LineIntersectionPointAndAngle



Computes the intersection point and angle between two lines (returns S_FALSE if the provided lines are parallel).

语法

定义：

```
FUNCTION F_VN_LineIntersectionPointAndAngle : HRESULT
VAR_IN_OUT
    aLine1          : TcVnVector4_LREAL;
    aLine2          : TcVnVector4_LREAL;
    aIntersecPoint  : TcVnPoint2_REAL;
    fAngle          : REAL;
END_VAR
VAR_INPUT
    bAngleInDegrees : BOOL;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
bAngleInDegrees	BOOL	如果为 TRUE，fAngle 以度为单位。如果为 FALSE，则 fAngle 以弧度为单位。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aLine1	TcVnVector4_LREAL [▶ 153]	第一行。第一个和第二个元素描述的是该直线共线向量的 x 和 y 分量。第三个和第四个元素描述该直线上一个点的 x 和 y 分量。
aLine2	TcVnVector4_LREAL [▶ 153]	第二行。第一个和第二个元素描述的是该直线共线向量的 x 和 y 分量。第三个和第四个元素描述该直线上一个点的 x 和 y 分量。
aIntersecPoint	TcVnPoint2_REAL [▶ 152]	返回 aLine1 和 aLine2 的交点。
fAngle	REAL	返回 aLine1 和 aLine2 的夹角。

返回值

HRESULT [▶ 135]

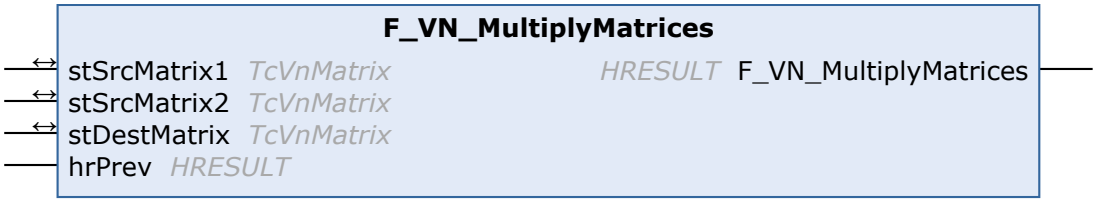
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.16 F_VN_MultiplyMatrices



Matrix multiplication of two matrices (A * B = C). All matrices are represented by structs holding a pointer to an array of the appropriate size.

语法

定义：

```
FUNCTION F_VN_MultiplyMatrices : HRESULT
VAR_IN_OUT
    stSrcMatrix1 : TcVnMatrix;
    stSrcMatrix2 : TcVnMatrix;
    stDestMatrix : TcVnMatrix;
END_VAR
VAR_INPUT
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stSrcMatrix1	TcVnMatrix [▶ 226]	第一源矩阵（A）
stSrcMatrix2	TcVnMatrix [▶ 226]	第二源矩阵（B）
stDestMatrix	TcVnMatrix [▶ 226]	目标矩阵（C）（目标矩阵由该函数填充，但需要提供所需的内存。）

返回值

HRESULT [▶ 135]

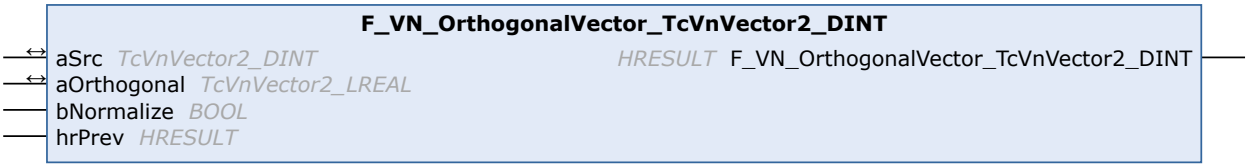
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.17 F_VN_OrthogonalVector_TcVnVector2_DINT



计算给定源向量的正交向量。

语法

定义：

```
FUNCTION F_VN_OrthogonalVector_TcVnVector2_DINT : HRESULT
VAR_IN_OUT
    aSrc         : TcVnVector2_DINT;
    aOrthogonal : TcVnVector2_LREAL;
END_VAR
VAR_INPUT
    bNormalize   : BOOL;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
bNormalize	BOOL	如果为 true，则 aOrthogonal 长度被归一化为 1，否则 aOrthogonal 的长度与 aSrc 相同
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aSrc	TcVnVector2_DINT [▶ 153]	源向量
aOrthogonal	TcVnVector2_LREAL [▶ 153]	返回与 aSrc 正交的向量

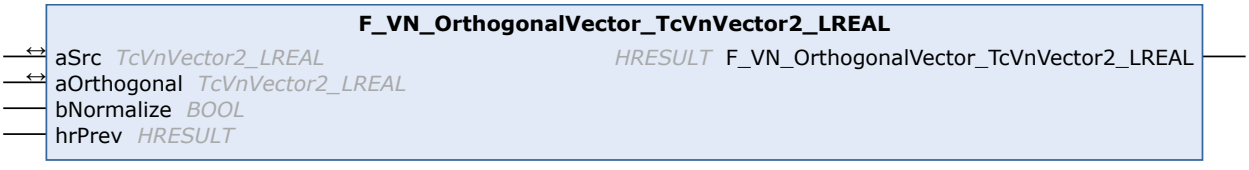
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.22.18 F_VN_OrthogonalVector_TcVnVector2_LREAL



计算给定源向量的正交向量。

语法

定义：

```
FUNCTION F_VN_OrthogonalVector_TcVnVector2_LREAL : HRESULT
VAR_IN_OUT
    aSrc         : TcVnVector2_LREAL;
```

```
    aOrthogonal : TcVnVector2_LREAL;
END_VAR
VAR_INPUT
    bNormalize  : BOOL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
bNormalize	BOOL	如果为 true，则 aOrthogonal 长度被归一化为 1，否则 aOrthogonal 的长度与 aSrc 相同
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aSrc	TcVnVector2_LREAL [▶ 153]	源向量
aOrthogonal	TcVnVector2_LREAL [▶ 153]	返回与 aSrc 正交的向量

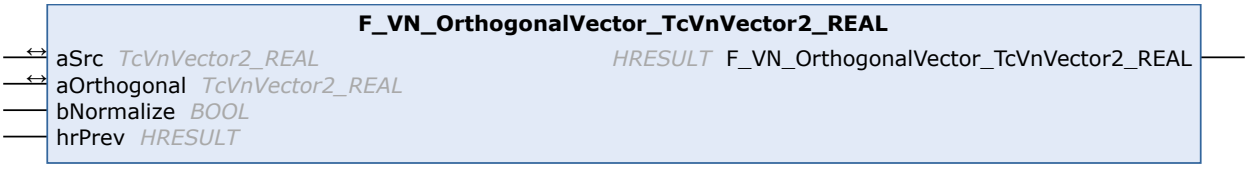
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.22.19 F_VN_OrthogonalVector_TcVnVector2_REAL



计算给定源向量的正交向量。

语法

定义：

```
FUNCTION F_VN_OrthogonalVector_TcVnVector2_REAL : HRESULT
VAR_IN_OUT
    aSrc        : TcVnVector2_REAL;
    aOrthogonal : TcVnVector2_REAL;
END_VAR
VAR_INPUT
    bNormalize  : BOOL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
bNormalize	BOOL	如果为 true，则 aOrthogonal 长度被归一化为 1，否则 aOrthogonal 的长度与 aSrc 相同
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aSrc	TcVnVector2_REAL [▶ 153]	源向量
aOrthogonal	TcVnVector2_REAL [▶ 153]	返回与 aSrc 正交的向量

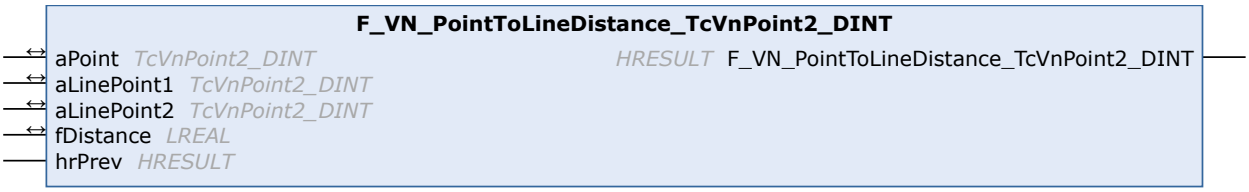
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.22.20 F_VN_PointToLineDistance_TcVnPoint2_DINT



计算点与直线之间的最短距离。

语法

定义：

```
FUNCTION F_VN_PointToLineDistance_TcVnPoint2_DINT : HRESULT
VAR_IN_OUT
    aPoint      : TcVnPoint2_DINT;
    aLinePoint1 : TcVnPoint2_DINT;
    aLinePoint2 : TcVnPoint2_DINT;
    fDistance   : LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aPoint	TcVnPoint2_DINT [▶ 152]	点
aLinePoint1	TcVnPoint2_DINT [▶ 152]	线点 1
aLinePoint2	TcVnPoint2_DINT [▶ 152]	线点 2
fDistance	LREAL	返回点与直线之间的距离

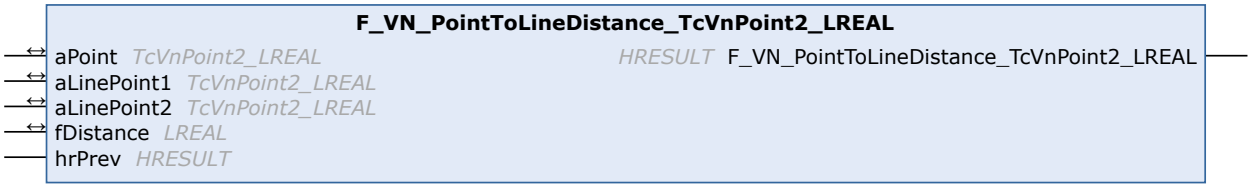
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.22.21 F_VN_PointToLineDistance_TcVnPoint2_LREAL



计算点与直线之间的最短距离。

语法

定义：

```
FUNCTION F_VN_PointToLineDistance_TcVnPoint2_LREAL : HRESULT
VAR_IN_OUT
    aPoint      : TcVnPoint2_LREAL;
    aLinePoint1 : TcVnPoint2_LREAL;
    aLinePoint2 : TcVnPoint2_LREAL;
    fDistance   : LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aPoint	TcVnPoint2_LREAL [▶ 152]	点
aLinePoint1	TcVnPoint2_LREAL [▶ 152]	线点 1
aLinePoint2	TcVnPoint2_LREAL [▶ 152]	线点 2
fDistance	LREAL	返回点与直线之间的距离

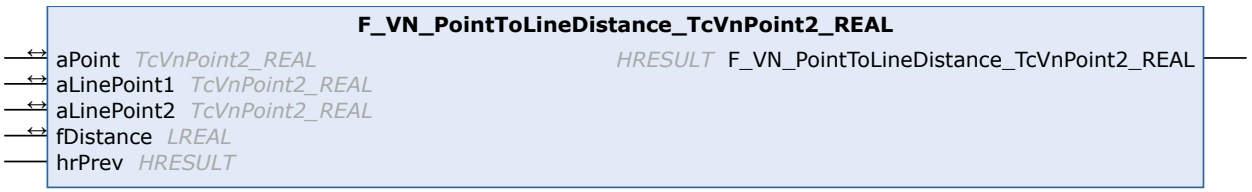
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.22.22 F_VN_PointToLineDistance_TcVnPoint2_REAL



计算点与直线之间的最短距离。

语法

定义：

```
FUNCTION F_VN_PointToLineDistance_TcVnPoint2_REAL : HRESULT
VAR_IN_OUT
    aPoint      : TcVnPoint2_REAL;
    aLinePoint1 : TcVnPoint2_REAL;
    aLinePoint2 : TcVnPoint2_REAL;
    fDistance   : LREAL;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aPoint	TcVnPoint2_REAL [▶ 152]	点
aLinePoint1	TcVnPoint2_REAL [▶ 152]	线点 1
aLinePoint2	TcVnPoint2_REAL [▶ 152]	线点 2
fDistance	LREAL	返回点与直线之间的距离

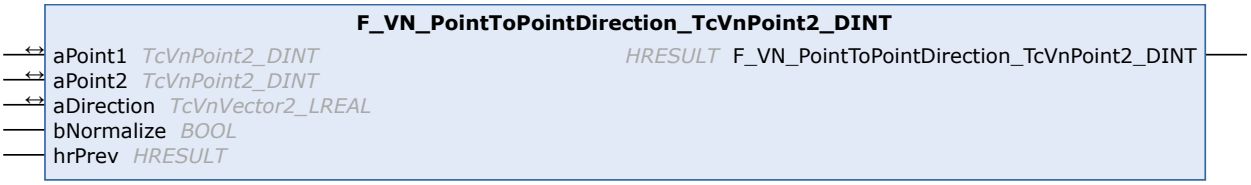
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.22.23 F_VN_PointToPointDirection_TcVnPoint2_DINT



计算 2 个点之间的方向向量。

语法

定义：

```
FUNCTION F_VN_PointToPointDirection_TcVnPoint2_DINT : HRESULT
VAR_IN_OUT
    aPoint1    : TcVnPoint2_DINT;
    aPoint2    : TcVnPoint2_DINT;
    aDirection : TcVnVector2_LREAL;
END_VAR
VAR_INPUT
    bNormalize : BOOL;
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
bNormalize	BOOL	如果为 true，则 aDirection 长度被归一化为 1，否则 aDirection 长度为 aPoint1 与 aPoint2 之间的距离
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aPoint1	TcVnPoint2_DINT [▶ 152]	点 1
aPoint2	TcVnPoint2_DINT [▶ 152]	点 2
aDirection	TcVnVector2_LREAL [▶ 153]	返回从 aPoint1 到 aPoint2 的方向向量

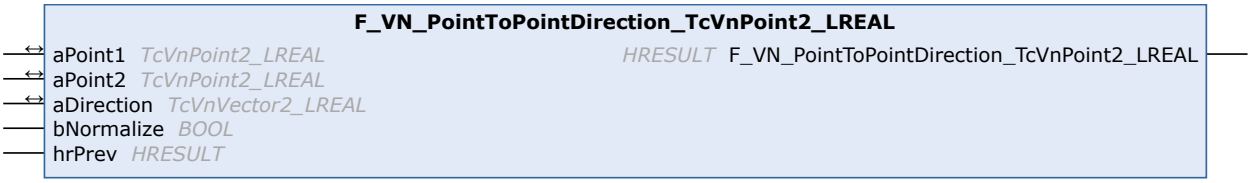
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.22.24 F_VN_PointToPointDirection_TcVnPoint2_LREAL



计算 2 个点之间的方向向量。

语法

定义：

```
FUNCTION F_VN_PointToPointDirection_TcVnPoint2_LREAL : HRESULT
VAR_IN_OUT
    aPoint1    : TcVnPoint2_LREAL;
    aPoint2    : TcVnPoint2_LREAL;
    aDirection : TcVnVector2_LREAL;
END_VAR
VAR_INPUT
    bNormalize : BOOL;
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
bNormalize	BOOL	如果为 true，则 aDirection 长度被归一化为 1，否则 aDirection 长度为 aPoint1 与 aPoint2 之间的距离
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aPoint1	TcVnPoint2_LREAL [▶ 152]	点 1
aPoint2	TcVnPoint2_LREAL [▶ 152]	点 2
aDirection	TcVnVector2_LREAL [▶ 153]	返回从 aPoint1 到 aPoint2 的方向向量

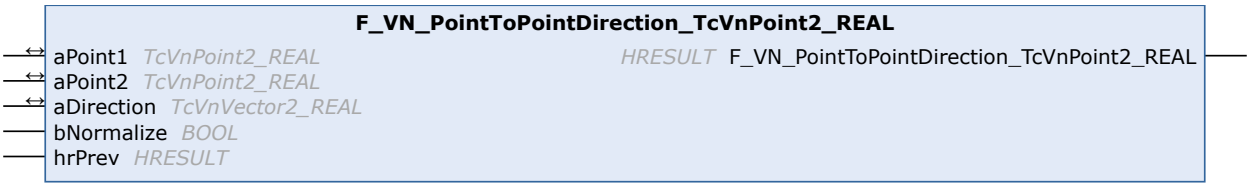
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.22.25 F_VN_PointToPointDirection_TcVnPoint2_REAL



计算 2 个点之间的方向向量。

语法

定义：

```
FUNCTION F_VN_PointToPointDirection_TcVnPoint2_REAL : HRESULT
VAR_IN_OUT
    aPoint1    : TcVnPoint2_REAL;
    aPoint2    : TcVnPoint2_REAL;
    aDirection : TcVnVector2_REAL;
END_VAR
VAR_INPUT
    bNormalize : BOOL;
    hrPrev     : HRESULT;
END_VAR
```

输入

名称	类型	描述
bNormalize	BOOL	如果为 true，则 aDirection 长度被归一化为 1，否则 aDirection 长度为 aPoint1 与 aPoint2 之间的距离
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aPoint1	TcVnPoint2_REAL [▶ 152]	点 1
aPoint2	TcVnPoint2_REAL [▶ 152]	点 2
aDirection	TcVnVector2_REAL [▶ 153]	返回从 aPoint1 到 aPoint2 的方向向量

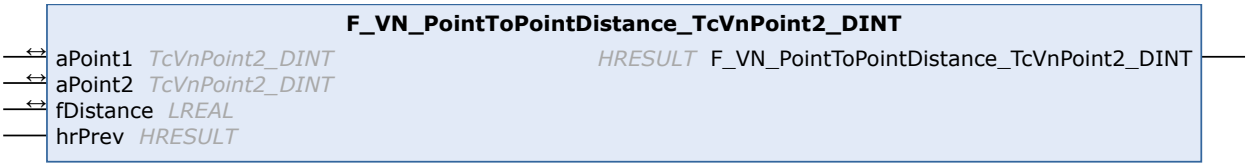
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.22.26 F_VN_PointToPointDistance_TcVnPoint2_DINT



计算 2 个点之间的距离。

语法

定义：

```
FUNCTION F_VN_PointToPointDistance_TcVnPoint2_DINT : HRESULT
VAR_IN_OUT
    aPoint1   : TcVnPoint2_DINT;
    aPoint2   : TcVnPoint2_DINT;
    fDistance : LREAL;
END_VAR
VAR_INPUT
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aPoint1	TcVnPoint2_DINT [▶ 152]	点 1
aPoint2	TcVnPoint2_DINT [▶ 152]	点 2
fDistance	LREAL	返回各点之间的距离

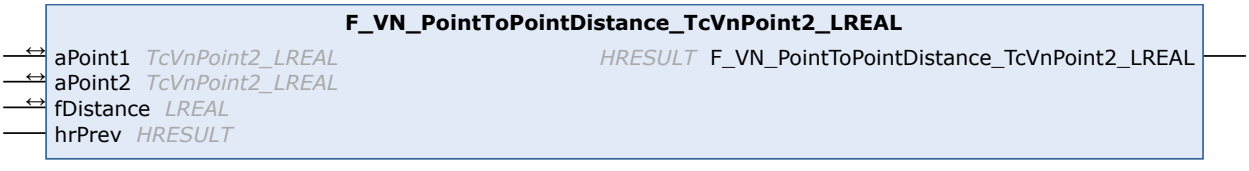
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.22.27 F_VN_PointToPointDistance_TcVnPoint2_LREAL



计算 2 个点之间的距离。

语法

定义：

```
FUNCTION F_VN_PointToPointDistance_TcVnPoint2_LREAL : HRESULT
VAR_IN_OUT
    aPoint1   : TcVnPoint2_LREAL;
    aPoint2   : TcVnPoint2_LREAL;
    fDistance : LREAL;
END_VAR
VAR_INPUT
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aPoint1	TcVnPoint2_LREAL [▶ 152]	点 1
aPoint2	TcVnPoint2_LREAL [▶ 152]	点 2
fDistance	LREAL	返回各点之间的距离

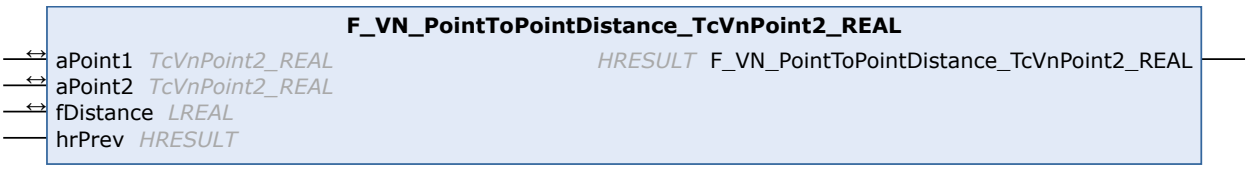
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.22.28 F_VN_PointToPointDistance_TcVnPoint2_REAL



计算 2 个点之间的距离。

语法

定义：

```
FUNCTION F_VN_PointToPointDistance_TcVnPoint2_REAL : HRESULT
VAR_IN_OUT
    aPoint1   : TcVnPoint2_REAL;
    aPoint2   : TcVnPoint2_REAL;
    fDistance : LREAL;
END_VAR
VAR_INPUT
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aPoint1	TcVnPoint2_REAL [▶ 152]	点 1
aPoint2	TcVnPoint2_REAL [▶ 152]	点 2
fDistance	LREAL	返回各点之间的距离

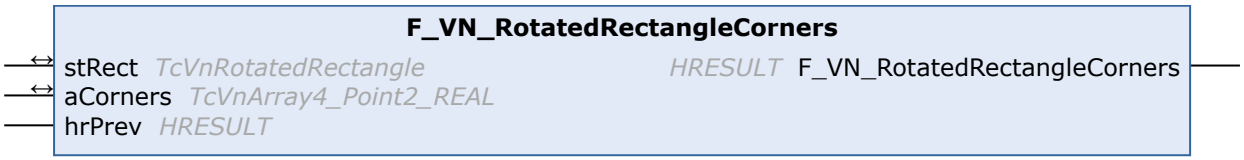
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.1.4.22.29 F_VN_RotatedRectangleCorners



计算旋转矩形的 4 个角点。

语法

定义：

```
FUNCTION F_VN_RotatedRectangleCorners : HRESULT
VAR_IN_OUT
    stRect   : TcVnRotatedRectangle;
    aCorners : TcVnArray4_Point2_REAL;
END_VAR
VAR_INPUT
    hrPrev   : HRESULT;
END_VAR
```

输入

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

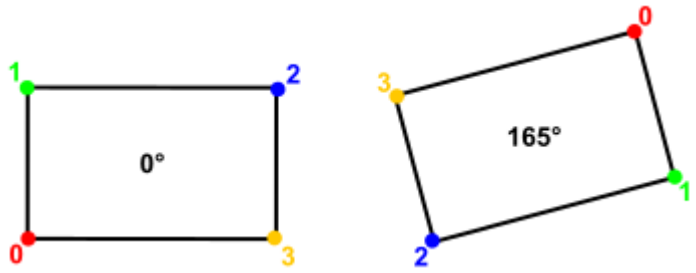
名称	类型	描述
stRect	TcVnRotatedRectangle [▶ 237]	旋转矩形
aCorners	TcVnArray4_Point2_REAL [▶ 153]	返回 4 个角点

返回值

HRESULT [▶ 135]

更多信息

该函数以 TcVnArray4_Point2_REAL 返回旋转矩形的四个角点 aCorners，对应于 Array [0..3] of TcVnPoint2_REAL 。当角度为 0° 时，aCorners[0] 包含与轴平行的矩形的左下角坐标。其余索引按顺时针顺序依次为左上角（aCorners[1]）、右上角（aCorners[2]）、右下角（aCorners[3]）。例如，如果矩形旋转了 165°，则角的分配也会相应旋转，使得原来的左下角现在位于右上角。



所需的许可证

TC3 Vision Base

6.1.4.22.30 F_VN_RotatedRectangleIntersection



Finds the intersection of 2 rotated rectangles.

语法

定义：

```
FUNCTION F_VN_RotatedRectangleIntersection : HRESULT
VAR_IN_OUT
    stRect1         : TcVnRotatedRectangle;
    stRect2         : TcVnRotatedRectangle;
END_VAR
VAR_INPUT
    ipIntersection : Reference To ITcVnContainer;
END_VAR
VAR_IN_OUT
    eIntersection  : ETcVnRectangleIntersection;
END_VAR
VAR_INPUT
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipIntersection	引用 ITcVnContainer [▶ 365]	返回交点（ContainerType_Vector_TcVnPoint2_REAL，重复使用非零接口指针）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
stRect1	TcVnRotatedRectangle [▶ 237]	旋转矩形 1

名称	类型	描述
stRect2	TcVnRotatedRectangle [▶ 237]	旋转矩形 2
eIntersection	ETcVnRectangleIntersection [▶ 208]	返回交点类型

返回值

HRESULT [▶ 135]

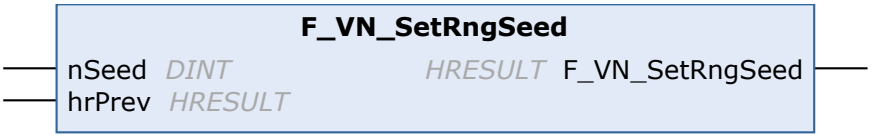
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.31 F_VN_SetRngSeed



Sets the internal pseudo random number generator seed (intended for testing purposes only).

语法

定义：

```
FUNCTION F_VN_SetRngSeed : HRESULT
VAR_INPUT
    nSeed  : DINT;
    hrPrev : HRESULT;
END_VAR
```

输入

名称	类型	描述
nSeed	DINT	种子（0 会使发生器回到初始状态）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

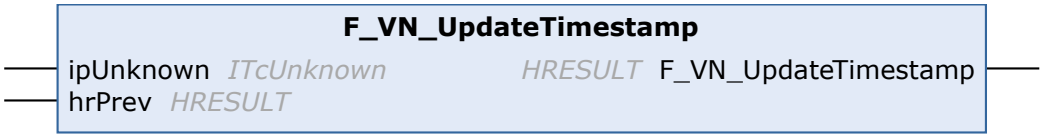
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.22.32 F_VN_UpdateTimestamp



Update the timestamp of an object to the actual DC time (e.g. ITcVnImage or ITcVnContainer)

语法

定义：

```
FUNCTION F_VN_UpdateTimestamp : HRESULT
VAR_INPUT
    ipUnknown : ITcUnknown;
    hrPrev    : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipUnknown	ITcUnknown [▶ 424]	实现 ITcUnknown 和 ITcVnTimestamp 的对象（例如 ITcVnImage 或 ITcVnContainer）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.23 神经网络

该组包含用于预处理和后处理以及执行神经网络模型的函数。

函数

预处理和后处理

- F_VN_ConvertDataLayout [▶ 1549]
- F_VN_ReshapeImage [▶ 1557]

执行神经网络

- F_VN_ExecuteNeuralNetwork [▶ 1550]
  具有一个输入和一个输出。
- F_VN_ExecuteNeuralNetwork_MI [▶ 1543]
  具有多个输入 (MI) 和一个输出。
- F_VN_ExecuteNeuralNetwork_MIMO_String [▶ 1544]
  具有多个输入和输出 (MIMO)。
- F_VN_ExecuteNeuralNetwork_MISO [▶ 1545]
  具有多个输入和单个指定输出 (MISO)。
- F_VN_ExecuteNeuralNetwork_MO [▶ 1546]
  一个输入和多个输出 (MO)。
- F_VN_ExecuteNeuralNetwork_SO [▶ 1546]
  具有一个输入和单个指定输出 (SO)。

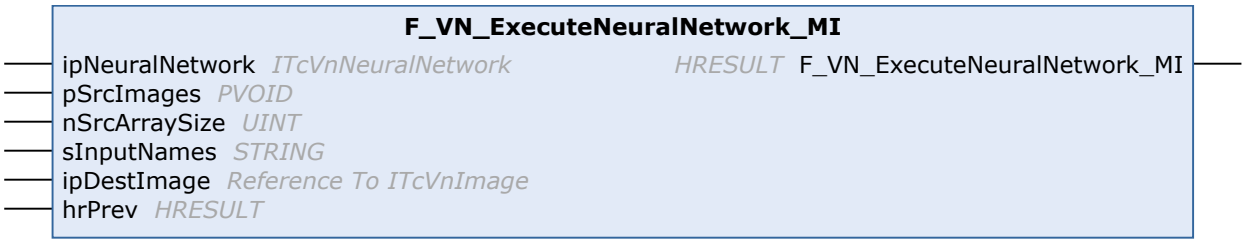
后处理

- F_VN_Sigmoid(Exp) [▶ 1558]
- F_VN_SoftMax [▶ 1560]
- F_VN_NonMaxSuppression(Exp) [▶ 1553]
- F_VN_Reduce [▶ 1555]
- F_VN_ReduceArg [▶ 1556]

获取图层名称

- F_VN_GetNeuralNetworkLayerNames [▶ 1552]
- F_VN_GetNeuralNetworkLayerNamesExp_String [▶ 1548]

6.1.4.23.1 F_VN_ExecuteNeuralNetwork_MI



使用提供的图像，执行具有多个输入（MI）的神经网络。确保所有模型输入名称均在 sInputNames 中列出，且图像和名称顺序一致。返回一个输出层的结果。该函数不执行任何预处理或转换。
可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_ExecuteNeuralNetwork_MI : HRESULT
VAR_INPUT
    ipNeuralNetwork : ITcVnNeuralNetwork;
    pSrcImages      : PVOID;
    nSrcArraySize   : UINT;
    sInputNames     : STRING;
    ipDestImage     : Reference To ITcVnImage;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipNeuralNetwork	ITcVnNeuralNetwork [▶ 426]	神经网络模型
pSrcImages	PVOID	指向源输入图像数组的指针

名称	类型	描述
nSrcArraySize	UINT	pSrcImages 引用的数组的大小。pSrcImages 必须引用一个大小等于模型输入数的数组。
sInputNames	STRING	以 “;” 分隔的输入名称
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（根据输出层创建适当的目标图像。要处理目标图像，必要时可使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

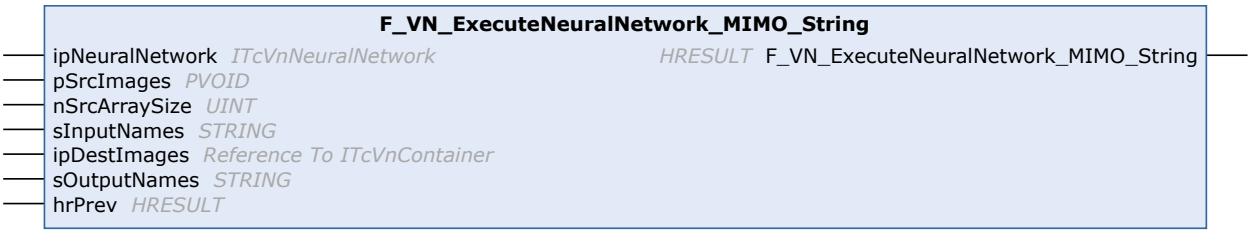
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

6.1.4.23.2 F_VN_ExecuteNeuralNetwork_MIMO_String



使用提供的图像，执行具有多个输入和多个输出（MIMO）的神经网络。确保所有模型输入名称均在 sInputNames 中列出，且图像和名称顺序一致。返回指定输出层的结果。该函数不执行任何预处理或转换。可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_ExecuteNeuralNetwork_MIMO_String : HRESULT
VAR_INPUT
    ipNeuralNetwork : ITcVnNeuralNetwork;
    pSrcImages      : PVOID;
    nSrcArraySize   : UINT;
    sInputNames     : STRING;
    ipDestImages    : Reference To ITcVnContainer;
    sOutputNames    : STRING;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipNeuralNetwork	ITcVnNeuralNetwork [▶ 426]	神经网络模型
pSrcImages	PVOID	指向源输入图像数组的指针
nSrcArraySize	UINT	pSrcImages 引用的数组的大小。pSrcImages 必须引用一个大小等于模型输入数的数组。
sInputNames	STRING	以 “;” 分隔的输入名称
ipDestImages	引用 ITcVnContainer [▶ 365]	包含指定层的输出的图像的容器。
sOutputNames	STRING	需要输出的输出层的名称，以 “;” 分隔。如果为空，则提供所有未连接输出层的输出。

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

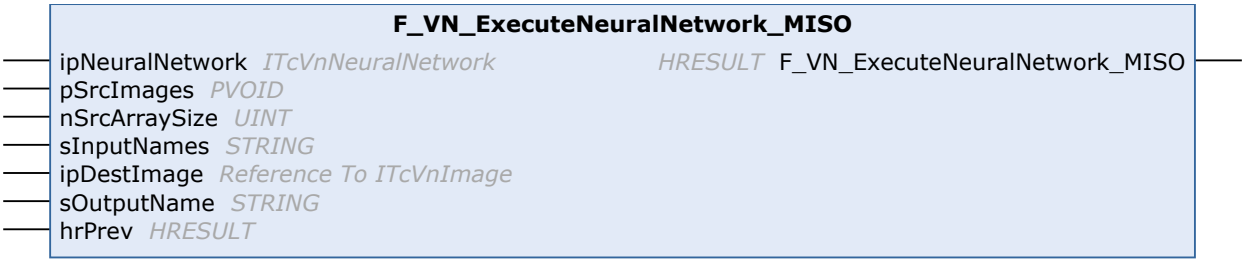
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

6.1.4.23.3 F_VN_ExecuteNeuralNetwork_MISO



使用提供的图像，执行具有多个输入和单个输出（MISO）的神经网络。确保所有模型输入名称均在 sInputNames 中列出，且图像和名称顺序一致。返回指定输出层的结果。该函数不执行任何预处理或转换。可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_ExecuteNeuralNetwork_MISO : HRESULT
VAR_INPUT
    ipNeuralNetwork : ITcVnNeuralNetwork;
    pSrcImages      : PVOID;
    nSrcArraySize   : UINT;
    sInputNames     : STRING;
    ipDestImage     : Reference To ITcVnImage;
    sOutputName     : STRING;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipNeuralNetwork	ITcVnNeuralNetwork [▶ 426]	神经网络模型
pSrcImages	PVOID	指向源输入图像数组的指针
nSrcArraySize	UINT	pSrcImages 引用的数组的大小。pSrcImages 必须引用一个大小等于模型输入数的数组。
sInputNames	STRING	以 “;” 分隔的输入名称
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（根据输出层创建适当的目标图像。要处理目标图像，必要时可使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数。
sOutputName	STRING	指定需要输出的层的名称。如果为空，则提供一个输出层的输出。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

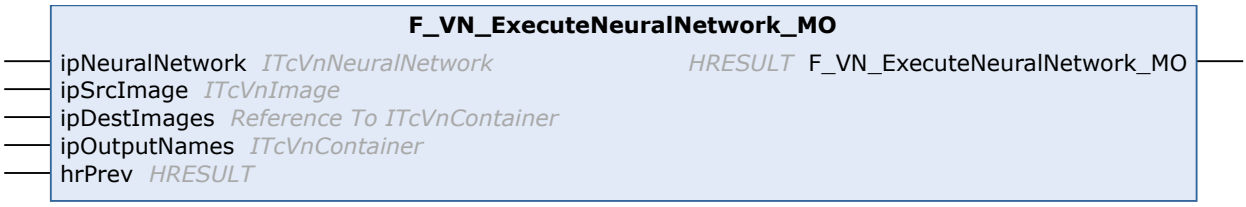
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

6.1.4.23.4 F_VN_ExecuteNeuralNetwork_MO



使用提供的图像作为输入来执行神经网络，并提供指定输出层的输出。该函数不执行任何预处理或转换。可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_ExecuteNeuralNetwork_MO : HRESULT
VAR_INPUT
    ipNeuralNetwork : ITcVnNeuralNetwork;
    ipSrcImage      : ITcVnImage;
    ipDestImages    : Reference To ITcVnContainer;
    ipOutputNames   : ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipNeuralNetwork	ITcVnNeuralNetwork [▶ 426]	神经网络模型
ipSrcImage	ITcVnImage [▶ 407]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）
ipDestImages	引用 ITcVnContainer [▶ 365]	包含指定层的输出的图像的容器。
ipOutputNames	ITcVnContainer [▶ 365]	要返回输出的层名称的容器（ContainerType_Vector_String_SINT）。如果设置为零，则将返回所有未连接输出层的输出。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

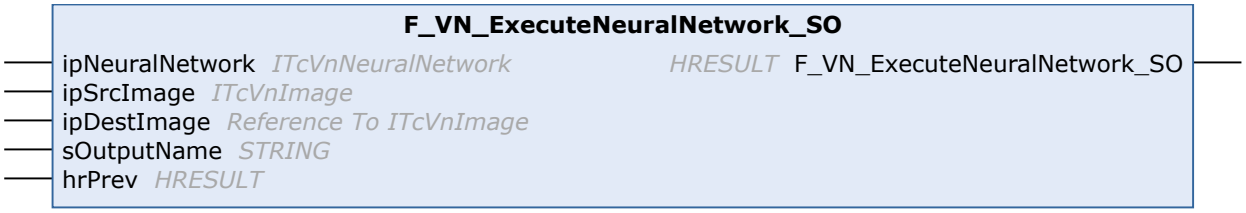
返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

6.1.4.23.5 F_VN_ExecuteNeuralNetwork_SO



使用提供的图像作为输入来执行神经网络，并提供指定输出层的输出。该函数不执行任何预处理或转换。可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_ExecuteNeuralNetwork_SO : HRESULT
VAR_INPUT
    ipNeuralNetwork : ITcVnNeuralNetwork;
    ipSrcImage      : ITcVnImage;
    ipDestImage     : Reference To ITcVnImage;
    sOutputName     : STRING;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipNeuralNetwork	ITcVnNeuralNetwork [▶ 426]	神经网络模型
ipSrcImage	ITcVnImage [▶ 407]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（根据输出层创建适当的目标图像。要处理目标图像，必要时可使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数。
sOutputName	STRING	指定需要输出的层的名称。如果为空，则提供一个输出层的输出。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

6.1.4.23.6 F_VN_GetNeuralNetworkLayerNamesExp



获取神经网络模型中模型层的名称。（专家函数）

语法

定义：

```
FUNCTION F_VN_GetNeuralNetworkLayerNamesExp : HRESULT
VAR_INPUT
    ipNeuralNetwork : ITcVnNeuralNetwork;
    ipLayerNames    : Reference To ITcVnContainer;
    eLayerNameType  : ETcVnLayerNameType;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipNeuralNetwork	ITcVnNeuralNetwork [▶ 426]	神经网络模型
ipLayerNames	引用 ITcVnContainer [▶ 365]	带有层名称的容器（ContainerType_Vector_String_SINT）
eLayerNameType	ETcVnLayerNameType [▶ 201]	层名称类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

6.1.4.23.7 F_VN_GetNeuralNetworkLayerNamesExp_String



获取神经网络模型中模型层的名称。（专家函数）

语法

定义：

```
FUNCTION F_VN_GetNeuralNetworkLayerNamesExp_String : HRESULT
VAR_INPUT
    ipNeuralNetwork : ITcVnNeuralNetwork;
    sLayerNames     : STRING;
    nMaxLength      : ULINT;
    eLayerNameType  : ETcVnLayerNameType;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipNeuralNetwork	ITcVnNeuralNetwork [▶ 426]	神经网络模型
sLayerNames	STRING	以 “;” 分隔的层名称。确保选择足够的字符串大小！
nMaxLength	ULINT	要导出的字符串的最大长度（包括 0 结尾）。如果容器内容较长，则在 nMaxLength - 1 处截断字符串，并添加 0 结尾。在这种情况下，将返回 S_FALSE。
eLayerNameType	ETcVnLayerNameType [▶ 201]	层名称类型
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

6.1.4.23.8 F_VN_ConvertDataLayout



Convert a data layout (the arrangement of dimensions in the input data) to another data layout.

语法

定义：

```
FUNCTION F_VN_ConvertDataLayout : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    ipDestImage     : Reference To ITcVnImage;
    nSrcDataLayout  : ULINT;
    nDestDataLayout : ULINT;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
nSrcDataLayout	ULINT	源图像的数据布局（在枚举 ETcVnDataLayout 中提供常用数据布局。如果 nSrcDataLayout 被设置为 0 或 TCVN_DL_DEFAULT，则 DataLayout 将根据 ipSrcImage 的维度自动确定：0x12 表示 2D，0x123 表示 3D，0x1234 表示 4D，以此类推。）
nDestDataLayout	ULINT	所需的输出数据布局（在枚举 ETcVnDataLayout 中提供常用数据布局。nDestDataLayout 不得等于 0 或 TCVN_DL_DEFAULT。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

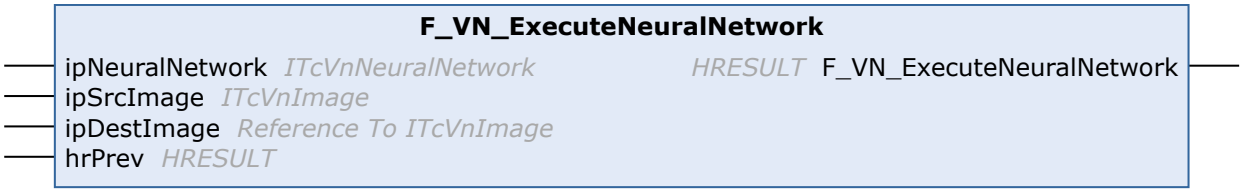
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.23.9 F_VN_ExecuteNeuralNetwork



Execute a neural network using the provided image as input and provide the output of one output layer. The function doesn't perform any pre-processing or transformation.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_ExecuteNeuralNetwork : HRESULT
VAR_INPUT
    ipNeuralNetwork : ITcVnNeuralNetwork;
    ipSrcImage      : ITcVnImage;
    ipDestImage     : Reference To ITcVnImage;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipNeuralNetwork	ITcVnNeuralNetwork [▶ 426]	神经网络模型
ipSrcImage	ITcVnImage [▶ 407]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（根据输出层创建适当的目标图像。要处理目标图像，必要时可使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

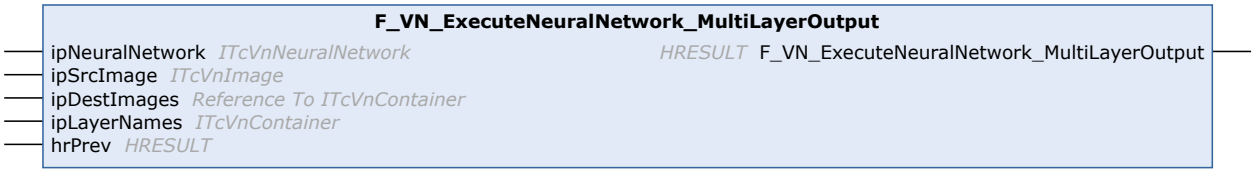
所需的许可证

TC3 Neural Network Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.23.10 F_VN_ExecuteNeuralNetwork_MultiLayerOutput



This function is deprecated, please use F_VN_ExecuteNeuralNetwork_MO. Execute a neural network using the provided image as input and provide the output of the specified output layers. The function doesn't perform any pre-processing or transformation.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_ExecuteNeuralNetwork_MultiLayerOutput : HRESULT
VAR_INPUT
    ipNeuralNetwork : ITcVnNeuralNetwork;
    ipSrcImage      : ITcVnImage;
    ipDestImages    : Reference To ITcVnContainer;
    ipLayerNames    : ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipNeuralNetwork	ITcVnNeuralNetwork [▶ 426]	神经网络模型
ipSrcImage	ITcVnImage [▶ 407]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）
ipDestImages	引用 ITcVnContainer [▶ 365]	包含指定层的输出的图像的容器。
ipLayerNames	ITcVnContainer [▶ 365]	要返回输出的层名称的容器（ContainerType_Vector_String_SINT）。如果设置为零，则将返回所有未连接输出层的输出。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

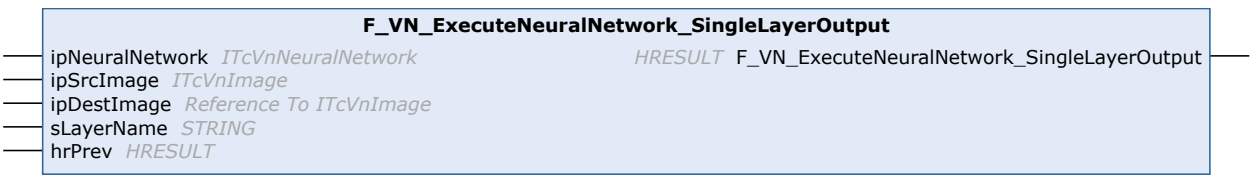
所需的许可证

TC3 Neural Network Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.23.11 F_VN_ExecuteNeuralNetwork_SingleLayerOutput



This function is deprecated, please use F_VN_ExecuteNeuralNetwork_SO. Execute a neural network using the provided image as input and provide the output of the specified output layer. The function doesn't perform any pre-processing or transformation.
Can use available TwinCAT Job Tasks for executing parallel code regions.

语法

定义：

```
FUNCTION F_VN_ExecuteNeuralNetwork_SingleLayerOutput : HRESULT
VAR_INPUT
    ipNeuralNetwork : ITcVnNeuralNetwork;
    ipSrcImage      : ITcVnImage;
    ipDestImage     : Reference To ITcVnImage;
    sLayerName      : STRING;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipNeuralNetwork	ITcVnNeuralNetwork [▶ 426]	神经网络模型
ipSrcImage	ITcVnImage [▶ 407]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（根据输出层创建适当的目标图像。要处理目标图像，必要时可使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数。
sLayerName	STRING	指定需要输出的层的名称。如果为空，则提供一个输出层的输出。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

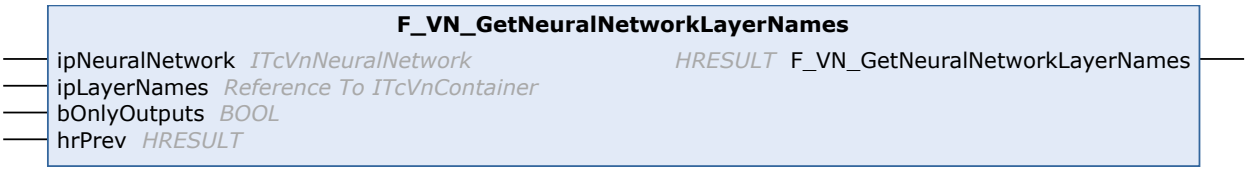
所需的许可证

TC3 Neural Network Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.23.12 F_VN_GetNeuralNetworkLayerNames



Get the name of all layers in the neural network model. If the bOnlyOuput flag is set to true, only the unconnected output layer names are returned.

语法

定义：

```
FUNCTION F_VN_GetNeuralNetworkLayerNames : HRESULT
VAR_INPUT
    ipNeuralNetwork : ITcVnNeuralNetwork;
    ipLayerNames    : Reference To ITcVnContainer;
    bOnlyOutputs    : BOOL;
    hrPrev          : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipNeuralNetwork	ITcVnNeuralNetwork [▶ 426]	神经网络模型
ipLayerNames	引用 ITcVnContainer [▶ 365]	带有层名称的容器（ContainerType_Vector_String_SINT）
bOnlyOutputs	BOOL	如果为 true，则仅返回未连接输出层的名称。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

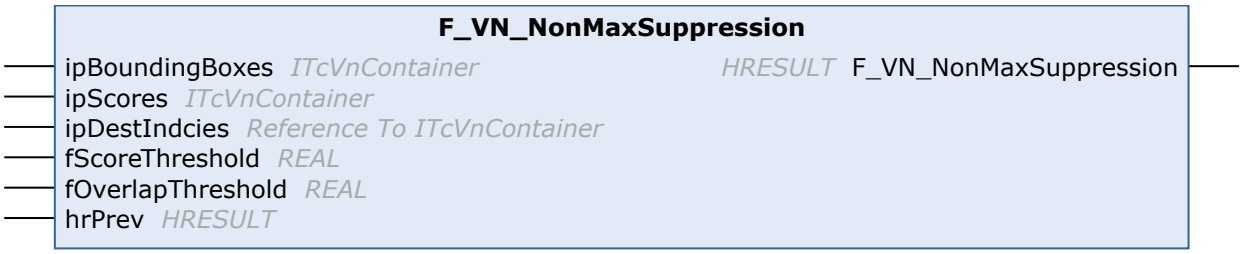
所需的许可证

TC3 Neural Network Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.23.13 F_VN_NonMaxSuppression



Perform non-maximum suppression on given bounding boxes and corresponding scores.

语法

定义：

```
FUNCTION F_VN_NonMaxSuppression : HRESULT
VAR_INPUT
    ipBoundingBoxes   : ITcVnContainer;
    ipScores          : ITcVnContainer;
    ipDestIndcies     : Reference To ITcVnContainer;
    fScoreThreshold   : REAL;
    fOverlapThreshold : REAL;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipBoundingBoxes	ITcVnContainer [▶ 365]	应用非极大值抑制的边界框的容器（ContainerType_Vector_TcVnRectangle_DINT）
ipScores	ITcVnContainer [▶ 365]	每个边界框的相应分数的容器（ContainerType_Vector_REAL）
ipDestIndcies	引用 ITcVnContainer [▶ 365]	应用非极大值抑制后所保留索引的容器（ContainerType_Vector_DINT）

名称	类型	描述
fScoreThreshold	REAL	用于按分数筛选边界框的阈值 [0.0 至 1.0]
fOverlapThreshold	REAL	用于确定边界框被视为重叠所需的最小交并比度（IoU）的阈值 [0.0 至 1.0]
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

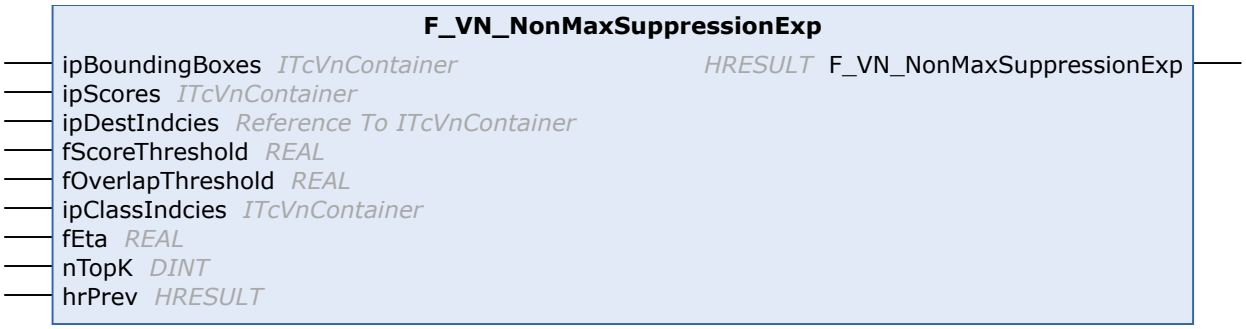
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.23.14 F_VN_NonMaxSuppressionExp



Perform non-maximum suppression on given bounding boxes and corresponding scores. (expert function)

语法

定义：

```
FUNCTION F_VN_NonMaxSuppressionExp : HRESULT
VAR_INPUT
    ipBoundingBoxes   : ITcVnContainer;
    ipScores          : ITcVnContainer;
    ipDestIndcies     : Reference To ITcVnContainer;
    fScoreThreshold   : REAL;
    fOverlapThreshold : REAL;
    ipClassIndcies    : ITcVnContainer;
    fEta              : REAL;
    nTopK             : DINT;
    hrPrev            : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipBoundingBoxes	ITcVnContainer [▶ 365]	应用非极大值抑制的边界框的容器（ContainerType_Vector_TcVnRectangle_DINT）
ipScores	ITcVnContainer [▶ 365]	每个边界框的相应分数的容器（ContainerType_Vector_REAL）

名称	类型	描述
ipDestIndcies	引用 ITcVnContainer [▶ 365]	应用非极大值抑制后所保留索引的容器（ContainerType_Vector_DINT）
fScoreThreshold	REAL	用于按分数筛选边界框的阈值 [0.0 至 1.0]
fOverlapThreshold	REAL	用于确定边界框被视为重叠所需的最小交并比度（IoU）的阈值 [0.0 至 1.0]
ipClassIndcies	ITcVnContainer [▶ 365]	相应类索引集合的容器（ContainerType_Vector_DINT）
fEta	REAL	在处理过程中用于动态降低重叠阈值的可调阈值 [0.0 至 1.0]。将其设置为 1 可保持重叠阈值的常量值
nTopK	DINT	采用抑制算法的最高分数的边界框的最大数量（必须 >= 0）。将其设置为 0 可将算法应用于所有边界框
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

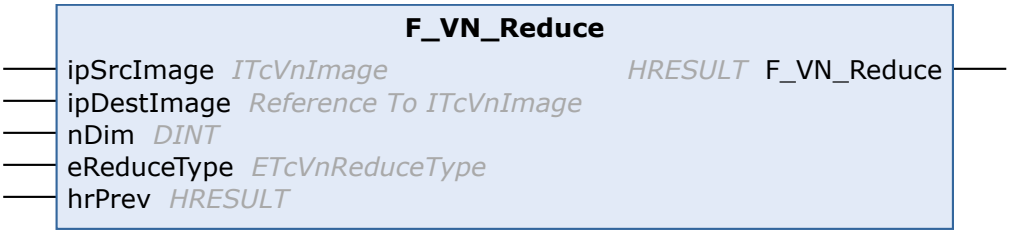
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.23.15 F_VN_Reduce



Reduce ipSrcImage by applying the specified operation across rows/columns/channels to obtain a single row/column/channel with the resulting values.

语法

定义：

```
FUNCTION F_VN_Reduce : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    nDim        : DINT;
    eReduceType : ETcVnReduceType;
    hrPrev      : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipSrcImage	ITcVnImage [▶ 407]	Source image (USINT or REAL elements). Must be single-channel when nDim is 0 or 1. When nDim is 2, multi-channel images are supported.
ipDestImage	Reference To ITcVnImage [▶ 407]	Destination image. If ipSrcImage is of type USINT and the reduction type is SUM or AVG, ipDestImage will be of type REAL; otherwise, it retains the same type as ipSrcImage (An appropriate single channel destination image will be created if required.)
nDim	DINT	Index along which the ITcVnImage is reduced: 0 reduces the matrix to a single row, 1 reduces it to a single column, and 2 reduces to a single channel
eReduceType	ETcVnReduceType [▶ 208]	Reduction operation
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.23.16 F_VN_ReduceArg



Finds indices of max/min elements along provided axis and reduce the ipSrcImage to a single row/column/channel.

语法

定义：

```
FUNCTION F_VN_ReduceArg : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    nDim        : DINT;
    eReduceType : DINT;
    hrPrev      : HRESULT;
END_VAR
```

Inputs

Name	Type	Description
ipSrcImage	ITcVnImage [▶ 407]	Source image. Must be single-channel when nDim is 0 or 1. When nDim is 2, multi-channel images are supported.
ipDestImage	Reference To ITcVnImage [▶ 407]	Destination image of type DINT (An appropriate single channel destination image will be created if required.)
nDim	DINT	Index along which the ipSrcImage is reduced: 0 reduces the matrix to a single row, 1 reduces it to a single column, and 2 reduces to a single channel
eReduceType	DINT	Reduction operation REDUCE_MAX or REDUCE_MIN
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.23.17 F_VN_ReshapeImage



Changes the shape of an ITcVnImage. The total number of elements in the new shape must be equal to the total number of elements in ipSrcImage.

语法

定义：

```
FUNCTION F_VN_ReshapeImage : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    aNewShape   : TcVnArray33_UDINT;
END_VAR
VAR_INPUT
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的单通道目标图像。）

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输入/输出

名称	类型	描述
aNewShape	TcVnArray33_UDINT [▶ 153]	指定 ipDestImage 的形状。第一个条目是维数（最多 32），然后是每个维度的大小。如果一个维度被设置为零，其值将根据 ipSrcImage 的元素总数自动计算（仅能有一个维度被设置为零。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.23.18 F_VN_Sigmoid



Apply the standard logistic function 1 / (1 + exp(-x)) element-wise to a single- or multi-channel REAL image.

语法

定义：

```
FUNCTION F_VN_Sigmoid : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（REAL）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.23.19 F_VN_SigmoidExp



Apply the logistic function 1 / (1 + exp(-fSteepness * (x - fShift))) element-wise to a single- or multi-channel REAL image. (expert function)

语法

定义：

```
FUNCTION F_VN_SigmoidExp : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    fShift      : REAL;
    fSteepness  : REAL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（REAL）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的目标图像。）
fShift	REAL	确定逻辑函数转换的强度级别。
fSteepness	REAL	控制转换的清晰度。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.23.20 F_VN_SoftMax



Apply softmax function pixel-wise to a single channel REAL image.

语法

定义：

```
FUNCTION F_VN_SoftMax : HRESULT
VAR_INPUT
    ipSrcImage  : ITcVnImage;
    ipDestImage : Reference To ITcVnImage;
    bFlatten    : BOOL;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（REAL，1 通道）
ipDestImage	引用 ITcVnImage [▶ 407]	目标图像（如果需要，将创建适当的单通道目标图像。）
bFlatten	BOOL	如果为 true，生成的图像将被重塑为单行。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.24 OCR

该组包含**光学**字符**识别**功能。

OCR 功能 [▶ 1563]识别图像中的字符，并将识别出的字符作为字符串返回。分类基于经典 Machine Learning（机器学习）模型。这些模型以训练形式提供，因此可以直接使用功能。无需额外设置或训练。支持的字符和字体取决于训练中使用的数据。因此，有多种模型，其功能范围以及一般前提条件和要求如下所述。

对于图像中识别出的所有字符，模型会将每个字符分配到匹配度最高的已知类别。因此，结果总是包含所使用模型的已知字符。不包括拒绝或删除未知字符。

函数仅支持单行字符串；必须将多行字符串分为单独的 ROI，并通过多次函数调用读取。

识别字符的一般要求

- 字符高度最小为 20 像素

- 线宽最小为 3 像素
- 点最小为 3 x 3 像素
- 线最小为 3 x 6 像素
- 字符间距/分隔最小为 4 像素
- 字符不得重叠
- 仅水平对齐/字符排列最大 ± 6° 偏差
- 字符线/轮廓不得中断
- 字符不得镜像或颠倒

图像的一般要求：

- ROI 只包含待识别字符和字符周围的无干扰区
- 在 ROI 内，字符不得被其他轮廓（如矩形）包围
- 字符与背景要对比鲜明
- 背景均匀、无噪音或干扰、不透明

对字体的要求：

- 只能使用字宽相等的比例字体
- 较大的间隙始终会被识别为单个空格
- 仅使用无衬线字体，如 Arial、Tahoma、Courier、Univers、Frutiger、Verdana、OCR-B 等
- 无混合字体
- 无圆点打印字体或斜体字体

模型：

枚举 ETcVnOcrModelType [▶ 204] 允许访问以下模型：

- `TCVN_OMT_NUMBERS`
  - 支持数字分类
  - 包含 `0-9` 的字符
- `TCVN_OMT_NUMBERS_SC`
  - 支持数字和特殊字符分类
  - 包含从 `0-9` 开始的字符和六个特殊字符 `. / - : = +`
- `TCVN_OMT_UCLETTERS`
  - 支持大写字母分类
  - 包含 `A-Z` 的字符
- `TCVN_OMT_NUMBERS_SC_UCLETTERS`
  - 支持数字、特殊字符和大写字母分类
  - 包含从 `0-9` 开始的字符、六个特殊字符 `. / - : =` + 和 `A-Z`

在使用组合模型（`TCVN_OMT_NUMBERS_SC_UCLETTERS`）时，由于某些字符非常相似，可能会出现混淆。例如 O 和 0、S 和 5 以及 B 和 8。

如果数字或字母的位置已知，则应使用具有默认格式的专家函数（`sPattern`）和单独模型的组合作为替代。另外，也可以单独分析结果字符串（`ipCharacters`）的字符，这样就可以单独决定是否接受 0 而非 O。

初始化功能

为了能够将 OCR 功能与一个或多个模型一起使用，必须首先使用功能块 FB_VN_InitializeFunction [▶ 1584] 初始化该功能以及相应的模型。

加载模型所需的时间取决于所使用 IPC 的性能和模型大小，可能需要几百毫秒甚至几秒钟。因此，在任务循环时间较短的情况下，预计会出现循环时间超限。

如果要使用组合模型 TCVN_OMT_NUMBERS_SC_UCLETTERS，由于文件大小的原因，路由器内存必须至少设置为 512 MB。

具体大小取决于路由器内存的其他用途和任何现有分段储存。例如，如果功能块 FB_VN_InitializeFunction 返回返回代码 0x80004005，则表示路由器内存不足。因此，应首先检查路由器内存，必要时增加内存。加载模型后，大部分内存会重新释放，供应用使用。

HRESULT 的解释

- 如果识别到图像上的字符，标准函数返回 S_OK。对于 Exp 函数，这取决于是否指定了 sPattern。如果没有定义格式规格，则返回对应于标准函数。如果已传递格式规格，只会在识别字符与模板规格相匹配的情况下才会返回 S_OK。
- 如果标准函数执行成功，但未找到任何字符，或者 Exp 函数中的字符与模板规格不匹配，则返回 S_FALSE。
- 如果在专家函数中输入了 sPattern，但缺少转移模型，则会返回 HRESULT = ADSERR_DEVICE_NOTINIT。
- 如果向标准函数传递了多个模型，则始终返回 HRESULT = ADSERR_DEVICE_INVALIDPARM，因为该函数仅支持每次调用一个模型。若为 Expert（专家）函数，这取决于是否指定了 sPattern。如果 sPattern 为空，则只使用 TCVN_OMT_NUMBERS_SC_UCLETTERS 模型。如果没有传递该模型，则返回 HRESULT = ADSERR_DEVICE_INVALIDPARM。

使用 sPattern 的示例

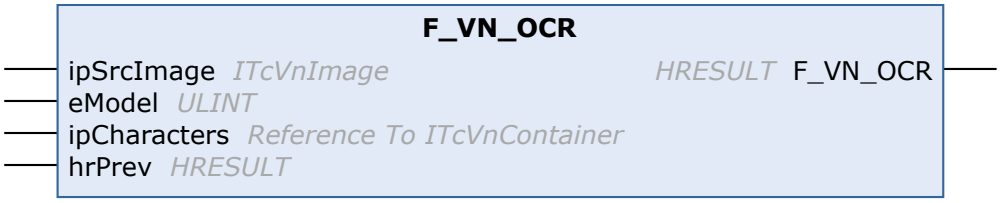
F_VN_OCRExp [▶ 1564] 通过参数 sPattern 和 eOcrOptions 提供其他选项。根据输入图像上的字符组合、来自 sPattern 和 eOcrOptions 的信息以及识别的字符，如果出现不匹配，则返回 S_FALSE。然后，还可以对结果字符串（ipCharacters）中的字符进行单独分析，例如，如果长度不同，也会返回 S_FALSE。这些样本涉及 TCVN_OMT_NUMBERS_SC_UCLETTERS 模型的使用。

ipSrcImage 上的字符	sPattern	eOcrOptions	ipCharacters	HRESULT
12/34	dd.dd		12/34	S_OK
12534	dd.dd		12534	S_OK
12/34	dd!dd		1234	S_OK
12 34	dd!d		124	S_OK
12534	dd!dd		1234	S_OK
12 34	dd_dd		12 34	S_OK
AB12/	uudd#		AB12/	S_OK
12 34	dddd	WITHBLANKS	12 34	S_OK
12/4	dd#d	WITHBLANKS	12/4	S_OK
12 34	dd_dd	WITHBLANKS	12 34	S_OK
12 34 56	dd_dddd	WITHBLANKS	12 34 56	S_OK
12 3 4	dd.dd	WITHBLANKS	12 3 4	S_FALSE
12 34	dd!dd		124	S_FALSE
12 34	dd.dd		1234	S_FALSE
1234	!dddd		234	S_FALSE
12 34	dd_dd		12.34	S_FALSE
12 3	dd_dd		123	S_FALSE
12 3 4	dd_d_d		123 4	S_FALSE
AB12/	uddd#		AB12/	S_FALSE
AB12/	uuddd		AB12/	S_FALSE

还请参阅有关此

- OCR [▶ 2810]

6.1.4.24.1 F_VN_OCR



检测并识别二值图像中的字符（黑底白字）。
可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_OCR : HRESULT
VAR_INPUT
    ipSrcImage   : ITcVnImage;
    eModel       : ULINT;
    ipCharacters : Reference To ITcVnContainer;
    hrPrev       : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道，二值）
eModel	ULINT	选择 OCR 模型类型（ETcVnOcrModelType）
ipCharacters	引用 ITcVnContainer [▶ 365]	返回识别的字符（ContainerType_Vector_String_SINT）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_OCR` 检测和识别二进制图像中的字符，并以容器形式返回识别出的字符。

参数

输入图像

传输到 `ipSrcImage` 的输入图像必须是单通道二进制图像。只允许使用黑底白字。

模型

必须将 ETcVnOcrModelType [▶ 204] 的模型类型传输到 `eModel`，该类型将用于对字符进行分类的函数调用。该函数仅支持每次调用使用一个模型。

要使用模型，首先必须使用功能块 FB_VN_InitializeFunction [▶ 1584] 对其进行初始化。

识别字符

识别字符通过引用 `ipCharacters` 返回。

专家参数

专家变体 F_VN_OCRExp [▶ 1565] 包含附加参数。

应用

例如，调用 OCR 函数，使用模型识别数字并在随后导出字符串的过程如下所示：

```
hr := F_VN_OCR(
    ipSrcImage   := ipBinaryImage,
    eModel       := TCVN_OMT_NUMBERS,
    ipCharacters := ipCharactersResults,
    hrPrev       := hr);

// Export character to string
hr := F_VN_ExportSubContainer_String(ipCharactersResults, 0, sText, 255, hr);
```

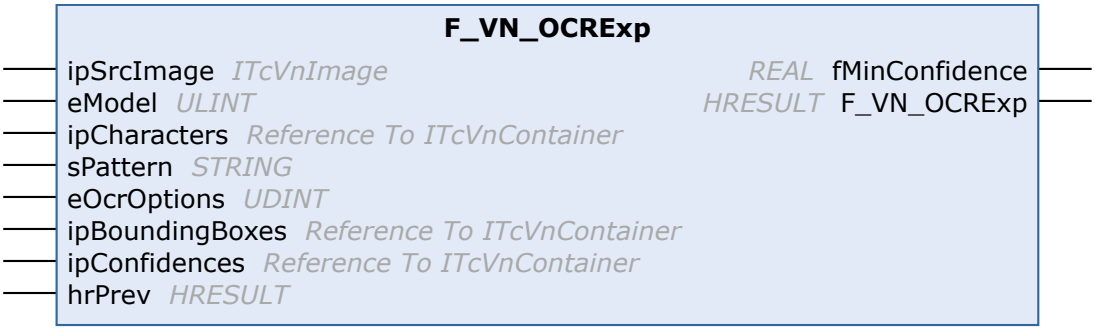
所需的许可证

TC3 Vision OCR

还请参阅有关此

OCR [▶ 2810]

6.1.4.24.2 F_VN_OCRExp



检测并识别二值图像中的字符（黑底白字）。（专家函数）
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
FUNCTION F_VN_OCRExp : HRESULT
VAR_INPUT
    ipSrcImage      : ITcVnImage;
    eModel          : ULINT;
    ipCharacters    : Reference To ITcVnContainer;
    sPattern        : STRING;
    eOcrOptions     : UDINT;
    ipBoundingBoxes : Reference To ITcVnContainer;
    ipConfidences   : Reference To ITcVnContainer;
    hrPrev          : HRESULT;
END_VAR
VAR_OUTPUT
    fMinConfidence  : REAL;
END_VAR
```

输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT，1 通道，二值）
eModel	ULINT	选择 OCR 模型类型（ETcVnOcrModelType）
ipCharacters	引用 ITcVnContainer [▶ 365]	返回识别的字符（ContainerType_Vector_String_SINT）
sPattern	STRING	包含字符显示格式的字符串模式
eOcrOptions	UDINT	指定必须对函数应用的选项（ETcVnOcrOptions）

名称	类型	描述
ipBoundingBoxes	引用 ITcVnContainer [▶ 365]	返回已识别字符的相应边界框（ContainerType_Vector_TcVnRectangle_DINT，可选，如果不需要，则设为 0）
ipConfidences	引用 ITcVnContainer [▶ 365]	返回已识别字符的相应分类置信度（ContainerType_Vector_REAL，可选，如果不需要，则设为 0）
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

输出

名称	类型	描述
fMinConfidence	REAL	返回置信度的最小值

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_OCRExp` 是 F_VN_OCR [▶ 1563] 的 Expert（专家）变体。它包含额外的参数。

参数

输入图像

传输到 `ipSrcImage` 的输入图像必须是单通道二进制图像。只允许使用黑底白字。

模型

在函数调用中用于对字符进行分类的 ETcVnOcrModelType [▶ 204] 模型类型必须传输到 `eModel`。要在一次函数调用中使用多个模型，必须通过 `sPattern` 指定字符串模式。这些模型可通过 `OR` 进行链接。

要使用单个模型，必须先使用功能块 FB_VN_InitializeFunction [▶ 1584] 对其进行初始化。

识别字符

识别字符通过引用 `ipCharacters` 返回。

字符串模式

带有待识别预期格式字符串的字符串模式可通过 `sPattern` 传输。

支持的格式及其含义：

- `!`（感叹号）表示忽略字符（不适用于空格）
- `.`（点）表示任何字符（不适用于空格）
  只有 TCVN_OMT_NUMBERS_SC_UCLETTERS 模型支持使用点，否则点的效果与感叹号相同，并会输出 S_FALSE。
- `_`（下划线）表示空格（不依赖 `eOcrOptions`）
- `d`（数字）表示 `0-9` 的预期数字
- `#`（特殊字符）表示预期特殊字符，支持 `. / - : = +`
- `u`（大写字母）表示 `A-Z` 的预期大写字母

选项

通过 eOcrOptions 可以指定影响功能和结果输出的选项（ETcVnOcrOptions [▶ 204]）。如果不需要任何选项，则必须传递 0。

边界框

引用 ipBoundingBoxes 可选用于返回识别字符的相应边界框。如果不需要边界框，则可以传输值 0。

置信度

引用 ipConfidences 可选用于返回识别字符的相应分类置信度。如果不需要置信度，则可以传输值 0。

最小置信度

可选返回值 fMinConfidence 返回找到的所有字符的最小置信度值。

应用

例如，调用 OCRExp 函数，使用两个模型识别数字和大写字母并随后导出字符串的过程如下所示：

```
sPattern := 'dd!uuu!dddd'; // e.g. 02-FEB-2024

hr := F_VN_OCRExp(
    ipSrcImage      := ipBinaryImage,
    eModel          := TCVN_OMT_NUMBERS OR TCVN_OMT_UCLETTERS,
    ipCharacters    := ipCharactersResults,
    sPattern        := sPattern,
    eOcrOptions     := eOcrOptions,
    ipBoundingBoxes := ipBoundingBoxes,
    ipConfidences   := ipConfidences,
    hrPrev          := hr,
    fMinConfidence  => fMinConfidence);

// Export character to string
hr := F_VN_ExportSubContainer_String(ipCharactersResults, 0, sText, 255, hr);

// Get the bounding box of the first element
hr := F_VN_GetAt_TcVnRectangle_DINT(ipBoundingBoxes, stRectangle, 0, hr);

// Get the Confidence value of the first element
hr := F_VN_GetAt_REAL(ipConfidences, fConfidence, 0, hr);
```

要评估置信度值或绘制边界框，可以通过 GetAt 访问容器的各个元素。但是，如果要访问容器的几个或所有元素，建议编写一个循环程序或将 ipConfidences 容器完全导出为数组。

所需的许可证

TC3 Vision OCR

还请参阅有关此

OCR [▶ 2810]

6.1.4.25 纹理分析

该组包含用于分析图像纹理的函数。

函数

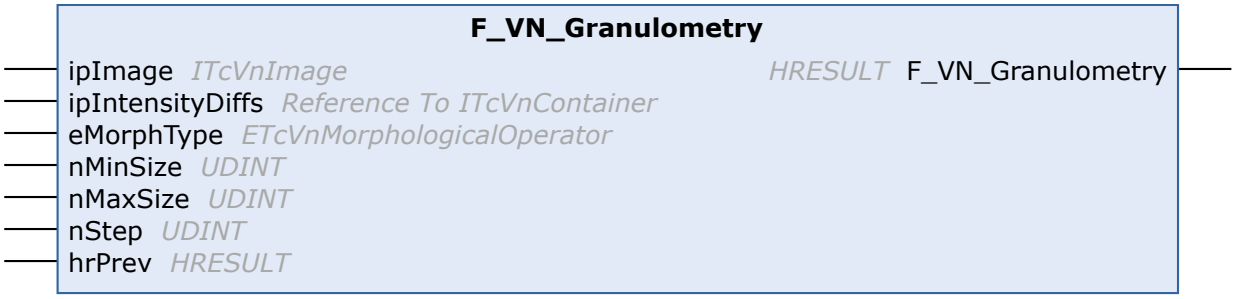
粒度属性

- F_VN_Granulometry [▶ 1567]

Haralick 特征

- F_VN_HaralickFeatures [▶ 1568]

6.1.4.25.1 F_VN_Granulometry



使用形态学运算计算纹理图像的粒度。

语法

定义：

```
FUNCTION F_VN_Granulometry : HRESULT
VAR_INPUT
    ipImage          : ITcVnImage;
    ipIntensityDiffs : Reference To ITcVnContainer;
    eMorphType       : ETcVnMorphologicalOperator;
    nMinSize         : UDINT;
    nMaxSize         : UDINT;
    nStep            : UDINT;
    hrPrev           : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	源图像（1 通道）
ipIntensityDiffs	引用 ITcVnContainer [▶ 365]	返回连续结构元素大小（CTcVnContainer_Vector_LREAL）之间的强度差异，可用作纹理描述的特征。
eMorphType	ETcVnMorphologicalOperator [▶ 202]	仅支持开运算或闭运算。
nMinSize	UDINT	最小结构元素大小（奇数，>= 3）
nMaxSize	UDINT	最大结构元素大小（>= nMinSize）
nStep	UDINT	步长，nMinSize 按此步长递增，直至超过 nMaxSize（偶数，>= 2）。
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_Granulometry` 计算图像的粒度属性。这些值可以显示不同图像区域的大小分布。

参数

图像

输入图像 `ipImage`。

特征

粒度特征通过引用 `ipIntensityDiffs` 作为容器返回。

算子类型

形态算子的类型通过 eMorphType 定义为 TCVN_MO_OPENING 或 TCVN_MO_CLOSING。

算子大小

形态算子的大小由最小 nMinSize、最大 nMaxSize 和步长 nStep 决定。算子的大小必须是奇数，步长必须是偶数。

应用

例如，可如下使用 3、7 和 11 大小的闭合算子计算粒度特征：

```
hr := F_VN_Granulometry(
    ipImage             :=  ipImage,
    ipIntensityDiffs    :=  ipIntensityDiffs,
    eMorphType          :=  TCVN_MO_CLOSING,
    nMinSize            :=  3,
    nMaxSize            :=  11,
    nStep               :=  4,
    hrPrev              :=  hr);
```

相关函数

- F_VN_HaralickFeatures [▶ 1568]

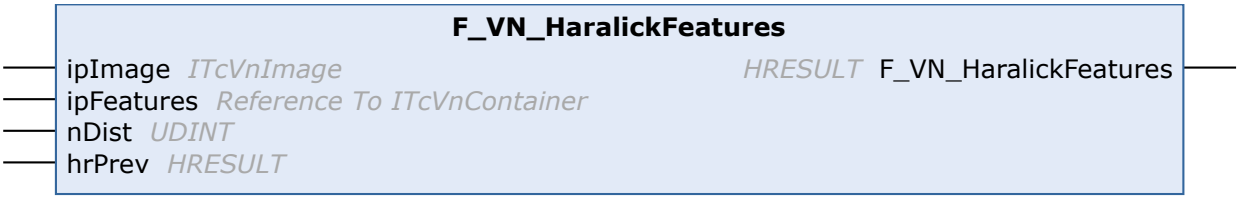
所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.4.25.2 F_VN_HaralickFeatures



计算灰度图像的哈拉利克（Haralick）特征，该特征描述纹理。

语法

定义：

```
FUNCTION F_VN_HaralickFeatures : HRESULT
VAR_INPUT
    ipImage     : ITcVnImage;
    ipFeatures  : Reference To ITcVnContainer;
    nDist       : UDINT;
    hrPrev      : HRESULT;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	源图像（1 通道，USINT）
ipFeatures	引用 ITcVnContainer [▶ 365]	返回计算出的哈拉利克（Haralick）特征（CTcVnContainer_Vector_LREAL）。返回的特征集包含前 13 个哈拉利克（Haralick）特征中每个特征的平均值和标准偏差，

名称	类型	描述
		分别按 0、45、90 和 135 度计算。因此，总共将有 26 个特征（f1 平均值、f1 标准偏差、f2 平均值、f2 标准偏差、...、f13 平均值、f13 标准偏差）。
nDist	UDINT	Distance
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

返回值

HRESULT [▶ 135]

更多信息

函数 `F_VN_HaralickFeatures` 计算图像的 Haralick 属性。这些值指示图像的结构。

参数

图像

输入图像 `ipImage`。

特征

Haralick 特征作为一个容器通过引用 `ipFeatures` 返回。

距离

灰度共生矩阵（GLCM）中相邻像素之间的距离指定为 `nDist`。

应用

例如，可如下计算距离为 5 的 Haralick 特征：

```
hr := F_VN_HaralickFeatures(
    ipImage     :=  ipImage,
    ipFeatures  :=  ipFeatures,
    nDist       :=  5,
    hrPrev      :=  hr);
```

相关函数

- F_VN_Granulometry [▶ 1567]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5 功能块

6.1.5.1 许可证概述

以下列表显示了 TwinCAT Vision 功能块与可用许可证的分配情况（见许可证模型 [▶ 12]）。它显示了某些功能块的应用需要哪些许可证。

始终需要基本许可证 TC3 Vision Base（TF7100）。

TC3 Vision Base

FB_VN_FileSourceControl [▶ 1601]

FB_VN_GevCameraControl [▶ 1609]

FB_VN_InitializeFunction [▶ 1584]

FB_VN_ReadCalibrationPattern [▶ 1586]

FB_VN_ReadCalibrationResult [▶ 1587]

FB_VN_ReadContainer [▶ 1589]

FB_VN_ReadImage [▶ 1590]

FB_VN_ReadMemory [▶ 1571]

FB_VN_ReadMlModel [▶ 1591]

FB_VN_ReadNeuralNetwork [▶ 1598]

FB_VN_ReadRegister_REAL [▶ 1572]

FB_VN_ReadRegister_UDINT [▶ 1573]

FB_VN_ReadRegister_ULINT [▶ 1574]

FB_VN_SimpleCameraControl [▶ 1630]

FB_VN_WriteCalibrationResult [▶ 1593]

FB_VN_WriteContainer [▶ 1595]

FB_VN_WriteImage [▶ 1596]

FB_VN_WriteMemory [▶ 1576]

FB_VN_WriteMlModel [▶ 1597]

FB_VN_WriteRegister_REAL [▶ 1578]

FB_VN_WriteRegister_UDINT [▶ 1579]

FB_VN_WriteRegister_ULINT [▶ 1581]

FB_VN_WriteRegisters_UDINT [▶ 1582]

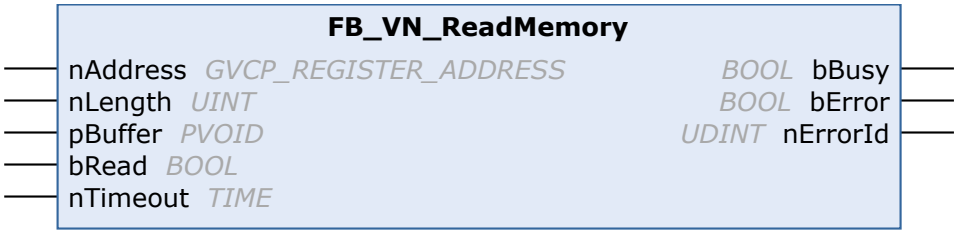
TC3 Vision Matching

FB_VN_GeneralizedHoughBallard [▶ 1633]

FB_VN_SSIM [▶ 1637]

6.1.5.2 相机寄存器访问

6.1.5.2.1 FB_VN_ReadMemory



该 FB 从相机读取连续内存位置。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()

语法

定义：

```
FUNCTION_BLOCK FB_VN_ReadMemory
VAR_INPUT
    nAddress                : GVCP_REGISTER_ADDRESS;
    nLength                 : UINT;
    pBuffer                 : PVOID;
    bRead                   : BOOL;
    nTimeout                : TIME;
END_VAR
VAR_OUTPUT
    bBusy                   : BOOL;
    bError                  : BOOL;
    nErrorId                : UDINT;
END_VAR
VAR
    oidITcVnGevImageProvider : OTCID;
END_VAR
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 152]		从相机读取的第一个字节的地址（必须 32 位对齐）
nLength	UINT		从 nAddress 开始读取的字节数（必须是 4 的倍数，不超过 536）
pBuffer	PVOID		该指针指向读取的内存内容写入的缓冲区（缓冲区的大小必须至少为 nLength 字节！）
bRead	BOOL		读取内存由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

更多信息

需要开放相机控制通道！

GigE Vision 相机的控制通道必须打开，才能使用此功能块。此外，有些相机参数只有在停止图像采集的情况下才能更改。因此，建议首先将相机置于 TCVN_CS_OPENED 状态。

参数

nAddress

寄存器地址标识要寻址的相机寄存器。它属于 GVCP_REGISTER_ADDRESS 类型，是 UDINT 类型的别名。因此，所有寄存器地址的大小都是 4 字节。相机参数的寄存器地址可在 GigE Vision 相机实例配置树中 [▶ 81]的相应点查看：

Property	Value
Name	PixelFormat
Value	BayerRG8
Address	0x00012120
Length	4 Bytes

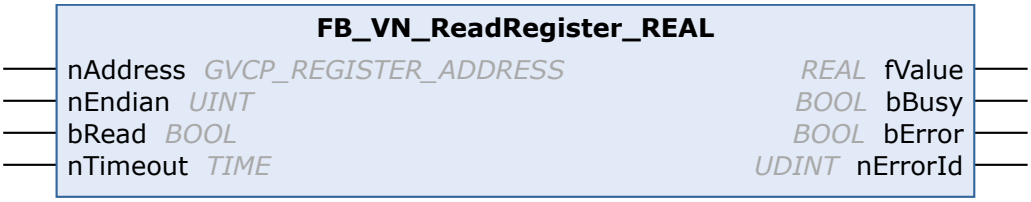
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.2.2 FB_VN_ReadRegister_REAL



该 FB 从相机读取一个 REAL 寄存器。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()）

语法

定义：

```
FUNCTION_BLOCK FB_VN_ReadRegister_REAL
VAR_INPUT
    nAddress                : GVCP_REGISTER_ADDRESS;
    nEndian                 : UINT;
    bRead                   : BOOL;
    nTimeout                : TIME;
END_VAR
VAR_OUTPUT
    fValue                  : REAL;
    bBusy                   : BOOL;
    bError                  : BOOL;
    nErrorId                : UDINT;
END_VAR
VAR
    oidITcVnGevImageProvider : OTCID;
END_VAR
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 152]		应读取的寄存器的地址
nEndian	UINT		可选：寄存器的字节序。0 = 大，1 = 小。默认值：0
bRead	BOOL		读取寄存器由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
fValue	REAL	读取的值
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

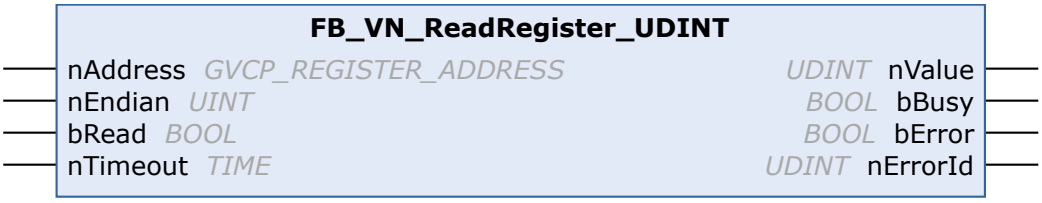
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.2.3 FB_VN_ReadRegister_UDINT



该 FB 从相机读取一个 DINT 寄存器。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()）

语法

定义：

```
FUNCTION_BLOCK FB_VN_ReadRegister_UDINT
VAR_INPUT
    nAddress                : GVCP_REGISTER_ADDRESS;
    nEndian                 : UINT;
    bRead                   : BOOL;
    nTimeout                : TIME;
END_VAR
VAR_OUTPUT
```

```
    nValue                    : UDINT;
    bBusy                     : BOOL;
    bError                    : BOOL;
    nErrorId                  : UDINT;
END_VAR
VAR
    oidITcVnGevImageProvider : OTCID;
END_VAR
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 152]		应读取的寄存器的地址
nEndian	UINT		可选：寄存器的字节序。0 = 大，1 = 小。默认值：0
bRead	BOOL		读取寄存器由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
nValue	UDINT	读取的值
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

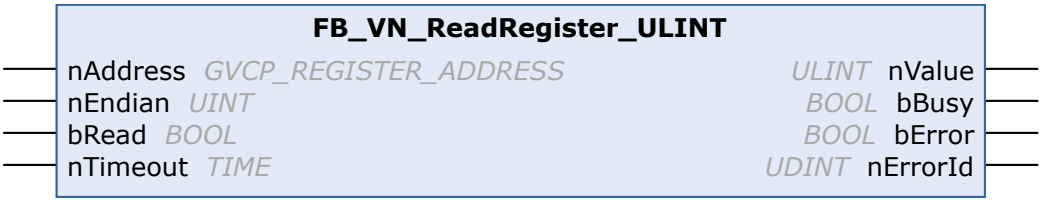
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.2.4 FB_VN_ReadRegister_ULINT



该 FB 从相机读取一个 ULINT 寄存器。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()

语法

定义：

```
FUNCTION_BLOCK FB_VN_ReadRegister_ULINT
VAR_INPUT
    nAddress                : GVCP_REGISTER_ADDRESS;
    nEndian                 : UINT;
    bRead                   : BOOL;
    nTimeout                : TIME;
END_VAR
VAR_OUTPUT
    nValue                  : ULINT;
    bBusy                   : BOOL;
    bError                  : BOOL;
    nErrorId                : UDINT;
END_VAR
VAR
    oidITcVnGevImageProvider : OTCID;
END_VAR
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 152]		应读取的寄存器的地址
nEndian	UINT		可选：寄存器的字节序。0 = 大，1 = 小。默认值：0
bRead	BOOL		读取寄存器由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
nValue	ULINT	读取的值
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

更多信息

需要开放相机控制通道！

GigE Vision 相机的控制通道必须打开，才能使用此功能块。此外，有些相机参数只有在停止图像采集的情况下才能更改。因此，建议首先将相机置于 TCVN_CS_OPENED 状态。

参数

nAddress

寄存器地址标识要寻址的相机寄存器。它属于 GVCP_REGISTER_ADDRESS 类型，是 UDINT 类型的别名。因此，所有寄存器地址的大小都是 4 字节。相机参数的寄存器地址可在 GigE Vision 相机实例配置树中 [▶ 81]的相应点查看：

Property	Value
Name	PixelFormat
Value	BayerRG8
Address	0x00012120
Length	4 Bytes

nEndian

字节顺序描述相机寄存器字节的排列顺序。这在应用中无需进一步考虑，可作为 GigE Vision 相机实例配置树中相应参数的属性：

Property	Value
Length	4 Bytes
Endianness	Big
Sign	Unsigned

大端序用 `nEndian := 0` 表示，小端序用 `nEndian := 1` 表示。默认使用大端序。

示例

读/写寄存器 [▶ 2816]

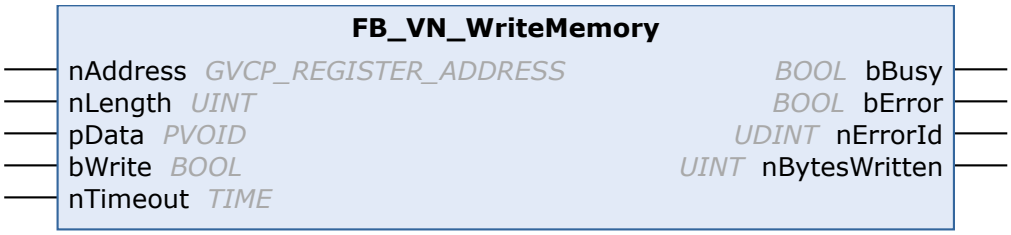
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.2.5 FB_VN_WriteMemory



该 FB 将数据写入相机的连续内存位置。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()

语法

定义：

```
FUNCTION_BLOCK FB_VN_WriteMemory
VAR_INPUT
    nAddress                : GVCP_REGISTER_ADDRESS;
    nLength                 : UINT;
    pData                   : PVOID;
    bWrite                  : BOOL;
    nTimeout                : TIME;
END_VAR
VAR_OUTPUT
    bBusy                   : BOOL;
    bError                  : BOOL;
    nErrorId                : UDINT;
    nBytesWritten           : UINT;
END_VAR
VAR
    oidITcVnGevImageProvider : OTCID;
END_VAR
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 152]		要写入相机的第一个字节的地址（必须 32 位对齐）
nLength	UINT		从 nAddress 开始写入的字节数（必须是 4 的倍数，不超过 536）
pData	PVOID		该指针指向应写入相机的数据（大小必须至少为 nLength 字节！）
bWrite	BOOL		写入内存由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。
nBytesWritten	UINT	已成功写入相机的字节数。

更多信息

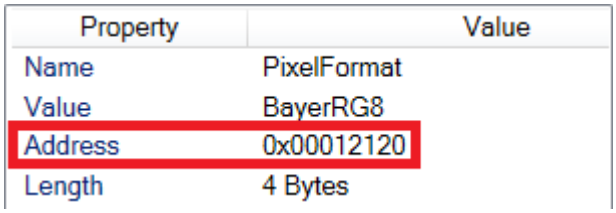
需要开放相机控制通道！

GigE Vision 相机的控制通道必须打开，才能使用此功能块。此外，有些相机参数只有在停止图像采集的情况下才能更改。因此，建议首先将相机置于 TCVN_CS_OPENED 状态。

参数

nAddress

寄存器地址标识要寻址的相机寄存器。它属于 `GVCP_REGISTER_ADDRESS` 类型，是 `UDINT` 类型的别名。因此，所有寄存器地址的大小都是 4 字节。相机参数的寄存器地址可在 GigE Vision 相机实例配置树中 [▶ 81]的相应点查看：



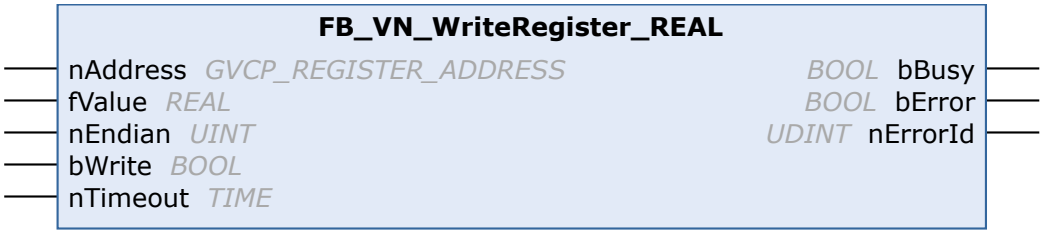
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.2.6 FB_VN_WriteRegister_REAL



该 FB 将一个 REAL 值写入相机的寄存器。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()）

语法

定义：

```
FUNCTION_BLOCK FB_VN_WriteRegister_REAL
VAR_INPUT
    nAddress                : GVCP_REGISTER_ADDRESS;
    fValue                  : REAL;
    nEndian                 : UINT;
    bWrite                  : BOOL;
    nTimeout                : TIME;
END_VAR
VAR_OUTPUT
    bBusy                   : BOOL;
    bError                  : BOOL;
    nErrorId                : UDINT;
END_VAR
VAR
    oidITcVnGevImageProvider : OTCID;
END_VAR
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 152]		应写入的寄存器的地址
fValue	REAL		要写入的值
nEndian	UINT		可选：寄存器的字节序。0 = 大，1 = 小。默认值：0
bWrite	BOOL		写入寄存器由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。

名称	类型	描述
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

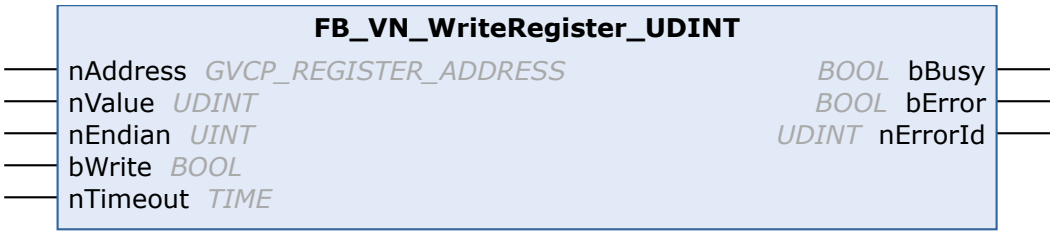
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.2.7 FB_VN_WriteRegister_UDINT



该 FB 将一个 DINT 值写入相机的寄存器。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()

语法

定义：

```
FUNCTION_BLOCK FB_VN_WriteRegister_UDINT
VAR_INPUT
    nAddress                : GVCP_REGISTER_ADDRESS;
    nValue                  : UDINT;
    nEndian                 : UINT;
    bWrite                  : BOOL;
    nTimeout                : TIME;
END_VAR
VAR_OUTPUT
    bBusy                   : BOOL;
    bError                  : BOOL;
    nErrorId                : UDINT;
END_VAR
VAR
    oidITcVnGevImageProvider : OTCID;
END_VAR
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 152]		应写入的寄存器的地址
nValue	UDINT		要写入的值
nEndian	UINT		可选：寄存器的字节序。0 = 大，1 = 小。默认值：0
bWrite	BOOL		写入寄存器由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

更多信息

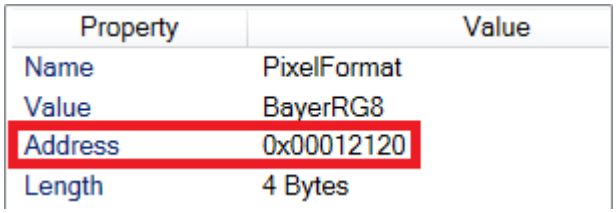
需要开放相机控制通道！

GigE Vision 相机的控制通道必须打开，才能使用此功能块。此外，有些相机参数只有在停止图像采集的情况下才能更改。因此，建议首先将相机置于 TCVN_CS_OPENED 状态。

参数

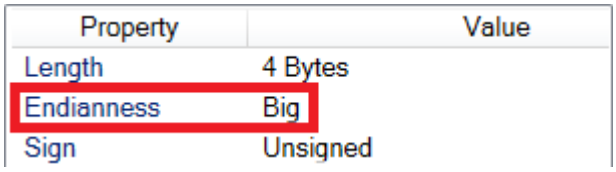
nAddress

寄存器地址标识要寻址的相机寄存器。它属于 `GVCP_REGISTER_ADDRESS` 类型，是 `UDINT` 类型的别名。因此，所有寄存器地址的大小都是 4 字节。相机参数的寄存器地址可在 GigE Vision 相机实例配置树中 [▶ 81]的相应点查看：



nEndian

字节顺序描述相机寄存器字节的排列顺序。这在应用中无需进一步考虑，可作为 GigE Vision 相机实例配置树中相应参数的属性：



大端序用 `nEndian := 0` 表示，小端序用 `nEndian := 1` 表示。默认使用大端序。

样本

- 读/写寄存器 [▶ 2816]

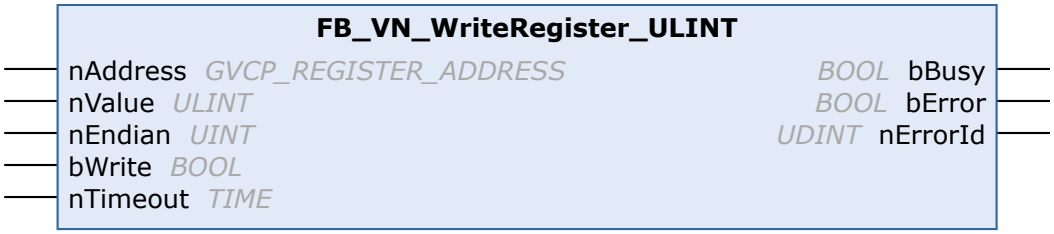
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.2.8 FB_VN_WriteRegister_ULINT



该 FB 将一个 ULINT 值写入相机的寄存器。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()

语法

定义：

```
FUNCTION_BLOCK FB_VN_WriteRegister_ULINT
VAR_INPUT
    nAddress                : GVCP_REGISTER_ADDRESS;
    nValue                  : ULINT;
    nEndian                 : UINT;
    bWrite                  : BOOL;
    nTimeout                : TIME;
END_VAR
VAR_OUTPUT
    bBusy                   : BOOL;
    bError                  : BOOL;
    nErrorId                : UDINT;
END_VAR
VAR
    oidITcVnGevImageProvider : OTCID;
END_VAR
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 152]		应写入的寄存器的地址
nValue	ULINT		要写入的值
nEndian	UINT		可选：寄存器的字节序。0 = 大，1 = 小。默认值：0
bWrite	BOOL		写入寄存器由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

更多信息

需要开放相机控制通道!

GigE Vision 相机的控制通道必须打开，才能使用此功能块。此外，有些相机参数只有在停止图像采集的情况下才能更改。因此，建议首先将相机置于 TCVN_CS_OPENED 状态。

参数

nAddress

寄存器地址标识要寻址的相机寄存器。它属于 `GVCP_REGISTER_ADDRESS` 类型，是 `UDINT` 类型的别名。因此，所有寄存器地址的大小都是 4 字节。相机参数的寄存器地址可在 GigE Vision 相机实例配置树中 [▶ 81]的相应点查看：

Property	Value
Name	PixelFormat
Value	BayerRG8
Address	0x00012120
Length	4 Bytes

nEndian

字节顺序描述相机寄存器字节的排列顺序。这在应用中无需进一步考虑，可作为 GigE Vision 相机实例配置树中相应参数的属性：

Property	Value
Length	4 Bytes
Endianness	Big
Sign	Unsigned

大端序用 `nEndian := 0` 表示，小端序用 `nEndian := 1` 表示。默认使用大端序。

示例

读/写寄存器 [▶ 2816]

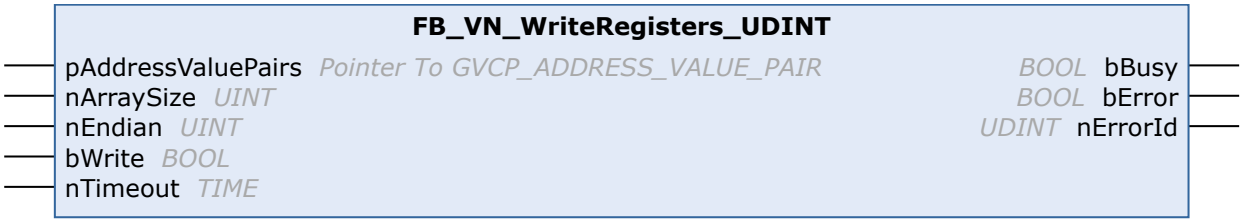
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.2.9 FB_VN_WriteRegisters_UDINT



该 FB 在相机上写入多个 UDINT 寄存器。需要指定的寄存器具有相同的字节序，以及开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()）。

语法

定义：

```
FUNCTION_BLOCK FB_VN_WriteRegisters_UDINT
VAR_INPUT
    pAddressValuePairs       : Pointer To GVCP_ADDRESS_VALUE_PAIR;
    nArraySize               : UINT;
    nEndian                  : UINT;
    bWrite                   : BOOL;
    nTimeout                 : TIME;
END_VAR
VAR_OUTPUT
    bBusy                    : BOOL;
    bError                   : BOOL;
    nErrorId                 : UDINT;
END_VAR
VAR
    oidITcVnGevImageProvider : OTCID;
END_VAR
```

输入

名称	类型	默认值	描述
pAddressValuePairs	指针指向 GVCP_ADDRESS_VALUE_PAIR [▶ 220]		一个数组，包含应写入的寄存器的地址/值对
nArraySize	UINT		pAddressValuePairs 中地址/值对的数量
nEndian	UINT		可选：寄存器的字节序。0 = 大，1 = 小。默认：0
bWrite	BOOL		写入寄存器由该输入端的上升沿触发。
nTimeout	TIME	VISION_ADS_TIMEOUT	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

更多信息

FB_VN_WriteRegisters_UDINT 在一次调用中写入相机的多个 UDINT 寄存器。

需要开放相机控制通道！

GigE Vision 相机的控制通道必须打开，才能使用此功能块。此外，有些相机参数只有在停止图像采集的情况下才能更改。因此，建议首先将相机置于 TCVN_CS_OPENED 状态。

参数

pAddressValuePairs

要写入的寄存器以类型为 GVCP_ADDRESS_VALUE_PAIR [▶ 220] 的数组形式传输。数组中的每个元素均包含寄存器地址（GVCP_REGISTER_ADDRESS）和相应的写入值（GVCP_REGISTER_VALUE）。指向该数组的指针通过输入 pAddressValuePairs 传输。

寄存器地址标识要寻址的相机寄存器。它属于 GVCP_REGISTER_ADDRESS 类型，是 UDINT 类型的别名。因此，所有寄存器地址的大小都是 4 字节。相机参数的寄存器地址可在 GigE Vision 相机实例配置树中 [▶ 81]的相应点查看：

Property	Value
Name	PixelFormat
Value	BayerRG8
Address	0x00012120
Length	4 Bytes

nArraySize

指定 pAddressValuePairs 数组中地址-值对的数量。

nEndian

字节顺序描述相机寄存器字节的排列顺序。该值可以作为相应参数的属性，从 GigE Vision 相机实例的配置树 [▶ 81]中获取：

Property	Value
Length	4 Bytes
Endianness	Big
Sign	Unsigned

大端序用 nEndian := 0 表示，小端序用 nEndian := 1 表示。默认使用大端序。

要求所有寄存器具有相同的字节序

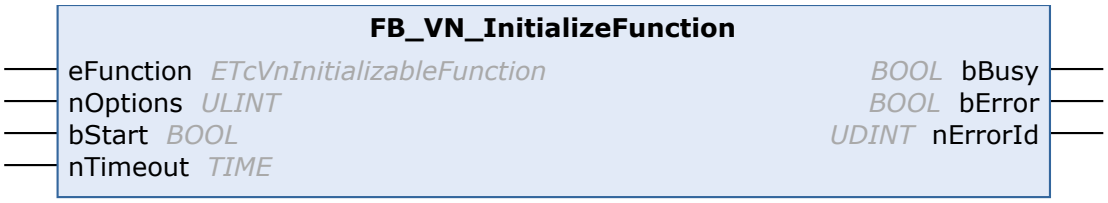
传输的数组中的所有地址-值对必须使用相同的字节序。如果要写入具有不同字节序的寄存器，则需要分别进行调用，并使用相应的数组。

所需的许可证

TC3 Vision Base

6.1.5.3 文件访问

6.1.5.3.1 FB_VN_InitializeFunction



该 FB 将初始化选定的函数。

语法

定义：

```
FUNCTION_BLOCK FB_VN_InitializeFunction
VAR_INPUT
    eFunction : ETcVnInitializableFunction;
    nOptions  : ULINT;
    bStart    : BOOL;
    nTimeout  : TIME;
END_VAR
VAR_OUTPUT
    bBusy     : BOOL;
    bError    : BOOL;
    nErrorId  : UDINT;
END_VAR
```

输入

名称	类型	默认值	描述
eFunction	ETcVnInitializableFunction [▶ 198]		选择要初始化的函数
nOptions	ULINT	0	为函数选择初始化选项
bStart	BOOL		函数初始化由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

更多信息

功能块 `FB_VN_InitializeFunction` 初始化在 `eFunction` 指定的函数。所需数据在后台异步加载到 PLC。

参数

文件访问参数部分对某些参数进行集中描述。与描述不同的是，必须为该功能块指定待初始化函数，而不是文件路径。

函数

待初始化的 `eFunction`。

选项

为指定函数选择初始化选项 `nOptions`。可能的枚举选项可在待初始化函数的描述中找到。这些选项可以与 `OR` 相链接，从而通过调用功能块加载。

应用

例如，带有大写字母分类选项的 OCR 函数的初始化过程如下所示：

```
fbInitOCR(
    eFunction   := TCVN_IF_OCR,
    nOptions    := ETcVnOcrModelType.TCVN_OMT_UCLETTERS,
    bStart      := TRUE,
    nTimeout    := T#500MS);

IF NOT fbInitOCR.bBusy AND NOT fbInitOCR.bError THEN
    // The function is initialized and can be used with the specified options
END_IF
```

相关函数

- F_VN_CheckFunctionInitialization [▶ 1514] 可用来检查函数是否已通过相应选项初始化。
- F_VN_DeinitializeFunction [▶ 1522] 提供取消初始化函数选项从而释放内存的可能性。

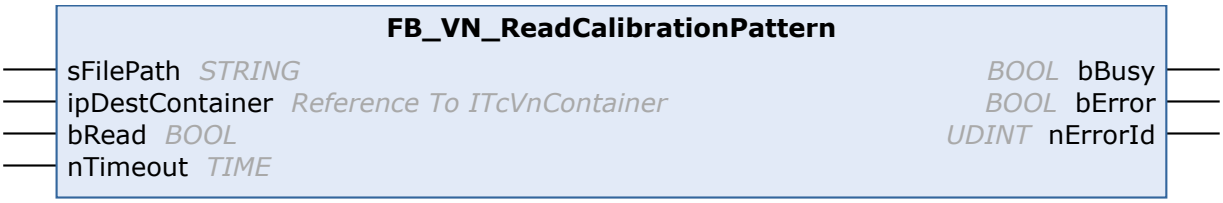
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.3.2 FB_VN_ReadCalibrationPattern



该 FB 可从 targte PC 上的 xml 文件中读取标定板参考点。

语法

定义：

```
FUNCTION_BLOCK FB_VN_ReadCalibrationPattern
VAR_INPUT
    sFilePath       : STRING;
    ipDestContainer : Reference To ITcVnContainer;
    bRead           : BOOL;
    nTimeout        : TIME;
END_VAR
VAR_OUTPUT
    bBusy           : BOOL;
    bError          : BOOL;
    nErrorId        : UDINT;
END_VAR
```

输入

名称	类型	默认值	描述
sFilePath	STRING		文件的完整路径或目标计算机上默认目录的相对路径
ipDestContainer	引用 ITcVnContainer [▶ 365]		返回包含标定板点的容器（ContainerType_Vector_TcVnPoint3_REAL）
bRead	BOOL		读取文件由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

更多信息

功能块FB_VN_ReadCalibrationPattern从文件中异步加载单个校准模式 [▶ 118]的描述。

加载修改的容器

在加载容器时，视觉服务在内部缓存数据。因此，无法在标准系统中加载更改的容器。为此，请调整缓存设置 [▶ 68]或者参见串行加载图像到 PLC 的样本 [▶ 2819]。

参数

在文件访问参数一节中集中描述了一些参数。

加载的容器 (返回值)

在成功的读取操作之后，可以通过接口指针ipPatternReferencePoints访问加载了校准模式参考点的容器。

应用

例如，从路径C:\TcVision\CalibrationPatternReferencePoints.xml加载校准模式参考点如下所示：

```
fbReadCalibrationPattern(
    sFilePath       := 'C:\TcVision\CalibrationPatternReferencePoints.xml',
    ipDestContainer := ipPatternReferencePoints,
    bRead           := TRUE,
    nTimeout        := T#500MS);

IF NOT fbReadCalibrationPattern.bBusy AND NOT fbReadCalibrationPattern.bError THEN
    // reference points are accessible in ipPatternReferencePoints
END_IF
```

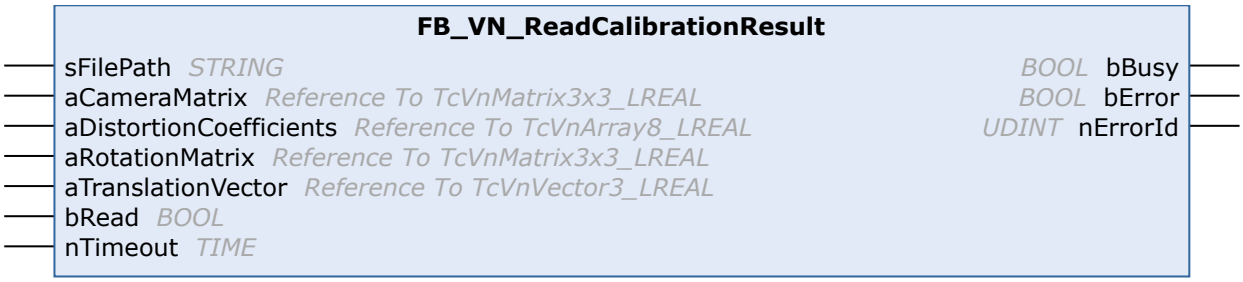
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.3.3 FB_VN_ReadCalibrationResult



该 FB 可从目标计算机上的 json 文件读取标定结果。

语法

定义：

```
FUNCTION_BLOCK FB_VN_ReadCalibrationResult
VAR_INPUT
    sFilePath               : STRING;
    aCameraMatrix           : Reference To TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : Reference To TcVnArray8_LREAL;
    aRotationMatrix         : Reference To TcVnMatrix3x3_LREAL;
    aTranslationVector      : Reference To TcVnVector3_LREAL;
    bRead                   : BOOL;
```

```
    nTimeout                : TIME;
END_VAR
VAR_OUTPUT
    bBusy                   : BOOL;
    bError                  : BOOL;
    nErrorId                : UDINT;
END_VAR
```

输入

名称	类型	默认值	描述
sFilePath	STRING		文件的完整路径或目标计算机上默认目录的相对路径
aCameraMatrix	引用 TcVnMatrix3x3_LREAL [▶ 153]		返回相机矩阵
aDistortionCoefficients	引用 TcVnArray8_LREAL [▶ 153]		返回畸变系数
aRotationMatrix	引用 TcVnMatrix3x3_LREAL [▶ 153]		返回旋转矩阵
aTranslationVector	引用 TcVnVector3_LREAL [▶ 153]		返回平移向量
bRead	BOOL		读取文件由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

更多信息

功能块FB_VN_ReadCalibrationResult将校准结果从文件中异步加载到 PLC。

加载更改后的校准结果。

在加载校准结果时，视觉服务在内部缓存了数据。因此，无法在标准系统中加载修改的结果。为此，请调整缓存设置 [▶ 68]或者参见串行加载图像到 PLC 的样本 [▶ 2819]。

参数

在文件访问参数一节中集中描述了一些参数。

校准结果

读取操作成功后，可以通过以下变量访问加载的校准结果：

```
// Calibration results
aCameraMatrix            : TcVnMatrix3x3_LREAL;
aDistortionCoefficients  : TcVnArray8_LREAL;
aRotationMatrix          : TcVnMatrix3x3_LREAL;
aTranslationVector       : TcVnVector3_LREAL;
```

应用

例如，从路径C:\TcVision\CalibrationResult.json加载校准结果如下所示：

```
fbReadCalibrationResult(
        sFilePath               := 'C:\TcVision\ CalibrationResult.json',
        aCameraMatrix           := aCameraMatrix,
        aDistortionCoefficients := aDistortionCoefficients,
        aRotationMatrix         := aRotationMatrix,
        aTranslationVector      := aTranslationVector,
        bRead                   := TRUE,
        nTimeout                := T#500MS);

IF NOT fbReadCalibrationResult.bBusy AND NOT fbReadCalibrationResult.bError THEN
    // calibration results are accessible
END_IF
```

相关功能块：FB_VN_WriteCalibrationResult [▶ 1593]。

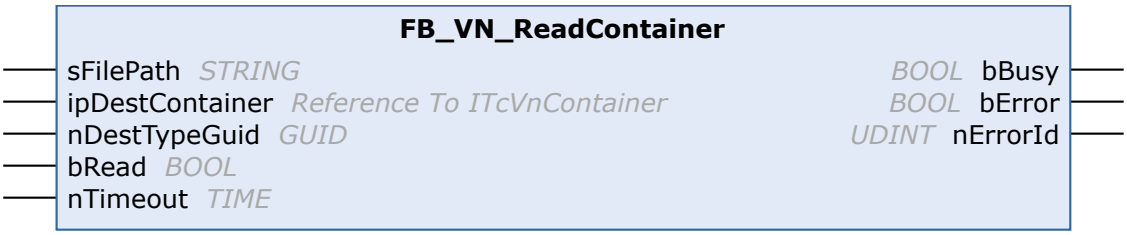
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.3.4 FB_VN_ReadContainer



该 FB 从目标计算机上的 xml 文件读取容器。

语法

定义：

```
FUNCTION_BLOCK FB_VN_ReadContainer
VAR_INPUT
    sFilePath       : STRING;
    ipDestContainer : Reference To ITcVnContainer;
    nDestTypeGuid   : GUID;
    bRead           : BOOL;
    nTimeout        : TIME;
END_VAR
VAR_OUTPUT
    bBusy           : BOOL;
    bError          : BOOL;
    nErrorId        : UDINT;
END_VAR
```

输入

名称	类型	默认值	描述
sFilePath	STRING		文件的完整路径或目标计算机上默认目录的相对路径

名称	类型	默认值	描述
ipDestContainer	引用 ITcVnContainer [▶ 365]		返回包含已加载内容的容器
nDestTypeGuid	GUID		要返回的容器的类型 ID。如果未设置该参数，将自动确定合适的容器类型（csv 文件除外）。
bRead	BOOL		读取文件由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.3.5 FB_VN_ReadImage



该 FB 从目标计算机上的文件读取图像。

语法

定义：

```
FUNCTION_BLOCK FB_VN_ReadImage
VAR_INPUT
    sFilePath   : STRING;
    ipDestImage : Reference To ITcVnImage;
    bRead       : BOOL;
    nTimeout    : TIME;
END_VAR
VAR_OUTPUT
    bBusy       : BOOL;
    bError      : BOOL;
    nErrorId    : UDINT;
END_VAR
```

输入

名称	类型	默认值	描述
sFilePath	STRING		文件的完整路径或目标计算机上默认图像目录的相对路径
ipDestImage	引用 ITcVnImage [▶ 407]		返回包含已加载内容的图像
bRead	BOOL		读取文件由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.3.6 FB_VN_ReadMlModel



该 FB 可从目标计算机上的文件读取机器学习模型。

语法

定义：

```
FUNCTION_BLOCK FB_VN_ReadMlModel
VAR_INPUT
    sFilePath      : STRING;
    ipDestMlModel : Reference To ITcVnMlModel;
    bRead          : BOOL;
    nTimeout       : TIME;
END_VAR
VAR_OUTPUT
    bBusy          : BOOL;
    bError         : BOOL;
    nErrorId       : UDINT;
END_VAR
```

输入

名称	类型	默认值	描述
sFilePath	STRING		文件的完整路径或目标计算机上默认目录的相对路径
ipDestMlModel	引用 ITcVnMlModel [▶ 417]		返回已加载的机器学习模型
bRead	BOOL		读取文件由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

更多信息

功能块FB_VN_ReadMlModel将一个ML 模型 [▶ 417]从一个文件异步加载到 PLC 中。

加载修改的 ML 模型

在加载 ML 模型时，视觉服务在内部缓存数据。因此，无法在标准系统中加载修改过的 ML 模型。为此，请调整缓存设置 [▶ 68]或参见示例连续加载图像到 PLC [▶ 2819]代替。

参数

在文件访问参数一节中集中描述了一些参数。

加载的 ML 模型（返回值）

读取成功后，加载的ITcVnMlModel可通过接口指针ipDestMlModel访问。

一个特殊的情况是作为ITcVnColorModel，因为在加载模型之后，仍然需要与下面的行进行转换。另外，ITcVnMlModel也可以通过相应的函数直接使用。

```
hr := ipReadMlModel.TcQueryInterface(IID_ITcVnColorModel, ADR(ipColorModel));
```

应用

例如，从路径C:\TcVision\MlModel.vmlm加载 ML 模型如下所示：

```
fbReadMlModel(
    sFilePath       := 'C:\TcVision\MlModel.vmlm',
    ipDestMlModel   := ipReadMlModel,
    bRead           := TRUE,
    nTimeout        := T#500MS);

IF NOT fbReadMlModel.bBusy AND NOT fbReadMlModel.bError THEN
    // Ml-Model is accessible in ipReadMlModel
END_IF
```

相关的功能块：FB_VN_WriteMlModel [▶ 1597]。

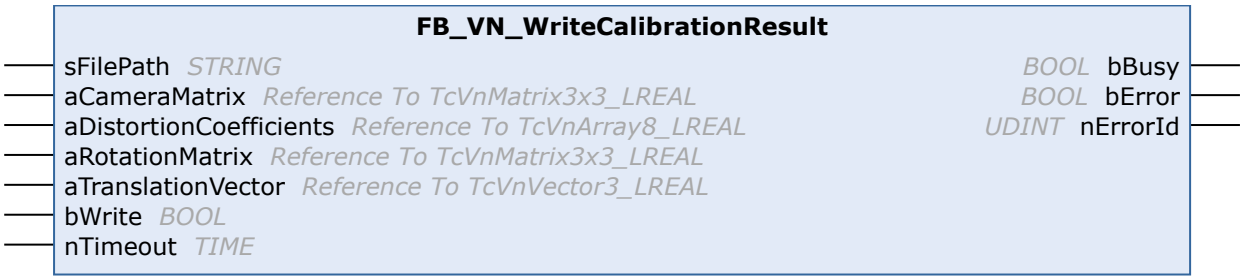
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.3.7 FB_VN_WriteCalibrationResult



该 FB 将标定结果以 json 文件的形式写入硬盘。

语法

定义：

```
FUNCTION_BLOCK FB_VN_WriteCalibrationResult
VAR_INPUT
    sFilePath               : STRING;
    aCameraMatrix           : Reference To TcVnMatrix3x3_LREAL;
    aDistortionCoefficients : Reference To TcVnArray8_LREAL;
    aRotationMatrix         : Reference To TcVnMatrix3x3_LREAL;
    aTranslationVector      : Reference To TcVnVector3_LREAL;
    bWrite                  : BOOL;
    nTimeout                : TIME;
END_VAR
VAR_OUTPUT
    bBusy                   : BOOL;
    bError                  : BOOL;
    nErrorId                : UDINT;
END_VAR
```

输入

名称	类型	默认值	描述
sFilePath	STRING		文件的完整路径或目标计算机上默认目录的相对路径（例如，只有文件名）。如果字符串为空，将生成一个文件名（包含当前时间）。
aCameraMatrix	引用 TcVnMatrix3x3_LREAL [▶ 153]		返回相机矩阵
aDistortionCoefficients	引用 TcVnArray8_LREAL [▶ 153]		返回畸变系数
aRotationMatrix	引用 TcVnMatrix3x3_LREAL [▶ 153]		返回旋转矩阵
aTranslationVector	引用 TcVnVector3_LREAL [▶ 153]		返回平移向量
bWrite	BOOL		写入容器由该输入端的上升沿触发。

名称	类型	默认值	描述
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

更多信息

功能块FB_VN_WriteCalibrationResult将 PLC 的校准结果异步保存在一个文件中。

参数

在文件访问参数一节中集中描述了一些参数。

校准结果

以下校准功能的结果以JSON格式保存在指定的文件路径下。

```
// Calibration results
aCameraMatrix             : TcVnMatrix3x3_LREAL;
aDistortionCoefficients   : TcVnArray8_LREAL;
aRotationMatrix           : TcVnMatrix3x3_LREAL;
aTranslationVector        : TcVnVector3_LREAL;
```

文件路径和格式

如果sFilePath为空，将生成带有当前时间的文件名，并使用扩展名json。

应用

例如，将校准结果保存到C:\TcVision\CalibrationResult.json文件如下所示：

```
fbWriteCalibrationResult(
    sFilePath               := 'C:\TcVision\CalibrationResult.json',
    aCameraMatrix           := aCameraMatrix,
    aDistortionCoefficients := aDistortionCoefficients,
    aRotationMatrix         := aRotationMatrix,
    aTranslationVector      := aTranslationVector,
    bWrite                  := TRUE,
    nTimeout                := T#500MS);

IF NOT fbWriteCalibrationResult.bBusy AND NOT fbWriteCalibrationResult.bError THEN
    // Calibration result was written successfully to file
END_IF
```

相关功能块：FB_VN_ReadCalibrationResult [▶ 1587]。

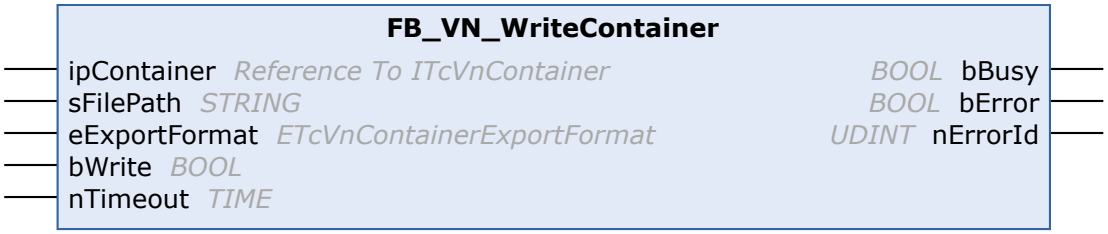
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.3.8 FB_VN_WriteContainer



该 FB 将容器作为 xml 文件写入硬盘。

语法

定义：

```
FUNCTION_BLOCK FB_VN_WriteContainer
VAR_INPUT
    ipContainer    : Reference To ITcVnContainer;
    sFilePath      : STRING;
    eExportFormat  : ETcVnContainerExportFormat;
    bWrite         : BOOL;
    nTimeout       : TIME;
END_VAR
VAR_OUTPUT
    bBusy          : BOOL;
    bError         : BOOL;
    nErrorId       : UDINT;
END_VAR
```

输入

名称	类型	默认值	描述
ipContainer	引用 ITcVnContainer [▶ 365]		要写入的容器
sFilePath	STRING		文件的完整路径或目标计算机上默认目录的相对路径（例如，只有文件名）。如果字符串为空，将生成一个文件名（包含当前时间）。
eExportFormat	ETcVnContainerExportFormat [▶ 178]	TCVN_CEF_XML	容器导出格式（默认为可人工读取的 xml 格式）
bWrite	BOOL		写入容器由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.3.9 FB_VN_WriteImage



该 FB 将图像写入硬盘。

语法

定义：

```
FUNCTION_BLOCK FB_VN_WriteImage
VAR_INPUT
    ipImage   : Reference To ITcVnImage;
    sFilePath : STRING;
    bWrite    : BOOL;
    nTimeout  : TIME;
END_VAR
VAR_OUTPUT
    bBusy     : BOOL;
    bError    : BOOL;
    nErrorId  : UDINT;
END_VAR
```

输入

名称	类型	默认值	描述
ipImage	引用 ITcVnImage [▶ 407]		要写入的图像
sFilePath	STRING		文件的完整路径或目标计算机上默认目录的相对路径（例如，只有文件名）。如果字符串为空，将生成一个文件名（包含当前时间）。
bWrite	BOOL		写入图像由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.3.10 FB_VN_WriteMlModel



该 FB 将机器学习模型写入硬盘。

语法

定义：

```
FUNCTION_BLOCK FB_VN_WriteMlModel
VAR_INPUT
    ipMlModel : Reference To ITcVnMlModel;
    sFilePath : STRING;
    bWrite    : BOOL;
    nTimeout  : TIME;
END_VAR
VAR_OUTPUT
    bBusy     : BOOL;
    bError    : BOOL;
    nErrorId  : UDINT;
END_VAR
```

输入

名称	类型	默认值	描述
ipMlModel	引用 ITcVnMlModel [▶ 417]		要写入的机器学习模型
sFilePath	STRING		文件的完整路径或目标计算机上默认目录的相对路径（例如，只有文件名）。如果字符串为空，将生成一个文件名（包含当前时间）。
bWrite	BOOL		写入机器学习模型由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

更多信息

功能块 FB_VN_WriteMlModel 以异步方式将 ML 模型 [▶ 417]从文件保存到 PLC 中。

参数

文件访问参数部分对某些参数进行集中描述。

ML 模型

ML 模型 ipMlModel 以二进制格式保存在指定文件路径下。

ITcVnColorModel 的保存代表一个特例，因为这仍然必须首先将下面一行转换为 ML 模型。另外，也可以用带有相应函数的 ITcVnMlModel 替代颜色模型。

```
hr := ipColorModel.TcQueryInterface(IID_ITcVnMlModel, ADR(ipWriteMlModel));
```

文件路径和格式

如果 sFilePath 为空，则会生成包含当前时间的文件名，并使用扩展名 vmlm。

应用

例如，将一个 ML 模型保存到文件 C:\TcVision\MlModel.vmlm 的过程如下所示：

```
fbWriteMlModel(
    ipMlModel    := ipWriteMlModel,
    sFilePath    := 'C:\TcVision\MlModel.vmlm',
    bWrite       := TRUE,
    nTimeout     := T#500MS);

IF NOT fbWriteMlModel.bBusy AND NOT fbWriteMlModel.bError THEN
    // Ml-Model was written successfully to file
END_IF
```

相关功能块：FB_VN_ReadMlModel [▶ 1591]。

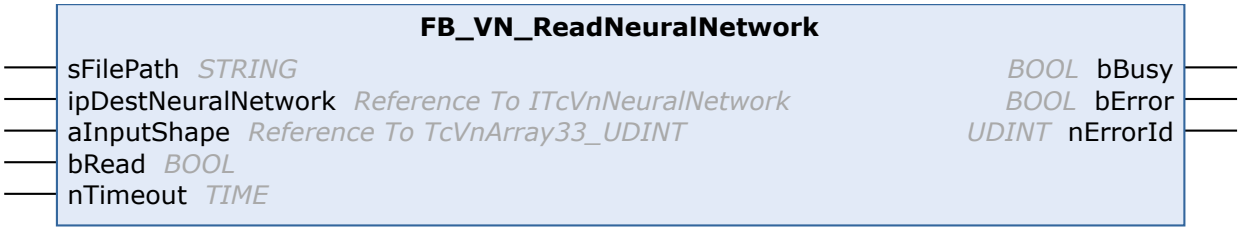
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.3.11 FB_VN_ReadNeuralNetwork



该 FB 从目标计算上的文件读取神经网络。

语法

定义：

```
FUNCTION_BLOCK FB_VN_ReadNeuralNetwork
VAR_INPUT
    sFilePath           : STRING;
    ipDestNeuralNetwork : Reference To ITcVnNeuralNetwork;
    aInputShape         : Reference To TcVnArray33_UDINT;
    bRead               : BOOL;
```

```
    nTimeout            : TIME;
END_VAR
VAR_OUTPUT
    bBusy               : BOOL;
    bError              : BOOL;
    nErrorId            : UDINT;
END_VAR
```

输入

名称	类型	默认值	描述
sFilePath	STRING		文件的完整路径或目标计算机上默认目录的相对路径
ipDestNeuralNetwork	引用 ITcVnNeuralNetwork [▶ 426]		返回神经网络
aInputShape	引用 TcVnArray33_UDINT [▶ 153]		用于测试神经网络模型的执行情况的可选输入形状。如果执行失败，将返回错误代码 INCOMPATIBLE。设置 aInputShape = 0 或 aInputShape[0] = 0 将禁用执行测试。
bRead	BOOL		读取文件由该输入端的上升沿触发。
nTimeout	TIME	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	UDINT	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.4 图像采集

该组包含用于通过 Vision 设备采集图像的功能块（GigE Vision 相机 [▶ 71]和文件源 [▶ 123]对象）。

功能块

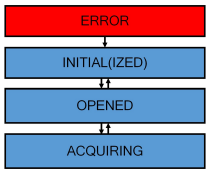
- FB_VN_SimpleCameraControl [▶ 1630]用于相机和文件源
- FB_VN_GevCameraControl [▶ 1609]用于 GigE Vision 相机
- FB_VN_FileSourceControl [▶ 1601]用于文件源

综述

该组中的功能块代表图像提供者 TcCOM 对象的代理。因此，它们不需要周期性地调用。相反，TcCOM 对象通过功能块的方法进行控制。

状态机

所有图像采集功能块均包括一个状态机，用于管理所代表设备的状态。基本上，功能块必须置于 ACQUIRING（采集中）状态，才能连续接收图像。INITIAL（初始）状态代表重启或复位后的起始点，可触发 INITIALIZING（初始化）。如为 OPENED（已打开）状态，与设备的通信通道会被打开，以读取或写入寄存器值。



所有状态都由枚举 ETcVnCameraState [▶ 162] 表示，并在下文中加以说明：

主要状态

主要的状态是静止状态。它们基本上表明是否有与 Vision 设备的连接（OPENED），该设备是否正在捕捉图像（ACQUIRING）以及与该设备的连接是否有错误（ERROR）。你可以通过 FB 的方法在这些状态之间切换。一般来说，两个主要状态之间的转换不会立即发生，而是至少经过一个周期。在这个中间时间，视觉设备处于中间状态 [▶ 1600]。

状态	描述
ERROR	错误状态，在这个状态下，每个设备只能通过调用方法Reset重新激活。
INITIAL	每个设备的初始状态。
INITIALIZED	与 INITIAL 类似，手动初始化相机（例如使用 Force-IP 和初始化命令）
OPENED	存在与相机的连接，可以触发单个图像，且可以读写相机寄存器。 **这个状态可以通过 StartAcquisition 方法跳过。**
ACQUIRING	设备处于采集状态，且正在发送图像，具体取决于设置。

中间状态

中间状态十分必要，因为大多数行动不能立即完成（如图像采集）。如果功能块处于这些中间状态之一，必须再次调用相应的方法，以完成向相应的主状态的过渡。

状态	描述
INITIALIZING	INITIAL > INITIALIZED
OPENING	INITIAL > OPENED 且 INITIALIZED > OPENED
STARTACQUISITION	OPENED > ACQUIRING.
STOPACQUISITION	ACQUIRING > OPENED
RESETTINGFEATURES	OPENED > OPENED
TRIGGERING	OPENED > OPENED 且 ACQUIRING > ACQUIRING
CLOSING	OPENED > INITIALIZED

必须在中间状态下调用方法！

在中间状态中也必须调用各自的状态转换方法。在相应的方法调用后使用GetState方法检测到静止状态时，状态转换才算完成，且不再需要调用该方法。触发一次方法，而随后的等待并不会导致所需的状态转换。

这样，PLC 中的状态机就可以用 CASE … OF 结构或 IF … ELSIF … 结构来实现。

样本

- 通过文件名触发图像 [▶ 2820]

6.1.5.4.1 FB_VN_FileSourceControl

该 FB 可访问文件源实例发送的图像。

请勿直接调用主 FB。只能使用可用的方法。

方法

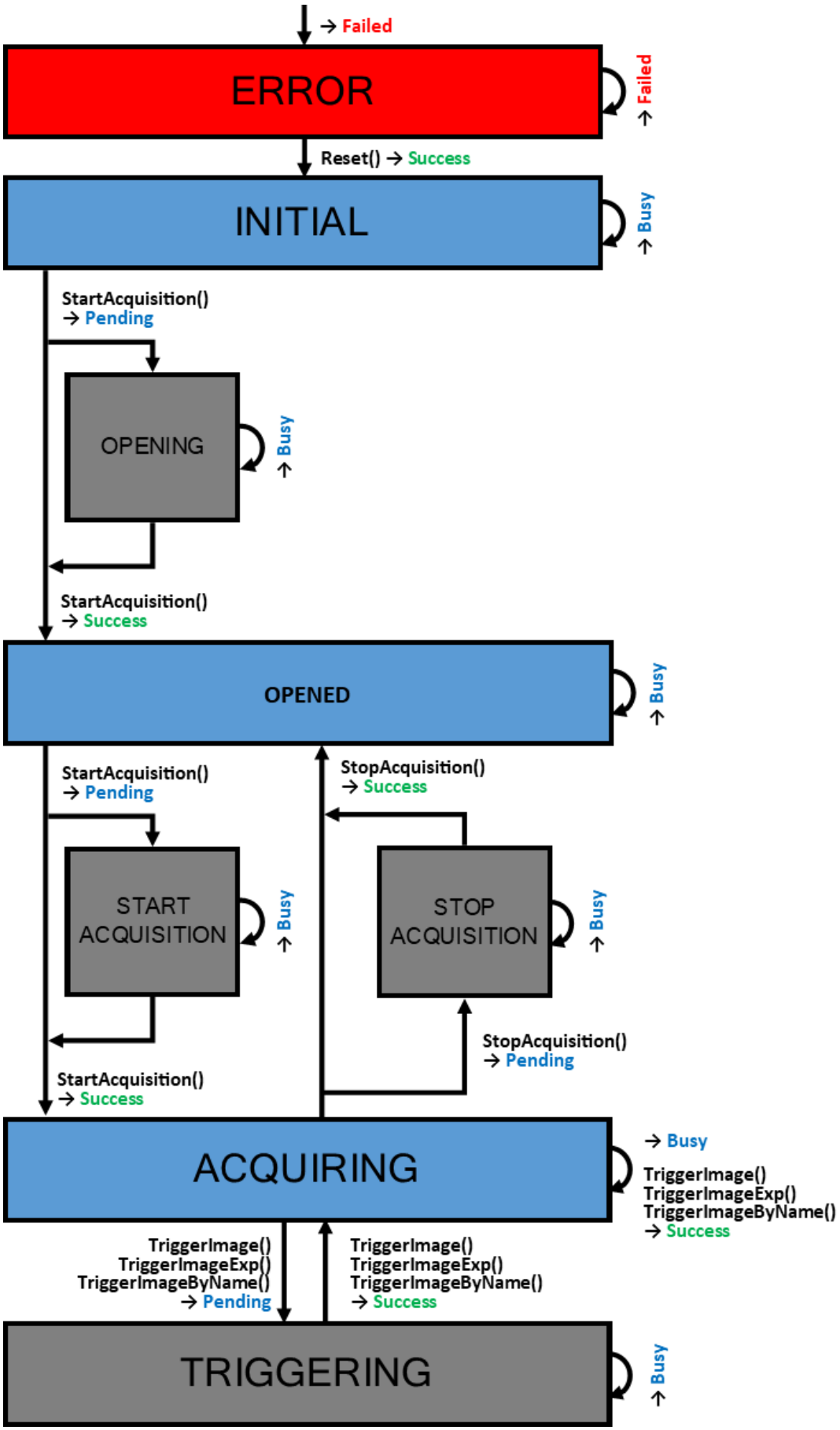
名称	描述
GetCurrentImage [▶ 1604]	获取当前可用的图像（如果有）。
GetCurrentImageAndFileName [▶ 1605]	获取当前可用的图像（如果有）和相应的文件名。
StartAcquisition [▶ 1605]	激活文件源。如果不在触发模式下，将开始发送图像。
StopAcquisition [▶ 1606]	停用文件源。将停止发送图像。
Reset [▶ 1606]	将控制器重置为初始状态（根据当前状态，可能需要多次调用，直至返回 S_OK）
GetState [▶ 1607]	获取内部相机控制状态机的当前状态。
TriggerImage [▶ 1608]	如果处于触发模式，该函数将触发列表中的下一个图像。
TriggerImageExp [▶ 1608]	如果处于触发模式，该函数将触发列表中的特定图像。
TriggerImageByName [▶ 1609]	如果处于触发模式，此函数会按名称触发列表中的特定图像。

更多信息

功能块`FB_VN_FileSourceControl`可以使用 PLC 中的全部函数控制文件源 [▶ 123]实例。另外，也可以通过功能块FB_VN_SimpleCameraControl [▶ 1630]实现有限的功能。

状态机

下图显示了状态机。主要状态以蓝色标记，过渡状态为灰色，而错误状态为红色。



以下方法可用于控制状态机：

- GetState [▶ 1607]

- StartAcquisition [▶ 1605]
- StopAcquisition [▶ 1606]
- TriggerImage [▶ 1608]
- TriggerImageByName [▶ 1609]
- Reset [▶ 1606]

如果在非预期状态下调用方法，将返回错误代码INVALIDSTATE（16#712）。

带有连续图像采集的应用

例如，连续检索收到的图像的状态机如下所示：

```
VAR
  fbFileSource  : FB_VN_FileSourceControl;
    eState          :   ETcVnCameraState;
    ipImageIn       :   ITcVnImage;
    ipImageInDisp   :   ITcVnDisplayableImage;
    hr              :   HRESULT;
    nNewImageCounter:   UINT;
END_VAR

eState := fbFileSource.GetState();

CASE eState OF

TCVN_CS_INITIAL, TCVN_CS_OPENING, TCVN_CS_OPENED, TCVN_CS_STARTACQUISITION:
  hr := fbFileSource.StartAcquisition();

TCVN_CS_ACQUIRING:
  hr := fbFileSource.GetCurrentImage(ipImageIn);

    // Check if new Image was received
    IF SUCCEEDED(hr) AND ipImageIn <> 0 THEN
        nNewImageCounter := nNewImageCounter + 1;

        // Place to call vision algorithms
        hr := F_VN_TransformIntoDisplayableImage(ipImageIn, ipImageInDisp, hr);
    END_IF

TCVN_CS_ERROR:
  hr := fbFileSource.Reset();

END_CASE
```

带有触发器的应用

例如，持续触发具有特殊文件名的图像的状态机如下所示：

```
VAR
  fbFileSource  : FB_VN_FileSourceControl;
    eState          :   ETcVnCameraState;
    ipImageIn       :   ITcVnImage;
    ipImageInDisp   :   ITcVnDisplayableImage;
    bTrigger        :   BOOL := TRUE;
    sFileName       :   STRING := 'FileName.bmp';
    hr              :   HRESULT;
    nNewImageCounter:   UINT;
END_VAR

eState := fbFileSource.GetState();

CASE eState OF

TCVN_CS_INITIAL, TCVN_CS_OPENING, TCVN_CS_OPENED, TCVN_CS_STARTACQUISITION:
  hr := fbFileSource.StartAcquisition();

TCVN_CS_TRIGGERING:
  hr := fbFileSource.TriggerImage();

TCVN_CS_ACQUIRING:

    IF bTrigger THEN
        hr := fbFileSource.TriggerImageByName(sFileName);
        IF SUCCEEDED(hr) THEN
            bTrigger := FALSE;
        END_IF
    ELSE
   hr := fbFileSource.GetCurrentImage(ipImageIn);

        // Check if new Image was received
        IF SUCCEEDED(hr) AND ipImageIn <> 0 THEN
```

```
            nNewImageCounter := nNewImageCounter + 1;

            // Trigger next image
            bTrigger := TRUE;

            // Place to call vision algorithms
            hr := F_VN_TransformIntoDisplayableImage(ipImageIn, ipImageInDisp, hr);
        END_IF
    END_IF

TCVN_CS_ERROR:
  hr := fbFileSource.Reset();

END_CASE
```

作为`TriggerImageByName(sFileName)`方法的一个替代方法，`TriggerImage()`可以用于始终从文件源控制列表中获得下一个图像。

样本

- 通过文件名触发图像 [▶ 2820]

相关功能块

- FB_VN_SimpleCameraControl [▶ 1630]用于相机和文件源
- FB_VN_GevCameraControl [▶ 1609]用于 GigE Vision 相机
- FB_VN_FileSourceControl [▶ 1601]用于文件源

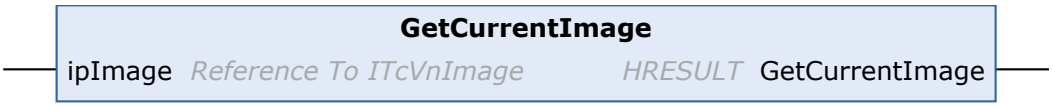
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.4.1.1 GetCurrentImage



获取当前可用的图像（如果有）。

语法

定义：

```
METHOD GetCurrentImage : HRESULT
VAR_INPUT
    ipImage : Reference To ITcVnImage;
END_VAR
```

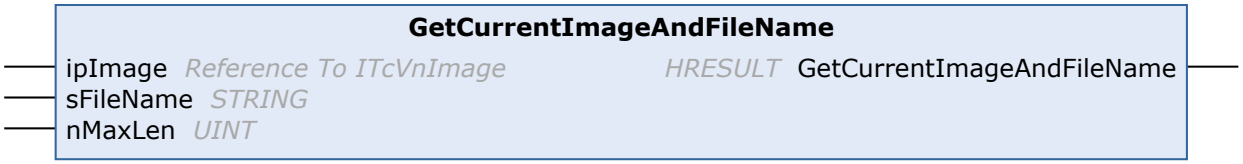
输入

名称	类型	描述
ipImage	引用 ITcVnImage [▶ 407]	返回图像

返回值

HRESULT [▶ 135]

6.1.5.4.1.2 GetCurrentImageAndFileName



获取当前可用的图像（如果有）和相应的文件名。

语法

定义：

```
METHOD GetCurrentImageAndFileName : HRESULT
VAR_INPUT
    ipImage    : Reference To ITcVnImage;
    sFileName  : STRING;
    nMaxLen    : UINT;
END_VAR
```

输入

名称	类型	描述
ipImage	引用 ITcVnImage [▶ 407]	返回图像
sFileName	STRING	返回文件名
nMaxLen	UINT	定义文件名的最大允许长度

返回值

HRESULT [▶ 135]

6.1.5.4.1.3 StartAcquisition



激活文件源。如果不在触发模式下，将开始发送图像。

语法

定义：

```
METHOD StartAcquisition : HRESULT
```

返回值

HRESULT [▶ 135]

更多信息

方法StartAcquisition开始加载图像。根据触发模式设置，这可能意味着直接提供图像或等待触发信号。图像加载可以从INITIAL和OPENED状态开始。

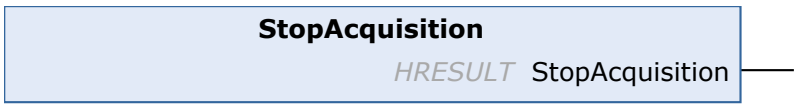
状态转换

通过StartAcquisition，文件源将置于ACQUIRING状态。这可在INITIAL和OPENED状态下进行。如果文件源处于INITIAL，则状态转换首先到OPENED。通常情况下，这通过中间状态OPENING进行。从OPENED到ACQUIRING的过渡通常通过中间状态STARTACQUISITION进行。

所有可能的状态转换和它们的返回代码都显示在以下的转换矩阵中：

返回代码（HRESULT）		结果状态					
		INITIAL	OPENING	OPENED	STARTACQUISITION	ACQUIRING	ERROR
初始状态	INITIAL	BUSY	PENDING	SUCCESS	-	-	FAILED
	OPENING	-	BUSY				
	OPENED		-	BUSY	PENDING	SUCCESS	
	STARTACQUISITION			-	BUSY		

6.1.5.4.1.4 StopAcquisition



停用文件源。将停止发送图像。

语法

定义：

```
METHOD StopAcquisition : HRESULT
```

返回值

HRESULT [▶ 135]

更多信息

方法StopAcquisition停止文件源的图像加载。

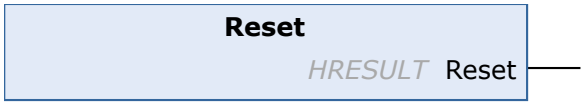
状态转换

通过StopAcquisition，会将文件源从ACQUIRING状态带入OPENED状态。通常情况下，这不会立即发生，而是通过中间状态STOPACQUISITION进行。

所有可能的状态转换和它们的返回代码都显示在以下的转换矩阵中：

返回代码（HRESULT）		结果状态			
		ACQUIRING	STOPACQUISITION	OPENED	ERROR
初始状态	ACQUIRING	BUSY	PENDING	SUCCESS	FAILED
	STOPACQUISITION	-	BUSY		

6.1.5.4.1.5 重置



将控制器重置为初始状态（根据当前状态，可能需要多次调用，直至返回 S_OK）

语法

定义：

```
METHOD Reset : HRESULT
```

返回值

HRESULT [▶ 135]

更多信息

如果文件源处于错误状态，方法Reset将重新设置文件源。

状态转换

通过Reset，会将文件源从ERROR状态带入INITIAL状态。如果复位成功，这个转换立即发生，否则功能块仍处于ERROR状态。

所有可能的状态转换和它们的返回代码都显示在以下的转换矩阵中：

返回代码（HRESULT）		结果状态	
		ERROR	INITIAL
初始状态	ERROR	FAILED	SUCCESS

在成功的情况下，S_OK 或 S_FALSE（如果没有错误需要重置）将作为 SUCCESS 代码返回。

6.1.5.4.1.6 GetState



获取内部相机控制状态机的当前状态。

语法

定义：

```
METHOD GetState : ETcVnCameraState
```

返回值

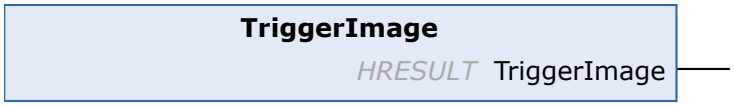
ETcVnCameraState [▶ 162]

更多信息

方法GetState将相机对象或文件源对象的状态作为枚举ETcVnCameraState [▶ 162]的值返回。可能出现以下状态：

- ERROR
- INITIAL
- OPENING
- OPENED
- STARTACQUISITION
- STOPACQUISITION
- ACQUIRING
- TRIGGERING

6.1.5.4.1.7 TriggerImage



如果处于触发模式，该函数将触发列表中的下一个图像。

语法

定义：

```
METHOD TriggerImage : HRESULT
```

返回值

HRESULT [▶ 135]

更多信息

方法 TriggerImage 从文件源触发一个图像。

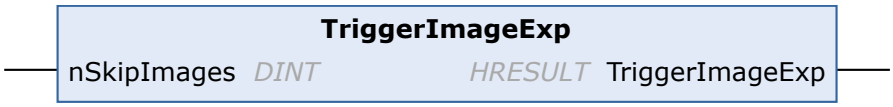
状态转换

通过 TriggerImage，文件源将仅在保持当前 ACQUIRING 状态的情况下加载一个图像。然而，由于图像采集通常需要几个周期，所以文件源通常会短暂地进入中间状态TRIGGERING。

所有可能的状态转换和它们的返回代码都显示在以下的转换矩阵中：

返回代码（HRESULT）		**结果状态**		
		TRIGGERING	**ACQUIRING**	**ERROR**
初始状态	**ACQUIRING**	PENDING	BUSY / SUCCESS	FAILED
	TRIGGERING	BUSY	SUCCESS	

6.1.5.4.1.8 TriggerImageExp



如果处于触发模式，该函数将触发列表中的特定图像。

语法

定义：

```
METHOD TriggerImageExp : HRESULT
VAR_INPUT
    nSkipImages : DINT;
END_VAR
```

输入

名称	类型	描述
nSkipImages	DINT	要跳过的图像的数量（相对于当前图像数量）

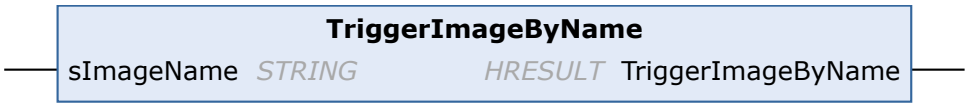
返回值

HRESULT [▶ 135]

更多信息

方法TriggerImageExp的状态转换与TriggerImage [▶ 1608]相同。

6.1.5.4.1.9 TriggerImageByName



如果处于触发模式，此函数会按名称触发列表中的特定图像。

语法

定义：

```
METHOD TriggerImageByName : HRESULT
VAR_INPUT
    sImageName : STRING;
END_VAR
```

输入

名称	类型	描述
sImageName	STRING	要触发的图像名称

返回值

HRESULT [▶ 135]

更多信息

方法TriggerImageByName的状态转换与TriggerImage [▶ 1608]相同。

6.1.5.4.2 FB_VN_GevCameraControl

该 FB 提供控制 GigE Vision 相机和访问其标定数据的基本功能。

请勿直接调用主 FB。只能使用可用的方法。

方法

名称	描述
ClearImageQueue [▶ 1614]	删除 TcVnGevImageProvider TcCOM 模块接收队列中的所有图像，并重置相应的被遗漏的图像计数。
CloseCamera [▶ 1614]	关闭相机的控制通道。
GetCalibPatternRef [▶ 1615]	从 TcVnGevImageProvider TcCOM 模块获取标定板参考点（可通过标定助手设置）
GetCameraMatrix [▶ 1616]	从 TcVnGevImageProvider TcCOM 模块获取相机矩阵（需要固有标定参数，例如由标定助手设置的参数）
GetCurrentImage [▶ 1616]	获取当前可用的图像（接收队列中的第一个）。
GetCurrentImageUndistorted [▶ 1616]	获取当前可用的去畸变图像（接收队列中的第一个）。与包含编码（例如，贝尔（Bayer）图）的像素格式不兼容。需要 TcVnGevImageProvider TcCOM 模块中的固有标定参数，例如由标定助手设置的参数。
GetCurrentImageWithGvspInfo [▶ 1617]	获取当前可用的图像（接收队列中的第一个）及其相应的 GVSP 信息。

名称	描述
GetCurrentImageWithGvspInfoUndistorted [▶ 1617]	获取当前可用的去畸变图像（接收队列中的第一个）及其相应的 GVSP 信息。与包含编码（例如，贝尔（Bayer）图）的像素格式不兼容。需要 TcVnGevImageProvider TcCOM 模块中的固有标定参数，例如由标定助手设置的参数。
GetCurrentImageWithTimestamps [▶ 1618]	获取当前可用的图像（接收队列中的第一个）和相应的时间戳。
GetDistortionCoefficients [▶ 1619]	从 TcVnGevImageProvider TcCOM 模块获取几何畸变系数（需要固有标定参数，例如由标定助手设置的参数）
GetImageQueueSize [▶ 1629]	获取接收队列中当前的图像数量。
GetLastImageFromQueue [▶ 1619]	获取队列中最后收到的图像。
GetOmittedImagesNum [▶ 1620]	获取自上次调用 ClearImageQueue() 起遗漏的图片数量。如果 TcVnGevImageProvider TcCOM 模块中的图像接收队列已满，而又有新图像到达，队列中的第一个图像将被删除，遗漏的计数也会增加。
GetRotationMatrix [▶ 1620]	从 TcVnGevImageProvider TcCOM 模块获取旋转矩阵（需要外部标定参数，例如由标定助手设置的参数）
GetState [▶ 1621]	获取内部相机控制状态机的当前状态。
GetTranslationVector [▶ 1621]	从 TcVnGevImageProvider TcCOM 模块获取平移向量（需要外部标定参数，例如由标定助手设置的参数）
InitializeCamera [▶ 1622]	将相机初始化到预期状态（包括向相机发送 GevImageAcquisition TcCOM 模块中定义的“InitCameraCommands”）。
OpenCamera [▶ 1623]	打开相机的控制通道。
Reset [▶ 1624]	将相机控制器重置为初始状态（根据当前状态可能需要多次调用，直至返回 S_OK）
ResetCameraFeatures [▶ 1625]	将相机特征重置为初始状态（向相机发送 GevImageAcquisition TcCOM 模块中定义的“InitCameraCommands”）。
SetCameraMatrix [▶ 1625]	将相机矩阵设置为 TcVnGevImageProvider TcCOM 模块
SetDistortionCoefficients [▶ 1626]	为 TcVnGevImageProvider TcCOM 模块设置畸变系数
SetRotationMatrix [▶ 1626]	为 TcVnGevImageProvider TcCOM 模块设置旋转矩阵
SetTranslationVector [▶ 1626]	为 TcVnGevImageProvider TcCOM 模块设置平移向量
StartAcquisition [▶ 1627]	向相机发送 TcVnGevImageProvider TcCOM 模块中定义的“StartAcquisitionCommands”。
StopAcquisition [▶ 1629]	向相机发送 TcVnGevImageProvider TcCOM 模块中定义的“StopAcquisitionCommands”。
TriggerImage [▶ 1629]	向相机发送 TcVnGevImageProvider TcCOM 模块中定义的“SoftwareTriggerCommands”。

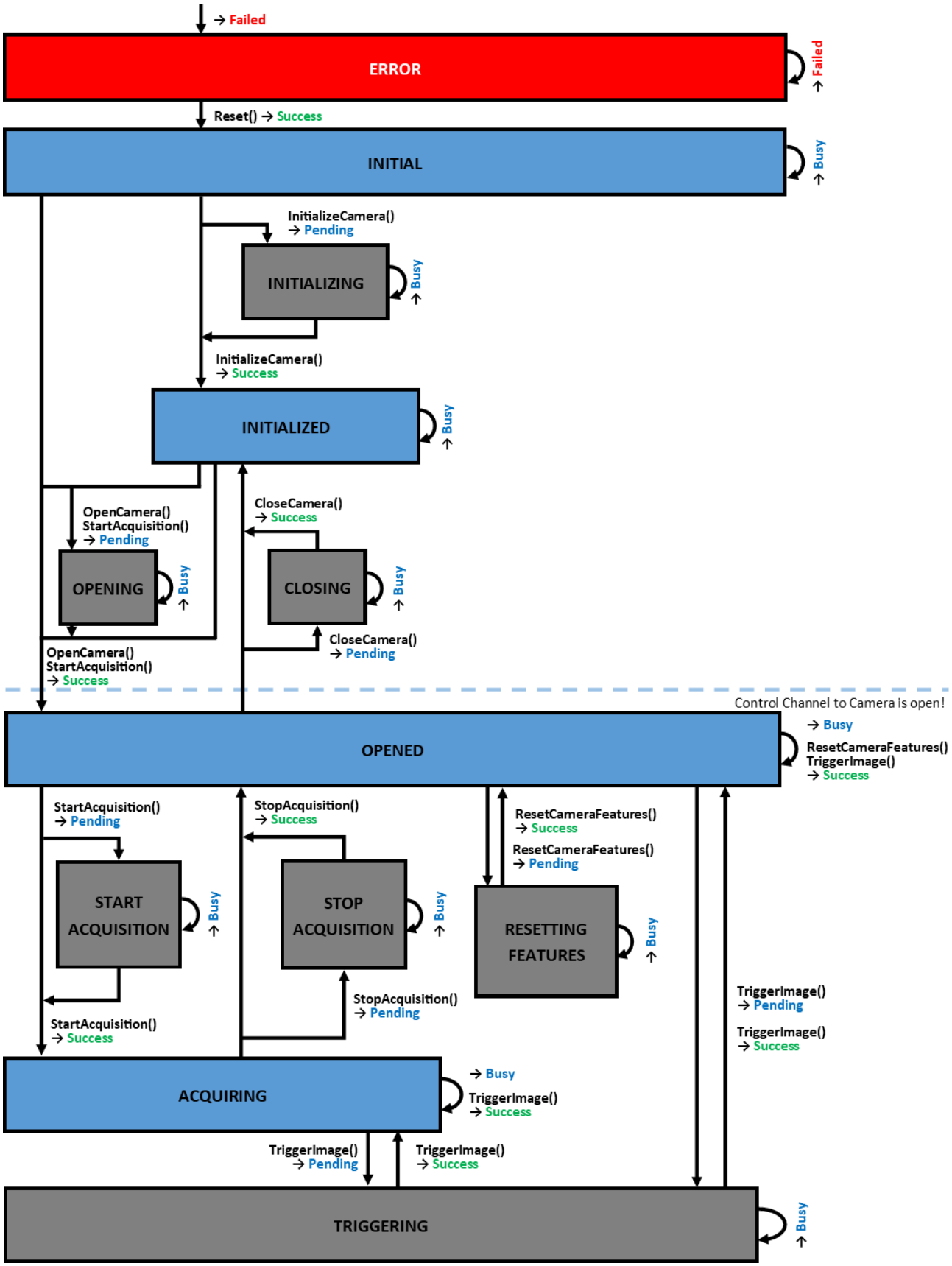
更多信息

功能块 `FB_VN_GevCameraControl` 在 PLC 中充分代表 GigE Vision 相机 [▶ 71]对象。另外，也可以使用功能块 FB_VN_SimpleCameraControl [▶ 1630] 限制访问相机对象。

此功能块不可直接调用。事实上，这些方法包含所有功能。

状态机

下图显示了状态机。主要状态用蓝色标记，过渡状态用灰色标记，错误状态用红色标记。GetState [▶ 1621] 方法可用于查询相机的状态。此外，还显示实现状态转换的方法以及每种方法的返回值。如果存在 Init 命令，并且选择了初始化自动模式 AUTOINIT_AFTER_SO，则还会简单地假设自动初始化发生的 INITIALIZING 状态。



您可以使用 GetState 方法查询当前状态。

以下方法可用于控制状态机：

- GetState [▶ 1621]
- InitializeCamera [▶ 1622]

- OpenCamera [▶ 1623]
- CloseCamera [▶ 1614]
- StartAcquisition [▶ 1627]
- StopAcquisition [▶ 1629]
- TriggerImage [▶ 1629]
- ResetCameraFeatures [▶ 1625]
- Reset [▶ 1624]

如果方法在非预期状态下被调用，则会返回错误代码 `INVALIDSTATE（16#712）`。

HRESULT

Hex	Dec	代码	描述
16#718	1816	NOTINIT	相机尚未初始化。使用 InitializeCamera [▶ 1622] 方法或将 TcCOM 参数 InitializationAutoMode [▶ 99] 设置为 AUTOINIT_SO 或 AUTOINIT_AFTER_SO，以对相机进行初始化。
16#71A	1818	NOINTERFACE	相机功能块未链接到相机 TcCOM 对象。在符号初始化下的 TwinCAT 项目实例中建立此链接。

连续图像采集的应用

例如，将相机置于图像采集模式、处理错误状态并连续显示成功采集图像的状态机如下所示：

```
VAR
    hr              :   HRESULT;
    ipImageIn       :   ITcVnImage;
    ipImageInDisp   :   ITcVnDisplayableImage;
  fbCameraControl :   FB_VN_GevCameraControl;
    eCameraState    :   ETcVnCameraState;
    nNewImageCounter:   UINT;
END_VAR

eCameraState :=   fbCameraControl.GetState();

// CameraControl is in error state, so try to reset the camera connection
IF eCameraState = TCVN_CS_ERROR THEN
  hr :=   fbCameraControl.Reset();

// Camera not yet initialized
ELSIF eCameraState < TCVN_CS_INITIALIZED THEN
  hr :=   fbCameraControl.InitializeCamera();

// Camera not yet opened
ELSIF eCameraState < TCVN_CS_OPENED THEN
  hr := fbCameraControl.OpenCamera();

// Camera not yet streaming
ELSIF eCameraState < TCVN_CS_ACQUIRING THEN
  hr := fbCameraControl.StartAcquisition();

// Camera streaming
ELSIF eCameraState = TCVN_CS_ACQUIRING THEN
  hr := fbCameraControl.GetCurrentImage(ipImageIn);

    // Check if new Image was received
    IF SUCCEEDED(hr) AND ipImageIn <> 0 THEN
        nNewImageCounter := nNewImageCounter + 1;

        // Place to call vision algorithms
        hr := F_VN_TransformIntoDisplayableImage(ipImageIn, ipImageInDisp, hr);
    END_IF

END_IF
```

在这种情况下，可将相机配置为连续图像采集 [▶ 2833]或硬件触发器 [▶ 2833]。

软件触发器的应用

例如，将相机调入图像采集模式、处理错误状态以及通过软件触发器触发图像采集的状态机如下所示：

```
VAR
    hr              :   HRESULT;
    ipImageIn       :   ITcVnImage;
```

```
    ipImageInDisp    :   ITcVnDisplayableImage;
  fbCameraControl :   FB_VN_GevCameraControl;
    eCameraState     :   ETcVnCameraState;
    nNewImageCounter:   UINT;
    bTrigger         :   BOOL := TRUE;
END_VAR

eCameraState :=   fbCameraControl.GetState();

// CameraControl is in error state, so try to reset the camera connection
IF eCameraState = TCVN_CS_ERROR THEN
  hr :=   fbCameraControl.Reset();

// Camera trigger image
ELSIF eCameraState = TCVN_CS_TRIGGERING THEN
  hr := fbCameraControl.TriggerImage();

// Camera not yet initialized
ELSIF eCameraState < TCVN_CS_INITIALIZED THEN
  hr :=   fbCameraControl.InitializeCamera();

// Camera not yet opened
ELSIF eCameraState < TCVN_CS_OPENED THEN
  hr := fbCameraControl.OpenCamera();

// Camera not yet streaming
ELSIF eCameraState < TCVN_CS_ACQUIRING THEN
  hr := fbCameraControl.StartAcquisition();

// Camera streaming
ELSIF eCameraState = TCVN_CS_ACQUIRING THEN

    IF bTrigger THEN
        hr := fbCameraControl.TriggerImage();
        IF SUCCEEDED(hr) THEN
            bTrigger := FALSE;
        END_IF
    ELSE
   hr := fbCameraControl.GetCurrentImage(ipImageIn);

        // Check if new Image was received
        IF SUCCEEDED(hr) AND ipImageIn <> 0 THEN
            nNewImageCounter := nNewImageCounter + 1;

            // Trigger next image
            bTrigger := TRUE;

            // Place to call vision algorithms
            hr := F_VN_TransformIntoDisplayableImage(ipImageIn, ipImageInDisp, hr);
        END_IF
    END_IF

END_IF
```

在这种情况下，必须将相机配置为软件触发器 [▶ 2833]（触发源 = 软件）。

相关功能块

- FB_VN_SimpleCameraControl [▶ 1630]用于相机和文件源
- FB_VN_GevCameraControl [▶ 1609]用于 GigE Vision 相机
- FB_VN_FileSourceControl [▶ 1601]用于文件源

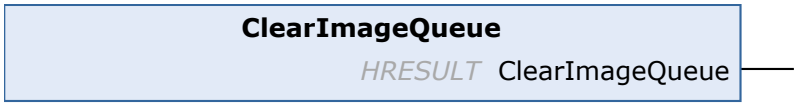
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.4.2.1 ClearImageQueue



删除 TcVnGevImageProvider TcCOM 模块接收队列中的所有图像，并重置相应的被遗漏的图像计数。

语法

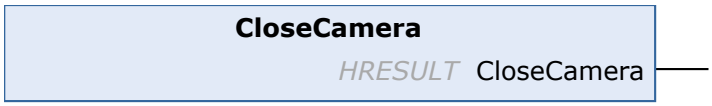
定义：

```
METHOD ClearImageQueue : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.5.4.2.2 CloseCamera



关闭相机的控制通道。

语法

定义：

```
METHOD CloseCamera : HRESULT
```

返回值

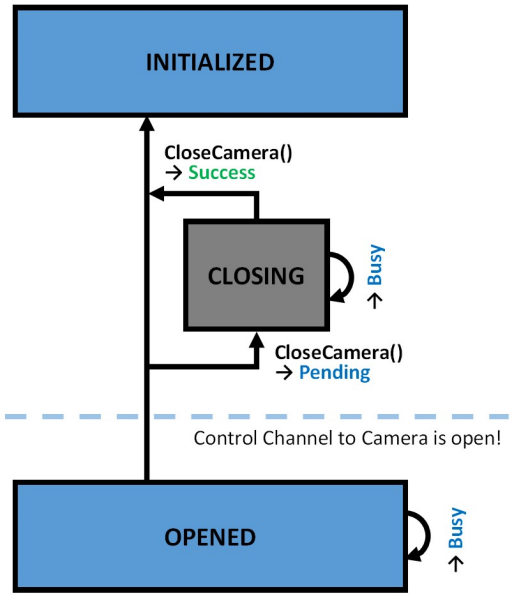
HRESULT [▶ 135]

更多信息

方法CloseCamera将关闭相机打开的通信通道。

状态转换

通过CloseCamera，会让相机从OPENED状态进入INITIALIZED状态。通常，这种转换通过中间状态CLOSING进行。



附图 25: 使用 FB_VN_GevCameraControl.CloseCamera 的状态转换

所有可能的状态转换和它们的返回代码都显示在以下的转换矩阵中：

返回代码（HRESULT）		结果状态			
		OPENED	CLOSING	INITIALIZED	ERROR
初始状态	OPENED	BUSY	PENDING	SUCCESS	FAILED
	CLOSING	-	BUSY		

6.1.5.4.2.3 GetCalibPatternRef



从 TcVnGevImageProvider TcCOM 模块获取标定板参考点（可通过标定助手设置）

语法

定义：

```
METHOD GetCalibPatternRef : HRESULT
VAR_INPUT
    ipCalibPatternRef : Reference To ITcVnContainer;
END_VAR
```

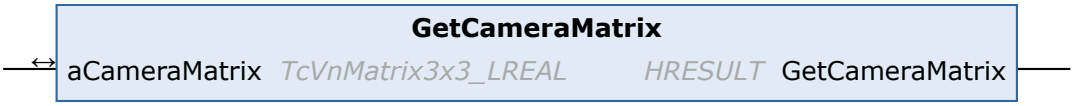
输入

名称	类型	描述
ipCalibPatternRef	引用 ITcVnContainer [▶ 365]	返回标定板参考点（ContainerType_Vector_TcVnPoint3_REAL）

返回值

HRESULT [▶ 135]

6.1.5.4.2.4 GetCameraMatrix



从 TcVnGevImageProvider TcCOM 模块获取相机矩阵（需要固有标定参数，例如由标定助手设置的参数）

语法

定义：

```
METHOD GetCameraMatrix : HRESULT
VAR_IN_OUT
    aCameraMatrix : TcVnMatrix3x3_LREAL;
END_VAR
```

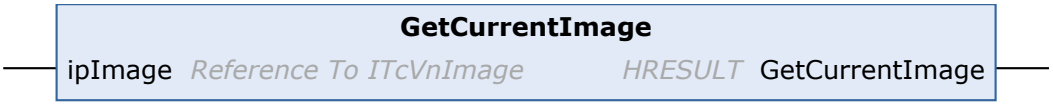
输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回相机矩阵

返回值

HRESULT [▶ 135]

6.1.5.4.2.5 GetCurrentImage



获取当前可用的图像（接收队列中的第一个）。

语法

定义：

```
METHOD GetCurrentImage : HRESULT
VAR_INPUT
    ipImage : Reference To ITcVnImage;
END_VAR
```

输入

名称	类型	描述
ipImage	引用 ITcVnImage [▶ 407]	返回图像

返回值

HRESULT [▶ 135]

6.1.5.4.2.6 GetCurrentImageUndistorted



获取当前可用的去畸变图像（接收队列中的第一个）。与包含编码（例如，贝尔（Bayer）图）的像素格式不兼容。需要 TcVnGevImageProvider TcCOM 模块中的固有标定参数，例如由标定助手设置的参数。

语法

定义：

```
METHOD GetCurrentImageUndistorted : HRESULT
VAR_INPUT
    ipImage : Reference To ITcVnImage;
END_VAR
```

输入

名称	类型	描述
ipImage	引用 ITcVnImage [▶ 407]	返回去畸变图像

返回值

HRESULT [▶ 135]

6.1.5.4.2.7 GetCurrentImageWithGvspInfo



获取当前可用的图像（接收队列中的第一个）及其相应的 GVSP 信息。

语法

定义：

```
METHOD GetCurrentImageWithGvspInfo : HRESULT
VAR_INPUT
    ipImage    : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    stGvspInfo : GVSP_IMAGE_INFO;
END_VAR
```

输入

名称	类型	描述
ipImage	引用 ITcVnImage [▶ 407]	返回图像

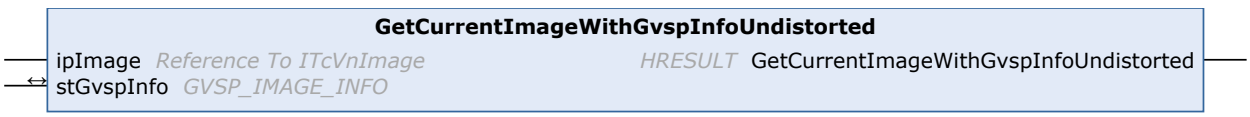
输入/输出

名称	类型	描述
stGvspInfo	GVSP_IMAGE_INFO [▶ 220]	返回 GVSP 信息

返回值

HRESULT [▶ 135]

6.1.5.4.2.8 GetCurrentImageWithGvspInfoUndistorted



获取当前可用的去畸变图像（接收队列中的第一个）及其相应的 GVSP 信息。与包含编码（例如，贝尔（Bayer）图）的像素格式不兼容。需要 TcVnGevImageProvider TcCOM 模块中的固有标定参数，例如由标定助手设置的参数。

语法

定义：

```
METHOD GetCurrentImageWithGvspInfoUndistorted : HRESULT
VAR_INPUT
    ipImage    : Reference To ITcVnImage;
END_VAR
VAR_IN_OUT
    stGvspInfo : GVSP_IMAGE_INFO;
END_VAR
```

输入

名称	类型	描述
ipImage	引用 ITcVnImage [▶ 407]	返回去畸变图像

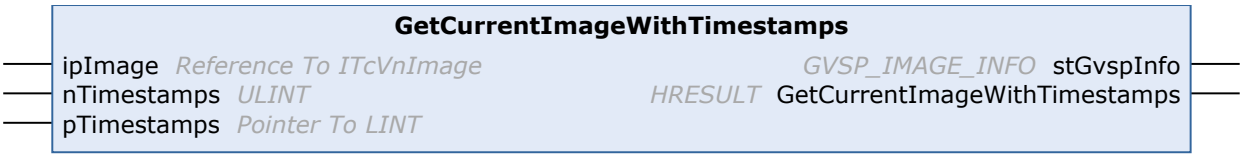
输入/输出

名称	类型	描述
stGvspInfo	GVSP_IMAGE_INFO [▶ 220]	返回 GVSP 信息

返回值

HRESULT [▶ 135]

6.1.5.4.2.9 GetCurrentImageWithTimestamps



获取当前可用的图像（接收队列中的第一个）和相应的时间戳。

语法

定义：

```
METHOD GetCurrentImageWithTimestamps : HRESULT
VAR_INPUT
    ipImage     : Reference To ITcVnImage;
    nTimestamps : ULINT;
    pTimestamps : Pointer To LINT;
END_VAR
VAR_OUTPUT
    stGvspInfo  : GVSP_IMAGE_INFO;
END_VAR
```

输入

名称	类型	描述
ipImage	引用 ITcVnImage [▶ 407]	返回图像
nTimestamps	ULINT	选择要返回的时间戳（使用按位 OR 组合多个 ETcVnTimestamp 值）
pTimestamps	指针指向 LINT	返回请求的时间戳（确保提供足够大小的数组）。

输出

名称	类型	描述
stGvspInfo	GVSP_IMAGE_INFO [▶ 220]	返回 GVSP 信息

返回值

HRESULT [▶ 135]

6.1.5.4.2.10 GetDistortionCoefficients



从 TcVnGevImageProvider TcCOM 模块获取畸变系数（需要固有标定参数，例如由标定助手设置的参数）

语法

定义：

```
METHOD GetDistortionCoefficients : HRESULT
VAR_IN_OUT
    aDistortionCoefficients : TcVnArray8_LREAL;
END_VAR
```

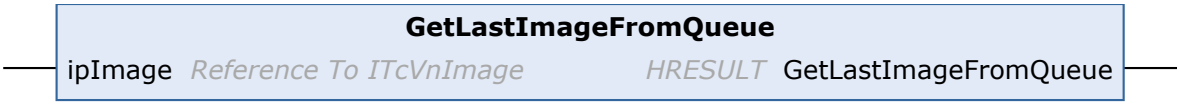
输入/输出

名称	类型	描述
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	返回畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]

返回值

HRESULT [▶ 135]

6.1.5.4.2.11 GetLastImageFromQueue



获取队列中最后收到的图像。

语法

定义：

```
METHOD GetLastImageFromQueue : HRESULT
VAR_INPUT
    ipImage : Reference To ITcVnImage;
END_VAR
```

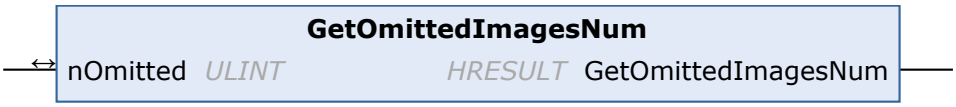
输入

名称	类型	描述
ipImage	引用 ITcVnImage [▶ 407]	返回图像

返回值

HRESULT [▶ 135]

6.1.5.4.2.12 GetOmittedImagesNum



获取自上次调用 ClearImageQueue() 起遗漏的图片数量。如果 TcVnGevImageProvider TcCOM 模块中的图像接收队列已满，而又有新图像到达，队列中的第一个图像将被删除，遗漏的计数也会增加。

语法

定义：

```
METHOD GetOmittedImagesNum : HRESULT
VAR_IN_OUT
    nOmitted : ULINT;
END_VAR
```

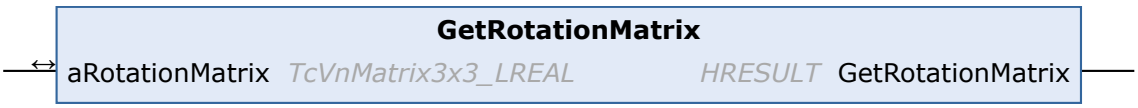
输入/输出

名称	类型	描述
nOmitted	ULINT	返回遗漏图像的数量

返回值

HRESULT [▶ 135]

6.1.5.4.2.13 GetRotationMatrix



从 TcVnGevImageProvider TcCOM 模块获取旋转矩阵（需要外部标定参数，例如由标定助手设置的参数）

语法

定义：

```
METHOD GetRotationMatrix : HRESULT
VAR_IN_OUT
    aRotationMatrix : TcVnMatrix3x3_LREAL;
END_VAR
```

输入/输出

名称	类型	描述
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	返回旋转矩阵

返回值

HRESULT [▶ 135]

6.1.5.4.2.14 GetState



获取内部相机控制状态机的当前状态。

语法

定义：

```
METHOD GetState : ETcVnCameraState
```

返回值

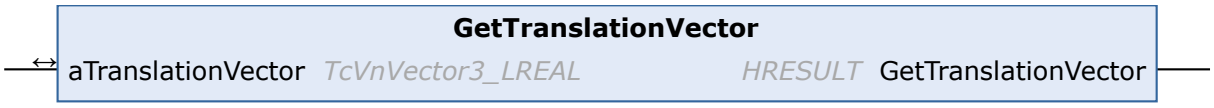
ETcVnCameraState [▶ 162]

更多信息

方法GetState将相机对象的状态作为枚举ETcVnCameraState [▶ 162]的值返回。可能出现以下状态：

- ERROR
- INITIAL
- INITIALIZING
- INITIALIZED
- OPENING
- OPENED
- CLOSING
- STARTACQUISITION
- STOPACQUISITION
- ACQUIRING
- TRIGGERING
- RESETTINGFEATURES

6.1.5.4.2.15 GetTranslationVector



从 TcVnGevImageProvider TcCOM 模块获取平移向量（需要外部标定参数，例如由标定助手设置的参数）

语法

定义：

```
METHOD GetTranslationVector : HRESULT
VAR_IN_OUT
    aTranslationVector : TcVnVector3_LREAL;
END_VAR
```

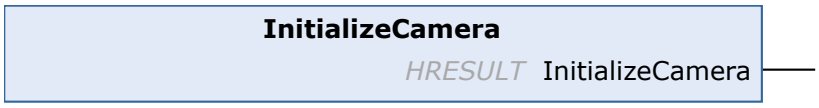
输入/输出

名称	类型	描述
aTranslationVector	TcVnVector3_LREAL [▶ 153]	返回平移向量

返回值

HRESULT [▶ 135]

6.1.5.4.2.16 InitializeCamera



将相机初始化到预期状态（包括向相机发送 GevImageAcquisition TcCOM 模块中定义的“InitCameraCommands”）。

语法

定义：

```
METHOD InitializeCamera : HRESULT
```

返回值

HRESULT [▶ 135]

更多信息

方法InitializeCamera初始化所连接的 GigE Vision 相机，正如存储的初始化命令中已经配置的参数。这个方法应用于明确控制相机的初始化时间。为此，图像采集对象 [▶ 99]的 TcCOM 参数中的InitializationAutoMode必须设置为NO_AUTOINIT。因此，未进行自动的相机初始化；相反，必须通过InitializeCamera手动完成。

如果使用InitializationAutoMode进行了不同的选择，将会发生自动初始化且不需要InitializeCamera方法。

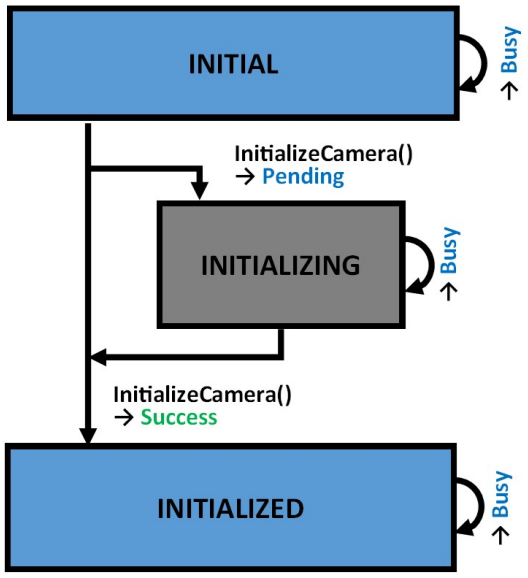
GigE Vision 相机的初始化包括以下步骤：

1. 打开相机的控制通道
2. 执行 Force-IP 命令（可选）
3. 设置 UserSet 或写入初始化命令 [▶ 89]。

在初始化过程中，该功能块处于 INITIALIZING 状态。一旦初始化完成，功能块就会进入 INITIALIZED 状态。如果通过调用CloseCamera [▶ 1614]方法关闭处于 OPENED 状态的相机连接，也会进入为这种状态。

状态转换

通过InitializeCamera，会让相机从INITIAL状态进入INITIALIZED状态。通常，这种转换通过中间状态INITIALIZING进行。

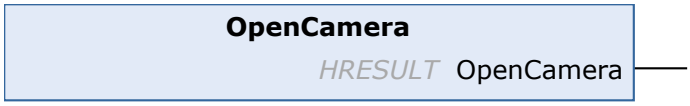


附图 26: 使用 FB_VN_GevCameraControl.InitializeCamera 的状态转换

所有可能的状态转换和它们的返回代码都显示在以下的转换矩阵中：

返回代码（HRESULT）		结果状态			
		INITIAL	INITIALIZING	INITIALIZED	ERROR
初始状态	INITIAL	BUSY	PENDING	SUCCESS	FAILED
	INITIALIZING	-	BUSY		

6.1.5.4.2.17 OpenCamera



打开相机的控制通道。

语法

定义：

```
METHOD OpenCamera : HRESULT
```

返回值

HRESULT [▶ 135]

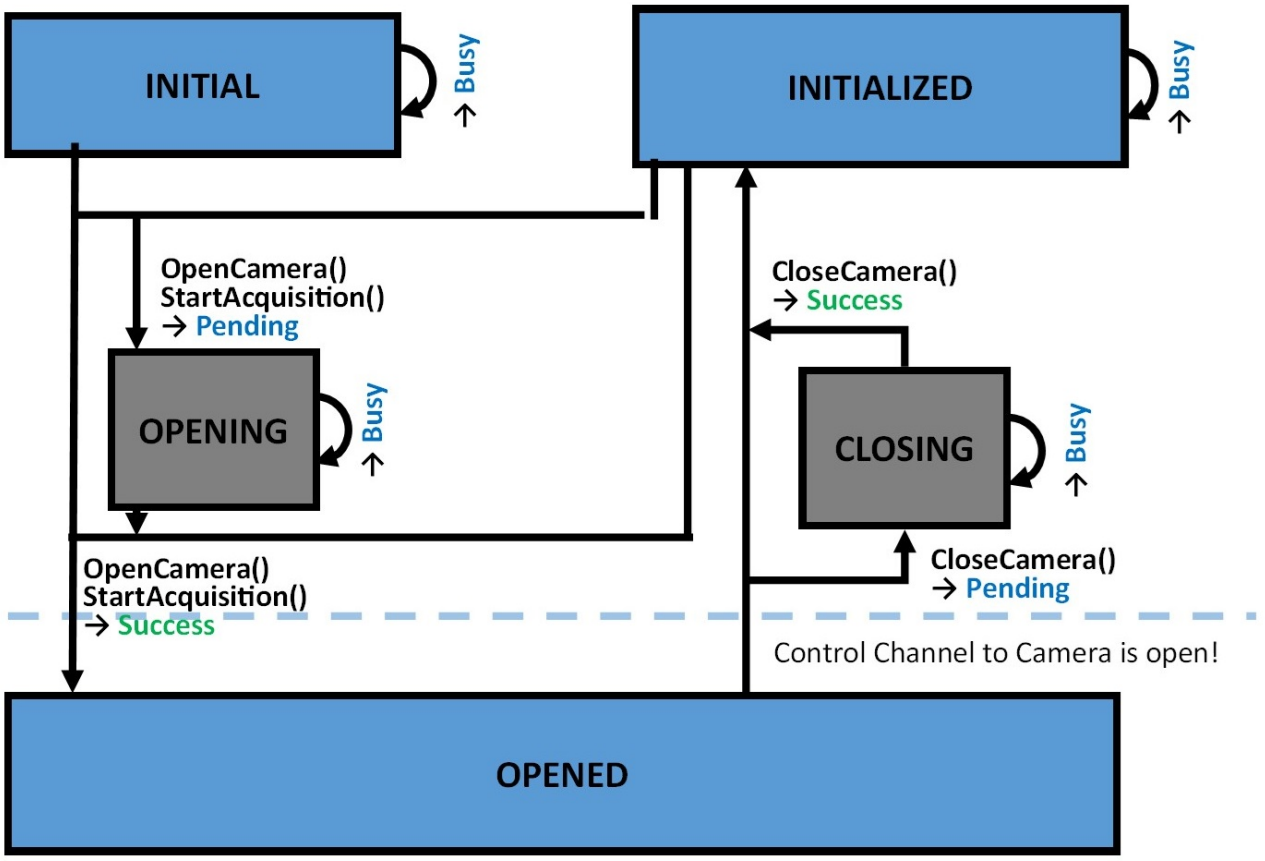
更多信息

方法OpenCamera打开了相机的通信通道。在相应的OPENED状态下，相机寄存器可以被读写，图像可以被采集，且连续的图像记录可以被启动。

如果在图像采集对象 [▶ 99]的 TcCOM 参数中InitializationAutoMode被设置为AUTOINIT_AFTER_SO，那么在第一次调用 OpenCamera 方法时 GigE Vision 相机就会被初始化。其中，这包括写入初始化命令 [▶ 89]或设置 UserSets。另外，也可以调用InitializeCamera [▶ 1622]方法，以便手动执行初始化。

状态转换

通过OpenCamera，相机将置于OPENED状态。这可在INITIAL和INITIALIZED状态下进行。通常，这种转换通过中间状态OPENING进行。



附图 27: 使用 FB_VN_GevCameraControl.OpenCamera 的状态转换

所有可能的状态转换和它们的返回代码都显示在以下的转换矩阵中：

返回代码（HRESULT）		结果状态				
		INITIAL	INITIALIZED	OPENING	OPENED	ERROR
初始状态	INITIAL	BUSY	-	PENDING	SUCCESS	FAILED
	INITIALIZED	-	BUSY			
	OPENING		-	BUSY		

6.1.5.4.2.18 重置

Reset

HRESULT Reset

将相机控制器重置为初始状态（根据当前状态可能需要多次调用，直至返回 S_OK）

语法

定义：

```
METHOD Reset : HRESULT
```

返回值

HRESULT [▶ 135]

更多信息

如果相机处于错误状态，方法Reset将对其进行复位。

状态转换

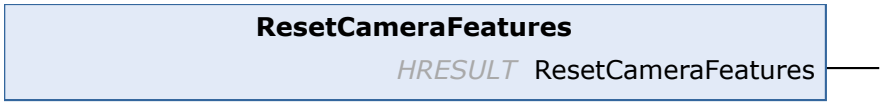
通过Reset，相机功能块应从ERROR状态进入INITIAL状态。如果复位成功，这个转换立即发生，否则功能块仍处于ERROR状态。

所有可能的状态转换和它们的返回代码都显示在以下的转换矩阵中：

返回代码（HRESULT）		结果状态	
		ERROR	INITIAL
初始状态	ERROR	FAILED	SUCCESS

在成功的情况下，S_OK 或 S_FALSE（如果没有错误需要重置）将作为 SUCCESS 代码返回。

6.1.5.4.2.19 ResetCameraFeatures



将相机特征重置为初始状态（向相机发送 GevImageAcquisition TcCOM 模块中定义的“InitCameraCommands”）。

语法

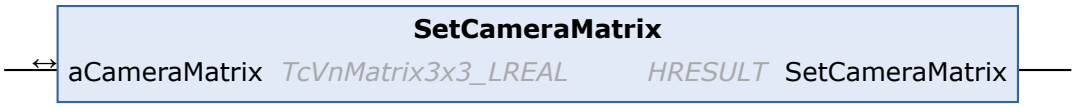
定义：

```
METHOD ResetCameraFeatures : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.5.4.2.20 SetCameraMatrix



将相机矩阵设置为 TcVnGevImageProvider TcCOM 模块

语法

定义：

```
METHOD SetCameraMatrix : HRESULT
VAR_IN_OUT
    aCameraMatrix : TcVnMatrix3x3_LREAL;
END_VAR
```

输入/输出

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 153]	要复制到 TcCOM 模块的相机矩阵

返回值

HRESULT [▶ 135]

6.1.5.4.2.21 SetDistortionCoefficients



为 TcVnGevImageProvider TcCOM 模块设置畸变系数

语法

定义：

```
METHOD SetDistortionCoefficients : HRESULT
VAR_IN_OUT
    aDistortionCoefficients : TcVnArray8_LREAL;
END_VAR
```

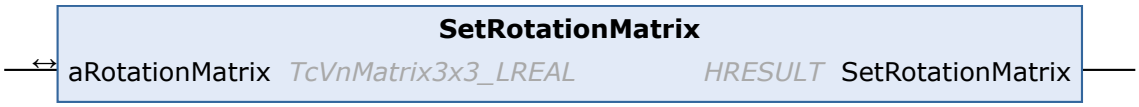
输入/输出

名称	类型	描述
aDistortionCoefficients	TcVnArray8_LREAL [▶ 153]	要复制到 TcCOM 模块的畸变系数

返回值

HRESULT [▶ 135]

6.1.5.4.2.22 SetRotationMatrix



为 TcVnGevImageProvider TcCOM 模块设置旋转矩阵

语法

定义：

```
METHOD SetRotationMatrix : HRESULT
VAR_IN_OUT
    aRotationMatrix : TcVnMatrix3x3_LREAL;
END_VAR
```

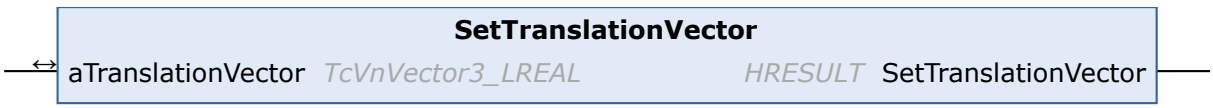
输入/输出

名称	类型	描述
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 153]	要复制到 TcCOM 模块的旋转矩阵

返回值

HRESULT [▶ 135]

6.1.5.4.2.23 SetTranslationVector



为 TcVnGevImageProvider TcCOM 模块设置平移向量

语法

定义：

```
METHOD SetTranslationVector : HRESULT
VAR_IN_OUT
    aTranslationVector : TcVnVector3_LREAL;
END_VAR
```

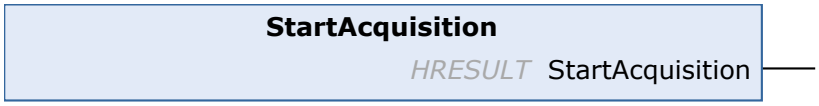
输入/输出

名称	类型	描述
aTranslationVector	TcVnVector3_LREAL [▶ 153]	要复制到 TcCOM 模块的平移向量

返回值

HRESULT [▶ 135]

6.1.5.4.2.24 StartAcquisition



向相机发送 TcVnGevImageProvider TcCOM 模块中定义的“StartAcquisitionCommands”。

语法

定义：

```
METHOD StartAcquisition : HRESULT
```

返回值

HRESULT [▶ 135]

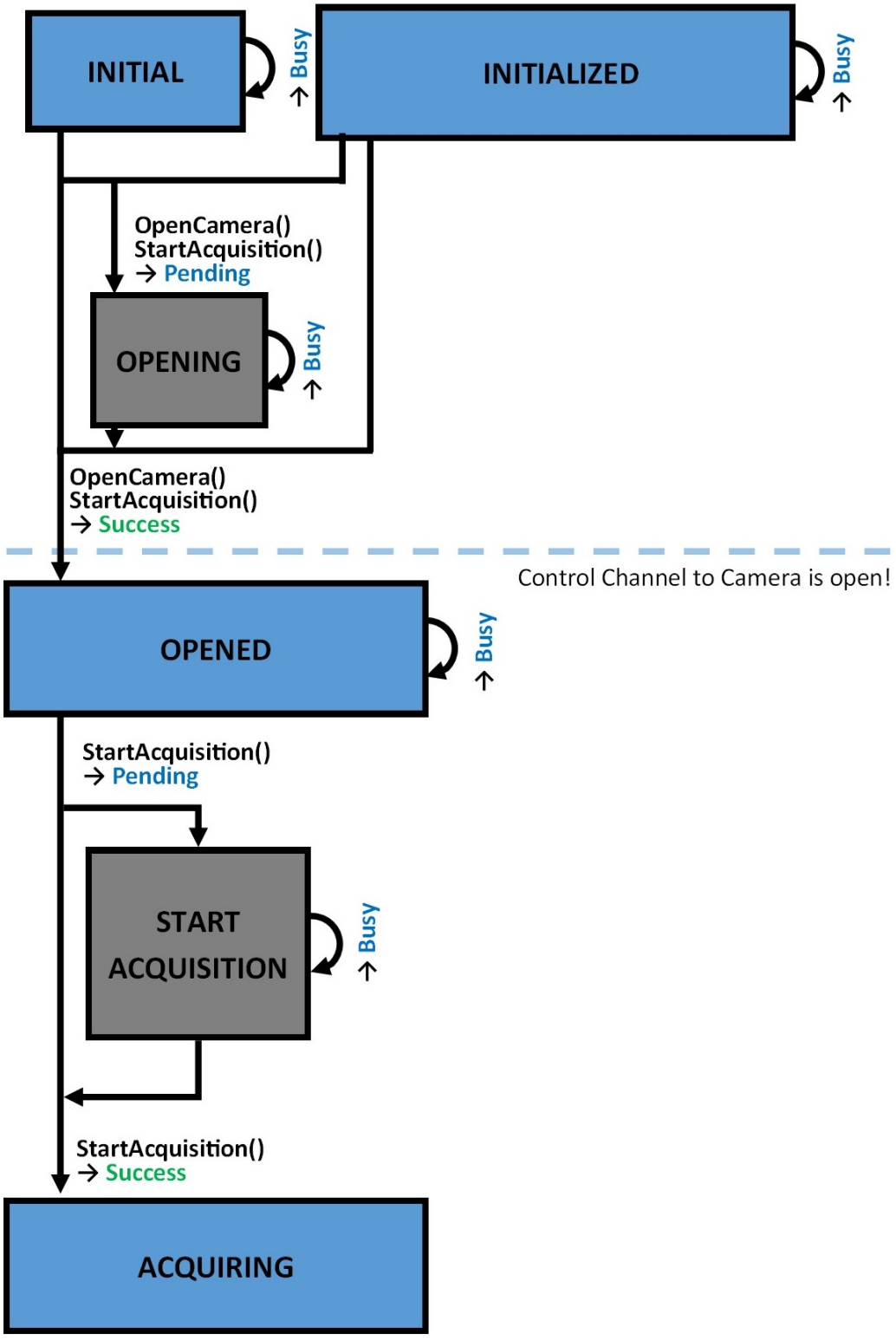
更多信息

方法StartAcquisition启动相机的图像采集。这在个别情况下意味着什么，取决于相机的参数化。关于更多信息，可查看样本图像采集和触发 [▶ 2831]。图像采集可以从INITIAL、INITIALIZED和OPENED状态开始。

如果相机仍在INITIAL或INITIALIZED状态，则方法StartAcquisition还执行方法OpenCamera [▶ 1623]的任务。通常情况下，不会进入OPENED状态。

状态转换

通过StartAcquisition，相机将处于ACQUIRING状态。这可以在INITIAL、INITIALIZED和OPENED状态下进行。如果相机处于 INITIAL 或 INITIALIZED 状态，则首先执行向 OPENED 的状态转换。通常情况下，这通过中间状态 OPENING 进行。通常，从 OPENED 到 ACQUIRING 的转变通过中间状态 STARTACQUISITION 进行。



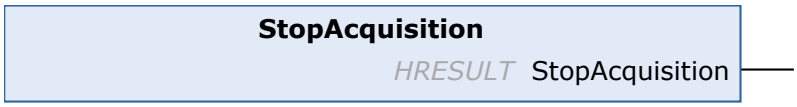
附图 28: 使用 FB_VN_GevCameraControl.StartAcquisition 的状态转换

所有可能的状态转换和它们的返回代码都显示在以下的转换矩阵中：

返回代码 (HRESULT)		**结果状态**						
		INITIAL	**INITIALIZED**	**OPENING**	**OPENED**	**STARTACQUISITION**	**ACQUIRING**	**ERROR**
初始状态	**INITIAL**	BUSY	-	PENDING	SUCCESS	-	-	FAILED

	INITIALIZED	-	BUSY	PENDING				
	OPENING		-	BUSY				
	OPENED			-	BUSY	PENDING	SUCCESS	
	STARTACQUISITION				-	BUSY		

6.1.5.4.2.25 StopAcquisition



向相机发送 TcVnGevImageProvider TcCOM 模块中定义的“StopAcquisitionCommands”。

语法

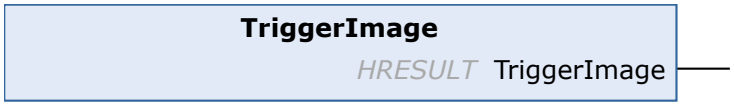
定义：

```
METHOD StopAcquisition : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.5.4.2.26 TriggerImage



向相机发送 TcVnGevImageProvider TcCOM 模块中定义的“SoftwareTriggerCommands”。

语法

定义：

```
METHOD TriggerImage : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.5.4.2.27 GetImageQueueSize



获取接收队列中当前的图像数量。

语法

定义：

```
METHOD GetImageQueueSize : HRESULT
VAR_IN_OUT
    nSize : ULINT;
END_VAR
```

输入/输出

名称	类型	描述
nSize	ULINT	返回接收队列中的图像数量

返回值

HRESULT [▶ 135]

6.1.5.4.3 FB_VN_SimpleCameraControl

该 FB 提供控制相机或文件源的基本功能。

请勿直接调用主 FB。只能使用可用的方法。

方法

名称	描述
GetCurrentImage [▶ 1630]	获取当前可用的图像（如果有）。
GetState [▶ 1631]	获取内部相机控制状态机的当前状态。
Reset [▶ 1631]	将相机重置为初始状态（根据当前状态可能需要多次调用，直至返回 S_OK）
StartAcquisition [▶ 1632]	开始图像采集。
StopAcquisition [▶ 1632]	停止图像采集。
TriggerImage [▶ 1632]	触发下一个图像。

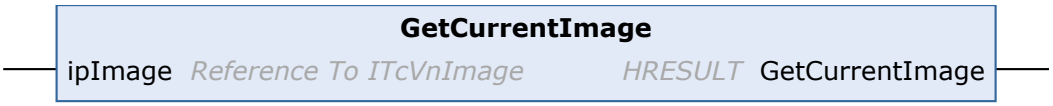
所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.4.3.1 GetCurrentImage



获取当前可用的图像（如果有）。

语法

定义：

```
METHOD GetCurrentImage : HRESULT
VAR_INPUT
    ipImage : Reference To ITcVnImage;
END_VAR
```

输入

名称	类型	描述
ipImage	引用 ITcVnImage [▶ 407]	返回图像

返回值

HRESULT [▶ 135]

6.1.5.4.3.2 GetState



获取内部相机控制状态机的当前状态。

语法

定义：

```
METHOD GetState : ETcVnCameraState
```

返回值

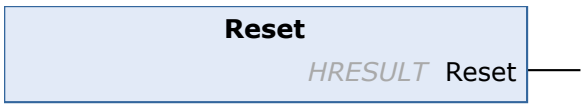
ETcVnCameraState [▶ 162]

更多信息

方法GetState将相机对象或文件源对象的状态作为枚举ETcVnCameraState [▶ 162]的值返回。可能出现以下状态：

- ERROR
- INITIAL
- OPENING
- OPENED
- STARTACQUISITION
- STOPACQUISITION
- ACQUIRING
- TRIGGERING

6.1.5.4.3.3 重置



将相机重置为初始状态（根据当前状态可能需要多次调用，直至返回 S_OK）

语法

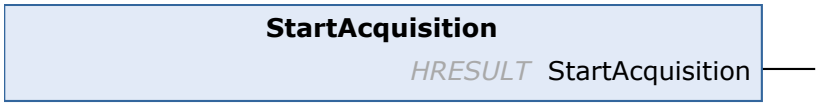
定义：

```
METHOD Reset : HRESULT
```

返回值

HRESULT [▶ 135]

6.1.5.4.3.4 StartAcquisition



开始图像采集。

语法

定义：

```
METHOD StartAcquisition : HRESULT
```

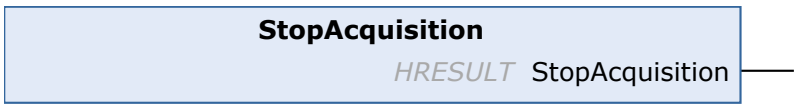
返回值

HRESULT [▶ 135]

更多信息

- 该方法与相机相关，对应于FB_VN_GevCameraControl [▶ 1609]的StartAcquisition [▶ 1627]方法。
- 该方法与文件源控制相关，对应于FB_VN_FileSourceControl [▶ 1601]的StartAcquisition [▶ 1605]方法。

6.1.5.4.3.5 StopAcquisition



停止图像采集。

语法

定义：

```
METHOD StopAcquisition : HRESULT
```

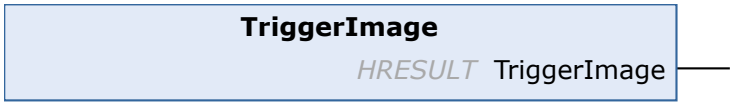
返回值

HRESULT [▶ 135]

更多信息

- 该方法与相机相关，对应于FB_VN_GevCameraControl [▶ 1609]的StopAcquisition [▶ 1629]方法。
- 该方法与文件源控制相关，对应于FB_VN_FileSourceControl [▶ 1601]的StopAcquisition [▶ 1606]方法。

6.1.5.4.3.6 TriggerImage



触发下一个图像。

语法

定义：

```
METHOD TriggerImage : HRESULT
```

返回值

HRESULT [▶ 135]

更多信息

- 该方法与相机相关，对应于FB_VN_GevCameraControl [▶ 1609]的TriggerImage [▶ 1629]方法。
- 该方法与文件源控制相关，对应于FB_VN_FileSourceControl [▶ 1601]的TriggerImage方法。

6.1.5.5 图像处理

该组包含用于图像处理的功能块。在这些综合功能块中，设定的值在执行后会被保留。此外，功能块也可以包含几个方法。

另一方面，章中包含的函数没有内部状态信息，且变量值也不保留。

功能块

- FB_VN_GeneralizedHoughBallard [▶ 1633]
- FB_VN_SSIM [▶ 1638]

6.1.5.5.1 FB_VN_GeneralizedHoughBallard

该 FB 提供广义霍夫变换功能。首先，通过相应的方法设置参数（可选，否则使用默认值）。然后，设置模板（必填）。随后即可执行检测方法。

请勿直接调用主 FB。只能使用可用的方法。

方法

名称	描述
Detect [▶ 1634]	检测图像中的模板（之前必须调用过 SetTemplate）。
SetCannyThreshold [▶ 1635]	设置 Canny 边缘检测阈值。
SetInvAccuRatio [▶ 1635]	设置累加器大小与源图像大小的反比（例如，值为 2 表示大小双向减半。）
SetLevels [▶ 1636]	设置 R 表的级数。
SetMinDist [▶ 1636]	设置不同对象中心之间的最小距离。
SetTemplate [▶ 1637]	设置检测方法中要搜索的模板图像。
SetVotesThreshold [▶ 1637]	设置累加器阈值，即检测匹配所需的最终评估数（数值太小会导致误检测）。

更多信息

功能块FB_VN_GeneralizedHoughBallard通过广义霍夫变换在灰度图像中找到任意形状。所识别模板形状的中心位置被确定，而模板的不同缩放和旋转未考虑在内。

参数

输入图像

模板和输入图像必须是元素类型为 USINT（8 位）的单通道图像。首先，必须使用SetTemplate方法传递模板图像。之后，可以通过调用Detect方法进行执行，然后传递输入图像。

检测的参数

以下参数可以通过方法调用设置。否则，将使用这些默认值。由于在传递模板图像时，这些参数已经用于内部计算，所以必须始终先设置这些值。

CannyThresholdLow = 50

CannyThresholdHigh = 100

投资回报率 = 1.0

Levels = 360

MinDist = 1.0

VotesThreshold = 100

找到的中心点列表（返回值）

所有在图像中找到的中心点位置都会在容器`ipPositions`中调用`Detect`方法后返回。

应用

执行过程包括 3 个步骤。首先设置参数，然后传递模板图像。之后，只需调用`Detect`方法，在图像上搜索模板即可。只有当需要更改参数或模板时，相应的步骤才需要重新执行。

```
// Set parameters first
hr := fbGenHoughBallard.SetCannyThreshold(50,150);
IF SUCCEEDED(hr) THEN hr := fbGenHoughBallard.SetInvAccuRatio(2); END_IF
IF SUCCEEDED(hr) THEN hr := fbGenHoughBallard.SetLevels(180); END_IF
IF SUCCEEDED(hr) THEN hr := fbGenHoughBallard.SetMinDist(20); END_IF
IF SUCCEEDED(hr) THEN hr := fbGenHoughBallard.SetVotesThreshold(80); END_IF

// Check if successful and set template image
IF SUCCEEDED(hr) THEN hr := fbGenHoughBallard.SetTemplate(ipTemplateImage); END_IF

// Check if successful and call detection
IF SUCCEEDED(hr) THEN hr := fbGenHoughBallard.Detect(ipSrcImage,ipPositions); END_IF
```

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.5.1.1 检测



检测图像中的模板（之前必须调用过 SetTemplate）。

语法

定义：

```
METHOD Detect : HRESULT
VAR_INPUT
    ipImage     : ITcVnImage;
    ipPositions : Reference To ITcVnContainer;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	源图像（1 通道，USINT）

名称	类型	描述
ipPositions	引用 ITcVnContainer [▶ 365]	返回检测到的模板位置的中心（ContainerType_Vector_TcVnPoint2_REAL；重复使用非零接口指针。）

返回值

HRESULT [▶ 135]

6.1.5.5.1.2 SetCannyThreshold



设置 Canny 边缘检测阈值。

语法

定义：

```
METHOD SetCannyThreshold : HRESULT
VAR_INPUT
    nLow  : DINT;
    nHigh : DINT;
END_VAR
```

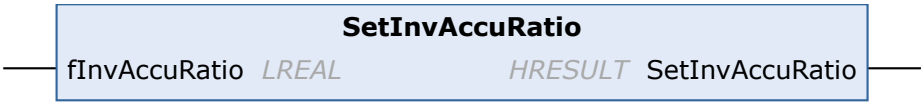
输入

名称	类型	描述
nLow	DINT	低阈值（> 0）
nHigh	DINT	高阈值（> nLow，通常为 2 至 3 * nLow）

返回值

HRESULT [▶ 135]

6.1.5.5.1.3 SetInvAccuRatio



设置累加器大小与源图像大小的反比（例如，值为 2 表示大小双向减半。）

语法

定义：

```
METHOD SetInvAccuRatio : HRESULT
VAR_INPUT
    fInvAccuRatio : LREAL;
END_VAR
```

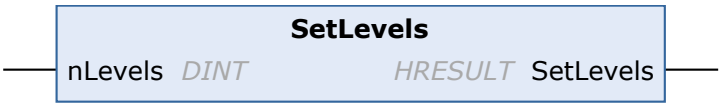
输入

名称	类型	描述
fInvAccuRatio	LREAL	反向累加器比率（> 0）

返回值

HRESULT [▶ 135]

6.1.5.5.1.4 SetLevels



设置 R 表的级数。

语法

定义：

```
METHOD SetLevels : HRESULT
VAR_INPUT
    nLevels : DINT;
END_VAR
```

输入

名称	类型	描述
nLevels	DINT	R 表级别数（> 0）

返回值

HRESULT [▶ 135]

6.1.5.5.1.5 SetMinDist



设置不同对象中心之间的最小距离。

语法

定义：

```
METHOD SetMinDist : HRESULT
VAR_INPUT
    fMinDist : LREAL;
END_VAR
```

输入

名称	类型	描述
fMinDist	LREAL	最小距离（> 0）

返回值

HRESULT [▶ 135]

6.1.5.5.1.6 SetTemplate



设置检测方法中要搜索的模板图像。

语法

定义：

```
METHOD SetTemplate : HRESULT
VAR_INPUT
    ipImage : ITcVnImage;
END_VAR
```

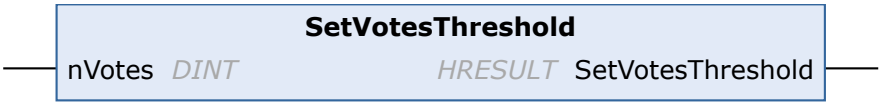
输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	模板图像（1 通道，USINT）

返回值

HRESULT [▶ 135]

6.1.5.5.1.7 SetVotesThreshold



设置累加器阈值，即检测匹配所需的最终评估数（数值太小会导致误检测）。

语法

定义：

```
METHOD SetVotesThreshold : HRESULT
VAR_INPUT
    nVotes : DINT;
END_VAR
```

输入

名称	类型	描述
nVotes	DINT	检测匹配所需的最终评估数（> 0）

返回值

HRESULT [▶ 135]

6.1.5.5.2 FB_VN_SSIM

该 FB 提供结构相似性（SSIM）功能。首先，设置参考图像。随后即可执行计算方法。

请勿直接调用主 FB。只能使用可用的方法。

方法

名称	描述
Compute [▶ 1638]	计算结构相似性（之前必须调用过 SetReference）。
SetReference [▶ 1639]	设置 SSIM 方法使用的参考图像。

更多信息

功能块FB_VN_SSIM计算两个灰度图像的结构相似性。作为参考，假设是一个无干扰的图像。

参数

输入图像

参考和输入图像必须是元素类型为USINT（8 位）的单通道图像。首先，必须使用SetReference 方法传递参考图像。之后，可以通过调用Compute方法进行执行，然后传递输入图像。

输出图像 (返回值)

输出图像的类型是Real，并按 0-1 的数值范围进行缩放。1 表示同意，而 0 表示最大偏离。

应用

执行过程包括 2 个步骤。首先，必须设置参考图像。之后，可以调用Compute方法，该方法被传递给要比较的当前输入图像。

为了进一步分析结果图像，例如使用F_VN_DetectBlobs [▶ 1171]函数，需要将图像转换成USINT。函数F_VN_ConvertElementTypeExp [▶ 776]可用于此目的。通过函数F_VN_ImageAverage [▶ 1223]或F_VN_MinPixelValue [▶ 1233]，可以直接根据结果图像确定匹配值。

```
// Set reference image first
hr := fbSSIM.SetReference(ipImage := ipRefImage);

// Check if successful, set imput image and compute the result
IF SUCCEEDED(hr) and ipRefImage <> 0 THEN
    hr := fbSSIM.Compute(ipSrcImage:= ipImageIn, ipSSIM:= ipResultImage);
    // Convert and scale the result image to USINT
    hr := F_VN_ConvertElementTypeExp(ipResultImage, ipResult2Image, ETcVnElementType.TCVN_ET_USINT,
(255.0/1.0), 0, hr);
END_IF
```

大图像的高内存使用率

由于该函数在后台用真实值进行许多计算，因此根据图像大小需要更多的路由器内存和计算能力。

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.1.5.5.2.1 计算



计算结构相似性（之前必须调用过 SetReference）。

语法

定义：

```
METHOD Compute : HRESULT
VAR_INPUT
    ipSrcImage : ITcVnImage;
    ipSSIM     : Reference To ITcVnImage;
END_VAR
```

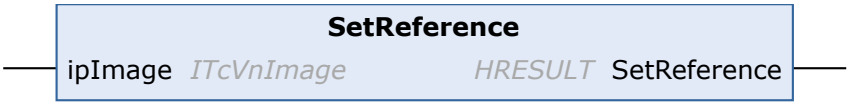
输入

名称	类型	描述
ipSrcImage	ITcVnImage [▶ 407]	源图像（USINT）
ipSSIM	引用 ITcVnImage [▶ 407]	返回 SSIM（REAL，数值范围 0 至 1）

返回值

HRESULT [▶ 135]

6.1.5.5.2.2 SetReference



设置 SSIM 方法使用的参考图像。

语法

定义：

```
METHOD SetReference : HRESULT
VAR_INPUT
    ipImage : ITcVnImage;
END_VAR
```

输入

名称	类型	描述
ipImage	ITcVnImage [▶ 407]	参考图像（USINT）

返回值

HRESULT [▶ 135]

6.2 C++

本章包含用于在 C++ 中编程使用库的 TwinCAT Vision API 说明。

- 数据类型 [▶ 1640]
  （别名、常量、数组、枚举、结构）
- 接口 [▶ 1719]
  （图像、容器等）
- 函数 [▶ 1882]
  （图像处理）
- 功能块 [▶ 2613]
  （与相机和文件系统通信、图像处理）

6.2.1 Data Types

共有四种数据类型，每种类型的前缀都是“TcVn”：

- 某些数组类型的别名 [▶ 1640]涉及图像处理
- 具有特定数据类型组合和定义数的数组 [▶ 1641]经常被图像处理函数使用。
- 枚举 [▶ 1642]有助于函数的参数设置，并包含附加前缀“E”
- 结构 [▶ 1697]整合多个相关信息片段（如几何描述）

此外，还提供用于描述容器类型的常量 [▶ 1640]。

6.2.1.1 Aliases

名称	类型
GVCP_REGISTER_ADDRESS	ULONG
GVCP_REGISTER_VALUE	ULONG
TcVnPoint	TcVnPoint2_DINT [▶ 1640]
TcVnPoint2_DINT	TcVnVector2_DINT [▶ 1641]
TcVnPoint2_LREAL	TcVnVector2_LREAL [▶ 1641]
TcVnPoint2_REAL	TcVnVector2_REAL [▶ 1641]
TcVnPoint3_LREAL	TcVnVector3_LREAL [▶ 1641]
TcVnPoint3_REAL	TcVnVector3_REAL [▶ 1641]
TcVnRectangle	TcVnRectangle_DINT [▶ 1717]

6.2.1.2 Constants

以下常量将容器 [▶ 146]类型表示为 `GUID`。例如，在使用 CreateContainer [▶ 2141] 等函数时就需要这些。

Name	Type
ContainerType_Vector_SINT	GUID
ContainerType_Vector_USINT	GUID
ContainerType_Vector_INT	GUID
ContainerType_Vector_UINT	GUID
ContainerType_Vector_DINT	GUID
ContainerType_Vector_UDINT	GUID
ContainerType_Vector_ULINT	GUID
ContainerType_Vector_REAL	GUID
ContainerType_Vector_LREAL	GUID
ContainerType_Vector_ITcVnImage	GUID
ContainerType_Vector_TcVnPoint2_DINT	GUID
ContainerType_Vector_TcVnPoint2_REAL	GUID
ContainerType_Vector_TcVnPoint2_LREAL	GUID
ContainerType_Vector_TcVnPoint3_REAL	GUID
ContainerType_Vector_TcVnPoint3_LREAL	GUID
ContainerType_Vector_Vector_TcVnPoint2_DINT	GUID
ContainerType_Vector_Vector_TcVnPoint2_REAL	GUID
ContainerType_Vector_Vector_TcVnPoint3_REAL	GUID
ContainerType_Vector_Vector_UDINT	GUID
ContainerType_Vector_Vector_ULINT	GUID

Name	Type
ContainerType_Vector_Vector_REAL	GUID
ContainerType_Vector_Vector_LREAL	GUID
ContainerType_Vector_TcVnVector2_SINT	GUID
ContainerType_Vector_TcVnVector2_USINT	GUID
ContainerType_Vector_TcVnVector2_INT	GUID
ContainerType_Vector_TcVnVector2_UINT	GUID
ContainerType_Vector_TcVnVector2_DINT	GUID
ContainerType_Vector_TcVnVector2_REAL	GUID
ContainerType_Vector_TcVnVector3_SINT	GUID
ContainerType_Vector_TcVnVector3_USINT	GUID
ContainerType_Vector_TcVnVector3_INT	GUID
ContainerType_Vector_TcVnVector3_UINT	GUID
ContainerType_Vector_TcVnVector3_REAL	GUID
ContainerType_Vector_TcVnVector4_SINT	GUID
ContainerType_Vector_TcVnVector4_USINT	GUID
ContainerType_Vector_TcVnVector4_INT	GUID
ContainerType_Vector_TcVnVector4_UINT	GUID
ContainerType_Vector_TcVnVector4_DINT	GUID
ContainerType_Vector_TcVnVector4_REAL	GUID
ContainerType_Vector_TcVnVector4_LREAL	GUID
ContainerType_Vector_TcVnRectangle_DINT	GUID
ContainerType_Vector_TcVnRotatedRectangle	GUID
ContainerType_Vector_TcVnKeyPoint	GUID
ContainerType_Vector_TcVnDMatch	GUID
ContainerType_Vector_Vector_TcVnDMatch	GUID
ContainerType_String_SINT	GUID
ContainerType_Vector_String_SINT	GUID

6.2.1.3 Arrays

Name	Type
TcVnArray10_ITcVnImage	ITcVnImage* [▶ 1863][10]
TcVnArray3_Point2_REAL	TcVnPoint2_REAL [▶ 1640][3]
TcVnArray33_UDINT	ULONG[33]
TcVnArray4_LREAL	double[4]
TcVnArray4_Point2_REAL	TcVnPoint2_REAL [▶ 1640][4]
TcVnArray7_LREAL	double[7]
TcVnArray8_LREAL	double[8]
TcVnMatrix2x3_LREAL	double[2][3]
TcVnMatrix3x3_LREAL	double[3][3]
TcVnVector2_DINT	LONG[2]
TcVnVector2_INT	SHORT[2]
TcVnVector2_LREAL	double[2]
TcVnVector2_REAL	float[2]
TcVnVector2_SINT	char[2]
TcVnVector2_UINT	USHORT[2]

Name	Type
TcVnVector2_USINT	unsigned char[2]
TcVnVector3_INT	SHORT[3]
TcVnVector3_LREAL	double[3]
TcVnVector3_REAL	float[3]
TcVnVector3_SINT	char[3]
TcVnVector3_UINT	USHORT[3]
TcVnVector3_USINT	unsigned char[3]
TcVnVector4_DINT	LONG[4]
TcVnVector4_INT	SHORT[4]
TcVnVector4_LREAL	double[4]
TcVnVector4_REAL	float[4]
TcVnVector4_SINT	char[4]
TcVnVector4_UINT	USHORT[4]
TcVnVector4_USINT	unsigned char[4]

6.2.1.4 Enums

6.2.1.4.1 ETcVnDetectBarcodesWalshOptions

Offers options for Walsh barcode detection.

Syntax

Definition:

```
enum ETcVnDetectBarcodesWalshOptions : ULONG
{
    DBWO_HORIZONTAL     = 1,
    DBWO_VERTICAL       = 2,
    DBWO_DIAGONAL_45    = 4,
    DBWO_DIAGONAL_135   = 8,
    DBWO_ALL_DIRECTIONS = 15
};
```

Values

Name	Description
DBWO_HORIZONTAL	Localize horizontal barcodes.
DBWO_VERTICAL	Localize vertical barcodes.
DBWO_DIAGONAL_45	Localize diagonal barcodes rotated by 45 degrees.
DBWO_DIAGONAL_135	Localize diagonal barcodes rotated by 135 degrees.
DBWO_ALL_DIRECTIONS	Localize arbitrarily rotated barcodes.

6.2.1.4.2 ETcVnEdgeSelection

Offers edge selection options.

Syntax

Definition:

```
enum ETcVnEdgeSelection : LONG
{
    ES_RISING  = 1,
```

```
    ES_FALLING = 2,
    ES_ANY     = 3
};
```

Values

Name	Description
ES_RISING	Only search for rising edges, i.e. dark to light transitions
ES_FALLING	Only search for falling edges, i.e. light to dark transitions
ES_ANY	Search for any edge direction, i.e. rising or falling

6.2.1.4.3 ETcVnFindRoisMethod

Offers ROI extraction methods.

Syntax

Definition:

```
enum ETcVnFindRoisMethod : ULONG
{
    FRM_GRID                 = 0,
    FRM_CONNECTEDCOMPONENTS = 1
};
```

Values

Name	Description
FRM_GRID	Extracts ROIs by laying a regular grid of stRoiSize cells (see TcVnParamsFindRois) over the image and keeping only cells that contain sufficient texture. This method supports both binary and grayscale images.
FRM_CONNECTEDCOMPONENTS	Extracts ROIs using connected-component analysis. The binary image is dilated to bridge nearby components, and then labeled.

6.2.1.4.4 ETcVnJpegSampling

Offers sampling factors for jpeg encoding, to trade off chroma resolution against file size.

Syntax

Definition:

```
enum ETcVnJpegSampling : LONG
{
    JS_444 = 11,
    JS_440 = 12,
    JS_422 = 21,
    JS_420 = 22,
    JS_411 = 41,
    JS_410 = 42
};
```

Values

Name	Description
JS_444	Sampling factors 1x1, 1x1, 1x1 (full chroma resolution)
JS_440	Sampling factors 1x2, 1x1, 1x1

Name	Description
JS_422	Sampling factors 2x1, 1x1, 1x1
JS_420	Sampling factors 2x2, 1x1, 1x1 (default)
JS_411	Sampling factors 4x1, 1x1, 1x1
JS_410	Sampling factors 4x2, 1x1, 1x1 (lowest chroma resolution)

6.2.1.4.5 ETcVnMergeRegionsOptions

Offers different options to merge regions.

Syntax

Definition:

```
enum ETcVnMergeRegionsOptions : ULONGLONG
{
    MRO_UPRIGHT_BOUNDING_RECT = 1
};
```

Values

Name	Description
MRO_UPRIGHT_BOUNDING_RECT	Merge each group into an axis-aligned (upright) bounding rectangle.

6.2.1.4.6 ETcVnPolarizedImageInterpolation

Offers interpolation options for polarized image demosaicing.

Syntax

Definition:

```
enum ETcVnPolarizedImageInterpolation : LONG
{
    PII_NONE             = -1,
    PII_NEAREST_NEIGHBOR = 0,
    PII_BILINEAR         = 1,
    PII_BICUBIC          = 2,
    PII_LANCZOS4         = 4
};
```

Values

Name	Description
PII_NONE	No interpolation, i.e. demosaicing returns images with half width and height.
PII_NEAREST_NEIGHBOR	Images are scaled to full resolution using nearest neighbor interpolation.
PII_BILINEAR	Images are scaled to full resolution using bilinear interpolation.
PII_BICUBIC	Images are scaled to full resolution using bicubic interpolation.
PII_LANCZOS4	Images are scaled to full resolution using Lanczos4 interpolation.

6.2.1.4.7 ETcVnSalientFeaturesMethod

Offers methods to compute salient features.

Syntax

Definition:

```
enum ETcVnSalientFeaturesMethod : ULONG
{
    SFM_WALSH_CODE = 0,
    SFM_SOBEL      = 1
};
```

Values

Name	Description
SFM_WALSH_CODE	Walsh transform-based code detection.
SFM_SOBEL	Sobel and Robinson compass filters for code detection.

6.2.1.4.8 ETcVn2dCodeSearchStrategy

Offers search strategies for 2d code reading (multiple TCVN_CSS_XXX_INVERTED or multiple TCVN_CSS_XXX_FLIPPED cannot be combined).

Syntax

Definition:

```
enum ETcVn2dCodeSearchStrategy : ULONG
{
    CSS_DEFAULT            = 1,
    CSS_ONLY_NOT_INVERTED  = 8,
    CSS_FIRST_NOT_INVERTED = 10,
    CSS_ONLY_INVERTED      = 12,
    CSS_FIRST_INVERTED     = 14,
    CSS_ONLY_NOT_FLIPPED   = 64,
    CSS_FIRST_NOT_FLIPPED  = 80,
    CSS_ONLY_FLIPPED       = 96,
    CSS_FIRST_FLIPPED      = 112
};
```

Values

Name	Description
CSS_DEFAULT	The algorithm searches for codes using the default searching strategy, all other flags are ignored (dependant on code type).
CSS_ONLY_NOT_INVERTED	The algorithm searches for codes only in the non-inverted image.
CSS_FIRST_NOT_INVERTED	The algorithm first searches for codes in the non-inverted and then in the inverted image.
CSS_ONLY_INVERTED	The algorithm searches for codes only in the inverted image.
CSS_FIRST_INVERTED	The algorithm first searches for codes in the inverted and then in the non-inverted image.
CSS_ONLY_NOT_FLIPPED	The algorithm searches for codes only in the non-mirrored image.
CSS_FIRST_NOT_FLIPPED	The algorithm first searches for codes in the non-mirrored and then in the mirrored image.
CSS_ONLY_FLIPPED	The algorithm searches for codes only in the mirrored image.
CSS_FIRST_FLIPPED	The algorithm first searches for codes in the mirrored and then in the non-mirrored image.

6.2.1.4.9 ETcVnAdaptiveThresholdMethod

Offers methods for adaptive threshold.

Syntax

Definition:

```
enum ETcVnAdaptiveThresholdMethod : LONG
{
    ATM_MEAN     = 0,
    ATM_GAUSSIAN = 1
};
```

Values

Name	Description
ATM_MEAN	The threshold value is calculated as the mean of the nBlockSize x nBlockSize neighborhood of (x,y) minus fConstant.
ATM_GAUSSIAN	The threshold value is the weighted sum (cross-correlation with a Gaussian window) of the nBlockSize x nBlockSize neighborhood of (x,y) minus fConstant.

6.2.1.4.10 ETcVnBarcodeSearchDirection

Offers search directions for linear barcodes.

Syntax

Definition:

```
enum ETcVnBarcodeSearchDirection : ULONG
{
    BSD_ANY        = 0,
    BSD_HORIZONTAL = 1,
    BSD_VERTICAL   = 2
};
```

Values

Name	Description
BSD_ANY	The algorithm searches for codes in any supported directions.
BSD_HORIZONTAL	The algorithm searches for codes in horizontal direction.
BSD_VERTICAL	The algorithm searches for codes in vertical direction.

6.2.1.4.11 ETcVnBarcodeType

Offers linear barcode types to search for.

Syntax

Definition:

```
enum ETcVnBarcodeType : ULONG
{
    BT_CODABAR        = 4,
    BT_CODE39         = 8,
    BT_CODE93         = 16,
    BT_CODE128        = 32,
    BT_EAN8           = 128,
```

```
    BT_EAN13          = 256,
    BT_ITF            = 512,
    BT_UPCA           = 32768,
    BT_UPCE           = 65536,
    BT_ANY            = 99260,
    BT_CODE39EXTENDED = 262144
};
```

Values

Name	Description
BT_CODABAR	The algorithm searches for Codabar codes.
BT_CODE39	The algorithm searches for Code-39 codes.
BT_CODE93	The algorithm searches for Code-93 codes.
BT_CODE128	The algorithm searches for Code-128 codes.
BT_EAN8	The algorithm searches for EAN-8 codes.
BT_EAN13	The algorithm searches for EAN-13 codes.
BT_ITF	The algorithm searches for ITF codes.
BT_UPCA	The algorithm searches for UPC-A codes.
BT_UPCE	The algorithm searches for UPC-E codes.
BT_ANY	The algorithm searches for any supported linear barcode. If the type of the code is known, it is recommended to select the specific type directly.
BT_CODE39EXTENDED	The algorithm searches for Code-93-Extended codes.

6.2.1.4.12 ETcVnBlobCombination

Offers multiple blob contours to choose from, which should be returned in a multi-threshold scenario (used in TcVnParamsBlobDetection, which in return is used in F_VN_DetectBlobs).

Syntax

Definition:

```
enum ETcVnBlobCombination : LONG
{
    BC_SMALLEST         = 0,
    BC_LARGEST          = 1,
    BC_MIN_THRESHOLD    = 2,
    BC_MAX_THRESHOLD    = 3,
    BC_MEDIAN_THRESHOLD = 4
};
```

Values

Name	Description
BC_SMALLEST	Returns the smallest blob contour.
BC_LARGEST	Returns the largest blob contour.
BC_MIN_THRESHOLD	Returns the blob contour for the minimum threshold.
BC_MAX_THRESHOLD	Returns the blob contour for the maximum threshold.
BC_MEDIAN_THRESHOLD	Returns the blob contour for the median threshold.

6.2.1.4.13 ETcVnBoostClassifierType

Offers different types of Boost classifiers.

Syntax

Definition:

```
enum ETcVnBoostClassifierType : LONG
{
    BCT_DISCRETE = 0,
    BCT_REAL     = 1,
    BCT_LOGIT    = 2,
    BCT_GENTLE   = 3
};
```

Values

Name	Description
BCT_DISCRETE	Discrete AdaBoost
BCT_REAL	Real AdaBoost
BCT_LOGIT	LogitBoost
BCT_GENTLE	Gentle AdaBoost

6.2.1.4.14 ETcVnBorderInterpolationMethod

Offers methods to extrapolate values of non-existing pixels. On the one hand, this is used for filtering functions to enable filtering at the image borders (where the filter mask reaches over the border). On the other hand, this is used to extrapolate the undefined pixels after a geometric image transformation.

Syntax

Definition:

```
enum ETcVnBorderInterpolationMethod : LONG
{
    BIM_CONSTANT             = 0,
    BIM_REPLICATE            = 1,
    BIM_REFLECT              = 2,
    BIM_WRAP                 = 3,
    BIM_REFLECT_101          = 4,
    BIM_DEFAULT              = 4,
    BIM_TRANSPARENT          = 5,
    BIM_ISOLATED_CONSTANT    = 16,
    BIM_ISOLATED_REPLICATE   = 17,
    BIM_ISOLATED_REFLECT     = 18,
    BIM_ISOLATED_WRAP        = 19,
    BIM_ISOLATED_REFLECT_101 = 20
};
```

Values

Name	Description
BIM_CONSTANT	iiiiiii | abcdefgh | iiiiiii (with some specified i)
BIM_REPLICATE	aaaaaaa | abcdefgh | hhhhhhh
BIM_REFLECT	gfedcba | abcdefgh | hgfedcb
BIM_WRAP	bcdefgh | abcdefgh | abcdefg
BIM_REFLECT_101	hgfedcb | abcdefgh | gfedcba
BIM_DEFAULT	Choose this if you don't know which method to select (same as REFLECT_101).
BIM_TRANSPARENT	Corresponding pixels in the destination image will not be modified. Only available for geometric image transformations.
BIM_ISOLATED_CONSTANT	Similar to CONSTANT, but ROIs are handled as isolated, which means surrounding image pixels outside the ROI are ignored.
BIM_ISOLATED_REPLICATE	Similar to REPLICATE, but ROIs are handled as isolated, which means surrounding image pixels outside the ROI are ignored.

Name	Description
BIM_ISOLATED_REFLECT	Similar to REFLECT, but ROIs are handled as isolated, which means surrounding image pixels outside the ROI are ignored.
BIM_ISOLATED_WRAP	Similar to WRAP, but ROIs are handled as isolated, which means surrounding image pixels outside the ROI are ignored.
BIM_ISOLATED_REFLECT_101	Similar to REFLECT_101, but ROIs are handled as isolated, which means surrounding image pixels outside the ROI are ignored.

6.2.1.4.15 ETcVnCalibrationPattern

Offers common calibration pattern types.

Syntax

Definition:

```
enum ETcVnCalibrationPattern : LONG
{
    CP_CIRCLES_SYM         = 0,
    CP_CIRCLES_ASYM        = 1,
    CP_CIRCLES_ASYM_INDENT = 2
};
```

Values

Name	Description
CP_CIRCLES_SYM	Symmetric circle pattern
CP_CIRCLES_ASYM	Asymmetric circle pattern, i.e. every 2nd row is indented, starting unindented
CP_CIRCLES_ASYM_INDENT	Asymmetric circle pattern, i.e. every 2nd row is indented, starting indented

6.2.1.4.16 ETcVnCalibrationPatternOrigin

Offers calibration pattern origins.

Syntax

Definition:

```
enum ETcVnCalibrationPatternOrigin : LONG
{
    CPO_TOPLEFT      = 0,
    CPO_TOPCENTER    = 1,
    CPO_TOPRIGHT     = 2,
    CPO_CENTERLEFT   = 3,
    CPO_CENTER       = 4,
    CPO_CENTERRIGHT  = 5,
    CPO_BOTTOMLEFT   = 6,
    CPO_BOTTOMCENTER = 7,
    CPO_BOTTOMRIGHT  = 8
};
```

Values

Name	Description
CPO_TOPLEFT	Origin is at top left
CPO_TOPCENTER	Origin is at top center
CPO_TOPRIGHT	Origin is at top right
CPO_CENTERLEFT	Origin is at center left

Name	Description
CPO_CENTER	Origin is at center
CPO_CENTERRIGHT	Origin is at center right
CPO_BOTTOMLEFT	Origin is at bottom left
CPO_BOTTOMCENTER	Origin is at bottom center
CPO_BOTTOMRIGHT	Origin is at bottom right

6.2.1.4.17 ETcVnCameraState

Specifies the state of a camera controller, which controls an attached camera.

Syntax

Definition:

```
enum ETcVnCameraState : LONG
{
    CS_ERROR              = -1,
    CS_INITIAL            = 0,
    CS_INITIALIZING       = 1,
    CS_INITIALIZED        = 2,
    CS_OPENING            = 3,
    CS_OPENED             = 4,
    CS_STARTACQUISITION   = 5,
    CS_ACQUIRING          = 6,
    CS_STOPACQUISITION    = 7,
    CS_RESETTINGFEATURES  = 8,
    CS_TRIGGERING         = 9,
    CS_CLOSING            = 10
};
```

Values

Name	Description
CS_ERROR	The camera controller is in an error state (e.g. the connection to the camera was lost).
CS_INITIAL	The camera controller is in the initial state and ready to establish the connection to the camera.
CS_INITIALIZING	The camera controller is initializing the camera.
CS_INITIALIZED	The camera controller has initialized the camera.
CS_OPENING	The camera controller is establishing the connection to the camera.
CS_OPENED	The connection to the camera has been established and the camera controller is ready to start the image acquisition.
CS_STARTACQUISITION	The camera controller is starting the image acquisition.
CS_ACQUIRING	The camera is sending images (either streaming or manually triggered, depending on the configuration).
CS_STOPACQUISITION	The camera controller is stopping the image acquisition.
CS_RESETTINGFEATURES	The camera controller is resetting the features on the camera.
CS_TRIGGERING	The camera controller is processing a softwaretrigger.
CS_CLOSING	The camera controller is closing the connection to the camera.

6.2.1.4.18 ETcVnCircularityCalculationMethod

Offers methods to calculate contour circularity

Syntax

Definition:

```
enum ETcVnCircularityCalculationMethod : LONG
{
    CCM_DEFAULT                = 0,
    CCM_CONVEXHULL_PERIMETER   = 1,
    CCM_MAX_RADIUS             = 2,
    CCM_ENCLOSINGCIRCLE_RADIUS = 3
};
```

Values

Name	Description
CCM_DEFAULT	4pi * area / perimeter^2
CCM_CONVEXHULL_PERIMETER	4pi * area / (convex hull perimeter)^2
CCM_MAX_RADIUS	area / (pi * r^2), r = max distance to the contour's center of mass
CCM_ENCLOSINGCIRCLE_RADIUS	area / (pi * r^2), r = radius of the minimum enclosing circle

6.2.1.4.19 ETcVnClusteringAlgorithm

Offers clustering algorithms

Syntax

Definition:

```
enum ETcVnClusteringAlgorithm : LONG
{
    CA_KMEANSPP = 0,
    CA_LBG      = 1
};
```

Values

Name	Description
CA_KMEANSPP	KMeans++ (fixed number of clusters)
CA_LBG	LBG variant (dynamic number of clusters)

6.2.1.4.20 ETcVnCodeDetectionModel

提供经过训练的神经网络模型，用于编码检测。

语法

定义：

```
enum ETcVnCodeDetectionModel : ULONGLONG
{
    CDM_DETECT_1_1 = 0x0001100000000001,
    CDM_DETECT_1_2 = 0x0001100000000002,
    CDM_DETECT_1_3 = 0x0001100000000003,
    CDM_DETECT_2_1 = 0x0001100000010001,
    CDM_DETECT_2_2 = 0x0001100000010002,
    CDM_DETECT_2_3 = 0x0001100000010003,
    CDM_DETECT_3_1 = 0x0001100000020001,
    CDM_DETECT_3_2 = 0x0001100000020002,
    CDM_DETECT_3_3 = 0x0001100000020003
};
```

值

名称	描述
CDM_DETECT_1_1	目标检测模型可检测一维条码和二维码，并将二维码分类为QR码、数据矩阵码和 DotCode（点码） 类型。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及相应的编码类型。输入：224×224 RGB 图像。
CDM_DETECT_1_2	目标检测模型可检测一维条码和二维码，并将二维码分类为QR码、数据矩阵码和 DotCode（点码） 类型。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及相应的编码类型。输入：352×256 RGB 图像。
CDM_DETECT_1_3	目标检测模型可检测一维条码和二维码，并将二维码分类为QR码、数据矩阵码和 DotCode（点码） 类型。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及相应的编码类型。输入：416×352 RGB 图像。
CDM_DETECT_2_1	目标检测模型可检测一维条码。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及 TCVN_CT_1D 编码类型。输入：224×224 RGB 图像。
CDM_DETECT_2_2	目标检测模型可检测一维条码。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及 TCVN_CT_1D 编码类型。输入：352×256 RGB 图像。
CDM_DETECT_2_3	目标检测模型可检测一维条码。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及 TCVN_CT_1D 编码类型。输入：416×352 RGB 图像。
CDM_DETECT_3_1	目标检测模型可检测二维码。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及 TCVN_CT_2D 编码类型。输入：224×224 RGB 图像。
CDM_DETECT_3_2	目标检测模型可检测二维码。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及 TCVN_CT_2D 编码类型。输入：352×256 RGB 图像。
CDM_DETECT_3_3	目标检测模型可检测二维码。它为每次检测提供边界框（ContainerType_Vector_TcVnRectangle_DINT）及 TCVN_CT_2D 编码类型。输入：416×352 RGB 图像。

6.2.1.4.21 ETcVnCodeType

提供要检测的编码类型。

语法

定义：

```
enum ETcVnCodeType : ULONGLONG
{
    CT_ANY        = 0xFFFFFFFFFFFFFFFF,
    CT_1D         = 0x0000FFFFFF000000,
    CT_2D         = 0x00000000000FFFFF,
    CT_QR         = 0x0000000000000001,
    CT_DATAMATRIX = 0x0000000000000002,
    CT_DOTCODE    = 0x0000000000000004
};
```

值

名称	描述
CT_ANY	一维条码和二维码
CT_1D	一维条码，例如 ETcVnBarcodeType 中的类型
CT_2D	二维编码，例如QR码、数据矩阵码和 DotCode （点码）
CT_QR	QR码
CT_DATAMATRIX	数据矩阵码
CT_DOTCODE	DotCode（点码）

6.2.1.4.22 ETcVnColorMap

Offers color maps (similar to GNU Octave/MATLAB types).

Syntax

Definition:

```
enum ETcVnColorMap : LONG
{
    CM_AUTUMN           = 0,
    CM_BONE             = 1,
    CM_JET              = 2,
    CM_WINTER           = 3,
    CM_RAINBOW          = 4,
    CM_OCEAN            = 5,
    CM_SUMMER           = 6,
    CM_SPRING           = 7,
    CM_COOL             = 8,
    CM_HSV              = 9,
    CM_PINK             = 10,
    CM_HOT              = 11,
    CM_PARULA           = 12,
    CM_MAGMA            = 13,
    CM_INFERNO          = 14,
    CM_PLASMA           = 15,
    CM_VIRIDIS          = 16,
    CM_CIVIDIS          = 17,
    CM_TWILIGHT         = 18,
    CM_TWILIGHT_SHIFTED = 19,
    CM_TURBO            = 20
};
```

Values

Name	Description
CM_AUTUMN	Red - orange - yellow
CM_BONE	Black - blueish gray - white
CM_JET	Dark blue - green - dark red
CM_WINTER	Blue - green
CM_RAINBOW	Red - green - purple
CM_OCEAN	Black - blue - white
CM_SUMMER	Green - yellow
CM_SPRING	Pink - yellow
CM_COOL	Cyan - magenta
CM_HSV	Red - green - blue - red

Name	Description
CM_PINK	Black - pink - light yellow - white
CM_HOT	Black - red - yellow - white
CM_PARULA	Blue - yellow
CM_MAGMA	Black - magenta - light yellow
CM_INFERNO	Black - magenta - yellow
CM_PLASMA	Blue - magenta - yellow
CM_VIRIDIS	Purple - yellow
CM_CIVIDIS	Blue - gray - yellow
CM_TWILIGHT	White - light blue - dark purple - light red - white
CM_TWILIGHT_SHIFTED	Dark purple - light blue - white - light red - dark purple
CM_TURBO	Dark blue - light green - dark red

6.2.1.4.23 ETcVnColorMapSize

Offers color map sizes.

Syntax

Definition:

```
enum ETcVnColorMapSize : ULONG
{
    CMS_256   = 256,
    CMS_65536 = 65536
};
```

Values

Name	Description
CMS_256	256 colors, used for 8 bit images (TCVN_ET_USINT)
CMS_65536	65536 colors, used for 16 bit images (TCVN_ET_UINT)

6.2.1.4.24 ETcVnColorSpaceTransform

Offers color space transformations.

语法

定义：

```
enum ETcVnColorSpaceTransform : LONG
{
    CST_BGR_TO_BGRA                  = 0,
    CST_RGB_TO_RGBA                  = 0,
    CST_BGRA_TO_BGR                  = 1,
    CST_RGBA_TO_RGB                  = 1,
    CST_BGR_TO_RGBA                  = 2,
    CST_RGB_TO_BGRA                  = 2,
    CST_BGRA_TO_RGB                  = 3,
    CST_RGBA_TO_BGR                  = 3,
    CST_BGR_TO_RGB                   = 4,
    CST_RGB_TO_BGR                   = 4,
    CST_BGRA_TO_RGBA                 = 5,
    CST_RGBA_TO_BGRA                 = 5,
    CST_BGR_TO_GRAY                  = 6,
    CST_RGB_TO_GRAY                  = 7,
    CST_GRAY_TO_BGR                  = 8,
    CST_GRAY_TO_RGB                  = 8,
    CST_GRAY_TO_BGRA                 = 9,
    CST_GRAY_TO_RGBA                 = 9,
    CST_BGRA_TO_GRAY                 = 10,
    CST_RGBA_TO_GRAY                 = 11,
    CST_RGB_TO_BGR_565               = 12,
```

```
    CST_BGR_TO_BGR_565                = 13,
    CST_BGR_565_TO_RGB                = 14,
    CST_BGR_565_TO_BGR                = 15,
    CST_RGBA_TO_BGR_565               = 16,
    CST_BGRA_TO_BGR_565               = 17,
    CST_BGR_565_TO_RGBA               = 18,
    CST_BGR_565_TO_BGRA               = 19,
    CST_GRAY_TO_BGR_565               = 20,
    CST_BGR_565_TO_GRAY               = 21,
    CST_RGB_TO_BGR_555                = 22,
    CST_BGR_TO_BGR_555                = 23,
    CST_BGR_555_TO_RGB                = 24,
    CST_BGR_555_TO_BGR                = 25,
    CST_RGBA_TO_BGR_555               = 26,
    CST_BGRA_TO_BGR_555               = 27,
    CST_BGR_555_TO_RGBA               = 28,
    CST_BGR_555_TO_BGRA               = 29,
    CST_GRAY_TO_BGR_555               = 30,
    CST_BGR_555_TO_GRAY               = 31,
    CST_BGR_TO_XYZ                    = 32,
    CST_RGB_TO_XYZ                    = 33,
    CST_XYZ_TO_BGR                    = 34,
    CST_XYZ_TO_RGB                    = 35,
    CST_BGR_TO_YCRCB                  = 36,
    CST_RGB_TO_YCRCB                  = 37,
    CST_YCRCB_TO_BGR                  = 38,
    CST_YCRCB_TO_RGB                  = 39,
    CST_BGR_TO_HSV                    = 40,
    CST_RGB_TO_HSV                    = 41,
    CST_BGR_TO_LAB                    = 44,
    CST_RGB_TO_LAB                    = 45,
    CST_BAYER_RG_TO_BGR               = 46,
    CST_BAYER_BG_TO_RGB               = 46,
    CST_BAYER_GR_TO_BGR               = 47,
    CST_BAYER_GB_TO_RGB               = 47,
    CST_BAYER_BG_TO_BGR               = 48,
    CST_BAYER_RG_TO_RGB               = 48,
    CST_BAYER_GB_TO_BGR               = 49,
    CST_BAYER_GR_TO_RGB               = 49,
    CST_BGR_TO_LUV                    = 50,
    CST_RGB_TO_LUV                    = 51,
    CST_BGR_TO_HLS                    = 52,
    CST_RGB_TO_HLS                    = 53,
    CST_HSV_TO_BGR                    = 54,
    CST_HSV_TO_RGB                    = 55,
    CST_LAB_TO_BGR                    = 56,
    CST_LAB_TO_RGB                    = 57,
    CST_LUV_TO_BGR                    = 58,
    CST_LUV_TO_RGB                    = 59,
    CST_HLS_TO_BGR                    = 60,
    CST_HLS_TO_RGB                    = 61,
    CST_BAYER_RG_TO_BGR_VNG           = 62,
    CST_BAYER_GR_TO_BGR_VNG           = 63,
    CST_BAYER_BG_TO_BGR_VNG           = 64,
    CST_BAYER_GB_TO_BGR_VNG           = 65,
    CST_BAYER_RG_TO_RGB_VNG           = 64,
    CST_BAYER_GR_TO_RGB_VNG           = 65,
    CST_BAYER_BG_TO_RGB_VNG           = 62,
    CST_BAYER_GB_TO_RGB_VNG           = 63,
    CST_BGR_TO_HSV_FULL               = 66,
    CST_RGB_TO_HSV_FULL               = 67,
    CST_BGR_TO_HLS_FULL               = 68,
    CST_RGB_TO_HLS_FULL               = 69,
    CST_HSV_TO_BGR_FULL               = 70,
    CST_HSV_TO_RGB_FULL               = 71,
    CST_HLS_TO_BGR_FULL               = 72,
    CST_HLS_TO_RGB_FULL               = 73,
    CST_LBGR_TO_LAB                   = 74,
    CST_LRGB_TO_LAB                   = 75,
    CST_LBGR_TO_LUV                   = 76,
    CST_LRGB_TO_LUV                   = 77,
    CST_LAB_TO_LBGR                   = 78,
    CST_LAB_TO_LRGB                   = 79,
    CST_LUV_TO_LBGR                   = 80,
    CST_LUV_TO_LRGB                   = 81,
    CST_BGR_TO_YUV                    = 82,
    CST_RGB_TO_YUV                    = 83,
    CST_YUV_TO_BGR                    = 84,
    CST_YUV_TO_RGB                    = 85,
    CST_BAYER_RG_TO_GRAY              = 86,
    CST_BAYER_GR_TO_GRAY              = 87,
    CST_BAYER_BG_TO_GRAY              = 88,
    CST_BAYER_GB_TO_GRAY              = 89,
    CST_YUV_420_NV12_TO_RGB           = 90,
    CST_YUV_420_NV12_TO_BGR           = 91,
    CST_YUV_420_NV21_TO_RGB           = 92,
    CST_YUV_420_NV21_TO_BGR           = 93,
    CST_YUV_420_SP_TO_RGB             = 92,
    CST_YUV_420_SP_TO_BGR             = 93,
    CST_YUV_420_NV12_TO_RGBA          = 94,
    CST_YUV_420_NV12_TO_BGRA          = 95,
    CST_YUV_420_NV21_TO_RGBA          = 96,
    CST_YUV_420_NV21_TO_BGRA          = 97,
    CST_YUV_420_SP_TO_RGBA            = 96,
```

```
    CST_YUV_420_SP_TO_BGRA            = 97,
    CST_YUV_420_YV12_TO_RGB           = 98,
    CST_YUV_420_YV12_TO_BGR           = 99,
    CST_YUV_420_IYUV_TO_RGB           = 100,
    CST_YUV_420_IYUV_TO_BGR           = 101,
    CST_YUV_420_I420_TO_RGB           = 100,
    CST_YUV_420_I420_TO_BGR           = 101,
    CST_YUV_420_P_TO_RGB              = 98,
    CST_YUV_420_P_TO_BGR              = 99,
    CST_YUV_420_YV12_TO_RGBA          = 102,
    CST_YUV_420_YV12_TO_BGRA          = 103,
    CST_YUV_420_IYUV_TO_RGBA          = 104,
    CST_YUV_420_IYUV_TO_BGRA          = 105,
    CST_YUV_420_I420_TO_RGBA          = 104,
    CST_YUV_420_I420_TO_BGRA          = 105,
    CST_YUV_420_P_TO_RGBA             = 102,
    CST_YUV_420_P_TO_BGRA             = 103,
    CST_YUV_420_TO_GRAY               = 106,
    CST_YUV_420_NV21_TO_GRAY          = 106,
    CST_YUV_420_NV12_TO_GRAY          = 106,
    CST_YUV_420_YV12_TO_GRAY          = 106,
    CST_YUV_420_IYUV_TO_GRAY          = 106,
    CST_YUV_420_I420_TO_GRAY          = 106,
    CST_YUV_420_SP_TO_GRAY            = 106,
    CST_YUV_420_P_TO_GRAY             = 106,
    CST_YUV_422_UYVY_TO_RGB           = 107,
    CST_YUV_422_UYVY_TO_BGR           = 108,
    CST_YUV_422_Y422_TO_RGB           = 107,
    CST_YUV_422_Y422_TO_BGR           = 108,
    CST_YUV_422_UYNV_TO_RGB           = 107,
    CST_YUV_422_UYNV_TO_BGR           = 108,
    CST_YUV_422_UYVY_TO_RGBA          = 111,
    CST_YUV_422_UYVY_TO_BGRA          = 112,
    CST_YUV_422_Y422_TO_RGBA          = 111,
    CST_YUV_422_Y422_TO_BGRA          = 112,
    CST_YUV_422_UYNV_TO_RGBA          = 111,
    CST_YUV_422_UYNV_TO_BGRA          = 112,
    CST_YUV_422_YUY2_TO_RGB           = 115,
    CST_YUV_422_YUY2_TO_BGR           = 116,
    CST_YUV_422_YVYU_TO_RGB           = 117,
    CST_YUV_422_YVYU_TO_BGR           = 118,
    CST_YUV_422_YUYV_TO_RGB           = 115,
    CST_YUV_422_YUYV_TO_BGR           = 116,
    CST_YUV_422_YUNV_TO_RGB           = 115,
    CST_YUV_422_YUNV_TO_BGR           = 116,
    CST_YUV_422_YUY2_TO_RGBA          = 119,
    CST_YUV_422_YUY2_TO_BGRA          = 120,
    CST_YUV_422_YVYU_TO_RGBA          = 121,
    CST_YUV_422_YVYU_TO_BGRA          = 122,
    CST_YUV_422_YUYV_TO_RGBA          = 119,
    CST_YUV_422_YUYV_TO_BGRA          = 120,
    CST_YUV_422_YUNV_TO_RGBA          = 119,
    CST_YUV_422_YUNV_TO_BGRA          = 120,
    CST_YUV_422_UYVY_TO_GRAY          = 123,
    CST_YUV_422_YUY2_TO_GRAY          = 124,
    CST_YUV_422_Y422_TO_GRAY          = 123,
    CST_YUV_422_UYNV_TO_GRAY          = 123,
    CST_YUV_422_YVYU_TO_GRAY          = 124,
    CST_YUV_422_YUYV_TO_GRAY          = 124,
    CST_YUV_422_YUNV_TO_GRAY          = 124,
    CST_RGBA_TO_PREMULTIPLICATED_RGBA = 125,
    CST_PREMULTIPLICATED_RGBA_TO_RGBA = 126,
    CST_RGB_TO_YUV_420_I420           = 127,
    CST_BGR_TO_YUV_420_I420           = 128,
    CST_RGB_TO_YUV_420_IYUV           = 127,
    CST_BGR_TO_YUV_420_IYUV           = 128,
    CST_RGBA_TO_YUV_420_I420          = 129,
    CST_BGRA_TO_YUV_420_I420          = 130,
    CST_RGBA_TO_YUV_420_IYUV          = 129,
    CST_BGRA_TO_YUV_420_IYUV          = 130,
    CST_RGB_TO_YUV_420_YV12           = 131,
    CST_BGR_TO_YUV_420_YV12           = 132,
    CST_RGBA_TO_YUV_420_YV12          = 133,
    CST_BGRA_TO_YUV_420_YV12          = 134,
    CST_BAYER_RG_TO_BGR_EA            = 135,
    CST_BAYER_GR_TO_BGR_EA            = 136,
    CST_BAYER_BG_TO_BGR_EA            = 137,
    CST_BAYER_GB_TO_BGR_EA            = 138,
    CST_BAYER_RG_TO_RGB_EA            = 137,
    CST_BAYER_GR_TO_RGB_EA            = 138,
    CST_BAYER_BG_TO_RGB_EA            = 135,
    CST_BAYER_GB_TO_RGB_EA            = 136,
    CST_BAYER_RG_TO_BGRA              = 139,
    CST_BAYER_GR_TO_BGRA              = 140,
    CST_BAYER_BG_TO_BGRA              = 141,
    CST_BAYER_GB_TO_BGRA              = 142,
    CST_BAYER_RG_TO_RGBA              = 141,
    CST_BAYER_GR_TO_RGBA              = 142,
    CST_BAYER_BG_TO_RGBA              = 139,
    CST_BAYER_GB_TO_RGBA              = 140,
    CST_RGB_TO_YUV_422_UYVY           = 143,
    CST_BGR_TO_YUV_422_UYVY           = 144,
    CST_RGB_TO_YUV_422_Y422           = 143,
    CST_BGR_TO_YUV_422_Y422           = 144,
```

```
    CST_RGB_TO_YUV_422_UYNV          = 143,
    CST_BGR_TO_YUV_422_UYNV          = 144,
    CST_RGBA_TO_YUV_422_UYVY         = 145,
    CST_BGRA_TO_YUV_422_UYVY         = 146,
    CST_RGBA_TO_YUV_422_Y422         = 145,
    CST_BGRA_TO_YUV_422_Y422         = 146,
    CST_RGBA_TO_YUV_422_UYNV         = 145,
    CST_BGRA_TO_YUV_422_UYNV         = 146,
    CST_RGB_TO_YUV_422_YUY2          = 147,
    CST_BGR_TO_YUV_422_YUY2          = 148,
    CST_RGB_TO_YUV_422_YVYU          = 149,
    CST_BGR_TO_YUV_422_YVYU          = 150,
    CST_RGB_TO_YUV_422_YUYV          = 147,
    CST_BGR_TO_YUV_422_YUYV          = 148,
    CST_RGB_TO_YUV_422_YUNV          = 147,
    CST_BGR_TO_YUV_422_YUNV          = 148,
    CST_RGBA_TO_YUV_422_YUY2         = 151,
    CST_BGRA_TO_YUV_422_YUY2         = 152,
    CST_RGBA_TO_YUV_422_YVYU         = 153,
    CST_BGRA_TO_YUV_422_YVYU         = 154,
    CST_RGBA_TO_YUV_422_YUYV         = 151,
    CST_BGRA_TO_YUV_422_YUYV         = 152,
    CST_RGBA_TO_YUV_422_YUNV         = 151,
    CST_BGRA_TO_YUV_422_YUNV         = 152,
    CST_MAX                          = 155
};
```

值

名称	描述
CST_BGR_TO_BGRA	将颜色空间从 BGR 转换为 BGRA。
CST_RGB_TO_RGBA	将颜色空间从 RGB 转换为 RGBA。
CST_BGRA_TO_BGR	将颜色空间从 BGRA 转换为 BGR。
CST_RGBA_TO_RGB	将颜色空间从 RGBA 转换为 RGB。
CST_BGR_TO_RGBA	将颜色空间从 BGR 转换为 RGBA。
CST_RGB_TO_BGRA	将颜色空间从 RGB 转换为 BGRA。
CST_BGRA_TO_RGB	将颜色空间从 BGRA 转换为 RGB。
CST_RGBA_TO_BGR	将颜色空间从 RGBA 转换为 BGR。
CST_BGR_TO_RGB	将颜色空间从 BGR 转换为 RGB。
CST_RGB_TO_BGR	将颜色空间从 RGB 转换为 BGR。
CST_BGRA_TO_RGBA	将颜色空间从 BGRA 转换为 RGBA。
CST_RGBA_TO_BGRA	将颜色空间从 RGBA 转换为 BGRA。
CST_BGR_TO_GRAY	将颜色空间从 BGR 转换为灰色。
CST_RGB_TO_GRAY	将颜色空间从 RGB 转换为灰色。
CST_GRAY_TO_BGR	将颜色空间从灰色转换为 BGR。
CST_GRAY_TO_RGB	将颜色空间从灰色转换为 RGB。
CST_GRAY_TO_BGRA	将颜色空间从灰色转换为 BGRA。
CST_GRAY_TO_RGBA	将颜色空间从灰色转换为 RGBA。
CST_BGRA_TO_GRAY	将颜色空间从 BGRA 转换为灰色。
CST_RGBA_TO_GRAY	将颜色空间从 RGBA 转换为灰色。
CST_RGB_TO_BGR_565	将颜色空间从 RGB（24 位 888）转换为 BGR（16 位 565）。
CST_BGR_TO_BGR_565	将颜色空间从 BGR（24 位 888）转换为 BGR（16 位 565）。
CST_BGR_565_TO_RGB	将颜色空间从 BGR（16 位 565）转换为 RGB（24 位 888）。
CST_BGR_565_TO_BGR	将颜色空间从 BGR（16 位 565）转换为 BGR（24 位 888）。
CST_RGBA_TO_BGR_565	将颜色空间从 RGBA（32 位 8888）转换为 BGR（16 位 565）。
CST_BGRA_TO_BGR_565	将颜色空间从 BGRA（32 位 8888）转换为 BGR（16 位 565）。

名称	描述
CST_BGR_565_TO_RGBA	将颜色空间从 BGR（16 位 565）转换为 RGBA（32 位 8888）。
CST_BGR_565_TO_BGRA	将颜色空间从 BGR（16 位 565）转换为 BGRA（32 位 8888）。
CST_GRAY_TO_BGR_565	将颜色空间从灰色转换为 BGR（16 位 565）。
CST_BGR_565_TO_GRAY	将颜色空间从 BGR（16 位 565）转换为灰色。
CST_RGB_TO_BGR_555	将颜色空间从 RGB（24 位 888）转换为 BGR（16 位 555）。
CST_BGR_TO_BGR_555	将颜色空间从 BGR（24 位 888）转换为 BGR（16 位 555）。
CST_BGR_555_TO_RGB	将颜色空间从 BGR（16 位 555）转换为 RGB（24 位 888）。
CST_BGR_555_TO_BGR	将颜色空间从 BGR（16 位 555）转换为 BGR（24 位 888）。
CST_RGBA_TO_BGR_555	将颜色空间从 RGBA（32 位 8888）转换为 BGR（16 位 555）。
CST_BGRA_TO_BGR_555	将颜色空间从 BGRA（32 位 8888）转换为 BGR（16 位 555）。
CST_BGR_555_TO_RGBA	将颜色空间从 BGR（16 位 555）转换为 RGBA（32 位 8888）。
CST_BGR_555_TO_BGRA	将颜色空间从 BGR（16 位 555）转换为 BGRA（32 位 8888）。
CST_GRAY_TO_BGR_555	将颜色空间从灰色转换为 BGR（16 位 555）。
CST_BGR_555_TO_GRAY	将颜色空间从 BGR（16 位 555）转换为灰色。
CST_BGR_TO_XYZ	将颜色空间从 BGR 转换为 CIE XYZ（缩放至图像的完整数值范围）。
CST_RGB_TO_XYZ	将颜色空间从 RGB 转换为 CIE XYZ（缩放至图像的完整数值范围）。
CST_XYZ_TO_BGR	将颜色空间从 CIE XYZ（缩放至图像的完整数值范围）转换为 BGR。
CST_XYZ_TO_RGB	将颜色空间从 CIE XYZ（缩放至图像的完整数值范围）转换为 RGB。
CST_BGR_TO_YCRCB	将颜色空间从 BGR 转换为 YCrCb（缩放至图像的完整数值范围）。
CST_RGB_TO_YCRCB	将颜色空间从 RGB 转换为 YCrCb（缩放至图像的完整数值范围）。
CST_YCRCB_TO_BGR	将颜色空间从 YCrCb（缩放至图像的完整数值范围）转换为 BGR。
CST_YCRCB_TO_RGB	将颜色空间从 YCrCb（缩放至图像的完整数值范围）转换为 RGB。
CST_BGR_TO_HSV	将颜色空间从 BGR 转换为 HSV（支持 USINT 或 REAL 类型的图像；对于 USINT 类型，H 的缩放范围为 0 至 180，S 和 V 的缩放范围为 0 至 255）。
CST_RGB_TO_HSV	将颜色空间从 RGB 转换为 HSV（支持 USINT 或 REAL 类型的图像；对于 USINT 类型，H 的缩放范围为 0 至 180，S 和 V 的缩放范围为 0 至 255）。
CST_BGR_TO_LAB	将颜色空间从 BGR 转换为 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
CST_RGB_TO_LAB	将颜色空间从 RGB 转换为 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。

名称	描述
CST_BAYER_RG_TO_BGR	将颜色空间从 BayerRG 转换为 BGR（支持 USINT 或 UINT 类型的图像）。
CST_BAYER_BG_TO_RGB	将颜色空间从 BayerBG 转换为 RGB（支持 USINT 或 UINT 类型的图像）。
CST_BAYER_GR_TO_BGR	将颜色空间从 BayerGR 转换为 BGR（支持 USINT 或 UINT 类型的图像）。
CST_BAYER_GB_TO_RGB	将颜色空间从 BayerGB 转换为 RGB（支持 USINT 或 UINT 类型的图像）。
CST_BAYER_BG_TO_BGR	将颜色空间从 BayerBG 转换为 BGR（支持 USINT 或 UINT 类型的图像）。
CST_BAYER_RG_TO_RGB	将颜色空间从 BayerRG 转换为 RGB（支持 USINT 或 UINT 类型的图像）。
CST_BAYER_GB_TO_BGR	将颜色空间从 BayerGB 转换为 BGR（支持 USINT 或 UINT 类型的图像）。
CST_BAYER_GR_TO_RGB	将颜色空间从 BayerGR 转换为 RGB（支持 USINT 或 UINT 类型的图像）。
CST_BGR_TO_LUV	将颜色空间从 BGR 转换为 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
CST_RGB_TO_LUV	将颜色空间从 RGB 转换为 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
CST_BGR_TO_HLS	将颜色空间从 BGR 转换为 HLS（对于 USINT 类型的图像，H 的缩放范围为 0 至 180，L 和 S 的缩放范围为 0 至 255）。
CST_RGB_TO_HLS	将颜色空间从 RGB 转换为 HLS（对于 USINT 类型的图像，H 的缩放范围为 0 至 180，L 和 S 的缩放范围为 0 至 255）。
CST_HSV_TO_BGR	将颜色空间从 HSV（对于 USINT 类型的图像，H 的缩放范围为 0 至 180，S 和 V 的缩放范围为 0 至 255）转换为 BGR。
CST_HSV_TO_RGB	将颜色空间从 HSV（对于 USINT 类型的图像，H 的缩放范围为 0 至 180，S 和 V 的缩放范围为 0 至 255）转换为 RGB。
CST_LAB_TO_BGR	将颜色空间从 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 BGR。
CST_LAB_TO_RGB	将颜色空间从 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 RGB。
CST_LUV_TO_BGR	将颜色空间从 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 BGR。
CST_LUV_TO_RGB	将颜色空间从 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 RGB。
CST_HLS_TO_BGR	将颜色空间从 HLS（对于 USINT 类型的图像，H 的范围为 0 至 180，L 和 S 的范围为 0 至 255）转换为 BGR。
CST_HLS_TO_RGB	将颜色空间从 HLS（对于 USINT 类型的图像，H 的范围为 0 至 180，L 和 S 的范围为 0 至 255）转换为 RGB。
CST_BAYER_RG_TO_BGR_VNG	使用可变梯度数将颜色空间从 BayerRG 转换为 BGR（支持 USINT 类型的图像）。
CST_BAYER_GR_TO_BGR_VNG	使用可变梯度数将颜色空间从 BayerGR 转换为 BGR（支持 USINT 类型的图像）。
CST_BAYER_BG_TO_BGR_VNG	使用可变梯度数将颜色空间从 BayerBG 转换为 BGR（支持 USINT 类型的图像）。

名称	描述
CST_BAYER_GB_TO_BGR_VNG	使用可变梯度数将颜色空间从 BayerGB 转换为 BGR（支持 USINT 类型的图像）。
CST_BAYER_RG_TO_RGB_VNG	使用可变梯度数将颜色空间从 BayerRG 转换为 RGB（支持 USINT 类型的图像）。
CST_BAYER_GR_TO_RGB_VNG	使用可变梯度数将颜色空间从 BayerGR 转换为 RGB（支持 USINT 类型的图像）。
CST_BAYER_BG_TO_RGB_VNG	使用可变梯度数将颜色空间从 BayerBG 转换为 RGB（支持 USINT 类型的图像）。
CST_BAYER_GB_TO_RGB_VNG	使用可变梯度数将颜色空间从 BayerGB 转换为 RGB（支持 USINT 类型的图像）。
CST_BGR_TO_HSV_FULL	将颜色空间从 BGR 转换为 HSV（支持 USINT 或 REAL 类型的图像；对于 USINT 类型，所有通道的缩放范围为 0 至 255）。
CST_RGB_TO_HSV_FULL	将颜色空间从 RGB 转换为 HSV（支持 USINT 或 REAL 类型的图像；对于 USINT 类型，所有通道的缩放范围为 0 至 255）。
CST_BGR_TO_HLS_FULL	将颜色空间从 BGR 转换为 HLS（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
CST_RGB_TO_HLS_FULL	将颜色空间从 RGB 转换为 HLS（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
CST_HSV_TO_BGR_FULL	将颜色空间从 HSV（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 BGR。
CST_HSV_TO_RGB_FULL	将颜色空间从 HSV（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 RGB。
CST_HLS_TO_BGR_FULL	将颜色空间从 HLS（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 BGR。
CST_HLS_TO_RGB_FULL	将颜色空间从 HLS（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 RGB。
CST_LBGR_TO_LAB	将颜色空间从 LBGR 转换为 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
CST_LRGB_TO_LAB	将颜色空间从 LRGB 转换为 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
CST_LBGR_TO_LUV	将颜色空间从 LBGR 转换为 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
CST_LRGB_TO_LUV	将颜色空间从 LRGB 转换为 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）。
CST_LAB_TO_LBGR	将颜色空间从 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 LBGR。
CST_LAB_TO_LRGB	将颜色空间从 CIE L*a*b*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 LRGB。
CST_LUV_TO_LBGR	将颜色空间从 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 LBGR。
CST_LUV_TO_LRGB	将颜色空间从 CIE L*u*v*（对于 USINT 类型的图像，所有通道的缩放范围为 0 至 255）转换为 LRGB。
CST_BGR_TO_YUV	将颜色空间从 BGR 转换为 YUV（缩放至图像的完整数值范围）。
CST_RGB_TO_YUV	将颜色空间从 RGB 转换为 YUV（缩放至图像的完整数值范围）。
CST_YUV_TO_BGR	将颜色空间从 YUV（缩放至图像的完整数值范围）转换为 BGR。

名称	描述
CST_YUV_TO_RGB	将颜色空间从 YUV（缩放至图像的完整数值范围）转换为 RGB。
CST_BAYER_RG_TO_GRAY	将颜色空间从 BayerRG 转换为灰度（支持 USINT 或 UINT 类型的图像）。
CST_BAYER_GR_TO_GRAY	将颜色空间从 BayerGR 转换为灰度（支持 USINT 或 UINT 类型的图像）。
CST_BAYER_BG_TO_GRAY	将颜色空间从 BayerBG 转换为灰度（支持 USINT 或 UINT 类型的图像）。
CST_BAYER_GB_TO_GRAY	将颜色空间从 BayerGB 转换为灰度（支持 USINT 或 UINT 类型的图像）。
CST_YUV_420_NV12_TO_RGB	将颜色空间从 YUV420 NV12 转换为 RGB。
CST_YUV_420_NV12_TO_BGR	将颜色空间从 YUV420 NV12 转换为 BGR。
CST_YUV_420_NV21_TO_RGB	将颜色空间从 YUV420 NV21（SP）转换为 RGB。
CST_YUV_420_NV21_TO_BGR	将颜色空间从 YUV420 NV21（SP）转换为 BGR。
CST_YUV_420_SP_TO_RGB	将颜色空间从 YUV420 NV21（SP）转换为 RGB。
CST_YUV_420_SP_TO_BGR	将颜色空间从 YUV420 NV21（SP）转换为 BGR。
CST_YUV_420_NV12_TO_RGBA	将颜色空间从 YUV420 NV12 转换为 RGBA。
CST_YUV_420_NV12_TO_BGRA	将颜色空间从 YUV420 NV12 转换为 BGRA。
CST_YUV_420_NV21_TO_RGBA	将颜色空间从 YUV420 NV21（SP）转换为 RGBA。
CST_YUV_420_NV21_TO_BGRA	将颜色空间从 YUV420 NV21（SP）转换为 BGRA。
CST_YUV_420_SP_TO_RGBA	将颜色空间从 YUV420 NV21（SP）转换为 RGBA。
CST_YUV_420_SP_TO_BGRA	将颜色空间从 YUV420 NV21（SP）转换为 BGRA。
CST_YUV_420_YV12_TO_RGB	将颜色空间从 YUV420 YV12（P）转换为 RGB。
CST_YUV_420_YV12_TO_BGR	将颜色空间从 YUV420 YV12（P）转换为 BGR。
CST_YUV_420_IYUV_TO_RGB	将颜色空间从 YUV420 IYUV（I420）转换为 RGB。
CST_YUV_420_IYUV_TO_BGR	将颜色空间从 YUV420 IYUV（I420）转换为 BGR。
CST_YUV_420_I420_TO_RGB	将颜色空间从 YUV420 IYUV（I420）转换为 RGB。
CST_YUV_420_I420_TO_BGR	将颜色空间从 YUV420 IYUV（I420）转换为 BGR。
CST_YUV_420_P_TO_RGB	将颜色空间从 YUV420 YV12（P）转换为 RGB。
CST_YUV_420_P_TO_BGR	将颜色空间从 YUV420 YV12（P）转换为 BGR。
CST_YUV_420_YV12_TO_RGBA	将颜色空间从 YUV420 YV12（P）转换为 RGBA。
CST_YUV_420_YV12_TO_BGRA	将颜色空间从 YUV420 YV12（P）转换为 BGRA。
CST_YUV_420_IYUV_TO_RGBA	将颜色空间从 YUV420 IYUV（I420）转换为 RGBA。
CST_YUV_420_IYUV_TO_BGRA	将颜色空间从 YUV420 IYUV（I420）转换为 BGRA。
CST_YUV_420_I420_TO_RGBA	将颜色空间从 YUV420 IYUV（I420）转换为 RGBA。
CST_YUV_420_I420_TO_BGRA	将颜色空间从 YUV420 IYUV（I420）转换为 BGRA。
CST_YUV_420_P_TO_RGBA	将颜色空间从 YUV420 YV12（P）转换为 RGBA。
CST_YUV_420_P_TO_BGRA	将颜色空间从 YUV420 YV12（P）转换为 BGRA。
CST_YUV_420_TO_GRAY	将颜色空间从 YUV420 转换为灰色。
CST_YUV_420_NV21_TO_GRAY	将颜色空间从 YUV420 转换为灰色。
CST_YUV_420_NV12_TO_GRAY	将颜色空间从 YUV420 转换为灰色。
CST_YUV_420_YV12_TO_GRAY	将颜色空间从 YUV420 转换为灰色。
CST_YUV_420_IYUV_TO_GRAY	将颜色空间从 YUV420 转换为灰色。
CST_YUV_420_I420_TO_GRAY	将颜色空间从 YUV420 转换为灰色。
CST_YUV_420_SP_TO_GRAY	将颜色空间从 YUV420 转换为灰色。
CST_YUV_420_P_TO_GRAY	将颜色空间从 YUV420 转换为灰色。

名称	描述
CST_YUV_422_UYVY_TO_RGB	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 RGB。
CST_YUV_422_UYVY_TO_BGR	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 BGR。
CST_YUV_422_Y422_TO_RGB	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 RGB。
CST_YUV_422_Y422_TO_BGR	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 BGR。
CST_YUV_422_UYNV_TO_RGB	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 RGB。
CST_YUV_422_UYNV_TO_BGR	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 BGR。
CST_YUV_422_UYVY_TO_RGBA	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 RGBA。
CST_YUV_422_UYVY_TO_BGRA	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 BGRA。
CST_YUV_422_Y422_TO_RGBA	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 RGB。
CST_YUV_422_Y422_TO_BGRA	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 BGRA。
CST_YUV_422_UYNV_TO_RGBA	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 RGB。
CST_YUV_422_UYNV_TO_BGRA	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为 BGRA。
CST_YUV_422_YUY2_TO_RGB	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 RGB。
CST_YUV_422_YUY2_TO_BGR	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 BGR。
CST_YUV_422_YVYU_TO_RGB	将颜色空间从 YUV422 YVYU 转换为 RGB。
CST_YUV_422_YVYU_TO_BGR	将颜色空间从 YUV422 YVYU 转换为 BGR。
CST_YUV_422_YUYV_TO_RGB	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 RGB。
CST_YUV_422_YUYV_TO_BGR	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 BGR。
CST_YUV_422_YUNV_TO_RGB	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 RGB。
CST_YUV_422_YUNV_TO_BGR	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 BGR。
CST_YUV_422_YUY2_TO_RGBA	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 RGBA。
CST_YUV_422_YUY2_TO_BGRA	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 BGRA。
CST_YUV_422_YVYU_TO_RGBA	将颜色空间从 YUV422 YVYU 转换为 RGBA。
CST_YUV_422_YVYU_TO_BGRA	将颜色空间从 YUV422 YVYU 转换为 BGRA。
CST_YUV_422_YUYV_TO_RGBA	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 RGBA。
CST_YUV_422_YUYV_TO_BGRA	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 BGRA。
CST_YUV_422_YUNV_TO_RGBA	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 RGBA。

名称	描述
CST_YUV_422_YUNV_TO_BGRA	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为 BGRA。
CST_YUV_422_UYVY_TO_GRAY	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为灰色。
CST_YUV_422_YUY2_TO_GRAY	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为灰色。
CST_YUV_422_Y422_TO_GRAY	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为灰色。
CST_YUV_422_UYNV_TO_GRAY	将颜色空间从 YUV422 UYVY（Y422、UYNV）转换为灰色。
CST_YUV_422_YVYU_TO_GRAY	将颜色空间从 YUV422 YVYU 转换为灰色。
CST_YUV_422_YUYV_TO_GRAY	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为灰色。
CST_YUV_422_YUNV_TO_GRAY	将颜色空间从 YUV422 YUYV（YUY2、YUNV）转换为灰色。
CST_RGBA_TO_PREMULTIPLICATED_RGBA	将颜色空间从 RGBA 转换为预乘 RGBA。
CST_PREMULTIPLICATED_RGBA_TO_RGBA	将颜色空间从预乘 RGBA 转换为 RGBA。
CST_RGB_TO_YUV_420_I420	将颜色空间从 RGB 转换为 YUV420 IYUV（I420）。
CST_BGR_TO_YUV_420_I420	将颜色空间从 BGR 转换为 YUV420 IYUV（I420）。
CST_RGB_TO_YUV_420_IYUV	将颜色空间从 RGB 转换为 YUV420 IYUV（I420）。
CST_BGR_TO_YUV_420_IYUV	将颜色空间从 BGR 转换为 YUV420 IYUV（I420）。
CST_RGBA_TO_YUV_420_I420	将颜色空间从 RGBA 转换为 YUV420 IYUV（I420）。
CST_BGRA_TO_YUV_420_I420	将颜色空间从 BGRA 转换为 YUV420 IYUV（I420）。
CST_RGBA_TO_YUV_420_IYUV	将颜色空间从 RGBA 转换为 YUV420 IYUV（I420）。
CST_BGRA_TO_YUV_420_IYUV	将颜色空间从 BGRA 转换为 YUV420 IYUV（I420）。
CST_RGB_TO_YUV_420_YV12	将颜色空间从 RGB 转换为 YUV420 YV12（P）。
CST_BGR_TO_YUV_420_YV12	将颜色空间从 BGR 转换为 YUV420 YV12（P）。
CST_RGBA_TO_YUV_420_YV12	将颜色空间从 RGBA 转换为 YUV420 YV12（P）。
CST_BGRA_TO_YUV_420_YV12	将颜色空间从 BGRA 转换为 YUV420 YV12（P）。
CST_BAYER_RG_TO_BGR_EA	使用边缘感知算法，将颜色空间从 BayerRG 转换为 BGR。
CST_BAYER_GR_TO_BGR_EA	使用边缘感知算法，将颜色空间从 BayerGR 转换为 BGR。
CST_BAYER_BG_TO_BGR_EA	使用边缘感知算法，将颜色空间从 BayerBG 转换为 BGR。
CST_BAYER_GB_TO_BGR_EA	使用边缘感知算法，将颜色空间从 BayerGB 转换为 BGR。
CST_BAYER_RG_TO_RGB_EA	使用边缘感知算法，将颜色空间从 BayerRG 转换为 RGB。
CST_BAYER_GR_TO_RGB_EA	使用边缘感知算法，将颜色空间从 BayerGR 转换为 RGB。
CST_BAYER_BG_TO_RGB_EA	使用边缘感知算法，将颜色空间从 BayerBG 转换为 RGB。
CST_BAYER_GB_TO_RGB_EA	使用边缘感知算法，将颜色空间从 BayerGB 转换为 RGB。
CST_BAYER_RG_TO_BGRA	将颜色空间从 BayerRG 转换为 BGRA。
CST_BAYER_GR_TO_BGRA	将颜色空间从 BayerGR 转换为 BGRA。
CST_BAYER_BG_TO_BGRA	将颜色空间从 BayerBG 转换为 BGRA。
CST_BAYER_GB_TO_BGRA	将颜色空间从 BayerGB 转换为 BGRA。

名称	描述
CST_BAYER_RG_TO_RGBA	将颜色空间从 BayerRG 转换为 RGBA。
CST_BAYER_GR_TO_RGBA	将颜色空间从 BayerGR 转换为 RGBA。
CST_BAYER_BG_TO_RGBA	将颜色空间从 BayerBG 转换为 RGBA。
CST_BAYER_GB_TO_RGBA	将颜色空间从 BayerGB 转换为 RGBA。
CST_RGB_TO_YUV_422_UYVY	将颜色空间从 RGB 转换为 YUV422 UYVY（Y422、UYNV）。
CST_BGR_TO_YUV_422_UYVY	将颜色空间从 BGR 转换为 YUV422 UYVY（Y422、UYNV）。
CST_RGB_TO_YUV_422_Y422	将颜色空间从 RGB 转换为 YUV422 UYVY（Y422、UYNV）。
CST_BGR_TO_YUV_422_Y422	将颜色空间从 BGR 转换为 YUV422 UYVY（Y422、UYNV）。
CST_RGB_TO_YUV_422_UYNV	将颜色空间从 RGB 转换为 YUV422 UYVY（Y422、UYNV）。
CST_BGR_TO_YUV_422_UYNV	将颜色空间从 BGR 转换为 YUV422 UYVY（Y422、UYNV）。
CST_RGBA_TO_YUV_422_UYVY	将颜色空间从 RGBA 转换为 YUV422 UYVY（Y422、UYNV）。
CST_BGRA_TO_YUV_422_UYVY	将颜色空间从 BGRA 转换为 YUV422 UYVY（Y422、UYNV）。
CST_RGBA_TO_YUV_422_Y422	将颜色空间从 RGBA 转换为 YUV422 UYVY（Y422、UYNV）。
CST_BGRA_TO_YUV_422_Y422	将颜色空间从 BGRA 转换为 YUV422 UYVY（Y422、UYNV）。
CST_RGBA_TO_YUV_422_UYNV	将颜色空间从 RGBA 转换为 YUV422 UYVY（Y422、UYNV）。
CST_BGRA_TO_YUV_422_UYNV	将颜色空间从 BGRA 转换为 YUV422 UYVY（Y422、UYNV）。
CST_RGB_TO_YUV_422_YUY2	将颜色空间从 RGB 转换为 YUV422 YUY2（YUYV、YUNV）。
CST_BGR_TO_YUV_422_YUY2	将颜色空间从 BGR 转换为 YUV422 YUY2（YUYV、UYNV）。
CST_RGB_TO_YUV_422_YVYU	将颜色空间从 RGB 转换为 YUV422 YVYU。
CST_BGR_TO_YUV_422_YVYU	将颜色空间从 BGR 转换为 YUV422 YVYU。
CST_RGB_TO_YUV_422_YUYV	将颜色空间从 RGB 转换为 YUV422 YUY2（YUYV、YUNV）。
CST_BGR_TO_YUV_422_YUYV	将颜色空间从 BGR 转换为 YUV422 YUY2（YUYV、UYNV）。
CST_RGB_TO_YUV_422_YUNV	将颜色空间从 RGB 转换为 YUV422 YUY2（YUYV、YUNV）。
CST_BGR_TO_YUV_422_YUNV	将颜色空间从 BGR 转换为 YUV422 YUY2（YUYV、UYNV）。
CST_RGBA_TO_YUV_422_YUY2	将颜色空间从 RGBA 转换为 YUV422 YUY2（YUYV、YUNV）。
CST_BGRA_TO_YUV_422_YUY2	将颜色空间从 BGRA 转换为 YUV422 YUY2（YUYV、UYNV）。
CST_RGBA_TO_YUV_422_YVYU	将颜色空间从 RGBA 转换为 YUV422 YVYU。
CST_BGRA_TO_YUV_422_YVYU	将颜色空间从 BGRA 转换为 YUV422 YVYU。
CST_RGBA_TO_YUV_422_YUYV	将颜色空间从 RGBA 转换为 YUV422 YUY2（YUYV、YUNV）。

名称	描述
CST_BGRA_TO_YUV_422_YUYV	将颜色空间从 BGRA 转换为 YUV422 YUY2（YUYV、UYNV）。
CST_RGBA_TO_YUV_422_YUNV	将颜色空间从 RGBA 转换为 YUV422 YUY2（YUYV、YUNV）。
CST_BGRA_TO_YUV_422_YUNV	将颜色空间从 BGRA 转换为 YUV422 YUY2（YUYV、UYNV）。
CST_MAX	仅供内部使用，在添加新值时进行调整。

6.2.1.4.25 ETcVnColorTrainingMethod

Offers color training methods.

Syntax

Definition:

```
enum ETcVnColorTrainingMethod : LONG
{
    CTM_LAB = 0,
    CTM_RGB = 1
};
```

Values

Name	Description
CTM_LAB	Trains the color in CIE L*a*b* color space.
CTM_RGB	Trains the color in RGB color space.

6.2.1.4.26 ETcVnComparisonOperator

Offers comparison operators for the SortContainer function.

Values

Name	Description
CO_LESS	Sort values ascending
CO_GREATER	Sort values descending

6.2.1.4.27 ETcVnConnectedComponentsAlgorithm

Offers connected components algorithms.

Syntax

Definition:

```
enum ETcVnConnectedComponentsAlgorithm : LONG
{
    CCA_WU    = 0,
    CCA_GRANA = 1
};
```

Values

Name	Description
CCA_WU	SAUF algorithm
CCA_GRANA	BBDT algorithm for 8-way connectivity, SAUF algorithm for 4-way connectivity

6.2.1.4.28 ETcVnContainerExportFormat

Offers container export formats.

Syntax

Definition:

```
enum ETcVnContainerExportFormat : USHORT
{
    CEF_XML            = 0,
    CEF_XML_SERIALIZED = 1,
    CEF_CSV            = 2,
    CEF_BINARY         = 3
};
```

Values

Name	Description
CEF_XML	XML format with human readable data
CEF_XML_SERIALIZED	XML format with serialized data
CEF_CSV	CSV format (limited to two dimensional representation, so not applicable for all container types)
CEF_BINARY	Binary serialized data (recommended for large containers)

6.2.1.4.29 ETcVnContourApproximationMethod

Offers methods for contour approximation.

Syntax

Definition:

```
enum ETcVnContourApproximationMethod : LONG
{
    CAM_NONE      = 1,
    CAM_SIMPLE    = 2,
    CAM_TC89_L1   = 3,
    CAM_TC89_KCOS = 4
};
```

Values

Name	Description
CAM_NONE	No approximation, every single point is stored, i.e. two subsequent points are always direct neighbors (horizontal, vertical or diagonal).
CAM_SIMPLE	Lossless compression of segments that form straight lines in horizontal, vertical or diagonal direction leaving only their endpoints (For instance, an up-right rectangle is reduced to its four corner points.)
CAM_TC89_L1	Compression using a Teh-Chin chain approximation algorithm (IEEE-paper C. Teh, R. Chin, On the Detection of Dominant Points on Digital Curves, 1989).
CAM_TC89_KCOS	Compression using a Teh-Chin chain approximation algorithm (IEEE-paper C. Teh, R. Chin, On the Detection of Dominant Points on Digital Curves, 1989).

6.2.1.4.30 ETcVnContourRetrievalMode

Offers retrieval modes for a contour search.

Syntax

Definition:

```
enum ETcVnContourRetrievalMode : LONG
{
    CRM_EXTERNAL             = 0,
    CRM_LIST                 = 1,
    CRM_CONNECTED_COMPONENTS = 2,
    CRM_TREE                 = 3,
    CRM_FLOODFILL            = 4
};
```

Values

Name	Description
CRM_EXTERNAL	Returns only external contours.
CRM_LIST	Returns all found contours, not considering their hierarchy.
CRM_CONNECTED_COMPONENTS	Returns all contours with a 2-level hierarchy (External contours are assigned level 0, internal contours level 1).
CRM_TREE	Returns all contours and their full hierarchy.
CRM_FLOODFILL	Returns the found contours using a floodfill algorithm (only available for DINT images).

6.2.1.4.31 ETcVnContoursMatchComparisonMethod

Offers comparison methods for contour matching.

Syntax

Definition:

```
enum ETcVnContoursMatchComparisonMethod : LONG
{
    CMCM_CONTOURS_MATCH_I1 = 1,
    CMCM_CONTOURS_MATCH_I2 = 2,
    CMCM_CONTOURS_MATCH_I3 = 3
};
```

Values

Name	Description
CMCM_CONTOURS_MATCH_I1	Contours are compared using the sum over the differences between the reciprocal individual characteristics.
CMCM_CONTOURS_MATCH_I2	Contours are compared using the sum over the differences between the individual characteristics.
CMCM_CONTOURS_MATCH_I3	Contours are compared using only the maximum difference between the individual characteristics.

6.2.1.4.32 ETcVnDataLayout

Offers image data layouts.

Syntax

Definition:

```
enum ETcVnDataLayout : ULONGLONG
{
    DL_DEFAULT = 0x0,
    DL_2D_HW   = 0x12,
    DL_2D_WH   = 0x21,
    DL_3D_CHW  = 0x123,
    DL_3D_HWC  = 0x231,
    DL_4D_NCHW = 0x1234,
    DL_4D_NHWC = 0x1342
};
```

Values

Name	Description
DL_DEFAULT	Default
DL_2D_HW	2D data HW, multi/single channel image
DL_2D_WH	2D data WH, multi/single channel image
DL_3D_CHW	3D data CHW, single channel image
DL_3D_HWC	3D data HWC, single channel image
DL_4D_NCHW	4D data NCHW, single channel image
DL_4D_NHWC	4D data NHWC, single channel image

6.2.1.4.33 ETcVnDataMatrixOptions

Offers Data Matrix code detection options.

Syntax

Definition:

```
enum ETcVnDataMatrixOptions : ULONGLONG
{
    DMO_DEFAULT          = 0,
    DMO_BINARY_INPUT     = 1,
    DMO_GLOBAL_THRESHOLD = 2,
    DMO_BINARY_GRID      = 4,
    DMO_STATIC_GRID      = 8,
    DMO_REGULAR_COLOR    = 16,
    DMO_INVERTED_COLOR   = 32
};
```

Values

Name	Description
DMO_DEFAULT	Expects a graylevel image as input. Applies an adaptive local threshold and creates a dynamic grid based on the clock pattern.
DMO_BINARY_INPUT	Expects a binary image as input (all pixels must contain 0 or 255, disables internal preprocessing)
DMO_GLOBAL_THRESHOLD	Applies a global instead of an adaptive local threshold (requires uniform illumination)
DMO_BINARY_GRID	Create the dynamic grid based on the binary instead of the graylevel image
DMO_STATIC_GRID	Use an equally spaced static grid instead of a dynamic one
DMO_REGULAR_COLOR	Only search for regular color (dark code on light background)
DMO_INVERTED_COLOR	Only search for inverted color (light code on dark background)

6.2.1.4.34 ETcVnDataMatrixOptions

Offers Data Matrix code detection options.

Syntax

Definition:

```
enum ETcVnDataMatrixOptions : ULONGLONG
{
    DMO_DEFAULT          = 0,
    DMO_BINARY_INPUT     = 1,
    DMO_GLOBAL_THRESHOLD = 2,
    DMO_BINARY_GRID      = 4,
    DMO_STATIC_GRID      = 8,
    DMO_REGULAR_COLOR    = 16,
    DMO_INVERTED_COLOR   = 32
};
```

Values

Name	Description
DMO_DEFAULT	Expects a graylevel image as input. Applies an adaptive local threshold and creates a dynamic grid based on the clock pattern.
DMO_BINARY_INPUT	Expects a binary image as input (all pixels must contain 0 or 255, disables internal preprocessing)
DMO_GLOBAL_THRESHOLD	Applies a global instead of an adaptive local threshold (requires uniform illumination)
DMO_BINARY_GRID	Create the dynamic grid based on the binary instead of the graylevel image
DMO_STATIC_GRID	Use an equally spaced static grid instead of a dynamic one
DMO_REGULAR_COLOR	Only search for regular color (dark code on light background)
DMO_INVERTED_COLOR	Only search for inverted color (light code on dark background)

6.2.1.4.35 ETcVnDetectPatternPointsOptions

提供 DetectPatternPoints2 的选项。

语法

定义：

```
enum ETcVnDetectPatternPointsOptions : ULONG
{
    DPPO_SYMMETRIC  = 1,
    DPPO_ASYMMETRIC = 2,
    DPPO_CLUSTERING = 4
};
```

值

名称	描述
DPPO_SYMMETRIC	对称网格
DPPO_ASYMMETRIC	非对称网格
DPPO_CLUSTERING	使用聚类算法进行网格检测（对透视几何畸变的适应性较强，但对背景杂乱的适应性较弱）

6.2.1.4.36 ETcVnDiffusivityTypeKAZE

Offers diffusivity types for feature detection methods KAZE and AKAZE.

Syntax

Definition:

```
enum ETcVnDiffusivityTypeKAZE : LONG
{
    DT1_KAZE_PM_G1       = 0,
    DT1_KAZE_PM_G2       = 1,
    DT1_KAZE_WEICKERT    = 2,
    DT1_KAZE_CHARBONNIER = 3
};
```

Values

Name	Description
DT1_KAZE_PM_G1	Uses the Perona-Malik diffusivity filter g1.
DT1_KAZE_PM_G2	Uses the Perona-Malik diffusivity filter g2.
DT1_KAZE_WEICKERT	Uses the Weickert diffusivity filter.
DT1_KAZE_CHARBONNIER	Uses the Charbonnier diffusivity filter.

6.2.1.4.37 ETcVnDistanceTransformationLabel

Offers types of the label array to build.

Syntax

Definition:

```
enum ETcVnDistanceTransformationLabel : LONG
{
    DTL_CCOMP = 0,
    DTL_PIXEL = 1
};
```

Values

Name	Description
DTL_CCOMP	Labels connected components (Each connected component of zeros in the source image and the pixels closest to the connected component will be assigned the same label).
DTL_PIXEL	Labels pixels (Each zero pixel and the non-zero pixels closest to it get their own label).

6.2.1.4.38 ETcVnDistanceTransformationMask

Offers different sizes of the distance transformation mask. Please note, that some sizes are not supported by certain distance types.

Syntax

Definition:

```
enum ETcVnDistanceTransformationMask : LONG
{
    DTM_PRECISE = 0,
    DTM_3       = 3,
    DTM_5       = 5
};
```

Values

Name	Description
DTM_PRECISE	Uses a precise mask.
DTM_3	Uses a 3x3 mask.
DTM_5	Uses a 5x5 mask.

6.2.1.4.39 ETcVnDistanceType

Offers distance types.

Syntax

Definition:

```
enum ETcVnDistanceType : LONG
{
    DT_USER   = -1,
    DT_L1     = 1,
    DT_L2     = 2,
    DT_C      = 3,
    DT_L12    = 4,
    DT_FAIR   = 5,
    DT_WELSCH = 6,
    DT_HUBER  = 7
};
```

Values

Name	Description
DT_USER	User defined distance
DT_L1	|x1 - x2| + |y1 - y2|
DT_L2	Euclidean distance
DT_C	max(|x1 - x2|, |y1 - y2|)
DT_L12	L1 - L2 : 2*(sqrt(1 + x*x/2) - 1))
DT_FAIR	c^2(|x|/c-log(1+|x|/c)), c = 1.3998
DT_WELSCH	c^2/2(1-exp(-(x/c)^2)), c = 2.9846
DT_HUBER	|x|<c ? x^2/2 : c(|x|-c/2), c=1.345

6.2.1.4.40 ETcVnDotCodeOptions

提供 DotCode （点码）检测选项。

语法

定义：

```
enum ETcVnDotCodeOptions : ULONGLONG
{
    DCO_DEFAULT         = 0,
    DCO_BINARY_INPUT    = 1,
    DCO_DETECTBLOBS     = 2,
    DCO_FILTERCONTOURS  = 4,
    DCO_REGULAR_COLOR   = 16,
    DCO_INVERTED_COLOR  = 32
};
```

值

名称	描述
DCO_DEFAULT	以灰度图像作为输入，并应用默认算法（类似于参考解码算法）

名称	描述
DCO_BINARY_INPUT	预期将二进制图像作为输入（所有像素必须包含 0 或 255，禁用内部预处理）
DCO_DETECTBLOBS	通过使用Blob检测来查找点，从而修改默认算法（无法与 BINARY_INPUT 或 FILTERCONTOURS 组合使用）
DCO_FILTERCONTOURS	作为预处理步骤，移除与图像边界相连以及尺寸明显错误（相对于所提供的 fModuleWidth 而言）的轮廓
DCO_REGULAR_COLOR	仅搜索常规颜色（浅色背景上的深色编码）
DCO_INVERTED_COLOR	仅搜索反色（深色背景上的浅色编码）

6.2.1.4.41 ETcVnDrawMatchesFlags

Offers a combination of flags to support overdrawing an existing destination image and/or skipping single (i.e. non-matched) keypoints and/or drawing additional (rich-)keypoint information (size and orientation). Used by F_VN_DrawKeypointsExp.

Syntax

Definition:

```
enum ETcVnDrawMatchesFlags : LONG
{
    DMF_DEFAULT                          = 0,
    DMF_OVERDRAW                         = 1,
    DMF_SKIPSINGLE                       = 2,
    DMF_OVERDRAW_SKIPSINGLE              = 3,
    DMF_RICHKEYPOINT                     = 4,
    DMF_OVERDRAW_RICHKEYPOINT            = 5,
    DMF_SKIPSINGLE_RICHKEYPOINT          = 6,
    DMF_OVERDRAW_SKIPSINGLE_RICHKEYPOINT = 7
};
```

Values

Name	Description
DMF_DEFAULT	Draw all keypoints into a new image.
DMF_OVERDRAW	Draw all keypoints into the existing destination image.
DMF_SKIPSINGLE	Draw the keypoints into a new image but skip single keypoints.
DMF_OVERDRAW_SKIPSINGLE	Draw the keypoints into the existing destination image but skip single keypoints.
DMF_RICHKEYPOINT	Draw all keypoints with rich information into a new image.
DMF_OVERDRAW_RICHKEYPOINT	Draw all keypoints with rich information into the existing destination image.
DMF_SKIPSINGLE_RICHKEYPOINT	Draw the keypoints with rich information into a new image but skip single keypoints.
DMF_OVERDRAW_SKIPSINGLE_RICHKEYPOINT	Draw the keypoints with rich information into the existing destination image but skip single keypoints.

6.2.1.4.42 ETcVnDrawShape

Offers shapes to be drawn.

Syntax

Definition:

```
enum ETcVnDrawShape : LONG
{
    DS_RANDOM  = -1,
    DS_CIRCLE  = 0,
    DS_SQUARE  = 1,
    DS_PLUS    = 2,
    DS_X       = 3,
    DS_DIAMOND = 4,
    DS_MAX     = 5
};
```

Values

Name	Description
DS_RANDOM	Randomly select a circle, square, plus, X or diamond.
DS_CIRCLE	Circle
DS_SQUARE	Square
DS_PLUS	+
DS_X	x
DS_DIAMOND	Diamond
DS_MAX	For internal use only, adapted when adding new values.

6.2.1.4.43 ETcVnEdgeDetectionAlgorithm

Offers edge detection algorithms.

Syntax

Definition:

```
enum ETcVnEdgeDetectionAlgorithm : LONG
{
    EDA_INTERPOLATION   = 0,
    EDA_APPROX_ERF      = 1,
    EDA_APPROX_GAUSSIAN = 2
};
```

Values

Name	Description
EDA_INTERPOLATION	Interpolates pixels (bilinear) and then finds the maximum gradient. This approach is fast and stable, but usually less precise than the function approximation methods.
EDA_APPROX_ERF	Approximates the edge with an erf function. This approach is slower than the interpolation method, but usually more precise. However, it can be inaccurate if the edge does not suit the erf model.
EDA_APPROX_GAUSSIAN	Approximates the edge with a gaussian function. This method is intended to find the center of relatively thin lines, so it is likely to be inaccurate for other edges.

6.2.1.4.44 ETcVnEdgeDirection

Offers edge directions relative to the search direction.

Syntax

Definition:

```
enum ETcVnEdgeDirection : LONG
{
    ED_DARK_TO_LIGHT = 0,
    ED_LIGHT_TO_DARK = 1
};
```

Values

Name	Description
ED_DARK_TO_LIGHT	Dark to light
ED_LIGHT_TO_DARK	Light to dark

6.2.1.4.45 ETcVnElementType

Offers element types.

Syntax

Definition:

```
enum ETcVnElementType : LONG
{
    ET_SAME_AS_SOURCE = -1,
    ET_USINT          = 0,
    ET_SINT           = 1,
    ET_UINT           = 2,
    ET_INT            = 3,
    ET_DINT           = 4,
    ET_REAL           = 5,
    ET_LREAL          = 6
};
```

Values

Name	Description
ET_SAME_AS_SOURCE	Sets the element type of the destination image to the source image element type.
ET_USINT	USINT (depth: 8 bit, 0..255)
ET_SINT	SINT (depth: 8 bit, -128..127)
ET_UINT	UINT (depth: 16 bit, 0..65535)
ET_INT	INT (depth: 16 bit, -32768..32767)
ET_DINT	DINT (depth: 32 bit, -2147483648..2147483647)
ET_REAL	REAL (depth: 32 bit, ~ -3.402823E 10^38 .. ~ 3.402823E 38)
ET_LREAL	LREAL (depth: 64 bit, ~ -1.79769313486231E 308 .. ~ 1.79769313486232E 308)

6.2.1.4.46 ETcVnEstimationAlgorithm

Offers estimation algorithms for matching point sets.

Syntax

Definition:

```
enum ETcVnEstimationAlgorithm : LONG
{
    EA_DEFAULT = 0,
    EA_LMEDS   = 4,
    EA_RANSAC  = 8,
    EA_RHO     = 16
};
```

Values

Name	Description
EA_DEFAULT	Use all points.
EA_LMEDS	Least Median of Squares
EA_RANSAC	Random Sample Consensus
EA_RHO	Progressive Sample Consensus

6.2.1.4.47 ETcVnExtremePointDirection

Offers search directions for the extreme point.

Syntax

Definition:

```
enum ETcVnExtremePointDirection : LONG
{
    EPD_TOP_LEFT      = 0,
    EPD_TOP_MEDIAN    = 1,
    EPD_TOP_RIGHT     = 2,
    EPD_BOTTOM_LEFT   = 3,
    EPD_BOTTOM_MEDIAN = 4,
    EPD_BOTTOM_RIGHT  = 5,
    EPD_LEFT_TOP      = 6,
    EPD_LEFT_MEDIAN   = 7,
    EPD_LEFT_BOTTOM   = 8,
    EPD_RIGHT_TOP     = 9,
    EPD_RIGHT_MEDIAN  = 10,
    EPD_RIGHT_BOTTOM  = 11
};
```

Values

Name	Description
EPD_TOP_LEFT	Find the topmost point (min y, take left one if more than 1).
EPD_TOP_MEDIAN	Find the topmost point (min y, take median one if more than 1).
EPD_TOP_RIGHT	Find the topmost point (min y, take right one if more than 1).
EPD_BOTTOM_LEFT	Find the bottommost point (max y, take left one if more than 1).
EPD_BOTTOM_MEDIAN	Find the bottommost point (max y, take median one if more than 1).
EPD_BOTTOM_RIGHT	Find the bottommost point (max y, take right one if more than 1).
EPD_LEFT_TOP	Find the leftmost point (min x, take top one if more than 1).
EPD_LEFT_MEDIAN	Find the leftmost point (min x, take median one if more than 1).
EPD_LEFT_BOTTOM	Find the leftmost point (min x, take bottom one if more than 1).
EPD_RIGHT_TOP	Find the rightmost point (max x, take top one if more than 1).
EPD_RIGHT_MEDIAN	Find the rightmost point (max x, take median one if more than 1).
EPD_RIGHT_BOTTOM	Find the rightmost point (max x, take bottom one if more than 1).

6.2.1.4.48 ETcVnFeatureDescriptorTypeAKAZE

Offers descriptor types for AKAZE method.

Syntax

Definition:

```
enum ETcVnFeatureDescriptorTypeAKAZE : LONG
{
    FDT_AKAZE_KAZE_UPRIGHT = 2,
    FDT_AKAZE_KAZE         = 3,
    FDT_AKAZE_MLDB_UPRIGHT = 4,
    FDT_AKAZE_MLDB         = 5
};
```

Values

Name	Description
FDT_AKAZE_KAZE_UPRIGHT	KAZE descriptor (upright, i.e. rotation dependend).
FDT_AKAZE_KAZE	KAZE descriptor (rotation invariant).
FDT_AKAZE_MLDB_UPRIGHT	MLDB descriptor (upright, i.e. rotation dependend).
FDT_AKAZE_MLDB	MLDB descriptor (rotation invariant).

6.2.1.4.49 ETcVnFeatureScalingType

Offers feature scaling types.

Syntax

Definition:

```
enum ETcVnFeatureScalingType : LONG
{
    FST1_MAXABS          = 0,
    FST1_MINMAX          = 1,
    FST1_STANDARDIZATION = 2
};
```

Values

Name	Description
FST1_MAXABS	Use the max of the absolute values of the feature vector. The normalized values are in the range [-1 to 1] or [0 to 1] if all the original values are positive.
FST1_MINMAX	Use the min and max values of the feature vector. The normalized values are in the range [0 to 1]
FST1_STANDARDIZATION	Use the mean and the standard deviation of the feature vector. The normalized vector has a mean value equal to zero and standard deviation equal to one

6.2.1.4.50 ETcVnFeatureScoreTypeORB

Offers algorithms used to rank features.

Syntax

Definition:

```
enum ETcVnFeatureScoreTypeORB : LONG
{
    FST_ORB_HARRIS = 0,
    FST_ORB_FAST   = 1
};
```

Values

Name	Description
FST_ORB_HARRIS	Harris algorithm (produces more stable keypoints, but computation takes longer).
FST_ORB_FAST	FAST algorithm (produces slightly less stable keypoints, but computation is faster).

6.2.1.4.51 ETcVnFilterDirection

Offers directions, in which to apply filter.

Syntax

Definition:

```
enum ETcVnFilterDirection : SHORT
{
    FD_X = 0,
    FD_Y = 1
};
```

Values

Name	Description
FD_X	X-direction
FD_Y	Y-direction

6.2.1.4.52 ETcVnFindTemplateWalshOptions

提供标志，用于自定义基于Walsh变换的模板搜索。

语法

定义：

```
enum ETcVnFindTemplateWalshOptions : ULONG
{
    FTWO_NONE                 = 0,
    FTWO_SKIP_DC              = 2,
    FTWO_WD_STOP_CORRECTION   = 256,
    FTWO_FUSE_MATCHES         = 512,
    FTWO_ROTATED_RECTANGLE    = 1024,
    FTWO_MATCH_OPPOSITE_RANGE = 65536
};
```

值

名称	描述
FTWO_NONE	不使用任何特殊选项。
FTWO_SKIP_DC	在相似度计算期间跳过 DC 部分。DC 或“直流”分量是Walshd第一个系数（全为+1的方波序列）。与其他分量相比，将其用于投影会捕获总像素强度，而非空间差异。因此，跳过它可提高对光源变化的适应性。
FTWO_WD_STOP_CORRECTION	使用试错式方法，在 Watchdog（看门狗）停止时调整匹配阈值
FTWO_FUSE_MATCHES	融合不同旋转角度且位置相近的匹配项。

名称	描述
FTWO_ROTATED_RECTANGLE	以 ContainerType_Vector_TcVnRotatedRectangle 类型的容器返回结果
FTWO_MATCH_OPPOSITE_RANGE	使用给定的旋转角度以及与之相反的旋转角度进行匹配（仅可与基于范围的函数配合使用）

6.2.1.4.53 ETcVnFlipAxis

Defines the axis around which to flip (mirror) the image.

Syntax

Definition:

```
enum ETcVnFlipAxis : SHORT
{
    FA_XY = -1,
    FA_X  = 0,
    FA_Y  = 1
};
```

Values

Name	Description
FA_XY	Flip the image around both axes
FA_X	Flip the image around the X axis
FA_Y	Flip the image around the Y axis

6.2.1.4.54 ETcVnFontType

Offers font types.

Syntax

Definition:

```
enum ETcVnFontType : LONG
{
    FT_HERSHEY_SIMPLEX              = 0,
    FT_HERSHEY_PLAIN                = 1,
    FT_HERSHEY_DUPLEX               = 2,
    FT_HERSHEY_COMPLEX              = 3,
    FT_HERSHEY_TRIPLEX              = 4,
    FT_HERSHEY_COMPLEX_SMALL        = 5,
    FT_HERSHEY_SCRIPT_SIMPLEX       = 6,
    FT_HERSHEY_SCRIPT_COMPLEX       = 7,
    FT_HERSHEY_PLAIN_ITALIC         = 17,
    FT_HERSHEY_COMPLEX_ITALIC       = 19,
    FT_HERSHEY_TRIPLEX_ITALIC       = 20,
    FT_HERSHEY_COMPLEX_SMALL_ITALIC = 21
};
```

Values

Name	Description
FT_HERSHEY_SIMPLEX	Normal size sans-serif font.
FT_HERSHEY_PLAIN	Small size sans-serif font.
FT_HERSHEY_DUPLEX	More complex normal size sans-serif font.
FT_HERSHEY_COMPLEX	Normal size serif font.
FT_HERSHEY_TRIPLEX	More complex normal size serif font.
FT_HERSHEY_COMPLEX_SMALL	Small size serif font.
FT_HERSHEY_SCRIPT_SIMPLEX	Hand-writing style font.
FT_HERSHEY_SCRIPT_COMPLEX	More complex hand-writing style font.

Name	Description
FT_HERSHEY_PLAIN_ITALIC	Small size sans-serif font (italic).
FT_HERSHEY_COMPLEX_ITALIC	Normal size serif font (italic).
FT_HERSHEY_TRIPLEX_ITALIC	More complex normal size serif font (italic).
FT_HERSHEY_COMPLEX_SMALL_ITALIC	Small size serif font (italic).

6.2.1.4.55 ETcVnHoughMethod

Offers Hough methods.

Syntax

Definition:

```
enum ETcVnHoughMethod : LONG
{
    HM_STANDARD      = 0,
    HM_PROBABILISTIC = 1,
    HM_MULTI_SCALE   = 2,
    HM_GRADIENT      = 3,
    HM_GRADIENT_ALT  = 4
};
```

Values

Name	Description
HM_STANDARD	Classical or standard Hough transform. Every line is represented by two floating-point numbers (f1, f2), where f1 is a distance between (0,0) origin and the line, and f2 is the angle between x-axis and the normal to the line.
HM_PROBABILISTIC	Probabilistic Hough transform. More efficient for pictures containing long linear segments. Returns line segments rather than the whole line, while each segment is represented by start and end point.
HM_MULTI_SCALE	Multi scale variant of classical Hough transform.
HM_GRADIENT	Method designed to find circles. See paper: HK Yuen, John Princen, John Illingworth, and Josef Kittler. Comparative study of hough transform methods for circle finding. Image and Vision Computing, 8(1):71–77, 1990
HM_GRADIENT_ALT	Variation of the GRADIENT method.

6.2.1.4.56 ETcVnInitializableFunction

Offers initializable functions.

语法

定义：

```
enum ETcVnInitializableFunction : LONG
{
    IF_OCR           = 0,
    IF_CODEDETECTION = 1
};
```

值

名称	描述
IF_OCR	F_VN_OCR
IF_CODEDETECTION	F_VN_DetectCodesNeuralNetwork

6.2.1.4.57 ETcVnInterpolationType

Offers interpolation types.

语法

定义：

```
enum ETcVnInterpolationType : LONG
{
    IT_NEAREST_NEIGHBOR       = 0,
    IT_BILINEAR               = 1,
    IT_BICUBIC                = 2,
    IT_AREA_BASED             = 3,
    IT_LANCZOS4               = 4,
    IT_BILINEAR_EXACT         = 5,
    IT_NEAREST_NEIGHBOR_EXACT = 6
};
```

值

名称	描述
IT_NEAREST_NEIGHBOR	最近邻插值
IT_BILINEAR	双线性插值
IT_BICUBIC	双三次插值
IT_AREA_BASED	基于面积的插值
IT_LANCZOS4	Lanczos4 插值
IT_BILINEAR_EXACT	位精确双线性插值
IT_NEAREST_NEIGHBOR_EXACT	位精确最近邻插值

6.2.1.4.58 ETcVnKeypointDetectionTypeAGAST

Offers different neighborhood types for AGAST method (For details see paper: E. Mair et al.: Adaptive and Generic Corner Detection Based on the Accelerated Segment Test, 2010).

Syntax

Definition:

```
enum ETcVnKeypointDetectionTypeAGAST : LONG
{
    KDT_AGAST_5_8   = 0,
    KDT_AGAST_7_12d = 1,
    KDT_AGAST_7_12s = 2,
    KDT_AGAST_9_16  = 3
};
```

Values

Name	Description
KDT_AGAST_5_8	5 of 8
KDT_AGAST_7_12d	7 of 12 diamond
KDT_AGAST_7_12s	7 of 12 square
KDT_AGAST_9_16	9 of 16

6.2.1.4.59 ETcVnKeypointDetectionTypeFAST

Offers different neighborhood types for FAST method (For details see paper: E. Rosten: Machine Learning for High-speed Corner Detection, 2006).

Syntax

Definition:

```
enum ETcVnKeypointDetectionTypeFAST : LONG
{
    KDT_FAST_5_8  = 0,
    KDT_FAST_7_12 = 1,
    KDT_FAST_9_16 = 2
};
```

Values

Name	Description
KDT_FAST_5_8	5 of 8
KDT_FAST_7_12	7 of 12
KDT_FAST_9_16	9 of 16

6.2.1.4.60 ETcVnKnn

Offers kNN model types.

Syntax

Definition:

```
enum ETcVnKnn : ULONG
{
    KNN_CLASSIFIER       = 1,
    KNN_NOVELTY_DETECTOR = 2,
    KNN_REGRESSOR        = 4
};
```

Values

Name	Description
KNN_CLASSIFIER	Classifier
KNN_NOVELTY_DETECTOR	Novelty detector
KNN_REGRESSOR	Regressor

6.2.1.4.61 ETcVnLayerNameType

Offer different types of neural network model layer names

Syntax

Definition:

```
enum ETcVnLayerNameType : LONG
{
    LNT_ALL     = 0,
    LNT_OUTPUTS = 1,
    LNT_INPUTS  = 2
};
```

Values

Name	Description
LNT_ALL	Return the name of all layers

Name	Description
LNT_OUTPUTS	Return the names of unconnected output layers
LNT_INPUTS	Return the names of model inputs

6.2.1.4.62 ETcVnLineType

Offers line types.

Syntax

Definition:

```
enum ETcVnLineType : LONG
{
    LT_4_CONNECTED = 4,
    LT_8_CONNECTED = 8,
    LT_ANTIALIASED = 16
};
```

Values

Name	Description
LT_4_CONNECTED	4-connected (pixels are connected horizontally and vertically).
LT_8_CONNECTED	8-connected (pixels are connected horizontally, vertically and diagonally).
LT_ANTIALIASED	Antialiased (drawn using Gaussian filtering, only implemented for 8-bit images).

6.2.1.4.63 ETcVnMatchTemplateWalshOptions

提供标志，用于自定义基于Walsh变换的模板匹配。

语法

定义：

```
enum ETcVnMatchTemplateWalshOptions : ULONG
{
    MTWO_NONE    = 0,
    MTWO_NORMED  = 1,
    MTWO_SKIP_DC = 2
};
```

值

名称	描述
MTWO_NONE	不使用任何特殊选项。
MTWO_NORMED	将输出图像的元素归一化到 [0，1.0] 区间内。
MTWO_SKIP_DC	在相似度计算期间跳过 DC 部分。DC 或“直流”分量是Walshd第一个系数（全为+1的方波序列）。与其他分量相比，将其用于投影会捕获总像素强度，而非空间差异。因此，跳过它可提高对光源变化的适应性。

6.2.1.4.64 ETcVnMorphologicalOperator

Offers morphological operators.

Syntax

Definition:

```
enum ETcVnMorphologicalOperator : LONG
{
    MO_EROSION                          = 0x0,
    MO_DILATION                         = 0x1,
    MO_OPENING                          = 0x2,
    MO_CLOSING                          = 0x3,
    MO_GRADIENT                         = 0x4,
    MO_WHITE_TOPHAT                     = 0x5,
    MO_BLACK_TOPHAT                     = 0x6,
    MO_OPENING_BY_RECONSTRUCTION        = 0x40000002,
    MO_CLOSING_BY_RECONSTRUCTION        = 0x40000003,
    MO_WHITE_TOPHAT_BY_RECONSTRUCTION = 0x40000005,
    MO_BLACK_TOPHAT_BY_RECONSTRUCTION = 0x40000006
};
```

Values

Name	Description
MO_EROSION	Shrinks objects (regions of foreground (i.e. white) pixels), removing regions smaller than the structuring element completely.
MO_DILATION	Expands objects and thereby closes small holes inside objects.
MO_OPENING	Applies an erosion first, a dilation second. Objects smaller than the structuring element are removed while outer shapes remain largely the same.
MO_CLOSING	Applies a dilation first, an erosion second. Holes inside objects that fit into the structuring element are closed completely while outer shapes remain largely the same.
MO_GRADIENT	Difference between the dilation and the erosion of an image.
MO_WHITE_TOPHAT	Difference between an input image and its opening.
MO_BLACK_TOPHAT	Difference between an input image and its closing.
MO_OPENING_BY_RECONSTRUCTION	Opening with subsequent reconstruction of objects that were not removed by the opening.
MO_CLOSING_BY_RECONSTRUCTION	Closing with subsequent reconstruction of objects that were not removed by the closing.
MO_WHITE_TOPHAT_BY_RECONSTRUCTION	White tophat with subsequent reconstruction of objects that were not removed by the white tophat.
MO_BLACK_TOPHAT_BY_RECONSTRUCTION	Black tophat with subsequent reconstruction of objects that were not removed by the black tophat.

6.2.1.4.65 ETcVnNbc

Offers normal Bayes classifier types.

Syntax

Definition:

```
enum ETcVnNbc : ULONG
{
    NBC_CLASSIFIER       = 1,
    NBC_NOVELTY_DETECTOR = 2
};
```

Values

Name	Description
NBC_CLASSIFIER	Classifier
NBC_NOVELTY_DETECTOR	Novelty detector

6.2.1.4.66 ETcVnNormalizationType

Offers normalization types.

Syntax

Definition:

```
enum ETcVnNormalizationType : LONG
{
    NT_INF               = 1,
    NT_L1                = 2,
    NT_L2                = 4,
    NT_L2SQR             = 5,
    NT_HAMMING           = 6,
    NT_HAMMING2          = 7,
    NT_RELATIVE_INF      = 9,
    NT_RELATIVE_L1       = 10,
    NT_RELATIVE_L2       = 12,
    NT_RELATIVE_L2SQR    = 13,
    NT_RELATIVE_HAMMING  = 14,
    NT_RELATIVE_HAMMING2 = 15,
    NT_MINMAX            = 32
};
```

Values

Name	Description
NT_INF	Infinity norm
NT_L1	L1 norm
NT_L2	L2 (euclidean) norm
NT_L2SQR	Squared L2 norm
NT_HAMMING	Hamming distance (bitwise)
NT_HAMMING2	Hamming distance (each 2 bits combined to one)
NT_RELATIVE_INF	Infinity norm (when comparing 2 arrays, the norm of their difference is divided by the norm of the second array)
NT_RELATIVE_L1	L1 norm (when comparing 2 arrays, the norm of their difference is divided by the norm of the second array)
NT_RELATIVE_L2	L2 (euclidean) norm (when comparing 2 arrays, the norm of their difference is divided by the norm of the second array)
NT_RELATIVE_L2SQR	Squared L2 norm (when comparing 2 arrays, the norm of their difference is divided by the norm of the second array)
NT_RELATIVE_HAMMING	Hamming distance (bitwise; when comparing 2 arrays, the norm of their difference is divided by the norm of the second array)
NT_RELATIVE_HAMMING2	Hamming distance (each 2 bits combined to one; when comparing 2 arrays, the norm of their difference is divided by the norm of the second array)
NT_MINMAX	Normalize the values to a range given by a minimum and a maximum value.

6.2.1.4.67 ETcVnOcrModelType

Offers different OCR model types.

Syntax

Definition:

```
enum ETcVnOcrModelType : ULONGLONG
{
    OMT_NUMBERS                = 580542139465730,
    OMT_NUMBERS_SC             = 580542139465732,
    OMT_UCLETTERS              = 580542139465736,
    OMT_NUMBERS_SC_UCLETTERS   = 580542139465744,
    OMT_CNN_NUMBERS_SC_LETTERS = 598134325510146
};
```

Values

Name	Description
OMT_NUMBERS	Classify numbers. Expects a binary image as input.
OMT_NUMBERS_SC	Classify numbers and special characters
OMT_UCLETTERS	Classify uppercase letters
OMT_NUMBERS_SC_UCLETTERS	Classify numbers, special characters and uppercase letters
OMT_CNN_NUMBERS_SC_LETTERS	Recognizes numbers, special characters, mixed-case letters. Bounding boxes are not returned. Pattern handling is not supported.

6.2.1.4.68 ETcVnOcrOptions

Provides different options to indicate whether certain actions must be performed to achieve the intended result.

Syntax

Definition:

```
enum ETcVnOcrOptions : ULONG
{
    OO_NONE           = 0,
    OO_WITHBLANKS     = 1,
    OO_NOVELTY_LEVEL1 = 131072,
    OO_NOVELTY_LEVEL2 = 524288,
    OO_NOVELTY_LEVEL3 = 2097152
};
```

Values

Name	Description
OO_NONE	No options
OO_WITHBLANKS	Indicates that if blanks were found, they should be included in the result. By default, all blanks are omitted from the result.
OO_NOVELTY_LEVEL1	The novelty detector is used with high sensitivity to novelty. It filters most unknown characters, but can reject valid characters. Novelty detector support depends on the selected model type.
OO_NOVELTY_LEVEL2	The novelty detector is used with balanced sensitivity to novelty. It filters many unknown characters with minimal rejection of valid characters. Novelty detector support depends on the selected model type.

Name	Description
OO_NOVELTY_LEVEL3	The novelty detector is used with reduced sensitivity to novelty. It filters clearly unknown characters and almost never rejects valid characters. Novelty detector support depends on the selected model type.

6.2.1.4.69 ETcVnOrientationMethod

Offers methods to calculate the orientation of a set of points.

Syntax

Definition:

```
enum ETcVnOrientationMethod : LONG
{
    OM_PCA           = 0,
    OM_FITELLIPSE    = 1,
    OM_MOMENTS       = 2,
    OM_ENCLOSINGRECT = 3
};
```

值

名称	描述
OM_PCA	将 PCA 应用于各点，并计算作为中心点的均值点和主轴的旋转角度
OM_FITELLIPSE	在各点周围拟合一个椭圆，并计算中心点和主轴的旋转角度
OM_MOMENTS	计算各点的力矩，估算质心和旋转角度
OM_ENCLOSINGRECT	计算包括各点的最小面积矩形，并计算该矩形的中心点和旋转角度

6.2.1.4.70 ETcVnPaddingMode

Offers padding modes.

Syntax

Definition:

```
enum ETcVnPaddingMode : LONG
{
    PM_NONE        = 0,
    PM_CROP_CENTER = 1,
    PM_LETTERBOX   = 2
};
```

Values

Name	Description
PM_NONE	Resize the image to the desired size without extra processing
PM_CROP_CENTER	Resize and crop the image
PM_LETTERBOX	Resize the image to the desired size while preserving the aspect ratio of the original image

6.2.1.4.71 ETcVnPixelConnectivity

Offers pixel connectivities.

Syntax

Definition:

```
enum ETcVnPixelConnectivity : LONG
{
    PC_4 = 4,
    PC_8 = 8
};
```

Values

Name	Description
PC_4	4-way
PC_8	8-way

6.2.1.4.72 ETcVnPixelEncoding

Offers pixel encodings.

Syntax

Definition:

```
enum ETcVnPixelEncoding : BYTE
{
    PE_NONE                = 0,
    PE_BAYER_GR            = 1,
    PE_BAYER_RG            = 2,
    PE_BAYER_GB            = 3,
    PE_BAYER_BG            = 4,
    PE_YUV_411_UYYVYY      = 5,
    PE_YUV_422_UYVY        = 6,
    PE_YUV_422_YUYV        = 7,
    PE_YCBCR_411_CBYYCRYY = 8,
    PE_YCBCR_422_CBYCRY    = 9,
    PE_YCBCR_422_YCBYCR    = 10
};
```

Values

Name	Description
PE_NONE	No encoding available, i.e. every pixel value is independent of other pixels.
PE_BAYER_GR	Pixels are encoded as a BayerGR pattern.
PE_BAYER_RG	Pixels are encoded as a BayerRG pattern.
PE_BAYER_GB	Pixels are encoded as a BayerGB pattern.
PE_BAYER_BG	Pixels are encoded as a BayerBG pattern.
PE_YUV_411_UYYVYY	Pixels are encoded as YUV411 (UYYVYY).
PE_YUV_422_UYVY	Pixels are encoded as YUV422 (UYVY).
PE_YUV_422_YUYV	Pixels are encoded as YUV422 (YUYV).
PE_YCBCR_411_CBYYCRYY	Pixels are encoded as YCbCr411 (CbYYCrYY).
PE_YCBCR_422_CBYCRY	Pixels are encoded as YCbCr422 (CbYCrY).
PE_YCBCR_422_YCBYCR	Pixels are encoded as YCbCr422 (YCbYCr).

6.2.1.4.73 ETcVnPixelPackMode

Offers pixel packing modes.

Syntax

Definition:

```
enum ETcVnPixelPackMode : BYTE
{
    PPM_NONE          = 0,
    PPM_MONO1P        = 1,
    PPM_MONO2P        = 2,
    PPM_MONO4P        = 3,
    PPM_MONO10PACKED  = 4,
    PPM_MONO12PACKED  = 5,
    PPM_RGB10V1PACKED = 6,
    PPM_RGB10P32      = 7,
    PPM_RGB12V1PACKED = 8,
    PPM_RGB565P       = 9,
    PPM_BGR565P       = 10,
    PPM_MONO10P       = 11,
    PPM_MONO12P       = 12,
    PPM_MONO14P       = 13
};
```

Values

Name	Description
PPM_NONE	No packing
PPM_MONO1P	Mono1p
PPM_MONO2P	Mono2p
PPM_MONO4P	Mono4p
PPM_MONO10PACKED	Mono10Packed or BayerXX10Packed
PPM_MONO12PACKED	Mono12Packed or BayerXX12Packed
PPM_RGB10V1PACKED	RGB10V1Packed
PPM_RGB10P32	RGB10V2Packed (RGB10p32)
PPM_RGB12V1PACKED	RGB12V1Packed
PPM_RGB565P	RGB565p
PPM_BGR565P	BGR565p
PPM_MONO10P	Mono10p or BayerXX10p
PPM_MONO12P	Mono12p or BayerXX12p
PPM_MONO14P	Mono14p or BayerXX14p

6.2.1.4.74 ETcVnPrototypeClusterer

Offers prototype clusterer model types.

Syntax

Definition:

```
enum ETcVnPrototypeClusterer : ULONG
{
    PC_NOVELTY_DETECTOR = 2,
    PC_CLUSTERER        = 8
};
```

Values

Name	Description
PC_NOVELTY_DETECTOR	Novelty detector
PC_CLUSTERER	Clusterer

6.2.1.4.75 ETcVnRectangleIntersection

Offers rectangle intersection types.

Syntax

Definition:

```
enum ETcVnRectangleIntersection : LONG
{
    RI_NONE    = 0,
    RI_PARTIAL = 1,
    RI_FULL    = 2
};
```

Values

Name	Description
RI_NONE	No intersection
RI_PARTIAL	Partial intersection
RI_FULL	Full intersection

6.2.1.4.76 ETcVnReduceType

Offers reduction types.

Syntax

Definition:

```
enum ETcVnReduceType : LONG
{
    RT_REDUCE_SUM = 0,
    RT_REDUCE_AVG = 1,
    RT_REDUCE_MAX = 2,
    RT_REDUCE_MIN = 3
};
```

Values

Name	Description
RT_REDUCE_SUM	The sum (column/row-wise) of all rows/columns
RT_REDUCE_AVG	The mean (column/row-wise) of all rows/columns
RT_REDUCE_MAX	The maximum (column/row-wise) of all rows/columns
RT_REDUCE_MIN	The minimum (column/row-wise) of all rows/columns

6.2.1.4.77 ETcVnRotationAngle

Offers rotation angles.

Syntax

Definition:

```
enum ETcVnRotationAngle : LONG
{
    RA_90_DEG  = 0,
    RA_180_DEG = 1,
    RA_270_DEG = 2
};
```

Values

Name	Description
RA_90_DEG	90 degrees
RA_180_DEG	180 degrees
RA_270_DEG	270 degrees

6.2.1.4.78 ETcVnRTrees

Offers RTrees model types.

Syntax

Definition:

```
enum ETcVnRTrees : ULONG
{
    RT_CLASSIFIER = 1,
    RT_REGRESSOR  = 4
};
```

Values

Name	Description
RT_CLASSIFIER	Classifier
RT_REGRESSOR	Regressor

6.2.1.4.79 ETcVnSignedNormalization

Offers options for normalizing images with signed data types.

Syntax

Definition:

```
enum ETcVnSignedNormalization : SHORT
{
    SN_FIX_ZERO   = 0,
    SN_FULL_SCALE = 1
};
```

Values

Name	Description
SN_FIX_ZERO	The value 0 is fixed, i.e. the normalized image might only reach either the minimum or maximum value.
SN_FULL_SCALE	The image is normalized to the full value range, i.e. a prior value of 0 might be != 0 after normalization.

6.2.1.4.80 ETcVnSolvePnPMethod

Offers SolvePnP methods.

Syntax

Definition:

```
enum ETcVnSolvePnPMethod : LONG
{
    SPM_ITERATIVE   = 0,
    SPM_EPNP        = 1,
    SPM_P3P         = 2,
    SPM_AP3P        = 5,
    SPM_IPPE        = 6,
    SPM_IPPE_SQUARE = 7,
    SPM_SQPNP       = 8
};
```

Values

Name	Description
SPM_ITERATIVE	Levenberg-Marquardt optimization. Requires at least 4 planar or 6 non-planar reference points.
SPM_EPNP	Efficient perspective-n-point camera pose estimation. Requires at least 4 reference points.
SPM_P3P	Complete solution classification for the perspective-three-point problem. Requires exactly 4 reference points to find a unique solution.
SPM_AP3P	An efficient algebraic solution to the perspective-three-point problem. Requires exactly 4 reference points to find a unique solution.
SPM_IPPE	Infinitesimal plane-based pose estimation. Requires at least 4 coplanar reference points.
SPM_IPPE_SQUARE	IPPE specialization for squares. Requires exactly 4 reference points in the following order (hsl: half square length): (-hsl, hsl, 0), (hsl, hsl, 0), (hsl, -hsl, 0), (-hsl, -hsl, 0)
SPM_SQPNP	A consistently fast and globally optimal solution to the perspective-n-point problem. Requires at least 4 reference points.

6.2.1.4.81 ETcVnSta

Offers Simplified TopoART model types.

Syntax

Definition:

```
enum ETcVnSta : ULONG
{
    STA_CLASSIFIER       = 1,
    STA_NOVELTY_DETECTOR = 2,
    STA_REGRESSOR        = 4,
    STA_CLUSTERER        = 8
};
```

Values

Name	Description
STA_CLASSIFIER	Classifier
STA_NOVELTY_DETECTOR	Novelty detector
STA_REGRESSOR	Regressor
STA_CLUSTERER	Clusterer

6.2.1.4.82 ETcVnStructuringElementShape

Offers shapes for a structuring element.

Syntax

Definition:

```
enum ETcVnStructuringElementShape : LONG
{
    SES_RECTANGLE = 0,
    SES_CROSS     = 1,
    SES_ELLIPSE   = 2
};
```

Values

Name	Description
SES_RECTANGLE	Rectangle
SES_CROSS	Cross
SES_ELLIPSE	Ellipse

6.2.1.4.83 ETcVnSvm

Offers SVM model types.

Syntax

Definition:

```
enum ETcVnSvm : ULONG
{
    SVM_C_CLASSIFIER     = 1,
    SVM_NU_CLASSIFIER    = 65537,
    SVM_NOVELTY_DETECTOR = 2,
    SVM_EPS_REGRESSOR    = 4,
    SVM_NU_REGRESSOR     = 65540
};
```

Values

Name	Description
SVM_C_CLASSIFIER	Classifier
SVM_NU_CLASSIFIER	Classifier
SVM_NOVELTY_DETECTOR	Novelty detector
SVM_EPS_REGRESSOR	Regressor
SVM_NU_REGRESSOR	Regressor

6.2.1.4.84 ETcVnSvmKernelType

Offers SVM kernel types.

Syntax

Definition:

```
enum ETcVnSvmKernelType : LONG
{
    SKT_LINEAR  = 0,
    SKT_POLY    = 1,
    SKT_RBF     = 2,
    SKT_SIGMOID = 3,
    SKT_CHI2    = 4,
    SKT_INTER   = 5
};
```

Values

Name	Description
SKT_LINEAR	Linear kernel (fast)
SKT_POLY	Polynomial kernel
SKT_RBF	Radial basis function (RBF) kernel (good default)
SKT_SIGMOID	Sigmoid kernel
SKT_CHI2	Chi-squared kernel
SKT_INTER	Histogram intersection kernel

6.2.1.4.85 ETcVnSvmSgdClassifierMarginType

Offers different margin types for SVMSGD classifiers.

Syntax

Definition:

```
enum ETcVnSvmSgdClassifierMarginType : LONG
{
    SSCMT_SOFT_MARGIN = 0,
    SSCMT_HARD_MARGIN = 1
};
```

Values

Name	Description
SSCMT_SOFT_MARGIN	Soft margin allowing outliers. (beneficial for classes that cannot be separated linearly)
SSCMT_HARD_MARGIN	Hard margin. (best-suited for linearly separable classes)

6.2.1.4.86 ETcVnSvmSgdClassifierType

Offers different types of SVMSGD classifiers.

Syntax

Definition:

```
enum ETcVnSvmSgdClassifierType : LONG
{
    SSCT_SGD  = 0,
    SSCT_ASGD = 1
};
```

Values

Name	Description
SSCT_SGD	Stochastic Gradient Descent
SSCT_ASGD	Average Stochastic Gradient Descent

6.2.1.4.87 ETcVnTemplateMatchMethod

Offers methods for template matching.

Syntax

Definition:

```
enum ETcVnTemplateMatchMethod : LONG
{
    TMM_SQDIFF        = 0,
    TMM_SQDIFF_NORMED = 1,
    TMM_CCORR         = 2,
    TMM_CCORR_NORMED  = 3,
    TMM_CCOEFF        = 4,
    TMM_CCOEFF_NORMED = 5
};
```

Values

Name	Description
TMM_SQDIFF	Squared difference (supports template mask)
TMM_SQDIFF_NORMED	Normalized squared difference

Name	Description
TMM_CCORR	Cross-correlation
TMM_CCORR_NORMED	Normalized cross-correlation (supports template mask)
TMM_CCOEFF	Correlation coefficient
TMM_CCOEFF_NORMED	Normalized correlation coefficient

6.2.1.4.88 ETcVnThresholdType

Offers threshold types.

Syntax

Definition:

```
enum ETcVnThresholdType : LONG
{
    TT_BINARY              = 0,
    TT_BINARY_INV          = 1,
    TT_TRUNC               = 2,
    TT_TOZERO              = 3,
    TT_TOZERO_INV          = 4,
    TT_OTSU_BINARY         = 8,
    TT_OTSU_BINARY_INV     = 9,
    TT_OTSU_TRUNC          = 10,
    TT_OTSU_TOZERO         = 11,
    TT_OTSU_TOZERO_INV     = 12,
    TT_TRIANGLE_BINARY     = 16,
    TT_TRIANGLE_BINARY_INV = 17,
    TT_TRIANGLE_TRUNC      = 18,
    TT_TRIANGLE_TOZERO     = 19,
    TT_TRIANGLE_TOZERO_INV = 20
};
```

Values

Name	Description
TT_BINARY	Binary threshold
TT_BINARY_INV	Inverted binary threshold
TT_TRUNC	Truncated threshold (pixels > thresh are set to thresh, others keep their value)
TT_TOZERO	To zero threshold (pixels < thresh are set to zero, others keep their value)
TT_TOZERO_INV	Inverted to zero threshold (pixels > thresh are set to zero, others keep their value)
TT_OTSU_BINARY	Binary threshold with the threshold value selected according to the Otsu algorithm
TT_OTSU_BINARY_INV	Inverted binary threshold with the threshold value selected according to the Otsu algorithm
TT_OTSU_TRUNC	Truncated threshold with the threshold value selected according to the Otsu algorithm
TT_OTSU_TOZERO	To zero threshold with the threshold value selected according to the Otsu algorithm
TT_OTSU_TOZERO_INV	Inverted to zero threshold with the threshold value selected according to the Otsu algorithm
TT_TRIANGLE_BINARY	Binary threshold with the threshold value selected according to the Triangle algorithm
TT_TRIANGLE_BINARY_INV	Inverted binary threshold with the threshold value selected according to the Triangle algorithm
TT_TRIANGLE_TRUNC	Truncated threshold with the threshold value selected according to the Triangle algorithm

Name	Description
TT_TRIANGLE_TOZERO	To zero threshold with the threshold value selected according to the Triangle algorithm
TT_TRIANGLE_TOZERO_INV	Inverted to zero threshold with the threshold value selected according to the Triangle algorithm

6.2.1.4.89 ETcVnTimestamp

Offers image acquisition timestamps.

语法

定义：

```
enum ETcVnTimestamp : ULONGLONG
{
    TS_IMAGE_COMPLETED    = 0x1,
    TS_RECEPTION_STARTED  = 0x10,
    TS_RECEPTION_FINISHED = 0x20,
    TS_GETCURRENTIMAGE    = 0x100,
    TS_CAMERA             = 0x10000,
    TS_CUSTOM             = 0x1000000,
    TS_LAST_CHANGE        = 0x1000000000000000
};
```

值

名称	描述
TS_IMAGE_COMPLETED	完成图像时的时间戳，即 GigEVision 驱动程序收到所有数据包并将其组合成完整图像的时间。
TS_RECEPTION_STARTED	图像接收开始时的时间戳，即 GigEVision 驱动程序收到 GVSP 引导数据包的时间。
TS_RECEPTION_FINISHED	图像接收完成时的时间戳，即 GigEVision 驱动程序收到 GVSP 结束数据包的时间。
TS_GETCURRENTIMAGE	通过调用 CameraControl FB 的一种 GetCurrentImage 方法在 PLC 中获取图像时的时间戳。
TS_CAMERA	由相机创建并作为 GVSP 引导数据包的一部分传输的时间戳。该时间戳是相机特定的，默认情况下不与 TwinCAT 同步。
TS_CUSTOM	自定义时间戳，可在创建可显示图像时选择设置（Copy/TransformIntoDisplayableImage 函数的 nTimestamp 参数）。
TS_LAST_CHANGE	最近一次通过 TwinCAT Vision 函数改变图像内容的时间戳。

6.2.1.4.90 ETcVnVectorCompareMethod

Offers methods for vector comparison.

Syntax

Definition:

```
enum ETcVnVectorCompareMethod : LONG
{
    VCM_EUCLIDEAN   = 0,
    VCM_ELEMENTWISE = 1
};
```

Values

Name	Description
VCM_EUCLIDEAN	Euclidean
VCM_ELEMENTWISE	Elementwise

6.2.1.4.91 ETcVnWorldCoordinateSystem

Offers world coordinate system orientations.

Syntax

Definition:

```
enum ETcVnWorldCoordinateSystem : LONG
{
    WCS_X_RIGHT_Y_DOWN = 0,
    WCS_X_DOWN_Y_RIGHT = 1,
    WCS_X_LEFT_Y_DOWN  = 2,
    WCS_X_UP_Y_RIGHT   = 3,
    WCS_X_RIGHT_Y_UP   = 4,
    WCS_X_DOWN_Y_LEFT  = 5,
    WCS_X_LEFT_Y_UP    = 6,
    WCS_X_UP_Y_LEFT    = 7
};
```

Values

Name	Description
WCS_X_RIGHT_Y_DOWN	X axis points right, Y axis down (same orientation as image coordinate system)
WCS_X_DOWN_Y_RIGHT	X axis points down, Y axis right
WCS_X_LEFT_Y_DOWN	X axis points left, Y axis down
WCS_X_UP_Y_RIGHT	X axis points up, Y axis right
WCS_X_RIGHT_Y_UP	X axis points right, Y axis up
WCS_X_DOWN_Y_LEFT	X axis points down, Y axis left
WCS_X_LEFT_Y_UP	X axis points left, Y axis up
WCS_X_UP_Y_LEFT	X axis points up, Y axis left

6.2.1.4.92 ETcWatchdogAccumulationType

Offers watchdog accumulation types to compute the fraction processed.

Syntax

Definition:

```
enum ETcWatchdogAccumulationType : USHORT
{
    WD_ACC_TYPE_MEAN    = 1,
    WD_ACC_TYPE_PRODUCT = 2
};
```

Values

Name	Description
WD_ACC_TYPE_MEAN	Calculates the mean of the individual fractions processed over all monitored functions (recommended for independent functions).
WD_ACC_TYPE_PRODUCT	Calculates the product of the individual fractions processed over all monitored functions (recommended for dependent functions).

6.2.1.4.93 GEV_CAMERA_STATE

Describes the GigEVision camera connection state.

Syntax

Definition:

```
enum GEV_CAMERA_STATE : BYTE
{
    GEV_CAMERA_IDLE               = 0,
    GEV_CONTROL_CHANNEL_OPEN_MASK = 1,
    GEV_STREAM_CHANNELS_OPEN_MASK = 2,
    GEV_CONTROL_CHANNEL_OPEN      = 1,
    GEV_STREAM_CHANNELS_OPEN      = 3
};
```

Values

Name	Description
GEV_CAMERA_IDLE	Camera is idle (no control or stream channel open).
GEV_CONTROL_CHANNEL_OPEN_MASK	The control channel is open.
GEV_STREAM_CHANNELS_OPEN_MASK	At least 1 stream channel is open.
GEV_CONTROL_CHANNEL_OPEN	The control channel is open, but no stream channel.
GEV_STREAM_CHANNELS_OPEN	At least 1 stream channel and the control channel are open.

6.2.1.5 Structs

6.2.1.5.1 TcVnParamsFindRois

Parameters for FindRois.

Syntax

Definition:

```
struct TcVnParamsFindRois
{
    TcVnSize2_DINT stRoiSize;
    double         fMinDensity;
    double         fDetectionScaleFactor;
    double         fMaxOversizeFraction;
};
```

Parameters

Name	Type	Default	Description
stRoiSize	TcVnSize2_DINT [▶ 1699]	(.nWidth := 416, .nHeight := 416)	Desired ROI dimensions in original-image pixels.
fMinDensity	double	0.2	Minimum feature density required to keep an ROI, given as a factor of the image's average density. Higher values keep fewer ROIs; suggested range [0.0, 2.0]. 0.0 disables the filter.

Name	Type	Default	Description
fDetectionScaleFactor	double	0.25	Scale factor that controls the resolution of the ROI detection (Lower = faster density evaluation; valid range [0.1, 1.0]).
fMaxOversizeFraction	double	1.0	Fraction by which ROIs may exceed stRoiSize (required only when eFindRoisMethod = TCVN_FRM_CONNECTED COMPONENTS).

6.2.1.5.2 TcVnParamsWalshBarcodeDetection

Parameters for bar code detection based on the Walsh transform.

Syntax

Definition:

```
struct TcVnParamsWalshBarcodeDetection
{
    ULONG                   nMaxBarDistance;
    float                   fMinAspectRatio;
    float                   fMaxAspectRatio;
    ULONG                   nIntensityDifference;
    LONG                    nMinArea;
    LONG                    nChannelIndex;
    ULONG                   nWalshFilterNum;
    float                   fOutlierReductionFactor;
    float                   fBarConnectionFactor;
    float                   fRegionFilterFactor;
    float                   fPostProcessingResizeFactor;
    ETcVnInterpolationType eInterpolationType;
};
```

Parameters

Name	Type	Default	Description
nMaxBarDistance	ULONG	8	Maximum distance between two consecutive dark or bright bars in pixels
fMinAspectRatio	float	1.0f	Minimum aspect ratio (width / height) of valid code regions
fMaxAspectRatio	float	5.0f	Maximum aspect ratio (width / height) of valid code regions
nIntensityDifference	ULONG	100	Approximate mean difference of dark and bright code pixels
nMinArea	LONG	-1	Expected minimum code area in pixels (-1: use an automatic guess based on nMaxBarDistance)
nChannelIndex	LONG	-1	Choose the index of a specific color channel (e.g., 0 = red, 1 = green, 2 = blue) or automatic graylevel conversion (-1).

Name	Type	Default	Description
			Using automatic graylevel conversion, 3-channel images and 4-channel images are considered as RGB and RGBA images, respectively, and converted to graylevel images before processing. For images with another number of channels the first one (index 0) is selected.
nWalshFilterNum	ULONG	5	Number of Walsh filters to be used (higher values may result in improved results but require more time to compute)
fOutlierReductionFactor	float	1.0f	Factor to control outlier reduction (Higher values reduce the number of outlier pixels but may disturb the barcodes to be localized.)
fBarConnectionFactor	float	1.0f	Factor to control bar linking (Higher values reduce the number of outlier pixels but may disturb the barcodes to be localized.)
fRegionFilterFactor	float	1.0f	Factor to control the rejection of small regions and region parts (Higher values reduce the number of found code regions, adjust their shape and increase the computation time.)
fPostProcessingResizeFactor	float	0.5f	Factor to control resizing before post-processiong (Smaller values reduce the required computation time but may deteriorate the results.)
eInterpolationType	ETcVnInterpolationType [▶ 1680]	IT_BILINEAR	Image rotation interpolation type (TCVN_IT_BILINEAR recommended for most cases)

6.2.1.5.3 TcVnSize2_DINT

Contains width and height.

Syntax

Definition:

```
struct TcVnSize2_DINT
{
    LONG nWidth;
    LONG nHeight;
};
```

Parameters

Name	Type	Description
nWidth	LONG	Width
nHeight	LONG	Height

6.2.1.5.4 GVCP_ADDRESS_VALUE_PAIR

描述一对 GVCP 寄存器地址与值。

语法

定义：

```
struct GVCP_ADDRESS_VALUE_PAIR
{
    GVCP_REGISTER_ADDRESS nAddress;
    GVCP_REGISTER_VALUE   nValue;
};
```

参数

名称	类型	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 1640]	寄存器地址
nValue	GVCP_REGISTER_VALUE [▶ 1640]	寄存器值

6.2.1.5.5 GVSP_IMAGE_INFO

Shows GVSP (GigE Vision Streaming Protocol) meta information.

Syntax

Definition:

```
struct GVSP_IMAGE_INFO
{
    ULONG                     CameraIpAddress;
    ULONG                     LocalIpAddress;
    USHORT                    CameraUdpPort;
    USHORT                    LocalUdpPort;
    USHORT                    GvspChannelId;
    USHORT                    GevStatus;
    ULONGLONG                 BlockId;
    GVSP_LEADER_PAYLOAD_IMAGE LeaderInfo;
};
```

Parameters

Name	Type	Description
CameraIpAddress	ULONG	Camera IP address
LocalIpAddress	ULONG	Local IP address
CameraUdpPort	USHORT	Camera UDP port
LocalUdpPort	USHORT	Local UDP port
GvspChannelId	USHORT	ID of GVSP channel

Name	Type	Description
GevStatus	USHORT	GigE Vision status code
BlockId	ULONGLONG	Block Id (incremented for each acquired image, but reset to 0 on overflow)
LeaderInfo	GVSP_LEADER_PAYLOAD_IMAGE [▶ 1701]	Contains information about timestamp, pixel format and size.

6.2.1.5.6 GVSP_LEADER_PAYLOAD_IMAGE

Shows information over the GVSP (GigE Vision Streaming Protocol) leader payload image.

Syntax

Definition:

```
struct GVSP_LEADER_PAYLOAD_IMAGE
{
    ULONGLONG         Timestamp;
    GVSP_PIXEL_FORMAT PixelFormat;
    ULONG             SizeX;
    ULONG             SizeY;
    ULONG             OffsetX;
    ULONG             OffsetY;
    USHORT            PaddingX;
    USHORT            PaddingY;
};
```

Parameters

Name	Type	Description
Timestamp	ULONGLONG	Image timestamp
PixelFormat	GVSP_PIXEL_FORMAT [▶ 1701]	Image pixel format
SizeX	ULONG	Image size in x direction
SizeY	ULONG	Image size in y direction
OffsetX	ULONG	Image x-offset from (0,0) origin
OffsetY	ULONG	Image y-offset from (0,0) origin
PaddingX	USHORT	Image padding in x direction
PaddingY	USHORT	Image padding in y direction

6.2.1.5.7 GVSP_PIXEL_FORMAT

Shows the GVSP (GigE Vision Streaming Protocol) pixel format.

Syntax

Definition:

```
struct GVSP_PIXEL_FORMAT
{
    BYTE   Color;
    BYTE   EffectivePixelSize;
    USHORT Id;
};
```

Parameters

Name	Type	Description
Color	BYTE	Indicates if the pixel format is mono or color.
EffectivePixelSize	BYTE	Effective pixel size in bit
Id	USHORT	GVSP pixel format ID

6.2.1.5.8 TcVnCameraCalibrationOptions

Offers camera calibration options.

Syntax

Definition:

```
struct TcVnCameraCalibrationOptions
{
    UCHAR bUseIntrinsicGuess : 1;
    UCHAR bFixAspectRatio : 1;
    UCHAR bFixPrincipalPoint : 1;
    UCHAR bZeroTangentDist : 1;
    UCHAR bFixFocalLength : 1;
    UCHAR bFixK1 : 1;
    UCHAR bFixK2 : 1;
    UCHAR bFixK3 : 1;
    UCHAR bFixK4 : 1;
    UCHAR bFixK5 : 1;
    UCHAR bFixK6 : 1;
    UCHAR bRationalModel : 1;
};
```

Parameters

Name	Type	Description
bUseIntrinsicGuess	UCHAR : 1	The camera matrix and distortion coefficients already contain a valid initial guess, which is optimized further.
bFixAspectRatio	UCHAR : 1	The ratio fx/fy stays the same as in the input camera matrix.
bFixPrincipalPoint	UCHAR : 1	The principal point is fixed to the image center (or provided cx,cy if bUseIntrinsicGuess).
bZeroTangentDist	UCHAR : 1	The tangential distortion coefficients (p1, p2) are forced to 0.
bFixFocalLength	UCHAR : 1	The parameters fx and fy stay the same as in the input camera matrix.
bFixK1	UCHAR : 1	The radial distortion coefficient k1 is fixed to 0 (or provided input if bUseIntrinsicGuess)
bFixK2	UCHAR : 1	The radial distortion coefficient k2 is fixed to 0 (or provided input if bUseIntrinsicGuess)
bFixK3	UCHAR : 1	The radial distortion coefficient k3 is fixed to 0 (or provided input if bUseIntrinsicGuess)
bFixK4	UCHAR : 1	The radial distortion coefficient k4 is fixed to 0 (or provided input if bUseIntrinsicGuess)
bFixK5	UCHAR : 1	The radial distortion coefficient k5 is fixed to 0 (or provided input if bUseIntrinsicGuess)
bFixK6	UCHAR : 1	The radial distortion coefficient k6 is fixed to 0 (or provided input if bUseIntrinsicGuess)
bRationalModel	UCHAR : 1	The radial distortion coefficients k4, k5, k6 are enabled.

6.2.1.5.9 TcVnCircularArc

Describes a circular arc.

Syntax

Definition:

```
struct TcVnCircularArc
{
    TcVnPoint2_REAL aCenter;
    float           fRadius;
    float           fStartAngle;
    float           fEndAngle;
};
```

Parameters

Name	Type	Description
aCenter	TcVnPoint2_REAL [▶ 1640]	Center of the circular arc
fRadius	float	Radius of the circular arc
fStartAngle	float	Start angle of the circular arc
fEndAngle	float	End angle of the circular arc

6.2.1.5.10 TcVnCodeGrades1D

Describes code quality grades for 1D barcodes according to ISO / IEC 15416:2016. The grades range from 0 (very bad) to 4 (very good) and are averaged over several scan lines.

Syntax

Definition:

```
struct TcVnCodeGrades1D
{
    float fDecode;
    float fSymbolContrast;
    float fMinReflectance;
    float fMinEdgeContrast;
    float fModulation;
    float fDefects;
    float fDecodability;
    float fOverall;
};
```

Parameters

Name	Type	Description
fDecode	float	Decode grade
fSymbolContrast	float	Symbol contrast grade
fMinReflectance	float	Minimum reflectance grade
fMinEdgeContrast	float	Minimum edge contrast grade
fModulation	float	Modulation grade
fDefects	float	Defects grade
fDecodability	float	Decodability grade
fOverall	float	Overall code grade, i.e. the minimum achieved individual grade averaged over all scan lines. If different scan lines lead to different decoded data, the overall grade is 0.

6.2.1.5.11 TcVnCodeGradesDM

Describes code quality grades for Data Matrix codes according to ISO / IEC 15415:2011. The grades range from 0 (very bad) to 4 (very good).

Syntax

Definition:

```
struct TcVnCodeGradesDM
{
    unsigned char nDecode;
    unsigned char nSymbolContrast;
    unsigned char nModulation;
    unsigned char nReflectanceMargin;
    unsigned char nFixedPatternDamage;
    unsigned char nAxialNonuniformity;
    unsigned char nGridNonuniformity;
    unsigned char nUnusedErrorCorrection;
    unsigned char nOverall;
};
```

Parameters

Name	Type	Description
nDecode	unsigned char	Decode grade
nSymbolContrast	unsigned char	Symbol contrast grade
nModulation	unsigned char	Modulation grade
nReflectanceMargin	unsigned char	Reflectance margin grade
nFixedPatternDamage	unsigned char	Fixed pattern damage grade
nAxialNonuniformity	unsigned char	Axial nonuniformity grade
nGridNonuniformity	unsigned char	Grid nonuniformity grade
nUnusedErrorCorrection	unsigned char	Unused error correction grade
nOverall	unsigned char	Overall code grade, i.e. the minimum achieved individual grade

6.2.1.5.12 TcVnCodeGradesQR

Describes code quality grades for QR codes according to ISO / IEC 15415:2011. The grades range from 0 (very bad) to 4 (very good).

Syntax

Definition:

```
struct TcVnCodeGradesQR
{
    unsigned char nDecode;
    unsigned char nSymbolContrast;
    unsigned char nModulation;
    unsigned char nReflectanceMargin;
    unsigned char nFixedPatternDamage;
    unsigned char nAxialNonuniformity;
    unsigned char nGridNonuniformity;
    unsigned char nUnusedErrorCorrection;
    unsigned char nFormatInfo;
    unsigned char nVersionInfo;
    unsigned char nOverall;
};
```

Parameters

Name	Type	Description
nDecode	unsigned char	Decode grade
nSymbolContrast	unsigned char	Symbol contrast grade

Name	Type	Description
nModulation	unsigned char	Modulation grade
nReflectanceMargin	unsigned char	Reflectance margin grade
nFixedPatternDamage	unsigned char	Fixed pattern damage grade
nAxialNonuniformity	unsigned char	Axial nonuniformity grade
nGridNonuniformity	unsigned char	Grid nonuniformity grade
nUnusedErrorCorrection	unsigned char	Unused error correction grade
nFormatInfo	unsigned char	Format information grade
nVersionInfo	unsigned char	Version information grade
nOverall	unsigned char	Overall code grade, i.e. the minimum achieved individual grade

6.2.1.5.13 TcVnDMatch

Describes a descriptor match

Syntax

Definition:

```
struct TcVnDMatch
{
    LONG  nQueryIdx;
    LONG  nTrainIdx;
    LONG  nImageIdx;
    float fDistance;
};
```

Parameters

Name	Type	Description
nQueryIdx	LONG	Query descriptor index
nTrainIdx	LONG	Train descriptor index
nImageIdx	LONG	Train image index
fDistance	float	Distance between the descriptors (smaller distance means better match)

6.2.1.5.14 TcVnImageInfo

Shows image information.

Syntax

Definition:

```
struct TcVnImageInfo
{
    ULONGLONG       nImageSize;
    ULONG           nWidth;
    ULONG           nHeight;
    USHORT          nXPadding;
    USHORT          nYPadding;
    TcVnPixelFormat stPixelFormat;
};
```

Parameters

Name	Type	Description
nImageSize	ULONGLONG	Image size (number of pixels)

Name	Type	Description
nWidth	ULONG	Image width
nHeight	ULONG	Image height
nXPadding	USHORT	Image x-padding
nYPadding	USHORT	Image y-padding
stPixelFormat	TcVnPixelFormat [▶ 1716]	Pixel format

6.2.1.5.15 TcVnKeyPoint

Describes a key point.

Syntax

Definition:

```
struct TcVnKeyPoint
{
    TcVnPoint2_REAL aPoint;
    float           fDiameter;
    float           fAngle;
    float           fResponse;
    LONG            nOctave;
    LONG            nClassId;
};
```

Parameters

Name	Type	Description
aPoint	TcVnPoint2_REAL [▶ 1640]	Position
fDiameter	float	Diameter
fAngle	float	Angle
fResponse	float	Response
nOctave	LONG	Octave
nClassId	LONG	Class ID

6.2.1.5.16 TcVnMatrix

Offers a user-defined matrix with variable rows, columns and element-type.

Syntax

Definition:

```
struct TcVnMatrix
{
    ULONG            nRows;
    ULONG            nCols;
    ETcVnElementType eType;
    PVOID            pData;
};
```

Parameters

Name	Type	Description
nRows	ULONG	Number of rows
nCols	ULONG	Number of columns
eType	ETcVnElementType [▶ 1674]	Element type
pData	PVOID	Pointer to the data

6.2.1.5.17 TcVnMoments

Offers image or contour moments.

Syntax

Definition:

```
struct TcVnMoments
{
    double fM00;
    double fM10;
    double fM01;
    double fM20;
    double fM11;
    double fM02;
    double fM30;
    double fM21;
    double fM12;
    double fM03;
    double fMu20;
    double fMu11;
    double fMu02;
    double fMu30;
    double fMu21;
    double fMu12;
    double fMu03;
    double fNu20;
    double fNu11;
    double fNu02;
    double fNu30;
    double fNu21;
    double fNu12;
    double fNu03;
};
```

Parameters

Name	Type	Description
fM00	double	Spatial moment 00
fM10	double	Spatial moment 10
fM01	double	Spatial moment 01
fM20	double	Spatial moment 20
fM11	double	Spatial moment 11
fM02	double	Spatial moment 02
fM30	double	Spatial moment 30
fM21	double	Spatial moment 21
fM12	double	Spatial moment 12
fM03	double	Spatial moment 03
fMu20	double	Central moment 20
fMu11	double	Central moment 11
fMu02	double	Central moment 02
fMu30	double	Central moment 30
fMu21	double	Central moment 21
fMu12	double	Central moment 12
fMu03	double	Central moment 03
fNu20	double	Normalized central moment 20
fNu11	double	Normalized central moment 11
fNu02	double	Normalized central moment 02
fNu30	double	Normalized central moment 30
fNu21	double	Normalized central moment 21
fNu12	double	Normalized central moment 12
fNu03	double	Normalized central moment 03

6.2.1.5.18 TcVnParamsAGAST

Offers parameters for AGAST method.

Syntax

Definition:

```
struct TcVnParamsAGAST
{
    LONG                            nThreshold;
    bool                            bNonMaxSuppression;
    ETcVnKeypointDetectionTypeAGAST eType;
};
```

Parameters

Name	Type	Default	Description
nThreshold	LONG	10	Threshold for the intensity difference between the center pixel and its surrounding circle
bNonMaxSuppression	bool	true	If true, non-maximum suppression is applied.
eType	ETcVnKeypointDetectionTypeAGAST [▶ 1680]	KDT_AGAST_9_16	Neighborhood type

6.2.1.5.19 TcVnParamsAKAZE

Offers parameters for AKAZE method.

Syntax

Definition:

```
struct TcVnParamsAKAZE
{
    ETcVnFeatureDescriptorTypeAKAZE eDescrType;
    ULONG                           nDescrSize;
    ULONG                           nDescrChannels;
    float                           fThreshold;
    ULONG                           nOctaves;
    ULONG                           nOctaveLayers;
    ETcVnDiffusivityTypeKAZE        eDiffusivity;
};
```

Parameters

Name	Type	Default	Description
eDescrType	ETcVnFeatureDescriptorTypeAKAZE [▶ 1676]	FDT_AKAZE_MLDB	Type of the descriptor
nDescrSize	ULONG	0	Size of the descriptor in bits (only for MLDB; 0 = full size)
nDescrChannels	ULONG	3	Number of descriptor channels (currently only 3 supported for MLDB)
fThreshold	float	0.001f	Detector response threshold

Name	Type	Default	Description
nOctaves	ULONG	2	Maximum octave evolution
nOctaveLayers	ULONG	1	Number of sublevels per scale level
eDiffusivity	ETcVnDiffusivityTypeKAZE [▶ 1670]	DT1_KAZE_PM_G2	Diffusivity type

6.2.1.5.20 TcVnParamsBlobDetection

Offers parameters for blob detection

Syntax

Definition:

```
struct TcVnParamsBlobDetection
{
    bool                 bFilterByArea;
    bool                 bFilterByCircularity;
    bool                 bFilterByConvexity;
    bool                 bFilterByEccentricity;
    bool                 bFilterByInertiaRatio;
    float                fMinArea;
    float                fMaxArea;
    float                fMinCircularity;
    float                fMaxCircularity;
    float                fMinConvexity;
    float                fMaxConvexity;
    float                fMinEccentricity;
    float                fMaxEccentricity;
    float                fMinInertiaRatio;
    float                fMaxInertiaRatio;
    ETcVnThresholdType   eThresholdType;
    float                fMinThreshold;
    float                fMaxThreshold;
    float                fThresholdStep;
    float                fMinBlobDistance;
    ULONG                nMinRepeatability;
    ETcVnBlobCombination eBlobCombination;
    bool                 bExcludeBorderBlobs;
};
```

Parameters

Name	Type	Default	Description
bFilterByArea	bool	true	Enable filtering by area (fMinArea <= area <= fMaxArea); strongly recommended to activate for filtering noise with fMinArea.
bFilterByCircularity	bool	false	Enable filtering by circularity (fMinCircularity <= circularity(4pi * area / perimeter^2) <= fMaxCircularity).
bFilterByConvexity	bool	false	Enable filtering by convexity (fMinConvexity <= convexity(area / hullArea) <= fMaxConvexity).

Name	Type	Default	Description
bFilterByEccentricity	bool	false	Enable filtering by eccentricity (fMinEccentricity <= eccentricity <= fMaxEccentricity).
bFilterByInertiaRatio	bool	false	Enable filtering by inertia ratio (fMinInertiaRatio <= inertia ratio <= fMaxInertiaRatio).
fMinArea	float	10	Min estimated blob area in pixel
fMaxArea	float	100000000	Max estimated blob area in pixel
fMinCircularity	float	0	Min circularity (1.0: ideal circle, < 1: less circular, 0: not circular at all)
fMaxCircularity	float	1	Max circularity (1.0: ideal circle, < 1: less circular, 0: not circular at all)
fMinConvexity	float	0	Min convexity (1.0: blob fully convex, < 1: less convex)
fMaxConvexity	float	1	Max convexity (1.0: blob fully convex, < 1: less convex)
fMinEccentricity	float	0	Min eccentricity (0.0: circular, 1.0: linear)
fMaxEccentricity	float	1	Max eccentricity (0.0: circular, 1.0: linear)
fMinInertiaRatio	float	0	Min inertia ratio (1.0: equal width and height, 0.0: linear)
fMaxInertiaRatio	float	1	Max inertia ratio (1.0: equal width and height, 0.0: linear)
eThresholdType	ETcVnThresholdType [▶ 1694]	TT_BINARY	Threshold type for internally applied threshold method (OTSU_XXX only supported for USINT images).
fMinThreshold	float	30	Threshold to start with (if fThresholdStep > 0, otherwise this is the only threshold used).
fMaxThreshold	float	225	Threshold to end with (if fThresholdStep > 0, otherwise this value is not used).

Name	Type	Default	Description
fThresholdStep	float	0	Sets to 0 if only 1 threshold should be used (much faster than multiple thresholds and combining the results).
fMinBlobDistance	float	5	Minimum distance between the center points of two different blobs (only used if fThresholdStep > 0; if distance < fMinBlobDistance, the blobs are treated as the same).
nMinRepeatability	ULONG	2	Minimum number of threshold steps, for which the same contour has to be detected (only used if fThresholdStep > 0; same means center point distance < fMinBlobDistance).
eBlobCombination	ETcVnBlobCombination [▶ 1647]	BC_MEDIAN_THRESHOLD	Selects, which of the multi-threshold blob contours should be returned.
bExcludeBorderBlobs	bool	false	If true, blobs touching the image border are excluded from the results.

6.2.1.5.21 TcVnParamsBRISK

Offers parameters for BRISK method.

Syntax

Definition:

```
struct TcVnParamsBRISK
{
    LONG  nThreshold;
    ULONG nOctaves;
    float fPatternScale;
};
```

Parameters

Name	Type	Default	Description
nThreshold	LONG	30	Detection threshold
nOctaves	ULONG	3	Detection octaves (0 for single scale)
fPatternScale	float	1	Scale factor for the neighborhood pattern

6.2.1.5.22 TcVnParamsFAST

Offers parameters for FAST method.

Syntax

Definition:

```
struct TcVnParamsFAST
{
    LONG                            nThreshold;
    bool                            bNonMaxSuppression;
    ETcVnKeypointDetectionTypeFAST eType;
};
```

Parameters

Name	Type	Default	Description
nThreshold	LONG	10	Threshold for the intensity difference between the center pixel and its surrounding circle
bNonMaxSuppression	bool	true	If true, non-maximum suppression is applied.
eType	ETcVnKeypointDetectionTypeFAST [▶ 1681]	KDT_FAST_9_16	Neighborhood type

6.2.1.5.23 TcVnParamsGFTT

Offers parameters for GFTT method.

Syntax

Definition:

```
struct TcVnParamsGFTT
{
    ULONG  nMaxCorners;
    double fQualityLevel;
    double fMinDistance;
    ULONG  nBlockSize;
    bool   bUseHarrisDetector;
    double fHarrisK;
};
```

Parameters

Name	Type	Default	Description
nMaxCorners	ULONG	1000	Maximum number of corners to return (strongest ones)
fQualityLevel	double	0.01	Minimum accepted corner quality, relative to the strongest one
fMinDistance	double	1	Minimum euclidean distance between returned keypoints
nBlockSize	ULONG	3	Neighborhood block size
bUseHarrisDetector	bool	false	If true, a Harris detector is used instead of the default method.

Name	Type	Default	Description
fHarrisK	double	0.04	Free parameter of the Harris detector

6.2.1.5.24 TcVnParamsKAZE

Offers parameters for KAZE method

Syntax

Definition:

```
struct TcVnParamsKAZE
{
    bool                       bExtended;
    bool                       bUpright;
    float                      fThreshold;
    ULONG                      nOctaves;
    ULONG                      nOctaveLayers;
    ETcVnDiffusivityTypeKAZE eDiffusivity;
};
```

Parameters

Name	Type	Default	Description
bExtended	bool	false	If true, the descriptor size is extended from the default 64 byte to 128 byte.
bUpright	bool	false	If true, upright (rotation dependent) descriptors are computed.
fThreshold	float	0.001f	Detector response threshold
nOctaves	ULONG	4	Maximum octave evolution of the image
nOctaveLayers	ULONG	2	Number of sublevels per scale level
eDiffusivity	ETcVnDiffusivityTypeKAZE [▶ 1670]	DT1_KAZE_PM_G2	Diffusivity type

6.2.1.5.25 TcVnParamsMSER

Offers parameters for MSER method

Syntax

Definition:

```
struct TcVnParamsMSER
{
    LONG   nDelta;
    ULONG  nMinArea;
    ULONG  nMaxArea;
    double fMaxVariation;
    double fMinDiversity;
    ULONG  nMaxEvolution;
    double fAreaThreshold;
    double fMinMargin;
    ULONG  nEdgeBlurSize;
};
```

Parameters

Name	Type	Default	Description
nDelta	LONG	5	Delta for size comparison
nMinArea	ULONG	60	Minimum blob size
nMaxArea	ULONG	14400	Maximum blob size
fMaxVariation	double	0.25	Maximum blob size variation
fMinDiversity	double	0.2	Minimum MSER diversity
nMaxEvolution	ULONG	200	Maximum evolution steps (only used for color images)
fAreaThreshold	double	1.01	Area threshold to cause re-initialization (only used for color images)
fMinMargin	double	0.003	Minimum margin (only used for color images)
nEdgeBlurSize	ULONG	5	Aperture size for edge bluring (only used for color images)

6.2.1.5.26 TcVnParamsORB

Offers parameters for ORB method.

Syntax

Definition:

```
struct TcVnParamsORB
{
    ULONG                     nMaxPoints;
    float                     fPyramidScale;
    ULONG                     nPyramidLevels;
    ULONG                     nEdgeThreshold;
    ULONG                     nFirstLevel;
    ULONG                     nBriefPoints;
    ETcVnFeatureScoreTypeORB  eScoreType;
    ULONG                     nPatchSize;
    LONG                      nFastThreshold;
};
```

Parameters

Name	Type	Default	Description
nMaxPoints	ULONG	500	Maximum number of returned keypoints
fPyramidScale	float	1.2f	Pyramid decimation ratio (must be greater than 1, should be smaller than 2)
nPyramidLevels	ULONG	8	Number of pyramid levels
nEdgeThreshold	ULONG	31	Size of the border, where no features are detected (should match nPatchSize)

Name	Type	Default	Description
nFirstLevel	ULONG	0	First pyramid level (currently, only 0 is supported)
nBriefPoints	ULONG	2	Number of points to produce each BRIEF descriptor element (2, 3, 4)
eScoreType	ETcVnFeatureScoreTypeORB [▶ 1676]	FST_ORB_HARRIS	Score type (HARRIS is more stable but slightly slower than FAST)
nPatchSize	ULONG	31	Patch size of the BRIEF descriptor
nFastThreshold	LONG	20	Threshold for the FAST keypoint detection

6.2.1.5.27 TcVnParamsSB

Offers parameters for SB method (a simple blob detector with multiple thresholds).

Syntax

Definition:

```
struct TcVnParamsSB
{
    bool          bFilterByArea;
    bool          bFilterByCircularity;
    bool          bFilterByColor;
    bool          bFilterByConvexity;
    bool          bFilterByInertia;
    float         fMinArea;
    float         fMaxArea;
    float         fMinCircularity;
    float         fMaxCircularity;
    unsigned char nBlobColor;
    float         fMinConvexity;
    float         fMaxConvexity;
    float         fMinInertiaRatio;
    float         fMaxInertiaRatio;
    float         fMinBlobDist;
    ULONG         nMinRepeatability;
    float         fMinThreshold;
    float         fMaxThreshold;
    float         fThresholdStep;
};
```

Parameters

Name	Type	Default	Description
bFilterByArea	bool	true	Enable filtering by area (fMinArea <= area < fMaxArea).
bFilterByCircularity	bool	false	Enable filtering by circularity (fMinCircularity <= circularity(4pi * area / perimeter^2) < fMaxCircularity).
bFilterByColor	bool	false	Enable filtering by color (thresholdedColor(0 or 255) = nBlobColor).

Name	Type	Default	Description
bFilterByConvexity	bool	false	Enable filtering by convexity (fMinConvexity <= convexity(area / hullArea) < fMaxConvexity).
bFilterByInertia	bool	false	Enable filtering by inertia ratio (fMinInertiaRatio <= inertia ratio < fMaxInertiaRatio).
fMinArea	float	25	Min estimated blob area in pixel
fMaxArea	float	15000	Max estimated blob area in pixel
fMinCircularity	float	0	Min circularity (1.0: ideal circle, < 1: less circular, 0: not circular at all)
fMaxCircularity	float	1	Max circularity (1.0: ideal circle, < 1: less circular, 0: not circular at all)
nBlobColor	unsigned char	255	0 or 255
fMinConvexity	float	0	Min convexity (1.0: blob fully convex, < 1: less convex)
fMaxConvexity	float	1	Max convexity (1.0: blob fully convex, < 1: less convex)
fMinInertiaRatio	float	0	Min inertia ratio (0.0 .. 1.0)
fMaxInertiaRatio	float	1	Max inertia ratio (0.0 .. 1.0)
fMinBlobDist	float	5	Min distance between different blobs
nMinRepeatability	ULONG	2	Min number of same detected blobs (dist < fMinBlobDist) in different thresholds
fMinThreshold	float	30	Min threshold (start)
fMaxThreshold	float	225	Max threshold (stop)
fThresholdStep	float	10	Threshold step between min and max threshold

6.2.1.5.28 TcVnPixelFormat

Contains detailed information about the pixel format.

Syntax

Definition:

```
struct TcVnPixelFormat
{
    UCHAR              bSupported : 1;
    UCHAR              bSigned : 1;
    UCHAR              bPlanar : 1;
    UCHAR              bFloat : 1;
    BYTE               nChannels;
    ETcVnPixelEncoding ePixelEncoding;
```

```
    ETcVnPixelPackMode ePixelPackMode;
    USHORT             nElementSize;
    USHORT             nTotalSize;
};
```

Parameters

Name	Type	Description
bSupported	UCHAR : 1	If false, the pixel format is not supported.
bSigned	UCHAR : 1	If true, pixel intensities are signed values.
bPlanar	UCHAR : 1	If true, the image channels are stored planar instead of interleaved (e.g. RRRRRGGGGGBBBBB instead of RGBRGBRGBRGBRGB).
bFloat	UCHAR : 1	If true, the pixel format is floating point.
nChannels	BYTE	Number of channels
ePixelEncoding	ETcVnPixelEncoding [▶ 1687]	Pixel encoding
ePixelPackMode	ETcVnPixelPackMode [▶ 1687]	Pixel pack mode
nElementSize	USHORT	Size (bit) of a single pixel channel
nTotalSize	USHORT	Size (bit) of all pixel channels

6.2.1.5.29 TcVnRectangle_DINT

Contains origin and size of a rectangle.

Syntax

Definition:

```
struct TcVnRectangle_DINT
{
    LONG nX;
    LONG nY;
    LONG nWidth;
    LONG nHeight;
};
```

Parameters

Name	Type	Description
nX	LONG	X coordinate of the top-left corner
nY	LONG	Y coordinate of the top-left corner
nWidth	LONG	Width
nHeight	LONG	Height

6.2.1.5.30 TcVnRectangle_UDINT

Contains origin and size of a rectangle.

Syntax

Definition:

```
struct TcVnRectangle_UDINT
{
    ULONG nX;
    ULONG nY;
    ULONG nWidth;
    ULONG nHeight;
};
```

Parameters

Name	Type	Description
nX	ULONG	X coordinate of the top-left corner
nY	ULONG	Y coordinate of the top-left corner
nWidth	ULONG	Width
nHeight	ULONG	Height

6.2.1.5.31 TcVnRotatedRectangle

Contains center, size and angle of a rotated rectangle.

Syntax

Definition:

```
struct TcVnRotatedRectangle
{
    TcVnPoint2_REAL aCenter;
    TcVnSize2_REAL  stSize;
    float           fAngle;
};
```

Parameters

Name	Type	Description
aCenter	TcVnPoint2_REAL [▶ 1640]	Center point
stSize	TcVnSize2_REAL [▶ 1718]	Size composed of fWidth and fHeight
fAngle	float	Angle in degree

6.2.1.5.32 TcVnSize2_REAL

Contains width and height.

Syntax

Definition:

```
struct TcVnSize2_REAL
{
    float fWidth;
    float fHeight;
};
```

Parameters

Name	Type	Description
fWidth	float	Width
fHeight	float	Height

6.2.2 接口

接口不需要许可证。

接口分组

TwinCAT Vision 库的所有接口按主题分类为以下几组：

- 图像 [▶ 1831] ()
- 容器 [▶ 1719] ()
- 机器学习 [▶ 1869] ()
- 其他 [▶ 1872] ()

6.2.2.1 容器

该组包含用于处理容器 [▶ 146]的接口。

常规信息

容器基本上是几个元素组成的矢量。元素的数量可以动态地改变。路由器内存 [▶ 54]中的相应内存为此自动分配和释放。

容器类型

容器的所有元素都为相同的类型。根据容器包含的元素类型，容器的特点为独特的 GUID。为了便于使用，这些 GUID 由常量 [▶ 152]定义。容器的所有类型名称以“ContainerType_Vector_”开始（“ContainerType_String_SINT”除外）。对于所有使用容器的 API 元素，类型要求都有记录。

容器类型可按以下方式确定：

```
ipContainer.GetElementTypeGuid(nTypeGuid);
```

6.2.2.1.1 功能块

该组包含用于实现某些功能块的接口。

6.2.2.1.1.1 ITcVnCustomContainerOperation_ITcVnContainer

为多达 3 个不同容器的自定义容器运算（容器的容器）提供接口。

继承层次

ITcVnCustomContainerOperation_ITcVnContainer

方法

名称	起源	描述
Execute [▶ 1720]	ITcVnCustomContainerOperation_ITcVnContainer	对容器元素执行自定义操作。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.1.1.1 *执行*

对容器元素执行自定义操作。

语法

定义：

```
HRESULT Execute(
    ITcVnContainer* ipElement1,
    ITcVnContainer* ipElement2,
    ITcVnContainer* ipElement3
)
```

参数

名称	类型	描述
ipElement1	ITcVnContainer* [▶ 1827]	第 1 个容器的当前元素。
ipElement2	ITcVnContainer* [▶ 1827]	第 2 个容器的当前元素。
ipElement3	ITcVnContainer* [▶ 1827]	第 3 个容器的当前元素。

返回值

HRESULT [▶ 135]

6.2.2.1.1.2 ITcVnCustomContainerOperation_ITcVnForwardIterator

为多达 3 个不同容器的自定义逐元素容器运算提供接口。

继承层次

ITcVnCustomContainerOperation_ITcVnForwardIterator

方法

名称	起源	描述
Execute [▶ 1720]	ITcVnCustomContainerOperation_ITcVnForwardIterator	对容器元素执行自定义操作。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.1.2.1 *执行*

对容器元素执行自定义操作。

语法

定义：

```
HRESULT Execute(
    ITcVnForwardIterator* ipElement1,
    ITcVnForwardIterator* ipElement2,
    ITcVnForwardIterator* ipElement3
)
```

参数

名称	类型	描述
ipElement1	ITcVnForwardIterator* [▶ 1819]	第 1 个容器的当前元素。
ipElement2	ITcVnForwardIterator* [▶ 1819]	第 2 个容器的当前元素。
ipElement3	ITcVnForwardIterator* [▶ 1819]	第 3 个容器的当前元素。

返回值

HRESULT [▶ 135]

6.2.2.1.1.3 ITcVnCustomElementCondition_ITcVnContainer

提供一个接口，用于为容器计算的自定义条件。

继承层次

ITcVnCustomElementCondition_ITcVnContainer

方法

名称	起源	描述
Condition [▶ 1721]	ITcVnCustomElementCondition_ITcVnContainer	评估容器的条件。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.1.3.1 *条件*

评估容器的条件。

语法

定义：

```
bool Condition(
    ITcVnContainer* ipElement
)
```

参数

名称	类型	描述
ipElement	ITcVnContainer* [▶ 1827]	为其评估条件的容器。

返回值

BOOL

6.2.2.1.1.4 ITcVnCustomElementCondition_ITcVnForwardIterator

提供一个接口，用于为 ITcVnForwardIterator 表示的元素计算自定义条件。

继承层次

ITcVnCustomElementCondition_ITcVnForwardIterator

方法

名称	起源	描述
Condition [▶ 1722]	ITcVnCustomElementCondition_ITcVnForwardIterator	评估所提供元素的条件。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.1.4.1 *条件*

评估所提供元素的条件。

语法

定义：

```
bool Condition(
    ITcVnForwardIterator* ipElement
)
```

参数

名称	类型	描述
ipElement	ITcVnForwardIterator* [▶ 1819]	为其评估条件的元素。

返回值

BOOL

6.2.2.1.2 ITcVnAccess

接口`ITcVnAccess`用于访问容器元素 [▶ 146]，并由迭代器 [▶ 1818]提供。与ITcVnRandomAccess [▶ 1767]不同，只有相关迭代器在那一刻指向的元素可以被寻址。

6.2.2.1.2.1 ITcVnAccess_DINT

提供 DINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_DINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。

名称	起源	描述
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1723]	ITcVnAccess_DINT	获取值
Set [▶ 1723]	ITcVnAccess_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.1.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    LONG& nValue
)
```

参数

名称	类型	描述
nValue	LONG&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.1.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    LONG nValue
)
```

参数

名称	类型	描述
nValue	LONG	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.2 ITcVnAccess_INT

提供 INT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
ITcVnAccess_INT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1724]	ITcVnAccess_INT	获取值
Set [▶ 1724]	ITcVnAccess_INT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.2.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    SHORT& nValue
)
```

参数

名称	类型	描述
nValue	SHORT&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.2.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    SHORT nValue
)
```

参数

名称	类型	描述
nValue	SHORT	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.3 ITcVnAccess_ITcVnImage

提供图像访问接口。

继承层次

ITcUnknown [▶ 1877]
ITcVnAccess_ITcVnImage

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1725]	ITcVnAccess_ITcVnImage	获取图像。
Set [▶ 1725]	ITcVnAccess_ITcVnImage	设置图像。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.3.1 *获取*

获取图像。

语法

定义：

```
HRESULT Get(
    ITcVnImage*& ipImage
)
```

参数

名称	类型	描述
ipImage	ITcVnImage* [▶ 1863]&	返回图像。

返回值

HRESULT [▶ 135]

6.2.2.1.2.3.2 *设置*

设置图像。

语法

定义：

```
HRESULT Set(
    ITcVnImage* ipImage
)
```

参数

名称	类型	描述
ipImage	ITcVnImage* [▶ 1863]	要设置的图像。

返回值

HRESULT [▶ 135]

6.2.2.1.2.4 ITcVnAccess_LREAL

提供 LREAL 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_LREAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1726]	ITcVnAccess_LREAL	获取值
Set [▶ 1727]	ITcVnAccess_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.4.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    double& fValue
)
```

参数

名称	类型	描述
fValue	double&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.4.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    double fValue
)
```

参数

名称	类型	描述
fValue	double	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.5 ITcVnAccess_REAL

提供 REAL 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
ITcVnAccess_REAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1727]	ITcVnAccess_REAL	获取值
Set [▶ 1728]	ITcVnAccess_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.5.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    float& fValue
)
```

参数

名称	类型	描述
fValue	float&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.5.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    float fValue
)
```

参数

名称	类型	描述
fValue	float	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.6 ITcVnAccess_SINT

提供 SINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_SINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1729]	ITcVnAccess_SINT	获取值
Set [▶ 1729]	ITcVnAccess_SINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.6.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    char& nValue
)
```

参数

名称	类型	描述
nValue	char&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.6.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    char nValue
)
```

参数

名称	类型	描述
nValue	char	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.7 ITcVnAccess_TcVnDMatch

提供 TcVnDMatch 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnDMatch

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1730]	ITcVnAccess_TcVnDMatch	获取值

名称	起源	描述
Set [▶ 1730]	ITcVnAccess_TcVnDMatch	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.7.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnDMatch& stDMatch
)
```

参数

名称	类型	描述
stDMatch	TcVnDMatch [▶ 1705]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.7.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnDMatch& stDMatch
)
```

参数

名称	类型	描述
stDMatch	TcVnDMatch [▶ 1705]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.8 ITcVnAccess_TcVnKeyPoint

提供 TcVnKeyPoint 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnKeyPoint

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1731]	ITcVnAccess_TcVnKeyPoint	获取值。
Set [▶ 1731]	ITcVnAccess_TcVnKeyPoint	设置值。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.8.1 *获取*

获取值。

语法

定义：

```
HRESULT Get(
    TcVnKeyPoint& stKeyPoint
)
```

参数

名称	类型	描述
stKeyPoint	TcVnKeyPoint [▶ 1706]&	返回值。

返回值

HRESULT [▶ 135]

6.2.2.1.2.8.2 *设置*

设置值。

语法

定义：

```
HRESULT Set(
    TcVnKeyPoint& stKeyPoint
)
```

参数

名称	类型	描述
stKeyPoint	TcVnKeyPoint [▶ 1706]&	要设置的值。

返回值

HRESULT [▶ 135]

6.2.2.1.2.9 ITcVnAccess_TcVnPoint2_DINT

提供 TcVnPoint2_DINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnPoint2_DINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1732]	ITcVnAccess_TcVnPoint2_DINT	获取值。
Set [▶ 1732]	ITcVnAccess_TcVnPoint2_DINT	设置值。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.9.1 *获取*

获取值。

语法

定义：

```
HRESULT Get(
    TcVnPoint2_DINT& aPoint
)
```

参数

名称	类型	描述
aPoint	TcVnPoint2_DINT [▶ 1640]&	返回值。

返回值

HRESULT [▶ 135]

6.2.2.1.2.9.2 *设置*

设置值。

语法

定义：

```
HRESULT Set(
    TcVnPoint2_DINT& aPoint
)
```

参数

名称	类型	描述
aPoint	TcVnPoint2_DINT [▶ 1640]&	要设置的值。

返回值

HRESULT [▶ 135]

6.2.2.1.2.10 ITcVnAccess_TcVnPoint2_LREAL

提供 TcVnPoint2_LREAL 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnPoint2_LREAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1733]	ITcVnAccess_TcVnPoint2_LREAL	获取值
Set [▶ 1734]	ITcVnAccess_TcVnPoint2_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.10.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnPoint2_LREAL& aPoint
)
```

参数

名称	类型	描述
aPoint	TcVnPoint2_LREAL [▶ 1640]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.10.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnPoint2_LREAL& aPoint
)
```

参数

名称	类型	描述
aPoint	TcVnPoint2_LREAL [▶ 1640]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.11 ITcVnAccess_TcVnPoint2_REAL

提供 TcVnPoint2_REAL 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnPoint2_REAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1734]	ITcVnAccess_TcVnPoint2_REAL	获取值
Set [▶ 1735]	ITcVnAccess_TcVnPoint2_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.11.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnPoint2_REAL& aPoint
)
```

参数

名称	类型	描述
aPoint	TcVnPoint2_REAL [▶ 1640]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.11.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnPoint2_REAL& aPoint
)
```

参数

名称	类型	描述
aPoint	TcVnPoint2_REAL [▶ 1640]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.12 ITcVnAccess_TcVnPoint3_LREAL

提供 TcVnPoint3_LREAL 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnPoint3_LREAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1736]	ITcVnAccess_TcVnPoint3_LREAL	获取值
Set [▶ 1736]	ITcVnAccess_TcVnPoint3_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.12.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnPoint3_LREAL& aPoint
)
```

参数

名称	类型	描述
aPoint	TcVnPoint3_LREAL [▶ 1640]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.12.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnPoint3_LREAL& aPoint
)
```

参数

名称	类型	描述
aPoint	TcVnPoint3_LREAL [▶ 1640]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.13 ITcVnAccess_TcVnPoint3_REAL

提供 TcVnPoint3_REAL 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnPoint3_REAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1737]	ITcVnAccess_TcVnPoint3_REAL	获取值
Set [▶ 1737]	ITcVnAccess_TcVnPoint3_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.13.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnPoint3_REAL& aPoint
)
```

参数

名称	类型	描述
aPoint	TcVnPoint3_REAL [▶ 1640]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.13.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnPoint3_REAL& aPoint
)
```

参数

名称	类型	描述
aPoint	TcVnPoint3_REAL [▶ 1640]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.14 ITcVnAccess_TcVnRectangle_DINT

提供 TcVnRectangle_DINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnRectangle_DINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1738]	ITcVnAccess_TcVnRectangle_DINT	获取值
Set [▶ 1738]	ITcVnAccess_TcVnRectangle_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.14.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnRectangle_DINT& stRectangle
)
```

参数

名称	类型	描述
stRectangle	TcVnRectangle_DINT [▶ 1717]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.14.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnRectangle_DINT& stRectangle
)
```

参数

名称	类型	描述
stRectangle	TcVnRectangle_DINT [▶ 1717]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.15 ITcVnAccess_TcVnVector2_DINT

提供 TcVnVector2_DINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector2_DINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1739]	ITcVnAccess_TcVnVector2_DINT	获取值
Set [▶ 1740]	ITcVnAccess_TcVnVector2_DINT	设置值。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.15.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector2_DINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector2_DINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.15.2 *设置*

设置值。

语法

定义：

```
HRESULT Set(
    TcVnVector2_DINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector2_DINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.16 ITcVnAccess_TcVnVector2_INT

提供 TcVnVector2_INT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector2_INT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1740]	ITcVnAccess_TcVnVector2_INT	获取值
Set [▶ 1741]	ITcVnAccess_TcVnVector2_INT	设置值。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.16.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector2_INT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector2_INT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.16.2 *设置*

设置值。

语法

定义：

```
HRESULT Set(
    TcVnVector2_INT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector2_INT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.17 ITcVnAccess_TcVnVector2_REAL

提供 TcVnVector2_REAL 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector2_REAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1742]	ITcVnAccess_TcVnVector2_REAL	获取值
Set [▶ 1742]	ITcVnAccess_TcVnVector2_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.17.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector2_REAL& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector2_REAL [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.17.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnVector2_REAL& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector2_REAL [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.18 ITcVnAccess_TcVnVector2_SINT

提供 TcVnVector2_SINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector2_SINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1743]	ITcVnAccess_TcVnVector2_SINT	获取值

名称	起源	描述
Set [▶ 1743]	ITcVnAccess_TcVnVector2_SINT	设置值。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.18.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector2_SINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector2_SINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.18.2 *设置*

设置值。

语法

定义：

```
HRESULT Set(
    TcVnVector2_SINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector2_SINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.19 ITcVnAccess_TcVnVector2_UINT

提供 TcVnVector2_UINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector2_UINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1744]	ITcVnAccess_TcVnVector2_UINT	获取值
Set [▶ 1744]	ITcVnAccess_TcVnVector2_UINT	设置值。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.19.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector2_UINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector2_UINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.19.2 *设置*

设置值。

语法

定义：

```
HRESULT Set(
    TcVnVector2_UINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector2_UINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.20 ITcVnAccess_TcVnVector2_USINT

提供 TcVnVector2_USINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector2_USINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1745]	ITcVnAccess_TcVnVector2_USINT	获取值
Set [▶ 1745]	ITcVnAccess_TcVnVector2_USINT	设置值。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.20.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector2_USINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector2_USINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.20.2 *设置*

设置值。

语法

定义：

```
HRESULT Set(
    TcVnVector2_USINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector2_USINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.21 ITcVnAccess_TcVnVector3_INT

提供 TcVnVector3_INT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector3_INT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1746]	ITcVnAccess_TcVnVector3_INT	获取值
Set [▶ 1747]	ITcVnAccess_TcVnVector3_INT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.21.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector3_INT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector3_INT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.21.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnVector3_INT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector3_INT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.22 ITcVnAccess_TcVnVector3_REAL

提供 TcVnVector3_REAL 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector3_REAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1747]	ITcVnAccess_TcVnVector3_REAL	获取值
Set [▶ 1748]	ITcVnAccess_TcVnVector3_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.22.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector3_REAL& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector3_REAL [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.22.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnVector3_REAL& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector3_REAL [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.23 ITcVnAccess_TcVnVector3_SINT

提供 TcVnVector3_SINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector3_SINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1749]	ITcVnAccess_TcVnVector3_SINT	获取值
Set [▶ 1749]	ITcVnAccess_TcVnVector3_SINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.23.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector3_SINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector3_SINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.23.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnVector3_SINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector3_SINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.24 ITcVnAccess_TcVnVector3_UINT

提供 TcVnVector3_UINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector3_UINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1750]	ITcVnAccess_TcVnVector3_UINT	获取值
Set [▶ 1750]	ITcVnAccess_TcVnVector3_UINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.24.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector3_UINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector3_UINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.24.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnVector3_UINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector3_UINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.25 ITcVnAccess_TcVnVector3_USINT

提供 TcVnVector3_USINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector3_USINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1751]	ITcVnAccess_TcVnVector3_USINT	获取值
Set [▶ 1751]	ITcVnAccess_TcVnVector3_USINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.25.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector3_USINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector3_USINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.25.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnVector3_USINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector3_USINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.26 ITcVnAccess_TcVnVector4_DINT

提供 TcVnVector4_DINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector4_DINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1752]	ITcVnAccess_TcVnVector4_DINT	获取值
Set [▶ 1753]	ITcVnAccess_TcVnVector4_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.26.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector4_DINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector4_DINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.26.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnVector4_DINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector4_DINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.27 ITcVnAccess_TcVnVector4_INT

提供 TcVnVector4_INT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector4_INT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1753]	ITcVnAccess_TcVnVector4_INT	获取值
Set [▶ 1754]	ITcVnAccess_TcVnVector4_INT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.27.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector4_INT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector4_INT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.27.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnVector4_INT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector4_INT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.28 ITcVnAccess_TcVnVector4_LREAL

提供 TcVnVector4_LREAL 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector4_LREAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1755]	ITcVnAccess_TcVnVector4_LREAL	获取值
Set [▶ 1755]	ITcVnAccess_TcVnVector4_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.28.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector4_LREAL& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector4_LREAL [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.28.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnVector4_LREAL& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector4_LREAL [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.29 ITcVnAccess_TcVnVector4_SINT

提供 TcVnVector4_SINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector4_SINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1756]	ITcVnAccess_TcVnVector4_SINT	获取值

名称	起源	描述
Set [▶ 1756]	ITcVnAccess_TcVnVector4_SINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.29.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector4_SINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector4_SINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.29.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnVector4_SINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector4_SINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.30 ITcVnAccess_TcVnVector4_UINT

提供 TcVnVector4_UINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector4_UINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1757]	ITcVnAccess_TcVnVector4_UINT	获取值
Set [▶ 1757]	ITcVnAccess_TcVnVector4_UINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.30.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector4_UINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector4_UINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.30.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnVector4_UINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector4_UINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.31 ITcVnAccess_TcVnVector4_USINT

提供 TcVnVector4_USINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector4_USINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1758]	ITcVnAccess_TcVnVector4_USINT	获取值
Set [▶ 1758]	ITcVnAccess_TcVnVector4_USINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.31.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector4_USINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector4_USINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.31.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnVector4_USINT& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector4_USINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.32 ITcVnAccess_UDINT

提供 UDINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_UDINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1759]	ITcVnAccess_UDINT	获取值
Set [▶ 1760]	ITcVnAccess_UDINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.32.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    ULONG& nValue
)
```

参数

名称	类型	描述
nValue	ULONG&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.32.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    ULONG nValue
)
```

参数

名称	类型	描述
nValue	ULONG	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.33 ITcVnAccess_UINT

提供 UINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_UINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1760]	ITcVnAccess_UINT	获取值
Set [▶ 1761]	ITcVnAccess_UINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.33.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    USHORT& nValue
)
```

参数

名称	类型	描述
nValue	USHORT&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.33.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    USHORT nValue
)
```

参数

名称	类型	描述
nValue	USHORT	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.34 ITcVnAccess_ULINT

提供 ULINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_ULINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1762]	ITcVnAccess_ULINT	获取值
Set [▶ 1762]	ITcVnAccess_ULINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.34.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    ULONGLONG& nValue
)
```

参数

名称	类型	描述
nValue	ULONGLONG&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.34.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    ULONGLONG nValue
)
```

参数

名称	类型	描述
nValue	ULONGLONG	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.35 ITcVnAccess_USINT

提供 USINT 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_USINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1763]	ITcVnAccess_USINT	获取值

名称	起源	描述
Set [▶ 1763]	ITcVnAccess_USINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.35.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    unsigned char& nValue
)
```

参数

名称	类型	描述
nValue	unsigned char&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.35.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    unsigned char nValue
)
```

参数

名称	类型	描述
nValue	unsigned char	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.36 ITcVnAccess_TcVnVector4_REAL

提供 TcVnVector4_REAL 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector4_REAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1764]	ITcVnAccess_TcVnVector4_REAL	获取值
Set [▶ 1764]	ITcVnAccess_TcVnVector4_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.36.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnVector4_REAL& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector4_REAL [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.36.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnVector4_REAL& aVector
)
```

参数

名称	类型	描述
aVector	TcVnVector4_REAL [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.2.37 ITcVnAccess_TcVnRotatedRectangle

Offers an access interface for TcVnRotatedRectangle values.

Inheritance Hierarchy

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnRotatedRectangle

Methods

Name	Origin	Description
TcAddRef [▶ 1877]	ITcUnknown	Increments the reference counter.
TcQueryInterface [▶ 1877]	ITcUnknown	Get a reference to an implemented interface.
TcRelease [▶ 1878]	ITcUnknown	Decrements the reference counter.
Get [▶ 1765]	ITcVnAccess_TcVnRotatedRectangle	Gets the value
Set [▶ 1765]	ITcVnAccess_TcVnRotatedRectangle	Sets the value

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.37.1 *Get*

Gets the value

Syntax

Definition:

```
HRESULT Get(
    TcVnRotatedRectangle& stRectangle
)
```

Parameters

Name	Type	Description
stRectangle	TcVnRotatedRectangle [▶ 1718]&	Returns the value

Return value

HRESULT [▶ 135]

6.2.2.1.2.37.2 *Set*

Sets the value

Syntax

Definition:

```
HRESULT Set(
    TcVnRotatedRectangle& stRectangle
)
```

Parameters

Name	Type	Description
stRectangle	TcVnRotatedRectangle [▶ 1718]&	The value to set

Return value

HRESULT [▶ 135]

6.2.2.1.2.38 ITcVnAccess_TcVnRotatedRectangle

提供 TcVnRotatedRectangle 值的访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnRotatedRectangle

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1766]	ITcVnAccess_TcVnRotatedRectangle	获取值
Set [▶ 1767]	ITcVnAccess_TcVnRotatedRectangle	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.2.38.1 *获取*

获取值

语法

定义：

```
HRESULT Get(
    TcVnRotatedRectangle& stRectangle
)
```

参数

名称	类型	描述
stRectangle	TcVnRotatedRectangle [▶ 1718]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.2.38.2 *设置*

设置值

语法

定义：

```
HRESULT Set(
    TcVnRotatedRectangle& stRectangle
)
```

参数

名称	类型	描述
stRectangle	TcVnRotatedRectangle [▶ 1718]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3 ITcVnRandomAccess

接口ITcVnRandomAccess用于访问容器的任何元素 [▶ 146]，并由ITcVnRandomAccessIterator [▶ 1824]提供。与ITcVnAccess [▶ 1722]不同，容器中的任何元素都可以被寻址。

6.2.2.1.3.1 ITcVnRandomAccess_DINT

提供 DINT 值的随机访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_DINT [▶ 1722]
    ITcVnRandomAccess_DINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1723]	ITcVnAccess_DINT	获取值
Set [▶ 1723]	ITcVnAccess_DINT	设置值
GetAt [▶ 1768]	ITcVnRandomAccess_DINT	获取值
SetAt [▶ 1768]	ITcVnRandomAccess_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.1.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG nOffset,
    LONG&    nValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
nValue	LONG&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.1.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG nOffset,
    LONG     nValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
nValue	LONG	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.2 ITcVnRandomAccess_INT

提供 INT 值的随机访问接口

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_INT [▶ 1723]
    ITcVnRandomAccess_INT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。

名称	起源	描述
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1724]	ITcVnAccess_INT	获取值
Set [▶ 1724]	ITcVnAccess_INT	设置值
GetAt [▶ 1769]	ITcVnRandomAccess_INT	获取值
SetAt [▶ 1769]	ITcVnRandomAccess_INT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.2.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG nOffset,
    SHORT&   nValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
nValue	SHORT&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.2.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG nOffset,
    SHORT    nValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
nValue	SHORT	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.3 ITcVnRandomAccess_ITcVnImage

提供图像的随机访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_ITcVnImage [▶ 1725]
    ITcVnRandomAccess_ITcVnImage

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1725]	ITcVnAccess_ITcVnImage	获取图像。
Set [▶ 1725]	ITcVnAccess_ITcVnImage	设置图像。
GetAt [▶ 1770]	ITcVnRandomAccess_ITcVnImage	获取图像。
SetAt [▶ 1771]	ITcVnRandomAccess_ITcVnImage	设置图像。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.3.1 *GetAt*

获取图像。

语法

定义：

```
HRESULT GetAt(
    LONGLONG     nOffset,
    ITcVnImage*& ipImage
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量。
ipImage	ITcVnImage* [▶ 1863]&	返回图像。

返回值

HRESULT [▶ 135]

6.2.2.1.3.3.2 *SetAt*

设置图像。

语法

定义：

```
HRESULT SetAt(
    LONGLONG    nOffset,
    ITcVnImage* ipImage
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
ipImage	ITcVnImage* [▶ 1863]	要设置的图像。

返回值

HRESULT [▶ 135]

6.2.2.1.3.4 ITcVnRandomAccess_LREAL

提供 LREAL 值的随机访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_LREAL [▶ 1726]
    ITcVnRandomAccess_LREAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1726]	ITcVnAccess_LREAL	获取值
Set [▶ 1727]	ITcVnAccess_LREAL	设置值
GetAt [▶ 1772]	ITcVnRandomAccess_LREAL	获取值
SetAt [▶ 1772]	ITcVnRandomAccess_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.4.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG nOffset,
    double&  fValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
fValue	double&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.4.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG nOffset,
    double   fValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量。
fValue	double	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.5 ITcVnRandomAccess_REAL

提供 REAL 值的随机访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_REAL [▶ 1727]
    ITcVnRandomAccess_REAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。

名称	起源	描述
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1727]	ITcVnAccess_REAL	获取值
Set [▶ 1728]	ITcVnAccess_REAL	设置值
GetAt [▶ 1773]	ITcVnRandomAccess_REAL	获取值
SetAt [▶ 1773]	ITcVnRandomAccess_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.5.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG nOffset,
    float&   fValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
fValue	float&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.5.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG nOffset,
    float    fValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
fValue	float	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.6 ITcVnRandomAccess_SINT

提供 SINT 值的随机访问接口

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_SINT [▶ 1728]
    ITcVnRandomAccess_SINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1729]	ITcVnAccess_SINT	获取值
Set [▶ 1729]	ITcVnAccess_SINT	设置值
GetAt [▶ 1774]	ITcVnRandomAccess_SINT	获取值
SetAt [▶ 1775]	ITcVnRandomAccess_SINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.6.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG nOffset,
    char&    nValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
nValue	char&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.6.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG nOffset,
    char     nValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
nValue	char	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.7 ITcVnRandomAccess_TcVnDMatch

提供TcVnDMatch 值的随机访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnDMatch [▶ 1729]
    ITcVnRandomAccess_TcVnDMatch

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1730]	ITcVnAccess_TcVnDMatch	获取值
Set [▶ 1730]	ITcVnAccess_TcVnDMatch	设置值
GetAt [▶ 1776]	ITcVnRandomAccess_TcVnDMatch	获取值
SetAt [▶ 1776]	ITcVnRandomAccess_TcVnDMatch	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.7.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG    nOffset,
    TcVnDMatch& stDMatch
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量。
stDMatch	TcVnDMatch [▶ 1705]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.7.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG    nOffset,
    TcVnDMatch& stDMatch
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
stDMatch	TcVnDMatch [▶ 1705]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.8 ITcVnRandomAccess_TcVnKeyPoint

提供 TcVnKeyPoint 值的随机访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnKeyPoint [▶ 1730]
    ITcVnRandomAccess_TcVnKeyPoint

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。

名称	起源	描述
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1731]	ITcVnAccess_TcVnKeyPoint	获取值。
Set [▶ 1731]	ITcVnAccess_TcVnKeyPoint	设置值。
GetAt [▶ 1777]	ITcVnRandomAccess_TcVnKeyPoint	获取值
SetAt [▶ 1777]	ITcVnRandomAccess_TcVnKeyPoint	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.8.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG      nOffset,
    TcVnKeyPoint& stKeyPoint
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
stKeyPoint	TcVnKeyPoint [▶ 1706]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.8.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG      nOffset,
    TcVnKeyPoint& stKeyPoint
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
stKeyPoint	TcVnKeyPoint [▶ 1706]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.9 ITcVnRandomAccess_TcVnPoint2_DINT

提供 TcVnPoint2_DINT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnPoint2_DINT [▶ 1732]
    ITcVnRandomAccess_TcVnPoint2_DINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1732]	ITcVnAccess_TcVnPoint2_DINT	获取值。
Set [▶ 1732]	ITcVnAccess_TcVnPoint2_DINT	设置值。
GetAt [▶ 1778]	ITcVnRandomAccess_TcVnPoint2_DINT	获取值
SetAt [▶ 1779]	ITcVnRandomAccess_TcVnPoint2_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.9.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG         nOffset,
    TcVnPoint2_DINT& aPoint
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aPoint	TcVnPoint2_DINT [▶ 1640]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.9.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG         nOffset,
    TcVnPoint2_DINT& aPoint
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aPoint	TcVnPoint2_DINT [▶ 1640]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.10 ITcVnRandomAccess_TcVnPoint2_LREAL

提供 TcVnPoint2_LREAL 值的随机访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnPoint2_LREAL [▶ 1733]
    ITcVnRandomAccess_TcVnPoint2_LREAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1733]	ITcVnAccess_TcVnPoint2_LREAL	获取值
Set [▶ 1734]	ITcVnAccess_TcVnPoint2_LREAL	设置值
GetAt [▶ 1780]	ITcVnRandomAccess_TcVnPoint2_LREAL	获取值
SetAt [▶ 1780]	ITcVnRandomAccess_TcVnPoint2_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.10.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG          nOffset,
    TcVnPoint2_LREAL& aPoint
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aPoint	TcVnPoint2_LREAL [▶ 1640]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.10.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG          nOffset,
    TcVnPoint2_LREAL& aPoint
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aPoint	TcVnPoint2_LREAL [▶ 1640]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.11 ITcVnRandomAccess_TcVnPoint2_REAL

提供 TcVnPoint2_REAL 值的随机访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnPoint2_REAL [▶ 1734]
    ITcVnRandomAccess_TcVnPoint2_REAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1734]	ITcVnAccess_TcVnPoint2_REAL	获取值
Set [▶ 1735]	ITcVnAccess_TcVnPoint2_REAL	设置值
GetAt [▶ 1781]	ITcVnRandomAccess_TcVnPoint2_REAL	获取值
SetAt [▶ 1781]	ITcVnRandomAccess_TcVnPoint2_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.11.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG         nOffset,
    TcVnPoint2_REAL& aPoint
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aPoint	TcVnPoint2_REAL [▶ 1640]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.11.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG          nOffset,
    TcVnPoint2_REAL& aPoint
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aPoint	TcVnPoint2_REAL [▶ 1640]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.12 ITcVnRandomAccess_TcVnPoint3_LREAL

提供 TcVnPoint3_LREAL 值的随机访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnPoint3_LREAL [▶ 1735]
    ITcVnRandomAccess_TcVnPoint3_LREAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1736]	ITcVnAccess_TcVnPoint3_LREAL	获取值
Set [▶ 1736]	ITcVnAccess_TcVnPoint3_LREAL	设置值
GetAt [▶ 1782]	ITcVnRandomAccess_TcVnPoint3_LREAL	获取值
SetAt [▶ 1783]	ITcVnRandomAccess_TcVnPoint3_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.12.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG          nOffset,
    TcVnPoint3_LREAL& aPoint
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aPoint	TcVnPoint3_LREAL [▶ 1640]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.12.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG          nOffset,
    TcVnPoint3_LREAL& aPoint
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aPoint	TcVnPoint3_LREAL [▶ 1640]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.13 ITcVnRandomAccess_TcVnPoint3_REAL

提供 TcVnPoint3_REAL 值的随机访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnPoint3_REAL [▶ 1736]
    ITcVnRandomAccess_TcVnPoint3_REAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1737]	ITcVnAccess_TcVnPoint3_REAL	获取值
Set [▶ 1737]	ITcVnAccess_TcVnPoint3_REAL	设置值
GetAt [▶ 1784]	ITcVnRandomAccess_TcVnPoint3_REAL	获取值

名称	起源	描述
SetAt [▶ 1784]	ITcVnRandomAccess_TcVnPoint3_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.13.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG         nOffset,
    TcVnPoint3_REAL& aPoint
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aPoint	TcVnPoint3_REAL [▶ 1640]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.13.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG         nOffset,
    TcVnPoint3_REAL& aPoint
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aPoint	TcVnPoint3_REAL [▶ 1640]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.14 ITcVnRandomAccess_TcVnRectangle_DINT

提供 TcVnRectangle_DINT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnRectangle_DINT [▶ 1738]
    ITcVnRandomAccess_TcVnRectangle_DINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1738]	ITcVnAccess_TcVnRectangle_DINT	获取值
Set [▶ 1738]	ITcVnAccess_TcVnRectangle_DINT	设置值
GetAt [▶ 1785]	ITcVnRandomAccess_TcVnRectangle_DINT	获取值
SetAt [▶ 1785]	ITcVnRandomAccess_TcVnRectangle_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.14.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG            nOffset,
    TcVnRectangle_DINT& stRectangle
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
stRectangle	TcVnRectangle_DINT [▶ 1717]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.14.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG           nOffset,
    TcVnRectangle_DINT& stRectangle
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
stRectangle	TcVnRectangle_DINT [▶ 1717]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.15 ITcVnRandomAccess_TcVnVector2_DINT

提供 TcVnVector2_DINT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector2_DINT [▶ 1739]
    ITcVnRandomAccess_TcVnVector2_DINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1739]	ITcVnAccess_TcVnVector2_DINT	获取值
Set [▶ 1740]	ITcVnAccess_TcVnVector2_DINT	设置值。
GetAt [▶ 1786]	ITcVnRandomAccess_TcVnVector2_DINT	获取值
SetAt [▶ 1787]	ITcVnRandomAccess_TcVnVector2_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.15.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG           nOffset,
    TcVnVector2_DINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector2_DINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.15.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG          nOffset,
    TcVnVector2_DINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector2_DINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.16 ITcVnRandomAccess_TcVnVector2_INT

提供 TcVnVector2_INT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector2_INT [▶ 1740]
    ITcVnRandomAccess_TcVnVector2_INT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1740]	ITcVnAccess_TcVnVector2_INT	获取值
Set [▶ 1741]	ITcVnAccess_TcVnVector2_INT	设置值。
GetAt [▶ 1788]	ITcVnRandomAccess_TcVnVector2_INT	获取值

名称	起源	描述
SetAt [▶ 1788]	ITcVnRandomAccess_TcVnVector2_INT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.16.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG         nOffset,
    TcVnVector2_INT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector2_INT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.16.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG         nOffset,
    TcVnVector2_INT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector2_INT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.17 ITcVnRandomAccess_TcVnVector2_REAL

提供 TcVnVector2_REAL 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector2_REAL [▶ 1741]
    ITcVnRandomAccess_TcVnVector2_REAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1742]	ITcVnAccess_TcVnVector2_REAL	获取值
Set [▶ 1742]	ITcVnAccess_TcVnVector2_REAL	设置值
GetAt [▶ 1789]	ITcVnRandomAccess_TcVnVector2_REAL	获取值
SetAt [▶ 1789]	ITcVnRandomAccess_TcVnVector2_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.17.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG          nOffset,
    TcVnVector2_REAL& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector2_REAL [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.17.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG          nOffset,
    TcVnVector2_REAL& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector2_REAL [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.18 ITcVnRandomAccess_TcVnVector2_SINT

提供 TcVnVector2_SINT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector2_SINT [▶ 1742]
    ITcVnRandomAccess_TcVnVector2_SINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1743]	ITcVnAccess_TcVnVector2_SINT	获取值
Set [▶ 1743]	ITcVnAccess_TcVnVector2_SINT	设置值。
GetAt [▶ 1790]	ITcVnRandomAccess_TcVnVector2_SINT	获取值
SetAt [▶ 1791]	ITcVnRandomAccess_TcVnVector2_SINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.18.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG          nOffset,
    TcVnVector2_SINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector2_SINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.18.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG          nOffset,
    TcVnVector2_SINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector2_SINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.19 ITcVnRandomAccess_TcVnVector2_UINT

提供 TcVnVector2_UINT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector2_UINT [▶ 1743]
    ITcVnRandomAccess_TcVnVector2_UINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1744]	ITcVnAccess_TcVnVector2_UINT	获取值
Set [▶ 1744]	ITcVnAccess_TcVnVector2_UINT	设置值。
GetAt [▶ 1792]	ITcVnRandomAccess_TcVnVector2_UINT	获取值

名称	起源	描述
SetAt [▶ 1792]	ITcVnRandomAccess_TcVnVector2_UINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.19.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG          nOffset,
    TcVnVector2_UINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector2_UINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.19.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG          nOffset,
    TcVnVector2_UINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector2_UINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.20 ITcVnRandomAccess_TcVnVector2_USINT

提供 TcVnVector2_USINT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector2_USINT [▶ 1745]
    ITcVnRandomAccess_TcVnVector2_USINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1745]	ITcVnAccess_TcVnVector2_USINT	获取值
Set [▶ 1745]	ITcVnAccess_TcVnVector2_USINT	设置值。
GetAt [▶ 1793]	ITcVnRandomAccess_TcVnVector2_USINT	获取值
SetAt [▶ 1793]	ITcVnRandomAccess_TcVnVector2_USINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.20.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG           nOffset,
    TcVnVector2_USINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector2_USINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.20.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG          nOffset,
    TcVnVector2_USINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector2_USINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.21 ITcVnRandomAccess_TcVnVector3_INT

提供 TcVnVector3_INT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector3_INT [▶ 1746]
    ITcVnRandomAccess_TcVnVector3_INT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1746]	ITcVnAccess_TcVnVector3_INT	获取值
Set [▶ 1747]	ITcVnAccess_TcVnVector3_INT	设置值
GetAt [▶ 1794]	ITcVnRandomAccess_TcVnVector3_INT	获取值
SetAt [▶ 1795]	ITcVnRandomAccess_TcVnVector3_INT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.21.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG          nOffset,
    TcVnVector3_INT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector3_INT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.21.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG         nOffset,
    TcVnVector3_INT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector3_INT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.22 ITcVnRandomAccess_TcVnVector3_REAL

提供 TcVnVector3_REAL 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector3_REAL [▶ 1747]
    ITcVnRandomAccess_TcVnVector3_REAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1747]	ITcVnAccess_TcVnVector3_REAL	获取值
Set [▶ 1748]	ITcVnAccess_TcVnVector3_REAL	设置值
GetAt [▶ 1796]	ITcVnRandomAccess_TcVnVector3_REAL	获取值

名称	起源	描述
SetAt [▶ 1796]	ITcVnRandomAccess_TcVnVector3_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.22.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG          nOffset,
    TcVnVector3_REAL& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector3_REAL [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.22.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG          nOffset,
    TcVnVector3_REAL& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector3_REAL [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.23 ITcVnRandomAccess_TcVnVector3_SINT

提供 TcVnVector3_SINT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector3_SINT [▶ 1748]
    ITcVnRandomAccess_TcVnVector3_SINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1749]	ITcVnAccess_TcVnVector3_SINT	获取值
Set [▶ 1749]	ITcVnAccess_TcVnVector3_SINT	设置值
GetAt [▶ 1797]	ITcVnRandomAccess_TcVnVector3_SINT	获取值
SetAt [▶ 1797]	ITcVnRandomAccess_TcVnVector3_SINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.23.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG          nOffset,
    TcVnVector3_SINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector3_SINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.23.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG          nOffset,
    TcVnVector3_SINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector3_SINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.24 ITcVnRandomAccess_TcVnVector3_UINT

提供 TcVnVector3_UINT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector3_UINT [▶ 1749]
    ITcVnRandomAccess_TcVnVector3_UINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1750]	ITcVnAccess_TcVnVector3_UINT	获取值
Set [▶ 1750]	ITcVnAccess_TcVnVector3_UINT	设置值
GetAt [▶ 1798]	ITcVnRandomAccess_TcVnVector3_UINT	获取值
SetAt [▶ 1799]	ITcVnRandomAccess_TcVnVector3_UINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.24.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG          nOffset,
    TcVnVector3_UINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector3_UINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.24.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG          nOffset,
    TcVnVector3_UINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector3_UINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.25 ITcVnRandomAccess_TcVnVector3_USINT

提供 TcVnVector3_USINT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector3_USINT [▶ 1751]
    ITcVnRandomAccess_TcVnVector3_USINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1751]	ITcVnAccess_TcVnVector3_USINT	获取值
Set [▶ 1751]	ITcVnAccess_TcVnVector3_USINT	设置值
GetAt [▶ 1800]	ITcVnRandomAccess_TcVnVector3_USINT	获取值

名称	起源	描述
SetAt [▶ 1800]	ITcVnRandomAccess_TcVnVector3_USINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.25.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG           nOffset,
    TcVnVector3_USINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector3_USINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.25.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG           nOffset,
    TcVnVector3_USINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector3_USINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.26 ITcVnRandomAccess_TcVnVector4_DINT

提供 TcVnVector4_DINT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector4_DINT [▶ 1752]
    ITcVnRandomAccess_TcVnVector4_DINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1752]	ITcVnAccess_TcVnVector4_DINT	获取值
Set [▶ 1753]	ITcVnAccess_TcVnVector4_DINT	设置值
GetAt [▶ 1801]	ITcVnRandomAccess_TcVnVector4_DINT	获取值
SetAt [▶ 1801]	ITcVnRandomAccess_TcVnVector4_DINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.26.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG          nOffset,
    TcVnVector4_DINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector4_DINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.26.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG          nOffset,
    TcVnVector4_DINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector4_DINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.27 ITcVnRandomAccess_TcVnVector4_INT

提供 TcVnVector4_INT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector4_INT [▶ 1753]
    ITcVnRandomAccess_TcVnVector4_INT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1753]	ITcVnAccess_TcVnVector4_INT	获取值
Set [▶ 1754]	ITcVnAccess_TcVnVector4_INT	设置值
GetAt [▶ 1802]	ITcVnRandomAccess_TcVnVector4_INT	获取值
SetAt [▶ 1803]	ITcVnRandomAccess_TcVnVector4_INT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.27.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG          nOffset,
    TcVnVector4_INT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector4_INT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.27.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG         nOffset,
    TcVnVector4_INT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector4_INT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.28 ITcVnRandomAccess_TcVnVector4_LREAL

提供 TcVnVector4_LREAL 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector4_LREAL [▶ 1754]
    ITcVnRandomAccess_TcVnVector4_LREAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1755]	ITcVnAccess_TcVnVector4_LREAL	获取值
Set [▶ 1755]	ITcVnAccess_TcVnVector4_LREAL	设置值
GetAt [▶ 1804]	ITcVnRandomAccess_TcVnVector4_LREAL	获取值

名称	起源	描述
SetAt [▶ 1804]	ITcVnRandomAccess_TcVnVector4_LREAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.28.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG           nOffset,
    TcVnVector4_LREAL& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector4_LREAL [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.28.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG           nOffset,
    TcVnVector4_LREAL& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector4_LREAL [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.29 ITcVnRandomAccess_TcVnVector4_SINT

提供 TcVnVector4_SINT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector4_SINT [▶ 1755]
    ITcVnRandomAccess_TcVnVector4_SINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1756]	ITcVnAccess_TcVnVector4_SINT	获取值
Set [▶ 1756]	ITcVnAccess_TcVnVector4_SINT	设置值
GetAt [▶ 1805]	ITcVnRandomAccess_TcVnVector4_SINT	获取值
SetAt [▶ 1805]	ITcVnRandomAccess_TcVnVector4_SINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.29.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG          nOffset,
    TcVnVector4_SINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector4_SINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.29.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG          nOffset,
    TcVnVector4_SINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector4_SINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.30 ITcVnRandomAccess_TcVnVector4_UINT

提供 TcVnVector4_UINT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector4_UINT [▶ 1756]
    ITcVnRandomAccess_TcVnVector4_UINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1757]	ITcVnAccess_TcVnVector4_UINT	获取值
Set [▶ 1757]	ITcVnAccess_TcVnVector4_UINT	设置值
GetAt [▶ 1806]	ITcVnRandomAccess_TcVnVector4_UINT	获取值
SetAt [▶ 1807]	ITcVnRandomAccess_TcVnVector4_UINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.30.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG          nOffset,
    TcVnVector4_UINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector4_UINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.30.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG          nOffset,
    TcVnVector4_UINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector4_UINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.31 ITcVnRandomAccess_TcVnVector4_USINT

提供 TcVnVector4_USINT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector4_USINT [▶ 1758]
    ITcVnRandomAccess_TcVnVector4_USINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1758]	ITcVnAccess_TcVnVector4_USINT	获取值
Set [▶ 1758]	ITcVnAccess_TcVnVector4_USINT	设置值
GetAt [▶ 1808]	ITcVnRandomAccess_TcVnVector4_USINT	获取值

名称	起源	描述
SetAt [▶ 1808]	ITcVnRandomAccess_TcVnVector4_USINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.31.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG           nOffset,
    TcVnVector4_USINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector4_USINT [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.31.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG           nOffset,
    TcVnVector4_USINT& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector4_USINT [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.32 ITcVnRandomAccess_UDINT

提供 UDINT 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_UDINT [▶ 1759]
    ITcVnRandomAccess_UDINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1759]	ITcVnAccess_UDINT	获取值
Set [▶ 1760]	ITcVnAccess_UDINT	设置值
GetAt [▶ 1809]	ITcVnRandomAccess_UDINT	获取值
SetAt [▶ 1809]	ITcVnRandomAccess_UDINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.32.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG nOffset,
    ULONG&   nValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
nValue	ULONG&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.32.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG nOffset,
    ULONG    nValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
nValue	ULONG	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.33 ITcVnRandomAccess_UINT

提供 UINT 值的随机存取接口

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_UINT [▶ 1760]
    ITcVnRandomAccess_UINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1760]	ITcVnAccess_UINT	获取值
Set [▶ 1761]	ITcVnAccess_UINT	设置值
GetAt [▶ 1810]	ITcVnRandomAccess_UINT	获取值
SetAt [▶ 1811]	ITcVnRandomAccess_UINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.33.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG nOffset,
    USHORT&  nValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
nValue	USHORT&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.33.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG nOffset,
    USHORT   nValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
nValue	USHORT	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.34 ITcVnRandomAccess_ULINT

提供 ULINT 值的随机访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_ULINT [▶ 1761]
    ITcVnRandomAccess_ULINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1762]	ITcVnAccess_ULINT	获取值
Set [▶ 1762]	ITcVnAccess_ULINT	设置值
GetAt [▶ 1812]	ITcVnRandomAccess_ULINT	获取值
SetAt [▶ 1812]	ITcVnRandomAccess_ULINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.34.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG   nOffset,
    ULONGLONG& nValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
nValue	ULONGLONG&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.34.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG  nOffset,
    ULONGLONG nValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
nValue	ULONGLONG	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.35 ITcVnRandomAccess_USINT

提供 USINT 值的随机存取接口

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_USINT [▶ 1762]
    ITcVnRandomAccess_USINT

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1763]	ITcVnAccess_USINT	获取值
Set [▶ 1763]	ITcVnAccess_USINT	设置值
GetAt [▶ 1813]	ITcVnRandomAccess_USINT	获取值
SetAt [▶ 1813]	ITcVnRandomAccess_USINT	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.35.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG       nOffset,
    unsigned char& nValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
nValue	unsigned char&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.35.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG      nOffset,
    unsigned char nValue
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
nValue	unsigned char	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.36 ITcVnRandomAccess_TcVnVector4_REAL

提供 TcVnVector4_REAL 值的随机存取接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnVector4_REAL [▶ 1763]
    ITcVnRandomAccess_TcVnVector4_REAL

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1764]	ITcVnAccess_TcVnVector4_REAL	获取值
Set [▶ 1764]	ITcVnAccess_TcVnVector4_REAL	设置值
GetAt [▶ 1814]	ITcVnRandomAccess_TcVnVector4_REAL	获取值
SetAt [▶ 1815]	ITcVnRandomAccess_TcVnVector4_REAL	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.36.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG          nOffset,
    TcVnVector4_REAL& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector4_REAL [▶ 1641]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.36.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG          nOffset,
    TcVnVector4_REAL& aVector
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
aVector	TcVnVector4_REAL [▶ 1641]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.3.37 ITcVnRandomAccess_TcVnRotatedRectangle

Offers a random access interface for TcVnRotatedRectangle values.

Inheritance Hierarchy

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnRotatedRectangle [▶ 1765]
    ITcVnRandomAccess_TcVnRotatedRectangle

Methods

Name	Origin	Description
TcAddRef [▶ 1877]	ITcUnknown	Increments the reference counter.
TcQueryInterface [▶ 1877]	ITcUnknown	Get a reference to an implemented interface.
TcRelease [▶ 1878]	ITcUnknown	Decrements the reference counter.
Get [▶ 1765]	ITcVnAccess_TcVnRotatedRectangle	Gets the value
Set [▶ 1765]	ITcVnAccess_TcVnRotatedRectangle	Sets the value
GetAt [▶ 1816]	ITcVnRandomAccess_TcVnRotatedRectangle	Gets the value
SetAt [▶ 1816]	ITcVnRandomAccess_TcVnRotatedRectangle	Sets the value

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.37.1 *GetAt*

Gets the value

Syntax

Definition:

```
HRESULT GetAt(
    LONGLONG              nOffset,
    TcVnRotatedRectangle& stRectangle
)
```

Parameters

Name	Type	Description
nOffset	LONGLONG	Offset to the current position
stRectangle	TcVnRotatedRectangle [▶ 1718]&	Returns the value

Return value

HRESULT [▶ 135]

6.2.2.1.3.37.2 *SetAt*

Sets the value

Syntax

Definition:

```
HRESULT SetAt(
    LONGLONG              nOffset,
    TcVnRotatedRectangle& stRectangle
)
```

Parameters

Name	Type	Description
nOffset	LONGLONG	Offset to the current position
stRectangle	TcVnRotatedRectangle [▶ 1718]&	The value to set

Return value

HRESULT [▶ 135]

6.2.2.1.3.38 ITcVnRandomAccess_TcVnRotatedRectangle

提供 TcVnRotatedRectangle 值的随机访问接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnAccess_TcVnRotatedRectangle [▶ 1766]
    ITcVnRandomAccess_TcVnRotatedRectangle

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Get [▶ 1766]	ITcVnAccess_TcVnRotatedRectangle	获取值
Set [▶ 1767]	ITcVnAccess_TcVnRotatedRectangle	设置值
GetAt [▶ 1817]	ITcVnRandomAccess_TcVnRotatedRectangle	获取值
SetAt [▶ 1817]	ITcVnRandomAccess_TcVnRotatedRectangle	设置值

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.3.38.1 *GetAt*

获取值

语法

定义：

```
HRESULT GetAt(
    LONGLONG              nOffset,
    TcVnRotatedRectangle& stRectangle
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
stRectangle	TcVnRotatedRectangle [▶ 1718]&	返回值

返回值

HRESULT [▶ 135]

6.2.2.1.3.38.2 *SetAt*

设置值

语法

定义：

```
HRESULT SetAt(
    LONGLONG              nOffset,
    TcVnRotatedRectangle& stRectangle
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前位置的偏移量
stRectangle	TcVnRotatedRectangle [▶ 1718]&	要设置的值

返回值

HRESULT [▶ 135]

6.2.2.1.4 迭代器

该组包含迭代器处理的接口，以便访问容器 [▶ 146]的元素。

6.2.2.1.4.1 ITcVnBidirectionalIterator

提供双向迭代器的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnIteratorBase [▶ 1821]
    ITcVnForwardIterator [▶ 1819]
      ITcVnBidirectionalIterator

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
CheckIfBegin [▶ 1822]	ITcVnIteratorBase	检查迭代器是否指向第一个元素。
CheckIfEnd [▶ 1822]	ITcVnIteratorBase	检查迭代器是否指向最后一个元素的后一个位置（或者使用 F_VN_CheckIfIteratorIsAtEnd）。
GetValueSize [▶ 1822]	ITcVnIteratorBase	获取迭代器指向的值所需的内存大小（以字节为单位）。
GetValueTypeGuid [▶ 1822]	ITcVnIteratorBase	获取迭代器指向的值的 GUID 类型。
SetToBegin [▶ 1823]	ITcVnIteratorBase	将迭代器设置为第一个元素（或者使用 F_VN_SetIteratorToBegin）。
SetToEnd [▶ 1823]	ITcVnIteratorBase	将迭代器设置为最后一个元素的后一个位置。
CheckIfEqualTo [▶ 1820]	ITcVnForwardIterator	检查迭代器是否等于另一个迭代器。
GetContainer [▶ 1820]	ITcVnForwardIterator	获取转换为 ITcVnContainer 接口的当前元素的指针，并递增其引用计数器（仅适用于容器类型）。（或者使用 F_VN_GetContainer。）
Increment [▶ 1820]	ITcVnForwardIterator	递增迭代器。（或者使用 F_VN_IncrementIterator。）
SetContainer [▶ 1821]	ITcVnForwardIterator	使用 ITcVnContainer 接口设置当前元素（仅适用于容器类型）。（或者使用 F_VN_SetContainer。）
Decrement [▶ 1819]	ITcVnBidirectionalIterator	递减迭代器。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.4.1.1 *递减*

递减迭代器。

语法

定义：

```
HRESULT Decrement()
```

返回值

HRESULT [▶ 135]

6.2.2.1.4.2 ITcVnForwardIterator

为前向迭代器提供一个接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnIteratorBase [▶ 1821]
    ITcVnForwardIterator

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
CheckIfBegin [▶ 1822]	ITcVnIteratorBase	检查迭代器是否指向第一个元素。
CheckIfEnd [▶ 1822]	ITcVnIteratorBase	检查迭代器是否指向最后一个元素的后一个位置（或者使用 F_VN_CheckIfIteratorIsAtEnd）。
GetValueSize [▶ 1822]	ITcVnIteratorBase	获取迭代器指向的值所需的内存大小（以字节为单位）。
GetValueTypeGuid [▶ 1822]	ITcVnIteratorBase	获取迭代器指向的值的 GUID 类型。
SetToBegin [▶ 1823]	ITcVnIteratorBase	将迭代器设置为第一个元素（或者使用 F_VN_SetIteratorToBegin）。
SetToEnd [▶ 1823]	ITcVnIteratorBase	将迭代器设置为最后一个元素的后一个位置。
CheckIfEqualTo [▶ 1820]	ITcVnForwardIterator	检查迭代器是否等于另一个迭代器。
GetContainer [▶ 1820]	ITcVnForwardIterator	获取转换为 ITcVnContainer 接口的当前元素的指针，并递增其引用计数器（仅适用于容器类型）。（或者使用 F_VN_GetContainer。）
Increment [▶ 1820]	ITcVnForwardIterator	递增迭代器。（或者使用 F_VN_IncrementIterator。）
SetContainer [▶ 1821]	ITcVnForwardIterator	使用 ITcVnContainer 接口设置当前元素（仅适用于容器类型）。（或者使用 F_VN_SetContainer。）

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.4.2.1 *CheckIfEqualTo*

检查迭代器是否等于另一个迭代器。

语法

定义：

```
HRESULT CheckIfEqualTo(
    ITcVnForwardIterator* ipIterator
)
```

参数

名称	类型	描述
ipIterator	ITcVnForwardIterator* [▶ 1819]	进行比较的迭代器接口

返回值

HRESULT [▶ 135]

6.2.2.1.4.2.2 *GetContainer*

获取转换为 ITcVnContainer 接口的当前元素的指针，并递增其引用计数器（仅适用于容器类型）。（或者使用 F_VN_GetContainer。）

语法

定义：

```
HRESULT GetContainer(
    ITcVnContainer** pipContainer
)
```

参数

名称	类型	描述
pipContainer	ITcVnContainer* [▶ 1827]*	返回容器接口。

返回值

HRESULT [▶ 135]

6.2.2.1.4.2.3 *增量*

递增迭代器。（或者使用 F_VN_IncrementIterator。）

语法

定义：

```
HRESULT Increment()
```

返回值

HRESULT [▶ 135]

6.2.2.1.4.2.4 *SetContainer*

使用 ITcVnContainer 接口设置当前元素（仅适用于容器类型）。（或者使用 F_VN_SetContainer。）

语法

定义：

```
HRESULT SetContainer(
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
ipContainer	ITcVnContainer* [▶ 1827]	将其内容分配给当前元素的容器接口。

返回值

HRESULT [▶ 135]

6.2.2.1.4.3 ITcVnIteratorBase

提供迭代器的基本接口

继承层次

ITcUnknown [▶ 1877]
  ITcVnIteratorBase

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
CheckIfBegin [▶ 1822]	ITcVnIteratorBase	检查迭代器是否指向第一个元素。
CheckIfEnd [▶ 1822]	ITcVnIteratorBase	检查迭代器是否指向最后一个元素的后一个位置（或者使用 F_VN_CheckIfIteratorIsAtEnd）。
GetValueSize [▶ 1822]	ITcVnIteratorBase	获取迭代器指向的值所需的内存大小（以字节为单位）。
GetValueTypeGuid [▶ 1822]	ITcVnIteratorBase	获取迭代器指向的值的 GUID 类型。
SetToBegin [▶ 1823]	ITcVnIteratorBase	将迭代器设置为第一个元素（或者使用 F_VN_SetIteratorToBegin）。
SetToEnd [▶ 1823]	ITcVnIteratorBase	将迭代器设置为最后一个元素的后一个位置。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.4.3.1 *CheckIfBegin*

检查迭代器是否指向第一个元素。

语法

定义：

```
HRESULT CheckIfBegin()
```

返回值

HRESULT [▶ 135]

6.2.2.1.4.3.2 *CheckIfEnd*

检查迭代器是否指向最后一个元素的后一个位置（或者使用 F_VN_CheckIfIteratorIsAtEnd）。

语法

定义：

```
HRESULT CheckIfEnd()
```

返回值

HRESULT [▶ 135]

6.2.2.1.4.3.3 *GetValueSize*

获取迭代器指向的值所需的内存大小（以字节为单位）。

语法

定义：

```
HRESULT GetValueSize(
    ULONGLONG& nSize
)
```

参数

名称	类型	描述
nSize	ULONGLONG&	返回以字节为单位的元素大小。

返回值

HRESULT [▶ 135]

6.2.2.1.4.3.4 *GetValueTypeGuid*

获取迭代器指向的值的 GUID 类型。

语法

定义：

```
HRESULT GetValueTypeGuid(
    GUID& nTypeGuid
)
```

参数

名称	类型	描述
nTypeGuid	GUID&	返回 GUID 类型。

返回值

HRESULT [▶ 135]

6.2.2.1.4.3.5 *SetToBegin*

将迭代器设置为第一个元素（或者使用 F_VN_SetIteratorToBegin）。

语法

定义：

```
HRESULT SetToBegin()
```

返回值

HRESULT [▶ 135]

6.2.2.1.4.3.6 *SetToEnd*

将迭代器设置为最后一个元素的后一个位置。

语法

定义：

```
HRESULT SetToEnd()
```

返回值

HRESULT [▶ 135]

6.2.2.1.4.4 ITcVnIteratorCopyCreator

提供可以在相同位置创建新迭代器的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnIteratorCopyCreator

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。

名称	起源	描述
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
Create [▶ 1824]	ITcVnIteratorCopyCreator	创建一个新迭代器，指向调用迭代器的位置。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.4.4.1 *创建*

创建一个新迭代器，指向调用迭代器的位置。

语法

定义：

```
HRESULT Create(
    ITcVnForwardIterator** pipIterator
)
```

参数

名称	类型	描述
pipIterator	ITcVnForwardIterator* [▶ 1819]*	返回创建的迭代器。

返回值

HRESULT [▶ 135]

6.2.2.1.4.5 ITcVnRandomAccessIterator

提供随机访问迭代器的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnIteratorBase [▶ 1821]
    ITcVnForwardIterator [▶ 1819]
      ITcVnBidirectionalIterator [▶ 1818]
        ITcVnRandomAccessIterator

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
CheckIfBegin [▶ 1822]	ITcVnIteratorBase	检查迭代器是否指向第一个元素。
CheckIfEnd [▶ 1822]	ITcVnIteratorBase	检查迭代器是否指向最后一个元素的后一个位置（或者使用 F_VN_CheckIfIteratorIsAtEnd）。
GetValueSize [▶ 1822]	ITcVnIteratorBase	获取迭代器指向的值所需的内存大小（以字节为单位）。

名称	起源	描述
GetValueTypeGuid [▶ 1822]	ITcVnIteratorBase	获取迭代器指向的值的 GUID 类型。
SetToBegin [▶ 1823]	ITcVnIteratorBase	将迭代器设置为第一个元素（或者使用 F_VN_SetIteratorToBegin）。
SetToEnd [▶ 1823]	ITcVnIteratorBase	将迭代器设置为最后一个元素的后一个位置。
CheckIfEqualTo [▶ 1820]	ITcVnForwardIterator	检查迭代器是否等于另一个迭代器。
GetContainer [▶ 1820]	ITcVnForwardIterator	获取转换为 ITcVnContainer 接口的当前元素的指针，并递增其引用计数器（仅适用于容器类型）。（或者使用 F_VN_GetContainer。）
Increment [▶ 1820]	ITcVnForwardIterator	递增迭代器。（或者使用 F_VN_IncrementIterator。）
SetContainer [▶ 1821]	ITcVnForwardIterator	使用 ITcVnContainer 接口设置当前元素（仅适用于容器类型）。（或者使用 F_VN_SetContainer。）
Decrement [▶ 1819]	ITcVnBidirectionalIterator	递减迭代器。
CheckIfGreaterThan [▶ 1825]	ITcVnRandomAccessIterator	检查该迭代器是否大于另一个迭代器。
CheckIfLessThan [▶ 1826]	ITcVnRandomAccessIterator	检查该迭代器是否小于另一个迭代器。
GetContainerAt [▶ 1826]	ITcVnRandomAccessIterator	获取一个转换为 ITcVnContainer 接口的指针，该指针指向在当前元素特定偏移位置的元素，并递增其引用计数器（仅适用于容器类型）。（或者使用 F_VN_GetAt_ITcVnContainer。）
SetContainerAt [▶ 1826]	ITcVnRandomAccessIterator	使用 ITcVnContainer 接口（仅适用于容器类型）在当前元素的特定偏移位置设置元素。（或者使用 F_VN_SetAt_ITcVnContainer。）

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.4.5.1 *CheckIfGreaterThan*

检查该迭代器是否大于另一个迭代器。

语法

定义：

```
HRESULT CheckIfGreaterThan(
    ITcVnRandomAccessIterator* ipIterator
)
```

参数

名称	类型	描述
ipIterator	ITcVnRandomAccessIterator* [▶ 1824]	进行比较的迭代器接口。

返回值

HRESULT [▶ 135]

6.2.2.1.4.5.2 *CheckIfLessThan*

检查该迭代器是否小于另一个迭代器。

语法

定义：

```
HRESULT CheckIfLessThan(
    ITcVnRandomAccessIterator* ipIterator
)
```

参数

名称	类型	描述
ipIterator	ITcVnRandomAccessIterator* [▶ 1824]	进行比较的迭代器接口。

返回值

HRESULT [▶ 135]

6.2.2.1.4.5.3 *GetContainerAt*

获取一个转换为 ITcVnContainer 接口的指针，该指针指向在当前元素特定偏移位置的元素，并递增其引用计数器（仅适用于容器类型）。（或者使用 F_VN_GetAt_ITcVnContainer。）

语法

定义：

```
HRESULT GetContainerAt(
    LONGLONG          nOffset,
    ITcVnContainer** pipContainer
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前元素的偏移量。
pipContainer	ITcVnContainer* [▶ 1827]*	返回指向容器接口的指针。

返回值

HRESULT [▶ 135]

6.2.2.1.4.5.4 *SetContainerAt*

使用 ITcVnContainer 接口（仅适用于容器类型）在当前元素的特定偏移位置设置元素。（或者使用 F_VN_SetAt_ITcVnContainer。）

语法

定义：

```
HRESULT SetContainerAt(
    LONGLONG         nOffset,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
nOffset	LONGLONG	与当前元素的偏移量。
ipContainer	ITcVnContainer* [▶ 1827]	将其内容分配给当前元素的容器接口。

返回值

HRESULT [▶ 135]

6.2.2.1.5 ITcVnContainer

提供用于对象容器的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnContainer

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
CheckIfBasicContainer [▶ 1828]	ITcVnContainer	检查容器是否只包含基本（非容器）元素。
CheckIfEmpty [▶ 1828]	ITcVnContainer	检查容器是否为空。（或者使用 F_VN_CheckIfEmpty。）
GetBidirectionalIterator [▶ 1828]	ITcVnContainer	获取双向迭代器的接口指针（如果容器支持这种迭代器类型），并递增其引用计数器。
GetElementNum [▶ 1828]	ITcVnContainer	获取容器的大小（元素数量）。（或者使用 F_VN_GetNumberOfElements。）
GetElementSize [▶ 1829]	ITcVnContainer	获取容器中每个元素的大小（以字节为单位）。
GetElementTypeGuid [▶ 1829]	ITcVnContainer	获取容器元素的 GUID。
GetExportSize [▶ 1829]	ITcVnContainer	获取容器中所有元素的合并大小（以字节为单位）。
GetForwardIterator [▶ 1830]	ITcVnContainer	获取指向前向迭代器的接口指针（如果容器支持该迭代器类型），并递增其引用计数器。（或者使用 F_VN_GetForwardIterator。）
GetRandomAccessIterator [▶ 1830]	ITcVnContainer	获取指向随机访问迭代器的接口指针（如果容器支持该迭代器类型），并递增其引用计数器。（或者使用 F_VN_GetRandomAccessIterator。）
GetTypeGuid [▶ 1831]	ITcVnContainer	获取容器的 GUID。
GetTypeName [▶ 1831]	ITcVnContainer	获取字符串形式的容器类型名称。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.1.5.1 CheckIfBasicContainer

检查容器是否只包含基本（非容器）元素。

语法

定义：

```
HRESULT CheckIfBasicContainer()
```

返回值

HRESULT [▶ 135]

6.2.2.1.5.2 CheckIfEmpty

检查容器是否为空。（或者使用 F_VN_CheckIfEmpty。）

语法

定义：

```
HRESULT CheckIfEmpty()
```

返回值

HRESULT [▶ 135]

6.2.2.1.5.3 GetBidirectionalIterator

获取双向迭代器的接口指针（如果容器支持这种迭代器类型），并递增其引用计数器。

语法

定义：

```
HRESULT GetBidirectionalIterator(
    ITcVnBidirectionalIterator** pipIterator
)
```

参数

名称	类型	描述
pipIterator	ITcVnBidirectionalIterator* [▶ 1818]*	返回迭代器接口。

返回值

HRESULT [▶ 135]

6.2.2.1.5.4 GetElementNum

获取容器的大小（元素数量）。（或者使用 F_VN_GetNumberOfElements。）

语法

定义：

```
HRESULT GetElementNum(
    ULONGLONG& nElementNum
)
```

参数

名称	类型	描述
nElementNum	ULONGLONG&	返回容器中元素的个数。

返回值

HRESULT [▶ 135]

6.2.2.1.5.5 GetElementSize

获取容器中每个元素的大小（以字节为单位）。

语法

定义：

```
HRESULT GetElementSize(
    ULONGLONG& nSize
)
```

参数

名称	类型	描述
nSize	ULONGLONG&	返回容器中单个元素的大小（以字节为单位）。

返回值

HRESULT [▶ 135]

6.2.2.1.5.6 GetElementTypeGuid

获取容器元素的 GUID。

语法

定义：

```
HRESULT GetElementTypeGuid(
    GUID& nTypeGuid
)
```

参数

名称	类型	描述
nTypeGuid	GUID&	返回容器元素的 GUID。

返回值

HRESULT [▶ 135]

6.2.2.1.5.7 GetExportSize

获取容器中所有元素的合并大小（以字节为单位）。

语法

定义：

```
HRESULT GetExportSize(
    ULONGLONG& nExportSize
)
```

参数

名称	类型	描述
nExportSize	ULONGLONG&	返回容器中所有元素的总大小（以字节为单位）。

返回值

HRESULT [▶ 135]

6.2.2.1.5.8 GetForwardIterator

获取指向前向迭代器的接口指针（如果容器支持该迭代器类型），并递增其引用计数器。（或者使用 F_VN_GetForwardIterator。）

语法

定义：

```
HRESULT GetForwardIterator(
    ITcVnForwardIterator** pipIterator
)
```

参数

名称	类型	描述
pipIterator	ITcVnForwardIterator* [▶ 1819]*	返回迭代器接口。

返回值

HRESULT [▶ 135]

6.2.2.1.5.9 GetRandomAccessIterator

获取指向随机访问迭代器的接口指针（如果容器支持该迭代器类型），并递增其引用计数器。（或者使用 F_VN_GetRandomAccessIterator。）

语法

定义：

```
HRESULT GetRandomAccessIterator(
    ITcVnRandomAccessIterator** pipIterator
)
```

参数

名称	类型	描述
pipIterator	ITcVnRandomAccessIterator* [▶ 1824]*	返回迭代器接口。

返回值

HRESULT [▶ 135]

6.2.2.1.5.10 GetTypeGuid

获取容器的 GUID。

语法

定义：

```
HRESULT GetTypeGuid(
    GUID& nTypeGuid
)
```

参数

名称	类型	描述
nTypeGuid	GUID&	返回容器的 GUID。

返回值

HRESULT [▶ 135]

6.2.2.1.5.11 GetTypeName

获取字符串形式的容器类型名称。

语法

定义：

```
HRESULT GetTypeName(
    PCHAR  sTypeName,
    USHORT nMaxLen
)
```

参数

名称	类型	描述
sTypeName	PCHAR	以字符串形式返回容器类型名称。
nMaxLen	USHORT	允许写入 sTypeName 的最大字符串长度

返回值

HRESULT [▶ 135]

6.2.2.2 图像

该组包含用于记录和处理图像 [▶ 143]的接口。

6.2.2.2.1 采集

该组包含用于处理图像采集的接口。

6.2.2.2.1.1 ITcIoFileImageAcquisition

文件图像采集接口。

继承层次

ITcUnknown [▶ 1877]
  ITcIoImageAcquisition [▶ 1835]
    ITcIoFileImageAcquisition

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
RegisterReceiver [▶ 1836]	ITcIoImageAcquisition	注册回调函数接口。
UnregisterReceiver [▶ 1836]	ITcIoImageAcquisition	取消注册回调函数接口。
OpenCamera [▶ 1836]	ITcIoImageAcquisition	打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。
CloseCamera [▶ 1837]	ITcIoImageAcquisition	关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。
StartAcquisition [▶ 1837]	ITcIoImageAcquisition	开始图像采集。
StopAcquisition [▶ 1837]	ITcIoImageAcquisition	停止图像采集。
SoftwareTrigger [▶ 1837]	ITcIoImageAcquisition	初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。
InitializeCamera [▶ 1838]	ITcIoImageAcquisition	向相机发送初始化命令。
CheckConnection [▶ 1832]	ITcIoFileImageAcquisition	检查连接
TriggerImage [▶ 1832]	ITcIoFileImageAcquisition	初始化软件触发器并触发单个图像。跳过 nSkipImages 并触发采集图像 nSkipImages+1。
TriggerImageByName [▶ 1833]	ITcIoFileImageAcquisition	初始化软件触发器，并触发客户端助手中按名称指定的单个图像。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.1.1.1 *CheckConnection*

检查连接

语法

定义：

```
HRESULT CheckConnection()
```

返回值

HRESULT [▶ 135]

6.2.2.2.1.1.2 *TriggerImage*

初始化软件触发器并触发单个图像。跳过 nSkipImages 并触发采集图像 nSkipImages+1。

语法

定义：

```
HRESULT TriggerImage(
    LONG nSkipImages
)
```

参数

名称	类型	描述
nSkipImages	LONG	要跳过的图像数量

返回值

HRESULT [▶ 135]

6.2.2.2.1.1.3 *TriggerImageByName*

初始化软件触发器，并触发客户端助手中按名称指定的单个图像。

语法

定义：

```
HRESULT TriggerImageByName(
    PCCH sImageName
)
```

参数

名称	类型	描述
sImageName	PCCH	要触发的图像名称

返回值

HRESULT [▶ 135]

6.2.2.2.1.2 ITcIoFileImageRecv

图像接收器的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcIoImageRecv [▶ 1838]
    ITcIoFileImageRecv

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
ReceiveImage [▶ 1838]	ITcIoImageRecv	从 ITcIoGevImageAcquisition 实例接收图像。
ReceiveOpResult [▶ 1839]	ITcIoImageRecv	从 ITcIoGevImageAcquisition 实例接收操作结果。
ReceiveImage [▶ 1834]	ITcIoFileImageRecv	从 ITcIoFileImageAcquisition 的实例中接收图像。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.1.2.1 *ReceiveImage*

从 ITcIoFileImageAcquisition 的实例中接收图像。

语法

定义：

```
HRESULT ReceiveImage(
    ITcVnImageBase* ipImage,
    PCHAR           sFileName,
    HRESULT         hrAcquisitionResult
)
```

参数

名称	类型	描述
ipImage	ITcVnImageBase* [▶ 1865]	图像的接口指针。
sFileName	PCHAR	返回图像的文件名。
hrAcquisitionResult	HRESULT [▶ 135]	指示图像采集状态的 HRESULT。

返回值

HRESULT [▶ 135]

6.2.2.2.1.3 ITcIoGevImageAcquisition

GigE Vision 图像采集接口。

继承层次

ITcUnknown [▶ 1877]
  ITcIoImageAcquisition [▶ 1835]
    ITcIoGevImageAcquisition

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
RegisterReceiver [▶ 1836]	ITcIoImageAcquisition	注册回调函数接口。
UnregisterReceiver [▶ 1836]	ITcIoImageAcquisition	取消注册回调函数接口。
OpenCamera [▶ 1836]	ITcIoImageAcquisition	打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。
CloseCamera [▶ 1837]	ITcIoImageAcquisition	关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。
StartAcquisition [▶ 1837]	ITcIoImageAcquisition	开始图像采集。
StopAcquisition [▶ 1837]	ITcIoImageAcquisition	停止图像采集。

名称	起源	描述
SoftwareTrigger [▶ 1837]	ITcIoImageAcquisition	初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。
InitializeCamera [▶ 1838]	ITcIoImageAcquisition	向相机发送初始化命令。
CheckConnection [▶ 1835]	ITcIoGevImageAcquisition	检查相机连接

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.1.3.1 *CheckConnection*

检查相机连接

语法

定义：

```
HRESULT CheckConnection(
    GEV_CAMERA_STATE& eAssumedState,
    GEV_CAMERA_STATE& eActualState
)
```

参数

名称	类型	描述
eAssumedState	GEV_CAMERA_STATE [▶ 1697]&	相机内部假定的状态。
eActualState	GEV_CAMERA_STATE [▶ 1697]&	相机的实际可观测状态。

返回值

HRESULT [▶ 135]

6.2.2.2.1.4 ITcIoImageAcquisition

图像采集接口。

继承层次

ITcUnknown [▶ 1877]
  ITcIoImageAcquisition

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
RegisterReceiver [▶ 1836]	ITcIoImageAcquisition	注册回调函数接口。
UnregisterReceiver [▶ 1836]	ITcIoImageAcquisition	取消注册回调函数接口。
OpenCamera [▶ 1836]	ITcIoImageAcquisition	打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。

名称	起源	描述
CloseCamera [▶ 1837]	ITcIoImageAcquisition	关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。
StartAcquisition [▶ 1837]	ITcIoImageAcquisition	开始图像采集。
StopAcquisition [▶ 1837]	ITcIoImageAcquisition	停止图像采集。
SoftwareTrigger [▶ 1837]	ITcIoImageAcquisition	初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。
InitializeCamera [▶ 1838]	ITcIoImageAcquisition	向相机发送初始化命令。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.1.4.1 *RegisterReceiver*

注册回调函数接口。

语法

定义：

```
HRESULT RegisterReceiver(
    ITcIoImageRecv* ipRecv
)
```

参数

名称	类型	描述
ipRecv	ITcIoImageRecv* [▶ 1838]	指向包含回调函数的接口的指针。

返回值

HRESULT [▶ 135]

6.2.2.2.1.4.2 *UnregisterReceiver*

取消注册回调函数接口。

语法

定义：

```
HRESULT UnregisterReceiver()
```

返回值

HRESULT [▶ 135]

6.2.2.2.1.4.3 *OpenCamera*

打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。

语法

定义：

```
HRESULT OpenCamera()
```

返回值

HRESULT [▶ 135]

6.2.2.2.1.4.4 *CloseCamera*

关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。

语法

定义：

```
HRESULT CloseCamera()
```

返回值

HRESULT [▶ 135]

6.2.2.2.1.4.5 *StartAcquisition*

开始图像采集。

语法

定义：

```
HRESULT StartAcquisition()
```

返回值

HRESULT [▶ 135]

6.2.2.2.1.4.6 *StopAcquisition*

停止图像采集。

语法

定义：

```
HRESULT StopAcquisition()
```

返回值

HRESULT [▶ 135]

6.2.2.2.1.4.7 *SoftwareTrigger*

初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。

语法

定义：

```
HRESULT SoftwareTrigger(
    bool bSplitConcatenatedCommands,
    bool bOmitAcknowledgement
)
```

参数

名称	类型	描述
bSplitConcatenatedCommands	BOOL	如果为 true，多读和多写命令将分别被拆分成单读和单写命令序列。
bOmitAcknowledgement	BOOL	表示不请求确认数据包。

返回值

HRESULT [▶ 135]

6.2.2.2.1.4.8 *InitializeCamera*

向相机发送初始化命令。

语法

定义：

```
HRESULT InitializeCamera()
```

返回值

HRESULT [▶ 135]

6.2.2.2.1.5 ITcIoImageRecv

图像接收器的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcIoImageRecv

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
ReceiveImage [▶ 1838]	ITcIoImageRecv	从 ITcIoGevImageAcquisition 实例接收图像。
ReceiveOpResult [▶ 1839]	ITcIoImageRecv	从 ITcIoGevImageAcquisition 实例接收操作结果。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.1.5.1 *ReceiveImage*

从 ITcIoGevImageAcquisition 实例接收图像。

语法

定义：

```
HRESULT ReceiveImage(
    ITcVnImageBase* ipImage,
    HRESULT         hrAcquisitionResult
)
```

参数

名称	类型	描述
ipImage	ITcVnImageBase* [▶ 1865]	图像的接口指针。
hrAcquisitionResult	HRESULT [▶ 135]	指示图像采集状态的 HRESULT。

返回值

HRESULT [▶ 135]

6.2.2.2.1.5.2 *ReceiveOpResult*

从 ITcIoGevImageAcquisition 实例接收操作结果。

语法

定义：

```
HRESULT ReceiveOpResult(
    HRESULT hrOperationResult
)
```

参数

名称	类型	描述
hrOperationResult	HRESULT [▶ 135]	指示已执行操作状态的 HRESULT。

返回值

HRESULT [▶ 135]

6.2.2.2.1.6 ITcVnFileImageProvider

用于外部实时环境图像来源的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnImageProvider [▶ 1843]
    ITcVnFileImageProvider

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
CheckIfBusy [▶ 1844]	ITcVnImageProvider	检查模块是否繁忙，即是否正在进行操作。

名称	起源	描述
CheckIfCameraConnected [▶ 1844]	ITcVnImageProvider	检查相机是否正确连接（如果已连接，返回 S_OK；未连接则返回 S_FALSE）。只能检查软件连接，即如果返回 S_FALSE，检查无法确定电缆是否连接）。
GetCurrentImage [▶ 1844]	ITcVnImageProvider	通过脱离内部引用获取当前图像。
GetError [▶ 1844]	ITcVnImageProvider	获取最后一次执行的操作的结果。
OpenCamera [▶ 1845]	ITcVnImageProvider	打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。
CloseCamera [▶ 1845]	ITcVnImageProvider	关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。
StartAcquisition [▶ 1845]	ITcVnImageProvider	开始图像采集。
StopAcquisition [▶ 1845]	ITcVnImageProvider	停止图像采集。
SoftwareTrigger [▶ 1846]	ITcVnImageProvider	初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。
InitializeCamera [▶ 1846]	ITcVnImageProvider	将相机设置为初始状态。
GetCurrentImageAndFileName [▶ 1840]	ITcVnFileImageProvider	通过脱离内部引用获取当前图像。
TriggerImage [▶ 1841]	ITcVnFileImageProvider	初始化软件触发器并触发单个图像。跳过 nSkipImages 并触发采集图像 nSkipImages+1。
TriggerImageByName [▶ 1841]	ITcVnFileImageProvider	初始化软件触发器，并触发客户端助手中按名称指定的单个图像。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.1.6.1 *GetCurrentImageAndFileName*

通过脱离内部引用获取当前图像。

语法

定义：

```
HRESULT GetCurrentImageAndFileName(
    ITcVnImage** pipImage,
    PCHAR        sFileName,
    USHORT       nMaxLen
)
```

参数

名称	类型	描述
pipImage	ITcVnImage* [▶ 1863]*	应返回的接口指针的指针，如果图像采集失败，可能为 0
sFileName	PCHAR	以字符串形式返回文件名
nMaxLen	USHORT	允许写入 sFileName 的最大字符串长度 、

返回值

HRESULT [▶ 135]

6.2.2.2.1.6.2 *TriggerImage*

初始化软件触发器并触发单个图像。跳过 nSkipImages 并触发采集图像 nSkipImages+1。

语法

定义：

```
HRESULT TriggerImage(
    LONG nSkipImages
)
```

参数

名称	类型	描述
nSkipImages	LONG	要跳过的图像数量

返回值

HRESULT [▶ 135]

6.2.2.2.1.6.3 *TriggerImageByName*

初始化软件触发器，并触发客户端助手中按名称指定的单个图像。

语法

定义：

```
HRESULT TriggerImageByName(
    PCCH sImageName
)
```

参数

名称	类型	描述
sImageName	PCCH	要触发的图像名称

返回值

HRESULT [▶ 135]

6.2.2.2.1.7 ITcVnGevImageProvider

GigE Vision 图像来源的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnImageProvider [▶ 1843]
    ITcVnGevImageProvider

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
CheckIfBusy [▶ 1844]	ITcVnImageProvider	检查模块是否繁忙，即是否正在进行操作。

名称	起源	描述
CheckIfCameraConnected [▶ 1844]	ITcVnImageProvider	检查相机是否正确连接（如果已连接，返回 S_OK；未连接则返回 S_FALSE）。只能检查软件连接，即如果返回 S_FALSE，检查无法确定电缆是否连接）。
GetCurrentImage [▶ 1844]	ITcVnImageProvider	通过脱离内部引用获取当前图像。
GetError [▶ 1844]	ITcVnImageProvider	获取最后一次执行的操作的结果。
OpenCamera [▶ 1845]	ITcVnImageProvider	打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。
CloseCamera [▶ 1845]	ITcVnImageProvider	关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。
StartAcquisition [▶ 1845]	ITcVnImageProvider	开始图像采集。
StopAcquisition [▶ 1845]	ITcVnImageProvider	停止图像采集。
SoftwareTrigger [▶ 1846]	ITcVnImageProvider	初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。
InitializeCamera [▶ 1846]	ITcVnImageProvider	将相机设置为初始状态。
GetCurrentGevImage [▶ 1842]	ITcVnGevImageProvider	通过脱离 GigE Vision 图像的内部引用，获取当前 GigE Vision 图像。
GetCurrentImageWithGvspInfo [▶ 1842]	ITcVnGevImageProvider	通过脱离内部引用获取当前图像，并提供 GVSP 信息。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.1.7.1 *GetCurrentGevImage*

通过脱离 GigE Vision 图像的内部引用，获取当前 GigE Vision 图像。

语法

定义：

```
HRESULT GetCurrentGevImage(
    ITcIoGevImage** pipGevImage
)
```

参数

名称	类型	描述
pipGevImage	ITcIoGevImage* [▶ 1860]*	应返回的接口指针的指针，如果图像采集失败，可能为 0。

返回值

HRESULT [▶ 135]

6.2.2.2.1.7.2 *GetCurrentImageWithGvspInfo*

通过脱离内部引用获取当前图像，并提供 GVSP 信息。

语法

定义：

```
HRESULT GetCurrentImageWithGvspInfo(
    ITcVnImage**      pipImage,
    GVSP_IMAGE_INFO& stGvspInfo
)
```

参数

名称	类型	描述
pipImage	ITcVnImage* [▶ 1863]*	应返回的接口指针的指针，如果图像采集失败，可能为 0。
stGvspInfo	GVSP_IMAGE_INFO [▶ 1700]&	包含有用的元信息，如图像 ID、时间戳等。

返回值

HRESULT [▶ 135]

6.2.2.2.1.8 ITcVnImageProvider

图像来源的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnImageProvider

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
CheckIfBusy [▶ 1844]	ITcVnImageProvider	检查模块是否繁忙，即是否正在进行操作。
CheckIfCameraConnected [▶ 1844]	ITcVnImageProvider	检查相机是否正确连接（如果已连接，返回 S_OK；未连接则返回 S_FALSE）。只能检查软件连接，即如果返回 S_FALSE，检查无法确定电缆是否连接）。
GetCurrentImage [▶ 1844]	ITcVnImageProvider	通过脱离内部引用获取当前图像。
GetError [▶ 1844]	ITcVnImageProvider	获取最后一次执行的操作的结果。
OpenCamera [▶ 1845]	ITcVnImageProvider	打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。
CloseCamera [▶ 1845]	ITcVnImageProvider	关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。
StartAcquisition [▶ 1845]	ITcVnImageProvider	开始图像采集。
StopAcquisition [▶ 1845]	ITcVnImageProvider	停止图像采集。
SoftwareTrigger [▶ 1846]	ITcVnImageProvider	初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。
InitializeCamera [▶ 1846]	ITcVnImageProvider	将相机设置为初始状态。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.1.8.1 CheckIfBusy

检查模块是否繁忙，即是否正在进行操作。

语法

定义：

```
HRESULT CheckIfBusy()
```

返回值

HRESULT [▶ 135]

6.2.2.2.1.8.2 CheckIfCameraConnected

检查相机是否正确连接（如果已连接，返回 S_OK；未连接则返回 S_FALSE）。只能检查软件连接，即如果返回 S_FALSE，检查无法确定电缆是否连接）。

语法

定义：

```
HRESULT CheckIfCameraConnected()
```

返回值

HRESULT [▶ 135]

6.2.2.2.1.8.3 GetCurrentImage

通过脱离内部引用获取当前图像。

语法

定义：

```
HRESULT GetCurrentImage(
    ITcVnImage** pipImage
)
```

参数

名称	类型	描述
pipImage	ITcVnImage* [▶ 1863]*	应返回的接口指针的指针，如果图像采集失败，可能为 0。

返回值

HRESULT [▶ 135]

6.2.2.2.1.8.4 GetError

获取最后一次执行的操作的结果。

语法

定义：

```
HRESULT GetError()
```

返回值

HRESULT [▶ 135]

6.2.2.2.1.8.5 *OpenCamera*

打开相机（打开控制通道，打开数据流通道并激活 GVSP 接收器）。

语法

定义：

```
HRESULT OpenCamera()
```

返回值

HRESULT [▶ 135]

6.2.2.2.1.8.6 *CloseCamera*

关闭相机（停用 GVSP 接收器，关闭数据流通道并关闭控制通道）。

语法

定义：

```
HRESULT CloseCamera()
```

返回值

HRESULT [▶ 135]

6.2.2.2.1.8.7 *StartAcquisition*

开始图像采集。

语法

定义：

```
HRESULT StartAcquisition()
```

返回值

HRESULT [▶ 135]

6.2.2.2.1.8.8 *StopAcquisition*

停止图像采集。

语法

定义：

```
HRESULT StopAcquisition()
```

返回值

HRESULT [▶ 135]

6.2.2.2.1.8.9 *SoftwareTrigger*

初始化软件触发器并进行触发。如果之前执行过初始化，则跳过该步骤。

语法

定义：

```
HRESULT SoftwareTrigger()
```

返回值

HRESULT [▶ 135]

6.2.2.2.1.8.10 *InitializeCamera*

将相机设置为初始状态。

语法

定义：

```
HRESULT InitializeCamera()
```

返回值

HRESULT [▶ 135]

6.2.2.2.2 导出

该组包含处理与图像有关的数据输出的接口。

6.2.2.2.2.1 ITcVnBitmapExport

将图像导出为 Windows 位图（BMP）的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnBitmapExport

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
GetBitmapSize [▶ 1847]	ITcVnBitmapExport	如果图像转换为 Windows 位图（BMP），则获取图像的大小。
GetBitmapImage [▶ 1847]	ITcVnBitmapExport	将图像以 Windows 位图（BMP）形式导出到给定的缓冲区中。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.2.1.1 *GetBitmapSize*

如果图像转换为 Windows 位图（BMP），则获取图像的大小。

语法

定义：

```
HRESULT GetBitmapSize(
    ULONGLONG& nBitmapSize,
    ULONG&     nBitmapWidth,
    ULONG&     nBitmapHeight
)
```

参数

名称	类型	描述
nBitmapSize	ULONGLONG&	输出参数，包含所需的缓冲区大小。
nBitmapWidth	ULONG&	所需的宽度，或使用 0 来保持原始宽度（输入）和实际宽度（输出）。
nBitmapHeight	ULONG&	所需的高度或 0，以保持原始高度（输入）和实际高度（输出）。

返回值

HRESULT [▶ 135]

6.2.2.2.2.1.2 *GetBitmapImage*

将图像以 Windows 位图（BMP）形式导出到给定的缓冲区中。

语法

定义：

```
HRESULT GetBitmapImage(
    ULONGLONG& nBitmapSize,
    PVOID      pDestBuffer,
    ULONG&     nBitmapWidth,
    ULONG&     nBitmapHeight
)
```

参数

名称	类型	描述
nBitmapSize	ULONGLONG&	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。
nBitmapWidth	ULONG&	所需的宽度，或使用 0 来保持原始宽度（输入）和实际宽度（输出）。
nBitmapHeight	ULONG&	所需的高度或 0，以保持原始高度（输入）和实际高度（输出）。

返回值

HRESULT [▶ 135]

6.2.2.2.2.2 ITcVnBitmapExportNotification

将可显示图像作为 ADS 通知发送所需的接口

继承层次

ITcUnknown [▶ 1877]
  ITcVnBitmapExportNotification

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
GetTimestamp [▶ 1848]	ITcVnBitmapExportNotification	获取最新图像更改的时间戳。
GetBitmapImageRpcUnlocked [▶ 1848]	ITcVnBitmapExportNotification	通过未锁定的远程过程调用，将图像以 Windows 位图（BMP）的形式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.2.2.1 *GetTimestamp*

获取最新图像更改的时间戳。

语法

定义：

```
HRESULT GetTimestamp(
    LONGLONG& nTimestamp
)
```

参数

名称	类型	描述
nTimestamp	LONGLONG&	返回时间戳。

返回值

HRESULT [▶ 135]

6.2.2.2.2.2.2 *GetBitmapImageRpcUnlocked*

通过未锁定的远程过程调用，将图像以 Windows 位图（BMP）的形式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

语法

定义：

```
HRESULT GetBitmapImageRpcUnlocked(
    ULONGLONG& nBitmapSize,
    PVOID      pDestBuffer,
    ULONG&     nBitmapWidth,
    ULONG&     nBitmapHeight
)
```

参数

名称	类型	描述
nBitmapSize	ULONGLONG&	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。
nBitmapWidth	ULONG&	所需的宽度，或使用 0 来保持原始宽度（输入）和实际宽度（输出）。
nBitmapHeight	ULONG&	所需的高度或 0，以保持原始高度（输入）和实际高度（输出）。

返回值

HRESULT [▶ 135]

6.2.2.2.2.3 ITcVnBitmapExportRpcUnlocked

通过未锁定的远程过程调用，将图像以 Windows 位图（BMP）形式导出到给定缓冲区的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnBitmapExport [▶ 1846]
    ITcVnBitmapExportRpcUnlocked

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
GetBitmapSize [▶ 1847]	ITcVnBitmapExport	如果图像转换为 Windows 位图（BMP），则获取图像的大小。
GetBitmapImage [▶ 1847]	ITcVnBitmapExport	将图像以 Windows 位图（BMP）形式导出到给定的缓冲区中。
GetBitmapImageRpcUnlocked [▶ 1850]	ITcVnBitmapExportRpcUnlocked	通过未锁定的远程过程调用，将图像以 Windows 位图（BMP）的形式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.2.3.1 GetBitmapImageRpcUnlocked

通过未锁定的远程过程调用，将图像以 Windows 位图（BMP）的形式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

语法

定义：

```
HRESULT GetBitmapImageRpcUnlocked(
    ULONGLONG& nBitmapSize,
    PVOID      pDestBuffer,
    ULONG&     nBitmapWidth,
    ULONG&     nBitmapHeight
)
```

参数

名称	类型	描述
nBitmapSize	ULONGLONG&	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。
nBitmapWidth	ULONG&	所需的宽度，或使用 0 来保持原始宽度（输入）和实际宽度（输出）。
nBitmapHeight	ULONG&	所需的高度或 0，以保持原始高度（输入）和实际高度（输出）。

返回值

HRESULT [▶ 135]

6.2.2.2.2.4 ITcVnHistogramExport

以数组形式导出图像直方图的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnHistogramExport

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
GetHistogramSize [▶ 1851]	ITcVnHistogramExport	获取直方图数组的大小。
GetHistogramArray [▶ 1851]	ITcVnHistogramExport	获取直方图数组。
GetCustomHistogramSize [▶ 1851]	ITcVnHistogramExport	获取自定义直方图数组的大小。
GetCustomHistogramArray [▶ 1852]	ITcVnHistogramExport	获取自定义直方图数组。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.2.4.1 *GetHistogramSize*

获取直方图数组的大小。

语法

定义：

```
HRESULT GetHistogramSize(
    ULONGLONG& nHistogramSize
)
```

参数

名称	类型	描述
nHistogramSize	ULONGLONG&	输出参数，包含所需的缓冲区大小。

返回值

HRESULT [▶ 135]

6.2.2.2.2.4.2 *GetHistogramArray*

获取直方图数组。

语法

定义：

```
HRESULT GetHistogramArray(
    ULONGLONG& nHistogramSize,
    PVOID      pDestBuffer
)
```

参数

名称	类型	描述
nHistogramSize	ULONGLONG&	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。

返回值

HRESULT [▶ 135]

6.2.2.2.2.4.3 *GetCustomHistogramSize*

获取自定义直方图数组的大小。

语法

定义：

```
HRESULT GetCustomHistogramSize(
    ULONGLONG& nHistogramSize,
    ULONG&     nBins
)
```

参数

名称	类型	描述
nHistogramSize	ULONGLONG&	输出参数，包含所需的缓冲区大小。
nBins	ULONG&	所需的分组数量，或用 0 保持相应图像格式的默认值（输入）和默认分组数量（输出）。

返回值

HRESULT [▶ 135]

6.2.2.2.2.4.4 *GetCustomHistogramArray*

获取自定义直方图数组。

语法

定义：

```
HRESULT GetCustomHistogramArray(
    ULONGLONG& nHistogramSize,
    PVOID      pDestBuffer,
    ULONG&     nBins,
    double&    fLowerBound,
    double&    fUpperBound
)
```

参数

名称	类型	描述
nHistogramSize	ULONGLONG&	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。
nBins	ULONG&	所需的分组数量，或用 0 保持相应图像格式的默认值（输入）和默认分组数量（输出）。
fLowerBound	double&	第 0 个直方图分组的下边界（包含）（输入），如果 fLowerBound 与 fUpperBound 都设置为 0，则接收默认值（输出）。
fUpperBound	double&	最后一个直方图分组 nBins-1 的上边界（不包含）（输入），如果 fLowerBound 与 fUpperBound 设置为 0，则接收默认值（输出）。

返回值

HRESULT [▶ 135]

6.2.2.2.2.5 ITcVnTiffExport

将图像导出为 tiff 格式的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnTiffExport

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
GetTiffSize [▶ 1853]	ITcVnTiffExport	如果图像转换为 tiff 格式，则获取图像的大小。
GetTiffImage [▶ 1853]	ITcVnTiffExport	将图像以 tiff 格式导出到给定的缓冲区中。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.2.5.1 *GetTiffSize*

如果图像转换为 tiff 格式，则获取图像的大小。

语法

定义：

```
HRESULT GetTiffSize(
    ULONGLONG& nTiffSize
)
```

参数

名称	类型	描述
nTiffSize	ULONGLONG&	输出参数，包含所需的缓冲区大小。

返回值

HRESULT [▶ 135]

6.2.2.2.2.5.2 *GetTiffImage*

将图像以 tiff 格式导出到给定的缓冲区中。

语法

定义：

```
HRESULT GetTiffImage(
    ULONGLONG& nTiffSize,
    PVOID      pDestBuffer
)
```

参数

名称	类型	描述
nTiffSize	ULONGLONG&	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。

返回值

HRESULT [▶ 135]

6.2.2.2.2.6 ITcVnTiffExportNotification

用于将可显示图像以 tiff 格式通过 ADS 通知发送的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnTiffExportNotification

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
GetTimestamp [▶ 1854]	ITcVnTiffExportNotification	获取最新图像更改的时间戳。
GetTiffImageRpcUnlocked [▶ 1855]	ITcVnTiffExportNotification	通过未锁定的远程过程调用，将图像以 tiff 格式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.2.6.1 *GetTimestamp*

获取最新图像更改的时间戳。

语法

定义：

```
HRESULT GetTimestamp(
    LONGLONG& nTimestamp
)
```

参数

名称	类型	描述
nTimestamp	LONGLONG&	返回时间戳。

返回值

HRESULT [▶ 135]

6.2.2.2.2.6.2 GetTiffImageRpcUnlocked

通过未锁定的远程过程调用，将图像以 tiff 格式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

语法

定义：

```
HRESULT GetTiffImageRpcUnlocked(
    ULONGLONG& nTiffSize,
    PVOID      pDestBuffer
)
```

参数

名称	类型	描述
nTiffSize	ULONGLONG&	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。

返回值

HRESULT [▶ 135]

6.2.2.2.2.7 ITcVnTiffExportRpcUnlocked

通过未锁定的远程过程调用，将图像导出为 tiff 格式的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnTiffExport [▶ 1852]
    ITcVnTiffExportRpcUnlocked

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
GetTiffSize [▶ 1853]	ITcVnTiffExport	如果图像转换为 tiff 格式，则获取图像的大小。
GetTiffImage [▶ 1853]	ITcVnTiffExport	将图像以 tiff 格式导出到给定的缓冲区中。
GetTiffImageRpcUnlocked [▶ 1856]	ITcVnTiffExportRpcUnlocked	通过未锁定的远程过程调用，将图像以 tiff 格式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.2.7.1 *GetTiffImageRpcUnlocked*

通过未锁定的远程过程调用，将图像以 tiff 格式导出到给定的缓冲区中。用户有责任确保不会发生冲突访问。

语法

定义：

```
HRESULT GetTiffImageRpcUnlocked(
    ULONGLONG& nTiffSize,
    PVOID      pDestBuffer
)
```

参数

名称	类型	描述
nTiffSize	ULONGLONG&	最大缓冲区大小（输入）和实际缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。

返回值

HRESULT [▶ 135]

6.2.2.2.2.8 ITcVnDataExport

Interface for exporting binary data.

Inheritance Hierarchy

ITcUnknown [▶ 1877]
  ITcVnDataExport

Methods

Name	Origin	Description
TcAddRef [▶ 1877]	ITcUnknown	Increments the reference counter.
TcQueryInterface [▶ 1877]	ITcUnknown	Get a reference to an implemented interface.
TcRelease [▶ 1878]	ITcUnknown	Decrements the reference counter.
GetSize [▶ 1856]	ITcVnDataExport	Get the size of the data.
GetData [▶ 1857]	ITcVnDataExport	Export the data into a given buffer.
GetDataIfNew [▶ 1857]	ITcVnDataExport	Export the data into a given buffer, if the data is new (i.e. not yet exported).

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.2.8.1 *GetSize*

Get the size of the data.

Syntax

Definition:

```
HRESULT GetSize(
    ULONGLONG& nSize
)
```

Parameters

Name	Type	Description
nSize	ULONGLONG&	Output parameter containing the data size.

Return value

HRESULT [▶ 135]

6.2.2.2.2.8.2 *GetData*

Export the data into a given buffer.

Syntax

Definition:

```
HRESULT GetData(
    ULONGLONG& nSize,
    PVOID      pDestBuffer
)
```

Parameters

Name	Type	Description
nSize	ULONGLONG&	Provided buffer size (in) and actual data size (out).
pDestBuffer	PVOID	Pointer to the destination buffer.

Return value

HRESULT [▶ 135]

6.2.2.2.2.8.3 *GetDataIfNew*

Export the data into a given buffer, if the data is new (i.e. not yet exported).

Syntax

Definition:

```
HRESULT GetDataIfNew(
    ULONGLONG& nSize,
    PVOID      pDestBuffer
)
```

Parameters

Name	Type	Description
nSize	ULONGLONG&	Provided buffer size (in) and actual data size (out).
pDestBuffer	PVOID	Pointer to the destination buffer.

Return value

HRESULT [▶ 135]

6.2.2.2.2.9 ITcVnDataExportRpcUnlocked

Interface for exporting binary data by means of an unlocked remote procedure call.

Inheritance Hierarchy

ITcUnknown [▶ 1877]
  ITcVnDataExport [▶ 1856]
    ITcVnDataExportRpcUnlocked

Methods

Name	Origin	Description
TcAddRef [▶ 1877]	ITcUnknown	Increments the reference counter.
TcQueryInterface [▶ 1877]	ITcUnknown	Get a reference to an implemented interface.
TcRelease [▶ 1878]	ITcUnknown	Decrements the reference counter.
GetSize [▶ 1856]	ITcVnDataExport	Get the size of the data.
GetData [▶ 1857]	ITcVnDataExport	Export the data into a given buffer.
GetDataIfNew [▶ 1857]	ITcVnDataExport	Export the data into a given buffer, if the data is new (i.e. not yet exported).
GetDataRpcUnlocked [▶ 1858]	ITcVnDataExportRpcUnlocked	Export the data into a given buffer by means of an unlocked remote procedure call. It lies within the responsibility of the user to ensure that no conflicting accesses can occur.
GetDataIfNewRpcUnlocked [▶ 1859]	ITcVnDataExportRpcUnlocked	Export the data into a given buffer by means of an unlocked remote procedure call, if the data is new (i.e. not yet exported). It lies within the responsibility of the user to ensure that no conflicting accesses can occur.

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.2.9.1 *GetDataRpcUnlocked*

Export the data into a given buffer by means of an unlocked remote procedure call. It lies within the responsibility of the user to ensure that no conflicting accesses can occur.

Syntax

Definition:

```
HRESULT GetDataRpcUnlocked(
    ULONGLONG& nSize,
    PVOID      pDestBuffer
)
```

Parameters

Name	Type	Description
nSize	ULONGLONG&	Provided buffer size (in) and actual data size (out).
pDestBuffer	PVOID	Pointer to the destination buffer.

Return value

HRESULT [▶ 135]

6.2.2.2.2.9.2 *GetDataIfNewRpcUnlocked*

Export the data into a given buffer by means of an unlocked remote procedure call, if the data is new (i.e. not yet exported). It lies within the responsibility of the user to ensure that no conflicting accesses can occur.

Syntax

Definition:

```
HRESULT GetDataIfNewRpcUnlocked(
    ULONGLONG& nSize,
    PVOID      pDestBuffer
)
```

Parameters

Name	Type	Description
nSize	ULONGLONG&	Provided buffer size (in) and actual data size (out).
pDestBuffer	PVOID	Pointer to the destination buffer.

Return value

HRESULT [▶ 135]

6.2.2.2.3 Import

该组包含用于处理与图像相关的数据导入的接口。

6.2.2.2.3.1 ITcVnTiffImport

导入 tiff 图像的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnTiffImport

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
ImportTiffImage [▶ 1860]	ITcVnTiffImport	导入一张 tiff 图像。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.3.1.1 *ImportTiffImage*

导入一张 tiff 图像。

语法

定义：

```
HRESULT ImportTiffImage(
    PVOID pTiffImage
)
```

参数

名称	类型	描述
pTiffImage	PVOID	指向 tiff 文件的指针。

返回值

HRESULT [▶ 135]

6.2.2.2.4 ITcIoGevImage

GigE Vision 图像的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnImageBase [▶ 1865]
    ITcIoGevImage

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
GetImageData [▶ 1866]	ITcVnImageBase	获取数据接口（不要在其中明确释放指针！）。
GetImageSize [▶ 1866]	ITcVnImageBase	获取图像大小。
GetWidth [▶ 1866]	ITcVnImageBase	获取图像宽度（或者使用 F_VN_GetImageWidth）。
GetHeight [▶ 1867]	ITcVnImageBase	获取图像高度（或者使用 F_VN_GetImageHeight）。
GetXPadding [▶ 1867]	ITcVnImageBase	获取水平填充。
GetYPadding [▶ 1867]	ITcVnImageBase	获取垂直填充。
GetPixelFormat [▶ 1868]	ITcVnImageBase	获取像素格式（或者使用 F_VN_GetPixelFormat）。
GetImageInfo [▶ 1868]	ITcVnImageBase	获取包含图像所有常见元信息的结构体。这基本上包含了可通过该接口访问的所有元信息。（或者使用 F_VN_GetImageInfo。）
CheckIfCopy [▶ 1861]	ITcIoGevImage	检查图像数据是否为副本。
DecreaseHeight [▶ 1861]	ITcIoGevImage	如果图像高度小于当前值，则将图像高度设置为 newHeight。
GetBlockId [▶ 1861]	ITcIoGevImage	获取 GVSP 块 ID。
GetGevStatus [▶ 1862]	ITcIoGevImage	获取块状态。
GetGvspImageInfo [▶ 1862]	ITcIoGevImage	获取指向 GVSP 前导有效载荷的指针。

名称	起源	描述
SetGevStatus [▶ 1862]	ITcIoGevImage	设置块状态。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.4.1 CheckIfCopy

检查图像数据是否为副本。

语法

定义：

```
HRESULT CheckIfCopy()
```

返回值

HRESULT [▶ 135]

6.2.2.2.4.2 DecreaseHeight

如果图像高度小于当前值，则将图像高度设置为 newHeight。

语法

定义：

```
HRESULT DecreaseHeight(
    ULONG nNewHeight
)
```

参数

名称	类型	描述
nNewHeight	ULONG	要设置的高度值。

返回值

HRESULT [▶ 135]

6.2.2.2.4.3 GetBlockId

获取 GVSP 块 ID。

语法

定义：

```
HRESULT GetBlockId(
    ULONGLONG& nBlockId
)
```

参数

名称	类型	描述
nBlockId	ULONGLONG&	返回块 ID。

返回值

HRESULT [▶ 135]

6.2.2.2.4.4 GetGevStatus

获取块状态。

语法

定义：

```
HRESULT GetGevStatus(
    USHORT& nGevStatus
)
```

参数

名称	类型	描述
nGevStatus	USHORT&	返回 GigE Vision 状态代码。

返回值

HRESULT [▶ 135]

6.2.2.2.4.5 GetGvspImageInfo

获取指向 GVSP 前导有效载荷的指针。

语法

定义：

```
HRESULT GetGvspImageInfo(
    GVSP_IMAGE_INFO** ppGvspImageInfo
)
```

参数

名称	类型	描述
ppGvspImageInfo	GVSP_IMAGE_INFO [▶ 1700]**	指向 GVSP 元信息的指针。

返回值

HRESULT [▶ 135]

6.2.2.2.4.6 SetGevStatus

设置块状态。

语法

定义：

```
HRESULT SetGevStatus(
    USHORT nGevStatus
)
```

参数

名称	类型	描述
nGevStatus	USHORT	GigE Vision 状态代码。

返回值

HRESULT [▶ 135]

6.2.2.2.5 ITcVnDisplayableImage

可显示图像的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnDisplayableImage

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.6 ITcVnImage

图像的基本接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnImageBase [▶ 1865]
    ITcVnImage

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
GetImageData [▶ 1866]	ITcVnImageBase	获取数据接口（不要在其中明确释放指针！）。
GetImageSize [▶ 1866]	ITcVnImageBase	获取图像大小。
GetWidth [▶ 1866]	ITcVnImageBase	获取图像宽度（或者使用 F_VN_GetImageWidth）。
GetHeight [▶ 1867]	ITcVnImageBase	获取图像高度（或者使用 F_VN_GetImageHeight）。
GetXPadding [▶ 1867]	ITcVnImageBase	获取水平填充。

名称	起源	描述
GetYPadding [▶ 1867]	ITcVnImageBase	获取垂直填充。
GetPixelFormat [▶ 1868]	ITcVnImageBase	获取像素格式（或者使用 F_VN_GetPixelFormat）。
GetImageInfo [▶ 1868]	ITcVnImageBase	获取包含图像所有常见元信息的结构体。这基本上包含了可通过该接口访问的所有元信息。（或者使用 F_VN_GetImageInfo。）
Init [▶ 1864]	ITcVnImage	使用 ITcVnImageBase 接口初始化图像。
GetRowPointer [▶ 1864]	ITcVnImage	获取指向图像特定行的指针。
ReleaseRowPointer [▶ 1865]	ITcVnImage	释放指向图像特定行的指针。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.6.1 Init

使用 ITcVnImageBase 接口初始化图像。

语法

定义：

```
HRESULT Init(
    ITcVnImageBase* ipImageBase
)
```

参数

名称	类型	描述
ipImageBase	ITcVnImageBase* [▶ 1865]	从中获取数据的图像。

返回值

HRESULT [▶ 135]

6.2.2.2.6.2 GetRowPointer

获取指向图像特定行的指针。

语法

定义：

```
HRESULT GetRowPointer(
    ULONG  nRowIndex,
    PVOID* ppRow
)
```

参数

名称	类型	描述
nRowIndex	ULONG	行索引。
ppRow	PVOID*	返回指向所请求图像行的指针。

返回值

HRESULT [▶ 135]

6.2.2.2.6.3 ReleaseRowPointer

释放指向图像特定行的指针。

语法

定义：

```
HRESULT ReleaseRowPointer(
    PVOID* ppRow
)
```

参数

名称	类型	描述
ppRow	PVOID*	指向要释放的行指针的指针。

返回值

HRESULT [▶ 135]

6.2.2.2.7 ITcVnImageBase

所有图像类型的基本接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnImageBase

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
GetImageData [▶ 1866]	ITcVnImageBase	获取数据接口（不要在其中明确释放指针！）。
GetImageSize [▶ 1866]	ITcVnImageBase	获取图像大小。
GetWidth [▶ 1866]	ITcVnImageBase	获取图像宽度（或者使用 F_VN_GetImageWidth）。
GetHeight [▶ 1867]	ITcVnImageBase	获取图像高度（或者使用 F_VN_GetImageHeight）。
GetXPadding [▶ 1867]	ITcVnImageBase	获取水平填充。
GetYPadding [▶ 1867]	ITcVnImageBase	获取垂直填充。
GetPixelFormat [▶ 1868]	ITcVnImageBase	获取像素格式（或者使用 F_VN_GetPixelFormat）。
GetImageInfo [▶ 1868]	ITcVnImageBase	获取包含图像所有常见元信息的结构体。这基本上包含了可通过该接口访问的所有元信息。（或者使用 F_VN_GetImageInfo。）

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.2.7.1 GetImageData

获取数据接口（不要在其中明确释放指针！）。

语法

定义：

```
HRESULT GetImageData(
    ITcVnData** pipImageData
)
```

参数

名称	类型	描述
pipImageData	ITcVnData* [▶ 1873]*	返回指向图像数据接口的指针。

返回值

HRESULT [▶ 135]

6.2.2.2.7.2 GetImageSize

获取图像大小。

语法

定义：

```
HRESULT GetImageSize(
    ULONGLONG& nSize
)
```

参数

名称	类型	描述
nSize	ULONGLONG&	返回以字节为单位的图像大小。

返回值

HRESULT [▶ 135]

6.2.2.2.7.3 GetWidth

获取图像宽度（或者使用 F_VN_GetImageWidth）。

语法

定义：

```
HRESULT GetWidth(
    ULONG& nWidth
)
```

参数

名称	类型	描述
nWidth	ULONG&	返回以像素为单位的图像宽度。

返回值

HRESULT [▶ 135]

6.2.2.2.7.4 GetHeight

获取图像高度（或者使用 F_VN_GetImageHeight）。

语法

定义：

```
HRESULT GetHeight(
    ULONG& nHeight
)
```

参数

名称	类型	描述
nHeight	ULONG&	返回以像素为单位的图像高度。

返回值

HRESULT [▶ 135]

6.2.2.2.7.5 GetXPadding

获取水平填充。

语法

定义：

```
HRESULT GetXPadding(
    USHORT& nXPadding
)
```

参数

名称	类型	描述
nXPadding	USHORT&	返回以字节为单位的水平填充。

返回值

HRESULT [▶ 135]

6.2.2.2.7.6 GetYPadding

获取垂直填充。

语法

定义：

```
HRESULT GetYPadding(
    USHORT& nYPadding
)
```

参数

名称	类型	描述
nYPadding	USHORT&	返回以字节为单位的垂直填充。

返回值

HRESULT [▶ 135]

6.2.2.2.7.7 GetPixelFormat

获取像素格式（或者使用 F_VN_GetPixelFormat）。

语法

定义：

```
HRESULT GetPixelFormat(
    TcVnPixelFormat& stPixelFormat
)
```

参数

名称	类型	描述
stPixelFormat	TcVnPixelFormat [▶ 1716]&	返回描述像素格式的结构体。

返回值

HRESULT [▶ 135]

6.2.2.2.7.8 GetImageInfo

获取包含图像所有常见元信息的结构体。这基本上包含了可通过该接口访问的所有元信息。（或者使用 F_VN_GetImageInfo。）

语法

定义：

```
HRESULT GetImageInfo(
    TcVnImageInfo& stImageInfo
)
```

参数

名称	类型	描述
stImageInfo	TcVnImageInfo [▶ 1705]&	返回描述图像的结构体。

返回值

HRESULT [▶ 135]

6.2.2.2.8 ITcVnImageTimestamps

用于读取图像时间戳的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnImageTimestamps

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
GetTimestamps [▶ 1869]	ITcVnImageTimestamps	获取选定的图像时间戳。

6.2.2.2.8.1 GetTimestamps

获取选定的图像时间戳。

语法

定义：

```
HRESULT GetTimestamps(
    ULONGLONG nTimestamps,
    ULONG     nArraySize,
    LONGLONG* pTimestamps
)
```

参数

名称	类型	描述
nTimestamps	ULONGLONG	选择要返回的时间戳（使用按位 OR 组合多个 ETcVnTimestamp 值）
nArraySize	ULONG	pTimestamps 数组大小（元素的数量）
pTimestamps	LONGLONG*	返回请求的时间戳，按其 ETcVnTimestamp 值升序排列（确保提供足够大小的数组）。

返回值

HRESULT [▶ 135]

6.2.2.3 机器学习

该组包含机器学习的接口。

6.2.2.3.1 ITcVnColorModel

对图像颜色进行训练和分类的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnMlModel [▶ 1871]
    ITcVnColorModel

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
GetTypeGuid [▶ 1871]	ITcVnMlModel	获取模型的 GUID。
GetTypeName [▶ 1872]	ITcVnMlModel	获取字符串形式的模型类型名称。
TrainImageColor [▶ 1870]	ITcVnColorModel	训练所提供图像的颜色。
ClassifyImageColor [▶ 1870]	ITcVnColorModel	计算每个图像像素与所训练参考色的相似度。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.3.1.1 TrainImageColor

训练所提供图像的颜色。

语法

定义：

```
HRESULT TrainImageColor(
    ITcVnImage* ipSrcImage,
    ULONG       nDifferentColors,
    ITcVnImage* ipMask,
    ULONG       nSkipPixels
)
```

参数

名称	类型	描述
ipSrcImage	ITcVnImage* [▶ 1863]	源图像。
nDifferentColors	ULONG	要区分的不同颜色的数量。
ipMask	ITcVnImage* [▶ 1863]	可选图像掩膜。
nSkipPixels	ULONG	在每个评估的颜色样本之间跳过的像素数量（以获得更好的性能）。0 将每个像素都考虑在内，往往更准确。

返回值

HRESULT [▶ 135]

6.2.2.3.1.2 ClassifyImageColor

计算每个图像像素与所训练参考色的相似度。

语法

定义：

```
HRESULT ClassifyImageColor(
    ITcVnImage*  ipSrcImage,
    ITcVnImage** pipDestImage,
```

```
    float        fVariance,
    float        fLuminanceWeight
)
```

参数

名称	类型	描述
ipSrcImage	ITcVnImage* [▶ 1863]	源图像。
pipDestImage	ITcVnImage* [▶ 1863]*	返回颜色相似度。
fVariance	float	允许的颜色方差。
fLuminanceWeight	float	权衡亮度的影响。

返回值

HRESULT [▶ 135]

6.2.2.3.2 ITcVnMlModel

机器学习模型的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnMlModel

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
GetTypeGuid [▶ 1871]	ITcVnMlModel	获取模型的 GUID。
GetTypeName [▶ 1872]	ITcVnMlModel	获取字符串形式的模型类型名称。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.3.2.1 GetTypeGuid

获取模型的 GUID。

语法

定义：

```
HRESULT GetTypeGuid(
    GUID& nTypeGuid
)
```

参数

名称	类型	描述
nTypeGuid	GUID&	返回模型的 GUID。

返回值

HRESULT [▶ 135]

6.2.2.3.2.2 GetTypeName

获取字符串形式的模型类型名称。

语法

定义：

```
HRESULT GetTypeName(
    PCHAR  sTypeName,
    USHORT nMaxLen
)
```

参数

名称	类型	描述
sTypeName	PCHAR	以字符串形式返回模型类型名称。
nMaxLen	USHORT	允许写入 sTypeName 的最大字符串长度。

返回值

HRESULT [▶ 135]

6.2.2.4 其它

该组包含其他接口。

6.2.2.4.1 ITcSerializableNotification

以 ADS 通知形式发送序列化对象所需的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcSerializableNotification

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
GetComparisonValue [▶ 1873]	ITcSerializableNotification	获取反映底层对象变化的值（例如对象最新变化的时间戳）。
SerializeRpcUnlocked [▶ 1873]	ITcSerializableNotification	将对象序列化到给定的缓冲区中。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.4.1.1 GetComparisonValue

获取反映底层对象变化的值（例如对象最新变化的时间戳）。

语法

定义：

```
HRESULT GetComparisonValue(
    LONGLONG& nComparisonValue
)
```

参数

名称	类型	描述
nComparisonValue	LONGLONG&	返回比较值。

返回值

HRESULT [▶ 135]

6.2.2.4.1.2 SerializeRpcUnlocked

将对象序列化到给定的缓冲区中。

语法

定义：

```
HRESULT SerializeRpcUnlocked(
    ULONGLONG& nBufferSize,
    PVOID      pDestBuffer
)
```

参数

名称	类型	描述
nBufferSize	ULONGLONG&	最大缓冲区大小（输入）和实际使用的缓冲区大小（输出）。
pDestBuffer	PVOID	指向目标缓冲区的指针。

返回值

HRESULT [▶ 135]

6.2.2.4.2 ITcVnData

用于访问数据的接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnData

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
CheckIfCopy [▶ 1874]	ITcVnData	检查数据是否为副本（副本返回 S_OK，否则返回 S_FALSE）。
CheckIfImage [▶ 1874]	ITcVnData	检查数据是否属于图像（属于返回 S_OK，不属于返回 S_FALSE）。
GetSize [▶ 1874]	ITcVnData	获取数据的大小。
CheckDataPointer [▶ 1875]	ITcVnData	检查数据指针是否非 0（不为 0 时返回 S_OK，否则返回 S_FALSE）。
GetDataPointer [▶ 1875]	ITcVnData	获取数据指针（必须通过调用 ReleaseDataPointer 释放获取的数据指针）。
ReleaseDataPointer [▶ 1875]	ITcVnData	释放数据指针。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.4.2.1 CheckIfCopy

检查数据是否为副本（副本返回 S_OK，否则返回 S_FALSE）。

语法

定义：

```
HRESULT CheckIfCopy()
```

返回值

HRESULT [▶ 135]

6.2.2.4.2.2 CheckIfImage

检查数据是否属于图像（属于返回 S_OK，不属于返回 S_FALSE）。

语法

定义：

```
HRESULT CheckIfImage()
```

返回值

HRESULT [▶ 135]

6.2.2.4.2.3 GetSize

获取数据的大小。

语法

定义：

```
HRESULT GetSize(
    ULONGLONG& nSize
)
```

参数

名称	类型	描述
nSize	ULONGLONG&	存储数据的大小（以字节为单位）。

返回值

HRESULT [▶ 135]

6.2.2.4.2.4 CheckDataPointer

检查数据指针是否非 0（不为 0 时返回 S_OK，否则返回 S_FALSE）。

语法

定义：

```
HRESULT CheckDataPointer()
```

返回值

HRESULT [▶ 135]

6.2.2.4.2.5 GetDataPointer

获取数据指针（必须通过调用 ReleaseDataPointer 释放获取的数据指针）。

语法

定义：

```
HRESULT GetDataPointer(
    PVOID* ppData
)
```

参数

名称	类型	描述
ppData	PVOID*	返回数据指针。

返回值

HRESULT [▶ 135]

6.2.2.4.2.6 ReleaseDataPointer

释放数据指针。

语法

定义：

```
HRESULT ReleaseDataPointer(
    PVOID* ppData
)
```

参数

名称	类型	描述
ppData	PVOID*	指向要释放的数据指针的指针。

返回值

HRESULT [▶ 135]

6.2.2.4.3 ITcVnTimestamp

提供一个时间戳接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnTimestamp

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
UpdateTimestamp [▶ 1876]	ITcVnTimestamp	将时间戳更新为当前时间
GetTimestamp [▶ 1876]	ITcVnTimestamp	获取时间戳

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.4.3.1 UpdateTimestamp

将时间戳更新为当前时间

语法

定义：

```
HRESULT UpdateTimestamp()
```

返回值

HRESULT [▶ 135]

6.2.2.4.3.2 GetTimestamp

获取时间戳

语法

定义：

```
HRESULT GetTimestamp(
    LONGLONG& nTimestamp
)
```

参数

名称	类型	描述
nTimestamp	LONGLONG&	返回时间戳

返回值

HRESULT [▶ 135]

6.2.2.4.4 接口 ITcUnknown

ITcUnknown 定义引用计数以及对更具体接口的引用查询。

语法

```
TCOM_DECL_INTERFACE("00000001-0000-0000-e000-000000000064", ITcUnknown)
```

声明形式：TcInterfaces.h

要求包括：

方法

名称	描述
TcAddRef [▶ 1877]	增加引用计数器。
TcQueryInterface [▶ 1877]	通过 IID 查询已实施接口的引用。
TcRelease [▶ 1878]	减少引用计数器。

每个 TcCOM 界面都直接或间接从 ITcUnknown 派生 因此，每个 TcCOM 模块类均执行 ITcUnknown，因为它是从 ITComObject 派生。

ITcUnknown 的标准执行可确保在释放最后一个引用后删除对象。因此，在调用 TcRelease() 后，不得取消引用接口指针。

6.2.2.4.4.1 方法 ITcUnknown:TcAddRef

增加引用计数器并返回新值。

语法

```
ULONG TcAddRef( )
```

返回值

产生引用计数值。

6.2.2.4.4.2 方法 ITcUnknown:TcQueryInterface

根据接口 ID (IID) 给出的接口查询接口指针。

语法

```
HRESULT TcQueryInterface(RITCID iid, PPVOID pipItf )
```

返回值

如果成功，将返回 S_OK（“0”）或其他正值，请参见返回值。关于扩展信息，请特别参见 ADS 返回代码 [▶ 2857]中的 HRESULT 栏。

如果所要求的接口不可用，该方法将返回 ADSERR_DEVICE_NOINTERFACE。

参数

名称	类型	描述
iid	RITCID	接口 IID。
pipItf	PPVOID	指向接口指针。当所请求的接口类型在相应实例中可用时设置。

描述

通过 IID 查询已实施接口的引用。建议使用智能指针来初始化和保存接口指针。

变量 1：

```
HRESULT GetTraceLevel(ITcUnkown* ip, TcTraceLevel& tl)
{
HRESULT hr = S_OK;
if (ip != NULL)
{
ITComObjectPtr spObj;
hr = ip->TcQueryInterface(spObj.GetIID(), &spObj);
if (SUCCEEDED(hr))
{
hr = spObj->TcGetObjPara(PID_TcTraceLevel, &tl, sizeof(tl));
}
return hr;
}
}
```

与智能指针相关的接口 ID 可用作 TcQueryInterface 中的参数。“&”操作符将返回指向智能指针内部接口指针成员变量的指针。变量 1 假定当 TcQueryInterface 指示成功时，接口指针被初始化。如果范围仍然存在，则智能指针 spObj 的析构函数会释放引用。

变量 2：

```
HRESULT GetTraceLevel(ITcUnkown* ip, TcTraceLevel& tl)
{
HRESULT hr = S_OK;
ITComObjectPtr spObj = ip;
if (spObj != NULL)
{
spObj->TcGetObjParam(PID_TcTraceLevel, &tl);
}
else
{
hr = ADS_E_NOINTERFACE;
}
return hr;
}
```

如果接口指针 ip 被分配给智能指针 spObj，则将在 ip 引用的实例上使用 IID_ITComObject 隐式调用 TcQueryInterface 方法。这样会缩短代码，但会丢失 TcQueryInterface 的原始返回代码。

6.2.2.4.4.3 方法 ITcUnknown:TcRelease

该方法减少引用计数器。

语法

```
ULONG TcRelease( )
```

返回值

产生引用计数值。

描述

减少引用计数器并返回新值。

如果引用计数器为 0，则对象会自行删除。

6.2.2.5 神经网络

6.2.2.5.1 ITcVnNeuralNetwork

神经网络接口。

继承层次

ITcUnknown [▶ 1877]
  ITcVnMlModel [▶ 1871]
    ITcVnNeuralNetwork

方法

名称	起源	描述
TcAddRef [▶ 1877]	ITcUnknown	增加引用计数器。
TcQueryInterface [▶ 1877]	ITcUnknown	获取已实现接口的引用。
TcRelease [▶ 1878]	ITcUnknown	减少引用计数器。
GetTypeGuid [▶ 1871]	ITcVnMlModel	获取模型的 GUID。
GetTypeName [▶ 1872]	ITcVnMlModel	获取字符串形式的模型类型名称。
Execute [▶ 1879]	ITcVnNeuralNetwork	使用提供的图像作为输入来执行神经网络，并提供一个输出层的输出。该方法不执行任何预处理或转换。
Execute_SingleLayerOutput [▶ 1880]	ITcVnNeuralNetwork	使用提供的图像作为输入来执行神经网络，并提供指定输出层的输出。该方法不执行任何预处理或转换。
Execute_MultiLayerOutput [▶ 1881]	ITcVnNeuralNetwork	使用提供的图像作为输入来执行神经网络，并提供指定输出层的输出。如果 sLayerNames 为空，该方法将返回未连接输出层的输出。该方法不执行任何预处理或转换。
GetLayerNames [▶ 1881]	ITcVnNeuralNetwork	获取模型中所有图层的名称。如果 bOnlyOuput 标志被设置为 true，则仅返回未连接输出层的名称。

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.2.5.1.1 执行

使用提供的图像作为输入来执行神经网络，并提供一个输出层的输出。该方法不执行任何预处理或转换。

语法

定义：

```
HRESULT Execute(
    ITcVnImage*  ipSrcImage,
    ITcVnImage** pipDestImage
)
```

参数

名称	类型	描述
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）
pipDestImage	ITcVnImage* [▶ 1863]*	目标图像（根据输出层创建适当的目标图像。要处理目标图像，必要时可使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数。

返回值

HRESULT [▶ 135]

6.2.2.5.1.2 Execute_SingleLayerOutput

使用提供的图像作为输入来执行神经网络，并提供指定输出层的输出。该方法不执行任何预处理或转换。

语法

定义：

```
HRESULT Execute_SingleLayerOutput(
    ITcVnImage*  ipSrcImage,
    ITcVnImage** pipDestImage,
    PCCH         sLayerName
)
```

参数

名称	类型	描述
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）
pipDestImage	ITcVnImage* [▶ 1863]*	目标图像（根据输出层创建适当的目标图像。要处理目标图像，必要时可使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数。
sLayerName	PCCH	指定需要输出的层的名称。如果为空，则提供一个输出层的输出。

返回值

HRESULT [▶ 135]

6.2.2.5.1.3 Execute_MultiLayerOutput

使用提供的图像作为输入来执行神经网络，并提供指定输出层的输出。如果 sLayerNames 为空，该方法将返回未连接输出层的输出。该方法不执行任何预处理或转换。

语法

定义：

```
HRESULT Execute_MultiLayerOutput(
    ITcVnImage*      ipSrcImage,
    ITcVnContainer** pipDestImages,
    ITcVnContainer*  ipLayerNames
)
```

参数

名称	类型	描述
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）
pipDestImages	ITcVnContainer* [▶ 1827]*	包含指定层的输出的图像的容器
ipLayerNames	ITcVnContainer* [▶ 1827]	要返回输出的层名称的容器（ContainerType_Vector_String_SINT）。如果设置为零，则容器 ipDestImages 将保留所有未连接输出层的输出。

返回值

HRESULT [▶ 135]

6.2.2.5.1.4 GetLayerNames

获取模型中所有图层的名称。如果 bOnlyOuput 标志被设置为 true，则仅返回未连接输出层的名称。

语法

定义：

```
HRESULT GetLayerNames(
    ITcVnContainer** pipLayerNames,
    bool             bOnlyOutputs
)
```

参数

名称	类型	描述
pipLayerNames	ITcVnContainer* [▶ 1827]*	带有层名称的容器（ContainerType_Vector_String_SINT）
bOnlyOutputs	BOOL	如果为 true，则仅返回未连接输出层的名称

返回值

HRESULT [▶ 135]

6.2.3 功能

图像处理算法被封装在函数中。更复杂的算法已作为功能块实现，可在 ImageProcessing [▶ 2654] 中找到。

函数调用的结构

函数结构如下：

```
hr := ProcessImage (hr, <...>, <...>);

Status variable of type HRESULT
Parameters and return values as references
Optional parameters with default values
```

hr	类型 HRESULT 的状态变量 [▶ 135]。借此可跟踪每个函数的执行状态。
参数 <...>	函数参数主要实现以下目的： • 待处理的输入参数（如图像或容器）。 • 用于设置算法的参数 • 作为返回结果引用的参数
可选参数 <...>	这些参数初始化为默认值，因此规格可选。

函数分组

TwinCAT Vision 库的所有函数按主题分为以下几组：

图像

- 基本图像运算 [▶ 2157]
- 代数图像运算 [▶ 1948]
- 图像分割 [▶ 2480]
- 图像颜色和对比处理 [▶ 2447]
- 图像分析 [▶ 2413]
- 图像滤波 [▶ 2464]

容器

- 基本容器运算 [▶ 1982]
- 代数容器运算 [▶ 1913]
- 容器统计 [▶ 2210]

轮廓

- 轮廓分析 [▶ 2282]

测量

- 测量 [▶ 2547]
- 几何和坐标转换 [▶ 2348]

编码质量

- 编码质量 [▶ 2189]

编码读取

- 编码读取 [▶ 2195]

光学字符识别 (OCR)

- 光学字符识别 [▶ 2608]

绘图和文本

- 绘图 [▶ 2307]

机器学习

- MachineLearning [▶ 2509]
- 神经网络 [▶ 2592]

高级功能

- 傅立叶分析 [▶ 2337]
- 关键点功能 [▶ 2486]

其他

- 控制函数 [▶ 2305]
- 其他 [▶ 2569]

专家函数

许多函数都有专家版本。它们包含附加参数，在 PLC 库中以名称后缀“Exp”标识。在 C++ 中，这些附加参数被指定为默认值，如果不需要，可以在调用时省略。

运行时行为

某些图像处理算法的执行时间取决于其参数设置和图像内容。因此，它并不确定。尽管如此，为了保证实时行为，可对这些函数或函数组应用 Watchdog（看门狗） [▶ 140]，其可在指定时间后终止函数。

⚠ 警告

函数使用不当会导致 TwinCAT 崩溃

如果某些函数使用不当，可能会导致错误（并导致 TwinCAT 崩溃）。这些错误不会在函数中被拦截，因为相应的检查会大大延长执行时间。因此，您必须注意正确使用这些函数！例如，我们建议对输入参数进行验证。

还请参阅有关此

- Polarized Image [▶ 1883]

6.2.3.1 Polarized Image

还请参阅有关此

- AngleAndDegreeOfLinearPolarization [▶ 1883]
- DemosaicPolarizedImage [▶ 1885]
- PolarizedImageGlareReduction [▶ 1887]

6.2.3.1.1 AngleAndDegreeOfLinearPolarization

Computes the Angle and the Degree of Linear Polarization (AoLP, DoLP) from the four polarization angle images (result of DemosaicPolarizedImage).
Can use available TwinCAT Job Tasks for executing parallel code regions.

Syntax

Definition:

```
HRESULT AngleAndDegreeOfLinearPolarization(
    HRESULT          hrPrev,
    ITcVnImage*      ip0Deg,
    ITcVnImage*      ip45Deg,
    ITcVnImage*      ip90Deg,
```

```
    ITcVnImage*       ip135Deg,
    ITcVnImage*&      ipAoLP,
    ITcVnImage*&      ipDoLP,
    ETcVnElementType eDestType
)
```

Parameters

Name	Type	Description
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)
ip0Deg	ITcVnImage* [▶ 1863]	The 0° component of the polarized image (USINT or UINT, 1 or 3 channels)
ip45Deg	ITcVnImage* [▶ 1863]	The 45° component of the polarized image (USINT or UINT, 1 or 3 channels)
ip90Deg	ITcVnImage* [▶ 1863]	The 90° component of the polarized image (USINT or UINT, 1 or 3 channels)
ip135Deg	ITcVnImage* [▶ 1863]	The 135° component of the polarized image (USINT or UINT, 1 or 3 channels)
ipAoLP	ITcVnImage* [▶ 1863]&	Returns the Angle of Linear Polarization (AoLP) image (Optional, set to 0 if not required. Must be different from the source images! An appropriate destination image will be created if required.)
ipDoLP	ITcVnImage* [▶ 1863]&	Returns the Degree of Linear Polarization (DoLP) image (Optional, set to 0 if not required. Must be different from the source images! An appropriate destination image will be created if required.)
eDestType	ETcVnElementType [▶ 1674]	Element type of the output images (ET_REAL or ET_USINT). For ET_REAL, AoLP is in the range of [-pi/2, pi/2] and DoLP in [0, 1]. For ET_USINT, both are scaled to a range of [0, 255].

Return value

HRESULT [▶ 135]

更多信息

`F_VN_AngleAndDegreeOfLinearPolarization` 函数根据四幅偏振角度图像计算线性偏振角（AoLP）和线性偏振度（DoLP）。这四幅图像是 F_VN_DemosaicPolarizedImage [▶ 1885] 函数的结果。

AoLP 描述偏振面的取向，而 DoLP 表示某个像素处光的偏振程度。例如，这两个量可用于可视化材料属性、表面取向或应力，而这些在传统强度图像中往往无法辨识。

参数

偏振角度的输入图像

四幅输入图像 `ip0Deg`、`ip45Deg`、`ip90Deg` 和 `ip135Deg` 分别代表分离后的偏振角度（0°、45°、90° 和 135°）。每幅图像的类型必须为 `USINT`（8 位）或 `UINT`（16 位），且具有 1 或 3 个通道。

结果图像（AoLP 和 DoLP）

线性偏振角（AoLP）以图像形式输出 `ipAoLP`，线性偏振度（DoLP）以图像形式输出 `ipDoLP`。如果不需要相应的输出，这两个参数均可设为 0。

输出元素类型

类型为 ETcVnElementType [▶ 1674] 的参数 `eDestType` 指定了输出图像的元素类型：

- `TCVN_ET_REAL`：AoLP 以 `[-π/2, π/2]` 范围输出，DoLP 以 `[0, 1]` 范围输出。
- `TCVN_ET_USINT`：两个值均缩放至 `[0, 255]` 范围。

Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.2.3.1.2 DemosaicPolarizedImage

Demosaics a polarized image (taken by a camera with a special polarization sensor) by separating it into four polarization angles (expected sensor layout: top left 90°, top right 45°, bottom left 135°, bottom right 0°).

Syntax

Definition:

```
HRESULT DemosaicPolarizedImage(
    HRESULT                          hrPrev,
    ITcVnImage*                      ipPolarizedImage,
    ITcVnImage*&                     ip0Deg,
    ITcVnImage*&                     ip45Deg,
    ITcVnImage*&                     ip90Deg,
    ITcVnImage*&                     ip135Deg,
    ETcVnPixelEncoding               eEncoding,
    ETcVnPolarizedImageInterpolation eInterpolation
)
```

Parameters

Name	Type	Description
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)
ipPolarizedImage	ITcVnImage* [▶ 1863]	Polarized image (USINT or UINT)
ip0Deg	ITcVnImage* [▶ 1863]&	Returns the 0° component of the polarized image (Optional, set to 0 if not required. Must be different from ipPolarizedImage! An appropriate destination image will be created if required.)

Name	Type	Description
ip45Deg	ITcVnImage* [▶ 1863]&	Returns the 45° component of the polarized image (Optional, set to 0 if not required. Must be different from ipPolarizedImage! An appropriate destination image will be created if required.)
ip90Deg	ITcVnImage* [▶ 1863]&	Returns the 90° component of the polarized image (Optional, set to 0 if not required. Must be different from ipPolarizedImage! An appropriate destination image will be created if required.)
ip135Deg	ITcVnImage* [▶ 1863]&	Returns the 135° component of the polarized image (Optional, set to 0 if not required. Must be different from ipPolarizedImage! An appropriate destination image will be created if required.)
eEncoding	ETcVnPixelEncoding [▶ 1687]	Select PE_NONE for Mono or PE_BAYER_XX for Bayer sensor
eInterpolation	ETcVnPolarizedImageInterpolation [▶ 1644]	Select PII_NONE to get the uninterpolated data, i.e. the ipXDeg result images only have the half width and height of ipPolarizedImage. The other options interpolate the data to the full resolution of ipPolarizedImage.

Return value

HRESULT [▶ 135]

更多信息

函数 `F_VN_DemosaicPolarizedImage` 可从符合特定偏振图像格式的原始传感器图像中重建与偏振角度（0°、45°、90° 和 135°）相对应的四幅独立图像。

如果原始图像包含颜色信息，也可以选择去马赛克处理。该函数要求四个偏振方向采用以下滤光片布局：左上 90°、右上 45°、左下 135°、右下 0°。

这四幅独立的角度图像构成了进一步分析的基础，例如使用 F_VN_AngleAndDegreeOfLinearPolarization [▶ 1883] 计算偏振角度和线性偏振度。

参数

输入图像

输入图像 `ipPolarizedImage` 必须是类型为 `USINT`（8 位）或 `UINT`（16 位）的偏振图像。

偏振角度的结果图像

四幅输出图像 `ip0Deg`、`ip45Deg`、`ip90Deg` 和 `ip135Deg` 分别返回对应偏振角度（0°、45°、90° 和 135°）的分离图像分量。如果不需要某个特定角度，可将相应参数设为 0。目标图像必须与输入图像 `ipPolarizedImage` 不同。

像素编码

类型为 ETcVnPixelEncoding [▶ 1687] 的参数 `eEncoding` 指定了传感器的像素编码。对于单色传感器，选择 `TCVN_PE_NONE`。对于彩色偏振传感器，必须将相应的 Bayer 模式作为 `TCVN_PE_BAYER_XX` 传入。

插值类型

类型为 ETcVnPolarizedImageInterpolation [▶ 1644] 的参数 `eInterpolation` 确定了分离后的角度图像如何进行插值或解析。由于每个偏振角度在水平和垂直方向上均仅占据每隔一个像素，因此未插值的结果图像的宽和高仅为输入图像的一半。
提供以下选项：

- `TCVN_PII_NONE`：不进行插值；结果图像的宽和高均为 `ipPolarizedImage` 的一半。
- `TCVN_PII_NEAREST_NEIGHBOR`、`TCVN_PII_BILINEAR`、`TCVN_PII_BICUBIC`、`TCVN_PII_LANCZOS4`：使用所选方法插值至完整传感器分辨率。

Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.2.3.1.3 PolarizedImageGlareReduction

Reduces glare in a polarized image (taken by a camera with a special polarization sensor) by selecting a small intensity value across the four polarization angles for each pixel (see fIndex parameter). For more advanced evaluation, you can use DemosaicPolarizedImage to access all the individual polarization angles.
Can use available TwinCAT Job Tasks for executing parallel code regions.

Syntax

Definition:

```
HRESULT PolarizedImageGlareReduction(
    HRESULT            hrPrev,
    ITcVnImage*        ipPolarizedImage,
    ITcVnImage*&       ipDestImage,
    ETcVnPixelEncoding eEncoding,
    float              fIndex
)
```

Parameters

Name	Type	Description
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)
ipPolarizedImage	ITcVnImage* [▶ 1863]	Polarized image (USINT or UINT)
ipDestImage	ITcVnImage* [▶ 1863]&	Returns the glare reduced image (An appropriate destination image will be created if required.)
eEncoding	ETcVnPixelEncoding [▶ 1687]	Select PE_NONE for Mono or PE_BAYER_XX for Bayer sensor
fIndex	float	The four polarization angles for each pixel are sorted by intensity. Select the index (in the range of [0, 3]) to use for the destination

Name	Type	Description
		image. Fractions compute a weighted average of 2 neighboring indices. Recommended to try 0.0 first, as this leads to the maximum possible glare reduction (but might lose some details). Values greater than 1 are valid, but usually increase glare instead of reducing it.

Return value

HRESULT [▶ 135]

更多信息

`F_VN_PolarizedImageGlareReduction` 函数可减少符合特定偏振图像格式的偏振图像中的眩光和反射。

对于每个像素，先对四个偏振角度的强度进行排序，然后使用 `fIndex` 确定结果图像的强度值。只有当强度在很大程度上取决于偏振角度时，也就是强度值存在显著差异时，才会对该像素进行调暗处理。非偏振图像区域在四个角度上的强度几乎相同，因此基本保持不变。

如需更详细地分析各个偏振角度，您可以使用 F_VN_DemosaicPolarizedImage [▶ 1885] 函数。

参数

输入图像

输入图像 `ipPolarizedImage` 必须是类型为 `USINT`（8 位）或 `UINT`（16 位）的偏振图像。

结果图像

亮度降低后的图像通过 `ipDestImage` 返回。

eEncoding（编码方式）

类型为 ETcVnPixelEncoding [▶ 1687] 的参数 `eEncoding` 指定了传感器的像素编码。对于单色传感器，选择 `TCVN_PE_NONE`。对于彩色偏振传感器，必须将相应的 Bayer 模式作为 `TCVN_PE_BAYER_XX` 传入。

选择索引

每个像素的四个偏振角度按强度排序。范围为 `0.0-3.0` 的参数 `fIndex` 决定了用于输出图像的强度值。如果指定了十进制值，则会对两个相邻索引或排序后的强度值计算加权平均值。建议先使用设置 `0.0`，因为这样可以实现最大程度的眩光减弱效果（但可能会丢失一些细节）。例如，`0.5` 会计算最小与次小强度值的平均值。不过，大于 `1.0` 的值通常会增强眩光，而非减弱眩光。

Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.2.3.2 许可证概述

以下列表显示了 TwinCAT Vision 功能与可用许可证的分配情况（见授权模式 [▶ 12]）。它显示了应用某些功能需要哪些许可证。

TF7100
始终需要基本许可证 TC3 Vision Base（TF7100）。

TC3 Vision Base

AdaptiveThreshold [▶ 2481]

AddContainers [▶ 1938]

AddImages [▶ 1949]

AddImagesWeighted [▶ 1950]

AddScalarToImage [▶ 1951]

AddToContainerElements (LONG) [▶ 1914]

AddToContainerElements (SHORT) [▶ 1915]

AddToContainerElements (double) [▶ 1915]

AddToContainerElements (float) [▶ 1916]

AddToContainerElements (char) [▶ 1917]

AddToContainerElements (TcVnKeyPoint) [▶ 1917]

AddToContainerElements (TcVnPoint2_DINT) [▶ 1918]

AddToContainerElements (TcVnPoint2_LREAL) [▶ 1919]

AddToContainerElements (TcVnPoint2_REAL) [▶ 1919]

AddToContainerElements (TcVnPoint3_LREAL) [▶ 1920]

AddToContainerElements (TcVnPoint3_REAL) [▶ 1921]

AddToContainerElements (TcVnRectangle_DINT) [▶ 1921]

AddToContainerElements (TcVnRotatedRectangle) [▶ 1937]

AddToContainerElements (TcVnVector2_DINT) [▶ 1922]

AddToContainerElements (TcVnVector2_INT) [▶ 1923]

AddToContainerElements (TcVnVector2_REAL) [▶ 1923]

AddToContainerElements (TcVnVector2_SINT) [▶ 1924]

AddToContainerElements (TcVnVector2_UINT) [▶ 1925]

AddToContainerElements (TcVnVector2_USINT) [▶ 1925]

AddToContainerElements (TcVnVector3_INT) [▶ 1926]

AddToContainerElements (TcVnVector3_REAL) [▶ 1927]

AddToContainerElements (TcVnVector3_SINT) [▶ 1927]

AddToContainerElements (TcVnVector3_UINT) [▶ 1928]

AddToContainerElements (TcVnVector3_USINT) [▶ 1929]
AddToContainerElements (TcVnVector4_DINT) [▶ 1929]
AddToContainerElements (TcVnVector4_INT) [▶ 1930]
AddToContainerElements (TcVnVector4_LREAL) [▶ 1931]
AddToContainerElements (TcVnVector4_REAL) [▶ 1936]
AddToContainerElements (TcVnVector4_SINT) [▶ 1931]
AddToContainerElements (TcVnVector4_UINT) [▶ 1932]
AddToContainerElements (TcVnVector4_USINT) [▶ 1933]
AddToContainerElements (ULONG) [▶ 1933]
AddToContainerElements (USHORT) [▶ 1934]
AddToContainerElements (ULONGLONG) [▶ 1935]
AddToContainerElements (unsigned char) [▶ 1935]
AddVectorToImage [▶ 1952]
AdjustActiveContour [▶ 2414]
AdvanceIterator [▶ 2136]
AlignRotatedImageRegion [▶ 2375]
AppendToContainer (LONG) [▶ 1983]
AppendToContainer (SHORT) [▶ 1983]
AppendToContainer (ITcVnContainer*) [▶ 1984]
AppendToContainer (ITcVnForwardIterator*) [▶ 1985]
AppendToContainer (ITcVnImage*) [▶ 1985]
AppendToContainer (double) [▶ 1986]
AppendToContainer (float) [▶ 1987]
AppendToContainer (char) [▶ 1987]
AppendToContainer (TcVnDMatch) [▶ 1988]
AppendToContainer (TcVnKeyPoint) [▶ 1989]
AppendToContainer (TcVnPoint2_DINT) [▶ 1989]
AppendToContainer (TcVnPoint2_LREAL) [▶ 1990]
AppendToContainer (TcVnPoint2_REAL) [▶ 1991]
AppendToContainer (TcVnPoint3_LREAL) [▶ 1991]
AppendToContainer (TcVnPoint3_REAL) [▶ 1992]
AppendToContainer (TcVnRectangle_DINT) [▶ 1993]
AppendToContainer (TcVnRotatedRectangle) [▶ 2008]
AppendToContainer (TcVnVector2_DINT) [▶ 1993]
AppendToContainer (TcVnVector2_INT) [▶ 1994]

AppendToContainer (TcVnVector2_REAL) [▶ 1995]
AppendToContainer (TcVnVector2_SINT) [▶ 1995]
AppendToContainer (TcVnVector2_UINT) [▶ 1996]
AppendToContainer (TcVnVector2_USINT) [▶ 1997]
AppendToContainer (TcVnVector3_INT) [▶ 1997]
AppendToContainer (TcVnVector3_REAL) [▶ 1998]
AppendToContainer (TcVnVector3_SINT) [▶ 1999]
AppendToContainer (TcVnVector3_UINT) [▶ 1999]
AppendToContainer (TcVnVector3_USINT) [▶ 2000]
AppendToContainer (TcVnVector4_DINT) [▶ 2001]
AppendToContainer (TcVnVector4_INT) [▶ 2001]
AppendToContainer (TcVnVector4_LREAL) [▶ 2002]
AppendToContainer (TcVnVector4_REAL) [▶ 2007]
AppendToContainer (TcVnVector4_SINT) [▶ 2003]
AppendToContainer (TcVnVector4_UINT) [▶ 2003]
AppendToContainer (TcVnVector4_USINT) [▶ 2004]
AppendToContainer (ULONG) [▶ 2005]
AppendToContainer (USHORT) [▶ 2005]
AppendToContainer (ULONGLONG) [▶ 2006]
AppendToContainer (unsigned char) [▶ 2007]
ApplyColorMap [▶ 2447]
ApplyLut [▶ 2448]
ApplyRotationToAffineTransformation [▶ 2376]
ApplyScalingToAffineTransformation [▶ 2377]
ApplyTranslationToAffineTransformation [▶ 2377]
ApplyYAxisInversionToAffineTransformation [▶ 2378]
ApproximatePolygon [▶ 2283]
BilateralFilter [▶ 2465]
BitwiseAndContainers [▶ 1939]
BitwiseAndImages [▶ 1952]
BitwiseAndImages (Exp) [▶ 1953]
BitwiseAndScalarWithImage [▶ 1954]
BitwiseAndVectorWithImage [▶ 1954]
BitwiseNotContainer [▶ 1939]
BitwiseNotImage [▶ 1955]

BitwiseNotImage (Exp) [▶ 1956]
BitwiseOrContainers [▶ 1940]
BitwiseOrImages [▶ 1956]
BitwiseOrImages (Exp) [▶ 1957]
BitwiseOrScalarWithImage [▶ 1958]
BitwiseOrVectorWithImage [▶ 1958]
BitwiseXorContainers [▶ 1941]
BitwiseXorImages [▶ 1959]
BitwiseXorImages (Exp) [▶ 1960]
BitwiseXorScalarWithImage [▶ 1961]
BitwiseXorVectorWithImage [▶ 1961]
BlendImages [▶ 1962]
BoxFilter [▶ 2466]
BrightBorderObjects [▶ 2467]
CalibrateCameraPlanar [▶ 2356]
CannyEdgeDetection [▶ 2430]
CheckColorRange [▶ 2482]
CheckFunctionInitialization [▶ 2570]
CheckIfEmpty [▶ 2137]
CheckIfIteratorIsAtEnd [▶ 2138]
CheckIfPointIsInsideContour [▶ 2283]
Clahe [▶ 2449]
ClipLineToBoundary [▶ 2571]
ClipLineToBoundary (ITcVnImage*) [▶ 2572]
ClipLineToBoundary (TcVnRectangle_DINT) [▶ 2572]
CombineImageChannels [▶ 2160]
ConnectedComponents [▶ 2431]
ConnectedComponentsWithStats [▶ 2432]
ContainerAverage [▶ 2275]
ContainerAverageElementwise (2) [▶ 2275]
ContainerAverageElementwise (3) [▶ 2276]
ContainerAverageElementwise (4) [▶ 2276]
ContainerAverageVariance [▶ 2277]
ContainerAverageVarianceElementwise (2) [▶ 2278]
ContainerAverageVarianceElementwise (3) [▶ 2278]

ContainerAverageVarianceElementwise (4) [▶ 2279]
ContourArea [▶ 2284]
ContourCenterOfMass [▶ 2285]
ContourCircularity [▶ 2286]
ContourConvexity [▶ 2286]
ContourEccentricity [▶ 2287]
ContourElongation [▶ 2288]
ContourExtremePoint [▶ 2289]
ContourInertiaRatio [▶ 2289]
ContourMoments [▶ 2290]
ContourOrientation [▶ 2291]
ContourPerimeter [▶ 2292]
ContourRoundness [▶ 2292]
ConvertCartesianToPolarAngleImage [▶ 2379]
ConvertCartesianToPolarAngles [▶ 2380]
ConvertCartesianToPolarImages [▶ 2381]
ConvertCartesianToPolarMagnitudeImage [▶ 2382]
ConvertCartesianToPolarMagnitudes [▶ 2382]
ConvertCartesianToPolarPoints [▶ 2383]
ConvertColorSpace [▶ 2450]
ConvertContainerType [▶ 2138]
ConvertDataLayout [▶ 2598]
ConvertElementType [▶ 2161]
ConvertITcUnknownToITcVnBitmapExport [▶ 2573]
ConvertITcUnknownToITcVnContainer [▶ 2574]
ConvertITcUnknownToITcVnImage [▶ 2574]
ConvertITcUnknownToITcVnMlModel [▶ 2575]
ConvertMaps [▶ 2384]
ConvertPolarToCartesianImages [▶ 2385]
ConvertPolarToCartesianPoints [▶ 2386]
ConvexHullPoints [▶ 2293]
ConvexityDefects [▶ 2294]
CopyContainer [▶ 2139]
CopyContainerElementsConditional (ITcVnContainer*) [▶ 2140]
CopyContainerElementsConditional (ITcVnForwardIterator*) [▶ 2140]

CopyImage [▶ 2161]

CopyImageRegion [▶ 2162]

CopyImageRegionToRegion [▶ 2163]

CopyIntoDisplayableImage [▶ 2164]

CountNonZeroPixels [▶ 2434]

CreateAssociatedImage [▶ 2165]

CreateBandpassButterworthFilter [▶ 2337]

CreateBandpassGaussianFilter [▶ 2338]

CreateBandrejectButterworthFilter [▶ 2339]

CreateBandrejectGaussianFilter [▶ 2340]

CreateContainer [▶ 2141]

CreateContainerFromArray [▶ 2142]

CreateEmptyImage [▶ 2165]

CreateHighpassButterworthFilter [▶ 2341]

CreateHighpassGaussianFilter [▶ 2342]

CreateImage [▶ 2166]

CreateImageAndSetPixels [▶ 2167]

CreateImageFromArray [▶ 2167]

CreateLowpassButterworthFilter [▶ 2343]

CreateLowpassGaussianFilter [▶ 2344]

CreateNDimensionalImage [▶ 2168]

CreateNDimensionalImageAndSetPixels [▶ 2169]

CreateNDimensionalImageFromArray [▶ 2170]

CreateStructuringElement [▶ 2467]

CustomElementWiseContainerOperation (ITcVnContainer*) [▶ 2295]

CustomElementWiseContainerOperation (ITcVnForwardIterator*) [▶ 2296]

CustomFilter [▶ 2468]

DarkBorderObjects [▶ 2469]

DecomposeAffineTransformation [▶ 2387]

DecomposeHomography [▶ 2388]

DeinitializeFunction [▶ 2576]

DetectBlobs [▶ 2415]

Dft [▶ 2345]

DistanceTransformation [▶ 2434]

DivideContainers [▶ 1941]

DivideImageByScalar [▶ 1963]

DivideImageByVector [▶ 1964]

DivideImages [▶ 1964]

DivideScalarByImage [▶ 1965]

DivideVectorByImage [▶ 1966]

DoubleThreshold [▶ 2482]

DrawArrow [▶ 2308]

DrawArrow (TcVnVector4_DINT) [▶ 2309]

DrawCircle [▶ 2310]

DrawCircles [▶ 2311]

DrawCircularArc [▶ 2312]

DrawComponents [▶ 2313]

DrawContours [▶ 2314]

DrawEllipse [▶ 2315]

DrawLine [▶ 2317]

DrawLine (TcVnVector4_DINT) [▶ 2318]

DrawLine (TcVnVector4_LREAL) [▶ 2319]

DrawLines [▶ 2320]

DrawOrientation [▶ 2322]

DrawOrientation (Exp) [▶ 2323]

DrawPoint [▶ 2324]

DrawPoints [▶ 2325]

DrawPolygon [▶ 2325]

DrawRectangle [▶ 2326]

DrawRectangle (TcVnRectangle_DINT) [▶ 2327]

DrawRectangle (TcVnRectangle_UDINT) [▶ 2328]

DrawRotatedRectangle [▶ 2329]

ElementwiseExp [▶ 1966]

ElementwiseLog [▶ 1967]

EnclosingCircle [▶ 2296]

EnclosingRectangle [▶ 2297]

EnclosingTriangle [▶ 2298]

EraseFromContainer [▶ 2142]

ExportContainer [▶ 2143]

ExportContainer_String [▶ 2144]

ExportContainerSize [▶ 2145]

ExportImage [▶ 2170]

ExportImageAsBmp [▶ 2171]

ExportImageAsBmpSize [▶ 2172]

ExportImageSize [▶ 2173]

ExportSubContainer [▶ 2145]

ExportSubContainer_String [▶ 2146]

ExportSubContainerSize [▶ 2147]

ExtractContainerRange [▶ 2147]

FillCircle [▶ 2329]

FillContainer (LONG) [▶ 2009]

FillContainer (SHORT) [▶ 2009]

FillContainer (ITcVnImage*) [▶ 2010]

FillContainer (double) [▶ 2010]

FillContainer (float) [▶ 2011]

FillContainer (char) [▶ 2012]

FillContainer (TcVnDMatch) [▶ 2012]

FillContainer (TcVnKeyPoint) [▶ 2013]

FillContainer (TcVnPoint2_DINT) [▶ 2014]

FillContainer (TcVnPoint2_LREAL) [▶ 2014]

FillContainer (TcVnPoint2_REAL) [▶ 2015]

FillContainer (TcVnPoint3_LREAL) [▶ 2016]

FillContainer (TcVnPoint3_REAL) [▶ 2016]

FillContainer (TcVnRectangle_DINT) [▶ 2017]

FillContainer (TcVnRotatedRectangle) [▶ 2032]

FillContainer (TcVnVector2_DINT) [▶ 2018]

FillContainer (TcVnVector2_INT) [▶ 2018]

FillContainer (TcVnVector2_REAL) [▶ 2019]

FillContainer (TcVnVector2_SINT) [▶ 2020]

FillContainer (TcVnVector2_UINT) [▶ 2020]

FillContainer (TcVnVector2_USINT) [▶ 2021]

FillContainer (TcVnVector3_INT) [▶ 2022]

FillContainer (TcVnVector3_REAL) [▶ 2022]

FillContainer (TcVnVector3_SINT) [▶ 2023]

FillContainer (TcVnVector3_UINT) [▶ 2024]

FillContainer (TcVnVector3_USINT) [▶ 2024]
FillContainer (TcVnVector4_DINT) [▶ 2025]
FillContainer (TcVnVector4_INT) [▶ 2026]
FillContainer (TcVnVector4_LREAL) [▶ 2026]
FillContainer (TcVnVector4_REAL) [▶ 2032]
FillContainer (TcVnVector4_SINT) [▶ 2027]
FillContainer (TcVnVector4_UINT) [▶ 2028]
FillContainer (TcVnVector4_USINT) [▶ 2028]
FillContainer (ULONG) [▶ 2029]
FillContainer (USHORT) [▶ 2030]
FillContainer (ULONGLONG) [▶ 2030]
FillContainer (unsigned char) [▶ 2031]
FillContainer (LONG) [▶ 2033]
FillContainer (SHORT) [▶ 2034]
FillContainer (ITcVnImage*) [▶ 2034]
FillContainer (double) [▶ 2035]
FillContainer (float) [▶ 2036]
FillContainer (char) [▶ 2036]
FillContainer (TcVnDMatch) [▶ 2037]
FillContainer (TcVnKeyPoint) [▶ 2038]
FillContainer (TcVnPoint2_DINT) [▶ 2039]
FillContainer (TcVnPoint2_LREAL) [▶ 2039]
FillContainer (TcVnPoint2_REAL) [▶ 2040]
FillContainer (TcVnPoint3_LREAL) [▶ 2041]
FillContainer (TcVnPoint3_REAL) [▶ 2041]
FillContainer (TcVnRectangle_DINT) [▶ 2042]
FillContainer (TcVnRotatedRectangle) [▶ 2058]
FillContainer (TcVnVector2_DINT) [▶ 2043]
FillContainer (TcVnVector2_INT) [▶ 2043]
FillContainer (TcVnVector2_REAL) [▶ 2044]
FillContainer (TcVnVector2_SINT) [▶ 2045]
FillContainer (TcVnVector2_UINT) [▶ 2045]
FillContainer (TcVnVector2_USINT) [▶ 2046]
FillContainer (TcVnVector3_INT) [▶ 2047]
FillContainer (TcVnVector3_REAL) [▶ 2048]

FillContainer (TcVnVector3_SINT) [▶ 2048]
FillContainer (TcVnVector3_UINT) [▶ 2049]
FillContainer (TcVnVector3_USINT) [▶ 2050]
FillContainer (TcVnVector4_DINT) [▶ 2050]
FillContainer (TcVnVector4_INT) [▶ 2051]
FillContainer (TcVnVector4_LREAL) [▶ 2052]
FillContainer (TcVnVector4_REAL) [▶ 2057]
FillContainer (TcVnVector4_SINT) [▶ 2052]
FillContainer (TcVnVector4_UINT) [▶ 2053]
FillContainer (TcVnVector4_USINT) [▶ 2054]
FillContainer (ULONG) [▶ 2054]
FillContainer (USHORT) [▶ 2055]
FillContainer (ULONGLONG) [▶ 2056]
FillContainer (unsigned char) [▶ 2057]
FillContours [▶ 2330]
FillEllipse [▶ 2331]
FillHoles [▶ 2470]
FillPolygon [▶ 2332]
FillRectangle [▶ 2332]
FillRotatedRectangle [▶ 2333]
FindContourHierarchy (Exp) [▶ 2416]
FindContours [▶ 2417]
FitEllipse [▶ 2299]
FitLine [▶ 2299]
FlipImage [▶ 2390]
FourierDescriptors [▶ 2300]
FuseImages [▶ 2174]
FuseImagesArray [▶ 2175]
GaussianFilter [▶ 2470]
GenerateAffineTransformationUnitMatrix2D [▶ 2390]
GenerateColorMap [▶ 2451]
GenerateCustomColorMap [▶ 2452]
GetAffineTransformation [▶ 2391]
GetAffineTransformation2D [▶ 2392]
GetAt (LONG) [▶ 2059]

GetAt (SHORT) [▶ 2059]

GetAt (ITcVnContainer*) [▶ 2060]

GetAt (ITcVnImage*) [▶ 2061]

GetAt (double) [▶ 2061]

GetAt (float) [▶ 2062]

GetAt (char) [▶ 2063]

GetAt (TcVnDMatch) [▶ 2063]

GetAt (TcVnKeyPoint) [▶ 2064]

GetAt (TcVnPoint2_DINT) [▶ 2065]

GetAt (TcVnPoint2_LREAL) [▶ 2065]

GetAt (TcVnPoint2_REAL) [▶ 2066]

GetAt (TcVnPoint3_LREAL) [▶ 2067]

GetAt (TcVnPoint3_REAL) [▶ 2067]

GetAt (TcVnRectangle_DINT) [▶ 2068]

GetAt (TcVnRotatedRectangle) [▶ 2083]

GetAt (TcVnVector2_DINT) [▶ 2069]

GetAt (TcVnVector2_INT) [▶ 2069]

GetAt (TcVnVector2_REAL) [▶ 2070]

GetAt (TcVnVector2_SINT) [▶ 2071]

GetAt (TcVnVector2_UINT) [▶ 2071]

GetAt (TcVnVector2_USINT) [▶ 2072]

GetAt (TcVnVector3_INT) [▶ 2073]

GetAt (TcVnVector3_REAL) [▶ 2073]

GetAt (TcVnVector3_SINT) [▶ 2074]

GetAt (TcVnVector3_UINT) [▶ 2075]

GetAt (TcVnVector3_USINT) [▶ 2075]

GetAt (TcVnVector4_DINT) [▶ 2076]

GetAt (TcVnVector4_INT) [▶ 2077]

GetAt (TcVnVector4_LREAL) [▶ 2077]

GetAt (TcVnVector4_REAL) [▶ 2083]

GetAt (TcVnVector4_SINT) [▶ 2078]

GetAt (TcVnVector4_UINT) [▶ 2079]

GetAt (TcVnVector4_USINT) [▶ 2079]

GetAt (ULONG) [▶ 2080]

GetAt (USHORT) [▶ 2081]

GetAt (ULONGLONG) [▶ 2081]

GetAt (unsigned char) [▶ 2082]

GetConnectedComponent [▶ 2435]

GetContainer [▶ 2148]

GetContainer (Exp) [▶ 2149]

GetForwardIterator [▶ 2149]

GetImageChannel [▶ 2175]

GetImageHeight [▶ 2176]

GetImageInfo [▶ 2177]

GetImageWidth [▶ 2177]

GetLayoutInfo [▶ 2178]

GetNumberOfElements [▶ 2150]

GetPerspectiveTransformation [▶ 2393]

GetPixel [▶ 2179]

GetPixelFormat [▶ 2179]

GetRandomAccessIterator [▶ 2151]

GetRoi [▶ 2180]

GetTimestamp [▶ 2576]

Histogram [▶ 2452]

HistogramEqualization [▶ 2453]

Homography [▶ 2394]

HoughCircles [▶ 2422]

HoughLines [▶ 2423]

HoughLinesP [▶ 2424]

HuMomentInvariants [▶ 2577]

ImageAbsoluteAverage [▶ 2443]

ImageAverage [▶ 2436]

ImageAverageStdDev [▶ 2437]

ImageCenterOfMass [▶ 2438]

ImageMedian [▶ 2439]

ImageMoments [▶ 2439]

IncrementIterator [▶ 2151]

InitMatrixStruct [▶ 2578]

InsertIntoContainer (LONG) [▶ 2084]

InsertIntoContainer (SHORT) [▶ 2085]

InsertIntoContainer (ITcVnContainer*) [▶ 2085]
InsertIntoContainer (ITcVnForwardIterator*) [▶ 2086]
InsertIntoContainer (ITcVnImage*) [▶ 2087]
InsertIntoContainer (double) [▶ 2087]
InsertIntoContainer (float) [▶ 2088]
InsertIntoContainer (char) [▶ 2089]
InsertIntoContainer (TcVnDMatch) [▶ 2089]
InsertIntoContainer (TcVnKeyPoint) [▶ 2090]
InsertIntoContainer (TcVnPoint2_DINT) [▶ 2091]
InsertIntoContainer (TcVnPoint2_LREAL) [▶ 2091]
InsertIntoContainer (TcVnPoint2_REAL) [▶ 2092]
InsertIntoContainer (TcVnPoint3_LREAL) [▶ 2093]
InsertIntoContainer (TcVnPoint3_REAL) [▶ 2093]
InsertIntoContainer (TcVnRectangle_DINT) [▶ 2094]
InsertIntoContainer (TcVnRotatedRectangle) [▶ 2109]
InsertIntoContainer (TcVnVector2_DINT) [▶ 2095]
InsertIntoContainer (TcVnVector2_INT) [▶ 2095]
InsertIntoContainer (TcVnVector2_REAL) [▶ 2096]
InsertIntoContainer (TcVnVector2_SINT) [▶ 2097]
InsertIntoContainer (TcVnVector2_UINT) [▶ 2097]
InsertIntoContainer (TcVnVector2_USINT) [▶ 2098]
InsertIntoContainer (TcVnVector3_INT) [▶ 2099]
InsertIntoContainer (TcVnVector3_REAL) [▶ 2099]
InsertIntoContainer (TcVnVector3_SINT) [▶ 2100]
InsertIntoContainer (TcVnVector3_UINT) [▶ 2101]
InsertIntoContainer (TcVnVector3_USINT) [▶ 2101]
InsertIntoContainer (TcVnVector4_DINT) [▶ 2102]
InsertIntoContainer (TcVnVector4_INT) [▶ 2103]
InsertIntoContainer (TcVnVector4_LREAL) [▶ 2103]
InsertIntoContainer (TcVnVector4_REAL) [▶ 2109]
InsertIntoContainer (TcVnVector4_SINT) [▶ 2104]
InsertIntoContainer (TcVnVector4_UINT) [▶ 2105]
InsertIntoContainer (TcVnVector4_USINT) [▶ 2105]
InsertIntoContainer (ULONG) [▶ 2106]
InsertIntoContainer (USHORT) [▶ 2107]

InsertIntoContainer (ULONGLONG) [▶ 2107]

InsertIntoContainer (unsigned char) [▶ 2108]

InverseDft [▶ 2346]

InvertAffineTransformation [▶ 2395]

InvertImageColor [▶ 2454]

InvertMatrix3x3 [▶ 2578]

IteratorDistance [▶ 2152]

LaplacianFilter [▶ 2471]

LineIntersectionPoint [▶ 2579]

LineIntersectionPointAndAngle [▶ 2580]

LocalMaxima [▶ 2472]

LocalMinima [▶ 2473]

MaxContainer [▶ 1942]

MaxElement [▶ 2280]

MaxElementElementwise (LONG) [▶ 2211]

MaxElementElementwise (SHORT) [▶ 2211]

MaxElementElementwise (double) [▶ 2212]

MaxElementElementwise (float) [▶ 2213]

MaxElementElementwise (char) [▶ 2213]

MaxElementElementwise (TcVnPoint2_DINT) [▶ 2214]

MaxElementElementwise (TcVnPoint2_LREAL) [▶ 2215]

MaxElementElementwise (TcVnPoint2_REAL) [▶ 2215]

MaxElementElementwise (TcVnPoint3_LREAL) [▶ 2216]

MaxElementElementwise (TcVnPoint3_REAL) [▶ 2217]

MaxElementElementwise (TcVnVector2_DINT) [▶ 2217]

MaxElementElementwise (TcVnVector2_INT) [▶ 2218]

MaxElementElementwise (TcVnVector2_REAL) [▶ 2219]

MaxElementElementwise (TcVnVector2_SINT) [▶ 2219]

MaxElementElementwise (TcVnVector2_UINT) [▶ 2220]

MaxElementElementwise (TcVnVector2_USINT) [▶ 2221]

MaxElementElementwise (TcVnVector3_INT) [▶ 2221]

MaxElementElementwise (TcVnVector3_REAL) [▶ 2222]

MaxElementElementwise (TcVnVector3_SINT) [▶ 2223]

MaxElementElementwise (TcVnVector3_UINT) [▶ 2223]

MaxElementElementwise (TcVnVector3_USINT) [▶ 2224]

MaxElementElementwise (TcVnVector4_DINT) [▶ 2225]
MaxElementElementwise (TcVnVector4_INT) [▶ 2225]
MaxElementElementwise (TcVnVector4_LREAL) [▶ 2226]
MaxElementElementwise (TcVnVector4_REAL) [▶ 2231]
MaxElementElementwise (TcVnVector4_SINT) [▶ 2227]
MaxElementElementwise (TcVnVector4_UINT) [▶ 2227]
MaxElementElementwise (TcVnVector4_USINT) [▶ 2228]
MaxElementElementwise (ULONG) [▶ 2229]
MaxElementElementwise (USHORT) [▶ 2229]
MaxElementElementwise (ULONGLONG) [▶ 2230]
MaxElementElementwise (unsigned char) [▶ 2231]
MaxImage [▶ 1968]
MaxImage (Exp) [▶ 1968]
MaxImageWithScalar [▶ 1969]
MaxImageWithScalar (Exp) [▶ 1970]
MaxImageWithVector [▶ 1970]
MaxImageWithVector (Exp) [▶ 1971]
MaxPixelValue [▶ 2440]
MedianElement [▶ 2280]
MedianElementElementwise (LONG) [▶ 2232]
MedianElementElementwise (SHORT) [▶ 2233]
MedianElementElementwise (double) [▶ 2233]
MedianElementElementwise (float) [▶ 2234]
MedianElementElementwise (char) [▶ 2235]
MedianElementElementwise (TcVnPoint2_DINT) [▶ 2235]
MedianElementElementwise (TcVnPoint2_LREAL) [▶ 2236]
MedianElementElementwise (TcVnPoint2_REAL) [▶ 2237]
MedianElementElementwise (TcVnPoint3_LREAL) [▶ 2237]
MedianElementElementwise (TcVnPoint3_REAL) [▶ 2238]
MedianElementElementwise (TcVnVector2_DINT) [▶ 2239]
MedianElementElementwise (TcVnVector2_INT) [▶ 2239]
MedianElementElementwise (TcVnVector2_REAL) [▶ 2240]
MedianElementElementwise (TcVnVector2_SINT) [▶ 2241]
MedianElementElementwise (TcVnVector2_UINT) [▶ 2241]
MedianElementElementwise (TcVnVector2_USINT) [▶ 2242]

MedianElementElementwise (TcVnVector3_INT) [▶ 2243]
MedianElementElementwise (TcVnVector3_REAL) [▶ 2243]
MedianElementElementwise (TcVnVector3_SINT) [▶ 2244]
MedianElementElementwise (TcVnVector3_UINT) [▶ 2245]
MedianElementElementwise (TcVnVector3_USINT) [▶ 2245]
MedianElementElementwise (TcVnVector4_DINT) [▶ 2246]
MedianElementElementwise (TcVnVector4_INT) [▶ 2247]
MedianElementElementwise (TcVnVector4_LREAL) [▶ 2247]
MedianElementElementwise (TcVnVector4_REAL) [▶ 2253]
MedianElementElementwise (TcVnVector4_SINT) [▶ 2248]
MedianElementElementwise (TcVnVector4_UINT) [▶ 2249]
MedianElementElementwise (TcVnVector4_USINT) [▶ 2249]
MedianElementElementwise (ULONG) [▶ 2250]
MedianElementElementwise (USHORT) [▶ 2251]
MedianElementElementwise (ULONGLONG) [▶ 2251]
MedianElementElementwise (unsigned char) [▶ 2252]
MedianFilter [▶ 2474]
MinContainer [▶ 1943]
MinElement [▶ 2281]
MinElementElementwise (LONG) [▶ 2253]
MinElementElementwise (SHORT) [▶ 2254]
MinElementElementwise (double) [▶ 2255]
MinElementElementwise (float) [▶ 2255]
MinElementElementwise (char) [▶ 2256]
MinElementElementwise (TcVnPoint2_DINT) [▶ 2257]
MinElementElementwise (TcVnPoint2_LREAL) [▶ 2257]
MinElementElementwise (TcVnPoint2_REAL) [▶ 2258]
MinElementElementwise (TcVnPoint3_LREAL) [▶ 2259]
MinElementElementwise (TcVnPoint3_REAL) [▶ 2259]
MinElementElementwise (TcVnVector2_DINT) [▶ 2260]
MinElementElementwise (TcVnVector2_INT) [▶ 2261]
MinElementElementwise (TcVnVector2_REAL) [▶ 2261]
MinElementElementwise (TcVnVector2_SINT) [▶ 2262]
MinElementElementwise (TcVnVector2_UINT) [▶ 2263]
MinElementElementwise (TcVnVector2_USINT) [▶ 2263]

MinElementElementwise (TcVnVector3_INT) [▶ 2264]
MinElementElementwise (TcVnVector3_REAL) [▶ 2265]
MinElementElementwise (TcVnVector3_SINT) [▶ 2265]
MinElementElementwise (TcVnVector3_UINT) [▶ 2266]
MinElementElementwise (TcVnVector3_USINT) [▶ 2267]
MinElementElementwise (TcVnVector4_DINT) [▶ 2267]
MinElementElementwise (TcVnVector4_INT) [▶ 2268]
MinElementElementwise (TcVnVector4_LREAL) [▶ 2269]
MinElementElementwise (TcVnVector4_REAL) [▶ 2274]
MinElementElementwise (TcVnVector4_SINT) [▶ 2269]
MinElementElementwise (TcVnVector4_UINT) [▶ 2270]
MinElementElementwise (TcVnVector4_USINT) [▶ 2271]
MinElementElementwise (ULONG) [▶ 2271]
MinElementElementwise (USHORT) [▶ 2272]
MinElementElementwise (ULONGLONG) [▶ 2273]
MinElementElementwise (unsigned char) [▶ 2273]
MinImage [▶ 1972]
MinImage (Exp) [▶ 1973]
MinImageWithScalar [▶ 1973]
MinImageWithScalar (Exp) [▶ 1974]
MinImageWithVector [▶ 1975]
MinImageWithVector (Exp) [▶ 1975]
MinPixelValue [▶ 2441]
MixImageChannels [▶ 2181]
MorphologicalOperator [▶ 2474]
MultiplyContainers [▶ 1943]
MultiplyImages [▶ 1976]
MultiplyImageWithScalar [▶ 1977]
MultiplyImageWithVector [▶ 1977]
MultiplyMatrices [▶ 2581]
MultiplyWithContainerElements (1) [▶ 1944]
MultiplyWithContainerElements (2) [▶ 1945]
MultiplyWithContainerElements (3) [▶ 1945]
MultiplyWithContainerElements (4) [▶ 1946]
NegateContainer [▶ 1947]

NonMaxSuppression [▶ 2603]

NormalizeImage [▶ 2455]

NormalizeImageForDisplay [▶ 2456]

OptimalDftSize [▶ 2347]

OrthogonalVector (TcVnVector2_DINT) [▶ 2584]

OrthogonalVector (TcVnVector2_LREAL) [▶ 2585]

OrthogonalVector (TcVnVector2_REAL) [▶ 2585]

PadImageBorder [▶ 2181]

PerspectiveTransformation [▶ 2395]

PlotIntensityProfile [▶ 2334]

PointToLineDistance (TcVnPoint2_DINT) [▶ 2586]

PointToLineDistance (TcVnPoint2_LREAL) [▶ 2587]

PointToLineDistance (TcVnPoint2_REAL) [▶ 2587]

PointToPointDirection (TcVnPoint2_DINT) [▶ 2588]

PointToPointDirection (TcVnPoint2_LREAL) [▶ 2589]

PointToPointDirection (TcVnPoint2_REAL) [▶ 2589]

PointToPointDistance (TcVnPoint2_DINT) [▶ 2590]

PointToPointDistance (TcVnPoint2_LREAL) [▶ 2591]

PointToPointDistance (TcVnPoint2_REAL) [▶ 2591]

PutLabel [▶ 2335]

PutText [▶ 2336]

PyramidDown [▶ 2396]

PyramidUp [▶ 2397]

Reduce [▶ 2604]

ReduceArg [▶ 2605]

ReferenceColorSimilarity (ITcVnColorModel*) [▶ 2457]

ReferenceColorSimilarity (ITcVnMlModel*) [▶ 2458]

ReferenceColorSimilarity (TcVnVector3_LREAL) [▶ 2459]

RegionOrientation [▶ 2442]

ReinterpretUnsupportedImage [▶ 2182]

RemapImage [▶ 2397]

RemapImageToLogPolarSpace [▶ 2399]

RemapImageToPolarSpace [▶ 2400]

RemoveLocalMaxima [▶ 2475]

RemoveLocalMinima [▶ 2476]

ReserveContainerMemory [▶ 2153]

ResetRoi [▶ 2183]

ReshapeImage [▶ 2605]

ResizeImage [▶ 2401]

ResizeRegion (TcVnRectangle_DINT) [▶ 2402]

ReverseContainer [▶ 2153]

RotatedRectangleCorners [▶ 2581]

RotatedRectangleIntersection [▶ 2582]

RotateImage [▶ 2403]

RotateImage (Exp) [▶ 2404]

SauvolaThreshold [▶ 2485]

ScharrFilter [▶ 2476]

SeparableCustomFilter [▶ 2477]

SetAt (LONG) [▶ 2110]

SetAt (SHORT) [▶ 2111]

SetAt (ITcVnContainer*) [▶ 2111]

SetAt (ITcVnImage*) [▶ 2112]

SetAt (double) [▶ 2113]

SetAt (float) [▶ 2114]

SetAt (char) [▶ 2114]

SetAt (TcVnDMatch) [▶ 2115]

SetAt (TcVnKeyPoint) [▶ 2116]

SetAt (TcVnPoint2_DINT) [▶ 2116]

SetAt (TcVnPoint2_LREAL) [▶ 2117]

SetAt (TcVnPoint2_REAL) [▶ 2118]

SetAt (TcVnPoint3_LREAL) [▶ 2118]

SetAt (TcVnPoint3_REAL) [▶ 2119]

SetAt (TcVnRectangle_DINT) [▶ 2120]

SetAt (TcVnRotatedRectangle) [▶ 2136]

SetAt (TcVnVector2_DINT) [▶ 2120]

SetAt (TcVnVector2_INT) [▶ 2121]

SetAt (TcVnVector2_REAL) [▶ 2122]

SetAt (TcVnVector2_SINT) [▶ 2123]

SetAt (TcVnVector2_UINT) [▶ 2123]

SetAt (TcVnVector2_USINT) [▶ 2124]

SetAt (TcVnVector3_INT) [▶ 2125]
SetAt (TcVnVector3_REAL) [▶ 2125]
SetAt (TcVnVector3_SINT) [▶ 2126]
SetAt (TcVnVector3_UINT) [▶ 2127]
SetAt (TcVnVector3_USINT) [▶ 2127]
SetAt (TcVnVector4_DINT) [▶ 2128]
SetAt (TcVnVector4_INT) [▶ 2129]
SetAt (TcVnVector4_LREAL) [▶ 2129]
SetAt (TcVnVector4_REAL) [▶ 2135]
SetAt (TcVnVector4_SINT) [▶ 2130]
SetAt (TcVnVector4_UINT) [▶ 2131]
SetAt (TcVnVector4_USINT) [▶ 2132]
SetAt (ULONG) [▶ 2132]
SetAt (USHORT) [▶ 2133]
SetAt (ULONGLONG) [▶ 2134]
SetAt (unsigned char) [▶ 2134]
SetContainer [▶ 2154]
SetImageChannel [▶ 2184]
SetIteratorToBegin [▶ 2155]
SetPixel [▶ 2184]
SetPixels [▶ 2185]
SetRngSeed [▶ 2583]
SetRoi [▶ 2186]
SetRoi (TcVnRectangle_DINT) [▶ 2187]
SetRoi (TcVnRectangle_UDINT) [▶ 2187]
Sigmoid [▶ 2606]
SobelFilter [▶ 2478]
SoftMax [▶ 2607]
SplitImageChannels [▶ 2188]
StartAbsWatchdog [▶ 2305]
StartRelWatchdog [▶ 2306]
StopWatchdog [▶ 2306]
SubtractContainers [▶ 1947]
SubtractImageFromScalar [▶ 1978]
SubtractImageFromVector [▶ 1979]

SubtractImages [▶ 1980]

SubtractScalarFromImage [▶ 1980]

SubtractVectorFromImage [▶ 1981]

Threshold [▶ 2483]

TrainImageColor [▶ 2460]

TrainImageColor (ITcVnMlModel*) [▶ 2462]

TransformCoordinatesPlanar (Container) [▶ 2372]

TransformCoordinatesPlanar (Point) [▶ 2373]

TransformIntoDisplayableImage [▶ 2189]

UpdateTimestamp [▶ 2583]

UprightBoundingRectangle [▶ 2304]

UprightInnerRectangle [▶ 2443]

VarianceFilter [▶ 2479]

WarpAffine [▶ 2405]

WarpAffine (Container) [▶ 2405]

WarpAffine (Point) [▶ 2406]

WarpAffine (Rectangle) [▶ 2407]

WarpAffine (Exp) [▶ 2408]

WarpPerspective [▶ 2409]

WarpPerspective (Container) [▶ 2409]

WarpPerspective (Point) [▶ 2410]

WarpPerspective (Rectangle) [▶ 2411]

WarpPerspective (Exp) [▶ 2412]

WatershedSegmentation (Exp) [▶ 2484]

WhiteBalance [▶ 2463]

TC3 Vision Code Reading

DetectCodesNeuralNetwork [▶ 2199]

ReadBarcode [▶ 2200]

ReadBarcode (Exp) [▶ 2200]

ReadBarcodeRoi [▶ 2201]

ReadDataMatrixCode [▶ 2204]

ReadDataMatrixCodeRoi [▶ 2205]

ReadDotCodeRoi [▶ 2206]

ReadPharmaCode [▶ 2208]

ReadQRCode [▶ 2209]

TC3 Vision Code Quality

GradeBarcode [▶ 2190]

GradeDataMatrixCode [▶ 2192]

GradeQRCode [▶ 2193]

TC3 Vision Metrology 2D

AdjustSearchWindowOrientationToLinearEdge [▶ 2550]

CalibrateCamera [▶ 2352]

CalibrateCamera (Exp) [▶ 2353]

CalibrateCameraManually [▶ 2355]

CalibrateLinescanCamera [▶ 2357]

ClosestPointsBF [▶ 2551]

CompensateLensDistortion [▶ 2359]

CompensateLensDistortion (Exp1) [▶ 2360]

CompensateLensDistortion (Exp2) [▶ 2361]

CompensateLensDistortionForPoints [▶ 2361]

CompensateLensDistortionForPoints (Exp1) [▶ 2362]

CompensateLensDistortionForPoints (Exp2) [▶ 2363]

DetectPatternPoints [▶ 2364]

DetectPatternPoints (2) [▶ 2365]

GenerateCalibrationPatternReferencePoints [▶ 2366]

ImagePointsWorldDistance [▶ 2367]

LocateCircularArc [▶ 2552]

LocateEdge [▶ 2554]

LocateEdges [▶ 2556]

LocateEllipse [▶ 2558]

MeasureAngleBetweenEdges [▶ 2560]

MeasureEdgeDistance [▶ 2562]

MeasureMinEdgeDistance [▶ 2564]

SolvePnP [▶ 2368]

SortDetectedPatternPoints [▶ 2369]

TransformCoordinatesImageToWorld (Container) [▶ 2370]

TransformCoordinatesImageToWorld (Point) [▶ 2371]

TransformCoordinatesWorldToImage (Container) [▶ 2373]

TransformCoordinatesWorldToImage (Point) [▶ 2374]

TC3 Vision Matching

DrawKeypoints [▶ 2316]

DrawMatches [▶ 2321]

FilterGoodMatches [▶ 2486]

FindReferenceKeyPointsInImage [▶ 2487]

FindReferenceKeyPointsInImageAKAZE [▶ 2489]

FindReferenceKeyPointsInImageBRISK [▶ 2491]

FindReferenceKeyPointsInImageORB [▶ 2493]

FindTemplateWalsh [▶ 2418]

FindTemplateWalsh (AngleRange) [▶ 2420]

GetMatchCoordinates [▶ 2495]

KeyPointsAGAST [▶ 2496]

KeyPointsAndDescriptorsAKAZE [▶ 2497]

KeyPointsAndDescriptorsBRISK [▶ 2498]

KeyPointsAndDescriptorsKAZE [▶ 2498]

KeyPointsAndDescriptorsORB [▶ 2499]

KeyPointsFAST [▶ 2500]

KeyPointsGFTT [▶ 2501]

KeyPointsMSER [▶ 2502]

KeyPointsSB [▶ 2503]

MatchContours [▶ 2301]

MatchContours1vsN [▶ 2303]

MatchDescriptorsBF [▶ 2504]

MatchDescriptorsFlannLsh [▶ 2505]

MatchDescriptorsKnnBF [▶ 2506]

MatchDescriptorsKnnFlannLsh [▶ 2507]

MatchImageHuMoments [▶ 2425]

MatchTemplate [▶ 2426]

MatchTemplateAndEvaluate [▶ 2427]

MatchTemplateWalsh [▶ 2429]

RegionsMSER [▶ 2508]

TC3 Vision OCR

OCR [▶ 2610]

TC3 Machine Learning Realtime Inference

CreateBoostClassifier [▶ 2511]

CreateKmppModel [▶ 2512]

CreateKnnModel [▶ 2513]

CreateLbgModel [▶ 2513]

CreateLdaTransform [▶ 2515]

CreateLdaTransformViaComponentNum [▶ 2515]

CreateNbcModel [▶ 2516]

CreatePcaTransform [▶ 2517]

CreatePcaTransformViaComponentNum [▶ 2518]

CreatePcaTransformViaVariance [▶ 2519]

CreateRTreesModel [▶ 2519]

CreateStaModel [▶ 2521]

CreateSvmModel [▶ 2522]

CreateSvmSgdClassifier [▶ 2524]

FeatureScaling [▶ 2525]

FeatureTransform [▶ 2526]

GetBatchClusters [▶ 2527]

GetBatchNovelties [▶ 2528]

GetClusterCenter [▶ 2529]

GetClusterNum [▶ 2530]

GetFeatureScales [▶ 2530]

GetSampleCluster [▶ 2531]

GetSampleNovelty [▶ 2532]

Granulometry [▶ 2611]

HaralickFeatures [▶ 2612]

InverseFeatureScaling [▶ 2533]

InverseFeatureScaling (float) [▶ 2534]

InverseFeatureTransform [▶ 2535]

PredictBatch [▶ 2536]

PredictSampleClass [▶ 2537]

PredictSampleScalar [▶ 2538]

PredictSampleVector [▶ 2539]

TrainBatch [▶ 2540]

TrainBatchClusters [▶ 2541]

TrainSample [▶ 2542]

TrainSampleClass [▶ 2542]

TrainSampleCluster [▶ 2543]

TrainSampleScalar [▶ 2544]

TrainSampleVector [▶ 2544]

TC3 Neural Network Realtime Inference

ExecuteNeuralNetwork [▶ 2599]

ExecuteNeuralNetwork (MI) [▶ 2593]

ExecuteNeuralNetwork (MIMO_String) [▶ 2594]

ExecuteNeuralNetwork (MISO) [▶ 2594]

ExecuteNeuralNetwork (MO) [▶ 2595]

ExecuteNeuralNetwork_MultiLayerOutput [▶ 2600]

ExecuteNeuralNetwork_SingleLayerOutput [▶ 2601]

ExecuteNeuralNetwork (SO) [▶ 2596]

GetNeuralNetworkLayerNames [▶ 2602]

GetNeuralNetworkLayerNames (Exp) [▶ 2597]

GetNeuralNetworkLayerNames (String) [▶ 2598]

还请参阅有关此

- AngleAndDegreeOfLinearPolarization [▶ 1883]
- DemosaicPolarizedImage [▶ 1885]
- ExportImageAsJpeg [▶ 2159]
- FindRois [▶ 2444]
- InnerCircle [▶ 2445]
- MergeRegions [▶ 2349]
- PolarizedImageGlareReduction [▶ 1887]
- SalientFeatures [▶ 2446]
- SortContainer [▶ 2155]
- AnalyzeBarcodeWalsh [▶ 2197]
- DetectBarcodesWalsh [▶ 2198]
- LocateAxisAlignedEdges [▶ 2566]
- SortAxisAlignedPatternPoints [▶ 2351]
- GetTransformParameters [▶ 2545]

6.2.3.3 容器代数运算

该组包含将代数算子按元素应用于容器 [▶ 146]的函数。

函数

代数

- AddContainers [▶ 1938]

- AddToContainerElements [▶ 1914]
- DivideContainers [▶ 1941]
- MultiplyContainers [▶ 1943]
- MultiplyWithContainerElements [▶ 1944]
- NegateContainer [▶ 1947]
- SubtractContainers [▶ 1947]

位运算符

- BitwiseAndContainers [▶ 1939]
- BitwiseNotContainer [▶ 1939]
- BitwiseOrContainers [▶ 1940]
- BitwiseXorContainers [▶ 1941]

统计学功能

- MaxContainer [▶ 1942]
- MinContainer [▶ 1943]

6.2.3.3.1 AddToContainerElements

本章包含根据数据类型向每个容器元素添加数值的函数。

6.2.3.3.1.1 AddToContainerElements (LONG)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT          hrPrev,
    LONG             nValue,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nValue	LONG	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 DINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.2 AddToContainerElements (SHORT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT         hrPrev,
    SHORT           nValue,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nValue	SHORT	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 INT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.3 AddToContainerElements (double)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT         hrPrev,
    double          fValue,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fValue	double	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 LREAL 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.4 AddToContainerElements (float)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT         hrPrev,
    float           fValue,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fValue	float	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 REAL 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.5 AddToContainerElements (char)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT         hrPrev,
    char            nValue,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nValue	char	值
ipContainer	ITcVnContainer* [▶ 1827]	装有 SINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.6 AddToContainerElements (TcVnKeyPoint)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT         hrPrev,
    TcVnKeyPoint&   stValue,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stValue	TcVnKeyPoint [▶ 1706]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnKeyPoint 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.7 AddToContainerElements (TcVnPoint2_DINT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT          hrPrev,
    TcVnPoint2_DINT& aValue,
    ITcVnContainer*  ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnPoint2_DINT [▶ 1640]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnPoint2_DINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.8 AddToContainerElements (TcVnPoint2_LREAL)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT           hrPrev,
    TcVnPoint2_LREAL& aValue,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnPoint2_LREAL [▶ 1640]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnPoint2_LREAL 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.9 AddToContainerElements (TcVnPoint2_REAL)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT           hrPrev,
    TcVnPoint2_REAL& aValue,
    ITcVnContainer*  ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnPoint2_REAL [▶ 1640]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnPoint2_REAL 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.10 AddToContainerElements (TcVnPoint3_LREAL)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT           hrPrev,
    TcVnPoint3_LREAL& aValue,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnPoint3_LREAL [▶ 1640]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnPoint3_LREAL 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.11 AddToContainerElements (TcVnPoint3_REAL)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT          hrPrev,
    TcVnPoint3_REAL& aValue,
    ITcVnContainer*  ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnPoint3_REAL [▶ 1640]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnPoint3_REAL 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.12 AddToContainerElements (TcVnRectangle_DINT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT             hrPrev,
    TcVnRectangle_DINT& stValue,
    ITcVnContainer*     ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stValue	TcVnRectangle_DINT [▶ 1717]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnRectangle_DINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.13 AddToContainerElements (TcVnVector2_DINT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT           hrPrev,
    TcVnVector2_DINT& aValue,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector2_DINT [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_DINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.14 AddToContainerElements (TcVnVector2_INT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT          hrPrev,
    TcVnVector2_INT& aValue,
    ITcVnContainer*  ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector2_INT [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_INT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.15 AddToContainerElements (TcVnVector2_REAL)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT           hrPrev,
    TcVnVector2_REAL& aValue,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector2_REAL [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_REAL 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.16 AddToContainerElements (TcVnVector2_SINT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT           hrPrev,
    TcVnVector2_SINT& aValue,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector2_SINT [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_SINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.17 AddToContainerElements (TcVnVector2_UINT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT           hrPrev,
    TcVnVector2_UINT& aValue,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector2_UINT [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_UINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.18 AddToContainerElements (TcVnVector2_USINT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT           hrPrev,
    TcVnVector2_USINT& aValue,
    ITcVnContainer*    ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector2_USINT [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_USINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.19 AddToContainerElements (TcVnVector3_INT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT          hrPrev,
    TcVnVector3_INT& aValue,
    ITcVnContainer*  ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector3_INT [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector3_INT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.20 AddToContainerElements (TcVnVector3_REAL)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT           hrPrev,
    TcVnVector3_REAL& aValue,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector3_REAL [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector3_REAL 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.21 AddToContainerElements (TcVnVector3_SINT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT           hrPrev,
    TcVnVector3_SINT& aValue,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector3_SINT [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector3_SINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.22 AddToContainerElements (TcVnVector3_UINT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT           hrPrev,
    TcVnVector3_UINT& aValue,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector3_UINT [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector3_UINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.23 AddToContainerElements (TcVnVector3_USINT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT            hrPrev,
    TcVnVector3_USINT& aValue,
    ITcVnContainer*    ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector3_USINT [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector3_USINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.24 AddToContainerElements (TcVnVector4_DINT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT            hrPrev,
    TcVnVector4_DINT& aValue,
    ITcVnContainer*    ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector4_DINT [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_DINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.25 AddToContainerElements (TcVnVector4_INT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT          hrPrev,
    TcVnVector4_INT& aValue,
    ITcVnContainer*  ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector4_INT [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_INT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.26 AddToContainerElements (TcVnVector4_LREAL)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT            hrPrev,
    TcVnVector4_LREAL& aValue,
    ITcVnContainer*    ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector4_LREAL [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_LREAL 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.27 AddToContainerElements (TcVnVector4_SINT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT           hrPrev,
    TcVnVector4_SINT& aValue,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector4_SINT [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_SINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.28 AddToContainerElements (TcVnVector4_UINT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT           hrPrev,
    TcVnVector4_UINT& aValue,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector4_UINT [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_UINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.29 AddToContainerElements (TcVnVector4_USINT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT            hrPrev,
    TcVnVector4_USINT& aValue,
    ITcVnContainer*    ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector4_USINT [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_USINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.30 AddToContainerElements (ULONG)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT         hrPrev,
    ULONG           nValue,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nValue	ULONG	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 UDINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.31 AddToContainerElements (USHORT)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT          hrPrev,
    USHORT           nValue,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nValue	USHORT	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 UINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.32 AddToContainerElements (ULONGLONG)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT         hrPrev,
    ULONGLONG       nValue,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nValue	ULONGLONG	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 ULINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.33 AddToContainerElements (unsigned char)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT         hrPrev,
    unsigned char   nValue,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nValue	unsigned char	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 USINT 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.34 AddToContainerElements (TcVnVector4_REAL)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT           hrPrev,
    TcVnVector4_REAL& aValue,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector4_REAL [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_REAL 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.35 AddToContainerElements (TcVnRotatedRectangle)

Add a value to each container element.

Syntax

Definition:

```
HRESULT AddToContainerElements(
    HRESULT               hrPrev,
    TcVnRotatedRectangle& stValue,
    ITcVnContainer*       ipContainer
)
```

Parameters

Name	Type	Description
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)
stValue	TcVnRotatedRectangle [▶ 1718]&	Value
ipContainer	ITcVnContainer* [▶ 1827]	Container with TcVnRotatedRectangle elements

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.1.36 AddToContainerElements (TcVnRotatedRectangle)

将每个容器元素与一个值相加。

语法

定义：

```
HRESULT AddToContainerElements(
    HRESULT               hrPrev,
    TcVnRotatedRectangle& stValue,
    ITcVnContainer*       ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stValue	TcVnRotatedRectangle [▶ 1718]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnRotatedRectangle 元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.2 AddContainers

两个容器（长度和类型相同）逐元素相加。

语法

定义：

```
HRESULT AddContainers(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer1,
    ITcVnContainer*  ipSrcContainer2,
    ITcVnContainer*& ipDestContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer1	ITcVnContainer* [▶ 1827]	源容器 1
ipSrcContainer2	ITcVnContainer* [▶ 1827]	源容器 2
ipDestContainer	ITcVnContainer* [▶ 1827]&	返回生成的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.3 BitwiseAndContainers

对两个容器（长度和类型相同，仅限整数）逐元素应用逐位 AND 算子。

语法

定义：

```
HRESULT BitwiseAndContainers(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer1,
    ITcVnContainer*  ipSrcContainer2,
    ITcVnContainer*& ipDestContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer1	ITcVnContainer* [▶ 1827]	源容器 1
ipSrcContainer2	ITcVnContainer* [▶ 1827]	源容器 2
ipDestContainer	ITcVnContainer* [▶ 1827]&	返回生成的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.4 BitwiseNotContainer

对一个容器（仅限整数）逐元素应用逐位 NOT 算子。

语法

定义：

```
HRESULT BitwiseNotContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer,
    ITcVnContainer*& ipDestContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	源容器
ipDestContainer	ITcVnContainer* [▶ 1827]&	返回生成的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.5 BitwiseOrContainers

对两个容器（长度和类型相同，仅限整数）逐元素应用逐位 OR 算子。

语法

定义：

```
HRESULT BitwiseOrContainers(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer1,
    ITcVnContainer*  ipSrcContainer2,
    ITcVnContainer*& ipDestContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer1	ITcVnContainer* [▶ 1827]	源容器 1
ipSrcContainer2	ITcVnContainer* [▶ 1827]	源容器 2
ipDestContainer	ITcVnContainer* [▶ 1827]&	返回生成的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.6 BitwiseXorContainers

对两个容器（长度和类型相同，仅限整数）逐元素应用逐位 XOR 算子。

语法

定义：

```
HRESULT BitwiseXorContainers(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer1,
    ITcVnContainer*  ipSrcContainer2,
    ITcVnContainer*& ipDestContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer1	ITcVnContainer* [▶ 1827]	源容器 1
ipSrcContainer2	ITcVnContainer* [▶ 1827]	源容器 2
ipDestContainer	ITcVnContainer* [▶ 1827]&	返回生成的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.7 DivideContainers

两个容器逐元素相除（长度和类型相同）。在整数除以 0 的情况下，结果元素将被设置为 nIntDivideByZeroResult，并返回 S_DIVISION_BY_ZERO，而不是 S_OK。

语法

定义：

```
HRESULT DivideContainers(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer1,
    ITcVnContainer*  ipSrcContainer2,
    ITcVnContainer*& ipDestContainer,
    LONGLONG         nIntDivideByZeroResult = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer1	ITcVnContainer* [▶ 1827]		源容器 1
ipSrcContainer2	ITcVnContainer* [▶ 1827]		源容器 2
ipDestContainer	ITcVnContainer* [▶ 1827]&		返回生成的容器
nIntDivideByZeroResult	LONGLONG	0	整数除以零时设为结果的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.8 MaxContainer

两个容器（长度和类型相同）的逐元素最大值。

语法

定义：

```
HRESULT MaxContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer1,
    ITcVnContainer*  ipSrcContainer2,
    ITcVnContainer*& ipDestContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer1	ITcVnContainer* [▶ 1827]	源容器 1
ipSrcContainer2	ITcVnContainer* [▶ 1827]	源容器 2
ipDestContainer	ITcVnContainer* [▶ 1827]&	返回生成的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.9 MinContainer

两个容器（长度和类型相同）的逐元素最小值。

语法

定义：

```
HRESULT MinContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer1,
    ITcVnContainer*  ipSrcContainer2,
    ITcVnContainer*& ipDestContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer1	ITcVnContainer* [▶ 1827]	源容器 1
ipSrcContainer2	ITcVnContainer* [▶ 1827]	源容器 2
ipDestContainer	ITcVnContainer* [▶ 1827]&	返回生成的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.10 MultiplyContainers

两个容器（长度和类型相同）逐元素相乘。

语法

定义：

```
HRESULT MultiplyContainers(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer1,
```

```
    ITcVnContainer*  ipSrcContainer2,
    ITcVnContainer*& ipDestContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer1	ITcVnContainer* [▶ 1827]	源容器 1
ipSrcContainer2	ITcVnContainer* [▶ 1827]	源容器 2
ipDestContainer	ITcVnContainer* [▶ 1827]&	返回生成的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.11 MultiplyWithContainerElements (1)

将每个容器元素乘以一个值。

语法

定义：

```
HRESULT MultiplyWithContainerElements(
    HRESULT         hrPrev,
    double          fValue,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fValue	double	值
ipContainer	ITcVnContainer* [▶ 1827]	包含一维元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.12 MultiplyWithContainerElements (2)

将每个容器元素乘以一个值。

语法

定义：

```
HRESULT MultiplyWithContainerElements(
    HRESULT            hrPrev,
    TcVnVector2_LREAL& aValue,
    ITcVnContainer*    ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector2_LREAL [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含二维元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.13 MultiplyWithContainerElements (3)

将每个容器元素乘以一个值。

语法

定义：

```
HRESULT MultiplyWithContainerElements(
    HRESULT            hrPrev,
    TcVnVector3_LREAL& aValue,
    ITcVnContainer*    ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector3_LREAL [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	带有三维元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.14 MultiplyWithContainerElements (4)

将每个容器元素乘以一个值。

语法

定义：

```
HRESULT MultiplyWithContainerElements(
    HRESULT           hrPrev,
    TcVnVector4_LREAL& aValue,
    ITcVnContainer*    ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aValue	TcVnVector4_LREAL [▶ 1641]&	值
ipContainer	ITcVnContainer* [▶ 1827]	包含四维元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.15 NegateContainer

对一个容器逐元素取反（二进制）。

语法

定义：

```
HRESULT NegateContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer,
    ITcVnContainer*& ipDestContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	源容器
ipDestContainer	ITcVnContainer* [▶ 1827]&	返回生成的容器（与 ipSrcContainer 类型相同）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.3.16 SubtractContainers

两个容器（长度和类型相同）逐元素相减。

语法

定义：

```
HRESULT SubtractContainers(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer1,
    ITcVnContainer*  ipSrcContainer2,
    ITcVnContainer*& ipDestContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer1	ITcVnContainer* [▶ 1827]	源容器 1
ipSrcContainer2	ITcVnContainer* [▶ 1827]	源容器 2
ipDestContainer	ITcVnContainer* [▶ 1827]&	返回生成的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4 代数图像运算

该组包含对图像 [▶ 143]逐像素应用代数运算的函数。

加法

- AddImages [▶ 1949]
- AddImagesWeighted [▶ 1950]
- AddScalarToImage [▶ 1951]
- AddVectorToImage [▶ 1952]

减法

- SubtractImages [▶ 1980]
- SubtractImageFromScalar [▶ 1978]
- SubtractImageFromVector [▶ 1979]
- SubtractScalarFromImage [▶ 1980]
- SubtractVectorFromImage [▶ 1981]

乘法

- MultiplyImages [▶ 1976]
- MultiplyImageWithScalar [▶ 1977]
- MultiplyImageWithVector [▶ 1977]

除法

- DivideImages [▶ 1964]
- DivideImageByScalar [▶ 1963]
- DivideImageByVector [▶ 1964]
- DivideScalarByImage [▶ 1965]
- DivideVectorByImage [▶ 1966]

乘方和对数化

- ElementwiseExp [▶ 1966]
- ElementwiseLog [▶ 1967]

逐位 AND

- BitwiseAndImages(Exp) [▶ 1952]
- BitwiseAndScalarWithImage [▶ 1954]
- BitwiseAndVectorWithImage [▶ 1954]

逐位 NOT

- BitwiseNotImage(Exp) [▶ 1955]

逐位 OR

- BitwiseOrImages(Exp) [▶ 1956]
- BitwiseOrScalarWithImage [▶ 1958]
- BitwiseOrVectorWithImage [▶ 1958]

逐位 XOR

- BitwiseXorImages(Exp) [▶ 1959]
- BitwiseXorScalarWithImage [▶ 1961]
- BitwiseXorVectorWithImage [▶ 1961]

交叉混合

- BlendImages [▶ 1962]

最大

- MaxImage(Exp) [▶ 1968]
- MaxImageWithScalar(Exp) [▶ 1969]
- MaxImageWithVector(Exp) [▶ 1970]

最小

- MinImage(Exp) [▶ 1972]
- MinImageWithScalar(Exp) [▶ 1973]
- MinImageWithVector(Exp) [▶ 1975]

6.2.3.4.1 AddImages

利用饱和运算将两个图像逐元素相加。

语法

定义：

```
HRESULT AddImages(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	第一个源图像
ipSrcImage2	ITcVnImage* [▶ 1863]	第二个源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

样本

- 复制图像区域 [▶ 2726]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.2 AddImagesWeighted

利用饱和运算（如8位无符号整数像素值运算，结果超出0-255范围时，通过饱和运算确保值在有效范围内）对两个图像进行加权逐元素相加。

语法

定义：

```
HRESULT AddImagesWeighted(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage1,
    ITcVnImage*      ipSrcImage2,
    ITcVnImage*&     ipDestImage,
    double           fWeight1,
    double           fWeight2,
    double           fDelta,
    ETcVnElementType eDestType = TCVN_ET_SAME_AS_SOURCE
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]		第一个源图像
ipSrcImage2	ITcVnImage* [▶ 1863]		第二个源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的目标图像。）

名称	类型	默认值	描述
fWeight1	double		应用于 ipSrcImage1 的权重系数
fWeight2	double		应用于 ipSrcImage2 的权重系数
fDelta	double		与两个图像的加权总和相加的值
eDestType	ETcVnElementType [▶ 1674]	TCVN_ET_SAME_AS_SOURCE	目标图像深度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.3 AddScalarToImage

利用饱和运算为每个图像像素添加一个标量值。

语法

定义：

```
HRESULT AddScalarToImage(
    HRESULT      hrPrev,
    double       fScalar,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fScalar	double	标量值
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.4 AddVectorToImage

利用饱和运算为每个图像像素添加一个向量（每个图像通道 1 个元素）。

语法

定义：

```
HRESULT AddVectorToImage(
    HRESULT            hrPrev,
    TcVnVector4_LREAL& aVector,
    ITcVnImage*        ipSrcImage,
    ITcVnImage*&       ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aVector	TcVnVector4_LREAL [▶ 1641]&	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.5 BitwiseAndImages

对两个图像逐元素应用逐位 AND 算子。

语法

定义：

```
HRESULT BitwiseAndImages(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	第一个源图像
ipSrcImage2	ITcVnImage* [▶ 1863]	第二个源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.6 BitwiseAndImages (Exp)

对两个图像逐元素应用逐位 AND 算子。

语法

定义：

```
HRESULT BitwiseAndImages(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*& ipDestImage,
    ITcVnImage*  ipMask
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	第一个源图像
ipSrcImage2	ITcVnImage* [▶ 1863]	第二个源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage* [▶ 1863]	掩膜用于将运算限制在特定像素位置（掩膜值 0：跳过像素，掩码值 > 0：对像素执行运算）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.7 BitwiseAndScalarWithImage

将标量值和每个图像像素进行逐位 AND 运算。

语法

定义：

```
HRESULT BitwiseAndScalarWithImage(
    HRESULT      hrPrev,
    double       fScalar,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fScalar	double	标量值
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.8 BitwiseAndVectorWithImage

将向量和每个图像元素进行逐位 AND 运算（每个图像通道 1 个元素）。

语法

定义：

```
HRESULT BitwiseAndVectorWithImage(
    HRESULT            hrPrev,
    TcVnVector4_LREAL& aVector,
```

```
    ITcVnImage*         ipSrcImage,
    ITcVnImage*&        ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aVector	TcVnVector4_LREAL [▶ 1641]&	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.9 BitwiseNotImage

对图像逐元素应用逐位 NOT 算子。

语法

定义：

```
HRESULT BitwiseNotImage(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.10 BitwiseNotImage (Exp)

对图像逐元素应用逐位 NOT 算子。

语法

定义：

```
HRESULT BitwiseNotImage(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    ITcVnImage*  ipMask
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage* [▶ 1863]	掩膜用于将运算限制在特定像素位置（掩膜值 0：跳过像素，掩码值 > 0：对像素执行运算）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.11 BitwiseOrImages

对两个图像逐元素应用逐位 OR 算子。

语法

定义：

```
HRESULT BitwiseOrImages(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	第一个源图像
ipSrcImage2	ITcVnImage* [▶ 1863]	第二个源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.12 BitwiseOrImages (Exp)

对两个图像逐元素应用逐位 OR 算子。

语法

定义：

```
HRESULT BitwiseOrImages(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*& ipDestImage,
    ITcVnImage*  ipMask
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	第一个源图像
ipSrcImage2	ITcVnImage* [▶ 1863]	第二个源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage* [▶ 1863]	掩膜用于将运算限制在特定像素位置（掩膜值 0：跳过像素，掩码值 > 0：对像素执行运算）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.13 BitwiseOrScalarWithImage

将标量值和每个图像像素进行逐位 OR 运算。

语法

定义：

```
HRESULT BitwiseOrScalarWithImage(
    HRESULT      hrPrev,
    double       fScalar,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fScalar	double	标量值
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.14 BitwiseOrVectorWithImage

将向量和每个图像元素进行逐位 OR 运算（每个图像通道 1 个元素）。

语法

定义：

```
HRESULT BitwiseOrVectorWithImage(
    HRESULT             hrPrev,
    TcVnVector4_LREAL& aVector,
    ITcVnImage*         ipSrcImage,
    ITcVnImage*&        ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aVector	TcVnVector4_LREAL [▶ 1641]&	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.15 BitwiseXorImages

对两个图像逐元素应用逐位 XOR 算子。

语法

定义：

```
HRESULT BitwiseXorImages(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	第一个源图像
ipSrcImage2	ITcVnImage* [▶ 1863]	第二个源图像

名称	类型	描述
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.16 BitwiseXorImages (Exp)

对两个图像逐元素应用逐位 XOR 算子。

语法

定义：

```
HRESULT BitwiseXorImages(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*& ipDestImage,
    ITcVnImage*  ipMask
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	第一个源图像
ipSrcImage2	ITcVnImage* [▶ 1863]	第二个源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage* [▶ 1863]	掩膜用于将运算限制在特定像素位置（掩膜值 0：跳过像素，掩码值 > 0：对像素执行运算）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.17 BitwiseXorScalarWithImage

将标量值和每个图像像素进行逐位 XOR 运算。

语法

定义：

```
HRESULT BitwiseXorScalarWithImage(
    HRESULT      hrPrev,
    double       fScalar,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fScalar	double	标量值
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.18 BitwiseXorVectorWithImage

将向量和每个图像元素进行逐位 XOR 运算（每个图像通道 1 个元素）。

语法

定义：

```
HRESULT BitwiseXorVectorWithImage(
    HRESULT            hrPrev,
    TcVnVector4_LREAL& aVector,
    ITcVnImage*        ipSrcImage,
    ITcVnImage*&       ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aVector	TcVnVector4_LREAL [▶ 1641]&	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.19 BlendImages

混合两个图像，即 dest =（src1 * weight1 + src2 * weight2）/（weight1 + weight2）。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT BlendImages(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*  ipWeights1,
    ITcVnImage*  ipWeights2,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	第一个源图像（ET_USINT 或 ET_REAL）
ipSrcImage2	ITcVnImage* [▶ 1863]	第二个源图像（类型和大小与 ipSrcImage1 相同）
ipWeights1	ITcVnImage* [▶ 1863]	ipSrcImage1 的权重（1 通道，ET_REAL）
ipWeights2	ITcVnImage* [▶ 1863]	ipSrcImage2 的权重（1 通道，ET_REAL）

名称	类型	描述
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.20 DivideImageByScalar

利用饱和运算将每个图像像素除以一个标量值。

语法

定义：

```
HRESULT DivideImageByScalar(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    double       fScalar,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
fScalar	double	标量值
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.21 DivideImageByVector

利用饱和运算将每个图像像素除以一个向量（每个图像通道 1 个元素）。

语法

定义：

```
HRESULT DivideImageByVector(
    HRESULT            hrPrev,
    ITcVnImage*        ipSrcImage,
    TcVnVector4_LREAL& aVector,
    ITcVnImage*&       ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
aVector	TcVnVector4_LREAL [▶ 1641]&	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.22 DivideImages

利用饱和运算对两个图像进行逐元素相除。（除以零等于零）。

语法

定义：

```
HRESULT DivideImages(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	第一张源图像（被除数）
ipSrcImage2	ITcVnImage* [▶ 1863]	第二个源图像（除数）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.23 DivideScalarByImage

利用饱和运算将标量值除以每个图像像素。

语法

定义：

```
HRESULT DivideScalarByImage(
    HRESULT      hrPrev,
    double       fScalar,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fScalar	double	标量值
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.24 DivideVectorByImage

利用饱和运算将一个向量除以每个图像像素（每个图像通道 1 个元素）。

语法

定义：

```
HRESULT DivideVectorByImage(
    HRESULT            hrPrev,
    TcVnVector4_LREAL& aVector,
    ITcVnImage*        ipSrcImage,
    ITcVnImage*&       ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aVector	TcVnVector4_LREAL [▶ 1641]&	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.25 ElementwiseExp

计算每个像素值的自然指数。

语法

定义：

```
HRESULT ElementwiseExp(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（REAL 或 LREAL）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（类型和大小与 ipSrcImage 相同，如果需要，将创建适当的图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.26 ElementwiseLog

计算每个像素值的自然对数。

语法

定义：

```
HRESULT ElementwiseLog(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（REAL 或 LREAL）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（类型和大小与 ipSrcImage 相同，如果需要，将创建适当的图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.27 MaxImage

两个图像的逐元素最大值。

语法

定义：

```
HRESULT MaxImage(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	第一个源图像
ipSrcImage2	ITcVnImage* [▶ 1863]	第二个源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.28 MaxImage (Exp)

两个图像的逐元素最大值。

语法

定义：

```
HRESULT MaxImage(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*& ipDestImage,
    ITcVnImage*  ipMask
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	第一个源图像
ipSrcImage2	ITcVnImage* [▶ 1863]	第二个源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage* [▶ 1863]	USINT 类型的掩膜

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.29 MaxImageWithScalar

图像和标量值进行逐元素比较的最大值。

语法

定义：

```
HRESULT MaxImageWithScalar(
    HRESULT      hrPrev,
    double       fScalar,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fScalar	double	标量值
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.30 MaxImageWithScalar (Exp)

图像和标量值进行逐元素比较的最大值。

语法

定义：

```
HRESULT MaxImageWithScalar(
    HRESULT      hrPrev,
    double       fScalar,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    ITcVnImage*  ipMask
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fScalar	double	标量值
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage* [▶ 1863]	USINT 类型的掩膜

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.31 MaxImageWithVector

图像和向量进行逐元素比较的最大值（每个图像通道 1 个元素）。

语法

定义：

```
HRESULT MaxImageWithVector(
    HRESULT             hrPrev,
    TcVnVector4_LREAL&  aVector,
    ITcVnImage*         ipSrcImage,
    ITcVnImage*&        ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aVector	TcVnVector4_LREAL [▶ 1641]&	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.32 MaxImageWithVector (Exp)

图像和向量进行逐元素比较的最大值（每个图像通道 1 个元素）。

语法

定义：

```
HRESULT MaxImageWithVector(
    HRESULT             hrPrev,
    TcVnVector4_LREAL&  aVector,
    ITcVnImage*         ipSrcImage,
    ITcVnImage*&        ipDestImage,
    ITcVnImage*         ipMask
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aVector	TcVnVector4_LREAL [▶ 1641]&	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。
ipSrcImage	ITcVnImage* [▶ 1863]	源图像

名称	类型	描述
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage* [▶ 1863]	USINT 类型的掩膜

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.33 MinImage

两个图像的逐元素最小值。

语法

定义：

```
HRESULT MinImage(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	第一个源图像
ipSrcImage2	ITcVnImage* [▶ 1863]	第二个源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.34 MinImage (Exp)

两个图像的逐元素最小值。

语法

定义：

```
HRESULT MinImage(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*& ipDestImage,
    ITcVnImage*  ipMask
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	第一个源图像
ipSrcImage2	ITcVnImage* [▶ 1863]	第二个源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage* [▶ 1863]	USINT 类型的掩膜

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.35 MinImageWithScalar

图像和标量值进行逐元素比较的最小值。

语法

定义：

```
HRESULT MinImageWithScalar(
    HRESULT      hrPrev,
    double       fScalar,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fScalar	double	标量值
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.36 MinImageWithScalar (Exp)

图像和标量值进行逐元素比较的最小值。

语法

定义：

```
HRESULT MinImageWithScalar(
    HRESULT      hrPrev,
    double       fScalar,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    ITcVnImage*  ipMask
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fScalar	double	标量值
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage* [▶ 1863]	USINT 类型的掩膜

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.37 MinImageWithVector

图像和向量进行逐元素比较的最小值（每个图像通道 1 个元素）。

语法

定义：

```
HRESULT MinImageWithVector(
    HRESULT             hrPrev,
    TcVnVector4_LREAL& aVector,
    ITcVnImage*         ipSrcImage,
    ITcVnImage*&        ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aVector	TcVnVector4_LREAL [▶ 1641]&	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.38 MinImageWithVector (Exp)

图像和向量进行逐元素比较的最小值（每个图像通道 1 个元素）。

语法

定义：

```
HRESULT MinImageWithVector(
    HRESULT          hrPrev,
    TcVnVector4_LREAL& aVector,
    ITcVnImage*      ipSrcImage,
    ITcVnImage*&     ipDestImage,
    ITcVnImage*      ipMask
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aVector	TcVnVector4_LREAL [▶ 1641]&	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
ipMask	ITcVnImage* [▶ 1863]	USINT 类型的掩膜

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.39 MultiplyImages

利用饱和运算对两个图像进行逐元素相乘。

语法

定义：

```
HRESULT MultiplyImages(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	第一个源图像
ipSrcImage2	ITcVnImage* [▶ 1863]	第二个源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.40 MultiplyImageWithScalar

利用饱和运算将每个图像像素乘以一个标量值。

语法

定义：

```
HRESULT MultiplyImageWithScalar(
    HRESULT      hrPrev,
    double       fScalar,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fScalar	double	标量值
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.41 MultiplyImageWithVector

使用饱和运算将每个图像像素乘以一个向量（每个图像通道 1 个元素）。

语法

定义：

```
HRESULT MultiplyImageWithVector(
    HRESULT             hrPrev,
    TcVnVector4_LREAL& aVector,
    ITcVnImage*         ipSrcImage,
    ITcVnImage*&        ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aVector	TcVnVector4_LREAL [▶ 1641]&	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.42 SubtractImageFromScalar

利用饱和运算从标量值中减去每个图像像素。

语法

定义：

```
HRESULT SubtractImageFromScalar(
    HRESULT      hrPrev,
    double       fScalar,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fScalar	double	标量值
ipSrcImage	ITcVnImage* [▶ 1863]	源图像

名称	类型	描述
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.43 SubtractImageFromVector

利用饱和运算从向量（每个图像通道 1 个元素）中减去每个图像像素。

语法

定义：

```
HRESULT SubtractImageFromVector(
    HRESULT            hrPrev,
    TcVnVector4_LREAL& aVector,
    ITcVnImage*        ipSrcImage,
    ITcVnImage*&       ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aVector	TcVnVector4_LREAL [▶ 1641]&	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.44 SubtractImages

利用饱和运算对两个图像进行逐元素相减。

语法

定义：

```
HRESULT SubtractImages(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	第一个源图像（被减数）
ipSrcImage2	ITcVnImage* [▶ 1863]	第二个源图像（减数）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.45 SubtractScalarFromImage

利用饱和运算从每个图像像素中减去标量值。

语法

定义：

```
HRESULT SubtractScalarFromImage(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    double       fScalar,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像

名称	类型	描述
fScalar	double	标量值
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.4.46 SubtractVectorFromImage

利用饱和运算从每个图像像素中减去向量（每个图像通道 1 个元素）。

语法

定义：

```
HRESULT SubtractVectorFromImage(
    HRESULT             hrPrev,
    ITcVnImage*         ipSrcImage,
    TcVnVector4_LREAL& aVector,
    ITcVnImage*&        ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
aVector	TcVnVector4_LREAL [▶ 1641]&	4 元素向量（每个图像通道 1 个元素）。如果图像的通道较少，后面的向量元素将被忽略。
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5 基本容器运算

该组包含处理容器 [▶ 146]的函数。

函数

创建容器

- CopyContainer [▶ 2139]
- CopyContainerElementsConditional_ITcVnContainer [▶ 2140]
- CopyContainerElementsConditional_ITcVnForwardIterator [▶ 2140]
- CreateContainer [▶ 2141]
- CreateContainerFromArray [▶ 2142]
- ExtractContainerRange [▶ 2147]
- ReserveContainerMemory [▶ 2153]

容器元素的排序

- SortContainer [▶ 2155]

容器元素的处理

- AppendToContainer [▶ 1983]
- CheckIfEmpty [▶ 2137]
- ConvertContainerType [▶ 2138]
- EraseFromContainer [▶ 2142]
- FillContainer(Exp) [▶ 2033]
- GetAt [▶ 2059]
- GetContainer(Exp) [▶ 2148]
- GetNumberOfElements [▶ 2150]
- InsertIntoContainer [▶ 2084]
- ReverseContainer [▶ 2153]
- SetAt [▶ 2110]
- SetContainer [▶ 2154]

将容器数据导出到内存区域

- ExportContainer [▶ 2143]
- ExportSubContainer [▶ 2145]
- ExportContainer_String [▶ 2144]
- ExportSubContainer_String [▶ 2146]
- ExportContainerSize [▶ 2145]
- ExportSubContainerSize [▶ 2147]

迭代器的处理

- AdvanceIterator [▶ 2136]
- CheckIfIteratorIsAtEnd [▶ 2138]

- GetForwardIterator [▶ 2149]
- GetRandomAccessIterator [▶ 2151]
- IncrementIterator [▶ 2151]
- IteratorDistance [▶ 2152]
- SetIteratorToBegin [▶ 2155]

6.2.3.5.1 AppendToContainer

本章包含根据数据类型向容器中添加元素的功能。

6.2.3.5.1.1 AppendToContainer (LONG)

将 DINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT         hrPrev,
    LONG            nElement,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	LONG	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.2 AppendToContainer (SHORT)

将 INT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT         hrPrev,
    SHORT           nElement,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	SHORT	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.3 AppendToContainer (ITcVnContainer*)

将单个元素添加到一个容器或多个连接容器（如果 ipElement 的 typeId 与 ipContainer 相同）。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT         hrPrev,
    ITcVnContainer* ipElement,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipElement	ITcVnContainer* [▶ 1827]	单个元素，添加到 ipContainer，或添加到包含多个元素的要连接的容器
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.4 AppendToContainer (ITcVnForwardIterator*)

将单个元素（由迭代器表示）添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT               hrPrev,
    ITcVnForwardIterator* ipElement,
    ITcVnContainer*       ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipElement	ITcVnForwardIterator* [▶ 1819]	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.5 AppendToContainer (ITcVnImage*)

将 ITcVnImage 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT         hrPrev,
    ITcVnImage*     ipElement,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipElement	ITcVnImage* [▶ 1863]	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.6 AppendToContainer (double)

将 LREAL 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT          hrPrev,
    double           fElement,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fElement	double	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.7 AppendToContainer (float)

将 REAL 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT          hrPrev,
    float            fElement,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fElement	float	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.8 AppendToContainer (char)

将 SINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT          hrPrev,
    char             nElement,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	char	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.9 AppendToContainer (TcVnDMatch)

将 TcVnDMatch 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT         hrPrev,
    TcVnDMatch&     stElement,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stElement	TcVnDMatch [▶ 1705]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.10 AppendToContainer (TcVnKeyPoint)

将 TcVnKeyPoint 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT         hrPrev,
    TcVnKeyPoint&   stElement,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stElement	TcVnKeyPoint [▶ 1706]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.11 AppendToContainer (TcVnPoint2_DINT)

将 TcVnPoint2_DINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT          hrPrev,
    TcVnPoint2_DINT& aElement,
    ITcVnContainer*  ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnPoint2_DINT [▶ 1640]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.12 AppendToContainer (TcVnPoint2_LREAL)

将 TcVnPoint2_LREAL 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT           hrPrev,
    TcVnPoint2_LREAL& aElement,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnPoint2_LREAL [▶ 1640]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.13 AppendToContainer (TcVnPoint2_REAL)

将 TcVnPoint2_REAL 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT          hrPrev,
    TcVnPoint2_REAL& aElement,
    ITcVnContainer*  ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnPoint2_REAL [▶ 1640]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.14 AppendToContainer (TcVnPoint3_LREAL)

将 TcVnPoint3_LREAL 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT           hrPrev,
    TcVnPoint3_LREAL& aElement,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnPoint3_LREAL [▶ 1640]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.15 AppendToContainer (TcVnPoint3_REAL)

将 TcVnPoint3_REAL 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT          hrPrev,
    TcVnPoint3_REAL& aElement,
    ITcVnContainer*  ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnPoint3_REAL [▶ 1640]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.16 AppendToContainer (TcVnRectangle_DINT)

将 TcVnRectangle_DINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT             hrPrev,
    TcVnRectangle_DINT& stElement,
    ITcVnContainer*     ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stElement	TcVnRectangle_DINT [▶ 1717]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.17 AppendToContainer (TcVnVector2_DINT)

将 TcVnVector2_DINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT           hrPrev,
    TcVnVector2_DINT& aElement,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_DINT [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.18 AppendToContainer (TcVnVector2_INT)

将 TcVnVector2_INT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT          hrPrev,
    TcVnVector2_INT& aElement,
    ITcVnContainer*  ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_INT [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.19 AppendToContainer (TcVnVector2_REAL)

将 TcVnVector2_REAL 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT           hrPrev,
    TcVnVector2_REAL& aElement,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_REAL [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.20 AppendToContainer (TcVnVector2_SINT)

将 TcVnVector2_SINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT           hrPrev,
    TcVnVector2_SINT& aElement,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_SINT [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.21 AppendToContainer (TcVnVector2_UINT)

将 TcVnVector2_UINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT           hrPrev,
    TcVnVector2_UINT& aElement,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_UINT [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.22 AppendToContainer (TcVnVector2_USINT)

将 TcVnVector2_USINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT            hrPrev,
    TcVnVector2_USINT& aElement,
    ITcVnContainer*    ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_USINT [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.23 AppendToContainer (TcVnVector3_INT)

将 TcVnVector3_INT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT          hrPrev,
    TcVnVector3_INT& aElement,
    ITcVnContainer*  ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector3_INT [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.24 AppendToContainer (TcVnVector3_REAL)

将 TcVnVector3_REAL 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT           hrPrev,
    TcVnVector3_REAL& aElement,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector3_REAL [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.25 AppendToContainer (TcVnVector3_SINT)

将 TcVnVector3_SINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT           hrPrev,
    TcVnVector3_SINT& aElement,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector3_SINT [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.26 AppendToContainer (TcVnVector3_UINT)

将 TcVnVector3_UINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT           hrPrev,
    TcVnVector3_UINT& aElement,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector3_UINT [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.27 AppendToContainer (TcVnVector3_USINT)

将 TcVnVector3_USINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT            hrPrev,
    TcVnVector3_USINT& aElement,
    ITcVnContainer*    ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector3_USINT [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.28 AppendToContainer (TcVnVector4_DINT)

将 TcVnVector4_DINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT           hrPrev,
    TcVnVector4_DINT& aElement,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_DINT [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.29 AppendToContainer (TcVnVector4_INT)

将 TcVnVector4_INT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT          hrPrev,
    TcVnVector4_INT& aElement,
    ITcVnContainer*  ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_INT [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.30 AppendToContainer (TcVnVector4_LREAL)

将 TcVnVector4_LREAL 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT            hrPrev,
    TcVnVector4_LREAL& aElement,
    ITcVnContainer*    ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_LREAL [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.31 AppendToContainer (TcVnVector4_SINT)

将 TcVnVector4_SINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT           hrPrev,
    TcVnVector4_SINT& aElement,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_SINT [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.32 AppendToContainer (TcVnVector4_UINT)

将 TcVnVector4_UINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT           hrPrev,
    TcVnVector4_UINT& aElement,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_UINT [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.33 AppendToContainer (TcVnVector4_USINT)

将 TcVnVector4_USINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT            hrPrev,
    TcVnVector4_USINT& aElement,
    ITcVnContainer*    ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_USINT [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.34 AppendToContainer (ULONG)

将 UDINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT          hrPrev,
    ULONG            nElement,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	ULONG	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.35 AppendToContainer (USHORT)

将 UINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT          hrPrev,
    USHORT           nElement,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	USHORT	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.36 AppendToContainer (ULONGLONG)

将 ULINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT         hrPrev,
    ULONGLONG       nElement,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	ULONGLONG	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.37 AppendToContainer (unsigned char)

将 USINT 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT         hrPrev,
    unsigned char   nElement,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	unsigned char	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.38 AppendToContainer (TcVnVector4_REAL)

将 TcVnVector4_REAL 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT           hrPrev,
    TcVnVector4_REAL& aElement,
    ITcVnContainer*   ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_REAL [▶ 1641]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.1.39 AppendToContainer (TcVnRotatedRectangle)

将 TcVnRotatedRectangle 类型的单个元素添加到容器中。

语法

定义：

```
HRESULT AppendToContainer(
    HRESULT               hrPrev,
    TcVnRotatedRectangle& stElement,
    ITcVnContainer*       ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stElement	TcVnRotatedRectangle [▶ 1718]&	添加到 ipContainer 的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	将要添加元素的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2 FillContainer

本章包含根据数据类型用特殊值填充容器的功能。这个函数适用于包括基本元素的容器。

6.2.3.5.2.1 FillContainer (LONG)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    LONG             nValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nValue	LONG	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.2 FillContainer (SHORT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    SHORT            nValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	描述
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nValue	SHORT	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.3 FillContainer (ITcVnImage*)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    ITcVnImage*     ipValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
ipValue	ITcVnImage* [▶ 1863]	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.4 FillContainer (double)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    double           fValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
fValue	double	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.5 FillContainer (float)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    float            fValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
fValue	float	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.6 FillContainer (char)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    char            nValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nValue	char	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.7 FillContainer (TcVnDMatch)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    TcVnDMatch&     stValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
stValue	TcVnDMatch [▶ 1705]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.8 FillContainer (TcVnKeyPoint)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    TcVnKeyPoint&   stValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
stValue	TcVnKeyPoint [▶ 1706]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.9 FillContainer (TcVnPoint2_DINT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint2_DINT& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnPoint2_DINT [▶ 1640]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.10 FillContainer (TcVnPoint2_LREAL)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnPoint2_LREAL& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnPoint2_LREAL [▶ 1640]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.11 FillContainer (TcVnPoint2_REAL)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint2_REAL& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnPoint2_REAL [▶ 1640]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.12 FillContainer (TcVnPoint3_LREAL)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnPoint3_LREAL& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnPoint3_LREAL [▶ 1640]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.13 FillContainer (TcVnPoint3_REAL)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint3_REAL& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnPoint3_REAL [▶ 1640]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.14 FillContainer (TcVnRectangle_DINT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT             hrPrev,
    ITcVnContainer*     ipContainer,
    TcVnRectangle_DINT& stValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
stValue	TcVnRectangle_DINT [▶ 1717]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.15 FillContainer (TcVnVector2_DINT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_DINT& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector2_DINT [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.16 FillContainer (TcVnVector2_INT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector2_INT& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector2_INT [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.17 FillContainer (TcVnVector2_REAL)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_REAL& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector2_REAL [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.18 FillContainer (TcVnVector2_SINT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_SINT& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector2_SINT [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.19 FillContainer (TcVnVector2_UINT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_UINT& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector2_UINT [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.20 FillContainer (TcVnVector2_USINT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector2_USINT& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector2_USINT [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.21 FillContainer (TcVnVector3_INT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector3_INT& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector3_INT [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.22 FillContainer (TcVnVector3_REAL)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_REAL& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector3_REAL [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.23 FillContainer (TcVnVector3_SINT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_SINT& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector3_SINT [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.24 FillContainer (TcVnVector3_UINT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_UINT& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector3_UINT [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.25 FillContainer (TcVnVector3_USINT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector3_USINT& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector3_USINT [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.26 FillContainer (TcVnVector4_DINT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_DINT& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector4_DINT [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.27 FillContainer (TcVnVector4_INT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector4_INT& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector4_INT [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.28 FillContainer (TcVnVector4_LREAL)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector4_LREAL& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector4_LREAL [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.29 FillContainer (TcVnVector4_SINT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_SINT& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector4_SINT [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.30 FillContainer (TcVnVector4_UINT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_UINT& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector4_UINT [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.31 FillContainer (TcVnVector4_USINT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector4_USINT& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector4_USINT [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.32 FillContainer (ULONG)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    ULONG            nValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nValue	ULONG	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.33 FillContainer (USHORT)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    USHORT          nValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nValue	USHORT	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.34 FillContainer (ULONGLONG)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    ULONGLONG       nValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nValue	ULONGLONG	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.35 FillContainer (unsigned char)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    unsigned char   nValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nValue	unsigned char	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.36 FillContainer (TcVnVector4_REAL)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_REAL& aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector4_REAL [▶ 1641]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.2.37 FillContainer (TcVnRotatedRectangle)

用指定值填充容器。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT               hrPrev,
    ITcVnContainer*       ipContainer,
    TcVnRotatedRectangle& stValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
stValue	TcVnRotatedRectangle [▶ 1718]&	用于设置容器元素的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3 FillContainerExp

本章包含专家函数，根据数据类型，可用于用特定的值填充容器，即使只是部分填充。这个函数适用于包括基本元素的容器。

6.2.3.5.3.1 FillContainer (LONG)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    LONG            nValue,
    ULONGLONG       nStartIdx,
    ULONGLONG       nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nValue	LONG	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.2 FillContainer (SHORT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    SHORT            nValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nValue	SHORT	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.3 FillContainer (ITcVnImage*)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    ITcVnImage*      ipValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
ipValue	ITcVnImage* [▶ 1863]	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.4 FillContainer (double)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    double           fValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
fValue	double	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.5 FillContainer (float)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    float           fValue,
    ULONGLONG       nStartIdx,
    ULONGLONG       nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
fValue	float	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.6 FillContainer (char)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    char             nValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nValue	char	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.7 FillContainer (TcVnDMatch)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    TcVnDMatch&      stValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
stValue	TcVnDMatch [▶ 1705]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引

名称	类型	描述
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.8 FillContainer (TcVnKeyPoint)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    TcVnKeyPoint&    stValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
stValue	TcVnKeyPoint [▶ 1706]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.9 FillContainer (TcVnPoint2_DINT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint2_DINT& aValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnPoint2_DINT [▶ 1640]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.10 FillContainer (TcVnPoint2_LREAL)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint2_LREAL& aValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	描述
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnPoint2_LREAL [▶ 1640]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.11 FillContainer (TcVnPoint2_REAL)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint2_REAL& aValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnPoint2_REAL [▶ 1640]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.12 FillContainer (TcVnPoint3_LREAL)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnPoint3_LREAL& aValue,
    ULONGLONG         nStartIdx,
    ULONGLONG         nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnPoint3_LREAL [▶ 1640]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.13 FillContainer (TcVnPoint3_REAL)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnPoint3_REAL&  aValue,
    ULONGLONG         nStartIdx,
    ULONGLONG         nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnPoint3_REAL [▶ 1640]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.14 FillContainer (TcVnRectangle_DINT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT             hrPrev,
    ITcVnContainer*     ipContainer,
    TcVnRectangle_DINT& stValue,
    ULONGLONG           nStartIdx,
    ULONGLONG           nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
stValue	TcVnRectangle_DINT [▶ 1717]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.15 FillContainer (TcVnVector2_DINT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_DINT& aValue,
    ULONGLONG         nStartIdx,
    ULONGLONG         nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector2_DINT [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.16 FillContainer (TcVnVector2_INT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector2_INT& aValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector2_INT [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.17 FillContainer (TcVnVector2_REAL)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_REAL& aValue,
    ULONGLONG         nStartIdx,
    ULONGLONG         nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector2_REAL [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.18 FillContainer (TcVnVector2_SINT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_SINT& aValue,
    ULONGLONG         nStartIdx,
    ULONGLONG         nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector2_SINT [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.19 FillContainer (TcVnVector2_UINT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_UINT& aValue,
    ULONGLONG         nStartIdx,
    ULONGLONG         nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector2_UINT [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.20 FillContainer (TcVnVector2_USINT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector2_USINT& aValue,
    ULONGLONG          nStartIdx,
    ULONGLONG          nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector2_USINT [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引

名称	类型	描述
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.21 FillContainer (TcVnVector3_INT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector3_INT& aValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector3_INT [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.22 FillContainer (TcVnVector3_REAL)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_REAL& aValue,
    ULONGLONG         nStartIdx,
    ULONGLONG         nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector3_REAL [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.23 FillContainer (TcVnVector3_SINT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_SINT& aValue,
    ULONGLONG         nStartIdx,
    ULONGLONG         nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	描述
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector3_SINT [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.24 FillContainer (TcVnVector3_UINT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_UINT& aValue,
    ULONGLONG         nStartIdx,
    ULONGLONG         nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector3_UINT [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.25 FillContainer (TcVnVector3_USINT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector3_USINT& aValue,
    ULONGLONG          nStartIdx,
    ULONGLONG          nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector3_USINT [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.26 FillContainer (TcVnVector4_DINT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector4_DINT& aValue,
    ULONGLONG          nStartIdx,
    ULONGLONG          nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector4_DINT [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.27 FillContainer (TcVnVector4_INT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector4_INT& aValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector4_INT [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.28 FillContainer (TcVnVector4_LREAL)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector4_LREAL& aValue,
    ULONGLONG          nStartIdx,
    ULONGLONG          nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector4_LREAL [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.29 FillContainer (TcVnVector4_SINT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_SINT& aValue,
    ULONGLONG         nStartIdx,
    ULONGLONG         nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector4_SINT [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.30 FillContainer (TcVnVector4_UINT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_UINT& aValue,
    ULONGLONG         nStartIdx,
    ULONGLONG         nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector4_UINT [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.31 FillContainer (TcVnVector4_USINT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector4_USINT& aValue,
    ULONGLONG          nStartIdx,
    ULONGLONG          nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector4_USINT [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.32 FillContainer (ULONG)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    ULONG            nValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nValue	ULONG	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.33 FillContainer (USHORT)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    USHORT           nValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nValue	USHORT	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引

名称	类型	描述
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.34 FillContainer (ULONGLONG)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    ULONGLONG        nValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nValue	ULONGLONG	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.35 FillContainer (unsigned char)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    unsigned char    nValue,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nValue	unsigned char	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.36 FillContainer (TcVnVector4_REAL)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_REAL& aValue,
    ULONGLONG         nStartIdx,
    ULONGLONG         nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	描述
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
aValue	TcVnVector4_REAL [▶ 1641]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.3.37 FillContainer (TcVnRotatedRectangle)

用指定值对容器进行部分填充。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT FillContainer(
    HRESULT               hrPrev,
    ITcVnContainer*       ipContainer,
    TcVnRotatedRectangle& stValue,
    ULONGLONG             nStartIdx,
    ULONGLONG             nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
stValue	TcVnRotatedRectangle [▶ 1718]&	用于设置容器元素的值
nStartIdx	ULONGLONG	起始索引
nCount	ULONGLONG	要设置的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4 GetAt

如果需要访问一个容器中的单个元素，应该使用F_VN_GetAt函数。如果需要连续处理容器中的所有元素，使用迭代器 [▶ 2149]更好。

6.2.3.5.4.1 GetAt (LONG)

获取容器中指定索引处的 DINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    LONG&            nElement,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 DINT 元素的容器
nElement	LONG&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.2 GetAt (SHORT)

获取容器中指定索引处的 INT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    SHORT&           nElement,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 INT 元素的容器
nElement	SHORT&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.3 GetAt (ITcVnContainer*)

获取源容器中指定索引处的 ITcVnContainer 元素。（或者使用接口方法 .GetAt。）

语法

定义：

```
HRESULT GetAt(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer,
    ITcVnContainer*& ipDestContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	包含 ITcVnContainer 元素的容器
ipDestContainer	ITcVnContainer* [▶ 1827]&	返回指定索引处的容器
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.4 GetAt (ITcVnImage*)

获取容器中指定索引处的 ITcVnImage 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    ITcVnImage*&     ipElement,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 ITcVnImage 元素的容器
ipElement	ITcVnImage* [▶ 1863]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.5 GetAt (double)

获取容器中指定索引处的 LREAL 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
```

```
    double&         fElement,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 LREAL 元素的容器
fElement	double&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.6 GetAt (float)

获取容器中指定索引处的 REAL 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    float&          fElement,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 REAL 元素的容器
fElement	float&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.7 GetAt (char)

获取容器中指定索引处的 SINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    char&            nElement,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	装有 SINT 元素的容器
nElement	char&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.8 GetAt (TcVnDMatch)

获取容器中指定索引处的 TcVnDMatch 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    TcVnDMatch&      stElement,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnDMatch 元素的容器
stElement	TcVnDMatch [▶ 1705]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.9 GetAt (TcVnKeyPoint)

获取容器中指定索引处的 TcVnKeyPoint 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    TcVnKeyPoint&   stElement,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnKeyPoint 元素的容器
stElement	TcVnKeyPoint [▶ 1706]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.10 GetAt (TcVnPoint2_DINT)

获取容器中指定索引处的 TcVnPoint2_DINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint2_DINT& aElement,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnPoint2_DINT 元素的容器
aElement	TcVnPoint2_DINT [▶ 1640]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.11 GetAt (TcVnPoint2_LREAL)

获取容器中指定索引处的 TcVnPoint2_LREAL 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint2_LREAL& aElement,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnPoint2_LREAL 元素的容器
aElement	TcVnPoint2_LREAL [▶ 1640]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.12 GetAt (TcVnPoint2_REAL)

获取容器中指定索引处的 TcVnPoint2_REAL 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint2_REAL& aElement,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnPoint2_REAL 元素的容器
aElement	TcVnPoint2_REAL [▶ 1640]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.13 GetAt (TcVnPoint3_LREAL)

获取容器中指定索引处的 TcVnPoint3_LREAL 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnPoint3_LREAL& aElement,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnPoint3_LREAL 元素的容器
aElement	TcVnPoint3_LREAL [▶ 1640]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.14 GetAt (TcVnPoint3_REAL)

获取容器中指定索引处的 TcVnPoint3_REAL 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint3_REAL& aElement,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnPoint3_REAL 元素的容器
aElement	TcVnPoint3_REAL [▶ 1640]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.15 GetAt (TcVnRectangle_DINT)

获取容器中指定索引处的 TcVnRectangle_DINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT             hrPrev,
    ITcVnContainer*     ipContainer,
    TcVnRectangle_DINT& stElement,
    ULONGLONG           nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnRectangle_DINT 元素的容器
stElement	TcVnRectangle_DINT [▶ 1717]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.16 GetAt (TcVnVector2_DINT)

获取容器中指定索引处的 TcVnVector2_DINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_DINT& aElement,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_DINT 元素的容器
aElement	TcVnVector2_DINT [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.17 GetAt (TcVnVector2_INT)

获取容器中指定索引处的 TcVnVector2_INT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_INT& aElement,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_INT 元素的容器
aElement	TcVnVector2_INT [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.18 GetAt (TcVnVector2_REAL)

获取容器中指定索引处的 TcVnVector2_REAL 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_REAL& aElement,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_REAL 元素的容器
aElement	TcVnVector2_REAL [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.19 GetAt (TcVnVector2_SINT)

获取容器中指定索引处的 TcVnVector2_SINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_SINT& aElement,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_SINT 元素的容器
aElement	TcVnVector2_SINT [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.20 GetAt (TcVnVector2_UINT)

获取容器中指定索引处的 TcVnVector2_UINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_UINT& aElement,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_UINT 元素的容器
aElement	TcVnVector2_UINT [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.21 GetAt (TcVnVector2_USINT)

获取容器中指定索引处的 TcVnVector2_USINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector2_USINT& aElement,
    ULONGLONG          nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_USINT 元素的容器
aElement	TcVnVector2_USINT [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.22 GetAt (TcVnVector3_INT)

获取容器中指定索引处的 TcVnVector3_INT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector3_INT& aElement,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector3_INT 元素的容器
aElement	TcVnVector3_INT [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.23 GetAt (TcVnVector3_REAL)

获取容器中指定索引处的 TcVnVector3_REAL 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_REAL& aElement,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector3_REAL 元素的容器
aElement	TcVnVector3_REAL [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.24 GetAt (TcVnVector3_SINT)

获取容器中指定索引处的 TcVnVector3_SINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_SINT& aElement,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector3_SINT 元素的容器
aElement	TcVnVector3_SINT [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.25 GetAt (TcVnVector3_UINT)

获取容器中指定索引处的 TcVnVector3_UINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_UINT& aElement,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector3_UINT 元素的容器
aElement	TcVnVector3_UINT [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.26 GetAt (TcVnVector3_USINT)

获取容器中指定索引处的 TcVnVector3_USINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_USINT& aElement,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector3_USINT 元素的容器
aElement	TcVnVector3_USINT [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.27 GetAt (TcVnVector4_DINT)

获取容器中指定索引处的 TcVnVector4_DINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_DINT& aElement,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_DINT 元素的容器
aElement	TcVnVector4_DINT [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.28 GetAt (TcVnVector4_INT)

获取容器中指定索引处的 TcVnVector4_INT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector4_INT& aElement,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_INT 元素的容器
aElement	TcVnVector4_INT [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.29 GetAt (TcVnVector4_LREAL)

获取容器中指定索引处的 TcVnVector4_LREAL 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector4_LREAL& aElement,
    ULONGLONG          nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_LREAL 元素的容器
aElement	TcVnVector4_LREAL [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.30 GetAt (TcVnVector4_SINT)

获取容器中指定索引处的 TcVnVector4_SINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_SINT& aElement,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_SINT 元素的容器
aElement	TcVnVector4_SINT [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.31 GetAt (TcVnVector4_UINT)

获取容器中指定索引处的 TcVnVector4_UINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_UINT& aElement,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_UINT 元素的容器
aElement	TcVnVector4_UINT [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.32 GetAt (TcVnVector4_USINT)

获取容器中指定索引处的 TcVnVector4_USINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_USINT& aElement,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_USINT 元素的容器
aElement	TcVnVector4_USINT [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.33 GetAt (ULONG)

获取容器中指定索引处的 UDINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    ULONG&          nElement,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 UDINT 元素的容器
nElement	ULONG&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.34 GetAt (USHORT)

获取容器中指定索引处的 UINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    USHORT&          nElement,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 UINT 元素的容器
nElement	USHORT&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.35 GetAt (ULONGLONG)

获取容器中指定索引处的 ULINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    ULONGLONG&       nElement,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 ULINT 元素的容器
nElement	ULONGLONG&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.36 GetAt (unsigned char)

获取容器中指定索引处的 USINT 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    unsigned char&  nElement,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 USINT 元素的容器
nElement	unsigned char&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.37 GetAt (TcVnVector4_REAL)

获取容器中指定索引处的 TcVnVector4_REAL 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_REAL& aElement,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_REAL 元素的容器
aElement	TcVnVector4_REAL [▶ 1641]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.4.38 GetAt (TcVnRotatedRectangle)

获取容器中指定索引处的 TcVnRotatedRectangle 元素。

语法

定义：

```
HRESULT GetAt(
    HRESULT               hrPrev,
    ITcVnContainer*       ipContainer,
    TcVnRotatedRectangle& stElement,
    ULONGLONG             nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnRotatedRectangle 元素的容器
stElement	TcVnRotatedRectangle [▶ 1718]&	返回指定索引处的元素
nIndex	ULONGLONG	请求元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5 InsertIntoContainer

本章包含根据数据类型在特定的容器位置插入一个元素的功能。

6.2.3.5.5.1 InsertIntoContainer (LONG)

将一个 DINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT         hrPrev,
    LONG            nElement,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	LONG	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.2 InsertIntoContainer (SHORT)

将一个 INT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT         hrPrev,
    SHORT           nElement,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	SHORT	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.3 InsertIntoContainer (ITcVnContainer*)

将一个元素（或一系列元素）插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT         hrPrev,
    ITcVnContainer* ipElement,
```

```
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipElement	ITcVnContainer* [▶ 1827]	插入到 ipContainer 中或包含多个元素的容器中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.4 InsertIntoContainer (ITcVnForwardIterator*)

将一个元素（由迭代器表示）插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT               hrPrev,
    ITcVnForwardIterator* ipElement,
    ITcVnContainer*       ipContainer,
    ULONGLONG             nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipElement	ITcVnForwardIterator* [▶ 1819]	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.5 InsertIntoContainer (ITcVnImage*)

将一个 ITcVnImage 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT          hrPrev,
    ITcVnImage*      ipElement,
    ITcVnContainer*  ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipElement	ITcVnImage* [▶ 1863]	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.6 InsertIntoContainer (double)

将一个 LREAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT          hrPrev,
    double           fElement,
```

```
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fElement	double	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.7 InsertIntoContainer (float)

将一个 REAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT         hrPrev,
    float           fElement,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fElement	float	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.8 InsertIntoContainer (char)

将一个 SINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT         hrPrev,
    char            nElement,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	char	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.9 InsertIntoContainer (TcVnDMatch)

将一个 TcVnDMatch 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT         hrPrev,
    TcVnDMatch&     stElement,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stElement	TcVnDMatch [▶ 1705]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.10 InsertIntoContainer (TcVnKeyPoint)

将一个 TcVnKeyPoint 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT         hrPrev,
    TcVnKeyPoint&   stElement,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stElement	TcVnKeyPoint [▶ 1706]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.11 InsertIntoContainer (TcVnPoint2_DINT)

将一个 TcVnPoint2_DINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT          hrPrev,
    TcVnPoint2_DINT& aElement,
    ITcVnContainer*  ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnPoint2_DINT [▶ 1640]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.12 InsertIntoContainer (TcVnPoint2_LREAL)

将一个 TcVnPoint2_LREAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnPoint2_LREAL& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnPoint2_LREAL [▶ 1640]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.13 InsertIntoContainer (TcVnPoint2_REAL)

将一个 TcVnPoint2_REAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT          hrPrev,
    TcVnPoint2_REAL& aElement,
    ITcVnContainer*  ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnPoint2_REAL [▶ 1640]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.14 InsertIntoContainer (TcVnPoint3_LREAL)

将一个 TcVnPoint3_LREAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnPoint3_LREAL& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnPoint3_LREAL [▶ 1640]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.15 InsertIntoContainer (TcVnPoint3_REAL)

将一个 TcVnPoint3_REAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnPoint3_REAL&  aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnPoint3_REAL [▶ 1640]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.16 InsertIntoContainer (TcVnRectangle_DINT)

将一个 TcVnRectangle_DINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT             hrPrev,
    TcVnRectangle_DINT& stElement,
    ITcVnContainer*     ipContainer,
    ULONGLONG           nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stElement	TcVnRectangle_DINT [▶ 1717]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.17 InsertIntoContainer (TcVnVector2_DINT)

将一个 TcVnVector2_DINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnVector2_DINT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_DINT [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.18 InsertIntoContainer (TcVnVector2_INT)

将一个 TcVnVector2_INT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnVector2_INT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_INT [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.19 InsertIntoContainer (TcVnVector2_REAL)

将一个 TcVnVector2_REAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnVector2_REAL& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_REAL [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.20 InsertIntoContainer (TcVnVector2_SINT)

将一个 TcVnVector2_SINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnVector2_SINT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_SINT [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.21 InsertIntoContainer (TcVnVector2_UINT)

将一个 TcVnVector2_UINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnVector2_UINT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_UINT [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.22 InsertIntoContainer (TcVnVector2_USINT)

将一个 TcVnVector2_USINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT            hrPrev,
    TcVnVector2_USINT& aElement,
    ITcVnContainer*    ipContainer,
    ULONGLONG          nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_USINT [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.23 InsertIntoContainer (TcVnVector3_INT)

将一个 TcVnVector3_INT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT          hrPrev,
    TcVnVector3_INT& aElement,
    ITcVnContainer*  ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector3_INT [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.24 InsertIntoContainer (TcVnVector3_REAL)

将一个 TcVnVector3_REAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnVector3_REAL& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector3_REAL [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.25 InsertIntoContainer (TcVnVector3_SINT)

将一个 TcVnVector3_SINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnVector3_SINT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector3_SINT [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.26 InsertIntoContainer (TcVnVector3_UINT)

将一个 TcVnVector3_UINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnVector3_UINT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector3_UINT [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.27 InsertIntoContainer (TcVnVector3_USINT)

将一个 TcVnVector3_USINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT            hrPrev,
    TcVnVector3_USINT& aElement,
    ITcVnContainer*    ipContainer,
    ULONGLONG          nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector3_USINT [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.28 InsertIntoContainer (TcVnVector4_DINT)

将一个 TcVnVector4_DINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnVector4_DINT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_DINT [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.29 InsertIntoContainer (TcVnVector4_INT)

将一个 TcVnVector4_INT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT          hrPrev,
    TcVnVector4_INT& aElement,
    ITcVnContainer*  ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_INT [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.30 InsertIntoContainer (TcVnVector4_LREAL)

将一个 TcVnVector4_LREAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnVector4_LREAL& aElement,
    ITcVnContainer*    ipContainer,
    ULONGLONG          nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_LREAL [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.31 InsertIntoContainer (TcVnVector4_SINT)

将一个 TcVnVector4_SINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnVector4_SINT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_SINT [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.32 InsertIntoContainer (TcVnVector4_UINT)

将一个 TcVnVector4_UINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnVector4_UINT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_UINT [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.33 InsertIntoContainer (TcVnVector4_USINT)

将一个 TcVnVector4_USINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT            hrPrev,
    TcVnVector4_USINT& aElement,
    ITcVnContainer*    ipContainer,
    ULONGLONG          nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_USINT [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.34 InsertIntoContainer (ULONG)

将一个 UDINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT         hrPrev,
    ULONG           nElement,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	ULONG	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.35 InsertIntoContainer (USHORT)

将一个 UINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT         hrPrev,
    USHORT          nElement,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	USHORT	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.36 InsertIntoContainer (ULONGLONG)

将一个 ULINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT         hrPrev,
    ULONGLONG       nElement,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	ULONGLONG	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.37 InsertIntoContainer (unsigned char)

将一个 USINT 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT         hrPrev,
    unsigned char   nElement,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	unsigned char	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.38 InsertIntoContainer (TcVnVector4_REAL)

将一个 TcVnVector4_REAL 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT           hrPrev,
    TcVnVector4_REAL& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_REAL [▶ 1641]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.5.39 InsertIntoContainer (TcVnRotatedRectangle)

将一个 TcVnRotatedRectangle 类型的元素插入容器中的指定位置之前。

语法

定义：

```
HRESULT InsertIntoContainer(
    HRESULT               hrPrev,
    TcVnRotatedRectangle& stElement,
    ITcVnContainer*       ipContainer,
    ULONGLONG             nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stElement	TcVnRotatedRectangle [▶ 1718]&	插入到 ipContainer 中的单个元素
ipContainer	ITcVnContainer* [▶ 1827]	向其中插入元素的容器
nIndex	ULONGLONG	在其之前插入元素的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6 SetAt

如果需要访问一个容器中的单个元素，应该使用`F_VN_SetAt`函数。如果需要连续处理容器中的所有元素，使用迭代器 [▶ 2149]更好。

6.2.3.5.6.1 SetAt (LONG)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT         hrPrev,
    LONG            nElement,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	LONG	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 DINT 元素的容器，其中 nIndex 位置的元素由 nElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.2 SetAt (SHORT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT          hrPrev,
    SHORT            nElement,
    ITcVnContainer* ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	SHORT	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 INT 元素的容器，其中 nIndex 位置的元素由 nElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.3 SetAt (ITcVnContainer*)

设置容器中指定索引处的元素。（或者使用接口方法 .SetAt。）

语法

定义：

```
HRESULT SetAt(
    HRESULT          hrPrev,
    ITcVnContainer* ipElement,
    ITcVnContainer* ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipElement	ITcVnContainer* [▶ 1827]	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 ITcVnContainer 元素的容器，其中 nIndex 位置的元素由 ipElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.4 SetAt (ITcVnImage*)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT          hrPrev,
    ITcVnImage*      ipElement,
    ITcVnContainer* ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipElement	ITcVnImage* [▶ 1863]	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 ITcVnImage 元素的容器，其中 nIndex 位置的元素由 ipElement 代替

名称	类型	描述
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.5 SetAt (double)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT         hrPrev,
    double          fElement,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fElement	double	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 LREAL 元素的容器，其中 nIndex 位置的元素由 fElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.6 SetAt (float)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT          hrPrev,
    float            fElement,
    ITcVnContainer* ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
fElement	float	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 REAL 元素的容器，其中 nIndex 位置的元素由 fElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.7 SetAt (char)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT          hrPrev,
    char             nElement,
    ITcVnContainer* ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	描述
nElement	char	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 SINT 元素的容器，其中 nIndex 位置的元素由 nElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.8 SetAt (TcVnDMatch)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT          hrPrev,
    TcVnDMatch&      stElement,
    ITcVnContainer* ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stElement	TcVnDMatch [▶ 1705]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnDMatch 元素的容器，其中 nIndex 位置的元素由 stElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.9 SetAt (TcVnKeyPoint)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT          hrPrev,
    TcVnKeyPoint&    stElement,
    ITcVnContainer*  ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stElement	TcVnKeyPoint [▶ 1706]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnKeyPoint 元素的容器，其中 nIndex 位置的元素由 stElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.10 SetAt (TcVnPoint2_DINT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT          hrPrev,
    TcVnPoint2_DINT& aElement,
    ITcVnContainer*  ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnPoint2_DINT [▶ 1640]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnPoint2_DINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.11 SetAt (TcVnPoint2_LREAL)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT           hrPrev,
    TcVnPoint2_LREAL& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnPoint2_LREAL [▶ 1640]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnPoint2_LREAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.12 SetAt (TcVnPoint2_REAL)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT          hrPrev,
    TcVnPoint2_REAL& aElement,
    ITcVnContainer*  ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnPoint2_REAL [▶ 1640]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnPoint2_REAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.13 SetAt (TcVnPoint3_LREAL)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT           hrPrev,
    TcVnPoint3_LREAL& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnPoint3_LREAL [▶ 1640]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnPoint3_LREAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.14 SetAt (TcVnPoint3_REAL)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT           hrPrev,
    TcVnPoint3_REAL& aElement,
    ITcVnContainer*  ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnPoint3_REAL [▶ 1640]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnPoint3_REAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.15 SetAt (TcVnRectangle_DINT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT              hrPrev,
    TcVnRectangle_DINT&  stElement,
    ITcVnContainer*      ipContainer,
    ULONGLONG            nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stElement	TcVnRectangle_DINT [▶ 1717]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnRectangle_DINT 元素的容器，其中 nIndex 位置的元素由 stElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.16 SetAt (TcVnVector2_DINT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT           hrPrev,
    TcVnVector2_DINT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_DINT [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_DINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.17 SetAt (TcVnVector2_INT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT           hrPrev,
    TcVnVector2_INT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_INT [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_INT 元素的容器，其中 nIndex 位置的元素由 aElement 代替

名称	类型	描述
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.18 SetAt (TcVnVector2_REAL)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT           hrPrev,
    TcVnVector2_REAL& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_REAL [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_REAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.19 SetAt (TcVnVector2_SINT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT           hrPrev,
    TcVnVector2_SINT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_SINT [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_SINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.20 SetAt (TcVnVector2_UINT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT           hrPrev,
    TcVnVector2_UINT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	描述
aElement	TcVnVector2_UINT [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_UINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.21 SetAt (TcVnVector2_USINT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT            hrPrev,
    TcVnVector2_USINT& aElement,
    ITcVnContainer*    ipContainer,
    ULONGLONG          nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector2_USINT [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector2_USINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.22 SetAt (TcVnVector3_INT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT          hrPrev,
    TcVnVector3_INT& aElement,
    ITcVnContainer*  ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector3_INT [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector3_INT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.23 SetAt (TcVnVector3_REAL)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT          hrPrev,
    TcVnVector3_REAL& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector3_REAL [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector3_REAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.24 SetAt (TcVnVector3_SINT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT           hrPrev,
    TcVnVector3_SINT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector3_SINT [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector3_SINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.25 SetAt (TcVnVector3_UINT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT           hrPrev,
    TcVnVector3_UINT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector3_UINT [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector3_UINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.26 SetAt (TcVnVector3_USINT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT            hrPrev,
    TcVnVector3_USINT& aElement,
    ITcVnContainer*    ipContainer,
    ULONGLONG          nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector3_USINT [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector3_USINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.27 SetAt (TcVnVector4_DINT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT            hrPrev,
    TcVnVector4_DINT&  aElement,
    ITcVnContainer*    ipContainer,
    ULONGLONG          nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_DINT [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_DINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.28 SetAt (TcVnVector4_INT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT          hrPrev,
    TcVnVector4_INT& aElement,
    ITcVnContainer*  ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_INT [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_INT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.29 SetAt (TcVnVector4_LREAL)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT             hrPrev,
    TcVnVector4_LREAL&  aElement,
    ITcVnContainer*     ipContainer,
    ULONGLONG           nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_LREAL [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_LREAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.30 SetAt (TcVnVector4_SINT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT             hrPrev,
    TcVnVector4_SINT&   aElement,
    ITcVnContainer*     ipContainer,
    ULONGLONG           nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_SINT [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_SINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替

名称	类型	描述
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.31 SetAt (TcVnVector4_UINT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT           hrPrev,
    TcVnVector4_UINT& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_UINT [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_UINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.32 SetAt (TcVnVector4_USINT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT             hrPrev,
    TcVnVector4_USINT&  aElement,
    ITcVnContainer*     ipContainer,
    ULONGLONG           nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_USINT [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_USINT 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.33 SetAt (ULONG)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT          hrPrev,
    ULONG            nElement,
    ITcVnContainer*  ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	描述
nElement	ULONG	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 UDINT 元素的容器，其中 nIndex 位置的元素由 nElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.34 SetAt (USHORT)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT          hrPrev,
    USHORT           nElement,
    ITcVnContainer* ipContainer,
    ULONGLONG        nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	USHORT	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 UINT 元素的容器，其中 nIndex 位置的元素由 nElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.35 SetAt (ULONGLONG)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT         hrPrev,
    ULONGLONG       nElement,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	ULONGLONG	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 ULINT 元素的容器，其中 nIndex 位置的元素由 nElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.36 SetAt (unsigned char)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT         hrPrev,
    unsigned char   nElement,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nElement	unsigned char	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 USINT 元素的容器，其中 nIndex 位置的元素被 nElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.37 SetAt (TcVnVector4_REAL)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT           hrPrev,
    TcVnVector4_REAL& aElement,
    ITcVnContainer*   ipContainer,
    ULONGLONG         nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aElement	TcVnVector4_REAL [▶ 1641]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnVector4_REAL 元素的容器，其中 nIndex 位置的元素由 aElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.6.38 SetAt (TcVnRotatedRectangle)

设置容器中指定索引处的元素。

语法

定义：

```
HRESULT SetAt(
    HRESULT               hrPrev,
    TcVnRotatedRectangle& stElement,
    ITcVnContainer*       ipContainer,
    ULONGLONG             nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stElement	TcVnRotatedRectangle [▶ 1718]&	要设置在指定容器位置的元素
ipContainer	ITcVnContainer* [▶ 1827]	包含 TcVnRotatedRectangle 元素的容器，其中 nIndex 位置的元素由 stElement 代替
nIndex	ULONGLONG	Index（索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.7 AdvanceIterator

将迭代器向前推进指定的偏移量。

语法

定义：

```
HRESULT AdvanceIterator(
    HRESULT                 hrPrev,
    ITcVnForwardIterator* ipIterator,
    LONGLONG                nOffset
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipIterator	ITcVnForwardIterator* [▶ 1819]	要前进的迭代器
nOffset	LONGLONG	ipIterator 前进的偏移量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.8 CheckIfEmpty

检查容器是否为空。（或者使用接口方法 .CheckIfEmpty。）

语法

定义：

```
HRESULT CheckIfEmpty(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.9 CheckIfIteratorIsAtEnd

检查迭代器是否指向最后一个元素的后一个位置。（或者使用接口方法 .CheckIfEnd。）

语法

定义：

```
HRESULT CheckIfIteratorIsAtEnd(
    HRESULT              hrPrev,
    ITcVnForwardIterator* ipIterator
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipIterator	ITcVnForwardIterator* [▶ 1819]	迭代器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.10 ConvertContainerType

将一个容器转换为另一种类型（不支持结构体元素类型）。

语法

定义：

```
HRESULT ConvertContainerType(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer,
    ITcVnContainer*& ipDestContainer,
    GUID             nDestTypeGuid
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	源容器
ipDestContainer	ITcVnContainer* [▶ 1827]&	返回转换后的容器
nDestTypeGuid	GUID	指定转换的目标容器类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.11 CopyContainer

复制一个容器。

语法

定义：

```
HRESULT CopyContainer(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer,
    ITcVnContainer*& ipDestContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	源容器
ipDestContainer	ITcVnContainer* [▶ 1827]&	目标容器（与 ipSrcContainer 的类型 ID 相同；重复使用非零接口指针。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.12 CopyContainerElementsConditional (ITcVnContainer*)

根据自定义条件将容器元素复制到新容器中。

语法

定义：

```
HRESULT CopyContainerElementsConditional(
    HRESULT                                     hrPrev,
    ITcVnContainer*                             ipSrcContainer,
    ITcVnContainer*&                            ipDestContainer,
    ITcVnCustomElementCondition_ITcVnContainer* ipConditionFB
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	源容器
ipDestContainer	ITcVnContainer* [▶ 1827]&	返回一个容器，其中包含符合条件的元素
ipConditionFB	ITcVnCustomElementCondition_ITcVnContainer* [▶ 1721]	自定义条件

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.13 CopyContainerElementsConditional (ITcVnForwardIterator*)

根据自定义条件将容器元素复制到新容器中。

语法

定义：

```
HRESULT CopyContainerElementsConditional(
    HRESULT                                         hrPrev,
    ITcVnContainer*                                 ipSrcContainer,
    ITcVnContainer*&                                ipDestContainer,
    ITcVnCustomElementCondition_ITcVnForwardIterator* ipConditionFB
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	源容器
ipDestContainer	ITcVnContainer* [▶ 1827]&	返回一个容器，其中包含符合条件的元素
ipConditionFB	ITcVnCustomElementCondition_ITcVnForwardIterator* [▶ 1721]	自定义条件

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.14 CreateContainer

创建 GUID 类型为 nTypeGuid 的容器，并返回其容器接口。

语法

定义：

```
HRESULT CreateContainer(
    HRESULT          hrPrev,
    ITcVnContainer*& ipContainer,
    GUID             nTypeGuid,
    ULONGLONG        nElementNum
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]&	返回创建的容器
nTypeGuid	GUID	要创建的容器的 GUID 类型
nElementNum	ULONGLONG	元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.15 CreateContainerFromArray

创建 GUID 类型为 nTypeGuid 的容器，使用提供的数据对其进行初始化，并返回其容器接口。仅支持用于基本容器类型，即不支持用于容器的容器。

语法

定义：

```
HRESULT CreateContainerFromArray(
    HRESULT          hrPrev,
    PVOID            pData,
    ITcVnContainer*& ipContainer,
    GUID             nTypeGuid,
    ULONGLONG        nElementNum
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
pData	PVOID	指向数据数组的指针。确保数组至少包含 nElementNum 个元素，且数组元素类型与容器元素类型一致。
ipContainer	ITcVnContainer* [▶ 1827]&	返回创建的容器
nTypeGuid	GUID	要创建的容器的 GUID 类型
nElementNum	ULONGLONG	从 pData 中复制的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.16 EraseFromContainer

擦除容器中的元素。

语法

定义：

```
HRESULT EraseFromContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    ULONGLONG        nStartIdx,
    ULONGLONG        nCount
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	擦除其中元素的容器
nStartIdx	ULONGLONG	要擦除的第一个元素的索引
nCount	ULONGLONG	要擦除的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.17 ExportContainer

将容器元素导出到缓冲区（如数组）中。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT ExportContainer(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    PVOID            pBuffer,
    ULONGLONG        nBufferSize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
pBuffer	PVOID	用于存储容器元素的缓冲区（确保分配足够的内存！所需大小（以字节为单位）可通过函数 F_VN_ExportContainerSize 来确定）
nBufferSize	ULONGLONG	缓冲区内存大小（以字节为单位）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.18 ExportContainer_String

将容器元素导出为字符串。仅适用于 ContainerType_String_SINT 类型的容器。

语法

定义：

```
HRESULT ExportContainer_String(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    PCHAR           sText,
    ULONGLONG       nMaxLength
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	ContainerType_String_SINT 类型的容器
sText	PCHAR	确保选择足够的字符串大小！所需大小可使用函数 VnExportContainerSize 确定。
nMaxLength	ULONGLONG	要导出的字符串的最大长度（包括 0 结尾）。如果容器内容较长，则在 nMaxLength - 1 处截断字符串，并添加 0 结尾。在这种情况下，将返回 S_FALSE。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.19 ExportContainerSize

确定存储所有容器元素所需的缓冲区大小（以字节为单位）（number_of_Elements * size_per_Element）。仅适用于具有基本元素的容器。

语法

定义：

```
HRESULT ExportContainerSize(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    ULONGLONG&      nBufferSize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nBufferSize	ULONGLONG&	输出参数，包含所需的缓冲区大小

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.20 ExportSubContainer

将子容器的容器元素导出到缓冲区（如数组）中。仅适用于具有基本元素的二维容器。

语法

定义：

```
HRESULT ExportSubContainer(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex,
    PVOID           pBuffer,
    ULONGLONG       nBufferSize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nIndex	ULONGLONG	请求元素的索引

名称	类型	描述
pBuffer	PVOID	用于存储容器元素的缓冲区（确保分配足够的内存！所需大小（以字节为单位）可通过函数 F_VN_ExportSubContainerSize 来确定）
nBufferSize	ULONGLONG	缓冲区内存大小（以字节为单位）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.21 ExportSubContainer_String

将子容器的容器元素导出为字符串。仅适用于 ContainerType_Vector_String_SINT 类型的二维容器。

语法

定义：

```
HRESULT ExportSubContainer_String(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    ULONGLONG        nIndex,
    PCHAR            sText,
    ULONGLONG        nMaxLength
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	ContainerType_Vector_String_SINT 类型的容器
nIndex	ULONGLONG	请求元素的索引
sText	PCHAR	确保选择足够的字符串大小！所需大小可使用函数 VnExportSubContainerSize 确定。
nMaxLength	ULONGLONG	要导出的字符串的最大长度（包括 0 结尾）。如果容器内容较长，则在 nMaxLength - 1 处截断字符串，并添加 0 结尾。在这种情况下，将返回 S_FALSE。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.22 ExportSubContainerSize

确定存储子容器中所有容器元素所需的缓冲区大小（以字节为单位）（number_of_Elements * size_per_Element）。仅适用于具有基本元素的二维容器。

语法

定义：

```
HRESULT ExportSubContainerSize(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    ULONGLONG       nIndex,
    ULONGLONG&      nBufferSize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	包含基本元素的容器
nIndex	ULONGLONG	请求元素的索引
nBufferSize	ULONGLONG&	输出参数，包含所需的缓冲区大小

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.23 ExtractContainerRange

将源容器的指定范围复制到目标容器中。（如果目标容器中已经包含元素，则这些元素将被移除）。

语法

定义：

```
HRESULT ExtractContainerRange(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer,
    ITcVnContainer*& ipDestContainer,
    ULONGLONG        nFirstIdx,
    ULONGLONG        nLastIdx
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	源容器
ipDestContainer	ITcVnContainer* [▶ 1827]&	目标容器
nFirstIdx	ULONGLONG	要复制的第一个元素的索引
nLastIdx	ULONGLONG	要复制的最后一个元素的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.24 GetContainer

获取转换为 ITcVnContainer 接口的当前元素的指针，并递增其引用计数器（仅适用于容器类型）。（或者使用接口方法 .GetContainer。）

语法

定义：

```
HRESULT GetContainer(
    HRESULT               hrPrev,
    ITcVnForwardIterator* ipIterator,
    ITcVnContainer*&      ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipIterator	ITcVnForwardIterator* [▶ 1819]	迭代器
ipContainer	ITcVnContainer* [▶ 1827]&	返回容器接口

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.25 GetContainer (Exp)

获取转换为 ITcVnContainer 接口的当前元素的指针，并递增其引用计数器（仅适用于容器类型）。此外，还可以提供一个偏移量，用于之后递增或递减迭代器。

语法

定义：

```
HRESULT GetContainer(
    HRESULT                hrPrev,
    ITcVnForwardIterator* ipIterator,
    ITcVnContainer*&       ipContainer,
    LONG                   nOffset
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipIterator	ITcVnForwardIterator* [▶ 1819]	迭代器
ipContainer	ITcVnContainer* [▶ 1827]&	返回容器接口
nOffset	LONG	之后递增（>0）或递减（<0）迭代器的偏移量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.26 GetForwardIterator

获取容器的前向迭代器。（或者使用接口方法 .GetForwardIterator。）

语法

定义：

```
HRESULT GetForwardIterator(
    HRESULT                hrPrev,
    ITcVnContainer*        ipContainer,
    ITcVnForwardIterator*& ipIterator
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	容器
ipIterator	ITcVnForwardIterator* [▶ 1819]&	返回迭代器接口

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.27 GetNumberOfElements

获取容器中元素的数量。（或者使用接口方法 .getElementNum。）

语法

定义：

```
HRESULT GetNumberOfElements(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    ULONGLONG&      nNumberOfElements
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	容器
nNumberOfElements	ULONGLONG&	返回 ipContainer 中元素的数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.28 GetRandomAccessIterator

获取容器的随机访问迭代器。（或者使用接口方法 .GetRandomAccessIterator。）

语法

定义：

```
HRESULT GetRandomAccessIterator(
    HRESULT                     hrPrev,
    ITcVnContainer*             ipContainer,
    ITcVnRandomAccessIterator*& ipIterator
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	容器
ipIterator	ITcVnRandomAccessIterator* [▶ 1824]&	返回迭代器接口

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.29 IncrementIterator

递增迭代器。（或者使用接口方法 .Increment。）

语法

定义：

```
HRESULT IncrementIterator(
    HRESULT                hrPrev,
    ITcVnForwardIterator* ipIterator
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipIterator	ITcVnForwardIterator* [▶ 1819]	迭代器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.30 IteratorDistance

计算 2 个迭代器之间的距离。

语法

定义：

```
HRESULT IteratorDistance(
    HRESULT                 hrPrev,
    ITcVnForwardIterator* ipIteratorFirst,
    ITcVnForwardIterator* ipIteratorLast,
    LONGLONG&               nDistance
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipIteratorFirst	ITcVnForwardIterator* [▶ 1819]	第一个迭代器
ipIteratorLast	ITcVnForwardIterator* [▶ 1819]	最后一个迭代器
nDistance	LONGLONG&	返回计算出的距离

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.31 ReserveContainerMemory

预留容器内存（在手动添加元素前应使用最大所需元素数量调用，以提高性能）

语法

定义：

```
HRESULT ReserveContainerMemory(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    ULONGLONG       nElements
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	为其预留内存的容器
nElements	ULONGLONG	容器应预留内存的元素数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.32 ReverseContainer

反转容器元素。

语法

定义：

```
HRESULT ReverseContainer(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	要反转的容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.33 SetContainer

使用 ITcVnContainer 接口设置当前元素（仅适用于容器类型）。（或者使用接口方法 .SetContainer。）

语法

定义：

```
HRESULT SetContainer(
    HRESULT                hrPrev,
    ITcVnForwardIterator* ipIterator,
    ITcVnContainer*       ipContainer
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipIterator	ITcVnForwardIterator* [▶ 1819]	迭代器
ipContainer	ITcVnContainer* [▶ 1827]	将其内容分配给当前元素的容器接口

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.34 SetIteratorToBegin

将迭代器设置为容器的第一个元素。（或者使用接口方法 .SetToBegin。）

语法

定义：

```
HRESULT SetIteratorToBegin(
    HRESULT                 hrPrev,
    ITcVnForwardIterator* ipIterator
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipIterator	ITcVnForwardIterator* [▶ 1819]	迭代器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.5.35 SortContainer

Sort container elements.

Parameters

Name	Type	Default	Description
hrPrev	HRESULT [▶ 135]		HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)
ipContainer	ITcVnContainer* [▶ 1827]		Container to sort (only basic arithmetic, TcVnPoint and TcVnVector containers are supported)
nSortByElemIdx	USHORT		Index of the element data to sort by (0 for arithmetic types, 0 or 1 for TcVnPoint2 and TcVnVector2, 0 to 2 for

Name	Type	Default	Description
			TcVnPoint3 and TcVnVector3, 0 to 3 for TcVnVector4)
eComparisonOp	ETcVnComparisonOperator [▶ 1665]		Element comparison operator (LESS to sort ascending or GREATER to sort descending)
nFirstIdx	ULONGLONG	0	First index of the container to sort (inclusive)
nLastIdx	ULONGLONG	18446744073709551615	Last index of the container to sort (inclusive). Must be >= nFirstIdx. Automatically cropped to the last available index in the container, so it is valid to specify larger values to sort until the end.

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.2.3.5.36 F_VN_SortContainerExp

Sort container elements. (expert function)

Syntax

Definition:

```
FUNCTION SortContainer: HRESULT
VAR_INPUT
ipContainer:     ITcVnContainer [▶ 1827];
nSortByElemIdx:  UINT;
eComparisonOp:   ETcVnComparisonOperator [▶ 1665];
nFirstIdx:       ULINT;
nLastIdx:        ULINT;
hrPrev:          HRESULT [▶ 135]
END_VAR
```

Inputs

Name	Type	Description
ipContainer	ITcVnContainer	Container to sort (only basic arithmetic, TcVnPoint and TcVnVector containers are supported).

Name	Type	Description
nSortByElemIdx	UINT	Index of the element data to sort by (0 for arithmetic types, 0 or 1 for TcVnPoint2 and TcVnVector2, 0 to 2 for TcVnPoint3 and TcVnVector3, 0 to 3 for TcVnVector4.
eComparisonOp	ETcVnComparisonOperator	Element comparison operator (LESS to sort ascending or GREATER to sort descending).
nFirstIdx	ULINT	First index of the container to sort (inclusive).
nLastIdx	ULINT	Last index of the container to sort (inclusive). Must be >= nFirstIdx. Automatically cropped to the last available index in the container, so it is valid to specify larger values to sort until the end.
hrPrev	HRESULT	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

Return Value

HRESULT [▶ 135]

Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6 基本图像操作

该组包含处理图像 [▶ 143]的函数。

函数

图像创建

- ConvertElementType [▶ 2161]
- CreateAssociatedImage [▶ 2165]
- CreateEmptyImage [▶ 2165]
- CreateImage [▶ 2166]
- CreateImageAndSetPixels [▶ 2167]
- CreateImageFromArray [▶ 2167]
- CreateNDimensionalImage [▶ 2168]
- CreateNDimensionalImageAndSetPixels [▶ 2169]
- CreateNDimensionalImageFromArray [▶ 2170]

图像信息

- GetImageHeight [▶ 2176]
- GetImageInfo [▶ 2177]
- GetImageWidth [▶ 2177]

- GetLayoutInfo [▶ 2178]
- GetPixelFormat [▶ 2179]

复制图像

- CopyImage [▶ 2161]
- CopyImageRegion [▶ 2162]
- CopyImageRegionToRegion [▶ 2163]

像素操作

- GetPixel [▶ 2179]
- SetPixel [▶ 2184]
- SetPixels [▶ 2185]

感兴趣区域

- GetRoi [▶ 2180]
- ResetRoi [▶ 2183]
- SetRoi [▶ 2186]
- SetRoi_TcVnRectangle_DINT [▶ 2187]
- SetRoi_TcVnRectangle_UDINT [▶ 2187]

将图像内容导出到内存区域

- ExportImage [▶ 2170]
- ExportImageAsBmp [▶ 2171]
- ExportImageSize [▶ 2173]
- ExportImageAsBmpSize [▶ 2172]

融合图像

- FuseImages [▶ 2174]
- FuseImagesArray [▶ 2175]

处理图像通道

- CombineImageChannels [▶ 2160]
- GetImageChannel [▶ 2175]
- MixImageChannels [▶ 2181]
- SetImageChannel [▶ 2184]
- SplitImageChannels [▶ 2188]

显示图像

- CopyIntoDisplayableImage [▶ 2164]
- TransformIntoDisplayableImage [▶ 2189]

其他

- PadImageBorder [▶ 2181]
- ReinterpretUnsupportedImage [▶ 2182]

还请参阅有关此

ExportImageAsJpeg [▶ 2159]

6.2.3.6.1 ExportImageAsJpeg

Exports the image as jpeg.

Syntax

Definition:

```
HRESULT ExportImageAsJpeg(
    HRESULT           hrPrev,
    ITcVnImage*       ipImage,
    ITcVnDataExport*& ipJpeg,
    LONG              nLumaQuality,
    LONG              nChromaQuality = -1,
    ETcVnJpegSampling eSamplingFactor = JS_420
)
```

Parameters

Name	Type	Default	Description
hrPrev	HRESULT [▶ 135]		HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)
ipImage	ITcVnImage* [▶ 1863]		Source image (ET_USINT or ET_UINT with 1, 3 or 4 channels). ET_UINT only supports lossless compression. 3 channel expects RGB, 4 channel RGBA (alpha is simply ignored, jpeg does not support it).
ipJpeg	ITcVnDataExport* [▶ 1856]&		Returns the jpeg image. The resulting data can be exported via RPC.
nLumaQuality	LONG		Luma quality, which is in a range of 0 - 100 for lossy or 101 - 107 for lossless compression (for lossless, this selects 1 - 7 as the predictor). Recommended: 90 for lossy or 101 for lossless. Lossless compression is significantly slower and leads to larger files, so usually a high-quality lossy compression is the better choice.
nChromaQuality	LONG	-1	Chroma quality (0 - 100) or -1 to use the same as luma. Only used for lossy compression. If chroma quality is different from luma quality, chroma subsampling is disabled

Name	Type	Default	Description
			automatically (samplingFactor JS_444).
eSamplingFactor	ETcVnJpegSampling [▶ 1643]	JS_420	Jpeg chroma sampling factor. Only used for lossy compression.

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

还请参阅有关此

F_VN_ExportImageAsJpegExp [▶ 773]

6.2.3.6.2 CombineImageChannels

将多个单通道图像合并为一个多通道图像。

语法

定义：

```
HRESULT CombineImageChannels(
    HRESULT      hrPrev,
    PVOID        pSrcImages,
    ITcVnImage*& ipDestImage,
    USHORT       nSrcArraySize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
pSrcImages	PVOID	指向单通道源图像数组的指针
ipDestImage	ITcVnImage* [▶ 1863]&	多通道目标图像（如果需要，将创建适当的目标图像）。
nSrcArraySize	USHORT	pSrcImages 数组大小

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.3 ConvertElementType

将图像转换为另一种元素类型。

语法

定义：

```
HRESULT ConvertElementType(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnImage*&     ipDestImage,
    ETcVnElementType eElementType,
    double           fScaleFactor = 1,
    double           fDelta = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的目标图像。）
eElementType	ETcVnElementType [▶ 1674]		所需的目标图像元素类型
fScaleFactor	double	1	像素值的比例系数
fDelta	double	0	与缩放像素值相加的值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.4 CopyImage

创建图像的深拷贝。

语法

定义：

```
HRESULT CopyImage(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.5 CopyImageRegion

将指定的感兴趣区域深拷贝到新图像中。

语法

定义：

```
HRESULT CopyImageRegion(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ULONG        nX,
    ULONG        nY,
    ULONG        nWidth,
    ULONG        nHeight,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
nX	ULONG	左边界（含基于 0 的索引）
nY	ULONG	上边界（含基于 0 的索引）
nWidth	ULONG	ROI 宽度

名称	类型	描述
nHeight	ULONG	ROI 高度
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.6 CopyImageRegionToRegion

将一个图像区域复制到另一个图像区域。

语法

定义：

```
HRESULT CopyImageRegionToRegion(
    HRESULT     hrPrev,
    ITcVnImage* ipSrcImage,
    ULONG       nXSrc,
    ULONG       nYSrc,
    ULONG       nWidth,
    ULONG       nHeight,
    ITcVnImage* ipDestImage,
    ULONG       nXDest,
    ULONG       nYDest
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
nXSrc	ULONG	ipSrcImage 中的左边界（含基于 0 的索引）
nYSrc	ULONG	ipSrcImage 中的上边界（含基于 0 的索引）
nWidth	ULONG	ROI 宽度
nHeight	ULONG	ROI 高度
ipDestImage	ITcVnImage* [▶ 1863]	目标图像（与 ipSrcImage 类型相同）
nXDest	ULONG	ipDestImage 中的左边界（含基于 0 的索引）
nYDest	ULONG	ipDestImage 中的上边界（含基于 0 的索引）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.7 CopyIntoDisplayableImage

将图像复制到可显示图像中，并设置指定时间戳，之后可通过 ITcVnImageTimestamps 读取该时间戳。如果不想在调用此函数后使用 ipSrcImage，则可能需要使用 F_VN_TransformIntoDisplayableImage 代替，以获得更好的性能。

语法

定义：

```
HRESULT CopyIntoDisplayableImage(
    HRESULT                 hrPrev,
    ITcVnImage*             ipSrcImage,
    ITcVnDisplayableImage*& ipDestImage,
    LONGLONG                nTimestamp = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnDisplayableImage* [▶ 1863]&		返回可显示图像
nTimestamp	LONGLONG	0	添加到图像信息中的指定时间戳

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.8 CreateAssociatedImage

创建与源图像共享数据的新图像。例如，并行处理不同的（不相交）ROI 时很有用。

语法

定义：

```
HRESULT CreateAssociatedImage(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	返回创建的图像（重复使用非零接口指针。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.9 CreateEmptyImage

创建一个空图像，不分配任何数据缓冲区。如果创建了新的图像接口，初始引用计数设为 1；如果未创建，则保留计数。

语法

定义：

```
HRESULT CreateEmptyImage(
    HRESULT      hrPrev,
    ITcVnImage*& ipImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]&	返回创建的图像

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.10 CreateImage

创建图像并分配适当的数据缓冲区。如果创建了新的图像接口，初始引用计数设为 1；如果未创建，则保留计数。

语法

定义：

```
HRESULT CreateImage(
    HRESULT          hrPrev,
    ITcVnImage*&     ipImage,
    ULONG            nWidth,
    ULONG            nHeight,
    ETcVnElementType ePixelType,
    ULONG            nChannelNum
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]&	返回创建的图像（重复使用非零接口指针。）
nWidth	ULONG	图像宽度
nHeight	ULONG	图像高度
ePixelType	ETcVnElementType [▶ 1674]	像素类型
nChannelNum	ULONG	通道数量（1 至 255）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.11 CreateImageAndSetPixels

创建图像，分配适当的数据缓冲区，并将所有像素设置为指定值。如果创建了新的图像接口，初始引用计数设为 1；如果未创建，则保留计数。

语法

定义：

```
HRESULT CreateImageAndSetPixels(
    HRESULT             hrPrev,
    ITcVnImage*&        ipImage,
    ULONG               nWidth,
    ULONG               nHeight,
    ETcVnElementType    ePixelType,
    ULONG               nChannelNum,
    TcVnVector4_LREAL&  aValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]&	返回创建的图像（重复使用非零接口指针。）
nWidth	ULONG	图像宽度
nHeight	ULONG	图像高度
ePixelType	ETcVnElementType [▶ 1674]	像素类型
nChannelNum	ULONG	通道数量（1 至 4）
aValue	TcVnVector4_LREAL [▶ 1641]&	像素值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.12 CreateImageFromArray

创建图像并使用提供的数据对其进行初始化。如果创建了新的图像接口，初始引用计数设为 1；如果未创建，则保留计数。

语法

定义：

```
HRESULT CreateImageFromArray(
    HRESULT          hrPrev,
    PVOID            pData,
    ITcVnImage*&     ipImage,
    ULONG            nWidth,
    ULONG            nHeight,
```

```
    ETcVnElementType ePixelType,
    ULONG            nChannelNum
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
pData	PVOID	指向一维数据数组的指针。确保数组至少包含 nWidth * nHeight * nChannelNum 元素，且数组元素类型与 ePixelType 匹配。
ipImage	ITcVnImage* [▶ 1863]&	返回创建的图像（重复使用非零接口指针。）
nWidth	ULONG	图像宽度
nHeight	ULONG	图像高度
ePixelType	ETcVnElementType [▶ 1674]	像素类型
nChannelNum	ULONG	通道数量（1 至 255）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.13 CreateNDimensionalImage

创建 N 维图像并分配适当的数据缓冲区。如果创建了新的图像接口，初始引用计数设为 1；如果未创建，则保留计数。

语法

定义：

```
HRESULT CreateNDimensionalImage(
    HRESULT            hrPrev,
    ITcVnImage*&       ipImage,
    TcVnArray33_UDINT& aShape,
    ETcVnElementType   ePixelType
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]&	返回创建的图像（重复使用非零接口指针。）

名称	类型	描述
aShape	TcVnArray33_UDINT [▶ 1641]&	指定 ipDestImage 的形状。第一个条目是 ipImage 的维数（最多 32），然后是每个维度的大小。
ePixelType	ETcVnElementType [▶ 1674]	像素类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.14 CreateNDimensionalImageAndSetPixels

创建 N 维图像，分配适当的数据缓冲区，并将所有像素设置为指定值。如果创建了新的图像接口，初始引用计数设为 1；如果未创建，则保留计数。

语法

定义：

```
HRESULT CreateNDimensionalImageAndSetPixels(
    HRESULT             hrPrev,
    ITcVnImage*&        ipImage,
    TcVnArray33_UDINT&  aShape,
    ETcVnElementType    ePixelType,
    double              fValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]&	返回创建的图像（重复使用非零接口指针。）
aShape	TcVnArray33_UDINT [▶ 1641]&	指定 ipDestImage 的形状。第一个条目是 ipImage 的维数（最多 32），然后是每个维度的大小。
ePixelType	ETcVnElementType [▶ 1674]	像素类型
fValue	double	像素值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.15 CreateNDimensionalImageFromArray

创建 N 维图像并使用提供的数据对其进行初始化。如果创建了新的图像接口，初始引用计数设为 1；如果未创建，则保留计数。

语法

定义：

```
HRESULT CreateNDimensionalImageFromArray(
    HRESULT             hrPrev,
    PVOID               pData,
    ITcVnImage*&        ipImage,
    TcVnArray33_UDINT&  aShape,
    ETcVnElementType    ePixelType
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
pData	PVOID	指向一维数据数组的指针。确保数组包含的元素数量至少等于 aShape 中所有维度的乘积，且数组元素类型与 ePixelType 匹配。
ipImage	ITcVnImage* [▶ 1863]&	返回创建的图像（重复使用非零接口指针。）
aShape	TcVnArray33_UDINT [▶ 1641]&	指定 ipDestImage 的形状。第一个条目是 ipImage 的维数（最多 32），然后是每个维度的大小。
ePixelType	ETcVnElementType [▶ 1674]	像素类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.16 ExportImage

将图像数据导出到给定的缓冲区（如数组）中。应先调用 F_VN_ExportImageSize，以获取所需的缓冲区大小。

语法

定义：

```
HRESULT ExportImage(
    HRESULT     hrPrev,
    ITcVnImage* ipImage,
    PVOID       pBuffer,
    ULONGLONG   nBufferSize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]	Image
pBuffer	PVOID	指向图像数据存储缓冲区的指针（确保分配足够的内存！所需大小（以字节为单位）可通过函数 F_VN_ExportImageSize 来确定）
nBufferSize	ULONGLONG	缓冲区内存大小（以字节为单位）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.17 ExportImageAsBmp

将图像以位图形式导出到给定的缓冲区（如数组）中。应先调用 F_VN_ExportImageAsBmpSizeExp，以获取所需的缓冲区大小。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT ExportImageAsBmp(
    HRESULT     hrPrev,
    ITcVnImage* ipImage,
    PVOID       pBuffer,
    ULONGLONG   nBufferSize,
    ULONG       nWidth = 0,
    ULONG       nHeight = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]		Image
pBuffer	PVOID		指向位图图像存储缓冲区的指针（确保分配足够的内存！所需大小（以字节为单位）可通过函数 F_VN_ExportImageAsBmpSizeExp 来确定）
nBufferSize	ULONGLONG		缓冲区内存大小（以字节为单位）
nWidth	ULONG	0	所需的宽度（或使用 0 来保持原始宽度）
nHeight	ULONG	0	所需的高度（或使用 0 来保持原始高度）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.18 ExportImageAsBmpSize

获取以位图形式存储图像所需的缓冲区大小（以字节为单位）。

语法

定义：

```
HRESULT ExportImageAsBmpSize(
    HRESULT     hrPrev,
    ITcVnImage* ipImage,
    ULONGLONG&  nBufferSize,
    ULONG       nWidth = 0,
    ULONG       nHeight = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]		Image
nBufferSize	ULONGLONG&		返回所需的缓冲区大小（以字节为单位）
nWidth	ULONG	0	所需的宽度（或使用 0 来保持原始宽度）
nHeight	ULONG	0	所需的高度（或使用 0 来保持原始高度）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.19 ExportImageSize

获取存储图像数据所需的缓冲区大小（以字节为单位）。

语法

定义：

```
HRESULT ExportImageSize(
    HRESULT     hrPrev,
    ITcVnImage* ipImage,
    ULONGLONG&  nBufferSize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]	Image
nBufferSize	ULONGLONG&	返回所需的缓冲区大小（以字节为单位）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.20 FuseImages

垂直融合 2 个图像（主要用于线扫相机）。

语法

定义：

```
HRESULT FuseImages(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage1,
    ITcVnImage*  ipSrcImage2,
    ITcVnImage*& ipDestImage,
    ULONG        nFirstLine,
    ULONG        nNumLines
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]	源图像 1
ipSrcImage2	ITcVnImage* [▶ 1863]	源图像 2
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像
nFirstLine	ULONG	ipSrcImage1 中的行索引，即 ipDestImage 中的第一行
nNumLines	ULONG	应复制到 ipDestImage 的行数，从 ipSrcImage1 中的 nFirstLine 开始。复制完 ipSrcImage1 的最后一行后，其余各行将从行索引 0 开始从 ipSrcImage2 复制。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.21 FuseImagesArray

最多可垂直融合 10 个图像（主要用于线扫相机）。

语法

定义：

```
HRESULT FuseImagesArray(
    HRESULT                 hrPrev,
    TcVnArray10_ITcVnImage& aSrcImageArray,
    ULONG                   nValidImages,
    ITcVnImage*&            ipDestImage,
    ULONG                   nFirstLine,
    ULONG                   nNumLines
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aSrcImageArray	TcVnArray10_ITcVnImage [▶ 1641]&	包含 10 个元素的源图像数组（并非需要填充所有元素，nValidImages 指定实际图像数量）
nValidImages	ULONG	aSrcImageArray 中有效图像的数量
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像
nFirstLine	ULONG	ipSrcImage1 中的行索引，即 ipDestImage 中的第一行
nNumLines	ULONG	应复制到 ipDestImage 的行数，从 aSrcImageArray[0] 中的 nFirstLine 开始。复制完 aSrcImageArray[0] 的最后一行后，其余各行将从 aSrcImageArray[1] 复制（从行索引 0 开始），以此类推。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.22 GetImageChannel

将指定的源图像通道作为单通道图像返回。

语法

定义：

```
HRESULT GetImageChannel(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    USHORT       nChannelIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的图像。）
nChannelIndex	USHORT	请求的源图像通道的索引

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.23 GetImageHeight

返回图像的高度。（或者使用接口方法 .GetHeight。）

语法

定义：

```
HRESULT GetImageHeight(
    HRESULT     hrPrev,
    ITcVnImage* ipImage,
    ULONG&      nHeight
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]	Image
nHeight	ULONG&	返回 ipImage 的高度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.24 GetImageInfo

获取包含图像所有常见元信息的结构体。（或者使用接口方法 .GetImageInfo。）

语法

定义：

```
HRESULT GetImageInfo(
    HRESULT        hrPrev,
    ITcVnImage*    ipImage,
    TcVnImageInfo& stImageInfo
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]	Image
stImageInfo	TcVnImageInfo [▶ 1705]&	返回描述图像的结构体

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.25 GetImageWidth

返回图像的宽度。（或者使用接口方法 .GetWidth。）

语法

定义：

```
HRESULT GetImageWidth(
    HRESULT     hrPrev,
    ITcVnImage* ipImage,
    ULONG&      nWidth
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]	Image
nWidth	ULONG&	返回 ipImage 的宽度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.26 GetLayoutInfo

返回图像的布局信息

语法

定义：

```
HRESULT GetLayoutInfo(
    HRESULT           hrPrev,
    ITcVnImage*       ipImage,
    TcVnArray33_UDINT& aLayoutInfo
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]	源图像
aLayoutInfo	TcVnArray33_UDINT [▶ 1641]&	包含 ipImage 的布局信息。aLayoutInfo 的第一个条目是 ipImage 的维数（最多 32），然后是每个维度的大小。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.27 GetPixel

获取特定像素。

语法

定义：

```
HRESULT GetPixel(
    HRESULT             hrPrev,
    ITcVnImage*         ipSrcImage,
    TcVnVector4_LREAL&  aValue,
    ULONG               nX,
    ULONG               nY
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
aValue	TcVnVector4_LREAL [▶ 1641]&	返回像素值（未使用的通道设为 0）。
nX	ULONG	像素的 x 坐标
nY	ULONG	像素的 y 坐标

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.28 GetPixelFormat

获取图像的像素格式。（或者使用接口方法 .GetPixelFormat。）

语法

定义：

```
HRESULT GetPixelFormat(
    HRESULT          hrPrev,
    ITcVnImage*      ipImage,
    TcVnPixelFormat& stPixelFormat
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]	Image
stPixelFormat	TcVnPixelFormat [▶ 1716]&	返回描述像素格式的结构体

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.29 GetRoi

获取图像中感兴趣区域（ROI）的坐标。

语法

定义：

```
HRESULT GetRoi(
    HRESULT              hrPrev,
    ITcVnImage*          ipSrcImage,
    TcVnRectangle_UDINT& stRoi
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
StRoi	TcVnRectangle_UDINT [▶ 1717]&	返回感兴趣区域的坐标

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.30 MixImageChannels

通过将源图像的指定通道复制到目标图像的指定通道来混合图像通道。

语法

定义：

```
HRESULT MixImageChannels(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnImage*&     ipDestImage,
    ITcVnContainer* ipIndicesFromTo
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的图像。）
ipIndicesFromTo	ITcVnContainer* [▶ 1827]	索引对（ContainerType_Vector_TcVnVector2_DINT），指定应将哪个源通道（TcVnVector2_DINT [0]）复制到哪个目标通道（TcVnVector2_DINT [1]）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.31 PadImageBorder

在原始图像边界周围添加填充。

语法

定义：

```
HRESULT PadImageBorder(
    HRESULT                          hrPrev,
    ITcVnImage*                      ipSrcImage,
    ITcVnImage*&                     ipPaddedImage,
    ULONG                            nTopBorder,
    ULONG                            nBottomBorder,
    ULONG                            nLeftBorder,
    ULONG                            nRightBorder,
    ETcVnBorderInterpolationMethod ePaddingType = BIM_CONSTANT,
    TcVnVector4_LREAL&               aPaddingValue = {0, 0, 0, 0}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipPaddedImage	ITcVnImage* [▶ 1863]&		填充的目标图像（与 ipSrcImage 类型相同，如果需要，将创建适当的目标图像。）
nTopBorder	ULONG		上边界上方的填充高度（以像素为单位）
nBottomBorder	ULONG		下边界下方的填充高度（以像素为单位）
nLeftBorder	ULONG		左边界前的填充宽度（以像素为单位）
nRightBorder	ULONG		右边界后的填充宽度（以像素为单位）
ePaddingType	ETcVnBorderInterpolationMethod [▶ 1648]	BIM_CONSTANT	指定如何确定填充区域的像素值
aPaddingValue	TcVnVector4_LREAL [▶ 1641]&	{0, 0, 0, 0}	如果使用 CONSTANT，则指定填充值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.32 ReinterpretUnsupportedImage

使用附加信息重新解释不受支持的图像数据。如果成功，该图像将变为受支持状态，并带有像素格式标志 bSigned 和 bFloat、像素元素大小 nElementSize 以及颜色通道数 nChannels。该信息根据指定的像素元素类型 ePixelType 确定。

语法

定义：

```
HRESULT ReinterpretUnsupportedImage(
    HRESULT          hrPrev,
    ITcVnImage*      ipImage,
    ETcVnElementType ePixelType
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]	要重新解释的图像
ePixelType	ETcVnElementType [▶ 1674]	用于重新解释图像数据的像素元素类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.33 ResetRoi

重置图像的感兴趣区域（ROI）。（执行此操作后，ROI 将设置为整个图像）。

语法

定义：

```
HRESULT ResetRoi(
    HRESULT     hrPrev,
    ITcVnImage* ipImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]	Image

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.34 SetImageChannel

将指定的目标图像通道设置为指定源图像通道的值。

语法

定义：

```
HRESULT SetImageChannel(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    USHORT       nSrcChannelIndex,
    USHORT       nDestChannelIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的图像。）
nSrcChannelIndex	USHORT	源图像通道的索引
nDestChannelIndex	USHORT	目标图像通道的索引，它将被指定的源图像通道取代

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.35 SetPixel

设置特定像素。

语法

定义：

```
HRESULT SetPixel(
    HRESULT            hrPrev,
    ITcVnImage*        ipSrcImage,
    TcVnVector4_LREAL& aValue,
```

```
    ULONG               nX,
    ULONG               nY
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
aValue	TcVnVector4_LREAL [▶ 1641]&	要设置的像素值（忽略其它通道）。
nX	ULONG	像素的 x 坐标
nY	ULONG	像素的 y 坐标

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.36 SetPixels

将图像的所有像素设置为给定值。

语法

定义：

```
HRESULT SetPixels(
    HRESULT             hrPrev,
    ITcVnImage*         ipDestImage,
    TcVnVector4_LREAL&  aValue,
    ITcVnImage*         ipMask = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
aValue	TcVnVector4_LREAL [▶ 1641]&		像素值
ipMask	ITcVnImage* [▶ 1863]	nullptr	掩膜图像

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.37 SetRoi

在图像中设置感兴趣区域（ROI）。

语法

定义：

```
HRESULT SetRoi(
    HRESULT     hrPrev,
    ULONG       nX,
    ULONG       nY,
    ULONG       nWidth,
    ULONG       nHeight,
    ITcVnImage* ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nX	ULONG	左边界（含基于 0 的索引）
nY	ULONG	上边界（含基于 0 的索引）
nWidth	ULONG	ROI 宽度
nHeight	ULONG	ROI 高度
ipDestImage	ITcVnImage* [▶ 1863]	目标图像

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.38 SetRoi (TcVnRectangle_DINT)

在图像中设置感兴趣区域（ROI）。

语法

定义：

```
HRESULT SetRoi(
    HRESULT              hrPrev,
    TcVnRectangle_DINT& stRoi,
    ITcVnImage*          ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
StRoi	TcVnRectangle_DINT [▶ 1717]&	感兴趣区域
ipDestImage	ITcVnImage* [▶ 1863]	目标图像

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.39 SetRoi (TcVnRectangle_UDINT)

在图像中设置感兴趣区域（ROI）。

语法

定义：

```
HRESULT SetRoi(
    HRESULT              hrPrev,
    TcVnRectangle_UDINT& stRoi,
    ITcVnImage*          ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
StRoi	TcVnRectangle_UDINT [▶ 1717]&	感兴趣区域
ipDestImage	ITcVnImage* [▶ 1863]	目标图像

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.40 SplitImageChannels

将多通道图像分割成多个单通道图像（每个源图像通道 1 个）。

语法

定义：

```
HRESULT SplitImageChannels(
    HRESULT     hrPrev,
    ITcVnImage* ipSrcImage,
    PVOID       pDestImages,
    USHORT      nArraySize,
    USHORT&     nImageChannels
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	多通道源图像
pDestImages	PVOID	指向 ITcVnImage 数组的指针（如果需要，将创建适当的目标图像）
nArraySize	USHORT	pDestImages 数组元素的数量（数组必须 >= ipSourceImage 通道）
nImageChannels	USHORT&	实际源图像通道

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.6.41 TransformIntoDisplayableImage

将图像转换为可显示图像，并设置指定时间戳，之后可通过 ITcVnImageTimestamps 读取该时间戳。源图像将被释放并归零，而现有的目标图像将被释放并被函数结果覆盖。如果 bAllowDeepCopy 等于 false，则源图像不得在其它任何地方使用，而且变换速度总是非常快。否则，如果 bAllowDeepCopy 等于 true，可能会根据需要创建源图像的深拷贝，这将导致执行时间延长。如果要在调用此函数后使用 ipSrcImage，请使用 F_VN_CopyIntoDisplayableImage 代替。

语法

定义：

```
HRESULT TransformIntoDisplayableImage(
    HRESULT                 hrPrev,
    ITcVnImage*&            ipSrcImage,
    ITcVnDisplayableImage*& ipDestImage,
    bool                    bAllowDeepCopy = false,
    LONGLONG                nTimestamp = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]&		源图像
ipDestImage	ITcVnDisplayableImage* [▶ 1863]&		返回可显示图像
bAllowDeepCopy	bool	false	如有需要，允许图像深拷贝
nTimestamp	LONGLONG	0	添加到图像信息中的指定时间戳

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.2.3.7 编码质量

该组包含对各种一维条码和二维条码进行质量评估的函数。评估以国际标准为基础。编码质量检查的目的是确保相应的编码在使用时能被可靠地解码。

为了获得可比较的结果，这些标准规定了测量和评估特定属性以及得出编码质量总体评估的方法。其目的是尽可能预测在不同应用、环境和条件下，使用不同类型的识读器时，编码的预期识读率。因此，通常会为每种应用确定最低可接受质量水平。

一方面，标准描述了所使用的系统必须满足的要求，以便获得可用的结果。另一方面，标准提供了诸如光源、镜头设置和几何结构等建议，以便能够将评级与其他系统进行比较。为了建立一个符合标准的系统，最好先获得相关的质量评估规范和需要检测的编码。

这些函数也可用于非标准系统。这样做的唯一后果是，确定的质量无法与其他系统相比。一般来说，不同的系统在各个评估类别上会有细微差别。

根据 ISO/IEC 15416:2016 标准，Code 39、Code 128、EAN 8、EAN 13、ITF、UPC-A、UPC-E 编码类型的条形码打印质量可通过以下函数确定：

- GradeBarcode [▶ 2190]

根据 ISO/IEC 15415:2011 标准，可以使用以下函数确定 ECC200 Data Matrix（数据矩阵）编码和二维码类型的二维矩阵符号的打印质量：

- GradeDataMatrixCode [▶ 2192]
- GradeQRCode [▶ 2193]

HRESULT 的解释

对于 Code Grading（编码分级）函数，HRESULT [▶ 135] 的返回值如下使用：

编码	名称	描述
0x000	S_OK	该函数已成功执行，所有评估均按照标准进行。
0x001	S_FALSE	函数已成功执行，但对于一维条码，并非所有扫描线都能解码，对于二维条码，则并非所有检查都能按照标准进行。因此，总分设为 0 分，但可以对各项结果进行评估。
0x256	s_watchdogtimeout	函数被 Watchdog（看门狗）中止。可根据终止时间对各项结果进行评估
0x70C	NOTFOUND	函数未成功执行，一维条码无法解码扫描线，二维条码未找到或无法解码。
0x7xx	所有错误代码	函数未成功执行。请参见：ADS Return Codes（ADS 返回代码） [▶ 2857]

6.2.3.7.1 GradeBarcode

根据 ISO/IEC 15416:2016 标准对 1D 条形码进行分级。需要均匀照度、良好的对焦以及图像传感器平面与编码平面平行，才能获得有意义的结果。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT GradeBarcode(
    HRESULT                     hrPrev,
    ITcVnImage*                 ipSrcImage,
    ITcVnContainer*&            ipDecodedData,
    TcVnCodeGrades1D&           stCodeGrades,
    ETcVnBarcodeType            eBarcodeType,
    float&                      fAngleDeg,
    ETcVnBarcodeSearchDirection eSearchDirection = BSD_ANY,
    ITcVnImage**                pipResultVisualization = nullptr,
    ITcVnContainer**            pipScanLineInfo = nullptr,
    ITcVnContainer**            pipContour = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像，仅包含水平对齐的条形码区域（USINT，1 通道，包括左右两侧的安静区，不包括顶部或底部超过 10% 总高度的文本或空白空间）
ipDecodedData	ITcVnContainer* [▶ 1827]&		返回解码数据（ContainerType_String_SINT）
stCodeGrades	TcVnCodeGrades1D [▶ 1703]&		返回编码等级，范围从 0（非常差）至 4（非常好）
eBarcodeType	ETcVnBarcodeType [▶ 1646]		条形码类型（支持：CODE39、CODE128、EAN8、EAN13、ITF、UPCA、UPCE）
fAngleDeg	float&		以度为单位返回大致的顺时针旋转角度
eSearchDirection	ETcVnBarcodeSearchDirection [▶ 1646]	BSD_ANY	条形码搜索方向（BSD_ANY 首先尝试水平方向，然后是垂直方向）
pipResultVisualization	ITcVnImage* [▶ 1863]*	nullptr	返回扫描线的可视化信息（可选项，如果不需要，则设为 0）。等级用颜色标示：>= 3 为绿色，>= 2 为浅蓝色，>= 1 为橙色，< 1 为红色
pipScanLineInfo	ITcVnContainer* [▶ 1827]*	nullptr	返回用于进一步评估的附加扫描线信息，例如用于可视化的 PlotIntensityProfile（可选项，如果不需要，则设为 0；ContainerType_Vector_TcVnVector3_REAL，包含[行索引、整体等级、阈值]）
pipContour	ITcVnContainer* [▶ 1827]*	nullptr	返回 4 个大致编码角点，包括安静区（可选项，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_DINT）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Code Quality

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.7.2 GradeDataMatrixCode

根据 ISO/IEC 15415:2011 标准对数据矩阵编码（ECC200）进行分级。需要均匀照度、良好的对焦以及图像传感器平面与编码平面平行，才能获得有意义的结果。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT GradeDataMatrixCode(
    HRESULT           hrPrev,
    ITcVnImage*       ipSrcImage,
    ITcVnContainer*&  ipDecodedData,
    TcVnCodeGradesDM& stCodeGrades,
    float             fModuleWidth,
    float&            fAngleDeg,
    ITcVnImage**      pipResultVisualization = nullptr,
    ITcVnContainer**  pipMarginGrades = nullptr,
    ITcVnContainer**  pipContour = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像，仅包含数据矩阵编码区域（USINT，1 通道，包括安静区）
ipDecodedData	ITcVnContainer* [▶ 1827]&		返回解码数据（ContainerType_String_SINT）
stCodeGrades	TcVnCodeGradesDM [▶ 1704]&		返回编码等级，范围从 0（非常差）至 4（非常好）
fModuleWidth	float		图像中编码的最小模块宽度（单位：像素）。必须至少为 3，最好在 5 - 8 之间。
fAngleDeg	float&		以度为单位返回顺时针旋转角度
pipResultVisualization	ITcVnImage* [▶ 1863]*	nullptr	返回扫描网格的可视化信息（可选项，如果不需要，则设为 0）。模块边

名称	类型	默认值	描述
			距等级用颜色标示：>= 3 为绿色，>= 2 为浅蓝色，>= 1 为橙色，< 1 为红色
pipMarginGrades	ITcVnContainer* [▶ 1827]*	nullptr	返回每个模块的单个边距等级，例如用于创建自定义可视化（可选项，如果不需要，则设为 0；ContainerType_Vector_TcVnVector3_REAL，包含 [x 位置、y 位置、等级]）
pipContour	ITcVnContainer* [▶ 1827]*	nullptr	返回 4 个编码角点，不包括安静区（可选项，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_DINT）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Code Quality

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.7.3 GradeQRCode

根据 ISO/IEC 15415:2011 标准对二维码进行分级。需要均匀照度、良好的对焦以及图像传感器平面与编码平面平行，才能获得有意义的结果。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT GradeQRCode(
    HRESULT           hrPrev,
    ITcVnImage*       ipSrcImage,
    ITcVnContainer*&  ipDecodedData,
    TcVnCodeGradesQR& stCodeGrades,
    float             fModuleWidth,
    float&            fAngleDeg,
    ITcVnImage**      pipResultVisualization = nullptr,
    ITcVnContainer**  pipMarginGrades = nullptr,
    ITcVnContainer**  pipContour = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像，仅包含二维码区域（USINT，1 通道，包括安静区）
ipDecodedData	ITcVnContainer* [▶ 1827]&		返回解码数据（ContainerType_String_SINT）
stCodeGrades	TcVnCodeGradesQR [▶ 1704]&		返回编码等级，范围从 0（非常差）至 4（非常好）
fModuleWidth	float		图像中编码的最小模块宽度（单位：像素）。必须至少为 3，最好在 5 - 8 之间。
fAngleDeg	float&		以度为单位返回顺时针旋转角度
pipResultVisualization	ITcVnImage* [▶ 1863]*	nullptr	返回扫描网格的可视化信息（可选项，如果不需要，则设为 0）。模块边距等级用颜色标示：>= 3 为绿色，>= 2 为浅蓝色，>= 1 为橙色，< 1 为红色
pipMarginGrades	ITcVnContainer* [▶ 1827]*	nullptr	返回每个模块的单个边距等级，例如用于创建自定义可视化（可选项，如果不需要，则设为 0；ContainerType_Vector_TcVnVector3_REAL，包含 [x 位置、y 位置、等级]）
pipContour	ITcVnContainer* [▶ 1827]*	nullptr	返回 4 个编码角点，不包括安静区（可选项，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_DINT）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Code Quality

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.8 编码读取

该组包含检测和读取一维和二维码的函数。

函数

编码检测

- DetectCodesNeuralNetwork [▶ 2199]

一维条码读取

- ReadBarcode(Exp) [▶ 2200]
  - CodaBar
  - Code39
  - Code93
  - Code128
  - EAN8
  - EAN13
  - ITF
  - UPCA
  - UPCE
  - Code39Extended
- ReadBarcodeRoi(Exp) [▶ 2201]
  - Code39
  - Code128
  - EAN8
  - EAN13
  - ITF
  - UPCA
  - UPCE
- ReadPharmaCode [▶ 2208]

二维码读取

- ReadDataMatrixCode [▶ 2204]
- ReadDataMatrixCodeRoi [▶ 2205]
- ReadQRCode [▶ 2209]
- ReadDotCodeRoi [▶ 2206]

编码要求

一般来说，模块大小至少需为三个像素；建议使用三个以上像素，以获得更稳定的结果。图像应该清晰且具有很高的对比度。此外，对于一维条码，建议条码方向为大约 0° 或 90°，以便使条码与图像轴平行。

HRESULT 的解释

对于读码函数，HRESULT [▶ 135] 的返回值如下使用：

代码	名称	描述
0x000	S_OK	函数已成功执行且所有预期代码已找到（对于标准函数为至少一个代码，而对于专家函数则为 nCodeNumber 个）。

代码	名称	描述
0x001	S_FALSE	函数已成功执行且所有预期代码已找到（对于标准函数为至少一个代码，而对于专家函数则为 nCodeNumber 个）。
0x256	s_watchdogtimeout	函数被 Watchdog（看门狗）中止。一些读码函数可以返回部分结果（请参见单个函数描述）。
0x7xx	所有错误代码	函数未成功执行。请参见：ADS 返回代码 [▶ 2857]

因此，使用 `SUCCEEDED()` 进行常规检查不足以推断出找到的代码，也就无法推断出 `ipDecodedData` 中的现有数据。为此可使用以下查询：

```
if (hr == S_OK)
{
    // Export Code into String
    hr = ExportSubContainer_String(hr, ipDecodedCode, 0, sCodeAsString, 255);
    // Use sCodeAsString
}
else if (SUCCEEDED(hr))
{
    // Process partial results
}
else
{
    // Error handling
}
```

二维码搜索策略

对于二维码，会对输入图像的不同变体应用搜索算法，以便识别尽可能多的编码。图像反转和镜像都用于生成各种变体。决定测试哪个变体称为搜索策略。为了让您对函数运行时有更大的控制，可以对这种搜索策略进行配置。ETcVn2dCodeSearchStrategy [▶ 1645] 枚举可用于此目的。借此，您可以为每个可能的图像变换（反转和镜像）指定是否应用，以及是否应首先检查原始图像或变换图像的编码。

`eSearchStrategy` 函数参数精确定义相应读码专家函数的搜索策略。下表显示不同的组合：

表 1: *反转*

eSearchStrategy	原始图像	反转图像
CSS_ONLY_NOT_INVERTED	1.	-
CSS_FIRST_NOT_INVERTED	1.	2.
CSS_ONLY_INVERTED	-	1.
CSS_FIRST_INVERTED	2.	1.

表 2: *镜像*

eSearchStrategy	原始图像	镜像图像
CSS_ONLY_NOT_FLIPPED	1.	-
CSS_FIRST_NOT_FLIPPED	1.	2.
CSS_ONLY_FLIPPED	-	1.
CSS_FIRST_FLIPPED	2.	1.

可以为每种类型的变换定义搜索策略。不同搜索策略的搜索策略可如下链接：

```
eSearchStrategy = CSS_ONLY_FLIPPED | CSS_FIRST_INVERTED;
```

`CSS_DEFAULT` 选项可用于选择根据编码类型而变化的默认设置。

无法链接搜索策略

不允许为同一变换类型链接多个搜索策略。仅可以相互链接一个反转策略和一个镜像策略。

还请参阅有关此

- AnalyzeBarcodeWalsh [▶ 2197]
- DetectBarcodesWalsh [▶ 2198]

6.2.3.8.1 AnalyzeBarcodeWalsh

Analyze an exemplary barcode image and estimate the following parameters for calls of F_VN_DetectBarcodesWalsh: nMaxBarDistance, fMinAspectRatio, fMaxAspectRatio, nIntensityDifference, nMinArea, and nChannelIndex. The depicted barcode must be representative for the barcodes to be detected by F_VN_DetectBarcodesWalsh.
Can use available TwinCAT Job Tasks for executing parallel code regions.

Syntax

Definition:

```
HRESULT AnalyzeBarcodeWalsh(
    HRESULT                          hrPrev,
    ITcVnImage*                      ipBarcodeImage,
    TcVnParamsWalshBarcodeDetection& stParams
)
```

Parameters

Name	Type	Description
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)
ipBarcodeImage	ITcVnImage* [▶ 1863]	Image showing a horizontal barcode surrounded by a narrow non-barcode region in each direction. The total size of the non-barcode region must not exceed 30% of the image size.
stParams	TcVnParamsWalshBarcodeDetection [▶ 1698]&	Returns the estimated parameters for use with F_VN_DetectBarcodesWalsh. The values of the other parameters are not changed.

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

还请参阅有关此

- DetectBarcodesWalsh [▶ 2198]
- TcVnParamsWalshBarcodeDetection [▶ 1698]

6.2.3.8.2 DetectBarcodesWalsh

Detect barcodes using the Walsh transform.
Can use available TwinCAT Job Tasks for executing parallel code regions.

Syntax

Definition:

```
HRESULT DetectBarcodesWalsh(
    HRESULT                          hrPrev,
    ITcVnImage*                      ipSrcImage,
    ITcVnContainer*&                 ipBarcodes,
    TcVnParamsWalshBarcodeDetection& stParams,
    ULONG                            eOptions = DBWO_HORIZONTAL
)
```

Parameters

Name	Type	Default	Description
hrPrev	HRESULT [▶ 135]		HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)
ipSrcImage	ITcVnImage* [▶ 1863]		Source image
ipBarcodes	ITcVnContainer* [▶ 1827]&		Returns the code positions as rotated rectangles (ContainerType_Vector_TcVnRotatedRectangle; Non-zero interface pointers are reused.)
stParams	TcVnParamsWalshBarcodeDetection [▶ 1698]&		Parameters for configuring the individual detection steps.
eOptions	ULONG	DBWO_HORIZONTAL	General flags controlling the detection process. (see ETcVnDetectBarcodesWalshOptions; default: TCVN_DBWO_HORIZONTAL)

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

还请参阅有关此

- DetectCodesNeuralNetwork [▶ 2199]
- ETcVnDetectBarcodesWalshOptions [▶ 1642]
- TcVnParamsWalshBarcodeDetection [▶ 1698]
- AnalyzeBarcodeWalsh [▶ 2197]

6.2.3.8.3 DetectCodesNeuralNetwork

使用基于 eModelType 的经过训练的神经网络模型，在提供的图像中检测各种编码类型，例如条形码、QR 码、数据矩阵码和 DotCode (点码）。检测到的编码区域和类型（如支持）将按其模型置信度从高到低排序。可以使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT DetectCodesNeuralNetwork(
    HRESULT                 hrPrev,
    ITcVnImage*             ipSrcImage,
    ITcVnContainer*&        ipCodeRegions,
    ITcVnContainer*&        ipCodeTypes,
    ETcVnCodeDetectionModel eModelType,
    ULONGLONG               nCodeType
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（USINT 元素，1 或 3 通道）。该函数会自动将输入归一化并转换为选定 eModelType 所需的格式。
ipCodeRegions	ITcVnContainer* [▶ 1827]&	根据 eModelType 以直立矩形或旋转矩形的形式返回码位置（ContainerType_Vector_TcVnRectangle_DINT 或 ContainerType_Vector_TcVnRotatedRectangle；重复使用非零接口指针。）
ipCodeTypes	ITcVnContainer* [▶ 1827]&	返回每个检测到的编码（ContainerType_Vector_ULINT；重复使用非零接口指针）的编码类型（ETcVnCodeType）；如果不需要，则设为 0。
eModelType	ETcVnCodeDetectionModel [▶ 1651]	用于编码检测的神经网络模型的类型。提供多种不同的神经网络模型，包括专用于一维条码、二维码或所有编码类型的模型。
nCodeType	ULONGLONG	指定要检测的编码类型。通过执行按位 OR 运算，可以将 ETcVnCodeType 枚举中的多个值组合在一起。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Code Reading

6.2.3.8.4 ReadBarcode

检测并解释所提供图像中的一维码。 可由看门狗取消。

语法

定义：

```
HRESULT ReadBarcode(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipDecodedData,
    ULONG            eBarcodeType
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（USINT 元素，1 通道或 3 通道（3 通道输入应为 RGB 并内部转换为灰色））
ipDecodedData	ITcVnContainer* [▶ 1827]&	返回解码后的编码（ContainerType_Vector_String_SINT）。
eBarcodeType	ULONG	要搜索的条形码类型（ETcVnBarcodeType）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.8.5 ReadBarcode (Exp)

检测并解释所提供图像中的一维码。 可由看门狗取消。

语法

定义：

```
HRESULT ReadBarcode(
    HRESULT                 hrPrev,
    ITcVnImage*             ipSrcImage,
    ITcVnContainer*&        ipDecodedData,
    ITcVnContainer*&        ipContours,
    ULONG                   eBarcodeType,
    LONG                    nCodeNumber,
```

```
    ETcVnBarcodeSearchDirection eSearchDirection,
    ITcVnContainer**            pipAngles = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT 元素，1 通道或 3 通道（3 通道输入应为 RGB 并内部转换为灰色））
ipDecodedData	ITcVnContainer* [▶ 1827]&		返回解码后的编码（ContainerType_Vector_String_SINT）。
ipContours	ITcVnContainer* [▶ 1827]&		以轮廓形式返回编码位置（可选项，如果不需要，则设为 0；ContainerType_Vector_Vector_TcVnPoint2_DINT）
eBarcodeType	ULONG		要搜索的条形码类型（ETcVnBarcodeType）
nCodeNumber	LONG		在提供的图像中应检测到的一维码数量（目前仅支持 1 个）。
eSearchDirection	ETcVnBarcodeSearchDirection [▶ 1646]		条形码搜索方向。
pipAngles	ITcVnContainer* [▶ 1827]*	nullptr	返回顺时针编码旋转角度（可选项，如果不需要，则设为 0；ContainerType_Vector_REAL）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.8.6 ReadBarcodeRoi

检测并解释所提供图像中的一维条码。为获得最佳性能，编码应该水平对齐并位于图像中心。

语法

定义：

```
HRESULT ReadBarcodeRoi(
    HRESULT                     hrPrev,
    ITcVnImage*                 ipSrcImage,
    ITcVnContainer*&            ipDecodedData,
    ETcVnBarcodeType            eBarcodeType,
    float&                      fAngleDeg,
    ETcVnBarcodeSearchDirection eSearchDirection = BSD_ANY,
    ITcVnContainer**            pipContour = nullptr,
    LONG                        nMaxScans = 30
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT，1 通道）
ipDecodedData	ITcVnContainer* [▶ 1827]&		返回解码数据（ContainerType_String_SINT）
eBarcodeType	ETcVnBarcodeType [▶ 1646]		条形码类型（支持：CODE39、CODE128、EAN8、EAN13、ITF、UPCA、UPCE）
fAngleDeg	float&		以度为单位返回大致的顺时针旋转角度
eSearchDirection	ETcVnBarcodeSearchDirection [▶ 1646]	BSD_ANY	条形码搜索方向（BSD_ANY 首先尝试水平方向，然后是垂直方向）
pipContour	ITcVnContainer* [▶ 1827]*	nullptr	返回 4 个码角点的大致位置，每个位置均包含了静区所占的区域。宽度是准确的，但出于性能原因，不测量实际高度。相反，将其静态设置为宽度的 10%，并以已解码的扫描线为中心。（可选项，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_DINT）
nMaxScans	LONG	30	为查找条形码而评估的扫描线的最大数量。扫描始终从图像中心开始，然后以（ipSrcImage 高度 / nMaxScans）为步长，交替向上和向下进行，距离逐渐增大。

返回值

HRESULT [▶ 135]

更多信息

`F_VN_ReadBarcodeRoi` 函数经过优化，可在规定图像区域（ROI）内检测单个条形码。

算法

为此，该函数使用了一种定制算法，要求编码在传输图像中居中。该函数首先会搜索图像中水平排列的编码，然后搜索垂直排列的编码。

参数

输入图像

`ipSrcImage` 输入图像必须是元素类型为 `USINT`（8 位）的单通道图像。编码必须在图像中心水平或垂直对齐。

读取数据

从编码中读取的数据以字符串形式返回 `ipDecodedData` 容器中。容器类型为 `ContainerType_String_SINT`。成功找到编码并解码后，便可以使用 F_VN_ExportContainer_String [▶ 2144] 函数将内容导出为字符串。

```
hr := F_VN_ExportContainer_String(ipDecodedData, sText, nMaxLength, hr);
```

条形码类型

`eBarcodeType` 参数定义了要读取的条形码类型。每次函数调用只能传输一种条形码类型。支持 ETcVnBarcodeType [▶ 1646] 枚举中的以下条形码类型：

- `TCVN_BT_CODE39`
- `TCVN_BT_CODE128`
- `TCVN_BT_EAN8`
- `TCVN_BT_EAN13`
- `TCVN_BT_ITF`
- `TCVN_BT_UPCA`
- `TCVN_BT_UPCE`

旋转角度 (返回值)

`fAngleDeg` 返回值以 90° 为单位返回顺时针旋转角度。

专家参数

专家版 F_VN_ReadBarcodeRoiExp [▶ 847] 包含附加参数。

样本

- EAN-13 条形码读取 [▶ 2734]
- 读码过程中的结果评估 [▶ 2745]

相关函数

- F_VN_GradeBarcode [▶ 2190]
- F_VN_GradeDataMatrixCode [▶ 2192]
- F_VN_GradeQRCode [▶ 2193]
- F_VN_ReadBarcode [▶ 2200]
- F_VN_ReadBarcodeRoi [▶ 2201]
- F_VN_ReadPharmaCode [▶ 2208]
- F_VN_ReadDataMatrixCode [▶ 2204]
- F_VN_ReadDataMatrixCodeRoi [▶ 2205]
- F_VN_ReadQRCode [▶ 2209]

- F_VN_ReadDotCodeRoi [▶ 2206]

所需的许可证

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.8.7 ReadDataMatrixCode

在提供的图像中检测并解释数据矩阵编码（ECC200）。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT ReadDataMatrixCode(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipDecodedData,
    ITcVnContainer** pipContours = nullptr,
    LONG             nCodeNumber = 1,
    ULONG            eSearchStrategy = CSS_DEFAULT,
    ITcVnContainer** pipAngles = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT 元素，1 或 3 通道）
ipDecodedData	ITcVnContainer* [▶ 1827]&		返回解码后的编码（ContainerType_Vector_String_SINT）。
pipContours	ITcVnContainer* [▶ 1827]*	nullptr	以轮廓形式返回编码位置（可选项，如果不需要，则设为 0；ContainerType_Vector_Vector_TcVnPoint2_DINT）
nCodeNumber	LONG	1	在提供的图像中应检测到的数据矩阵编码数量。（如果设置为 -1，则会尝试检测所有数据矩阵编码）。
eSearchStrategy	ULONG	CSS_DEFAULT	使用的搜索策略（ETcVn2dCodeSearchStrategy）

名称	类型	默认值	描述
pipAngles	ITcVnContainer* [▶ 1827]*	nullptr	返回顺时针编码旋转角度（可选项，如果不需要，则设为 0；ContainerType_Vector_REAL）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.8.8 ReadDataMatrixCodeRoi

在提供的图像中检测并解释数据矩阵编码（ECC200）。代码应该位于图像中心。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT ReadDataMatrixCodeRoi(
    HRESULT           hrPrev,
    ITcVnImage*       ipSrcImage,
    ITcVnContainer*&  ipDecodedData,
    float             fModuleWidth,
    float&            fAngleDeg,
    ITcVnContainer**  pipContour = nullptr,
    ULONGLONG         eOptions = DMO_DEFAULT,
    unsigned char     nRows = 0,
    unsigned char     nCols = 0,
    bool              bExtensiveSearch = true
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT 元素，1 通道）
ipDecodedData	ITcVnContainer* [▶ 1827]&		返回解码后的编码（ContainerType_String_SINT）
fModuleWidth	float		图像中编码的最小模块宽度（单位：像素）。必须至少为 3，最好在 5 - 8 之间。

名称	类型	默认值	描述
fAngleDeg	float&		以度为单位返回顺时针旋转角度
pipContour	ITcVnContainer* [▶ 1827]*	nullptr	返回检测到的编码的轮廓（ContainerType_Vector_TcVnPoint2_DINT；可选项，如果不需要，则设为 0）
eOptions	ULONGLONG	DMO_DEFAULT	指定一个或多个 ETcVnDataMatrixOptions 的组合
nRows	unsigned char	0	指定编码模块行数，包括查找器和时钟模式，不包括安静区（8 - 144；设为 0，可搜索任何大小）
nCols	unsigned char	0	指定编码模块列数，包括查找器和时钟模式，不包括安静区（10 - 144；设为 0，可搜索任何大小）
bExtensiveSearch	BOOL	true	如果为 true，则在指定选项失败时尝试不同的方法

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.8.9 ReadDotCodeRoi

检测并解释所提供图像中的 DotCode (点码）。编码应在图像中心，并水平或垂直对齐。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT ReadDotCodeRoi(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipDecodedData,
    float            fModuleWidth,
    float&           fAngleDeg,
    ITcVnContainer** pipContour = nullptr,
    ULONGLONG        eOptions = DCO_DEFAULT
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT 元素，1 通道）
ipDecodedData	ITcVnContainer* [▶ 1827]&		返回解码后的编码（ContainerType_String_SINT）
fModuleWidth	float		图像中编码的平均点直径（以像素为单位）。必须至少为 3，最好在 5 - 8 之间。
fAngleDeg	float&		以度为单位返回顺时针旋转角度
pipContour	ITcVnContainer* [▶ 1827]*	nullptr	返回检测到的编码的轮廓（ContainerType_Vector_TcVnPoint2_DINT；可选项，如果不需要，则设为 0）
eOptions	ULONGLONG	DCO_DEFAULT	指定一个或多个 ETcVnDotCodeOptions 的组合

返回值

HRESULT [▶ 135]

更多信息

`F_VN_ReadDotCodeRoi` 函数在指定的感兴趣区域（ROI）内搜索单个 DotCode（点码） 并对其进行解码。这要求编码在所提供的图像中居中。

fModuleWidth

`fModuleWidth` 参数定义了单个 DotCode（点码） 模块（点）的预期平均直径（以像素为单位）。模块大小至少须为 3 个像素，建议使用 5 到 8 个像素以获得最佳且可靠的结果。与参数设置值存在显著偏差可能会导致处理时间增加或无结果。

旋转角度 (返回值)

返回值 `fAngleDeg` 返回顺时针旋转角度。

示例

- DotCode 读取 [▶ 2743]
- 读码过程中的结果评估 [▶ 2745]

相关函数

- F_VN_GradeBarcode [▶ 2190]
- F_VN_GradeDataMatrixCode [▶ 2192]
- F_VN_GradeQRCode [▶ 2193]
- F_VN_ReadBarcode [▶ 2200]

- F_VN_ReadBarcodeRoi [▶ 2201]
- F_VN_ReadPharmaCode [▶ 2208]
- F_VN_ReadDataMatrixCode [▶ 2204]
- F_VN_ReadDataMatrixCodeRoi [▶ 2205]
- F_VN_ReadQRCode [▶ 2209]
- F_VN_ReadDotCodeRoi [▶ 2206]

所需的许可证

TC3 Vision Code Reading

6.2.3.8.10 ReadPharmaCode

检测并解释所提供图像中的药品二进制码。 可由看门狗取消。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT ReadPharmaCode(
    HRESULT           hrPrev,
    ITcVnImage*       ipSrcImage,
    ITcVnContainer*&  ipDecodedData,
    ITcVnContainer**  pipContours = nullptr,
    LONG              nCodeNumber = 1,
    SHORT             nMinBarNumber = 4,
    ITcVnContainer**  pipAngles = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（1 通道（所有元素类型）或带有 TCVN_ET_USINT、TCVN_ET_UINT 或 TCVN_ET_REAL 类型元素的 3 通道（3 通道输入应为 RGB 并在内部转换为灰色））
ipDecodedData	ITcVnContainer* [▶ 1827]&		返回解码后的编码（ContainerType_Vector_String_SINT）。
pipContours	ITcVnContainer* [▶ 1827]*	nullptr	以轮廓形式返回编码位置（可选项，如果不需要，则设为 0；ContainerType_Vector_Vector_TcVnPoint2_DINT）
nCodeNumber	LONG	1	在提供的图像中应检测到的药品二进制码数量（目前仅支持 1 个）。

名称	类型	默认值	描述
nMinBarNumber	SHORT	4	编码必须具有的最少（暗）条数。
pipAngles	ITcVnContainer* [▶ 1827]*	nullptr	返回顺时针编码旋转角度（可选项，如果不需要，则设为 0；ContainerType_Vector_REAL）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.8.11 ReadQRCode

在提供的图像中检测并解释二维码。 可由看门狗取消。

语法

定义：

```
HRESULT ReadQRCode(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipDecodedData,
    ITcVnContainer** pipContours = nullptr,
    LONG             nCodeNumber = 1,
    ULONG            eSearchStrategy = CSS_DEFAULT,
    ITcVnContainer** pipAngles = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT 元素，1 通道或 3 通道（3 通道输入应为 RGB 并内部转换为灰色））
ipDecodedData	ITcVnContainer* [▶ 1827]&		返回解码后的编码（ContainerType_Vector_String_SINT）。
pipContours	ITcVnContainer* [▶ 1827]*	nullptr	以轮廓形式返回编码位置（可选项，如果不需要，则设为 0；

名称	类型	默认值	描述
			ContainerType_Vector_Vector_TcVnPoint2_DINT)
nCodeNumber	LONG	1	在提供的图像中应检测到的二维码数量（目前仅支持 1 个）。
eSearchStrategy	ULONG	CSS_DEFAULT	使用的搜索策略（ETcVn2dCodeSearchStrategy）
pipAngles	ITcVnContainer* [▶ 1827]*	nullptr	返回顺时针编码旋转角度（可选项，如果不需要，则设为 0；ContainerType_Vector_REAL）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Code Reading

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9 容器统计

该组包含对容器 [▶ 146]进行统计分析的功能。

函数

使用欧几里得范数

- ContainerAverage [▶ 2275]
- ContainerAverageVariance [▶ 2277]
- MaxElement [▶ 2280]
- MedianElement [▶ 2280]
- MinElement [▶ 2281]

按元素

- ContainerAverageElementwise [▶ 2275]
- ContainerAverageVarianceElementwise [▶ 2278]
- MaxElementElementwise [▶ 2211]
- MedianElementElementwise [▶ 2232]
- MinElementElementwise [▶ 2253]

6.2.3.9.1 MaxElementElementwise

本章包含一些函数，通过这些函数可根据每个元素的数据类型来确定最大值。这个函数适用于包括基本元素的容器。

例如，对于 TcVnPoint2_DINT，最大 X 值和最大 Y 值相互独立确定，所以这些值不一定来自于同一个点。

6.2.3.9.1.1 MaxElementElementwise (LONG)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    LONG&           nMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMax	LONG&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.2 MaxElementElementwise (SHORT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    SHORT&          nMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMax	SHORT&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.3 MaxElementElementwise (double)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    double&         fMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
fMax	double&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.4 MaxElementElementwise (float)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    float&          fMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
fMax	float&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.5 MaxElementElementwise (char)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    char&           nMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMax	char&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.6 MaxElementElementwise (TcVnPoint2_DINT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint2_DINT& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnPoint2_DINT [▶ 1640]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.7 MaxElementElementwise (TcVnPoint2_LREAL)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnPoint2_LREAL& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnPoint2_LREAL [▶ 1640]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.8 MaxElementElementwise (TcVnPoint2_REAL)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint2_REAL& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnPoint2_REAL [▶ 1640]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.9 MaxElementElementwise (TcVnPoint3_LREAL)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnPoint3_LREAL& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnPoint3_LREAL [▶ 1640]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.10 MaxElementElementwise (TcVnPoint3_REAL)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint3_REAL& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnPoint3_REAL [▶ 1640]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.11 MaxElementElementwise (TcVnVector2_DINT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_DINT& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector2_DINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.12 MaxElementElementwise (TcVnVector2_INT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector2_INT& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector2_INT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.13 MaxElementElementwise (TcVnVector2_REAL)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_REAL& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector2_REAL [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.14 MaxElementElementwise (TcVnVector2_SINT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_SINT& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector2_SINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.15 MaxElementElementwise (TcVnVector2_UINT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_UINT& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector2_UINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.16 MaxElementElementwise (TcVnVector2_USINT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector2_USINT& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector2_USINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.17 MaxElementElementwise (TcVnVector3_INT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector3_INT& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector3_INT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.18 MaxElementElementwise (TcVnVector3_REAL)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_REAL& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector3_REAL [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.19 MaxElementElementwise (TcVnVector3_SINT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_SINT& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector3_SINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.20 MaxElementElementwise (TcVnVector3_UINT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_UINT& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector3_UINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.21 MaxElementElementwise (TcVnVector3_USINT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector3_USINT& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector3_USINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.22 MaxElementElementwise (TcVnVector4_DINT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_DINT& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector4_DINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.23 MaxElementElementwise (TcVnVector4_INT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector4_INT& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector4_INT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.24 MaxElementElementwise (TcVnVector4_LREAL)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector4_LREAL& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector4_LREAL [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.25 MaxElementElementwise (TcVnVector4_SINT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_SINT& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector4_SINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.26 MaxElementElementwise (TcVnVector4_UINT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_UINT& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector4_UINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.27 MaxElementElementwise (TcVnVector4_USINT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector4_USINT& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector4_USINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.28 MaxElementElementwise (ULONG)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    ULONG&          nMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMax	ULONG&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.29 MaxElementElementwise (USHORT)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    USHORT&         nMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMax	USHORT&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.30 MaxElementElementwise (ULONGLONG)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    ULONGLONG&      nMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMax	ULONGLONG&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.31 MaxElementElementwise (unsigned char)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    unsigned char&  nMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMax	unsigned char&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.1.32 MaxElementElementwise (TcVnVector4_REAL)

获取容器元素的逐元素最大值。

语法

定义：

```
HRESULT MaxElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_REAL& aMax
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMax	TcVnVector4_REAL [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2 MedianElementElementwise

本章包含根据每个元素的数据类型确定中位数的函数。这个函数适用于包括基本元素的容器。

例如，对于 TcVnPoint2_DINT 来说，中位 X 值和中位 Y 值相互独立确定，所以这些值不一定来自于同一个点。

6.2.3.9.2.1 MedianElementElementwise (LONG)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    LONG&           nMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMedian	LONG&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.2 MedianElementElementwise (SHORT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    SHORT&          nMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMedian	SHORT&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.3 MedianElementElementwise (double)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    double&         fMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
fMedian	double&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.4 MedianElementElementwise (float)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    float&          fMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
fMedian	float&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.5 MedianElementElementwise (char)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    char&           nMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMedian	char&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.6 MedianElementElementwise (TcVnPoint2_DINT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint2_DINT& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnPoint2_DINT [▶ 1640]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.7 MedianElementElementwise (TcVnPoint2_LREAL)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnPoint2_LREAL& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnPoint2_LREAL [▶ 1640]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.8 MedianElementElementwise (TcVnPoint2_REAL)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint2_REAL& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnPoint2_REAL [▶ 1640]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.9 MedianElementElementwise (TcVnPoint3_LREAL)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnPoint3_LREAL& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnPoint3_LREAL [▶ 1640]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.10 MedianElementElementwise (TcVnPoint3_REAL)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint3_REAL& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnPoint3_REAL [▶ 1640]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.11 MedianElementElementwise (TcVnVector2_DINT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_DINT& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector2_DINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.12 MedianElementElementwise (TcVnVector2_INT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector2_INT& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector2_INT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.13 MedianElementElementwise (TcVnVector2_REAL)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_REAL& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector2_REAL [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.14 MedianElementElementwise (TcVnVector2_SINT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_SINT& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector2_SINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.15 MedianElementElementwise (TcVnVector2_UINT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_UINT& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector2_UINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.16 MedianElementElementwise (TcVnVector2_USINT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector2_USINT& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector2_USINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.17 MedianElementElementwise (TcVnVector3_INT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector3_INT& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector3_INT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.18 MedianElementElementwise (TcVnVector3_REAL)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_REAL& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector3_REAL [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.19 MedianElementElementwise (TcVnVector3_SINT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_SINT& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector3_SINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.20 MedianElementElementwise (TcVnVector3_UINT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_UINT& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector3_UINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.21 MedianElementElementwise (TcVnVector3_USINT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector3_USINT& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector3_USINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.22 MedianElementElementwise (TcVnVector4_DINT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_DINT& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector4_DINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.23 MedianElementElementwise (TcVnVector4_INT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector4_INT& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector4_INT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.24 MedianElementElementwise (TcVnVector4_LREAL)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector4_LREAL& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector4_LREAL [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.25 MedianElementElementwise (TcVnVector4_SINT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_SINT& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector4_SINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.26 MedianElementElementwise (TcVnVector4_UINT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_UINT& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector4_UINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.27 MedianElementElementwise (TcVnVector4_USINT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector4_USINT& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector4_USINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.28 MedianElementElementwise (ULONG)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer* ipContainer,
    ULONG&           nMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMedian	ULONG&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.29 MedianElementElementwise (USHORT)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    USHORT&         nMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMedian	USHORT&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.30 MedianElementElementwise (ULONGLONG)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    ULONGLONG&      nMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMedian	ULONGLONG&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.31 MedianElementElementwise (unsigned char)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    unsigned char&  nMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMedian	unsigned char&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.2.32 MedianElementElementwise (TcVnVector4_REAL)

获取容器元素的逐元素中值。

语法

定义：

```
HRESULT MedianElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_REAL& aMedian
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMedian	TcVnVector4_REAL [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3 MinElementElementwise

本章包含根据每个元素的数据类型来确定最小值的函数。这个函数适用于包括基本元素的容器。

例如，对于 TcVnPoint2_DINT，最小 X 值和最小 Y 值相互独立确定，所以这些值不一定来自于同一个点。

6.2.3.9.3.1 MinElementElementwise (LONG)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    LONG&           nMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMin	LONG&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.2 MinElementElementwise (SHORT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    SHORT&          nMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMin	SHORT&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.3 MinElementElementwise (double)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    double&         fMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
fMin	double&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.4 MinElementElementwise (float)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    float&          fMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
fMin	float&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.5 MinElementElementwise (char)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    char&           nMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMin	char&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.6 MinElementElementwise (TcVnPoint2_DINT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint2_DINT& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnPoint2_DINT [▶ 1640]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.7 MinElementElementwise (TcVnPoint2_LREAL)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnPoint2_LREAL& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnPoint2_LREAL [▶ 1640]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.8 MinElementElementwise (TcVnPoint2_REAL)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint2_REAL& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnPoint2_REAL [▶ 1640]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.9 MinElementElementwise (TcVnPoint3_LREAL)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnPoint3_LREAL& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnPoint3_LREAL [▶ 1640]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.10 MinElementElementwise (TcVnPoint3_REAL)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnPoint3_REAL& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnPoint3_REAL [▶ 1640]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.11 MinElementElementwise (TcVnVector2_DINT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_DINT& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector2_DINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.12 MinElementElementwise (TcVnVector2_INT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector2_INT& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector2_INT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.13 MinElementElementwise (TcVnVector2_REAL)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_REAL& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector2_REAL [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.14 MinElementElementwise (TcVnVector2_SINT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_SINT& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector2_SINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.15 MinElementElementwise (TcVnVector2_UINT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector2_UINT& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector2_UINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.16 MinElementElementwise (TcVnVector2_USINT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector2_USINT& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector2_USINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.17 MinElementElementwise (TcVnVector3_INT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector3_INT& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector3_INT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.18 MinElementElementwise (TcVnVector3_REAL)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_REAL& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector3_REAL [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.19 MinElementElementwise (TcVnVector3_SINT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_SINT& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector3_SINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.20 MinElementElementwise (TcVnVector3_UINT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector3_UINT& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector3_UINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.21 MinElementElementwise (TcVnVector3_USINT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector3_USINT& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector3_USINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.22 MinElementElementwise (TcVnVector4_DINT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_DINT& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector4_DINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.23 MinElementElementwise (TcVnVector4_INT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContainer,
    TcVnVector4_INT& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector4_INT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.24 MinElementElementwise (TcVnVector4_LREAL)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector4_LREAL& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector4_LREAL [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.25 MinElementElementwise (TcVnVector4_SINT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_SINT& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector4_SINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.26 MinElementElementwise (TcVnVector4_UINT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_UINT& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector4_UINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.27 MinElementElementwise (TcVnVector4_USINT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT            hrPrev,
    ITcVnContainer*    ipContainer,
    TcVnVector4_USINT& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector4_USINT [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.28 MinElementElementwise (ULONG)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    ULONG&          nMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMin	ULONG&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.29 MinElementElementwise (USHORT)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    USHORT&         nMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMin	USHORT&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.30 MinElementElementwise (ULONGLONG)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    ULONGLONG&      nMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMin	ULONGLONG&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.31 MinElementElementwise (unsigned char)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT         hrPrev,
    ITcVnContainer* ipContainer,
    unsigned char&  nMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
nMin	unsigned char&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.3.32 MinElementElementwise (TcVnVector4_REAL)

获取容器元素的逐元素最小值。

语法

定义：

```
HRESULT MinElementElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContainer,
    TcVnVector4_REAL& aMin
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer	ITcVnContainer* [▶ 1827]	源容器
aMin	TcVnVector4_REAL [▶ 1641]&	返回请求的元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.4 ContainerAverage

计算一个容器中的元素平均值。

语法

定义：

```
HRESULT ContainerAverage(
    HRESULT          hrPrev,
    ITcVnContainer* ipSrcContainer,
    double&          fAverage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	源容器（支持算术、点和向量元素类型。对于点和向量，使用欧几里得范数）。
fAverage	double&	返回计算出的平均值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.5 ContainerAverageElementwise (2)

计算一个容器中二维元素的逐元素平均值。

语法

定义：

```
HRESULT ContainerAverageElementwise(
    HRESULT            hrPrev,
    ITcVnContainer*    ipSrcContainer,
    TcVnVector2_LREAL& aAverage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	带有二维点或向量的源容器
aAverage	TcVnVector2_LREAL [▶ 1641]&	返回计算出的平均值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.6 ContainerAverageElementwise (3)

计算一个容器中三维元素的逐元素平均值。

语法

定义：

```
HRESULT ContainerAverageElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipSrcContainer,
    TcVnVector3_LREAL& aAverage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	带有三维点或向量的源容器
aAverage	TcVnVector3_LREAL [▶ 1641]&	返回计算出的平均值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.7 ContainerAverageElementwise (4)

计算一个容器中四维元素的逐元素平均值。

语法

定义：

```
HRESULT ContainerAverageElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipSrcContainer,
    TcVnVector4_LREAL& aAverage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	带有四维向量的源容器
aAverage	TcVnVector4_LREAL [▶ 1641]&	返回计算出的平均值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.8 ContainerAverageVariance

计算一个容器中元素的平均值和方差。

语法

定义：

```
HRESULT ContainerAverageVariance(
    HRESULT         hrPrev,
    ITcVnContainer* ipSrcContainer,
    double&         fAverage,
    double&         fVariance
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	源容器（支持算术、点和向量元素类型。对于点和向量，使用欧几里得范数）。
fAverage	double&	返回计算出的平均值
fVariance	double&	返回计算出的方差值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.9 ContainerAverageVarianceElementwise (2)

计算一个容器中二维元素的逐元素平均值和方差。

语法

定义：

```
HRESULT ContainerAverageVarianceElementwise(
    HRESULT            hrPrev,
    ITcVnContainer*    ipSrcContainer,
    TcVnVector2_LREAL& aAverage,
    TcVnVector2_LREAL& aVariance
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	带有二维点或向量的源容器
aAverage	TcVnVector2_LREAL [▶ 1641]&	返回计算出的平均值
aVariance	TcVnVector2_LREAL [▶ 1641]&	返回计算出的方差值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.10 ContainerAverageVarianceElementwise (3)

计算一个容器中三维元素的逐元素平均值和方差。

语法

定义：

```
HRESULT ContainerAverageVarianceElementwise(
    HRESULT            hrPrev,
    ITcVnContainer*    ipSrcContainer,
```

```
    TcVnVector3_LREAL& aAverage,
    TcVnVector3_LREAL& aVariance
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	带有三维点或向量的源容器
aAverage	TcVnVector3_LREAL [▶ 1641]&	返回计算出的平均值
aVariance	TcVnVector3_LREAL [▶ 1641]&	返回计算出的方差值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.11 ContainerAverageVarianceElementwise (4)

计算一个容器中四维元素的逐元素平均值和方差。

语法

定义：

```
HRESULT ContainerAverageVarianceElementwise(
    HRESULT           hrPrev,
    ITcVnContainer*   ipSrcContainer,
    TcVnVector4_LREAL& aAverage,
    TcVnVector4_LREAL& aVariance
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	带有四维向量的源容器
aAverage	TcVnVector4_LREAL [▶ 1641]&	返回计算出的平均值
aVariance	TcVnVector4_LREAL [▶ 1641]&	返回计算出的方差值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.12 MaxElement

获取一个容器的最大元素。

语法

定义：

```
HRESULT MaxElement(
    HRESULT                       hrPrev,
    ITcVnContainer*               ipSrcContainer,
    ITcVnBidirectionalIterator*&  ipDestIterator,
    ULONGLONG&                    nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	源容器（支持算术、点和向量元素类型。对于点和向量，使用欧几里得范数）。
ipDestIterator	ITcVnBidirectionalIterator* [▶ 1818]&	返回请求元素的迭代器
nIndex	ULONGLONG&	返回请求的元素在容器中的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.13 MedianElement

获取一个容器的元素中值。

语法

定义：

```
HRESULT MedianElement(
    HRESULT                       hrPrev,
    ITcVnContainer*               ipSrcContainer,
    ITcVnBidirectionalIterator*&  ipDestIterator,
    ULONGLONG&                    nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	源容器（支持算术、点和向量元素类型。对于点和向量，使用欧几里得范数）。
ipDestIterator	ITcVnBidirectionalIterator* [▶ 1818]&	返回请求元素的迭代器
nIndex	ULONGLONG&	返回请求的元素在容器中的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.9.14 MinElement

获取一个容器的最小元素。

语法

定义：

```
HRESULT MinElement(
    HRESULT                     hrPrev,
    ITcVnContainer*             ipSrcContainer,
    ITcVnBidirectionalIterator*& ipDestIterator,
    ULONGLONG&                  nIndex
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContainer	ITcVnContainer* [▶ 1827]	源容器（支持算术、点和向量元素类型。对于点和向量，使用欧几里得范数）。
ipDestIterator	ITcVnBidirectionalIterator* [▶ 1818]&	返回请求元素的迭代器
nIndex	ULONGLONG&	返回请求的元素在容器中的位置

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10 轮廓分析

该组包含用于分析轮廓的函数：

函数

形状近似

- ApproximatePolygon [▶ 2283]
- ConvexHullPoints [▶ 2293]
- EnclosingCircle [▶ 2296]
- EnclosingRectangle [▶ 2297]
- EnclosingTriangle [▶ 2298]
- FitEllipse [▶ 2299]
- FitLine [▶ 2299]
- UprightBoundingRectangle [▶ 2304]

点

- CheckIfPointIsInsideContour [▶ 2283]
- ContourExtremePoint [▶ 2289]
- ConvexityDefects [▶ 2294]

几何特征

- ContourArea [▶ 2284]
- ContourCenterOfMass [▶ 2285]
- ContourCircularity [▶ 2286]
- ContourConvexity [▶ 2286]
- ContourEccentricity [▶ 2287]
- ContourElongation [▶ 2288]
- ContourInertiaRatio [▶ 2289]
- ContourMoments [▶ 2290]
- ContourOrientation [▶ 2291]
- ContourPerimeter [▶ 2292]
- ContourRoundness [▶ 2292]
- FourierDescriptors [▶ 2300]

匹配

- MatchContours [▶ 2301]
- MatchContours1vsN [▶ 2303]

用户定义运算

- CustomElementWiseContainerOperation_ITcVnContainer [▶ 2295]
- CustomElementWiseContainerOperation_ITcVnForwardIterator [▶ 2296]

6.2.3.10.1 ApproximatePolygon

将轮廓近似为简化多边形（使用道格拉斯-普克（Douglas-Peucker）算法）。

语法

定义：

```
HRESULT ApproximatePolygon(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContour,
    ITcVnContainer*& ipDestContour,
    double           fMaxDist,
    bool             bClosed
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcContour	ITcVnContainer* [▶ 1827]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
ipDestContour	ITcVnContainer* [▶ 1827]&	返回一个以近似多边形点填充的容器（与 ipSrcContour 的类型 ID 相同；重复使用非零接口指针。）
fMaxDist	double	原始轮廓与近似值之间的最大距离
bClosed	BOOL	指定轮廓是否封闭（首尾点相连）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.2 CheckIfPointIsInsideContour

检查点是否位于轮廓内（可选返回与轮廓之间的距离）。

语法

定义：

```
HRESULT CheckIfPointIsInsideContour(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContour,
    TcVnPoint2_REAL& aPoint,
    double&          fDist,
    bool             bMeasureDistance
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContour	ITcVnContainer* [▶ 1827]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL）
aPoint	TcVnPoint2_REAL [▶ 1640]&	待检查的点位置
fDist	double&	返回结果（> 0：点在轮廓内；0：点在轮廓上；< 0：点在轮廓外）
bMeasureDistance	BOOL	如果为 true，fDist 将返回到最近轮廓边缘的距离。否则，fDist 只返回 -1、0 或 +1（如果不需要距离，建议设置为 false，这样速度更快）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.3 ContourArea

使用格林公式估算轮廓面积。

语法

定义：

```
HRESULT ContourArea(
    HRESULT          hrPrev,
    ITcVnContainer* ipContour,
    double&          fArea
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	描述
ipContour	ITcVnContainer* [▶ 1827]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
fArea	double&	返回估算的轮廓面积（实际轮廓面积可能因轮廓形状而异）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.4 ContourCenterOfMass

计算轮廓的质心。

语法

定义：

```
HRESULT ContourCenterOfMass(
    HRESULT           hrPrev,
    ITcVnContainer*   ipContour,
    TcVnPoint2_LREAL& aCenterOfMass
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContour	ITcVnContainer* [▶ 1827]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
aCenterOfMass	TcVnPoint2_LREAL [▶ 1640]&	返回轮廓的质心

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.5 ContourCircularity

计算轮廓的圆度（1.0：理想圆，0.0：直线）。

语法

定义：

```
HRESULT ContourCircularity(
    HRESULT                          hrPrev,
    ITcVnContainer*                  ipContour,
    double&                          fCircularity,
    ETcVnCircularityCalculationMethod eMethod = CCM_DEFAULT
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContour	ITcVnContainer* [▶ 1827]		单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
fCircularity	double&		返回轮廓的圆度 [0..1]
eMethod	ETcVnCircularityCalculationMethod [▶ 1651]	CCM_DEFAULT	用于计算圆度的方法

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.6 ContourConvexity

计算轮廓的凸度，即轮廓面积与凸包面积之比（1.0：凸形（即矩形、圆形...等），0.0：零面积（即直线或空轮廓））。

语法

定义：

```
HRESULT ContourConvexity(
    HRESULT          hrPrev,
    ITcVnContainer* ipContour,
    double&          fConvexity
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContour	ITcVnContainer* [▶ 1827]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
fConvexity	double&	返回轮廓的凸度 [0..1]

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.7 ContourEccentricity

计算轮廓的偏心率（0.0：圆形，1.0：直线）。

语法

定义：

```
HRESULT ContourEccentricity(
    HRESULT          hrPrev,
    ITcVnContainer* ipContour,
    double&          fEccentricity
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContour	ITcVnContainer* [▶ 1827]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或

名称	类型	描述
		ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
fEccentricity	double&	返回轮廓的偏心率 [0..1]

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.8 ContourElongation

计算轮廓的伸长系数。

语法

定义：

```
HRESULT ContourElongation(
    HRESULT         hrPrev,
    ITcVnContainer* ipContour,
    double&         fElongation
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContour	ITcVnContainer* [▶ 1827]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
fElongation	double&	返回轮廓的伸长率

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.9 ContourExtremePoint

查找轮廓极值点。

语法

定义：

```
HRESULT ContourExtremePoint(
    HRESULT                    hrPrev,
    ITcVnContainer*            ipContour,
    ETcVnExtremePointDirection eDirection,
    TcVnPoint2_LREAL&          aExtremePoint
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContour	ITcVnContainer* [▶ 1827]	包含二维点的源容器（TcVnPoint2_DINT 或 TcVnPoint2_REAL 或 TcVnPoint2_LREAL）
eDirection	ETcVnExtremePointDirection [▶ 1675]	选择极值点的搜索方向
aExtremePoint	TcVnPoint2_LREAL [▶ 1640]&	根据 eDirection 返回轮廓的极值点

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.10 ContourInertiaRatio

计算轮廓的惯性比，该比值反映轮廓/形状的伸长程度（即圆：1.0，线：0.0）。

语法

定义：

```
HRESULT ContourInertiaRatio(
    HRESULT         hrPrev,
    ITcVnContainer* ipContour,
    double&         fInertiaRatio
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContour	ITcVnContainer* [▶ 1827]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
fInertiaRatio	double&	返回轮廓的惯性比 [0..1]

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.11 ContourMoments

计算轮廓的空间矩、中心矩和归一化中心矩，最高可达三阶。

语法

定义：

```
HRESULT ContourMoments(
    HRESULT         hrPrev,
    ITcVnContainer* ipContour,
    TcVnMoments&    stMoments
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContour	ITcVnContainer* [▶ 1827]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
stMoments	TcVnMoments [▶ 1707]&	返回包含矩的结构体

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.12 ContourOrientation

计算轮廓或点集的方向。

语法

定义：

```
HRESULT ContourOrientation(
    HRESULT                 hrPrev,
    ITcVnContainer*         ipContour,
    TcVnRotatedRectangle&   stOrientation,
    ETcVnOrientationMethod eMethod = OM_PCA
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContour	ITcVnContainer* [▶ 1827]		源点集（ContainerType_Vector_TcVnPoint2_DINT、ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL。此容器的元素为各个点）。
stOrientation	TcVnRotatedRectangle [▶ 1718]&		旋转后的矩形，包含中心点、轴长和顺时针方向的轮廓旋转角（取决于所使用的计算方法）。
eMethod	ETcVnOrientationMethod [▶ 1686]	OM_PCA	计算方向的方法。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.13 ContourPerimeter

计算轮廓的周长（如果轮廓不是封闭的，则计算曲线长度）。

语法

定义：

```
HRESULT ContourPerimeter(
    HRESULT         hrPrev,
    ITcVnContainer* ipContour,
    double&         fPerimeter,
    bool            bClosed
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContour	ITcVnContainer* [▶ 1827]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
fPerimeter	double&	返回轮廓的周长
bClosed	BOOL	指定轮廓是否封闭（首尾点相连）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.14 ContourRoundness

计算轮廓的圆度（周长^2/面积）。

语法

定义：

```
HRESULT ContourRoundness(
    HRESULT          hrPrev,
    ITcVnContainer* ipContour,
    double&          fRoundness
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContour	ITcVnContainer* [▶ 1827]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓的点。）
fRoundness	double&	返回轮廓的圆度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.15 ConvexHullPoints

确定点集的凸包。

语法

定义：

```
HRESULT ConvexHullPoints(
    HRESULT          hrPrev,
    ITcVnContainer*  ipPointSet,
    ITcVnContainer*& ipConvexHull,
    bool             bClockwise = false
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipPointSet	ITcVnContainer* [▶ 1827]		源点集（ContainerType_Vector_TcVnPoint2_DINT 或

名称	类型	默认值	描述
			ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为各个点。）
ipConvexHull	ITcVnContainer* [▶ 1827]&		返回一个以已排序凸包点填充的容器（与 ipPointSet 的类型 ID 相同；重复使用非零接口指针）。
bClockwise	BOOL	false	假定为笛卡尔（Cartesian）坐标，选择凸包点的排序方向。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.16 ConvexityDefects

查找点集的凸度缺陷。

语法

定义：

```
HRESULT ConvexityDefects(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContour,
    ITcVnContainer*& ipConvexityDefects
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContour	ITcVnContainer* [▶ 1827]	源轮廓（ContainerType_Vector_TcVnPoint2_DINT；此容器的元素为各个点。）
ipConvexityDefects	ITcVnContainer* [▶ 1827]&	返回一个以凸度缺陷填充的容器（ContainerType_Vector_TcVnVector4_DINT，包含 [startIndex、endIndex、furthestPointIndex、fixedPointDistance（8 个尾数位，即除以 256 得到以像素为单位的距离）]；重复使用非零接口指针）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.17 CustomElementWiseContainerOperation (ITcVnContainer*)

在一组最多 3 个容器上执行自定义逐元素运算。

语法

定义：

```
HRESULT CustomElementWiseContainerOperation(
    HRESULT                                     hrPrev,
    ITcVnContainer*                             ipContainer1,
    ITcVnContainer*                             ipContainer2,
    ITcVnContainer*                             ipContainer3,
    ITcVnCustomContainerOperation_ITcVnContainer* ipOperationFB
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer1	ITcVnContainer* [▶ 1827]	容器 1
ipContainer2	ITcVnContainer* [▶ 1827]	容器 2
ipContainer3	ITcVnContainer* [▶ 1827]	容器 3
ipOperationFB	ITcVnCustomContainerOperation_ITcVnContainer* [▶ 1719]	对容器元素进行自定义运算

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.18 CustomElementWiseContainerOperation (ITcVnForwardIterator*)

在一组最多 3 个容器上执行自定义逐元素运算。

语法

定义：

```
HRESULT CustomElementWiseContainerOperation(
    HRESULT                                          hrPrev,
    ITcVnContainer*                                  ipContainer1,
    ITcVnContainer*                                  ipContainer2,
    ITcVnContainer*                                  ipContainer3,
    ITcVnCustomContainerOperation_ITcVnForwardIterator* ipOperationFB
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContainer1	ITcVnContainer* [▶ 1827]	容器 1
ipContainer2	ITcVnContainer* [▶ 1827]	容器 2
ipContainer3	ITcVnContainer* [▶ 1827]	容器 3
ipOperationFB	ITcVnCustomContainerOperation_ITcVnForwardIterator* [▶ 1720]	对元素进行自定义运算

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.19 EnclosingCircle

搜索包围一组点的最小面积圆。

语法

定义：

```
HRESULT EnclosingCircle(
    HRESULT          hrPrev,
    ITcVnContainer*  ipPointSet,
    TcVnPoint2_REAL& aCenter,
    float&           fRadius
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipPointSet	ITcVnContainer* [▶ 1827]	源点集（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为各个点。）
aCenter	TcVnPoint2_REAL [▶ 1640]&	返回圆心
fRadius	float&	返回圆的半径

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.20 EnclosingRectangle

搜索包围一组点的最小面积矩形。

语法

定义：

```
HRESULT EnclosingRectangle(
    HRESULT               hrPrev,
    ITcVnContainer*       ipPointSet,
    TcVnRotatedRectangle& stRectangle
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipPointSet	ITcVnContainer* [▶ 1827]	源点集（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为各个点。）
stRectangle	TcVnRotatedRectangle [▶ 1718]&	返回测定的矩形

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.21 EnclosingTriangle

搜索包围一组点的最小面积三角形。

语法

定义：

```
HRESULT EnclosingTriangle(
    HRESULT                 hrPrev,
    ITcVnContainer*         ipPointSet,
    TcVnArray3_Point2_REAL& aTriangleVertices,
    double&                 fArea
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipPointSet	ITcVnContainer* [▶ 1827]	源点集（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为各个点。）
aTriangleVertices	TcVnArray3_Point2_REAL [▶ 1641]&	返回 3 个三角形顶点
fArea	double&	返回三角形面积

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.22 FitEllipse

在点集中拟合一个椭圆。

语法

定义：

```
HRESULT FitEllipse(
    HRESULT               hrPrev,
    ITcVnContainer*       ipPointSet,
    TcVnRotatedRectangle& stEllipse
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipPointSet	ITcVnContainer* [▶ 1827]	源点集（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；至少需要 5 个合理的点！此容器的元素为各个点）。
stEllipse	TcVnRotatedRectangle [▶ 1718]&	产生的椭圆，由旋转矩形描述。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.23 FitLine

将直线拟合到点集中

语法

定义：

```
HRESULT FitLine(
    HRESULT             hrPrev,
    ITcVnContainer*     ipPointSet,
    TcVnVector4_LREAL&  aFitLine,
    ETcVnDistanceType   eDistanceType = DT_L2,
    double              fParam = 0,
    double              fEpsRadius = 0.001,
    double              fEpsAngle = 0.001
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipPointSet	ITcVnContainer* [▶ 1827]		源点集（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为各个点。）
aFitLine	TcVnVector4_LREAL [▶ 1641]&		产生的直线。第一个和第二个元素描述的是该直线共线向量的 x 和 y 分量。第三个和第四个元素描述该直线上一个点的 x 和 y 分量。
eDistanceType	ETcVnDistanceType [▶ 1671]	DT_L2	距离计算方法（支持：L1、L2、L12、FAIR、WELSCH、HUBER）
fParam	double	0	某个 eDistanceType 的数值参数（c）（应 >= 0）。如果为 0，则在内部选择一个最佳值。
fEpsRadius	double	0.001	半径精度（直线与坐标原点的距离，应大于 0）。数值越小，精度越高。
fEpsAngle	double	0.001	角度精度（应大于 0）。数值越小，精度越高。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.24 FourierDescriptors

计算封闭轮廓的傅立叶描述符。

语法

定义：

```
HRESULT FourierDescriptors(
    HRESULT          hrPrev,
    ITcVnContainer*  ipContour,
```

```
    ITcVnContainer*& ipDescriptors,
    LONG             nDescriptors
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContour	ITcVnContainer* [▶ 1827]	封闭轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL。强烈建议提供完整的轮廓，即轮廓检测算法使用 TCVN_CAM_NONE）。
ipDescriptors	ITcVnContainer* [▶ 1827]&	返回 ipContour 的傅立叶描述符（Re0、Im0、Re1、Im1、...；如果 ipContour 属于 ContainerType_Vector_LREAL 类型，则返回 ContainerType_Vector_TcVnPoint2_LREAL，否则返回 ContainerType_Vector_REAL。）
nDescriptors	LONG	指定返回的描述符数量（设为 -1 则返回所有算得的描述符，即至少与轮廓点的数量一样多）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.25 MatchContours

使用 Hu 不变矩（也可以选择其它技术）比较轮廓。如果每个容器中有多个轮廓，则会找出最匹配的轮廓，并返回所有匹配轮廓的平均相似度。

语法

定义：

```
HRESULT MatchContours(
    HRESULT                          hrPrev,
    ITcVnContainer*                  ipContour1,
    ITcVnContainer*                  ipContour2,
    ETcVnContoursMatchComparisonMethod eComparisonMethod,
    double&                          fDissimilarity,
    double                           fAreaFactor = 0,
```

```
    double                              fAbsPositionFactor = 0,
    double                              fRelPositionFactor = 0,
    double                              fNumDiffFactor = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContour1	ITcVnContainer* [▶ 1827]		第一个轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；该容器的元素为轮廓的点）或多条轮廓线的集合（ContainerType_Vector_Vector_TcVnPoint2_DINT）
ipContour2	ITcVnContainer* [▶ 1827]		第二个轮廓（与 ipContour1 类型相同）
eComparisonMethod	ETcVnContoursMatchComparisonMethod [▶ 1667]		用于比较轮廓 Hu 不变矩的方法
fDissimilarity	double&		根据所选的比较方法，返回轮廓的相异度
fAreaFactor	double	0	如果 > 0，则轮廓之间的相对面积差将按此系数缩放，并与算得的相异度相加
fAbsPositionFactor	double	0	如果 > 0，则轮廓之间的绝对位置差（即几何轮廓中心的坐标）按此系数缩放，并与算得的相异度相加
fRelPositionFactor	double	0	如果 > 0，则轮廓之间的相对位置差将按此系数缩放，并与算得的相异度相加（仅在匹配多个轮廓时使用）
fNumDiffFactor	double	0	如果 > 0，则两个容器中的轮廓数量之差将按此系数缩放，并与算得的相异度相加（仅在匹配多个轮廓时使用）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.26 MatchContours1vsN

使用 Hu 不变矩（也可以选择其它技术）将参考轮廓与其它多个轮廓进行比较。返回最佳匹配的排序列表。

语法

定义：

```
HRESULT MatchContours1vsN(
    HRESULT                           hrPrev,
    ITcVnContainer*                   ipRefContour,
    ITcVnContainer*                   ipContours,
    ITcVnContainer*&                  ipMatchIndexes,
    ITcVnContainer*&                  ipDissimilarities,
    double                            fDissimilarityThreshold,
    ETcVnContoursMatchComparisonMethod eComparisonMethod,
    double                            fAreaFactor = 0,
    double                            fAbsPositionFactor = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipRefContour	ITcVnContainer* [▶ 1827]		参考轮廓（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为轮廓上的点）
ipContours	ITcVnContainer* [▶ 1827]		多个轮廓的集合（CTcVnContainer_Vector_Vector_TcVnPoint2_DINT）
ipMatchIndexes	ITcVnContainer* [▶ 1827]&		返回最佳匹配项的索引（CTcVnContainer_Vector_ULINT；已排序，第一个元素为最佳匹配项，即相异度最低的元素）
ipDissimilarities	ITcVnContainer* [▶ 1827]&		返回算得的最佳匹配项的相似度（CTcVnContainer_Vector_LREAL；可选，如果不需要，则设为 0；根据 ipMatchIndexes 排序）
fDissimilarityThreshold	double		忽略不相关的匹配，即相异度 > fDissimilarityThreshold

名称	类型	默认值	描述
eComparisonMethod	ETcVnContoursMatchComparisonMethod [▶ 1667]		用于比较轮廓 Hu 不变矩的方法
fAreaFactor	double	0	如果 > 0，则轮廓之间的相对面积差将按此系数缩放，并与算得的相异度相加
fAbsPositionFactor	double	0	如果 > 0，则轮廓之间的绝对位置差（即几何轮廓中心的坐标）按此系数缩放，并与算得的相异度相加

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.10.27 UprightBoundingRectangle

确定一组点的直立外接矩形。

语法

定义：

```
HRESULT UprightBoundingRectangle(
    HRESULT              hrPrev,
    ITcVnContainer*      ipPointSet,
    TcVnRectangle_UDINT& stRectangle
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipPointSet	ITcVnContainer* [▶ 1827]	源点集（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL；此容器的元素为各个点。）
stRectangle	TcVnRectangle_UDINT [▶ 1717]&	返回测定的矩形

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.11 控制函数

该组包含用于顺序控制的函数，例如用于处理 Watchdog（看门狗） [▶ 140]的函数。

函数

Watchdog (看门狗)

- StartAbsWatchdog [▶ 2305]
- StartRelWatchdog [▶ 2306]
- StopWatchdog [▶ 2306]

6.2.3.11.1 StartAbsWatchdog

给定一个绝对停止时间，启动协作看门狗。

语法

定义：

```
HRESULT StartAbsWatchdog(
    HRESULT                     hrPrev,
    LONG                        tStop,
    ETcWatchdogAccumulationType eWatchdogAccType = WD_ACC_TYPE_PRODUCT
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
tStop	LONG		停止时间，单位：us
eWatchdogAccType	ETcWatchdogAccumulationType [▶ 1696]	WD_ACC_TYPE_PRODUCT	用于合并该看门狗所包含的多个函数结果的累加方法

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.11.2 StartRelWatchdog

给定一个相对于当前时间的停止时间，启动协作看门狗。

语法

定义：

```
HRESULT StartRelWatchdog(
    HRESULT                     hrPrev,
    LONG                        tStop,
    ETcWatchdogAccumulationType eWatchdogAccType = WD_ACC_TYPE_PRODUCT
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
tStop	LONG		停止时间，单位：us
eWatchdogAccType	ETcWatchdogAccumulationType [▶ 1696]	WD_ACC_TYPE_PRODUCT	用于合并该看门狗所包含的多个函数结果的累加方法

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.11.3 StopWatchdog

停止看门狗并提供运行时信息。

语法

定义：

```
HRESULT StopWatchdog(
    HRESULT   hrStartWatchdog,
    ULONGLONG& nFunctionsMonitored,
    ULONG&    nFractionProcessed,
    LONG&     tRest
)
```

参数

名称	类型	描述
hrStartWatchdog	HRESULT [▶ 135]	HRESULT 表示用于启动看门狗的函数的结果（如果 SUCCEEDED（hrStartWatchdog）等于 false，则不执行任何操作。）
nFunctionsMonitored	ULONGLONG&	返回受监控的函数数量
nFractionProcessed	ULONG&	返回被监控函数的累计处理进度（以百分比为单位）
tRest	LONG&	返回以 us 为单位的剩余计算时间（可能为负值）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12 绘图

该组包含绘制和描述图像的函数。

处理 4 通道颜色

4 通道颜色直接接受到结果图像中。不会进行阿尔法混合或类似操作。

函数

箭头

- DrawArrow [▶ 2308]
- DrawArrow_TcVnVector4_DINT [▶ 2309]

圆圈和弧形

- DrawCircle [▶ 2310]
- DrawCircles [▶ 2311]
- DrawCircularArc [▶ 2312]
- FillCircle [▶ 2329]

组件

- DrawComponents [▶ 2313]

轮廓

- DrawContours [▶ 2314]
- FillContours [▶ 2330]

椭圆

- DrawEllipse [▶ 2315]
- FillEllipse [▶ 2331]

关键点和匹配

- DrawKeypoints [▶ 2316]
- DrawMatches [▶ 2321]

线

- DrawLine [▶ 2317]
- DrawLines [▶ 2320]
- DrawLine_TcVnVector4_DINT [▶ 2318]
- DrawLine_TcVnVector4_LREAL [▶ 2319]

定向

- DrawOrientation(Exp) [▶ 2322]

点

- DrawPoint [▶ 2324]
- DrawPoints [▶ 2325]

多边形

- DrawPolygon [▶ 2325]
- FillPolygon [▶ 2332]

矩形

- DrawRectangle [▶ 2326]
- DrawRectangle_TcVnRectangle_DINT [▶ 2327]
- DrawRectangle_TcVnRectangle_UDINT [▶ 2328]
- DrawRotatedRectangle [▶ 2329]
- FillRectangle [▶ 2332]
- FillRotatedRectangle [▶ 2333]

强度分布

- PlotIntensityProfile [▶ 2334]

文本

- PutLabel [▶ 2335]
- PutText [▶ 2336]

绘图和填充

对于许多形状，可以选择同时绘制形状轮廓和填充形状。为此提供了两个函数组（绘制……和填充……）。此外，将轮廓厚度 nThickness 设置为 -1 会填充形状。

6.2.3.12.1 DrawArrow

绘制一条箭头线。

语法

定义：

```
HRESULT DrawArrow(
    HRESULT             hrPrev,
    ULONG               nX1,
    ULONG               nY1,
    ULONG               nX2,
    ULONG               nY2,
    ITcVnImage*         ipDestImage,
    TcVnVector4_LREAL&  aColor,
    LONG                nThickness,
    bool                bDoubleHead,
    ETcVnLineType       eLineType = LT_8_CONNECTED,
    ULONG               nShift = 0,
    double              fTipLenght = 0.1
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nX1	ULONG		起点的 x 坐标
nY1	ULONG		起点的 y 坐标
nX2	ULONG		终点的 x 坐标
nY2	ULONG		终点的 y 坐标
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&		颜色
nThickness	LONG		线条粗细
bDoubleHead	BOOL		在直线两侧各画一个箭头。
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型
nShift	ULONG	0	坐标的小数位数（位移）
fTipLenght	double	0.1	箭尖长度占线段长度的百分比

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.2 DrawArrow (TcVnVector4_DINT)

绘制一条箭头线。

语法

定义：

```
HRESULT DrawArrow(
    HRESULT             hrPrev,
    TcVnVector4_DINT&   aLine,
    ITcVnImage*         ipDestImage,
    TcVnVector4_LREAL&  aColor,
    LONG                nThickness,
    bool                bDoubleHead,
    ETcVnLineType       eLineType = LT_8_CONNECTED,
    ULONG               nShift = 0,
    double              fTipLenght = 0.1
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aLine	TcVnVector4_DINT [▶ 1641]&		线段 [x1，y1，x2，y2] 的起点和终点
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&		颜色
nThickness	LONG		线条粗细
bDoubleHead	BOOL		在直线两侧各画一个箭头。
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型
nShift	ULONG	0	坐标的小数位数（位移）
fTipLenght	double	0.1	箭尖长度占线段长度的百分比

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.3 DrawCircle

绘制一个圆。

语法

定义：

```
HRESULT DrawCircle(
    HRESULT             hrPrev,
    ULONG               nCenterX,
    ULONG               nCenterY,
```

```
    ULONG               nRadius,
    ITcVnImage*         ipDestImage,
    TcVnVector4_LREAL&  aColor,
    LONG                nThickness,
    ETcVnLineType       eLineType = LT_8_CONNECTED
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nCenterX	ULONG		中心的 x 坐标
nCenterY	ULONG		中心的 y 坐标
nRadius	ULONG		半径
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&		颜色
nThickness	LONG		线条粗细（如果为负值，则填充圆形）
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.4 DrawCircles

绘制圆。

语法

定义：

```
HRESULT DrawCircles(
    HRESULT             hrPrev,
    ITcVnContainer*     ipCircles,
    LONG                nCircleIndex,
    ITcVnImage*         ipDestImage,
    TcVnVector4_LREAL&  aColor,
    LONG                nThickness,
    ETcVnLineType       eLineType = LT_8_CONNECTED
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipCircles	ITcVnContainer* [▶ 1827]		包含圆的容器（ContainerType_Vector_TcVnVector3_REAL；每个容器元素都包含圆心的 x 坐标 [0]、圆心的 y 坐标 [1] 和半径 [2]。）
nCircleIndex	LONG		要绘制的特定圆的索引（如果为负值，则绘制容器内的所有圆）
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&		颜色
nThickness	LONG		线条粗细（如果为负值，则填充圆形）
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.5 DrawCircularArc

绘制圆弧。

语法

定义：

```
HRESULT DrawCircularArc(
    HRESULT             hrPrev,
    TcVnCircularArc&    stCircularArc,
    ITcVnImage*         ipDestImage,
    TcVnVector4_LREAL&  aColor,
    LONG                nThickness,
    ETcVnLineType       eLineType = LT_8_CONNECTED
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stCircularArc	TcVnCircularArc [▶ 1703]&		圆弧定义
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&		颜色
nThickness	LONG		线条粗细（如果为负值，则填充圆弧）
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.6 DrawComponents

绘制连通分量函数的结果。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT DrawComponents(
    HRESULT            hrPrev,
    ITcVnImage*        ipLabelImage,
    LONG               nLabelID,
    ITcVnImage*&       ipDestImage,
    bool               bShowLabelID,
    TcVnVector4_LREAL& aColor = {-1, -1, -1, -1},
    LONG               nThickness = 1,
    double             fFontScale = 1
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	默认值	描述
ipLabelImage	ITcVnImage* [▶ 1863]		图像包含每个像素的标签（1 通道 DINT 或 UINT）。
nLabelID	LONG		要绘制的特定标签的值（如果为负值，则绘制图像中的所有分量）
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的图像。输出图像为 1 或 3 通道 USINT）。
bShowLabelID	BOOL		在目标图像的分量中心显示标签 ID
aColor	TcVnVector4_LREAL [▶ 1641]&	{-1, -1, -1, -1}	绘制分量的颜色（对于 {-1、-1、-1、-1}，将为每个分量随机选择一种颜色）
nThickness	LONG	1	文本线条粗细
fFontScale	double	1	文本缩放系数

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.7 DrawContours

绘制被解释为轮廓的单个点集或多个点集。

语法

定义：

```
HRESULT DrawContours(
    HRESULT             hrPrev,
    ITcVnContainer*     ipContours,
    LONG                nContourIndex,
    ITcVnImage*         ipDestImage,
    TcVnVector4_LREAL&  aColor,
    LONG                nThickness,
    ETcVnLineType       eLineType = LT_8_CONNECTED,
    ITcVnContainer*     ipHierarchy = nullptr,
    LONG                nMaxLevel = 2147483647,
    TcVnPoint&          aOffset = {0, 0}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipContours	ITcVnContainer* [▶ 1827]		单个轮廓（ContainerType_Vector_TcVnPoint2_DINT）或多个轮廓（ContainerType_Vector_Vector_TcVnPoint2_DINT）
nContourIndex	LONG		要绘制的特定轮廓的索引（如果为负值，则绘制容器内的所有轮廓）
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&		颜色
nThickness	LONG		线条粗细（如果为负值，则填充轮廓）
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型
ipHierarchy	ITcVnContainer* [▶ 1827]	nullptr	轮廓层次（ContainerType_Vector_TcVnVector4_DINT）
nMaxLevel	LONG	2147483647	要绘制的轮廓的最大等级
aOffset	TcVnPoint [▶ 1640]&	{0, 0}	每个轮廓点移动的偏移量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.8 DrawEllipse

绘制椭圆。

语法

定义：

```
HRESULT DrawEllipse(
    HRESULT                hrPrev,
    TcVnRotatedRectangle&  stEllipse,
    ITcVnImage*            ipDestImage,
    TcVnVector4_LREAL&     aColor,
```

```
    LONG                    nThickness,
    ETcVnLineType           eLineType = LT_8_CONNECTED
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stEllipse	TcVnRotatedRectangle [▶ 1718]&		要绘制的椭圆（旋转角，单位：度）
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&		颜色
nThickness	LONG		线条粗细（如果为负值，则填充椭圆）
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.9 DrawKeypoints

绘制关键点，以实现可视化。

语法

定义：

```
HRESULT DrawKeypoints(
    HRESULT                 hrPrev,
    ITcVnImage*             ipSrcImage,
    ITcVnContainer*         ipKeyPoints,
    ITcVnImage*&            ipDestImage,
    TcVnVector4_LREAL&      aColor = {-1, -1, -1, -1},
    ETcVnDrawMatchesFlags   eFlags = DMF_DEFAULT
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	默认值	描述
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT 类型元素）
ipKeyPoints	ITcVnContainer* [▶ 1827]		包含关键点的容器（ContainerType_Vector_TcVnKeyPoint）
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的彩色图像。）
aColor	TcVnVector4_LREAL [▶ 1641]&	{-1, -1, -1, -1}	绘制关键点的颜色（对于{-1、-1、-1、-1}，会为每个点随机选择一种颜色）
eFlags	ETcVnDrawMatchesFlags [▶ 1672]	DMF_DEFAULT	标志组合，支持过度绘制现有目标图像和/或绘制附加（富）关键点信息（尺寸和方向）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.10 DrawLine

绘制一条线。

语法

定义：

```
HRESULT DrawLine(
    HRESULT             hrPrev,
    ULONG               nX1,
    ULONG               nY1,
    ULONG               nX2,
    ULONG               nY2,
    ITcVnImage*         ipDestImage,
    TcVnVector4_LREAL&  aColor,
    LONG                nThickness,
    ETcVnLineType       eLineType = LT_8_CONNECTED,
    ULONG               nShift = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nX1	ULONG		起点的 x 坐标
nY1	ULONG		起点的 y 坐标

名称	类型	默认值	描述
nX2	ULONG		终点的 x 坐标
nY2	ULONG		终点的 y 坐标
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&		颜色
nThickness	LONG		线条粗细
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型
nShift	ULONG	0	坐标的小数位数（位移）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.11 DrawLine (TcVnVector4_DINT)

绘制一条线。

语法

定义：

```
HRESULT DrawLine(
    HRESULT            hrPrev,
    TcVnVector4_DINT&  aLine,
    ITcVnImage*        ipDestImage,
    TcVnVector4_LREAL& aColor,
    LONG               nThickness,
    ETcVnLineType      eLineType = LT_8_CONNECTED,
    ULONG              nShift = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aLine	TcVnVector4_DINT [▶ 1641]&		线段 [x1，y1，x2，y2] 的起点和终点
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&		颜色
nThickness	LONG		线条粗细
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型
nShift	ULONG	0	坐标的小数位数（位移）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.12 DrawLine (TcVnVector4_LREAL)

绘制一条线。

语法

定义：

```
HRESULT DrawLine(
    HRESULT             hrPrev,
    TcVnVector4_LREAL&  aLine,
    ITcVnImage*         ipDestImage,
    TcVnVector4_LREAL&  aColor,
    LONG                nThickness,
    ETcVnLineType       eLineType = LT_8_CONNECTED,
    ULONG               nShift = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aLine	TcVnVector4_LREAL [▶ 1641]&		第一个和第二个元素描述了方向向量的 x 和 y 分量。第三个和第四个元素描述位置向量的 x 和 y 分量。
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&		颜色
nThickness	LONG		线条粗细
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型
nShift	ULONG	0	坐标的小数位数（位移）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.13 DrawLines

绘制直线。

语法

定义：

```
HRESULT DrawLines(
    HRESULT             hrPrev,
    ITcVnContainer*     ipLines,
    LONG                nLineIndex,
    ITcVnImage*         ipDestImage,
    TcVnVector4_LREAL&  aColor,
    LONG                nThickness,
    ETcVnLineType       eLineType = LT_8_CONNECTED,
    ULONG               nShift = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipLines	ITcVnContainer* [▶ 1827]		带有线条描述的容器（ContainerType_Vector_TcVnVector2_REAL：与原点的距离 [0]（以像素为单位）和旋转角 [1]（以弧度为单位）。ContainerType_Vector_TcVnVector4_LREAL：第一个和第二个元素描述的是该直线共线向量的 x 和 y 分量。第三个和第四个元素描述该直线上一个点的 x 和 y 分量。ContainerType_Vector_TcVnVector4_DINT: 起点和终点 [x1，y1，x2，y2]）
nLineIndex	LONG		要绘制的特定直线的索引（如果为负值，则绘制容器内的所有直线）
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&		颜色
nThickness	LONG		线条粗细
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型
nShift	ULONG	0	坐标的小数位数（位移）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.14 DrawMatches

绘制关键点并进行匹配，以实现可视化。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT DrawMatches(
    HRESULT               hrPrev,
    ITcVnImage*           ipSrcImage1,
    ITcVnContainer*       ipKeyPoints1,
    ITcVnImage*           ipSrcImage2,
    ITcVnContainer*       ipKeyPoints2,
    ITcVnContainer*       ipMatches1To2,
    ITcVnImage*&          ipDestImage,
    TcVnVector4_LREAL&    aMatchColor = {-1, -1, -1, -1},
    TcVnVector4_LREAL&    aSingleColor = {-1, -1, -1, -1},
    ITcVnContainer*       ipMatchesMask = nullptr,
    ETcVnDrawMatchesFlags eFlags = DMF_DEFAULT
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage1	ITcVnImage* [▶ 1863]		源图像 1（USINT 类型元素）
ipKeyPoints1	ITcVnContainer* [▶ 1827]		包含关键点的容器 1（ContainerType_Vector_TcVnKeyPoint）
ipSrcImage2	ITcVnImage* [▶ 1863]		源图像 2（USINT 类型元素，通道数量必须与 ipSrcImage1 相同）
ipKeyPoints2	ITcVnContainer* [▶ 1827]		包含关键点的容器 2（ContainerType_Vector_TcVnKeyPoint）
ipMatches1To2	ITcVnContainer* [▶ 1827]		描述符匹配的容器（ContainerType_Vector_TcVnDMatch）
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的彩色图像。）

名称	类型	默认值	描述
aMatchColor	TcVnVector4_LREAL [▶ 1641]&	{-1, -1, -1, -1}	绘制匹配项的颜色（对于 {-1、-1、-1、-1}，会为每个点和直线随机选择一种颜色）
aSingleColor	TcVnVector4_LREAL [▶ 1641]&	{-1, -1, -1, -1}	绘制单个关键点的颜色（对于 {-1、-1、-1、-1}，会为每个点随机选择一种颜色）
ipMatchesMask	ITcVnContainer* [▶ 1827]	nullptr	用于选择要绘制的匹配项的掩膜（ContainerType_Vector_SINT；如果要绘制所有匹配项，则设为 0。）
eFlags	ETcVnDrawMatchesFlags [▶ 1672]	DMF_DEFAULT	标志组合，支持过度绘制现有目标图像和/或绘制附加（富）关键点信息（大小和方向）和/或跳过单个关键点

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.15 DrawOrientation

根据旋转矩形绘制一组点/轮廓的主轴。

语法

定义：

```
HRESULT DrawOrientation(
    HRESULT                 hrPrev,
    TcVnRotatedRectangle&   stOrientation,
    ITcVnImage*             ipDestImage,
    TcVnVector4_LREAL&      aColorMainAxis,
    LONG                    nThickness
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stOrientation	TcVnRotatedRectangle [▶ 1718]&	旋转矩形，包含中心点、轴长和顺时针方向的轮廓旋转角
ipDestImage	ITcVnImage* [▶ 1863]	目标图像

名称	类型	描述
aColorMainAxis	TcVnVector4_LREAL [▶ 1641]&	主轴的颜色
nThickness	LONG	线条粗细

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.16 DrawOrientation (Exp)

根据旋转矩形绘制一组点/轮廓的主轴、水平轴和角。

语法

定义：

```
HRESULT DrawOrientation(
    HRESULT                hrPrev,
    TcVnRotatedRectangle&  stOrientation,
    ITcVnImage*            ipDestImage,
    TcVnVector4_LREAL&     aColorMainAxis,
    LONG                   nThickness,
    TcVnVector4_LREAL&     aColorHorizontalAxis,
    double                 fFontScale
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stOrientation	TcVnRotatedRectangle [▶ 1718]&	旋转矩形，包含中心点、轴长和顺时针方向的轮廓旋转角
ipDestImage	ITcVnImage* [▶ 1863]	目标图像
aColorMainAxis	TcVnVector4_LREAL [▶ 1641]&	主轴的颜色
nThickness	LONG	线条粗细
aColorHorizontalAxis	TcVnVector4_LREAL [▶ 1641]&	水平轴的颜色
fFontScale	double	文本缩放系数

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.17 DrawPoint

绘制一个点。

语法

定义：

```
HRESULT DrawPoint(
    HRESULT             hrPrev,
    ULONG               nX,
    ULONG               nY,
    ITcVnImage*         ipDestImage,
    ETcVnDrawShape      eShape,
    TcVnVector4_LREAL&  aColor,
    ULONG               nSize = 3,
    LONG                nThickness = 1,
    ETcVnLineType       eLineType = LT_8_CONNECTED
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nX	ULONG		点的 x 坐标
nY	ULONG		点的 y 坐标
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
eShape	ETcVnDrawShape [▶ 1672]		点形状
aColor	TcVnVector4_LREAL [▶ 1641]&		颜色（对于 {-1、-1、-1、-1}，随机选择一种颜色）
nSize	ULONG	3	形状大小（半宽）
nThickness	LONG	1	线条粗细（如果为负值，则填充封闭的形状）
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.18 DrawPoints

绘制点集合。

语法

定义：

```
HRESULT DrawPoints(
    HRESULT             hrPrev,
    ITcVnContainer*     ipPoints,
    ITcVnImage*         ipDestImage,
    ETcVnDrawShape      eShape,
    TcVnVector4_LREAL&  aColor,
    ULONG               nSize = 3,
    LONG                nThickness = 1,
    ETcVnLineType       eLineType = LT_8_CONNECTED
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipPoints	ITcVnContainer* [▶ 1827]		包含 TcVnPoint2_DINT 或 TcVnPoint2_REAL 元素的容器
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
eShape	ETcVnDrawShape [▶ 1672]		点形状
aColor	TcVnVector4_LREAL [▶ 1641]&		颜色（对于 {-1、-1、-1、-1}，为每个点随机选择一种颜色）
nSize	ULONG	3	形状大小（半宽）
nThickness	LONG	1	线条粗细（如果为负值，则填充封闭的形状）
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.19 DrawPolygon

利用点的集合绘制多边形。

语法

定义：

```
HRESULT DrawPolygon(
    HRESULT             hrPrev,
    ITcVnContainer*     ipPoints,
    ITcVnImage*         ipDestImage,
    TcVnVector4_LREAL&  aColor,
    LONG                nThickness,
    bool                bIsClosed = true,
    ETcVnLineType       eLineType = LT_8_CONNECTED,
    ULONG               nShift = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipPoints	ITcVnContainer* [▶ 1827]		包含 TcVnPoint2_DINT、TcVnPoint2_REAL 或 TcVnPoint2_LREAL 元素的容器
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&		颜色
nThickness	LONG		线条粗细（如果为负值，则填充形状）
bIsClosed	BOOL	true	如果为 true，函数会从最后一个点向第一个点绘制一条线来封闭多边形。
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型
nShift	ULONG	0	坐标的小数位数（位移）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.20 DrawRectangle

绘制一个矩形。

语法

定义：

```
HRESULT DrawRectangle(
    HRESULT            hrPrev,
    ULONG              nTopLeftX,
    ULONG              nTopLeftY,
    ULONG              nBottomRightX,
    ULONG              nBottomRightY,
    ITcVnImage*        ipDestImage,
    TcVnVector4_LREAL& aColor,
    LONG               nThickness
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nTopLeftX	ULONG	左上角的 x 坐标
nTopLeftY	ULONG	左上角的 y 坐标
nBottomRightX	ULONG	右下角的 x 坐标
nBottomRightY	ULONG	右下角的 y 坐标
ipDestImage	ITcVnImage* [▶ 1863]	目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&	颜色
nThickness	LONG	线条粗细（如果为负值，则填充矩形）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.21 DrawRectangle (TcVnRectangle_DINT)

绘制一个矩形。

语法

定义：

```
HRESULT DrawRectangle(
    HRESULT             hrPrev,
    TcVnRectangle_DINT& stRectangle,
    ITcVnImage*         ipDestImage,
    TcVnVector4_LREAL&  aColor,
    LONG                nThickness
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	描述
stRectangle	TcVnRectangle_DINT [▶ 1717]&	要绘制的矩形
ipDestImage	ITcVnImage* [▶ 1863]	目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&	颜色
nThickness	LONG	线条粗细（如果为负值，则填充矩形）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.22 DrawRectangle (TcVnRectangle_UDINT)

绘制一个矩形。

语法

定义：

```
HRESULT DrawRectangle(
    HRESULT              hrPrev,
    TcVnRectangle_UDINT& stRectangle,
    ITcVnImage*          ipDestImage,
    TcVnVector4_LREAL&   aColor,
    LONG                 nThickness
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stRectangle	TcVnRectangle_UDINT [▶ 1717]&	要绘制的矩形
ipDestImage	ITcVnImage* [▶ 1863]	目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&	颜色
nThickness	LONG	线条粗细（如果为负值，则填充矩形）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.23 DrawRotatedRectangle

绘制一个矩形。

语法

定义：

```
HRESULT DrawRotatedRectangle(
    HRESULT                hrPrev,
    TcVnRotatedRectangle&  stRectangle,
    ITcVnImage*            ipDestImage,
    TcVnVector4_LREAL&     aColor,
    LONG                   nThickness,
    ETcVnLineType          eLineType = LT_8_CONNECTED
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stRectangle	TcVnRotatedRectangle [▶ 1718]&		要绘制的矩形（旋转角，单位：度）
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&		颜色
nThickness	LONG		线条粗细（如果为负值，则填充矩形）
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.24 FillCircle

绘制一个填充圆。

语法

定义：

```
HRESULT FillCircle(
    HRESULT             hrPrev,
    ULONG               nCenterX,
    ULONG               nCenterY,
    ULONG               nRadius,
    ITcVnImage*         ipDestImage,
    TcVnVector4_LREAL&  aColor
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nCenterX	ULONG	中心的 x 坐标
nCenterY	ULONG	中心的 y 坐标
nRadius	ULONG	半径
ipDestImage	ITcVnImage* [▶ 1863]	目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&	颜色

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.25 FillContours

绘制被解释为轮廓的单个点集或多个点集。

语法

定义：

```
HRESULT FillContours(
    HRESULT             hrPrev,
    ITcVnContainer*     ipContours,
    LONG                nContourIndex,
    ITcVnImage*         ipDestImage,
    TcVnVector4_LREAL&  aColor
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	描述
ipContours	ITcVnContainer* [▶ 1827]	单个轮廓（ContainerType_Vector_TcVnPoint2_DINT）或多个轮廓（ContainerType_Vector_Vector_TcVnPoint2_DINT）
nContourIndex	LONG	要绘制的特定轮廓的索引（如果为负值，则绘制容器内的所有轮廓）
ipDestImage	ITcVnImage* [▶ 1863]	目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&	颜色

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.26 FillEllipse

绘制一个填充椭圆。

语法

定义：

```
HRESULT FillEllipse(
    HRESULT              hrPrev,
    TcVnRotatedRectangle& stEllipse,
    ITcVnImage*          ipDestImage,
    TcVnVector4_LREAL&   aColor
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stEllipse	TcVnRotatedRectangle [▶ 1718]&	要绘制的椭圆（旋转角，单位：度）
ipDestImage	ITcVnImage* [▶ 1863]	目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&	颜色

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.27 FillPolygon

使用一组点绘制一个填充多边形。

语法

定义：

```
HRESULT FillPolygon(
    HRESULT            hrPrev,
    ITcVnContainer*    ipPoints,
    ITcVnImage*        ipDestImage,
    TcVnVector4_LREAL& aColor
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipPoints	ITcVnContainer* [▶ 1827]	包含 TcVnPoint2_DINT、TcVnPoint2_REAL 或 TcVnPoint2_LREAL 元素的容器
ipDestImage	ITcVnImage* [▶ 1863]	目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&	颜色

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.28 FillRectangle

绘制一个填充矩形。

语法

定义：

```
HRESULT FillRectangle(
    HRESULT            hrPrev,
    ULONG              nTopLeftX,
    ULONG              nTopLeftY,
    ULONG              nBottomRightX,
    ULONG              nBottomRightY,
```

```
    ITcVnImage*         ipDestImage,
    TcVnVector4_LREAL&  aColor
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nTopLeftX	ULONG	左上角的 x 坐标
nTopLeftY	ULONG	左上角的 y 坐标
nBottomRightX	ULONG	右下角的 x 坐标
nBottomRightY	ULONG	右下角的 y 坐标
ipDestImage	ITcVnImage* [▶ 1863]	目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&	颜色

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.29 FillRotatedRectangle

绘制一个填充矩形。

语法

定义：

```
HRESULT FillRotatedRectangle(
    HRESULT               hrPrev,
    TcVnRotatedRectangle& stRectangle,
    ITcVnImage*           ipDestImage,
    TcVnVector4_LREAL&    aColor
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stRectangle	TcVnRotatedRectangle [▶ 1718]&	要绘制的矩形（旋转角，单位：度）
ipDestImage	ITcVnImage* [▶ 1863]	目标图像
aColor	TcVnVector4_LREAL [▶ 1641]&	颜色

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.30 PlotIntensityProfile

沿图像中的线段绘制像素强度分布图。

语法

定义：

```
HRESULT PlotIntensityProfile(
    HRESULT            hrPrev,
    ITcVnImage*        ipSrcImage,
    ITcVnImage*&       ipDestImage,
    TcVnPoint2_REAL&   aStartPoint,
    TcVnPoint2_REAL&   aEndPoint,
    TcVnVector4_LREAL& aBackgroundColor = {0, 0, 0, 0},
    TcVnVector4_LREAL& aLineColor = {255, 255, 255, 255},
    TcVnVector4_LREAL& aGridColor = {64, 64, 64, 64},
    LONG               nScaleX = 2,
    LONG               nScaleY = 2,
    LONG               nThickness = 1,
    LONG               nDestChannels = 1
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT，1 通道）
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（USINT，1 通道。x：位置，y：强度，原点：左下角。如有需要，将创建适当的目标图像。）
aStartPoint	TcVnPoint2_REAL [▶ 1640]&		线段的起点
aEndPoint	TcVnPoint2_REAL [▶ 1640]&		线段的终点
aBackgroundColor	TcVnVector4_LREAL [▶ 1641]&	{0, 0, 0, 0}	Background color（背景颜色）
aLineColor	TcVnVector4_LREAL [▶ 1641]&	{255, 255, 255, 255}	线条颜色
aGridColor	TcVnVector4_LREAL [▶ 1641]&	{64, 64, 64, 64}	网格颜色

名称	类型	默认值	描述
nScaleX	LONG	2	在 x 方向上缩放（ipDestImage 的宽度为 nScaleX *（ipSrcImage 宽度 - 1）+ 1）
nScaleY	LONG	2	按 Y 方向缩放（ipDestImage 的高度为 nScaleY * 255 + 1）
nThickness	LONG	1	线条粗细
nDestChannels	LONG	1	ipDestImage 通道（1 或 3）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.31 PutLabel

将标签（统一背景上的文本）写入图像。

语法

定义：

```
HRESULT PutLabel(
    HRESULT             hrPrev,
    PCCH                sText,
    ITcVnImage*         ipDestImage,
    ULONG               nX,
    ULONG               nY,
    double              fFontScale,
    ULONG               nThickness = 1,
    ETcVnFontType       eFontType = FT_HERSHEY_SIMPLEX,
    TcVnVector4_LREAL&  aFontColor = {0, 0, 0, 0},
    TcVnVector4_LREAL&  aBackgroundColor = {255, 255, 255, 255},
    ETcVnLineType       eLineType = LT_4_CONNECTED
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
sText	PCCH		Text
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
nX	ULONG		x 坐标（左下角）
nY	ULONG		y 坐标（左下角）

名称	类型	默认值	描述
fFontScale	double		缩放系数
nThickness	ULONG	1	线条粗细
eFontType	ETcVnFontType [▶ 1678]	FT_HERSHEY_SIMPLEX	字体类型
aFontColor	TcVnVector4_LREAL [▶ 1641]&	{0, 0, 0, 0}	字体颜色
aBackgroundColor	TcVnVector4_LREAL [▶ 1641]&	{255, 255, 255, 255}	Background color（背景颜色）
eLineType	ETcVnLineType [▶ 1682]	LT_4_CONNECTED	线条类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.12.32 PutText

将文本写入图像。

语法

定义：

```
HRESULT PutText(
    HRESULT             hrPrev,
    PCCH                sText,
    ITcVnImage*         ipDestImage,
    ULONG               nX,
    ULONG               nY,
    ETcVnFontType       eFontType,
    double              fFontScale,
    TcVnVector4_LREAL&  aColor,
    LONG                nThickness = 1,
    ETcVnLineType       eLineType = LT_8_CONNECTED,
    bool                bBottomLeftOrigin = false
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
sText	PCCH		Text
ipDestImage	ITcVnImage* [▶ 1863]		目标图像
nX	ULONG		x 坐标（左下角）
nY	ULONG		y 坐标（左下角）
eFontType	ETcVnFontType [▶ 1678]		字体类型
fFontScale	double		缩放系数

名称	类型	默认值	描述
aColor	TcVnVector4_LREAL [▶ 1641]&		文本颜色
nThickness	LONG	1	线条粗细
eLineType	ETcVnLineType [▶ 1682]	LT_8_CONNECTED	线条类型
bBottomLeftOrigin	BOOL	false	如果为 true，则将图像原点设置为左下角

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.13 傅立叶分析

该组包含用于图像频率分析和频率滤波的函数。

函数

巴特沃斯滤波器

- CreateBandpassButterworthFilter [▶ 2337]
- CreateBandrejectButterworthFilter [▶ 2339]
- CreateHighpassButterworthFilter [▶ 2341]
- CreateLowpassButterworthFilter [▶ 2343]

高斯滤波器

- CreateBandpassGaussianFilter [▶ 2338]
- CreateBandrejectGaussianFilter [▶ 2340]
- CreateHighpassGaussianFilter [▶ 2342]
- CreateLowpassGaussianFilter [▶ 2344]

离散傅立叶变换

- Dft [▶ 2345]
- InverseDft [▶ 2346]
- OptimalDftSize [▶ 2347]

6.2.3.13.1 CreateBandpassButterworthFilter

创建带通巴特沃斯（Butterworth）滤波器，通过逐元素乘法将其应用于频域图像。

语法

定义：

```
HRESULT CreateBandpassButterworthFilter(
    HRESULT      hrPrev,
    ITcVnImage*& ipFilter,
    ULONG        nWidth,
    ULONG        nHeight,
    bool         bDoublePrecision,
    bool         bOriginAtCenter,
    double       fCutoffDistance,
    double       fBandWidth,
    ULONG        nOrder,
    double       fScale
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipFilter	ITcVnImage* [▶ 1863]&	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	ULONG	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	ULONG	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	double	巴特沃斯（Butterworth）滤波器的截断距离（> 0）。
fBandWidth	double	巴特沃斯（Butterworth）滤波器的带宽（> 0）。
nOrder	ULONG	巴特沃斯（Butterworth）滤波器的阶数（> 0）。
fScale	double	巴特沃斯（Butterworth）滤波器分母项的比例系数（> 0，例如 0.414 或 1.0）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.13.2 CreateBandpassGaussianFilter

创建带通高斯（Gaussian）滤波器，通过逐元素乘法将其应用于频域图像。

语法

定义：

```
HRESULT CreateBandpassGaussianFilter(
    HRESULT      hrPrev,
    ITcVnImage*& ipFilter,
    ULONG        nWidth,
    ULONG        nHeight,
    bool         bDoublePrecision,
    bool         bOriginAtCenter,
    double       fCutoffDistance,
    double       fBandWidth
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipFilter	ITcVnImage* [▶ 1863]&	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	ULONG	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	ULONG	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	double	高斯（Gaussian）滤波器的截断距离（> 0）。
fBandWidth	double	高斯（Gaussian）滤波器的带宽（> 0）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.13.3 CreateBandrejectButterworthFilter

创建带阻巴特沃斯（Butterworth）滤波器，通过逐元素乘法将其应用于频域图像。

语法

定义：

```
HRESULT CreateBandrejectButterworthFilter(
    HRESULT      hrPrev,
    ITcVnImage*& ipFilter,
    ULONG        nWidth,
    ULONG        nHeight,
    bool         bDoublePrecision,
    bool         bOriginAtCenter,
    double       fCutoffDistance,
    double       fBandWidth,
    ULONG        nOrder,
    double       fScale
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipFilter	ITcVnImage* [▶ 1863]&	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	ULONG	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	ULONG	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	double	巴特沃斯（Butterworth）滤波器的截断距离（> 0）。
fBandWidth	double	巴特沃斯（Butterworth）滤波器的带宽（> 0）。
nOrder	ULONG	巴特沃斯（Butterworth）滤波器的阶数（> 0）。
fScale	double	巴特沃斯（Butterworth）滤波器分母项的比例系数（> 0，例如 0.414 或 1.0）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.13.4 CreateBandrejectGaussianFilter

创建带阻高斯（Gaussian）滤波器，通过逐元素乘法将其应用于频域图像。

语法

定义：

```
HRESULT CreateBandrejectGaussianFilter(
    HRESULT      hrPrev,
    ITcVnImage*& ipFilter,
    ULONG        nWidth,
    ULONG        nHeight,
    bool         bDoublePrecision,
    bool         bOriginAtCenter,
    double       fCutoffDistance,
    double       fBandWidth
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipFilter	ITcVnImage* [▶ 1863]&	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	ULONG	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	ULONG	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	double	高斯（Gaussian）滤波器的截断距离（> 0）。
fBandWidth	double	高斯（Gaussian）滤波器的带宽（> 0）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.13.5 CreateHighpassButterworthFilter

创建高通巴特沃斯（Butterworth）滤波器，通过逐元素乘法将其应用于频域图像。

语法

定义：

```
HRESULT CreateHighpassButterworthFilter(
    HRESULT      hrPrev,
    ITcVnImage*& ipFilter,
    ULONG        nWidth,
    ULONG        nHeight,
    bool         bDoublePrecision,
    bool         bOriginAtCenter,
    double       fCutoffDistance,
    ULONG        nOrder,
    double       fScale
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipFilter	ITcVnImage* [▶ 1863]&	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	ULONG	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	ULONG	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	double	巴特沃斯（Butterworth）滤波器的截断距离（> 0）。
nOrder	ULONG	巴特沃斯（Butterworth）滤波器的阶数（> 0）。
fScale	double	巴特沃斯（Butterworth）滤波器分母项的比例系数（> 0，例如 0.414 或 1.0）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.13.6 CreateHighpassGaussianFilter

创建高通高斯（Gaussian）滤波器，通过逐元素乘法将其应用于频域图像。

语法

定义：

```
HRESULT CreateHighpassGaussianFilter(
    HRESULT       hrPrev,
    ITcVnImage*& ipFilter,
    ULONG         nWidth,
    ULONG         nHeight,
    bool          bDoublePrecision,
    bool          bOriginAtCenter,
    double        fCutoffDistance
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipFilter	ITcVnImage* [▶ 1863]&	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	ULONG	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	ULONG	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	double	高斯（Gaussian）滤波器的截断距离（> 0）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.13.7 CreateLowpassButterworthFilter

创建低通巴特沃斯（Butterworth）滤波器，通过逐元素乘法将其应用于频域图像。

语法

定义：

```
HRESULT CreateLowpassButterworthFilter(
    HRESULT       hrPrev,
    ITcVnImage*& ipFilter,
    ULONG         nWidth,
    ULONG         nHeight,
    bool          bDoublePrecision,
    bool          bOriginAtCenter,
    double        fCutoffDistance,
```

```
    ULONG        nOrder,
    double       fScale
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipFilter	ITcVnImage* [▶ 1863]&	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	ULONG	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	ULONG	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	double	巴特沃斯（Butterworth）滤波器的截断距离（> 0）。
nOrder	ULONG	巴特沃斯（Butterworth）滤波器的阶数（> 0）。
fScale	double	巴特沃斯（Butterworth）滤波器分母项的比例系数（> 0，例如 0.414 或 1.0）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.13.8 CreateLowpassGaussianFilter

创建低通高斯（Gaussian）滤波器，通过逐元素乘法将其应用于频域图像。

语法

定义：

```
HRESULT CreateLowpassGaussianFilter(
    HRESULT      hrPrev,
    ITcVnImage*& ipFilter,
    ULONG        nWidth,
    ULONG        nHeight,
    bool         bDoublePrecision,
```

```
    bool          bOriginAtCenter,
    double        fCutoffDistance
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipFilter	ITcVnImage* [▶ 1863]&	返回创建的滤波器（1 通道，REAL 或 LREAL 类型，取决于 bDoublePrecision。重复使用非零接口指针）。
nWidth	ULONG	滤波宽度（偶数，>= 2，必须与所应用的图像宽度相匹配）
nHeight	ULONG	滤波高度（偶数，>= 2，必须与所应用的图像高度相匹配）
bDoublePrecision	BOOL	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成滤波器。
bOriginAtCenter	BOOL	如果为 true，滤波器原点将移至图像中心。
fCutoffDistance	double	高斯（Gaussian）滤波器的截断距离（> 0）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.13.9 Dft

计算给定图像的 DFT。

语法

定义：

```
HRESULT Dft(
    HRESULT      hrPrev,
    ITcVnImage*  ipSpatialImage,
    ITcVnImage*& ipFrequencyImage,
    bool         bPackedCCS,
    bool         bAutoPadding
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSpatialImage	ITcVnImage* [▶ 1863]	源图像（REAL 或 LREAL 类型的元素，1（Re）或 2（Re + Im）通道）
ipFrequencyImage	ITcVnImage* [▶ 1863]&	目标图像（类型与 ipSpatialImage 相同，但通道数量和大小可能因 bPackedCCS 和 bAutoPadding 而异）。
bPackedCCS	BOOL	如果为 true，且 ipSpatialImage 只有 1 个通道，则结果图像将只有 1 个通道，并带有打包的复合共轭对称形式结果。否则，结果图像将有 2 个独立通道（Re + Im），包含整个光谱。
bAutoPadding	BOOL	如果为 true，输入图像会自动填充（0）到最佳尺寸（如果需要），以加快 dft 的速度（创建临时副本，使 ipSpatialImage 的内容保持不变，这也需要一些额外的计算能力。因此，建议将有填充的执行时间和无填充的执行时间进行比较）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.13.10 InverseDft

计算给定频率图像的离散傅里叶逆变换（inverse DFT）。

语法

定义：

```
HRESULT InverseDft(
    HRESULT      hrPrev,
    ITcVnImage*  ipFrequencyImage,
    ITcVnImage*& ipSpatialImage,
    bool         bRealOutput
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipFrequencyImage	ITcVnImage* [▶ 1863]	源频率图像（REAL 或 LREAL 类型的元素，1（打包 CCS）或 2（Re + Im）通道）
ipSpatialImage	ITcVnImage* [▶ 1863]&	目标图像（类型与 ipFrequencyImage 相同，但通道数量可因 bRealOutput 而异。如有需要，将创建适当的图像）。
bRealOutput	BOOL	仅当 ipFrequencyImage 有 2 个通道时才相关。如果为 true，结果图像将只有 1 个通道。否则，结果图像将有 2 个独立的通道（Re + Im）。如果 ipFrequencyImage 只有 1 个通道，则假定为打包 CCS 格式，结果图像将始终只有 1 个通道。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.13.11 OptimalDftSize

计算应用离散傅里叶变换（DFT）的最佳图像大小（可提高性能）。

语法

定义：

```
HRESULT OptimalDftSize(
    HRESULT     hrPrev,
    ITcVnImage* ipImage,
    ULONG&      nOptWidth,
    ULONG&      nOptHeight
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]	为其计算最佳宽度和高度的源图像
nOptWidth	ULONG&	返回图像的最佳宽度

名称	类型	描述
nOptHeight	ULONG&	返回图像的最佳高度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14 几何变换和坐标变换

该组包含用于图像和像素转换的函数。

函数

标定 [▶ 2350]

- 内部和外部标定
- 光学几何畸变补偿
- 图像坐标与世界坐标之间的转换

仿射转换

- ApplyRotationToAffineTransformation [▶ 2376]
- ApplyScalingToAffineTransformation [▶ 2377]
- ApplyTranslationToAffineTransformation [▶ 2377]
- ApplyYAxisInversionToAffineTransformation [▶ 2378]
- DecomposeAffineTransformation [▶ 2387]
- GenerateAffineTransformationUnitMatrix2D [▶ 2390]
- GetAffineTransformation [▶ 2391]
- GetAffineTransformation2D [▶ 2392]
- InvertAffineTransformation [▶ 2395]
- WarpAffine(Exp) [▶ 2405]
- WarpAffine_Container [▶ 2405]
- WarpAffine_Point [▶ 2406]
- WarpAffine_Rectangle [▶ 2407]

笛卡尔坐标和极坐标之间的转换

- ConvertCartesianToPolarAngleImage [▶ 2379]
- ConvertCartesianToPolarAngles [▶ 2380]
- ConvertCartesianToPolarImages [▶ 2381]
- ConvertCartesianToPolarMagnitudeImage [▶ 2382]
- ConvertCartesianToPolarMagnitudes [▶ 2382]
- ConvertCartesianToPolarPoints [▶ 2383]

- ConvertPolarToCartesianImages [▶ 2385]
- ConvertPolarToCartesianPoints [▶ 2386]
- RemapImage [▶ 2397]
- RemapImageToLogPolarSpace [▶ 2399]
- RemapImageToPolarSpace [▶ 2400]
- ConvertMaps [▶ 2384]

简单图像运算

- FlipImage [▶ 2390]
- PyramidDown [▶ 2396]
- PyramidUp [▶ 2397]
- ResizeImage [▶ 2401]

简单几何运算

- ResizeRegion_TcVnRectangle_DINT [▶ 2402]

图像对齐和旋转

- AlignRotatedImageRegion [▶ 2375]
- RotateImage(Exp) [▶ 2403]

透视转换

- DecomposeHomography [▶ 2388]
- GetPerspectiveTransformation [▶ 2393]
- Homography [▶ 2394]
- PerspectiveTransformation [▶ 2395]
- WarpPerspective(Exp) [▶ 2409]
- WarpPerspective_Container [▶ 2409]
- WarpPerspective_Point [▶ 2410]
- WarpPerspective_Rectangle [▶ 2411]

还请参阅有关此

MergeRegions [▶ 2349]

6.2.3.14.1 MergeRegions

Merge rectangles that overlap or whose edge-to-edge gap is within the specified distance.

Syntax

Definition:

```
HRESULT MergeRegions(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcContainer,
    ITcVnContainer*& ipDestContainer,
    ULONG            nDistance,
    ULONGLONG        nOptions
)
```

Parameters

Name	Type	Description
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)
ipSrcContainer	ITcVnContainer* [▶ 1827]	Source container containing rectangles to be merged (ContainerType_Vector_TcVnRectangle_DINT)
ipDestContainer	ITcVnContainer* [▶ 1827]&	Returns the resulting container, same type as ipSrcContainer
nDistance	ULONG	Maximum edge-to-edge gap (in pixels) for two non-overlapping rectangles to be merged (overlapping rectangles are always merged)
nOptions	ULONGLONG	Specifies the method for merging regions. Valid values are defined in ETcVnMergeRegionsOptions.

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.2.3.14.2 标定

该子组包含用于像素与现实世界像素之间相关性的函数。它们通常与测量函数 [▶ 2547]一起使用。

函数

标定

- CalibrateCamera(Exp) [▶ 2352]
- CalibrateCameraManually [▶ 2355]
- CalibrateCameraPlanar [▶ 2356]
- CalibrateLinescanCamera [▶ 2357]
- DetectPatternPoints [▶ 2364]
- DetectPatternPoints2 [▶ 2365]
- GenerateCalibrationPatternReferencePoints [▶ 2366]
- SortDetectedPatternPoints [▶ 2369]
- SolvePnP [▶ 2368]

镜头几何畸变补偿

- CompensateLensDistortion [▶ 2359]
- CompensateLensDistortionForPoints [▶ 2361]

图像坐标与世界坐标之间的转换

- TransformCoordinatesImageToWorld [▶ 2371]
- TransformCoordinatesWorldToImage [▶ 2374]
- TransformCoordinatesPlanar [▶ 2373]
- ImagePointsWorldDistance [▶ 2367]

输入图像要求

输入图像不应使用压缩或滤波格式，如Bayer格式。就标定函数而言，这样做的后果是干扰模板识别，从而导致标定结果较差或根本没有。由于在补偿函数过程中进行了插值，改变了像素的原始排列，因此之后无法再正确转换图像。因此，建议在使用该子组中的函数之前一定要先进行转换。

相机矩阵要求

$$CameraMatrix = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

相机矩阵由焦距 fx 和 fy 以及主点（cx, cy）组成，以像素单位表示。主点是光轴与图像的交点，通常位于图像中心附近。必须满足以下条件：

- 0 < fx 和 fy
- cx >= 0 和 <= 图像宽度 -1
- cy >= 0 和 <= 图像高度
- 如图所示，矩阵中的其他位置必须填 0 或 1。不允许其他值。

还请参阅有关此

SortAxisAlignedPatternPoints [▶ 2351]

6.2.3.14.2.1 SortAxisAlignedPatternPoints

Sorts the points of a horizontally aligned asymmetric or symmetric calibration pattern in place. Rows top to bottom (ascending y), and points within each row left to right (ascending x). Individual pattern rows must not have overlapping y-coordinates.

Syntax

Definition:

```
HRESULT SortAxisAlignedPatternPoints(
    HRESULT         hrPrev,
    ITcVnContainer* ipImagePoints,
    ULONG           nPatternRows,
    ULONG           nPatternCols
)
```

Parameters

Name	Type	Description
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)

Name	Type	Description
ipImagePoints	ITcVnContainer* [▶ 1827]	Detected pattern point positions in the image coordinate system (will be sorted by this function, ContainerType_Vector_TcVnPoint2_REAL)
nPatternRows	ULONG	Number of pattern rows
nPatternCols	ULONG	Number of pattern columns

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.2.3.14.2.2 CalibrateCamera

通过评估包含标定图案（圆圈）的图像来计算相机参数（内参+外参）。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT CalibrateCamera(
    HRESULT                       hrPrev,
    ITcVnImage*                   ipSrcImage,
    ITcVnContainer*               ipReferencePoints,
    TcVnMatrix3x3_LREAL&          aCameraMatrix,
    TcVnArray8_LREAL&             aDistortionCoefficients,
    TcVnMatrix3x3_LREAL&          aRotationMatrix,
    TcVnVector3_LREAL&            aTranslationVector,
    double&                       fReprojError,
    TcVnParamsBlobDetection&      stBlobDetectionParams,
    TcVnCameraCalibrationOptions& stCalibrationOptions
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	包含标定板的源图像（1 通道，USINT 元素）
ipReferencePoints	ITcVnContainer* [▶ 1827]	标定板参考点的位置（ContainerType_Vector_TcVnPoint3_REAL）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	返回相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&	返回镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	返回旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 1641]&	返回平移向量

名称	类型	描述
fReprojError	double&	返回重投影误差
stBlobDetectionParams	TcVnParamsBlobDetection [▶ 1709]&	用于内部使用的 F_VN_DetectBlobs 函数的参数
stCalibrationOptions	TcVnCameraCalibrationOptions [▶ 1702]&	标定选项

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.3 CalibrateCamera (Exp)

通过评估包含标定板（圆）的图像来计算相机参数（内参+外参）。为数组中的第一个图像计算外参。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT CalibrateCamera(
    HRESULT                         hrPrev,
    PVOID                           pSrcImages,
    USHORT                          nSrcArraySize,
    ITcVnContainer*                 ipReferencePoints,
    TcVnMatrix3x3_LREAL&            aCameraMatrix,
    TcVnArray8_LREAL&               aDistortionCoefficients,
    TcVnMatrix3x3_LREAL&            aRotationMatrix,
    TcVnVector3_LREAL&              aTranslationVector,
    double&                         fReprojError,
    TcVnParamsBlobDetection&        stBlobDetectionParams,
    TcVnCameraCalibrationOptions&   stCalibrationOptions,
    bool                            bSubpixelAccuracy = false,
    ETcVnEdgeDirection              eEdgeDirection = ED_DARK_TO_LIGHT,
    float                           fMinStrength = 50,
    ULONG                           nMaxThickness = 10,
    ULONG                           nSubpixelsIterations = 10,
    ULONG                           nSearchLines = 92,
    float                           fApproxPrecision = 0.001f,
    ETcVnEdgeDetectionAlgorithm     eAlgorithm = EDA_INTERPOLATION,
    ETcVnWorldCoordinateSystem      eWorldSystem = WCS_X_RIGHT_Y_DOWN
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
pSrcImages	PVOID		指向源图像数组的指针，每个源图像包含相同的标定板（1 通道，USINT 元素）

名称	类型	默认值	描述
nSrcArraySize	USHORT		pSrcImages 数组大小
ipReferencePoints	ITcVnContainer* [▶ 1827]		标定板参考点的位置（ContainerType_Vector_TcVnPoint3_REAL）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&		返回相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&		返回镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&		返回旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 1641]&		返回平移向量
fReprojError	double&		返回重投影误差
stBlobDetectionParams	TcVnParamsBlobDetection [▶ 1709]&		用于内部使用的 F_VN_DetectBlobs 函数的参数
stCalibrationOptions	TcVnCameraCalibrationOptions [▶ 1702]&		标定选项
bSubpixelAccuracy	BOOL	false	如果为 true，则以亚像素精度检测图案点
eEdgeDirection	ETcVnEdgeDirection [▶ 1673]	ED_DARK_TO_LIGHT	指定要搜索的边缘方向（从中心到椭圆外侧）
fMinStrength	float	50	指定要搜索的边缘的最小强度（强度差）
nMaxThickness	ULONG	10	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
nSubpixelsIterations	ULONG	10	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
nSearchLines	ULONG	92	指定搜索线的数量，在所有方向上平均分配（必须 >= 8 且是 4 的倍数）
fApproxPrecision	float	0.001f	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 1673]	EDA_INTERPOLATION	选择边缘检测算法
eWorldSystem	ETcVnWorldCoordinateSystem [▶ 1696]	WCS_X_RIGHT_Y_DOWN	世界坐标系方向

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.4 CalibrateCameraManually

计算相机参数（内参+外参）。外参是针对第一个内部容器计算的。

语法

定义：

```
HRESULT CalibrateCameraManually(
    HRESULT                         hrPrev,
    ITcVnContainer*                 ipImagePoints,
    ULONG                           nImageWidth,
    ULONG                           nImageHeight,
    ITcVnContainer*                 ipReferencePoints,
    TcVnMatrix3x3_LREAL&            aCameraMatrix,
    TcVnArray8_LREAL&               aDistortionCoefficients,
    TcVnMatrix3x3_LREAL&            aRotationMatrix,
    TcVnVector3_LREAL&              aTranslationVector,
    double&                         fReprojError,
    TcVnCameraCalibrationOptions&   stCalibrationOptions,
    ETcVnWorldCoordinateSystem      eWorldSystem = WCS_X_RIGHT_Y_DOWN
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImagePoints	ITcVnContainer* [▶ 1827]		提供多个图像的图像点（ContainerType_Vector_Vector_TcVnPoint2_REAL，每个内部容器至少 6 个点）。容器和点的阶数必须与 ipReferencePoints 匹配。
nImageWidth	ULONG		图像宽度
nImageHeight	ULONG		图像高度
ipReferencePoints	ITcVnContainer* [▶ 1827]		世界参考点（ContainerType_Vector_Vector_TcVnPoint3_REAL）。内部容器的数量及其点数必须与 ipImagePoints 匹配。

名称	类型	默认值	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&		返回相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&		返回镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&		返回（第一个内部容器的）旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 1641]&		返回（第一个内部容器的）平移向量
fReprojError	double&		返回重投影误差
stCalibrationOptions	TcVnCameraCalibrationOptions [▶ 1702]&		标定选项
eWorldSystem	ETcVnWorldCoordinateSystem [▶ 1696]	WCS_X_RIGHT_Y_DOWN	世界坐标系方向

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.5 CalibrateCameraPlanar

使用平面标定板标定相机，该标定板由四个标记矩形四角的圆圈组成。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT CalibrateCameraPlanar(
    HRESULT                 hrPrev,
    ITcVnImage*             ipSrcImage,
    TcVnArray4_Point2_REAL& aMarkerPositions,
    TcVnMatrix3x3_LREAL&    aTransformationMatrix,
    TcVnArray4_Point2_REAL& aMarkerImagePositions,
    ULONG                   nMinMarkerArea = 100,
    ULONG                   nMaxMarkerArea = 10000,
    double                  fMaxMarkerEccentricity = 0.05
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	默认值	描述
ipSrcImage	ITcVnImage* [▶ 1863]		二值源图像（背景为零，标记非零）
aMarkerPositions	TcVnArray4_Point2_REAL [▶ 1641]&		在其平面内使用世界单位的标记位置（右侧坐标系）
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&		返回一个 3×3 矩阵，用于将图像坐标转换为参考坐标系
aMarkerImagePositions	TcVnArray4_Point2_REAL [▶ 1641]&		返回标记位置的图像坐标（左侧坐标系）
nMinMarkerArea	ULONG	100	标记检测的最小面积（较小的标记将被忽略。）
nMaxMarkerArea	ULONG	10000	标记检测的最面积（较大的标记将被忽略）。
fMaxMarkerEccentricity	double	0.05	用于标记检测的最大偏心率（偏心率用于测量偏离圆形的程度（偏心率 = 0：圆形物体；偏心率 = 1.0：线形物体）。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.6 CalibrateLinescanCamera

通过评估包含直线标定板的图像来计算相机参数（内参+外参）。标定板必须由垂直线和对角线交替组成，每条对角线都有相同的恒定斜率：|\|\|...\|。结果只适用于线扫图像的 X 方向。要获得转换为世界坐标的精确 x 值，y 坐标必须设置为传感器宽度的一半，等于相机矩阵的 cy 值。对于其它 y 值，结果是近似值，可能不准确。为数组中的第一个图像计算外参。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT CalibrateLinescanCamera(
    HRESULT                         hrPrev,
    PVOID                           pSrcImages,
    USHORT                          nSrcArraySize,
    ITcVnContainer*                 ipPatternLineOrigins,
    double                          fSlopeRad,
    TcVnMatrix3x3_LREAL&            aCameraMatrix,
    TcVnArray8_LREAL&               aDistortionCoefficients,
    TcVnMatrix3x3_LREAL&            aRotationMatrix,
    TcVnVector3_LREAL&              aTranslationVector,
    double&                         fReprojError,
    TcVnCameraCalibrationOptions&   stCalibrationOptions,
    ETcVnEdgeDirection              eEdgeDirection,
    float                           fMinStrength = 100,
```

```
    ULONG                           nMaxSearchLines = 11,
    ULONG                           nMaxThickness = 5,
    ULONG                           nSubpixelsIterations = 50,
    float                           fApproxPrecision = 0.001f,
    ETcVnEdgeDetectionAlgorithm     eAlgorithm = EDA_APPROX_ERF
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
pSrcImages	PVOID		指向包含 1 个或多个源图像（1 通道）的数组的指针。如果图像包含不止 1 行，则每行必须包含相同标定位置的图像。
nSrcArraySize	USHORT		pSrcImages 数组大小
ipPatternLineOrigins	ITcVnContainer* [▶ 1827]		标定板线原点的 X 位置（ContainerType_Vector_REAL，垂直线和其后的对角线的原点通常相同）
fSlopeRad	double		对角线相对于垂直线的斜率，单位：弧度（> 0，< PI/2）（即 0 为垂直线，PI/2 为水平线）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&		返回相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&		返回镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&		返回旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 1641]&		返回平移向量
fReprojError	double&		返回重投影误差
stCalibrationOptions	TcVnCameraCalibrationOptions [▶ 1702]&		标定选项（建议设置 bFixAspectRatio、bFixPrincipalPoint 和 bZeroTangentDist）
eEdgeDirection	ETcVnEdgeDirection [▶ 1673]		指定要搜索的边缘方向
fMinStrength	float	100	指定要搜索的边缘的最小强度（强度差）
nMaxSearchLines	ULONG	11	搜索线最大数量（在图像高度上平均分配，每行最多 1 条搜索线）
nMaxThickness	ULONG	5	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
nSubpixelsIterations	ULONG	50	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优

名称	类型	默认值	描述
			化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	float	0.001f	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 1673]	EDA_APPROX_ERF	选择边缘检测算法

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.7 CompensateLensDistortion

转换图像，以补偿镜头畸变。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT CompensateLensDistortion(
    HRESULT              hrPrev,
    ITcVnImage*          ipSrcImage,
    ITcVnImage*&         ipDestImage,
    TcVnMatrix3x3_LREAL& aCameraMatrix,
    TcVnArray4_LREAL&    aDistortionCoefficients
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（必须与 ipSrcImage 不同！如有需要，将创建适当的目标图像。）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	包含内参的相机矩阵
aDistortionCoefficients	TcVnArray4_LREAL [▶ 1641]&	镜头几何畸变系数 [k1、k2、p1、p2]

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.8 CompensateLensDistortion (Exp1)

转换图像，以补偿镜头畸变。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT CompensateLensDistortion(
    HRESULT              hrPrev,
    ITcVnImage*          ipSrcImage,
    ITcVnImage*&         ipDestImage,
    TcVnMatrix3x3_LREAL& aCameraMatrix,
    TcVnArray8_LREAL&    aDistortionCoefficients
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（必须与 ipSrcImage 不同！如有需要，将创建适当的目标图像。）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	包含内参的相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&	镜头几何畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.9 CompensateLensDistortion (Exp2)

转换图像，以补偿镜头畸变。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT CompensateLensDistortion(
    HRESULT              hrPrev,
    ITcVnImage*          ipSrcImage,
    ITcVnImage*&         ipDestImage,
    TcVnMatrix3x3_LREAL& aCameraMatrix,
    TcVnArray8_LREAL&    aDistortionCoefficients,
    TcVnMatrix3x3_LREAL& aNewCameraMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（必须与 ipSrcImage 不同！如有需要，将创建适当的目标图像。）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	包含内参的相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&	镜头几何畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aNewCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	允许对结果图像进行额外的缩放和移动

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.10 CompensateLensDistortionForPoints

转换点坐标，以补偿镜头畸变。

语法

定义：

```
HRESULT CompensateLensDistortionForPoints(
    HRESULT              hrPrev,
    ITcVnContainer*      ipSrcPoints,
    ITcVnContainer*&     ipDestPoints,
    TcVnMatrix3x3_LREAL& aCameraMatrix,
    TcVnArray4_LREAL&    aDistortionCoefficients
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcPoints	ITcVnContainer* [▶ 1827]	包含源点坐标的容器（ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_DINT）
ipDestPoints	ITcVnContainer* [▶ 1827]&	返回包含转换后点坐标的容器（ContainerType_Vector_TcVnPoint2_REAL）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	包含内参的相机矩阵
aDistortionCoefficients	TcVnArray4_LREAL [▶ 1641]&	镜头畸变系数 [k1、k2、p1、p2]

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.11 CompensateLensDistortionForPoints (Exp1)

转换点坐标，以补偿镜头畸变。

语法

定义：

```
HRESULT CompensateLensDistortionForPoints(
    HRESULT              hrPrev,
    ITcVnContainer*      ipSrcPoints,
    ITcVnContainer*&     ipDestPoints,
    TcVnMatrix3x3_LREAL& aCameraMatrix,
    TcVnArray8_LREAL&    aDistortionCoefficients
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcPoints	ITcVnContainer* [▶ 1827]	包含源点坐标的容器（ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_DINT）

名称	类型	描述
ipDestPoints	ITcVnContainer* [▶ 1827]&	返回包含转换后点坐标的容器（ContainerType_Vector_TcVnPoint2_REAL）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	包含内参的相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.12 CompensateLensDistortionForPoints (Exp2)

转换点坐标，以补偿镜头畸变。

语法

定义：

```
HRESULT CompensateLensDistortionForPoints(
    HRESULT              hrPrev,
    ITcVnContainer*      ipSrcPoints,
    ITcVnContainer*&     ipDestPoints,
    TcVnMatrix3x3_LREAL& aCameraMatrix,
    TcVnArray8_LREAL&    aDistortionCoefficients,
    TcVnMatrix3x3_LREAL& aNewCameraMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcPoints	ITcVnContainer* [▶ 1827]	包含源点坐标的容器（ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_DINT）
ipDestPoints	ITcVnContainer* [▶ 1827]&	返回包含转换后点坐标的容器（ContainerType_Vector_TcVnPoint2_REAL）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	包含内参的相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aNewCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	允许对结果点坐标进行额外移动

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.13 DetectPatternPoints

在提供的图像中检测标定板点（圆）。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT DetectPatternPoints(
    HRESULT                     hrPrev,
    ITcVnImage*                 ipSrcImage,
    ITcVnContainer*&            ipPatternPoints,
    TcVnParamsBlobDetection&    stBlobDetectionParams,
    ULONG                       nNumberOfPoints,
    float                       fSizeDiffThresFactor = 0.3f,
    bool                        bSubpixelAccuracy = false,
    ETcVnEdgeDirection          eEdgeDirection = ED_DARK_TO_LIGHT,
    float                       fMinStrength = 50,
    ULONG                       nMaxThickness = 10,
    ULONG                       nSubpixelsIterations = 10,
    ULONG                       nSearchLines = 92,
    float                       fApproxPrecision = 0.001f,
    ETcVnEdgeDetectionAlgorithm eAlgorithm = EDA_INTERPOLATION
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT，1 通道）
ipPatternPoints	ITcVnContainer* [▶ 1827]&		返回标定板的点位置（ContainerType_Vector_TcVnPoint2_REAL；重复使用非零接口指针。）
stBlobDetectionParams	TcVnParamsBlobDetection [▶ 1709]&		用于检测和筛选轮廓的参数。
nNumberOfPoints	ULONG		预期的图案点数
fSizeDiffThresFactor	float	0.3f	用于过滤轮廓的阈值：轮廓面积与平均轮廓面积的偏差超过（平均面积 * fSizeDiffThresFactor）时被过滤

名称	类型	默认值	描述
bSubpixelAccuracy	bool	false	如果为 true，则以亚像素精度检测图案点
eEdgeDirection	ETcVnEdgeDirection [▶ 1673]	ED_DARK_TO_LIGHT	指定要搜索的边缘方向（从圆心到外圆）
fMinStrength	float	50	指定要搜索的边缘的最小强度（强度差）
nMaxThickness	ULONG	10	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
nSubpixelsIterations	ULONG	10	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
nSearchLines	ULONG	92	指定搜索线的数量，在所有方向上平均分配（必须 >= 8 且是 4 的倍数）
fApproxPrecision	float	0.001f	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 1673]	EDA_INTERPOLATION	选择边缘检测算法

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.14 DetectPatternPoints (2)

在提供的图像中检测标定板点（圆），并对其进行排序。
可以使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT DetectPatternPoints(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipPatternPoints,
```

```
    TcVnParamsSB&     stBlobDetectionParams,
    ULONG             nPatternRows,
    ULONG             nPatternCols,
    ULONG             nOptions
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（USINT、1 或 3 通道）
ipPatternPoints	ITcVnContainer* [▶ 1827]&	返回标定板的点位置，按行从左到右排序（ContainerType_Vector_TcVnPoint2_REAL；重复使用非零接口指针。）
stBlobDetectionParams	TcVnParamsSB [▶ 1715]&	用于检测和筛选轮廓的参数。至少应为 fMinArea 和 fMaxArea 指定合适的值。
nPatternRows	ULONG	预期的图案行数
nPatternCols	ULONG	预期的图案列数
nOptions	ULONG	提供来自 ETcVnDetectPatternPointsOptions 的选项。始终需要选择 DPPO_SYMMETRIC 或 DPPO_ASYMMETRIC。可以选择通过按位 OR 与 DPPO_CLUSTERING 组合使用。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

6.2.3.14.2.15 GenerateCalibrationPatternReferencePoints

为常见标定板生成参考点。

语法

定义：

```
HRESULT GenerateCalibrationPatternReferencePoints(
    HRESULT                       hrPrev,
    ITcVnContainer*&              ipReferencePoints,
    ETcVnCalibrationPattern       ePattern,
    ETcVnCalibrationPatternOrigin eOrigin,
    ULONG                         nWidth,
    ULONG                         nHeight,
    float                         fDistX,
    float                         fDistY
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipReferencePoints	ITcVnContainer* [▶ 1827]&	返回生成的参考点（ContainerType_Vector_TcVnPoint3_REAL）。
ePattern	ETcVnCalibrationPattern [▶ 1649]	标定板类型
eOrigin	ETcVnCalibrationPatternOrigin [▶ 1649]	标定板原点
nWidth	ULONG	图案宽度，即 x 方向上的图案点数
nHeight	ULONG	图案高度，即 y 方向上的图案点数
fDistX	float	x 方向上图案点之间的距离
fDistY	float	y 方向上图案点之间的距离

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.16 ImagePointsWorldDistance

计算图像坐标中给定的两点（位于同一世界坐标 z 层）在世界坐标系中的距离。

语法

定义：

```
HRESULT ImagePointsWorldDistance(
    HRESULT              hrPrev,
    TcVnPoint2_REAL&     aImagePoint1,
    TcVnPoint2_REAL&     aImagePoint2,
    double&              fWorldDistance,
    TcVnMatrix3x3_LREAL& aCameraMatrix,
    TcVnArray8_LREAL&    aDistortionCoefficients,
    TcVnMatrix3x3_LREAL& aRotationMatrix,
    TcVnVector3_LREAL&   aTranslationVector,
    double               fZ
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aImagePoint1	TcVnPoint2_REAL [▶ 1640]&	图像坐标系中的第一个点
aImagePoint2	TcVnPoint2_REAL [▶ 1640]&	图像坐标系中的第二点

名称	类型	描述
fWorldDistance	double&	返回 aImagePoint1 和 aImagePoint2 在世界坐标系中的距离
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	包含内参的 3x3 相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	3x3 旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 1641]&	平移矢量
fZ	double	给定点的 z 坐标（世界坐标系）（0 表示在（顶层）标定板处）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.17 SolvePnP

根据 3D-2D 点对应关系计算旋转矩阵和平移向量。

语法

定义：

```
HRESULT SolvePnP(
    HRESULT              hrPrev,
    ITcVnContainer*      ipImagePoints,
    ITcVnContainer*      ipReferencePoints,
    TcVnMatrix3x3_LREAL& aCameraMatrix,
    TcVnArray8_LREAL&    aDistortionCoefficients,
    TcVnMatrix3x3_LREAL& aRotationMatrix,
    TcVnVector3_LREAL&   aTranslationVector,
    double&              fReprojError,
    ETcVnSolvePnPMethod  eMethod = SPM_ITERATIVE,
    bool                 bUseExtrinsicGuess = false
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImagePoints	ITcVnContainer* [▶ 1827]		图像点（ContainerType_Vector_TcVnPoint2_REAL，至少需要有 4 个点的顺序与 ipReferencePoints 中的顺序相同）

名称	类型	默认值	描述
ipReferencePoints	ITcVnContainer* [▶ 1827]		世界参考点（ContainerType_Vector_TcVnPoint3_REAL，与 ipImagePoints 中的点数相同）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&		相机矩阵（例如，使用其中一个 CalibrateCamera 函数计算得出）
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&		镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]（例如，使用其中一个 CalibrateCamera 函数计算得出）
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&		返回旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 1641]&		返回平移向量
fReprojError	double&		返回重投影误差
eMethod	ETcVnSolvePnPMethod [▶ 1690]	SPM_ITERATIVE	PnP 求解算法
bUseExtrinsicGuess	BOOL	false	如果为 true，所提供的旋转矩阵和平移向量将用作初始猜测，以实现进一步优化（仅支持 SPM_ITERATIVE 方法）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.18 SortDetectedPatternPoints

根据标定板点的相对位置对检测到的图案点进行排序。

语法

定义：

```
HRESULT SortDetectedPatternPoints(
    HRESULT         hrPrev,
    ITcVnContainer* ipImagePoints,
    ITcVnContainer* ipPatternPoints
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImagePoints	ITcVnContainer* [▶ 1827]	图像坐标系中检测到的图案点位置（将通过函数 ContainerType_Vector_TcVnPoint2_REAL 进行排序）
ipPatternPoints	ITcVnContainer* [▶ 1827]	标定板坐标系中的图案点，用作对 ipImagePoints（ContainerType_Vector_TcVnPoint3_REAL）进行排序时的参考值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.19 TransformCoordinatesImageToWorld (Container)

将二维图像点坐标转换为三维世界坐标（使用内参和外参标定结果）。

语法

定义：

```
HRESULT TransformCoordinatesImageToWorld(
    HRESULT             hrPrev,
    ITcVnContainer*     ipSrcPoints2D,
    ITcVnContainer*&    ipDestPoints3D,
    TcVnMatrix3x3_LREAL& aCameraMatrix,
    TcVnArray8_LREAL&   aDistortionCoefficients,
    TcVnMatrix3x3_LREAL& aRotationMatrix,
    TcVnVector3_LREAL&  aTranslationVector,
    double              fZ
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcPoints2D	ITcVnContainer* [▶ 1827]	包含二维源点的容器（TcVnPoint2_DINT 或 TcVnPoint2_REAL 或 TcVnPoint2_LREAL）
ipDestPoints3D	ITcVnContainer* [▶ 1827]&	返回变换后的三维点（TcVnPoint3_LREAL）

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	包含内参的 3x3 相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	3x3 旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 1641]&	平移矢量
fZ	double	变换点的 z 坐标（世界坐标系）（0 表示在（顶层）标定板处）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.20 TransformCoordinatesImageToWorld (Point)

将二维图像点坐标转换为三维世界坐标（使用内参和外参标定结果）。

语法

定义：

```
HRESULT TransformCoordinatesImageToWorld(
    HRESULT             hrPrev,
    TcVnPoint2_LREAL&   aSrcPoint,
    TcVnPoint3_LREAL&   aDestPoint,
    TcVnMatrix3x3_LREAL& aCameraMatrix,
    TcVnArray8_LREAL&   aDistortionCoefficients,
    TcVnMatrix3x3_LREAL& aRotationMatrix,
    TcVnVector3_LREAL&  aTranslationVector,
    double              fZ
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aSrcPoint	TcVnPoint2_LREAL [▶ 1640]&	图像坐标系中的点
aDestPoint	TcVnPoint3_LREAL [▶ 1640]&	返回世界坐标系中的点
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	包含内参的 3x3 相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	3x3 旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 1641]&	平移矢量
fZ	double	变换点的 z 坐标（世界坐标系）（0 表示在（顶层）标定板处）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.21 TransformCoordinatesPlanar (Container)

为给定的图像点计算世界坐标，反之亦然。

语法

定义：

```
HRESULT TransformCoordinatesPlanar(
    HRESULT              hrPrev,
    ITcVnContainer*      ipSrcPoints,
    ITcVnContainer*&     ipDestPoints,
    TcVnMatrix3x3_LREAL& aTransformationMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcPoints	ITcVnContainer* [▶ 1827]	包含二维源点的容器（TcVnPoint2_DINT 或 TcVnPoint2_REAL 或 TcVnPoint2_LREAL）
ipDestPoints	ITcVnContainer* [▶ 1827]&	返回变换后的二维点（TcVnPoint2_LREAL）
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	通过 VnCalibrateCameraPlanar() 获得的变换矩阵或通过 VnInvertMatrix3x3() 获得的逆矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.22 TransformCoordinatesPlanar (Point)

为给定的图像点计算世界坐标，反之亦然。

语法

定义：

```
HRESULT TransformCoordinatesPlanar(
    HRESULT              hrPrev,
    TcVnPoint2_LREAL&    aSrcPoint,
    TcVnPoint2_LREAL&    aDestPoint,
    TcVnMatrix3x3_LREAL& aTransformationMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aSrcPoint	TcVnPoint2_LREAL [▶ 1640]&	源坐标系中的点
aDestPoint	TcVnPoint2_LREAL [▶ 1640]&	返回目标坐标系中的点
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	通过 VnCalibrateCameraPlanar() 获得的变换矩阵或通过 VnInvertMatrix3x3() 获得的逆矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.23 TransformCoordinatesWorldToImage (Container)

将三维世界坐标转换为二维图像坐标（使用内参和外参标定结果）。

语法

定义：

```
HRESULT TransformCoordinatesWorldToImage(
    HRESULT              hrPrev,
    ITcVnContainer*      ipSrcPoints3D,
    ITcVnContainer*&     ipDestPoints2D,
    TcVnMatrix3x3_LREAL& aCameraMatrix,
    TcVnArray8_LREAL&    aDistortionCoefficients,
    TcVnMatrix3x3_LREAL& aRotationMatrix,
    TcVnVector3_LREAL&   aTranslationVector,
    bool                 bInteger
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcPoints3D	ITcVnContainer* [▶ 1827]	包含三维源点的容器（TcVnPoint3_LREAL）
ipDestPoints2D	ITcVnContainer* [▶ 1827]&	返回变换后的二维点（TcVnPoint2_REAL 或 TcVnPoint2_DINT，取决于 bInteger）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	包含内参的 3x3 相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	3x3 旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 1641]&	平移矢量
bInteger	BOOL	选择函数返回的是 TcVnPoint2_DINT 还是 TcVnPoint2_REAL 容器

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.2.24 TransformCoordinatesWorldToImage (Point)

将三维世界坐标转换为二维图像坐标（使用内参和外参标定结果）。

语法

定义：

```
HRESULT TransformCoordinatesWorldToImage(
    HRESULT              hrPrev,
    TcVnPoint3_LREAL&    aSrcPoint,
    TcVnPoint2_REAL&     aDestPoint,
    TcVnMatrix3x3_LREAL& aCameraMatrix,
    TcVnArray8_LREAL&    aDistortionCoefficients,
    TcVnMatrix3x3_LREAL& aRotationMatrix,
    TcVnVector3_LREAL&   aTranslationVector
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	描述
aSrcPoint	TcVnPoint3_LREAL [▶ 1640]&	三维源点（世界坐标系）
aDestPoint	TcVnPoint2_REAL [▶ 1640]&	返回变换后的二维点（图像坐标系）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	包含内参的 3x3 相机矩阵
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&	镜头畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	3x3 旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 1641]&	平移矢量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.3 AlignRotatedImageRegion

复制提供的图像区域并将其与图像轴对齐。
可使用可用的 TwinCAT Job Task并行执行函数代码实现加速。

语法

定义：

```
HRESULT AlignRotatedImageRegion(
    HRESULT                         hrPrev,
    ITcVnImage*                     ipSrcImage,
    ITcVnImage*&                    ipDestImage,
    TcVnRotatedRectangle&           stRegion,
    ETcVnInterpolationType          eInterpolationType = IT_BILINEAR,
    ETcVnBorderInterpolationMethod eBorderInterpolation = BIM_CONSTANT,
    TcVnVector4_LREAL&              aBorderValue = {0, 0, 0, 0}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的图像。）
stRegion	TcVnRotatedRectangle [▶ 1718]&		要对齐的图像区域

名称	类型	默认值	描述
eInterpolationType	ETcVnInterpolationType [▶ 1680]	IT_BILINEAR	插值法（不支持 BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）
eBorderInterpolation	ETcVnBorderInterpolationMethod [▶ 1648]	BIM_CONSTANT	边界插值法（不支持 ISOLATED）
aBorderValue	TcVnVector4_LREAL [▶ 1641]&	{0, 0, 0, 0}	边界值（如果使用 CONSTANT）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.4 ApplyRotationToAffineTransformation

对现有的二维仿射变换矩阵进行旋转。

语法

定义：

```
HRESULT ApplyRotationToAffineTransformation(
    HRESULT              hrPrev,
    TcVnMatrix2x3_LREAL& aTransformationMatrix,
    double               fAngle,
    bool                 bUsePreMultiplication = true
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 1641]&		对其应用旋转的仿射变换矩阵
fAngle	double		以弧度为单位的角度（正值表示逆时针方向）
bUsePreMultiplication	BOOL	true	选择使用预乘还是后乘

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.5 ApplyScalingToAffineTransformation

对现有的二维仿射变换矩阵进行缩放。

语法

定义：

```
HRESULT ApplyScalingToAffineTransformation(
    HRESULT              hrPrev,
    TcVnMatrix2x3_LREAL& aTransformationMatrix,
    double               fScaleX,
    double               fScaleY,
    bool                 bUsePreMultiplication = true
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 1641]&		对其进行缩放的仿射变换矩阵
fScaleX	double		x 方向的缩放系数
fScaleY	double		y 方向的缩放系数
bUsePreMultiplication	BOOL	true	选择使用预乘还是后乘

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.6 ApplyTranslationToAffineTransformation

对现有的二维仿射变换矩阵进行平移。

语法

定义：

```
HRESULT ApplyTranslationToAffineTransformation(
    HRESULT              hrPrev,
    TcVnMatrix2x3_LREAL& aTransformationMatrix,
    double               fDeltaX,
    double               fDeltaY,
    bool                 bUsePreMultiplication = true
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 1641]&		对其应用平移的仿射变换矩阵
fDeltaX	double		x 方向的平移
fDeltaY	double		y 方向的平移
bUsePreMultiplication	BOOL	true	选择使用预乘还是后乘

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.7 ApplyYAxisInversionToAffineTransformation

对现有的二维仿射变换矩阵进行 y 轴方向的反转。

语法

定义：

```
HRESULT ApplyYAxisInversionToAffineTransformation(
    HRESULT              hrPrev,
    TcVnMatrix2x3_LREAL& aTransformationMatrix,
    bool                 bUsePreMultiplication = true
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	默认值	描述
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 1641]&		对其进行 y 轴方向反转的仿射变换矩阵
bUsePreMultiplication	BOOL	true	选择使用预乘还是后乘

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.8 ConvertCartesianToPolarAngleImage

将笛卡尔坐标（x、y）转换为极角。

语法

定义：

```
HRESULT ConvertCartesianToPolarAngleImage(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImageX,
    ITcVnImage*  ipSrcImageY,
    ITcVnImage*& ipDestImageAng,
    bool         bAngleInDegrees = false
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImageX	ITcVnImage* [▶ 1863]		包含 x 值的源图像（REAL 或 LREAL 类型元素）
ipSrcImageY	ITcVnImage* [▶ 1863]		包含 y 值的源图像（与 ipSrcImageX 的元素类型相同）
ipDestImageAng	ITcVnImage* [▶ 1863]&		包含角度的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
bAngleInDegrees	BOOL	false	指定角度的单位是度还是弧度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.9 ConvertCartesianToPolarAngles

将笛卡尔坐标（x、y）转换为极角。

语法

定义：

```
HRESULT ConvertCartesianToPolarAngles(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcPointsCart,
    ITcVnContainer*& ipDestAngles,
    bool             bAngleInDegrees = false
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcPointsCart	ITcVnContainer* [▶ 1827]		包含笛卡尔点的容器（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL）
ipDestAngles	ITcVnContainer* [▶ 1827]&		返回一个以角度填充的容器（如果源点的类型为 TcVnPoint2_DINT 或 TcVnPoint2_REAL，则返回 ContainerType_Vector_REAL；否则返回 ContainerType_Vector_LREAL。重复使用非零接口指针。）
bAngleInDegrees	BOOL	false	指定角度的单位是度还是弧度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.10 ConvertCartesianToPolarImages

将笛卡尔坐标（x、y）转换为极坐标（幅值、角度）。

语法

定义：

```
HRESULT ConvertCartesianToPolarImages(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImageX,
    ITcVnImage*  ipSrcImageY,
    ITcVnImage*& ipDestImageMag,
    ITcVnImage*& ipDestImageAng,
    bool         bAngleInDegrees = false
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImageX	ITcVnImage* [▶ 1863]		包含 x 值的源图像（REAL 或 LREAL 类型元素）
ipSrcImageY	ITcVnImage* [▶ 1863]		包含 y 值的源图像（与 ipSrcImageX 的元素类型相同）
ipDestImageMag	ITcVnImage* [▶ 1863]&		包含幅值的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
ipDestImageAng	ITcVnImage* [▶ 1863]&		包含角度的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
bAngleInDegrees	BOOL	false	指定角度的单位是度还是弧度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.11 ConvertCartesianToPolarMagnitudeImage

将笛卡尔坐标（x、y）转换为极坐标幅值。

语法

定义：

```
HRESULT ConvertCartesianToPolarMagnitudeImage(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImageX,
    ITcVnImage*  ipSrcImageY,
    ITcVnImage*& ipDestImageMag
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImageX	ITcVnImage* [▶ 1863]	包含 x 值的源图像（REAL 或 LREAL 类型元素）
ipSrcImageY	ITcVnImage* [▶ 1863]	包含 y 值的源图像（与 ipSrcImageX 的元素类型相同）
ipDestImageMag	ITcVnImage* [▶ 1863]&	包含幅值的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.12 ConvertCartesianToPolarMagnitudes

将笛卡尔坐标（x、y）转换为极坐标幅值。

语法

定义：

```
HRESULT ConvertCartesianToPolarMagnitudes(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcPointsCart,
    ITcVnContainer*& ipDestMags
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcPointsCart	ITcVnContainer* [▶ 1827]	包含笛卡尔点的容器（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL）
ipDestMags	ITcVnContainer* [▶ 1827]&	返回一个以幅值填充的容器（如果源点的类型是 TcVnPoint2_DINT 或 TcVnPoint2_REAL，则返回 ContainerType_Vector_REAL；否则返回 ContainerType_Vector_LREAL。重复使用非零接口指针。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.13 ConvertCartesianToPolarPoints

将笛卡尔坐标（x、y）转换为极坐标（幅值、角度）。

语法

定义：

```
HRESULT ConvertCartesianToPolarPoints(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcPointsCart,
    ITcVnContainer*& ipDestPointsPolar,
    bool             bAngleInDegrees = false
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	默认值	描述
ipSrcPointsCart	ITcVnContainer* [▶ 1827]		包含笛卡尔点的容器（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL）
ipDestPointsPolar	ITcVnContainer* [▶ 1827]&		返回一个以极坐标转换后的点填充的容器（如果源点的类型是 TcVnPoint2_DINT 或 TcVnPoint2_REAL，则返回 ContainerType_Vector_TcVnPoint2_REAL；否则返回 ContainerType_Vector_TcVnPoint2_LREAL。重复使用非零接口指针。）
bAngleInDegrees	BOOL	false	指定角度的单位是度还是弧度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.14 ConvertMaps

在不同映射类型之间进行转换，通常从浮点转换为定点，以加速 RemapImage。支持的映射对包括（map1 = ET_REAL，2 通道；map2 = 0）、（map1 = ET_REAL，1 通道；map2 = ET_REAL，1 通道）、（map1 = ET_INT，2 通道；map2 = ET_UINT，1 通道）、（map1 = ET_INT，2 通道；map2 = 0）。

语法

定义：

```
HRESULT ConvertMaps(
    HRESULT           hrPrev,
    ITcVnImage*       ipSrcMap1,
    ITcVnImage*       ipSrcMap2,
    ITcVnImage*&      ipDestMap1,
    ITcVnImage*&      ipDestMap2,
    ETcVnElementType  eDestMap1Type,
    USHORT            nDestMap1Channels
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcMap1	ITcVnImage* [▶ 1863]	源映射 1（ET_REAL 带 2 通道（包含 mapX、mapY），或 ET_REAL 带 1 通道（包含 mapX），或 ET_INT 带 2 通道（包含取整后的 mapX、mapY））
ipSrcMap2	ITcVnImage* [▶ 1863]	源映射 2（ET_REAL 带 1 通道（包含 mapY），或 ET_UINT 带 1 通道（如果 ipSrcMap1 为 ET_INT 带 2 通道，则包含插值表系数），如果不需要，则设为 0。
ipDestMap1	ITcVnImage* [▶ 1863]&	目标映射 1（不得与 ipSrcMap1 或 ipSrcMap2 相同！如有需要，将创建适当的目标图像。）
ipDestMap2	ITcVnImage* [▶ 1863]&	目标映射 2（不得与 ipSrcMap1 或 ipSrcMap2 相同！如果不需要，即如果 ipDestMap1 为 ET_REAL 带 2 通道，则设为 0。否则，如有需要，将创建适当的目标图像。）
eDestMap1Type	ETcVnElementType [▶ 1674]	生成的 ipDestMap1 的请求类型（ET_REAL 或 ET_INT）。ipDestMap2 的类型将自动确定，即 ET_REAL 或 ET_UINT 或无。
nDestMap1Channels	USHORT	生成的 ipDestMap1 的请求通道数（1 或 2，如果 eDestMap1Type 为 ET_INT，则必须为 2）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.2.3.14.15 ConvertPolarToCartesianImages

将极坐标（幅值、角度）转换为笛卡尔坐标（x、y）。

语法

定义：

```
HRESULT ConvertPolarToCartesianImages(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImageMag,
    ITcVnImage*  ipSrcImageAng,
    ITcVnImage*& ipDestImageX,
    ITcVnImage*& ipDestImageY,
    bool         bAngleInDegrees = false
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImageMag	ITcVnImage* [▶ 1863]		包含幅值的源图像（REAL 或 LREAL 类型元素）
ipSrcImageAng	ITcVnImage* [▶ 1863]		包含角度的源图像（与 ipSrcImageMag 的元素类型相同）
ipDestImageX	ITcVnImage* [▶ 1863]&		包含 x 值的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
ipDestImageY	ITcVnImage* [▶ 1863]&		包含 y 值的目标图像（与源图像的元素类型相同。如有需要，将创建适当的目标图像。）
bAngleInDegrees	BOOL	false	指定角度的单位是度还是弧度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.16 ConvertPolarToCartesianPoints

将极坐标（幅值、角度）转换为笛卡尔坐标（x、y）。

语法

定义：

```
HRESULT ConvertPolarToCartesianPoints(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSrcPointsPolar,
    ITcVnContainer*& ipDestPointsCart,
    bool             bAngleInDegrees = false
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcPointsPolar	ITcVnContainer* [▶ 1827]		包含极坐标的容器（ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL）
ipDestPointsCart	ITcVnContainer* [▶ 1827]&		返回以笛卡尔坐标转换后的点填充的容器（与 ipSrcPointsPolar 的类型 ID 相同。重复使用非零接口指针。）
bAngleInDegrees	BOOL	false	指定角度的单位是度还是弧度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.17 DecomposeAffineTransformation

分解仿射变换矩阵，并返回以弧度为单位的旋转角（顺时针）、平移向量（像素）、缩放比例和倾斜向量（以弧度为单位的角）。分解法假定变换顺序如下：平移、旋转、缩放和倾斜。

语法

定义：

```
HRESULT DecomposeAffineTransformation(
    HRESULT              hrPrev,
    TcVnMatrix2x3_LREAL& aAffineTransform,
    double&              fRotationAngle,
    TcVnVector2_LREAL&   aTranslationVector = {0, 0},
    TcVnVector2_LREAL&   aScaleVector = {0, 0},
    TcVnVector2_LREAL&   aSkewVector = {0, 0}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aAffineTransform	TcVnMatrix2x3_LREAL [▶ 1641]&		仿射变换矩阵
fRotationAngle	double&		顺时针方向的旋转角（单位：弧度）。旋转中心为原点（0，0）
aTranslationVector	TcVnVector2_LREAL [▶ 1641]&	{0, 0}	二维平移向量。数值单位为像素
aScaleVector	TcVnVector2_LREAL [▶ 1641]&	{0, 0}	二维比例向量
aSkewVector	TcVnVector2_LREAL [▶ 1641]&	{0, 0}	二维倾斜向量。数值是 x 和 y 方向的倾斜角，单位为弧度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.18 DecomposeHomography

分解单应性矩阵，并验证旋转、平移和平面法线的四个解。它利用相机前的可见参考点来确认四个解中有哪几个解（最多两个）与所有参考点一致。

语法

定义：

```
HRESULT DecomposeHomography(
    HRESULT              hrPrev,
    TcVnMatrix3x3_LREAL& aHomography,
    TcVnMatrix3x3_LREAL& aCameraMatrix,
    TcVnMatrix3x3_LREAL& aRotationMatrixA,
    TcVnVector3_LREAL&   aTranslationVectorA,
    TcVnVector3_LREAL&   aNormVectorA,
    TcVnMatrix3x3_LREAL& aRotationMatrixB,
    TcVnVector3_LREAL&   aTranslationVectorB,
    TcVnVector3_LREAL&   aNormVectorB,
    ULONG&               nSolutions,
    ITcVnContainer*      ipPointsSrc = nullptr,
    ITcVnContainer*      ipPointsDes = nullptr,
    ITcVnContainer*      ipInlierMask = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aHomography	TcVnMatrix3x3_LREAL [▶ 1641]&		单应性矩阵（两个平面之间的透视变换）
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&		相机矩阵
aRotationMatrixA	TcVnMatrix3x3_LREAL [▶ 1641]&		第一个解的旋转矩阵（Ra）
aTranslationVectorA	TcVnVector3_LREAL [▶ 1641]&		第一个解的平移向量（Ta）
aNormVectorA	TcVnVector3_LREAL [▶ 1641]&		第一个解的范数向量（Na）
aRotationMatrixB	TcVnMatrix3x3_LREAL [▶ 1641]&		第二个解的旋转矩阵（Rb）
aTranslationVectorB	TcVnVector3_LREAL [▶ 1641]&		第二个解的平移向量（Tb）
aNormVectorB	TcVnVector3_LREAL [▶ 1641]&		第二个解的范数向量（Nb）
nSolutions	ULONG&		返回潜在真解的数量 [0、1 或 2]。 0：没有确认给定相应点的潜在真解。 1：确认解（Ra、Ta、Na）。 2：解（Ra、Ta、Na）和（Rb、Tb、Nb）均得到确认。 在所有情况下，函数都会返回两个解（Ra、Ta、Nb）和（Rb、Tb、Nb）。第三个和第四个解的计算方法如下：（Ra、-Ta、-Na）和（Rb、-Tb、-Nb）。
ipPointsSrc	ITcVnContainer* [▶ 1827]	nullptr	包含源点的容器（ContainerType_Vector_TcVnPoint2_REAL）
ipPointsDes	ITcVnContainer* [▶ 1827]	nullptr	包含目标点的容器（数量与 ipPoints1 相同，ContainerType_Vector_TcVnPoint2_REAL）。
ipInlierMask	ITcVnContainer* [▶ 1827]	nullptr	标记内群值的掩码（可选，如果不可用，则设为 0；ContainerType_Vector_SINT 或 ContainerType_Vector_USINT）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.19 FlipImage

翻转图像（镜像并移回之前的坐标）。

语法

定义：

```
HRESULT FlipImage(
    HRESULT       hrPrev,
    ITcVnImage*   ipSrcImage,
    ITcVnImage*&  ipDestImage,
    ETcVnFlipAxis eFlipAxis
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
eFlipAxis	ETcVnFlipAxis [▶ 1678]	选择绕其翻转图像的轴线

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.20 GenerateAffineTransformationUnitMatrix2D

生成仿射变换二维单位矩阵。

语法

定义：

```
HRESULT GenerateAffineTransformationUnitMatrix2D(
    HRESULT               hrPrev,
    TcVnMatrix2x3_LREAL& aTransformationMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 1641]&	结果仿射变换单位矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.21 GetAffineTransformation

计算三个相应点对之间的仿射变换。这些点标记相应三角形的角。

语法

定义：

```
HRESULT GetAffineTransformation(
    HRESULT                hrPrev,
    TcVnArray3_Point2_REAL& aSrcPoints,
    TcVnArray3_Point2_REAL& aDestPoints,
    TcVnMatrix2x3_LREAL&    aTransformationMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aSrcPoints	TcVnArray3_Point2_REAL [▶ 1641]&	源点
aDestPoints	TcVnArray3_Point2_REAL [▶ 1641]&	目标点
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 1641]&	结果变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.22 GetAffineTransformation2D

估算两个平面点集之间的仿射变换。

语法

定义：

```
HRESULT GetAffineTransformation2D(
    HRESULT                  hrPrev,
    ITcVnContainer*          ipSrcPoints,
    ITcVnContainer*          ipDestPoints,
    TcVnMatrix2x3_LREAL&     aAffineTransform,
    ETcVnEstimationAlgorithm eAlgorithm = EA_RANSAC,
    double                   fReprojThreshold = 3,
    ITcVnContainer**         pipInlierMask = nullptr,
    ULONG                    nMaxIterations = 2000,
    double                   fConfidence = 0.995,
    ULONG                    nRefineIters = 10
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcPoints	ITcVnContainer* [▶ 1827]		至少包含 3 个源点的容器（ContainerType_Vector_TcVnPoint2_REAL）
ipDestPoints	ITcVnContainer* [▶ 1827]		包含目标点的容器（数量与 ipSrcPoints 相同，ContainerType_Vector_TcVnPoint2_REAL）
aAffineTransform	TcVnMatrix2x3_LREAL [▶ 1641]&		返回将源点转换为目标点的仿射变换矩阵
eAlgorithm	ETcVnEstimationAlgorithm [▶ 1674]	EA_RANSAC	估计算法（仅支持 RANSAC 和 LMEDS）
fReprojThreshold	double	3	将点对视为内群时允许的最大重投影误差
pipInlierMask	ITcVnContainer* [▶ 1827]*	nullptr	返回标记内群值的掩码（可选，如果不需要，则设为 0；ContainerType_Vector_USINT）
nMaxIterations	ULONG	2000	最大迭代次数
fConfidence	double	0.995	置信度（0..1）

名称	类型	默认值	描述
nRefineIters	ULONG	10	Levenberg-Marquardt 算法进一步改进仿射变换（仅使用内群值）的最大迭代次数。设为 0 则禁用改进。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.23 GetPerspectiveTransformation

计算四个对应点对之间的透视变换。这些点标记相应矩形的角。

语法

定义：

```
HRESULT GetPerspectiveTransformation(
    HRESULT                hrPrev,
    TcVnArray4_Point2_REAL& aSrcPoints,
    TcVnArray4_Point2_REAL& aDestPoints,
    TcVnMatrix3x3_LREAL&    aTransformationMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aSrcPoints	TcVnArray4_Point2_REAL [▶ 1641]&	源点
aDestPoints	TcVnArray4_Point2_REAL [▶ 1641]&	目标点
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	结果变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.24 Homography

找出两个平面点集之间的单应性（透视变换）。

语法

定义：

```
HRESULT Homography(
    HRESULT                 hrPrev,
    ITcVnContainer*         ipSrcPoints,
    ITcVnContainer*         ipDestPoints,
    TcVnMatrix3x3_LREAL&    aPerspectiveTransform,
    ETcVnEstimationAlgorithm eAlgorithm = EA_DEFAULT,
    double                  fReprojThreshold = 3,
    ITcVnContainer**        pipInlierMask = nullptr,
    ULONG                   nMaxIterations = 2000,
    double                  fConfidence = 0.995
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcPoints	ITcVnContainer* [▶ 1827]		至少包含 4 个源点的容器（ContainerType_Vector_TcVnPoint2_REAL）
ipDestPoints	ITcVnContainer* [▶ 1827]		包含目标点的容器（数量与 ipSrcPoints 相同，ContainerType_Vector_TcVnPoint2_REAL）
aPerspectiveTransform	TcVnMatrix3x3_LREAL [▶ 1641]&		返回将源点转换为目标点的透视变换矩阵
eAlgorithm	ETcVnEstimationAlgorithm [▶ 1674]	EA_DEFAULT	估计算法
fReprojThreshold	double	3	将点对视为内群时允许的最大重投影误差（仅适用于 RANSAC 和 RHO）
pipInlierMask	ITcVnContainer* [▶ 1827]*	nullptr	返回标记内群值的掩码（可选，如果不需要，则设为 0；ContainerType_Vector_SINT；仅适用于 RANSAC 和 LMEDS）
nMaxIterations	ULONG	2000	最大 RANSAC 迭代次数
fConfidence	double	0.995	置信度（0..1）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.25 InvertAffineTransformation

反转二维仿射变换矩阵。如果在 F_VN_WarpAffine 函数中使用了逆变换，则逆变换会补偿图像上的仿射变换。

语法

定义：

```
HRESULT InvertAffineTransformation(
    HRESULT              hrPrev,
    TcVnMatrix2x3_LREAL& aTransformationMatrix,
    TcVnMatrix2x3_LREAL& aInvertedTransformationMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 1641]&	源仿射变换矩阵
aInvertedTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 1641]&	结果逆仿射变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.26 PerspectiveTransformation

对三维点坐标进行透视变换（旋转 + 平移，例如从外参标定）。

语法

定义：

```
HRESULT PerspectiveTransformation(
    HRESULT             hrPrev,
    ITcVnContainer*     ipSrcPoints,
    ITcVnContainer*&    ipDestPoints,
    TcVnMatrix3x3_LREAL& aRotationMatrix,
    TcVnVector3_LREAL&   aTranslationVector
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcPoints	ITcVnContainer* [▶ 1827]	包含三维源点的容器（TcVnPoint3_REAL 或 TcVnPoint3_LREAL）
ipDestPoints	ITcVnContainer* [▶ 1827]&	返回变换后的点（与 ipSrcPoints 类型相同）
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	3x3 旋转矩阵
aTranslationVector	TcVnVector3_LREAL [▶ 1641]&	平移矢量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.27 PyramidDown

将图像向下采样为一半的宽度和一半的高度。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT PyramidDown(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.28 PyramidUp

将图像向上采样为双倍宽度和双倍高度。

语法

定义：

```
HRESULT PyramidUp(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.29 RemapImage

使用自定义映射对图像进行重映射，即 ipDestImage(x,y) = ipSrcImage(mapX(x,y), mapY(x,y))。支持的映射对包括（ipMap1 = ET_REAL，2 通道；ipMap2 = 0）、（ipMap1 = ET_REAL，1 通道；ipMap2 = ET_REAL，1 通道）、（ipMap1 = ET_INT，2 通道；ipMap2 = ET_UINT，1 通道）、（ipMap1 = ET_INT，2 通道；ipMap2 = 0）。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT RemapImage(
    HRESULT                         hrPrev,
    ITcVnImage*                     ipSrcImage,
    ITcVnImage*&                    ipDestImage,
    ITcVnImage*                     ipMap1,
    ITcVnImage*                     ipMap2,
    bool                            bRelativeMap = false,
    ETcVnInterpolationType          eInterpolationType = IT_BILINEAR,
    ETcVnBorderInterpolationMethod eBorderInterpolation = BIM_CONSTANT,
    TcVnVector4_LREAL&              aBorderValue = {0, 0, 0, 0}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（不得与 ipSrcImage 相同！与 ipMap1 尺寸相同，与 ipSrcImage 类型相同。如有需要，将创建适当的目标图像。）
ipMap1	ITcVnImage* [▶ 1863]		映射 1（ET_REAL 带 2 通道（包含 mapX、mapY），或 ET_REAL 带 1 通道（包含 mapX），或 ET_INT 带 2 通道（包含取整后的 mapX、mapY））
ipMap2	ITcVnImage* [▶ 1863]		映射 2（ET_REAL 带 1 通道（包含 mapY），或 ET_UINT 带 1 通道（如果 ipMap1 为 ET_INT 带 2 通道，则包含插值表系数），如果不需要，则设为 0。
bRelativeMap	bool	false	如果为 true，则映射包含相对坐标，即 ipDestImage(x,y) = ipSrcImage(x + mapX(x,y), y + mapY(x,y))。
eInterpolationType	ETcVnInterpolationType [▶ 1680]	IT_BILINEAR	插值类型（不支持 AREA_BASED、BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）
eBorderInterpolation	ETcVnBorderInterpolationMethod [▶ 1648]	BIM_CONSTANT	边界插值法（不支持 ISOLATED）
aBorderValue	TcVnVector4_LREAL [▶ 1641]&	{0, 0, 0, 0}	边界值（如果使用 CONSTANT）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.2.3.14.30 RemapImageToLogPolarSpace

将图像重映射到对数极空间。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT RemapImageToLogPolarSpace(
    HRESULT                hrPrev,
    ITcVnImage*            ipSrcImage,
    ITcVnImage*&           ipDestImage,
    TcVnPoint2_REAL&       aCenter,
    double                 fScale = 0,
    ETcVnInterpolationType eInterpolationType = IT_BILINEAR,
    LONG                   nDestWidth = -1,
    LONG                   nDestHeight = -1
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（不得与 ipSrcImage 相同！如有需要，将创建适当的目标图像。）
aCenter	TcVnPoint2_REAL [▶ 1640]&		变换中心点
fScale	double	0	幅值标度参数，即图像宽度/ln（半径）（自动选择则设为 0）
eInterpolationType	ETcVnInterpolationType [▶ 1680]	IT_BILINEAR	插值类型（不支持 BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）
nDestWidth	LONG	-1	目标图像宽度（设为 -1 表示源图像宽度，0 表示自动缩放至转换的半径，> 0 表示用户自定义的宽度）
nDestHeight	LONG	-1	目标图像高度（设为 -1 表示源图像高度，0 表示自动缩放至转换的半径 * PI，> 0 表示用户自定义的高度）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.31 RemapImageToPolarSpace

将图像重新映射到极坐标空间。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT RemapImageToPolarSpace(
    HRESULT                 hrPrev,
    ITcVnImage*             ipSrcImage,
    ITcVnImage*&            ipDestImage,
    TcVnPoint2_REAL&        aCenter,
    double                  fMaxRadius = 0,
    ETcVnInterpolationType eInterpolationType = IT_BILINEAR,
    LONG                    nDestWidth = -1,
    LONG                    nDestHeight = -1
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（不得与 ipSrcImage 相同！如有需要，将创建适当的目标图像。）
aCenter	TcVnPoint2_REAL [▶ 1640]&		变换中心点
fMaxRadius	double	0	变换的最大半径（自动选择则设为 0）
eInterpolationType	ETcVnInterpolationType [▶ 1680]	IT_BILINEAR	插值类型（不支持 BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）
nDestWidth	LONG	-1	目标图像宽度（设为 -1 表示源图像宽度，0 表示自动缩放至转换的半径，> 0 表示用户自定义的宽度）

名称	类型	默认值	描述
nDestHeight	LONG	-1	目标图像高度（设为 -1 表示源图像高度，0 表示自动缩放至转换的半径 * PI，> 0 表示用户自定义的高度）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.32 ResizeImage

使用特定的插值类型调整图像大小，同时保持其长宽比。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT ResizeImage(
    HRESULT                hrPrev,
    ITcVnImage*            ipSrcImage,
    ITcVnImage*&           ipDestImage,
    ULONG                  nWidth,
    ULONG                  nHeight,
    ETcVnInterpolationType eInterpolationType,
    ETcVnPaddingMode       ePaddingMode = PM_NONE,
    TcVnVector4_LREAL&     aBorderValue = {0, 0, 0, 0}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的图像。）
nWidth	ULONG		新宽度
nHeight	ULONG		新高度
eInterpolationType	ETcVnInterpolationType [▶ 1680]		插值类型（SINT 或 DINT 类型的图像仅支持 NEAREST_NEIGHBOR）
ePaddingMode	ETcVnPaddingMode [▶ 1686]	PM_NONE	图像填充模式

名称	类型	默认值	描述
aBorderValue	TcVnVector4_LREAL [▶ 1641]&	{0, 0, 0, 0}	边界值，如果使用 TCVN_PM_LETTERBOX

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.33 ResizeRegion (TcVnRectangle_DINT)

按给定矩形当前大小的比例和/或通过显式的绝对偏移量来扩展或缩小该矩形。水平和垂直缩放比定义了按比例的尺寸变化。如果缩放比等于 1.0，则仅由相应的最小偏移量决定尺寸变化。如果提供了参考图像，则生成的矩形将被裁剪到图像边界内。如果发生了裁剪，则函数返回 S_FALSE。

语法

定义：

```
HRESULT ResizeRegion(
    HRESULT              hrPrev,
    TcVnRectangle_DINT& stSrcRect,
    TcVnRectangle_DINT& stDestRect,
    float                fWidthRatio,
    float                fHeightRatio,
    LONG                 nMinWidthOffset,
    LONG                 nMinHeightOffset,
    ITcVnImage*          ipImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stSrcRect	TcVnRectangle_DINT [▶ 1717]&	要扩展或收缩的源矩形。
stDestRect	TcVnRectangle_DINT [▶ 1717]&	扩展或收缩后生成的矩形。
fWidthRatio	float	水平缩放比。如果 fWidthRatio <> 1（> 1 表示扩展，< 1 表示收缩），则 nMinWidthOffset 的绝对值可用作最小宽度变化约束。如果 fWidthRatio = 1.0，则宽度变化仅由 nMinWidthOffset 定义。
fHeightRatio	float	垂直缩放比。如果 fHeightRatio <> 1（> 1 表示扩展，< 1 表示收缩），则 nMinHeightOffset 的绝对值可用作最小宽度变化约束。如果 fHeightRatio = 1.0，则高度变化仅由 nMinHeightOffset 定义。

名称	类型	描述
nMinWidthOffset	LONG	最小宽度偏移量（以像素为单位），围绕矩形中心对称应用。如果 fWidthRatio <> 1.0，则 nMinWidthOffset 的绝对值定义了最小宽度变化。如果 fWidthRatio = 1.0，则符号定义了操作方向（> 0 表示扩展，< 0 表示收缩，= 0 表示宽度不变）。
nMinHeightOffset	LONG	最小高度偏移量（以像素为单位），围绕矩形中心对称应用。如果 fHeightRatio <> 1.0，则 nMinHeightOffset 的绝对值定义了最小高度变化。如果 fHeightRatio = 1.0，则符号定义了操作方向（> 0 表示扩展，< 0 表示收缩，= 0 表示高度不变）。
ipImage	ITcVnImage* [▶ 1863]	可选参考图像，用于定义裁剪边界，如果不需要，则设为 0。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

6.2.3.14.34 RotateImage

按顺时针方向将图像旋转 90、180 或 270 度。

语法

定义：

```
HRESULT RotateImage(
    HRESULT            hrPrev,
    ITcVnImage*        ipSrcImage,
    ITcVnImage*&       ipDestImage,
    ETcVnRotationAngle eAngle
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的图像。）
eAngle	ETcVnRotationAngle [▶ 1689]	旋转角

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.35 RotateImage (Exp)

按单个旋转角度（以度为单位）旋转图像。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT RotateImage(
    HRESULT                          hrPrev,
    ITcVnImage*                      ipSrcImage,
    ITcVnImage*&                     ipDestImage,
    double                           fAngleDeg,
    bool                             bAdjustDestSize,
    ETcVnInterpolationType           eInterpolationType = IT_BILINEAR,
    ETcVnBorderInterpolationMethod eBorderInterpolation = BIM_CONSTANT,
    TcVnVector4_LREAL&               aBorderValue = {0, 0, 0, 0}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的图像。）
fAngleDeg	double		旋转角（以度为单位）（正：逆时针，负：顺时针）
bAdjustDestSize	bool		如果为 true，将调整 ipDestImage 的大小，从而包含整个旋转后的 ipSrcImage。否则，ipDestImage 大小将设置为 ipSrcImage 大小。
eInterpolationType	ETcVnInterpolationType [▶ 1680]	IT_BILINEAR	插值法（不支持 BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）
eBorderInterpolation	ETcVnBorderInterpolationMethod [▶ 1648]	BIM_CONSTANT	边界插值法（不支持 ISOLATED）
aBorderValue	TcVnVector4_LREAL [▶ 1641]&	{0, 0, 0, 0}	边界值（如果使用 CONSTANT）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.36 WarpAffine

对图像进行仿射变换。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被看门狗取消时，可返回部分结果。

语法

定义：

```
HRESULT WarpAffine(
    HRESULT              hrPrev,
    ITcVnImage*          ipSrcImage,
    ITcVnImage*&         ipDestImage,
    TcVnMatrix2x3_LREAL& aTransformationMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的图像。）
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 1641]&	仿射变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.37 WarpAffine (Container)

对二维点容器进行仿射变换。

语法

定义：

```
HRESULT WarpAffine(
    HRESULT              hrPrev,
    ITcVnContainer*      ipSrcPoints,
    ITcVnContainer*&     ipDestPoints,
    TcVnMatrix2x3_LREAL& aTransformationMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcPoints	ITcVnContainer* [▶ 1827]	源点（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL）
ipDestPoints	ITcVnContainer* [▶ 1827]&	返回变换后的点（与 ipSrcPoints 的类型 ID 相同；重复使用非零接口指针。）
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 1641]&	仿射变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.38 WarpAffine (Point)

对二维点进行仿射变换。

语法

定义：

```
HRESULT WarpAffine(
    HRESULT              hrPrev,
    TcVnPoint2_LREAL&    aSrcPoint,
    TcVnPoint2_LREAL&    aDestPoint,
    TcVnMatrix2x3_LREAL& aTransformationMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aSrcPoint	TcVnPoint2_LREAL [▶ 1640]&	源点
aDestPoint	TcVnPoint2_LREAL [▶ 1640]&	目标点
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 1641]&	仿射变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.39 WarpAffine (Rectangle)

使用仿射变换来扭曲矩形。

语法

定义：

```
HRESULT WarpAffine(
    HRESULT              hrPrev,
    ULONG                nTopLeftX,
    ULONG                nTopLeftY,
    ULONG                nBottomRightX,
    ULONG                nBottomRightY,
    ITcVnContainer*&     ipDestPoints,
    TcVnMatrix2x3_LREAL& aTransformationMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nTopLeftX	ULONG	左上角的 x 坐标
nTopLeftY	ULONG	左上角的 y 坐标
nBottomRightX	ULONG	右下角的 x 坐标
nBottomRightY	ULONG	右下角的 y 坐标
ipDestPoints	ITcVnContainer* [▶ 1827]&	返回 4 个变换后的点（ContainerType_Vector_TcVnPoint2_REAL）。
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 1641]&	仿射变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.40 WarpAffine (Exp)

对图像进行仿射变换。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT WarpAffine(
    HRESULT                         hrPrev,
    ITcVnImage*                     ipSrcImage,
    ITcVnImage*&                    ipDestImage,
    TcVnMatrix2x3_LREAL&            aTransformationMatrix,
    ULONG                           nDestWidth,
    ULONG                           nDestHeight,
    ETcVnInterpolationType          eInterpolationType,
    bool                            bWarpInverse,
    ETcVnBorderInterpolationMethod eBorderInterpolation,
    double                          fBorderValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的图像。）
aTransformationMatrix	TcVnMatrix2x3_LREAL [▶ 1641]&	仿射变换矩阵
nDestWidth	ULONG	目标图像的宽度
nDestHeight	ULONG	目标图像的高度
eInterpolationType	ETcVnInterpolationType [▶ 1680]	插值法（不支持 BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）
bWarpInverse	bool	将 aTransformationMatrix 作为逆变换处理
eBorderInterpolation	ETcVnBorderInterpolationMethod [▶ 1648]	边界插值法（不支持 ISOLATED）
fBorderValue	double	边界值（如果使用 CONSTANT）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.41 WarpPerspective

对图像进行透视变换。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
在看门狗取消时，可返回部分结果。

语法

定义：

```
HRESULT WarpPerspective(
    HRESULT              hrPrev,
    ITcVnImage*          ipSrcImage,
    ITcVnImage*&         ipDestImage,
    TcVnMatrix3x3_LREAL& aTransformationMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的图像。）
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	透视变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.42 WarpPerspective (Container)

对二维点容器进行透视变换。

语法

定义：

```
HRESULT WarpPerspective(
    HRESULT              hrPrev,
    ITcVnContainer*      ipSrcPoints,
    ITcVnContainer*&     ipDestPoints,
    TcVnMatrix3x3_LREAL& aTransformationMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcPoints	ITcVnContainer* [▶ 1827]	源点（ContainerType_Vector_TcVnPoint2_DINT 或 ContainerType_Vector_TcVnPoint2_REAL 或 ContainerType_Vector_TcVnPoint2_LREAL）
ipDestPoints	ITcVnContainer* [▶ 1827]&	返回变换后的点（与 ipSrcPoints 的类型 ID 相同；重复使用非零接口指针。）
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	透视变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.43 WarpPerspective (Point)

对二维点进行透视变换。

语法

定义：

```
HRESULT WarpPerspective(
    HRESULT              hrPrev,
    TcVnPoint2_LREAL&    aSrcPoint,
    TcVnPoint2_LREAL&    aDestPoint,
    TcVnMatrix3x3_LREAL& aTransformationMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aSrcPoint	TcVnPoint2_LREAL [▶ 1640]&	源点
aDestPoint	TcVnPoint2_LREAL [▶ 1640]&	目标点
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	透视变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.44 WarpPerspective (Rectangle)

使用透视变换来扭曲矩形。

语法

定义：

```
HRESULT WarpPerspective(
    HRESULT              hrPrev,
    ULONG                nTopLeftX,
    ULONG                nTopLeftY,
    ULONG                nBottomRightX,
    ULONG                nBottomRightY,
    ITcVnContainer*&     ipDestPoints,
    TcVnMatrix3x3_LREAL& aTransformationMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nTopLeftX	ULONG	左上角的 x 坐标
nTopLeftY	ULONG	左上角的 y 坐标
nBottomRightX	ULONG	右下角的 x 坐标
nBottomRightY	ULONG	右下角的 y 坐标
ipDestPoints	ITcVnContainer* [▶ 1827]&	返回 4 个变换后的点（ContainerType_Vector_TcVnPoint2_REAL）。
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	透视变换矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.14.45 WarpPerspective (Exp)

对图像进行透视变换
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT WarpPerspective(
    HRESULT                         hrPrev,
    ITcVnImage*                     ipSrcImage,
    ITcVnImage*&                    ipDestImage,
    TcVnMatrix3x3_LREAL&            aTransformationMatrix,
    ULONG                           nDestWidth,
    ULONG                           nDestHeight,
    ETcVnInterpolationType          eInterpolationType,
    bool                            bWarpInverse,
    ETcVnBorderInterpolationMethod eBorderInterpolation,
    double                          fBorderValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的图像。）
aTransformationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	透视变换矩阵
nDestWidth	ULONG	目标图像的宽度
nDestHeight	ULONG	目标图像的高度
eInterpolationType	ETcVnInterpolationType [▶ 1680]	插值法（不支持 BILINEAR_EXACT 和 NEAREST_NEIGHBOR_EXACT）
bWarpInverse	bool	将 aTransformationMatrix 作为逆变换处理
eBorderInterpolation	ETcVnBorderInterpolationMethod [▶ 1648]	边界插值法（不支持 ISOLATED）
fBorderValue	double	边界值（如果使用 CONSTANT）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15 图像分析

该组包含从图像中提取信息的函数。

函数

对象检测 [▶ 2414]

- 轮廓搜索
- 通过 Hough 变换查找对象
- 通过图像对比进行匹配

统计图像功能

- ImageAverage [▶ 2436]
- ImageAverageStdDev [▶ 2437]
- ImageAbsoluteAverage [▶ 2443]
- ImageMedian [▶ 2439]
- MaxPixelValue [▶ 2440]
- MinPixelValue [▶ 2441]

边缘检测

- CannyEdgeDetection [▶ 2430]

查找连续图像区域

- ConnectedComponents [▶ 2431]
- ConnectedComponentsWithStats [▶ 2432]
- GetConnectedComponent [▶ 2435]

区域对齐

- RegionOrientation [▶ 2442]
- UprightInnerRectangle [▶ 2443]

其他

- CountNonZeroPixels [▶ 2434]
- DistanceTransformation [▶ 2434]
- ImageCenterOfMass [▶ 2438]
- ImageMoments [▶ 2439]

还请参阅有关此

- FindRois [▶ 2444]
- InnerCircle [▶ 2445]
- SalientFeatures [▶ 2446]

6.2.3.15.1 目标检测

该子组包含对象检测函数。

函数

目标检测

- DetectBlobs [▶ 2415]
- FindContours [▶ 2417]
- FindContourHierarchyExp [▶ 2416]
- F_VN_FindTemplateWalshExp [▶ 2418]
- FindTemplateWalsh_AngleRangeExp [▶ 2420]
- HoughCircles [▶ 2422]
- HoughLines [▶ 2423]
- HoughLinesP [▶ 2424]

匹配

- MatchImageHuMoments [▶ 2425]
- MatchTemplate [▶ 2426]
- MatchTemplateWalsh [▶ 2429]
- MatchTemplateAndEvaluate [▶ 2427]

其他

- AdjustActiveContour [▶ 2414]

6.2.3.15.1.1 AdjustActiveContour

调整活动轮廓（蛇形），最大限度减小其累积能量（内部和外部）。

语法

定义：

```
HRESULT AdjustActiveContour(
    HRESULT          hrPrev,
    ITcVnImage*      ipImage,
    ITcVnContainer*  ipActiveContour,
    float            fAlpha,
    float            fBeta,
    float            fGamma,
    ULONG            nWindowWidth,
    ULONG            nWindowHeight,
    ULONG            nMaxIterations,
    bool             bUseGradient
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	描述
ipImage	ITcVnImage* [▶ 1863]	源图像（1 通道，USINT 元素）
ipActiveContour	ITcVnContainer* [▶ 1827]	初始轮廓，将进行调整
fAlpha	float	连续能量系数
fBeta	float	曲率能量系数
fGamma	float	图像能量系数
nWindowWidth	ULONG	窗口宽度（3、5、7……）
nWindowHeight	ULONG	窗口高度（3、5、7……）
nMaxIterations	ULONG	最大迭代
bUseGradient	BOOL	如果为 true，则使用梯度幅值作为图像能量（否则：像素亮度）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.1.2 DetectBlobs

检测斑点轮廓。应用阈值和轮廓搜索，并提供多种选项，用于筛选找到的轮廓。可轻松设置多阈值和结果组合。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT DetectBlobs(
    HRESULT                  hrPrev,
    ITcVnImage*              ipSrcImage,
    ITcVnContainer*&         ipBlobContours,
    TcVnParamsBlobDetection& stParams,
    TcVnPoint&               aOffset = {0, 0}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT 或 REAL，1 通道或 3 通道（3 通道输入应为 RGB 并内部转换为灰色））

名称	类型	默认值	描述
ipBlobContours	ITcVnContainer* [▶ 1827]&		返回一个以所找到的轮廓填充的容器（ContainerType_Vector_Vector_TcVnPoint2_DINT；此容器的元素为单一轮廓。重复使用非零接口指针。）
stParams	TcVnParamsBlobDetection [▶ 1709]&		用于筛选检测到的轮廓的参数。
aOffset	TcVnPoint [▶ 1640]&	{0, 0}	每个轮廓点移动的偏移量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.1.3 FindContourHierarchy (Exp)

在二值图像中搜索目标轮廓，并确定其层级关系。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT FindContourHierarchy(
    HRESULT                          hrPrev,
    ITcVnImage*                      ipSrcImage,
    ITcVnContainer*&                 ipContours,
    ITcVnContainer*&                 ipHierarchy,
    ETcVnContourRetrievalMode        eRetrievalMode,
    ETcVnContourApproximationMethod eApproximationMethod,
    TcVnPoint&                       aOffset
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（1 通道，二值）
ipContours	ITcVnContainer* [▶ 1827]&	返回一个以所找到的轮廓填充的容器（ContainerType_Vector_Vector_TcVnPoint2_DINT；此容器的元素为单一轮廓。重复使用非零接口指针。）

名称	类型	描述
ipHierarchy	ITcVnContainer* [▶ 1827]&	返回一个容器，其中包含所找到轮廓的层级信息（ContainerType_Vector_TcVnVector4_DINT；该容器的元素为四维向量，这种向量包含同层级下一个轮廓 [0] 和上一个轮廓 [1]、第一个子轮廓 [2] 以及父轮廓 [3] 基于 0 的索引。重复使用非零接口指针。）
eRetrievalMode	ETcVnContourRetrievalMode [▶ 1667]	指定检索哪些轮廓以及如何编码它们之间的关系
eApproximationMethod	ETcVnContourApproximationMethod [▶ 1666]	轮廓编码
aOffset	TcVnPoint [▶ 1640]&	每个轮廓点移动的偏移量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.1.4 FindContours

在二值图像中搜索目标轮廓。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT FindContours(
    HRESULT                         hrPrev,
    ITcVnImage*                     ipSrcImage,
    ITcVnContainer*&                ipContours,
    ETcVnContourRetrievalMode       eRetrievalMode = CRM_EXTERNAL,
    ETcVnContourApproximationMethod eApproximationMethod = CAM_SIMPLE,
    TcVnPoint&                      aOffset = {0, 0}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（1 通道，二值）

名称	类型	默认值	描述
ipContours	ITcVnContainer* [▶ 1827]&		返回一个以所找到的轮廓填充的容器（ContainerType_Vector_Vector_TcVnPoint2_DINT；此容器的元素为单一轮廓。重复使用非零接口指针。）
eRetrievalMode	ETcVnContourRetrievalMode [▶ 1667]	CRM_EXTERNAL	指定检索哪些轮廓以及如何编码它们之间的关系
eApproximationMethod	ETcVnContourApproximationMethod [▶ 1666]	CAM_SIMPLE	轮廓编码
aOffset	TcVnPoint [▶ 1640]&	{0, 0}	每个轮廓点移动的偏移量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.1.5 FindTemplateWalsh

将模板图像与源图像中的每个位置进行匹配（使用Walsh变换），并评估比较结果。返回匹配的排序列表（最佳匹配优先）。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT FindTemplateWalsh(
    HRESULT                 hrPrev,
    ITcVnImage*             ipSrcImage,
    ITcVnImage*             ipTemplateImage,
    ITcVnContainer*&        ipMatches,
    float                   fMatchThreshold,
    ULONG                   nProjections,
    ULONG                   eOptions = FTWO_NONE,
    ULONG                   nRotations = 1,
    float                   fScaleFactor = 1,
    ETcVnInterpolationType  eInterpolationType = IT_BILINEAR,
    ITcVnContainer**        pipMatchValues = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT 或 SINT，1 或 3 通道）
ipTemplateImage	ITcVnImage* [▶ 1863]		模板图像（与 ipSrcImage 类型相同，宽度和高度较小）
ipMatches	ITcVnContainer* [▶ 1827]&		返回匹配位置（容器类型取决于 eOptions；ContainerType_Vector_TcVnRotatedRectangle 或 ContainerType_Vector_TcVnVector4_REAL，其中每个元素包含中心点 (x, y)、以度为单位的旋转角以及 ipTemplateImage 在 ipSrcImage 中的匹配值；容器元素按相关性排序（最佳匹配项优先））
fMatchThreshold	float		用于区分相关匹配和不相关匹配的阈值（0...1，1.0 表示完全匹配。）
nProjections	ULONG		要使用的Walsh投影数量。nProjections 值越高，匹配时考虑的细节就越多。由于最相关的信息由前几个Walsh投影所捕获，因此较少的数量（例如 5、10、25 或 50）即可满足许多使用场景。每个投影都需要额外的计算时间。因此，只要满足匹配要求，选择较少的Walsh投影数量是有利的。允许的最大 nProjections 值由模板图像的宽度和高度决定。
eOptions	ULONG	FTWO_NONE	控制匹配过程的标志。（参见 ETcVnFindTemplateWalshOptions；默认值：TCVN_FTWO_NONE）
nRotations	ULONG	1	要检查的旋转数量。（有效值：1（0°）、2（0° 和 180°）、4（0°、90°、180° 和 270°）、8（0°、45°、……、315°）、16（0°、22.5°、……、337.5°）和 32（0°、11.25°、……、348.75°）；默认值：1）

名称	类型	默认值	描述
fScaleFactor	float	1	系数（0..1]，用于减小源图像和模板图像的宽度和高度，以提高性能（但精度降低！）
eInterpolationType	ETcVnInterpolationType [▶ 1680]	IT_BILINEAR	图像大小调整插值类型（仅在 fScaleFactor != 1 时使用，建议大多数情况下使用 TCVN_IT_BILINEAR）
pipMatchValues	ITcVnContainer* [▶ 1827]*	nullptr	可选择返回匹配值（ContainerType_Vector_REAL，大小和排序与 ipMatches 相同。如果不需要，则设为 0。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

6.2.3.15.1.6 FindTemplateWalsh (AngleRange)

将模板图像与源图像中的每个位置进行匹配（使用Walsh变换），并评估比较结果。返回匹配的排序列表（最佳匹配优先）。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT FindTemplateWalsh(
    HRESULT                hrPrev,
    ITcVnImage*            ipSrcImage,
    ITcVnImage*            ipTemplateImage,
    ITcVnContainer*&       ipMatches,
    float                  fMatchThreshold,
    ULONG                  nProjections,
    double                 fStartAngle,
    double                 fStopAngle,
    ULONG                  nAngles,
    ULONG                  eOptions = FTWO_NONE,
    float                  fScaleFactor = 1,
    ETcVnInterpolationType eInterpolationType = IT_BILINEAR,
    ITcVnContainer**       pipMatchValues = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT 或 SINT，1 或 3 通道）

名称	类型	默认值	描述
ipTemplateImage	ITcVnImage* [▶ 1863]		模板图像（与 ipSrcImage 类型相同，宽度和高度较小）
ipMatches	ITcVnContainer* [▶ 1827]&		返回匹配位置（容器类型取决于 eOptions；ContainerType_Vector_TcVnRotatedRectangle 或 ContainerType_Vector_TcVnVector4_REAL，其中每个元素包含中心点 (x, y)、以度为单位的旋转角以及 ipTemplateImage 在 ipSrcImage 中的匹配值；容器元素按相关性排序（最佳匹配项优先））
fMatchThreshold	float		用于区分相关匹配和不相关匹配的阈值（0...1，1.0 表示完全匹配。）
nProjections	ULONG		要使用的Walsh投影数量。nProjections 值越高，匹配时考虑的细节就越多。由于最相关的信息由前几个Walsh投影所捕获，因此较少的数量（例如 5、10、25 或 50）即可满足许多使用场景。每个投影都需要额外的计算时间。因此，只要满足匹配要求，选择较少的Walsh投影数量是有利的。允许的最大 nProjections 值由模板图像的宽度和高度决定。
fStartAngle	double		起始搜索角度
fStopAngle	double		终止搜索角度
nAngles	ULONG		要使用的不同搜索角度的数量
eOptions	ULONG	FTWO_NONE	控制匹配过程的标志。（参见 ETcVnFindTemplateWalshOptions；默认值：TCVN_FTWO_NONE）
fScaleFactor	float	1	系数（0..1]，用于减小源图像和模板图像的宽度和高度，以提高性能（但精度降低！）
eInterpolationType	ETcVnInterpolationType [▶ 1680]	IT_BILINEAR	图像大小调整插值类型（仅在 fScaleFactor != 1 时使用，建议大多数情况下使用 TCVN_IT_BILINEAR）

名称	类型	默认值	描述
pipMatchValues	ITcVnContainer* [▶ 1827]*	nullptr	可选择返回匹配值（ContainerType_Vector_REAL，大小和排序与 ipMatches 相同。如果不需要，则设为 0。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

6.2.3.15.1.7 HoughCircles

使用霍夫（Hough）变换搜索圆。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT HoughCircles(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipCircles,
    double           fInvAccuRatio,
    double           fMinDist,
    ETcVnHoughMethod eHoughMethod = HM_GRADIENT,
    double           fParam1 = 100,
    double           fParam2 = 100,
    ULONG            nMinRadius = 0,
    ULONG            nMaxRadius = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT，1 通道，灰度等级）
ipCircles	ITcVnContainer* [▶ 1827]&		返回一个以找到的圆填充的容器（ContainerType_Vector_TcVnVector3_REAL；每个容器元素分别包含圆心的 x 坐标 [0]、圆心的 y 坐标 [1] 和半径 [2]。重复使用非零接口指针。）
fInvAccuRatio	double		累加器大小与源图像大小的反比（必须大于 0。值为 2 表示大小双向减半）。
fMinDist	double		两个圆的最小允许距离（必须大于 0）

名称	类型	默认值	描述
eHoughMethod	ETcVnHoughMethod [▶ 1679]	HM_GRADIENT	要使用的霍夫（Hough）变换方法（GRADIENT 或 GRADIENT_ALT）
fParam1	double	100	第一个方法特定参数（GRADIENT、GRADIENT_ALT：Canny 边缘检测的阈值上限，必须大于 0）
fParam2	double	100	第二个方法特定参数（GRADIENT：用于检测圆心的累加器阈值，必须大于 0。GRADIENT_ALT：所需的圆完美度，必须大于 0 且小于 1，1 表示一个完美圆）
nMinRadius	ULONG	0	允许的最小圆半径
nMaxRadius	ULONG	0	允许的最大圆半径（如果为 0，则该值被内设为最大值（图像行，图像列））

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.1.8 HoughLines

使用标准霍夫（Hough）变换搜索线条。

语法

定义：

```
HRESULT HoughLines(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipLines,
    double           fDistRes,
    double           fAngleRes,
    ULONG            nAccuThreshold,
    double           fDistResDiv = 0,
    double           fAngleResDiv = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT，1 通道，二值，可由函数修改）
ipLines	ITcVnContainer* [▶ 1827]&		返回一个以找到的线条填充的容器（ContainerType_Vector_TcVnVector2_REAL；每个容器元素分别包含与原点的距离 [0]（以像素为单位）和旋转角 [1]（以弧度为单位）。重复使用非零接口指针。）
fDistRes	double		累加器的距离分辨率（以像素为单位，必须大于 0）
fAngleRes	double		累加器的角度分辨率（以弧度为单位，必须大于 0）
nAccuThreshold	ULONG		累加器阈值
fDistResDiv	double	0	多尺度霍夫（Hough）变换的距离分辨率除数（>= 0，如果 fAngleResDiv > 0，则必须 > 0）
fAngleResDiv	double	0	多尺度霍夫（Hough）变换的角度分辨率除数（>= 0，如果 fDistResDiv > 0，则必须 > 0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.1.9 HoughLinesP

使用概率霍夫（Hough）变换搜索线段。

语法

定义：

```
HRESULT HoughLinesP(
    HRESULT           hrPrev,
    ITcVnImage*       ipSrcImage,
    ITcVnContainer*& ipLines,
    double            fDistRes,
    double            fAngleRes,
    ULONG             nAccuThreshold,
    double            fMinLineLength = 0,
    double            fMaxLineGap = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT，1 通道，二值，可由函数修改）
ipLines	ITcVnContainer* [▶ 1827]&		返回一个以找到的线段填充的容器（ContainerType_Vector_TcVnVector4_DINT；每个容器元素包含两个终点。重复使用非零接口指针。）
fDistRes	double		累加器的距离分辨率（以像素为单位，必须大于 0）
fAngleRes	double		累加器的角度分辨率（以弧度为单位，必须大于 0）
nAccuThreshold	ULONG		累加器阈值
fMinLineLength	double	0	搜索的最小直线长度
fMaxLineGap	double	0	同一直线上各点之间的最大间距

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.1.10 MatchImageHuMoments

使用 Hu 不变矩比较两个图像。

语法

定义：

```
HRESULT MatchImageHuMoments(
    HRESULT                           hrPrev,
    ITcVnImage*                       ipImage1,
    ITcVnImage*                       ipImage2,
    ETcVnContoursMatchComparisonMethod eComparisonMethod,
    double&                           fDissimilarity
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage1	ITcVnImage* [▶ 1863]	第一个图像（1 通道）
ipImage2	ITcVnImage* [▶ 1863]	第二个图像（1 通道）
eComparisonMethod	ETcVnContoursMatchComparisonMethod [▶ 1667]	用于比较图像 Hu 不变矩的方法
fDissimilarity	double&	根据所选的比较方法，返回图像 Hu 不变矩的相异度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.1.11 MatchTemplate

将模板图像与源图像中的每个位置进行匹配，并保存比较结果。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT MatchTemplate(
    HRESULT                   hrPrev,
    ITcVnImage*               ipSrcImage,
    ITcVnImage*               ipTemplateImage,
    ITcVnImage*&              ipResultImage,
    ETcVnTemplateMatchMethod eMatchMethod = TMM_CCORR_NORMED,
    ITcVnImage*               ipTemplateMask = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT 或 REAL，1 或 3 通道）
ipTemplateImage	ITcVnImage* [▶ 1863]		模板图像（与 ipSrcImage 类型相同，宽度和高度较小）
ipResultImage	ITcVnImage* [▶ 1863]&		返回结果图像（REAL，1 通道，尺寸：（ipSrcImage.width - ipTemplateImage.width + 1）x（ipSrcImage.height - ipTemplateImage.height + 1）。最佳匹配是全局最小值（SQDIFF（_NORMED））或最大值（CCORR（_NORMED），CCOEFF（_NORMED））。ipResultImage 中的位置是 ipSrcImage 中 ipTemplateImage 位置的左上角）
eMatchMethod	ETcVnTemplateMatchMethod [▶ 1693]	TMM_CCORR_NORMED	指定模板匹配方法
ipTemplateMask	ITcVnImage* [▶ 1863]	nullptr	ipTemplateImage 的可选掩膜（类型和大小与 ipTemplateImage 相同）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.1.12 MatchTemplateAndEvaluate

将模板图像与源图像中的每个位置进行匹配，并评估比较结果。返回匹配的排序列表（最佳匹配优先）。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT MatchTemplateAndEvaluate(
    HRESULT                  hrPrev,
    ITcVnImage*              ipSrcImage,
    ITcVnImage*              ipTemplateImage,
    ITcVnContainer*&         ipMatches,
    float                    fMatchThreshold,
    ETcVnTemplateMatchMethod eMatchMethod = TMM_CCORR_NORMED,
    ITcVnImage*              ipTemplateMask = nullptr,
    float                    fScaleFactor = 1,
    ETcVnInterpolationType   eInterpolationType = IT_BILINEAR,
    ITcVnContainer**         pipMatchValues = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT 或 REAL，1 或 3 通道）
ipTemplateImage	ITcVnImage* [▶ 1863]		模板图像（与 ipSrcImage 类型相同，宽度和高度较小）
ipMatches	ITcVnContainer* [▶ 1827]&		返回 ipSrcImage 中的匹配位置（ContainerType_Vector_TcVnPoint2_DINT，其中每个元素代表 ipTemplateImage 的左上角），按相关性排序（最佳匹配项优先）
fMatchThreshold	float		用于区分相关匹配和不相关匹配的阈值（NORMED 方法为 0...1，其它取决于模板大小和内容。为了找到合适的值，可以对 F_VN_MatchTemplateExp 的一些示例结果图像进行评估。）
eMatchMethod	ETcVnTemplateMatchMethod [▶ 1693]	TMM_CCORR_NORMED	指定模板匹配方法
ipTemplateMask	ITcVnImage* [▶ 1863]	nullptr	ipTemplateImage 的可选掩膜（类型和大小与 ipTemplateImage 相同）
fScaleFactor	float	1	系数（0..1]，用于减小源图像和模板图像的宽度和高度，以提高性能（但精度降低！）
eInterpolationType	ETcVnInterpolationType [▶ 1680]	IT_BILINEAR	图像大小调整插值类型（仅在 fScaleFactor != 1 时使用，建议大多数情况下使用 TCVN_IT_BILINEAR）

名称	类型	默认值	描述
pipMatchValues	ITcVnContainer* [▶ 1827]*	nullptr	可选择返回匹配值（ContainerType_Vector_REAL，大小和排序与 ipMatches 相同。如果不需要，则设为 0。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.1.13 MatchTemplateWalsh

将模板图像与源图像中的每个位置进行匹配（使用Walsh变换），并返回比较结果。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT MatchTemplateWalsh(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*  ipTemplateImage,
    ITcVnImage*& ipResultImage,
    ULONG        nProjections,
    ULONG        eOptions = MTWO_NONE,
    ULONG        nRotations = 1
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT 或 SINT，1 或 3 通道）
ipTemplateImage	ITcVnImage* [▶ 1863]		模板图像（与 ipSrcImage 类型相同，宽度和高度较小）
ipResultImage	ITcVnImage* [▶ 1863]&		返回结果图像（REAL，归一化为 [0...1]，nRotations 通道，尺寸：（ipSrcImage.width - ipTemplateImage.width + 1）x

名称	类型	默认值	描述
			（ipSrcImage.height - ipTemplateImage.height + 1）。最佳匹配是所有通道中的全局最小值。ipResultImage 中的位置是 ipSrcImage 中 ipTemplateImage 位置的左上角。旋转可以通过最小值所在通道来推断。）
nProjections	ULONG		要使用的Walsh投影数量。nProjections 值越高，匹配时考虑的细节就越多。由于最相关的信息由前几个Walsh投影所捕获，因此较少的数量（例如 5、10、25 或 50）即可满足许多使用场景。每个投影都需要额外的计算时间。因此，只要满足匹配要求，选择较少的Walsh投影数量是有利的。允许的最大 nProjections 值由模板图像的宽度和高度决定。
eOptions	ULONG	MTWO_NONE	控制匹配过程的标志。（请参见 ETcVnMatchTemplateWalshOptions）
nRotations	ULONG	1	要检查的旋转数量。（有效值：1（0°）、2（0° 和 180°）和 4（0°、90°、180° 和 270°；仅适用于正方形图像））

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

6.2.3.15.2 CannyEdgeDetection

使用 Canny 边缘检测算法查找边缘。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT CannyEdgeDetection(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    double       fThresholdLow,
    double       fThresholdHigh,
    ULONG        nApertureSize = 3,
    bool         bL2Gradient = false
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT 类型元素）
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（USINT.ID 类型元素。如有需要，将创建适当的目标图像。）
fThresholdLow	double		低阈值
fThresholdHigh	double		高阈值
nApertureSize	ULONG	3	索贝尔（Sobel）算子的孔径大小（3、5、7）
bL2Gradient	BOOL	false	如果为 true，则使用更精确（也更慢）的 L2 范数，而不是 L1 范数

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.3 ConnectedComponents

计算图像的连通分量。

语法

定义：

```
HRESULT ConnectedComponents(
    HRESULT                           hrPrev,
    ITcVnImage*                       ipSrcImage,
    ITcVnImage*&                      ipLabelImage,
    LONG&                             nLabels,
    ETcVnPixelConnectivity            eConnectivity = PC_8,
    ETcVnElementType                  eLabelType = TCVN_ET_DINT,
    ETcVnConnectedComponentsAlgorithm eAlgorithm = CCA_GRANA
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT，1 通道，二值）
ipLabelImage	ITcVnImage* [▶ 1863]&		返回每个源图像像素的标签（1 通道，类型取决于 eLabelType。如有需要，将创建适当的目标图像。）
nLabels	LONG&		返回标签的数量
eConnectivity	ETcVnPixelConnectivity [▶ 1686]	PC_8	选择使用像素 4 连通还是 8 连通
eLabelType	ETcVnElementType [▶ 1674]	TCVN_ET_DINT	选择 ipLabelImage 的类型（仅支持 UINT 或 DINT）
eAlgorithm	ETcVnConnectedComponentsAlgorithm [▶ 1665]	CCA_GRANA	选择应用的算法

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.4 ConnectedComponentsWithStats

计算图像的连通分量和相应的统计数据。

语法

定义：

```
HRESULT ConnectedComponentsWithStats(
    HRESULT                           hrPrev,
    ITcVnImage*                       ipSrcImage,
    ITcVnImage*&                      ipLabelImage,
    ITcVnContainer*&                  ipBoundingBoxes,
    ITcVnContainer*&                  ipNumPixels,
    ITcVnContainer*&                  ipCentroids,
    LONG&                             nLabels,
    ETcVnPixelConnectivity            eConnectivity = PC_8,
    ETcVnElementType                  eLabelType = TCVN_ET_DINT,
    ETcVnConnectedComponentsAlgorithm eAlgorithm = CCA_GRANA
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT，1 通道，二值）
ipLabelImage	ITcVnImage* [▶ 1863]&		返回每个源图像像素的标签（1 通道，DINT. 如有需要，将创建适当的目标图像。）
ipBoundingBoxes	ITcVnContainer* [▶ 1827]&		返回每个标记区域的边界框。（ContainerType_Vector_TcVnRectangle_DINT；重复使用非零接口指针。）
ipNumPixels	ITcVnContainer* [▶ 1827]&		返回每个标记区域的像素数量。（ContainerType_Vector_DINT；重复使用非零接口指针。）
ipCentroids	ITcVnContainer* [▶ 1827]&		返回每个标记区域的重心。（ContainerType_Vector_TcVnPoint2_LREAL；重复使用非零接口指针。）
nLabels	LONG&		返回标签的数量
eConnectivity	ETcVnPixelConnectivity [▶ 1686]	PC_8	选择使用像素 4 连通还是 8 连通
eLabelType	ETcVnElementType [▶ 1674]	TCVN_ET_DINT	选择 ipLabelImage 的类型（仅支持 UINT 或 DINT）
eAlgorithm	ETcVnConnectedComponentsAlgorithm [▶ 1665]	CCA_GRANA	选择应用的算法

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.5 CountNonZeroPixels

统计单通道图像中的非零像素（例如，适用于分析阈值结果）。

语法

定义：

```
HRESULT CountNonZeroPixels(
    HRESULT     hrPrev,
    ITcVnImage* ipSrcImage,
    ULONGLONG&  nNonZero
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（1 通道）
nNonZero	ULONGLONG&	返回 ipSrcImage 中非零像素的数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.6 DistanceTransformation

计算距离变换，即二值图像中与最近零像素之间的距离。此外，还创建了包含分量标签的图像（离散维诺图（Voronoi））。

语法

定义：

```
HRESULT DistanceTransformation(
    HRESULT                          hrPrev,
    ITcVnImage*                      ipSrcImage,
    ITcVnImage*&                     ipDestImage,
    ETcVnDistanceType                eDistanceType,
    ETcVnDistanceTransformationMask  eMaskSize,
    ITcVnImage**                     pipDestLabels = nullptr,
    ETcVnDistanceTransformationLabel eLabelType = DTL_CCOMP
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT，1 通道）
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（REAL，1 通道。如有需要，将创建适当的目标图像。）
eDistanceType	ETcVnDistanceType [▶ 1671]		距离计算方法（支持：L1、L2、C）
eMaskSize	ETcVnDistanceTransformationMask [▶ 1670]		距离变换掩膜的大小（不支持 PRECISE 用于标签计算）
pipDestLabels	ITcVnImage* [▶ 1863]*	nullptr	返回分量标签（离散维诺图（Voronoi）；DINT，1 通道。如有需要，将创建适当的图像）。
eLabelType	ETcVnDistanceTransformationLabel [▶ 1670]	DTL_CCOMP	标签类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.7 GetConnectedComponent

根据标签 ID，从标签图像中获取连通分量的掩膜图像（还可以选择外轮廓）。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT GetConnectedComponent(
    HRESULT          hrPrev,
    ITcVnImage*      ipLabelImage,
    LONG             nLabelID,
    ITcVnImage*&     ipMaskImage,
    ITcVnContainer** pipContourPoints = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipLabelImage	ITcVnImage* [▶ 1863]		包含每个像素标签的图像（1 通道 DINT 或 UINT）。
nLabelID	LONG		请求标签的值
ipMaskImage	ITcVnImage* [▶ 1863]&		返回掩膜图像，该图像中标记属于分量的像素（1 通道 USINT。掩膜为可选项，如果不需要，则设为 0。）
pipContourPoints	ITcVnContainer* [▶ 1827]*	nullptr	返回分量的外轮廓（ContainerType_Vector_TcVnPoint2_DINT。轮廓是可选项，如果不需要，则设为 0。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.8 ImageAverage

计算图像的（通道）平均像素值。

语法

定义：

```
HRESULT ImageAverage(
    HRESULT             hrPrev,
    ITcVnImage*         ipSrcImage,
    TcVnVector4_LREAL&  aAverage,
    ITcVnImage*         ipMask = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（1 - 4 通道）
aAverage	TcVnVector4_LREAL [▶ 1641]&		返回 ipSrcImage 的（通道）平均像素值
ipMask	ITcVnImage* [▶ 1863]	nullptr	USINT 类型的掩膜（1 通道）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.9 ImageAverageStdDev

计算图像的（通道）像素平均值和相应的标准偏差。

语法

定义：

```
HRESULT ImageAverageStdDev(
    HRESULT           hrPrev,
    ITcVnImage*       ipSrcImage,
    TcVnVector4_LREAL& aAverage,
    TcVnVector4_LREAL& aStdDev,
    ITcVnImage*       ipMask = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（1 - 4 通道）
aAverage	TcVnVector4_LREAL [▶ 1641]&		返回 ipSrcImage 的（通道）平均像素值
aStdDev	TcVnVector4_LREAL [▶ 1641]&		返回 ipSrcImage 的（通道）像素值标准偏差
ipMask	ITcVnImage* [▶ 1863]	nullptr	USINT 类型的掩膜（1 通道）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.10 ImageCenterOfMass

计算图像的质心。

语法

定义：

```
HRESULT ImageCenterOfMass(
    HRESULT           hrPrev,
    ITcVnImage*       ipSrcImage,
    TcVnPoint2_LREAL& aCenterOfMass,
    ITcVnImage*       ipMask = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（1 通道）
aCenterOfMass	TcVnPoint2_LREAL [▶ 1640]&		返回图像的质心
ipMask	ITcVnImage* [▶ 1863]	nullptr	可选掩膜（1 通道，USINT 类型，宽度和高度与 ipSrcImage 相同）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.11 ImageMedian

计算图像的（近似、通道）像素中值。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT ImageMedian(
    HRESULT            hrPrev,
    ITcVnImage*        ipSrcImage,
    TcVnVector4_LREAL& aMedian,
    ITcVnImage*        ipMask = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（1 - 4 通道）
aMedian	TcVnVector4_LREAL [▶ 1641]&		返回 ipSrcImage 的（近似、通道）像素中值
ipMask	ITcVnImage* [▶ 1863]	nullptr	USINT 类型的掩膜（1 通道）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.12 ImageMoments

计算图像的空间矩、中心矩和归一化中心矩，最高可达三阶。

语法

定义：

```
HRESULT ImageMoments(
    HRESULT      hrPrev,
    ITcVnImage*  ipImage,
    TcVnMoments& stMoments,
    bool         bBinaryImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]	源图像（1 通道，USINT 或 REAL 类型元素）
stMoments	TcVnMoments [▶ 1707]&	返回包含矩的结构体
bBinaryImage	BOOL	如果为 true，所有非零像素都被视为 1

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.13 MaxPixelValue

查找图像中的最大像素值（支持 1 - 4 个通道）。

语法

定义：

```
HRESULT MaxPixelValue(
    HRESULT                 hrPrev,
    ITcVnImage*             ipImage,
    TcVnVector4_LREAL&      aMaxValue,
    TcVnPoint2_DINT&        aPosition,
    ITcVnImage*             ipMask = nullptr,
    ETcVnVectorCompareMethod eVectorCompareMethod = VCM_EUCLIDEAN
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]		源图像
aMaxValue	TcVnVector4_LREAL [▶ 1641]&		返回最大像素值
aPosition	TcVnPoint2_DINT [▶ 1640]&		返回第一个找到的 aMaxValue 位置（不支持使用 ELEMENTWISE 的多通道图像）

名称	类型	默认值	描述
ipMask	ITcVnImage* [▶ 1863]	nullptr	可选掩膜，用于指定考虑哪些像素位置（USINT，如果不需要，将参数设为 0）
eVectorCompareMethod	ETcVnVectorCompareMethod [▶ 1695]	VCM_EUCLIDEAN	为多通道图像选择向量比较方法

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.14 MinPixelValue

查找图像中的最小像素值（支持 1 - 4 个通道）。

语法

定义：

```
HRESULT MinPixelValue(
    HRESULT                 hrPrev,
    ITcVnImage*             ipImage,
    TcVnVector4_LREAL&      aMinValue,
    TcVnPoint2_DINT&        aPosition,
    ITcVnImage*             ipMask = nullptr,
    ETcVnVectorCompareMethod eVectorCompareMethod = VCM_EUCLIDEAN
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]		源图像
aMinValue	TcVnVector4_LREAL [▶ 1641]&		返回最小像素值
aPosition	TcVnPoint2_DINT [▶ 1640]&		返回第一个找到的 aMinValue 位置（不支持使用 ELEMENTWISE 的多通道图像）
ipMask	ITcVnImage* [▶ 1863]	nullptr	可选掩膜，用于指定考虑哪些像素位置（USINT，如果不需要，将参数设为 0）

名称	类型	默认值	描述
eVectorCompareMethod	ETcVnVectorCompareMethod [▶ 1695]	VCM_EUCLIDEAN	为多通道图像选择向量比较方法

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.15 RegionOrientation

根据二值图像计算一个区域的方向，该二值图像包含该区域每个像素的非零值。

语法

定义：

```
HRESULT RegionOrientation(
    HRESULT                hrPrev,
    ITcVnImage*            ipSrcImage,
    TcVnRotatedRectangle&  stOrientation,
    ETcVnOrientationMethod eMethod = OM_PCA
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（1 通道，二值）
stOrientation	TcVnRotatedRectangle [▶ 1718]&		旋转后的矩形，包含中心点、轴长和顺时针方向的区域旋转角。
eMethod	ETcVnOrientationMethod [▶ 1686]	OM_PCA	计算方向的方法。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.16 ImageAbsoluteAverage

计算图像的平均绝对像素值。
可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT ImageAbsoluteAverage(
    HRESULT     hrPrev,
    ITcVnImage* ipSrcImage,
    double&     fAverage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
fAverage	double&	返回平均绝对像素值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.17 UprightInnerRectangle

计算区域内直立内矩形的最大面积。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT UprightInnerRectangle(
    HRESULT             hrPrev,
    ITcVnImage*         ipSrcImage,
    TcVnRectangle_DINT& stRectangle,
    LONG                nDownScaleFactor = -1
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT，1 通道，二进制，值 > 0 标记有效区域）
stRectangle	TcVnRectangle_DINT [▶ 1717]&		返回直立内矩形的最大面积
nDownScaleFactor	LONG	-1	为了权衡性能与精度，可以提供一个系数来降低内部使用的图像分辨率。只要有效区域中的无效区域大于 nDownScaleFactor，结果通常仍然是最优的。使用 -1 可根据 ipSrcImage 的分辨率自动选择，使用 0 可始终使用完整分辨率（速度较慢，但始终是最优结果）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.15.18 FindRois

Extracts rectangular regions of interest (ROIs) from an image by partitioning it into tiles using the selected method and parameters.

Syntax

Definition:

```
HRESULT FindRois(
    HRESULT             hrPrev,
    ITcVnImage*         ipSrcImage,
    ITcVnContainer*&    ipRois,
    ETcVnFindRoisMethod eFindRoisMethod,
    TcVnParamsFindRois& stParams
)
```

Parameters

Name	Type	Description
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)
ipSrcImage	ITcVnImage* [▶ 1863]	Source image (1 channel, non-negative values).
ipRois	ITcVnContainer* [▶ 1827]&	Returns the ROIs (ContainerType_Vector_TcVnRectangle_DINT; Non-zero interface pointers are reused.)
eFindRoisMethod	ETcVnFindRoisMethod [▶ 1643]	Method to create ROIs
stParams	TcVnParamsFindRois [▶ 1697]&	Parameters for extracting ROIs

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.2.3.15.19 InnerCircle

Calculates the position and radius of the largest inscribed circle within the white region of a binary image.

Syntax

Definition:

```
HRESULT InnerCircle(
    HRESULT                          hrPrev,
    ITcVnImage*                      ipSrcImage,
    double&                          fRadius,
    TcVnPoint2_REAL&                 aCenter,
    ETcVnDistanceType                eDistanceType = DT_L2,
    ETcVnDistanceTransformationMask  eMaskSize = DTM_5
)
```

Parameters

Name	Type	Default	Description
hrPrev	HRESULT [▶ 135]		HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)
ipSrcImage	ITcVnImage* [▶ 1863]		Source image (USINT, 1 channel)

Name	Type	Default	Description
fRadius	double&		Returns the radius of the inner circle
aCenter	TcVnPoint2_REAL [▶ 1640]&		Returns the center of the inner circle
eDistanceType	ETcVnDistanceType [▶ 1671]	DT_L2	Distance computation method (supported: L1, L2, C)
eMaskSize	ETcVnDistanceTransformationMask [▶ 1670]	DTM_5	Size of the distance transformation mask

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.2.3.15.20 SalientFeatures

Compute salient features from the input image using the specified method.

Syntax

Definition:

```
HRESULT SalientFeatures(
    HRESULT                     hrPrev,
    ITcVnImage*                 ipSrcImage,
    ITcVnImage*&                ipDestImage,
    ETcVnSalientFeaturesMethod eMethod
)
```

Parameters

Name	Type	Description
hrPrev	HRESULT [▶ 135]	HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)
ipSrcImage	ITcVnImage* [▶ 1863]	Source image (USINT elements, 1 or 3 channels; 3 channel input is expected to be RGB and internally converted to Gray).
ipDestImage	ITcVnImage* [▶ 1863]&	Destination image (1 channel image of type TCVN_ET_UINT, an appropriate destination image will be created if required.)
eMethod	ETcVnSalientFeaturesMethod [▶ 1644]	Method to compute salient features.

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.2.3.16 图像颜色和对比度处理

该组包含色彩和对比处理函数。

函数

图像的频率分布、强度和对比调整

- Clahe [▶ 2449]
- Histogram [▶ 2452]
- HistogramEqualization [▶ 2453]
- NormalizeImage [▶ 2455]
- NormalizeImageForDisplay [▶ 2456]

查找表和色图

- ApplyColorMap [▶ 2447]
- ApplyLut [▶ 2448]
- GenerateColorMap [▶ 2451]
- GenerateCustomColorMap [▶ 2452]

颜色匹配

- ReferenceColorSimilarity [▶ 2457]
- TrainImageColor [▶ 2460]
- TrainImageColor_ITcVnMlModel [▶ 2462]

白平衡

- WhiteBalance [▶ 2463]

其他

- ConvertColorSpace [▶ 2450]
- InvertImageColor [▶ 2454]

6.2.3.16.1 ApplyColorMap

在灰度图像上应用色图。

语法

定义：

```
HRESULT ApplyColorMap(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnImage*&     ipDestImage,
    ITcVnContainer*  ipColorMap
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（TCVN_ET_USINT 或 TCVN_ET_UINT，1 通道）
ipDestImage	ITcVnImage* [▶ 1863]&	返回彩色图像（TCVN_ET_USINT，3 通道。如有需要，将创建适当的图像）。
ipColorMap	ITcVnContainer* [▶ 1827]	应用于 ipSrcImage 的色图（ContainerType_Vector_TcVnVector3_REAL，包含 256 或 65536 个元素，视 ipSrcImage 类型而定）。可进行自定义，也可以通过 F_VN_GenerateColorMap 创建。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.2 ApplyLut

将查找表应用于图像，对其颜色进行处理。

语法

定义：

```
HRESULT ApplyLut(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnImage*&     ipDestImage,
    ITcVnContainer*  ipLut
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	描述
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（1-4 个通道，8 或 16 位，SINT、USINT、INT、UINT 类型）
ipDestImage	ITcVnImage* [▶ 1863]&	返回结果图像（大小、类型和通道与 ipSrcImage 相同。如有需要，将创建适当的图像）。
ipLut	ITcVnContainer* [▶ 1827]	查找表，包含与 ipSrcImage 类型相匹配的元素，256（8 位图像）或 65536（16 位图像）个元素。例如，用于 1-4 通道 USINT 图像的 ContainerType_Vector_USINT（在这种情况下，每个通道使用相同的查找值）或用于 3 通道 USINT 图像的 ContainerType_TcVnVector3_USINT（在这种情况下，每个通道使用单独的查找值）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.3 Clahe

应用限制对比度自适应性直方图均衡（CLAHE）。对单个平铺图应用局部直方图均衡。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT Clahe(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    double       fClipLimit = 40,
    ULONG        nTilesX = 8,
    ULONG        nTilesY = 8
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	默认值	描述
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（TCVN_ET_USINT 或 TCVN_ET_UINT，1 通道）
ipDestImage	ITcVnImage* [▶ 1863]&		返回结果图像（如果需要，将创建合适的图像。）
fClipLimit	double	40	对比度限制阈值（设置为 <= 0 时禁用裁剪）
nTilesX	ULONG	8	x 方向的平铺图数量
nTilesY	ULONG	8	y 方向的平铺图数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.4 ConvertColorSpace

将图像从一种颜色空间转换到另一种颜色空间。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被看门狗取消时，可返回部分结果。

语法

定义：

```
HRESULT ConvertColorSpace(
    HRESULT                 hrPrev,
    ITcVnImage*             ipSrcImage,
    ITcVnImage*&            ipDestImage,
    ETcVnColorSpaceTransform eTransform
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
eTransform	ETcVnColorSpaceTransform [▶ 1654]	要应用的变换

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.5 GenerateColorMap

生成预定义的色图。

语法

定义：

```
HRESULT GenerateColorMap(
    HRESULT           hrPrev,
    ITcVnContainer*&  ipColorMap,
    ETcVnColorMap     eColorMap,
    ETcVnColorMapSize eColorMapSize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipColorMap	ITcVnContainer* [▶ 1827]&	返回色图（ContainerType_Vector_TcVnVector3_REAL，包含 256 或 65536 个元素）
eColorMap	ETcVnColorMap [▶ 1653]	选择色图（类似于 GNU Octave/MATLAB 型）
eColorMapSize	ETcVnColorMapSize [▶ 1654]	定义生成的色图应包含多少个元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.6 GenerateCustomColorMap

通过对用户定义的颜色（对于每个通道均为平均线性分布）进行内插，生成自定义色图。

语法

定义：

```
HRESULT GenerateCustomColorMap(
    HRESULT           hrPrev,
    ITcVnContainer*&  ipColorMap,
    ITcVnContainer*   ipInitialColors,
    ETcVnColorMapSize eColorMapSize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipColorMap	ITcVnContainer* [▶ 1827]&	返回色图（ContainerType_Vector_TcVnVector3_REAL，包含 256 或 65536 个元素）
ipInitialColors	ITcVnContainer* [▶ 1827]	用户自定义颜色（ContainerType_Vector_TcVnVector3_REAL，至少 2 个元素）
eColorMapSize	ETcVnColorMapSize [▶ 1654]	定义生成的色图应包含多少个元素

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.7 直方图

计算图像的（多通道）直方图。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT Histogram(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipDestHistogram,
    ULONG&           nBins = 0,
    double&          fLowerBound = 0,
    double&          fUpperBound = 0,
    ITcVnImage*      ipMask = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestHistogram	ITcVnContainer* [▶ 1827]&		返回包含多通道直方图的容器，其中每个通道都以 UDINT 向量表示（ContainerType_Vector_Vector_UDINT。重复使用非零接口指针。）
nBins	ULONG&	0	所需的分组数量，或用 0 保持相应图像格式的默认值（输入）和默认分组数量（输出）
fLowerBound	double&	0	第 0 个直方图分组的下边界（包含）（输入），如果 fLowerBound 与 fUpperBound 都设置为 0，则接收默认值（输出）
fUpperBound	double&	0	最后一个直方图分组 nBins-1 的上边界（不包含）（输入），如果 fLowerBound 与 fUpperBound 设置为 0，则接收默认值（输出）
ipMask	ITcVnImage* [▶ 1863]	nullptr	源图像掩膜（TCVN_ET_USINT，1 通道。掩膜为可选项，如果不需要，则设为 0。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.8 HistogramEqualization

均衡灰度图或彩色图像的直方图，使亮度归一化并提高对比度。 必须指定应均衡化的通道索引（-1 应为 RGB 图像，将其转换为 YCbCr，均衡 Y 通道并将图像转换回 RGB）。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT HistogramEqualization(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    SHORT        nChannelIdx = -1
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT）
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的目标图像。）
nChannelIdx	SHORT	-1	应均衡化的图像通道的索引（-1 应为 RGB 图像，将其转换为 YCbCr，均衡 Y 通道并将图像转换回 RGB）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.9 InvertImageColor

反转图像的颜色。 对于带符号的整数或负浮点数值，只会切换符号。 如果图像只包含正浮点数值，则每个像素值都要减去可用的最大像素值（或 1.0，取两者较高值）。

语法

定义：

```
HRESULT InvertImageColor(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    double       fMaxValue = -1
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的图像。）
fMaxValue	double	-1	最大像素值（例如，如果 16 位图像包含 12 位值）。-1 表示使用默认值。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.10 NormalizeImage

将图像的值范围归一化（例如，将像素值 [50...150] 扩大到全范围 [0...255]），或根据特定的归一化（例如 L2-范数）对值进行缩放。

语法

定义：

```
HRESULT NormalizeImage(
    HRESULT                 hrPrev,
    ITcVnImage*             ipSrcImage,
    ITcVnImage*&            ipDestImage,
    double                  fAlpha = 0,
    double                  fBeta = 255,
    ETcVnNormalizationType  eNormType = NT_MINMAX,
    ETcVnElementType        eDestType = TCVN_ET_SAME_AS_SOURCE,
    ITcVnImage*             ipMask = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像

名称	类型	默认值	描述
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的目标图像。）
fAlpha	double	0	范围的下边界（在对值范围进行归一化时）或要归一化为的数值（|| ipDestImage|| = fAlpha）
fBeta	double	255	范围的上边界（在对值范围进行归一化时）
eNormType	ETcVnNormalizationType [▶ 1684]	NT_MINMAX	归一化类型（仅限 INF、L1、L2 或 MINMAX）
eDestType	ETcVnElementType [▶ 1674]	TCVN_ET_SAME_AS_SOURCE	目标图像深度（通常为 SAME_AS_SOURCE）
ipMask	ITcVnImage* [▶ 1863]	nullptr	用于将归一化限制在特定像素位置的掩膜（设为 0 则对整个图像进行归一化）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.11 NormalizeImageForDisplay

归一化进行显示的图像，即：将图像缩放至基础数据类型的全值范围（浮点为 -1 至 1）。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。
被看门狗取消时，可返回部分结果。

语法

定义：

```
HRESULT NormalizeImageForDisplay(
    HRESULT                 hrPrev,
    ITcVnImage*             ipSrcImage,
    ITcVnImage*&            ipDestImage,
    ETcVnSignedNormalization eSignedNormalization
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像

名称	类型	描述
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
eSignedNormalization	ETcVnSignedNormalization [▶ 1690]	可对带符号的值进行归一化

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.12 ReferenceColorSimilarity (ITcVnColorModel*)

计算源图像中每个像素与参考颜色模型的相似度。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT ReferenceColorSimilarity(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnImage*&     ipDestImage,
    ITcVnColorModel* ipColorModel,
    float            fVariance = 0.15f,
    float            fLuminanceWeight = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		RGB 源图像（USINT，3 通道）
ipDestImage	ITcVnImage* [▶ 1863]&		返回 ipSrcImage 中每个像素与 ipColorModel 的相似度（USINT，1 通道。如有需要，将创建适当的目标图像。）
ipColorModel	ITcVnColorModel* [▶ 1869]		颜色模型
fVariance	float	0.15f	允许的颜色方差（0.1 - 0.3 可能是一个不错的尝试起点）

名称	类型	默认值	描述
fLuminanceWeight	float	0	对亮度的影响进行加权（[0...1]，例如，设置为 0 更能抵御不均匀光照的影响，但可能需要用于区分某些颜色。如果 ipColorModel 为 RGB 类型，则忽略。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.13 ReferenceColorSimilarity (ITcVnMlModel*)

计算源图像中每个像素与参考颜色模型的相似度。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT ReferenceColorSimilarity(
    HRESULT       hrPrev,
    ITcVnImage*   ipSrcImage,
    ITcVnImage*&  ipDestImage,
    ITcVnMlModel* ipColorModel,
    float         fVariance = 0.15f,
    float         fLuminanceWeight = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		RGB 源图像（USINT，3 通道）
ipDestImage	ITcVnImage* [▶ 1863]&		返回 ipSrcImage 中每个像素与 ipColorModel 的相似度（USINT，1 通道。如有需要，将创建适当的目标图像。）
ipColorModel	ITcVnMlModel* [▶ 1871]		颜色模型

名称	类型	默认值	描述
fVariance	float	0.15f	允许的颜色方差（0.1 - 0.3 可能是一个不错的尝试起点）
fLuminanceWeight	float	0	对亮度的影响进行加权（[0...1]，例如，设置为 0 更能抵御不均匀光照的影响，但可能需要用于区分某些颜色。如果 ipColorModel 为 RGB 类型，则忽略。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.14 ReferenceColorSimilarity (TcVnVector3_LREAL)

计算源图像中每个像素与参考颜色的相似度。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT ReferenceColorSimilarity(
    HRESULT             hrPrev,
    ITcVnImage*         ipSrcImage,
    ITcVnImage*&        ipDestImage,
    TcVnVector3_LREAL&  aRefColor,
    float               fVariance = 0.15f,
    float               fLuminanceWeight = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		RGB 源图像（USINT，3 通道）
ipDestImage	ITcVnImage* [▶ 1863]&		返回 ipSrcImage 中每个像素与 aRefColor 的相似度（USINT，1 通道。如有需要，将创建适当的目标图像。）

名称	类型	默认值	描述
aRefColor	TcVnVector3_LREAL [▶ 1641]&		参考颜色（RGB，[0...255]）
fVariance	float	0.15f	允许的颜色方差（0.1 - 0.3 可能是一个不错的尝试起点）
fLuminanceWeight	float	0	对亮度的影响进行加权（[0...1]，例如，设置为 0 更能抵御不均匀光照的影响，但可能需要用于区分某些颜色）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.15 TrainImageColor

创建新的颜色模型，用于描述图像颜色。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT TrainImageColor(
    HRESULT                  hrPrev,
    ITcVnImage*              ipSrcImage,
    ITcVnColorModel*&        ipColorModel,
    ULONG                    nDifferentColors,
    ETcVnColorTrainingMethod eMethod = CTM_LAB,
    ITcVnImage*              ipMask = nullptr,
    ULONG                    nSkipPixels = 0,
    ETcVnClusteringAlgorithm eClusteringAlgorithm = CA_KMEANSPP,
    LONG                     nMaxClusterRadius = 50,
    bool                     bSingleSplitSteps = true
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（3 通道（RGB），USINT 类型）
ipColorModel	ITcVnColorModel* [▶ 1869]&		返回颜色模型

名称	类型	默认值	描述
nDifferentColors	ULONG		要区分的不同颜色的最大数量（如果使用 LBG 作为聚类算法，则结果中的不同颜色数量可能较少，具体取决于 nMaxClusterRadius）
eMethod	ETcVnColorTrainingMethod [▶ 1665]	CTM_LAB	色彩训练方法
ipMask	ITcVnImage* [▶ 1863]	nullptr	可选的图像掩膜（1 通道，USINT 类型，如果不需要，则设为 0）
nSkipPixels	ULONG	0	在每个评估的颜色样本之间跳过的像素数量（以获得更好的性能）。0 将每个像素都考虑在内，往往更准确。
eClusteringAlgorithm	ETcVnClusteringAlgorithm [▶ 1651]	CA_KMEANSPP	聚类算法
nMaxClusterRadius	LONG	50	仅用于 LBG 聚类算法。单个聚类允许的最大半径（> 0），即聚类半径越大，被分割成的聚类越小，直到达到 nDifferentColors 的全局数量。
bSingleSplitSteps	BOOL	true	仅用于 LBG 聚类算法。如果为 true，则总是在拆分单个聚类后运行全局优化。如果为 false，则会在应用全局优化之前，在同一步骤中拆分多个聚类。减少全局优化的应用次数会加快速度，但可能会导致结果不理想，尤其是有 2 个可由 1 表示的邻近聚类时

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.16 TrainImageColor (ITcVnMlModel*)

创建新的颜色模型，用于描述图像颜色。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT TrainImageColor(
    HRESULT                 hrPrev,
    ITcVnImage*             ipSrcImage,
    ITcVnMlModel*&          ipColorModel,
    ULONG                   nDifferentColors,
    ETcVnColorTrainingMethod eMethod = CTM_LAB,
    ITcVnImage*             ipMask = nullptr,
    ULONG                   nSkipPixels = 0,
    ETcVnClusteringAlgorithm eClusteringAlgorithm = CA_KMEANSPP,
    LONG                    nMaxClusterRadius = 50,
    bool                    bSingleSplitSteps = true
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（3 通道（RGB），USINT 类型）
ipColorModel	ITcVnMlModel* [▶ 1871]&		返回颜色模型
nDifferentColors	ULONG		要区分的不同颜色的最大数量（如果使用 LBG 作为聚类算法，则结果中的不同颜色数量可能较少，具体取决于 nMaxClusterRadius）
eMethod	ETcVnColorTrainingMethod [▶ 1665]	CTM_LAB	色彩训练方法
ipMask	ITcVnImage* [▶ 1863]	nullptr	可选的图像掩膜（1 通道，USINT 类型，如果不需要，则设为 0）
nSkipPixels	ULONG	0	在每个评估的颜色样本之间跳过的像素数量（以获得更好的性能）。0 将每个像素都考虑在内，往往更准确。
eClusteringAlgorithm	ETcVnClusteringAlgorithm [▶ 1651]	CA_KMEANSPP	聚类算法
nMaxClusterRadius	LONG	50	仅用于 LBG 聚类算法。单个聚类允许的最大半径（> 0），即聚类半径越大，被分割成的聚类越小，直到达到 nDifferentColors 的全局数量。

名称	类型	默认值	描述
bSingleSplitSteps	BOOL	true	仅用于 LBG 聚类算法。如果为 true，则总是在拆分单个聚类后运行全局优化。如果为 false，则会在应用全局优化之前，在同一步骤中拆分多个聚类。减少全局优化的应用次数会加快速度，但可能会导致结果不理想，尤其是有 2 个可由 1 表示的邻近聚类时

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.16.17 WhiteBalance

计算 RGB 图像的白平衡比率。将绿色通道用作参考，计算红色通道和蓝色通道的相对比率。要求在图像的至少 100 个像素内有一个不反光、不过曝的白色对象（例如一张纸），以提供合理的结果。然后，得出的比率可在相机上设置（与现有比率相乘），或通过 MultiplyImageWithVector 在 PLC 中应用。

语法

定义：

```
HRESULT WhiteBalance(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage** pipDestImage,
    float&       fRatioRed,
    float&       fRatioBlue,
    ULONG        nMaxValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（3 通道（RGB），USINT 或 UINT）
pipDestImage	ITcVnImage* [▶ 1863]*	应用白平衡的目标图像（可选项，如果不需要，则设为 0。如有需要，将创建适当的图像）。
fRatioRed	float&	返回红色通道的比率
fRatioBlue	float&	返回蓝色通道的比率

名称	类型	描述
nMaxValue	ULONG	指定曝光过度所能达到的最大像素值（8 位数据通常为 255，12 位数据通常为 4095）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17 图像滤波

该组包含用于滤波图像的函数。

函数

模糊化

- BilateralFilter [▶ 2465]
- BoxFilter [▶ 2466]
- GaussianFilter [▶ 2470]
- MedianFilter [▶ 2474]

边缘检测

- LaplacianFilter [▶ 2471]
- ScharrFilter [▶ 2476]
- SobelFilter [▶ 2478]

用户定义滤波

- CustomFilter [▶ 2468]
- SeparableCustomFilter [▶ 2477]

填充区域

- BrightBorderObjects [▶ 2467]
- DarkBorderObjects [▶ 2469]
- FillHoles [▶ 2470]

形态学算子

- CreateStructuringElement [▶ 2467]
- MorphologicalOperator [▶ 2474]

局部极值

- LocalMaxima [▶ 2472]
- LocalMinima [▶ 2473]

- RemoveLocalMaxima [▶ 2475]
- RemoveLocalMinima [▶ 2476]

局部方差

- VarianceFilter [▶ 2479]

6.2.3.17.1 BilateralFilter

应用双边滤波来平滑图像，但保留边缘。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT BilateralFilter(
    HRESULT                       hrPrev,
    ITcVnImage*                   ipSrcImage,
    ITcVnImage*&                  ipDestImage,
    LONG                          nDiameter,
    double                        fSigmaColor,
    double                        fSigmaSpace,
    ETcVnBorderInterpolationMethod eBorderType = BIM_DEFAULT
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（必须与 ipSrcImage 不同！如有需要，将创建适当的目标图像。）
nDiameter	LONG		用于滤波的像素邻域直径（如 5、7、9）。如果 <= 0，则根据 fSigmaSpace 自动选择。
fSigmaColor	double		用于颜色空间滤波的西格玛值（> 0）。数值越大，表示混合的颜色越多。
fSigmaSpace	double		用于坐标空间滤波的西格玛值（> 0）。数值越大，表示能相互影响的像素越多。
eBorderType	ETcVnBorderInterpolationMethod [▶ 1648]	BIM_DEFAULT	图像边框处理

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.2 BoxFilter

对图像应用方框滤波。

语法

定义：

```
HRESULT BoxFilter(
    HRESULT                         hrPrev,
    ITcVnImage*                     ipSrcImage,
    ITcVnImage*&                    ipDestImage,
    ULONG                           nFilterWidth,
    ULONG                           nFilterHeight,
    ETcVnElementType                eDestDepth = TCVN_ET_SAME_AS_SOURCE,
    TcVnPoint&                      aAnchor = {-1, -1},
    bool                            bNormalize = true,
    ETcVnBorderInterpolationMethod eBorderType = BIM_DEFAULT
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的目标图像。）
nFilterWidth	ULONG		滤波宽度，单位：像素
nFilterHeight	ULONG		滤波高度，单位：像素
eDestDepth	ETcVnElementType [▶ 1674]	TCVN_ET_SAME_AS_SOURCE	目标图像深度
aAnchor	TcVnPoint [▶ 1640]&	{-1, -1}	内核的锚点（[-1，-1] 为中心点）
bNormalize	BOOL	true	如果为 true，内核将根据 nFilterWidth * nFilterHeight 进行归一化处理
eBorderType	ETcVnBorderInterpolationMethod [▶ 1648]	BIM_DEFAULT	图像边框处理

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.3 BrightBorderObjects

在单通道灰度图像中查找与图像边界相连的明亮对象。

语法

定义：

```
HRESULT BrightBorderObjects(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（USINT、UINT、INT、REAL 或 LREAL，1 通道）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.4 CreateStructuringElement

为形态学算子的使用创建结构元素，并分配适当的数据缓冲区。如果创建了新的图像接口，初始引用计数设为 1；如果未创建，则保留计数。

语法

定义：

```
HRESULT CreateStructuringElement(
    HRESULT                      hrPrev,
    ITcVnImage*&                 ipStructuringElement,
    ETcVnStructuringElementShape eShape,
    ULONG                        nWidth,
    ULONG                        nHeight
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipStructuringElement	ITcVnImage* [▶ 1863]&	返回创建的结构元素（重复使用非零接口指针。）
eShape	ETcVnStructuringElementShape [▶ 1691]	结构元素的形状（矩形、十字形或椭圆形）
nWidth	ULONG	Width（宽度）
nHeight	ULONG	高度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.5 CustomFilter

在图片上应用自定义滤波。

语法

定义：

```
HRESULT CustomFilter(
    HRESULT                          hrPrev,
    ITcVnImage*                      ipSrcImage,
    ITcVnImage*&                     ipDestImage,
    ETcVnElementType                 eDestDepth,
    TcVnMatrix&                      stKernel,
    TcVnPoint&                       aAnchor = {-1, -1},
    double                           fDelta = 0,
    ETcVnBorderInterpolationMethod eBorderType = BIM_DEFAULT
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的目标图像。）
eDestDepth	ETcVnElementType [▶ 1674]		目标图像深度

名称	类型	默认值	描述
stKernel	TcVnMatrix [▶ 1706]&		包含 REAL 或 LREAL 类型值的自定义滤波核
aAnchor	TcVnPoint [▶ 1640]&	{-1, -1}	内核的锚点
fDelta	double	0	滤波后每个像素的附加值
eBorderType	ETcVnBorderInterpolationMethod [▶ 1648]	BIM_DEFAULT	图像边框处理

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.6 DarkBorderObjects

在单通道灰度图像中查找与图像边界相连的暗色对象。（相当于填充孔洞）

语法

定义：

```
HRESULT DarkBorderObjects(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（USINT、UINT、INT、REAL 或 LREAL，1 通道）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.7 FillHoles

填充单通道灰度图像中的孔洞。（相当于查找暗色边界对象）

语法

定义：

```
HRESULT FillHoles(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（USINT、UINT、INT、REAL 或 LREAL，1 通道）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.8 GaussianFilter

应用高斯（Gaussian）滤波器来平滑图像。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT GaussianFilter(
    HRESULT                     hrPrev,
    ITcVnImage*                 ipSrcImage,
    ITcVnImage*&                ipDestImage,
    ULONG                       nFilterWidth,
    ULONG                       nFilterHeight,
    double                      fSigmaX = 0,
```

```
    double                          fSigmaY = 0,
    ETcVnBorderInterpolationMethod eBorderType = BIM_DEFAULT
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的目标图像。）
nFilterWidth	ULONG		滤波宽度，单位：像素（1、3、5、7...）
nFilterHeight	ULONG		滤波高度，单位：像素（1、3、5、7...）
fSigmaX	double	0	高斯核在 X 方向上的标准偏差（>= 0，如果为 0 则自动选择）
fSigmaY	double	0	高斯核在 Y 方向上的标准偏差（>= 0，如果为 0 则自动选择）
eBorderType	ETcVnBorderInterpolationMethod [▶ 1648]	BIM_DEFAULT	图像边框处理

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.9 LaplacianFilter

对图像应用拉普拉斯（Laplacian）滤波。

语法

定义：

```
HRESULT LaplacianFilter(
    HRESULT                         hrPrev,
    ITcVnImage*                     ipSrcImage,
    ITcVnImage*&                    ipDestImage,
    ETcVnElementType                eDestDepth,
    ULONG                           nKernelSize = 1,
    double                          fScale = 1,
    double                          fDelta = 0,
    ETcVnBorderInterpolationMethod eBorderType = BIM_DEFAULT
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的目标图像。）
eDestDepth	ETcVnElementType [▶ 1674]		目标图像深度
nKernelSize	ULONG	1	用于计算二阶导数滤波的孔径大小（1、3、5...31）
fScale	double	1	导数计算值的比例系数
fDelta	double	0	在存储结果前加入这些结果的 Delta 值
eBorderType	ETcVnBorderInterpolationMethod [▶ 1648]	BIM_DEFAULT	图像边框处理

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.10 LocalMaxima

在单通道灰度图像中查找局部最大值。找到的最大值在目标图像中标为 1。

语法

定义：

```
HRESULT LocalMaxima(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（USINT 或 UINT，1 通道）

名称	类型	描述
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（必须与 ipSrcImage 不同！如有需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.11 LocalMinima

在单通道灰度图像中查找局部最小值。找到的最小值在目标图像中标为 1。

语法

定义：

```
HRESULT LocalMinima(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（USINT 或 UINT，1 通道）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（必须与 ipSrcImage 不同！如有需要，将创建适当的目标图像。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.12 MedianFilter

对图像应用中值滤波。

语法

定义：

```
HRESULT MedianFilter(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    ULONG        nFilterSize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（对于 nFilterSize 3 或 5：USINT、UINT、REAL。对于较大的滤波，只支持 USINT。
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
nFilterSize	ULONG	滤波大小（宽度和高度）（3、5、7...）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.13 MorphologicalOperator

应用形态学算子。

语法

定义：

```
HRESULT MorphologicalOperator(
    HRESULT                   hrPrev,
    ITcVnImage*               ipSrcImage,
    ITcVnImage*&              ipDestImage,
    ETcVnMorphologicalOperator eOperator,
    ITcVnImage*               ipStructuringElement
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（重建算子仅支持 1 个通道）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
eOperator	ETcVnMorphologicalOperator [▶ 1682]	算子类型
ipStructuringElement	ITcVnImage* [▶ 1863]	要应用的结构元素（通常通过 F_VN_CreateStructuringElement 创建。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.14 RemoveLocalMaxima

从单通道灰度图像中去除不超过给定高度值的局部最大值。

语法

定义：

```
HRESULT RemoveLocalMaxima(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    double       fHeight
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（USINT、UINT、INT、REAL 或 LREAL，1 通道）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
fHeight	double	要删除的最大值的最大高度（必须大于 0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.15 RemoveLocalMinima

从单通道灰度图像中去除不超过给定高度值的局部最小值。

语法

定义：

```
HRESULT RemoveLocalMinima(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    double       fHeight
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（USINT、UINT、INT、REAL 或 LREAL，1 通道）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
fHeight	double	要删除的最小值的最大高度（必须大于 0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.16 ScharrFilter

使用沙尔（Scharr）滤波计算 x 或 y 方向的一阶导数。

语法

定义：

```
HRESULT ScharrFilter(
    HRESULT                         hrPrev,
    ITcVnImage*                     ipSrcImage,
    ITcVnImage*&                    ipDestImage,
    ETcVnElementType                eDestDepth,
    ETcVnFilterDirection            eFilterDirection,
    double                          fScale = 1,
    double                          fDelta = 0,
    ETcVnBorderInterpolationMethod eBorderType = BIM_DEFAULT
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的目标图像。）
eDestDepth	ETcVnElementType [▶ 1674]		目标图像深度
eFilterDirection	ETcVnFilterDirection [▶ 1677]		滤波方向
fScale	double	1	导数计算值的比例系数
fDelta	double	0	将结果存储到目标位置之前，加入这些结果的 Delta 值
eBorderType	ETcVnBorderInterpolationMethod [▶ 1648]	BIM_DEFAULT	图像边框处理

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.17 SeparableCustomFilter

对图像应用可分离的自定义滤波。

语法

定义：

```
HRESULT SeparableCustomFilter(
    HRESULT                         hrPrev,
    ITcVnImage*                     ipSrcImage,
    ITcVnImage*&                    ipDestImage,
    ETcVnElementType                eDestDepth,
    TcVnMatrix&                     stKernelX,
    TcVnMatrix&                     stKernelY,
    TcVnPoint&                      aAnchor = {-1, -1},
    double                          fDelta = 0,
    ETcVnBorderInterpolationMethod eBorderType = BIM_DEFAULT
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的目标图像。）
eDestDepth	ETcVnElementType [▶ 1674]		目标图像深度
stKernelX	TcVnMatrix [▶ 1706]&		包含 REAL 或 LREAL 类型值的一维自定义行滤波核
stKernelY	TcVnMatrix [▶ 1706]&		包含 REAL 或 LREAL 类型值的一维自定义列滤波核
aAnchor	TcVnPoint [▶ 1640]&	{-1, -1}	内核的锚点
fDelta	double	0	滤波后每个像素的附加值
eBorderType	ETcVnBorderInterpolationMethod [▶ 1648]	BIM_DEFAULT	图像边框处理

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.18 SobelFilter

使用扩展索贝尔（Sobel）滤波计算一阶、二阶、三阶或混合图像导数。

语法

定义：

```
HRESULT SobelFilter(
    HRESULT                         hrPrev,
    ITcVnImage*                     ipSrcImage,
    ITcVnImage*&                    ipDestImage,
    ETcVnElementType                eDestDepth,
```

```
    ULONG                           nXOrder,
    ULONG                           nYOrder,
    ULONG                           nKernelSize = 3,
    double                          fScale = 1,
    double                          fDelta = 0,
    ETcVnBorderInterpolationMethod eBorderType = BIM_DEFAULT
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的目标图像。）
eDestDepth	ETcVnElementType [▶ 1674]		目标图像深度
nXOrder	ULONG		x 导数的阶数（必须 < nKernelSize）
nYOrder	ULONG		y 导数的阶数（必须 < nKernelSize）
nKernelSize	ULONG	3	扩展索贝尔（Sobel）内核的大小（3、5、7...31）
fScale	double	1	导数计算值的比例系数
fDelta	double	0	将结果存储到目标位置之前，加入这些结果的 Delta 值
eBorderType	ETcVnBorderInterpolationMethod [▶ 1648]	BIM_DEFAULT	图像边框处理

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.17.19 VarianceFilter

计算图像的局部方差。

语法

定义：

```
HRESULT VarianceFilter(
    HRESULT                         hrPrev,
    ITcVnImage*                     ipSrcImage,
```

```
    ITcVnImage*&                    ipDestImage,
    ULONG                           nKernelSize,
    double                          fScale = 1,
    ETcVnBorderInterpolationMethod eBorderType = BIM_DEFAULT
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像。仅支持 TCVN_ET_USINT 类型。
ipDestImage	ITcVnImage* [▶ 1863]&		TCVN_ET_UINT 类型的目标图像（如果需要，将创建适当的目标图像）。
nKernelSize	ULONG		内核大小
fScale	double	1	方差计算值的比例系数
eBorderType	ETcVnBorderInterpolationMethod [▶ 1648]	BIM_DEFAULT	图像边框处理

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.18 图像分割

该组包含用于图像分割的函数。

函数

Segmentation

- AdaptiveThreshold [▶ 2481]
- CheckColorRange [▶ 2482]
- DoubleThreshold [▶ 2482]
- Threshold [▶ 2483]
- WatershedSegmentationExp [▶ 2484]
- SauvolaThreshold [▶ 2485]

6.2.3.18.1 AdaptiveThreshold

对灰度图像应用自适应阈值。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT AdaptiveThreshold(
    HRESULT                         hrPrev,
    ITcVnImage*                     ipSrcImage,
    ITcVnImage*&                    ipDestImage,
    double                          fMaxValue,
    ETcVnAdaptiveThresholdMethod eAdaptiveMethod = ATM_MEAN,
    ETcVnThresholdType              eThresholdType = TT_BINARY,
    ULONG                           nBlockSize = 3,
    double                          fConstant = 5
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（1 通道，USINT 类型元素）
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的目标图像。）
fMaxValue	double		为阈值条件为 true 的像素分配的值
eAdaptiveMethod	ETcVnAdaptiveThresholdMethod [▶ 1646]	ATM_MEAN	要应用的自适应阈值方法（MEAN：阈值将以（x，y）的 nBlockSize x nBlockSize 邻域的平均值减去 fConstant 来计算。GAUSSIAN：阈值是（x,y）的 nBlockSize x nBlockSize 邻域的加权和再减去 fConstant（与高斯窗口互相关）。
eThresholdType	ETcVnThresholdType [▶ 1694]	TT_BINARY	要应用的阈值类型（仅支持 BINARY 和 BINARY_INV）
nBlockSize	ULONG	3	用于计算局部阈值的像素邻域大小（3、5、7...）
fConstant	double	5	从像素邻域的加权平均值中减去的常数，从而得出局部阈值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.18.2 CheckColorRange

检查图像的像素值是否在给定范围内。目标图像的大小与源图像相同，但只有一个通道包含 8 位像素元素。如果源图像的相应像素在检查范围内，则其元素设置为 255，否则设置为 0。

语法

定义：

```
HRESULT CheckColorRange(
    HRESULT            hrPrev,
    ITcVnImage*        ipSrcImage,
    ITcVnImage*&       ipDestImage,
    TcVnVector4_LREAL& aLowerBounds,
    TcVnVector4_LREAL& aUpperBounds
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
aLowerBounds	TcVnVector4_LREAL [▶ 1641]&	通道下界（未使用的通道将被忽略。）
aUpperBounds	TcVnVector4_LREAL [▶ 1641]&	通道上界（未使用的通道将被忽略。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.18.3 DoubleThreshold

使用形态学重构应用双阈值（也被称为滞后性阈值）。
可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
HRESULT DoubleThreshold(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    double       fLowThreshold,
    double       fHighThreshold,
    double       fMaxValue
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（USINT、UINT、INT、REAL 或 LREAL，1 通道）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
fLowThreshol	double	决定已找到区域形状的低阈值
fHighThreshold	double	选择已找到区域的高阈值（如果应用高阈值，则每个区域必须至少包含一个像素。）
fMaxValue	double	最大像素值

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.18.4 Threshold

根据大津（Otsu）算法应用固定阈值或动态阈值。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被看门狗取消时，可返回部分结果。

语法

定义：

```
HRESULT Threshold(
    HRESULT             hrPrev,
    ITcVnImage*         ipSrcImage,
    ITcVnImage*&        ipDestImage,
    double              fThreshold,
    double              fMaxValue,
    ETcVnThresholdType eThresholdType
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（对于大津（Otsu）和三角阈值类型，仅支持 1 通道 TCVN_ET_USINT 类型）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
fThreshold	double	固定阈值（选择动态阈值时未使用）
fMaxValue	double	最大像素值
eThresholdType	ETcVnThresholdType [▶ 1694]	要应用的阈值类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.18.5 WatershedSegmentation (Exp)

应用基于标记的分水岭分割。

语法

定义：

```
HRESULT WatershedSegmentation(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    ITcVnImage*  ipMarkers
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（USINT，3 通道）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（DINT，1 通道。如有需要，将创建适当的图像）。
ipMarkers	ITcVnImage* [▶ 1863]	标记图像（DINT，1 通道）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.18.6 SauvolaThreshold

对源图像应用 Sauvola 本地阈值。

语法

定义：

```
HRESULT SauvolaThreshold(
    HRESULT             hrPrev,
    ITcVnImage*         ipSrcImage,
    ITcVnImage*&        pipDestImage,
    ULONG               nBlockSize,
    double              fParamK,
    double              fMaxSigma,
    ETcVnThresholdType  eThresholdType
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（USINT、UINT、INT、REAL 或 LREAL，1 通道）
pipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的单通道目标图像。））
nBlockSize	ULONG	用于计算局部阈值的像素邻域大小（3、5、7...）
fParamK	double	在范围（0，1）内取正值的参数 K。建议 K 在范围 [0.2，0.5] 内
fMaxSigm	double	标准偏差的最大值，必须严格为正值。（对于灰度文档，fMaxSigma = 128）。
eThresholdType	ETcVnThresholdType [▶ 1694]	要应用的阈值类型。支持 TCVN_TT_BINARY 和 TCVN_TT_BINARY_INV。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19 关键点特征

该组包含用于查找和匹配图像中明显特征的函数。

注意

计算时间长

许多查找关键点特征的函数都需要很长的计算时间。因此，请注意设定循环时间，且必要时使用Watchdog（看门狗） [▶ 140]，以保证实时行为。

函数

关键点

- KeyPointsAGAST [▶ 2496]
- KeyPointsFAST [▶ 2500]
- KeyPointsGFTT [▶ 2501]
- KeyPointsMSER [▶ 2502]
- KeyPointsSB [▶ 2503]

关键点和描述子

- KeyPointsAndDescriptorsAKAZE [▶ 2497]
- KeyPointsAndDescriptorsBRISK [▶ 2498]
- KeyPointsAndDescriptorsaKAZE [▶ 2498]
- KeyPointsAndDescriptorsORB [▶ 2499]

描述子匹配

- FilterGoodMatches [▶ 2486]
- GetMatchCoordinates [▶ 2495]
- MatchDescriptorsBF [▶ 2504]
- MatchDescriptorsFlannLsh [▶ 2505]
- MatchDescriptorsKnnBF [▶ 2506]
- MatchDescriptorsKnnFlannLsh [▶ 2507]

关键点引用匹配

- FindReferenceKeyPointsInImage [▶ 2487]
- FindReferenceKeyPointsInImageAKAZE [▶ 2489]
- FindReferenceKeyPointsInImageBRISK [▶ 2491]
- FindReferenceKeyPointsInImageORB [▶ 2493]

其它

- RegionsMSER [▶ 2508]

6.2.3.19.1 FilterGoodMatches

过滤描述符匹配项，仅返回匹配的描述符。

语法

定义：

```
HRESULT FilterGoodMatches(
    HRESULT          hrPrev,
    ITcVnContainer*  ipMatches,
    ITcVnContainer*& ipGoodMatches,
    float            fMaxDist,
    float            fMaxKnnRatio
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMatches	ITcVnContainer* [▶ 1827]	包含描述符匹配项的容器（ContainerType_Vector_TcVnDMatch 或 ContainerType_Vector_Vector_TcVnDMatch）
ipGoodMatches	ITcVnContainer* [▶ 1827]&	返回包含良好匹配项的容器（ContainerType_Vector_TcVnDMatch）
fMaxDist	float	允许的最大描述符距离（-1 禁用此过滤标准）
fMaxKnnRatio	float	第一和第二最佳匹配结果之间允许的最大距离比（[0...1]，仅用于 knn 匹配结果，-1 则禁用此过滤标准）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.2 FindReferenceKeyPointsInImage

在给定的源图像中搜索以其关键点和相应描述符表示的参考图像。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT FindReferenceKeyPointsInImage(
    HRESULT                  hrPrev,
    ITcVnContainer*          ipSrcKeyPoints,
    ITcVnImage*              ipSrcDescriptors,
    ITcVnContainer*          ipRefKeyPoints,
    ITcVnImage*              ipRefDescriptors,
    ULONG                    nRefImageWidth,
    ULONG                    nRefImageHeight,
    ITcVnContainer*&         ipEdgePoints,
```

```
    float                       fMaxDist,
    float                       fMaxKnnRatio,
    ETcVnNormalizationType      eNormType,
    TcVnMatrix3x3_LREAL&        aPerspectiveTransform,
    ULONG&                      nNumberOfGoodMatches,
    ULONG&                      nNumberOfInliers,
    ETcVnEstimationAlgorithm    eAlgorithm = EA_RANSAC,
    double                      fReprojThreshold = 3,
    ULONG                       nMaxIterations = 2000,
    double                      fConfidence = 0.995,
    ITcVnContainer**            pipGoodMatches = nullptr,
    ITcVnContainer**            pipInlierMask = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcKeyPoints	ITcVnContainer* [▶ 1827]		源图像的关键点，例如（ContainerType_Vector_TcVnKeyPoint）
ipSrcDescriptors	ITcVnImage* [▶ 1863]		源图像关键点的描述符
ipRefKeyPoints	ITcVnContainer* [▶ 1827]		参考图像的关键点（ContainerType_Vector_TcVnKeyPoint）
ipRefDescriptors	ITcVnImage* [▶ 1863]		参考图像关键点的描述符
nRefImageWidth	ULONG		参考图像的宽度，单位：像素
nRefImageHeight	ULONG		参考图像的高度，单位：像素
ipEdgePoints	ITcVnContainer* [▶ 1827]&		返回一个容器，其中包含转换为源图像坐标的 4 个参考图像边缘点（ContainerType_Vector_TcVnPoint2_REAL）
fMaxDist	float		允许的最大描述符距离（-1 禁用此过滤标准）
fMaxKnnRatio	float		第一和第二最佳匹配结果之间允许的最大距离比（[0...1]，-1 则禁用此过滤标准）
eNormType	ETcVnNormalizationType [▶ 1684]		用于描述符匹配的归一化类型（AKAZE、ORB 和 BRISK 建议使用 HAMMING。如果 ORB nBriefPoints 参数为 3 或 4，建议 ORB 使用 HAMMING2。）
aPerspectiveTransform	TcVnMatrix3x3_LREAL [▶ 1641]&		返回透视变换矩阵，该矩阵将参考点变换为源点
nNumberOfGoodMatches	ULONG&		返回良好匹配项的数量
nNumberOfInliers	ULONG&		返回内群匹配项的数量
eAlgorithm	ETcVnEstimationAlgorithm [▶ 1674]	EA_RANSAC	用于计算两个点集之间透视变换的估计算法

名称	类型	默认值	描述
fReprojThreshold	double	3	将点对视为内群时允许的最大重投影误差（仅在 eAlgorithm 为 RANSAC、RHO 时使用）
nMaxIterations	ULONG	2000	最大 RANSAC 迭代次数
fConfidence	double	0.995	置信度（0..1）
pipGoodMatches	ITcVnContainer* [▶ 1827]*	nullptr	返回包含良好匹配项的容器（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnDMatch）
pipInlierMask	ITcVnContainer* [▶ 1827]*	nullptr	返回标记内群值的掩码（可选，如果不需要，则设为 0；ContainerType_Vector_SINT；仅适用于 RANSAC 和 LMEDS）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.3 FindReferenceKeyPointsInImageAKAZE

在给定的源图像中搜索以其 AKAZE 关键点和相应描述符表示的参考图像。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT FindReferenceKeyPointsInImageAKAZE(
    HRESULT                  hrPrev,
    ITcVnImage*              ipSrcImage,
    ITcVnContainer*          ipRefKeyPoints,
    ITcVnImage*              ipRefDescriptors,
    ULONG                    nRefImageWidth,
    ULONG                    nRefImageHeight,
    ITcVnContainer*&         ipEdgePoints,
    float                    fMaxDist,
    float                    fMaxKnnRatio,
    TcVnMatrix3x3_LREAL&     aPerspectiveTransform,
    ULONG&                   nNumberOfGoodMatches,
    ULONG&                   nNumberOfInliers,
    ETcVnNormalizationType   eNormType = NT_HAMMING,
    ITcVnImage*              ipSrcImageMask = nullptr,
    TcVnParamsAKAZE&         stAkazeParams = {FDT_AKAZE_MLDB, 0, 3, 0.001f, 2, 1, DT1_KAZE_PM_G2},
    ETcVnEstimationAlgorithm eAlgorithm = EA_RANSAC,
    double                   fReprojThreshold = 3,
    ULONG                    nMaxIterations = 2000,
    double                   fConfidence = 0.995
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipRefKeyPoints	ITcVnContainer* [▶ 1827]		参考图像的关键点，例如使用 F_VN_KeyPointsAndDescriptorsAKAZE（ContainerType_Vector_TcVnKeyPoint）计算得出的关键点
ipRefDescriptors	ITcVnImage* [▶ 1863]		参考图像关键点的描述符，例如使用 F_VN_KeyPointsAndDescriptorsAKAZE 计算得出的描述符
nRefImageWidth	ULONG		参考图像的宽度，单位：像素
nRefImageHeight	ULONG		参考图像的高度，单位：像素
ipEdgePoints	ITcVnContainer* [▶ 1827]&		返回一个容器，其中包含转换为 ipSrcImage 坐标的 4 个参考图像边缘点（ContainerType_Vector_TcVnPoint2_REAL）
fMaxDist	float		允许的最大描述符距离（-1 禁用此过滤标准）
fMaxKnnRatio	float		第一和第二最佳匹配结果之间允许的最大距离比（[0...1]，-1 则禁用此过滤标准）
aPerspectiveTransform	TcVnMatrix3x3_LREAL [▶ 1641]&		返回透视变换矩阵，该矩阵将参考点变换为源点
nNumberOfGoodMatches	ULONG&		返回良好匹配项的数量
nNumberOfInliers	ULONG&		返回内群匹配项的数量
eNormType	ETcVnNormalizationType [▶ 1684]	NT_HAMMING	用于描述符匹配的归一化类型
ipSrcImageMask	ITcVnImage* [▶ 1863]	nullptr	该掩膜用于指定在哪里查找 ipSrcImage 的关键点（如果不需要掩膜，则设为 0）
stAkazeParams	TcVnParamsAKAZE [▶ 1708]&	{FDT_AKAZE_MLDB, 0, 3, 0.001f, 2, 1, DT1_KAZE_PM_G2}	用于为 ipSrcImage 配置关键点和描述符计算的参数（产生的描述符必须与 ipRefDescriptors 兼容！）

名称	类型	默认值	描述
eAlgorithm	ETcVnEstimationAlgorithm [▶ 1674]	EA_RANSAC	用于计算两个点集之间透视变换的估计算法
fReprojThreshold	double	3	将点对视为内群时允许的最大重投影误差（仅在 eAlgorithm 为 RANSAC、RHO 时使用）
nMaxIterations	ULONG	2000	最大 RANSAC 迭代次数
fConfidence	double	0.995	置信度（0..1）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.4 FindReferenceKeyPointsInImageBRISK

在给定的源图像中搜索以其 BRISK 关键点和相应描述符表示的参考图像。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT FindReferenceKeyPointsInImageBRISK(
    HRESULT                   hrPrev,
    ITcVnImage*               ipSrcImage,
    ITcVnContainer*           ipRefKeyPoints,
    ITcVnImage*               ipRefDescriptors,
    ULONG                     nRefImageWidth,
    ULONG                     nRefImageHeight,
    ITcVnContainer*&          ipEdgePoints,
    float                     fMaxDist,
    float                     fMaxKnnRatio,
    TcVnMatrix3x3_LREAL&      aPerspectiveTransform,
    ULONG&                    nNumberOfGoodMatches,
    ULONG&                    nNumberOfInliers,
    ETcVnNormalizationType    eNormType = NT_HAMMING,
    ITcVnImage*               ipSrcImageMask = nullptr,
    TcVnParamsBRISK&          stBriskParams = {30, 3, 1},
    ETcVnEstimationAlgorithm  eAlgorithm = EA_RANSAC,
    double                    fReprojThreshold = 3,
    ULONG                     nMaxIterations = 2000,
    double                    fConfidence = 0.995
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	默认值	描述
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipRefKeyPoints	ITcVnContainer* [▶ 1827]		参考图像的关键点，例如使用 F_VN_KeyPointsAndDescriptorsBRISK（ContainerType_Vector_TcVnKeyPoint）计算得出的关键点
ipRefDescriptors	ITcVnImage* [▶ 1863]		参考图像关键点的描述符，例如使用 F_VN_KeyPointsAndDescriptorsBRISK 计算得出的描述符
nRefImageWidth	ULONG		参考图像的宽度，单位：像素
nRefImageHeight	ULONG		参考图像的高度，单位：像素
ipEdgePoints	ITcVnContainer* [▶ 1827]&		返回一个容器，其中包含转换为 ipSrcImage 坐标的 4 个参考图像边缘点（ContainerType_Vector_TcVnPoint2_REAL）
fMaxDist	float		允许的最大描述符距离（-1 禁用此过滤标准）
fMaxKnnRatio	float		第一和第二最佳匹配结果之间允许的最大距离比（[0...1]，-1 则禁用此过滤标准）
aPerspectiveTransform	TcVnMatrix3x3_LREAL [▶ 1641]&		返回透视变换矩阵，该矩阵将参考点变换为源点
nNumberOfGoodMatches	ULONG&		返回良好匹配项的数量
nNumberOfInliers	ULONG&		返回内群匹配项的数量
eNormType	ETcVnNormalizationType [▶ 1684]	NT_HAMMING	用于描述符匹配的归一化类型
ipSrcImageMask	ITcVnImage* [▶ 1863]	nullptr	该掩膜用于指定在哪里查找 ipSrcImage 的关键点（如果不需要掩膜，则设为 0）
stBriskParams	TcVnParamsBRISK [▶ 1711]&	{30, 3, 1}	用于为 ipSrcImage 配置关键点和描述符计算的参数（产生的描述符必须与 ipRefDescriptors 兼容！）
eAlgorithm	ETcVnEstimationAlgorithm [▶ 1674]	EA_RANSAC	用于计算两个点集之间透视变换的估计算法
fReprojThreshold	double	3	将点对视为内群时允许的最大重投影误差（仅在 eAlgorithm 为 RANSAC、RHO 时使用）
nMaxIterations	ULONG	2000	最大 RANSAC 迭代次数

名称	类型	默认值	描述
fConfidence	double	0.995	置信度（0..1）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.5 FindReferenceKeyPointsInImageORB

在给定的源图像中搜索以其 ORB 关键点和相应描述符表示的参考图像。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT FindReferenceKeyPointsInImageORB(
    HRESULT                 hrPrev,
    ITcVnImage*             ipSrcImage,
    ITcVnContainer*         ipRefKeyPoints,
    ITcVnImage*             ipRefDescriptors,
    ULONG                   nRefImageWidth,
    ULONG                   nRefImageHeight,
    ITcVnContainer*&        ipEdgePoints,
    float                   fMaxDist,
    float                   fMaxKnnRatio,
    TcVnMatrix3x3_LREAL&    aPerspectiveTransform,
    ULONG&                  nNumberOfGoodMatches,
    ULONG&                  nNumberOfInliers,
    ETcVnNormalizationType  eNormType = NT_HAMMING,
    ITcVnImage*             ipSrcImageMask = nullptr,
    TcVnParamsORB&          stOrbParams = {500, 1.2f, 8, 31, 0, 2, FST_ORB_HARRIS, 31, 20},
    ETcVnEstimationAlgorithm eAlgorithm = EA_RANSAC,
    double                  fReprojThreshold = 3,
    ULONG                   nMaxIterations = 2000,
    double                  fConfidence = 0.995
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipRefKeyPoints	ITcVnContainer* [▶ 1827]		参考图像的关键点，例如使用 F_VN_KeyPointsAndDescriptorsORB（ContainerType_Vector_TcVnKeyPoint）计算得出的关键点

名称	类型	默认值	描述
ipRefDescriptors	ITcVnImage* [▶ 1863]		参考图像关键点的描述符，例如使用 F_VN_KeyPointsAndDescriptorsORB 计算得出的描述符
nRefImageWidth	ULONG		参考图像的宽度，单位：像素
nRefImageHeight	ULONG		参考图像的高度，单位：像素
ipEdgePoints	ITcVnContainer* [▶ 1827]&		返回一个容器，其中包含转换为 ipSrcImage 坐标的 4 个参考图像边缘点（ContainerType_Vector_TcVnPoint2_REAL）
fMaxDist	float		允许的最大描述符距离（-1 禁用此过滤标准）
fMaxKnnRatio	float		第一和第二最佳匹配结果之间允许的最大距离比（[0...1]，-1 则禁用此过滤标准）
aPerspectiveTransform	TcVnMatrix3x3_LREAL [▶ 1641]&		返回透视变换矩阵，该矩阵将参考点变换为源点
nNumberOfGoodMatches	ULONG&		返回良好匹配项的数量
nNumberOfInliers	ULONG&		返回内群匹配项的数量
eNormType	ETcVnNormalizationType [▶ 1684]	NT_HAMMING	用于描述符匹配的归一化类型（建议 ORB 使用 HAMMING，如果 ORB nBriefPoints 参数为 3 或 4，则使用 HAMMING2）
ipSrcImageMask	ITcVnImage* [▶ 1863]	nullptr	该掩膜用于指定在哪里查找 ipSrcImage 的关键点（如果不需要掩膜，则设为 0）
stOrbParams	TcVnParamsORB [▶ 1714]&	{500, 1.2f, 8, 31, 0, 2, FST_ORB_HARRIS, 31, 20}	用于为 ipSrcImage 配置关键点和描述符计算的参数（产生的描述符必须与 ipRefDescriptors 兼容！）
eAlgorithm	ETcVnEstimationAlgorithm [▶ 1674]	EA_RANSAC	用于计算两个点集之间透视变换的估计算法
fReprojThreshold	double	3	将点对视为内群时允许的最大重投影误差（仅在 eAlgorithm 为 RANSAC、RHO 时使用）
nMaxIterations	ULONG	2000	最大 RANSAC 迭代次数
fConfidence	double	0.995	置信度（0..1）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.6 GetMatchCoordinates

返回相互匹配的关键点的坐标。

语法

定义：

```
HRESULT GetMatchCoordinates(
    HRESULT          hrPrev,
    ITcVnContainer*  ipQueryKeyPoints,
    ITcVnContainer*  ipTrainKeyPoints,
    ITcVnContainer*  ipMatches,
    ITcVnContainer*& ipQueryCoordinates,
    ITcVnContainer*& ipTrainCoordinates
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipQueryKeyPoints	ITcVnContainer* [▶ 1827]	包含 CTcVnContainer_Vector_TcVnKeyPoint 类型的查询关键点（即参考图像/模板图像的关键点）的容器
ipTrainKeyPoints	ITcVnContainer* [▶ 1827]	包含 CTcVnContainer_Vector_TcVnKeyPoint 类型的训练关键点（即源图像/输入图像的关键点）的容器
ipMatches	ITcVnContainer* [▶ 1827]	查询关键点与训练关键点匹配的容器（CTcVnContainer_Vector_TcVnDMatch）。
ipQueryCoordinates	ITcVnContainer* [▶ 1827]&	返回一个包含关键点坐标的容器，这些关键点同时存在于查询关键点和 ipMatches 中（CTcVnContainer_Vector_TcVnPoint2_REAL）
ipTrainCoordinates	ITcVnContainer* [▶ 1827]&	返回一个包含关键点坐标的容器，这些关键点同时存在于训练关键点和 ipMatches 中（CTcVnContainer_Vector_TcVnPoint2_REAL）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.7 KeyPointsAGAST

使用 AGAST 算法检测关键点。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT KeyPointsAGAST(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipKeyPoints,
    ITcVnImage*      ipMask = nullptr,
    TcVnParamsAGAST& stParams = {10, true, KDT_AGAST_9_16}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipKeyPoints	ITcVnContainer* [▶ 1827]&		返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipMask	ITcVnImage* [▶ 1863]	nullptr	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
stParams	TcVnParamsAGAST [▶ 1708]&	{10, true, KDT_AGAST_9_16}	其它专家参数

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.8 KeyPointsAndDescriptorsAKAZE

使用 AKAZE 算法检测关键点并计算描述符。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT KeyPointsAndDescriptorsAKAZE(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipKeyPoints,
    ITcVnImage*&     ipDescriptors,
    ITcVnImage*      ipMask = nullptr,
    TcVnParamsAKAZE& stParams = {FDT_AKAZE_MLDB, 0, 3, 0.001f, 2, 1, DT1_KAZE_PM_G2}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipKeyPoints	ITcVnContainer* [▶ 1827]&		返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipDescriptors	ITcVnImage* [▶ 1863]&		描述符图像（如果不需要，则设为 0；每行 1 个描述符；如有需要，将创建适当的图像。）
ipMask	ITcVnImage* [▶ 1863]	nullptr	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
stParams	TcVnParamsAKAZE [▶ 1708]&	{FDT_AKAZE_MLDB, 0, 3, 0.001f, 2, 1, DT1_KAZE_PM_G2}	其它专家参数

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.9 KeyPointsAndDescriptorsBRISK

使用 BRISK 算法检测关键点并计算描述符。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT KeyPointsAndDescriptorsBRISK(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipKeyPoints,
    ITcVnImage*&     ipDescriptors,
    ITcVnImage*      ipMask = nullptr,
    TcVnParamsBRISK& stParams = {30, 3, 1}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipKeyPoints	ITcVnContainer* [▶ 1827]&		返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipDescriptors	ITcVnImage* [▶ 1863]&		描述符图像（如果不需要，则设为 0；每行 1 个描述符；如有需要，将创建适当的图像。）
ipMask	ITcVnImage* [▶ 1863]	nullptr	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
stParams	TcVnParamsBRISK [▶ 1711]&	{30, 3, 1}	其它专家参数

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.10 KeyPointsAndDescriptorsKAZE

使用 KAZE 算法检测关键点并计算描述符。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT KeyPointsAndDescriptorsKAZE(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipKeyPoints,
    ITcVnImage*&     ipDescriptors,
    ITcVnImage*      ipMask = nullptr,
    TcVnParamsKAZE&  stParams = {false, false, 0.001f, 4, 2, DT1_KAZE_PM_G2}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipKeyPoints	ITcVnContainer* [▶ 1827]&		返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipDescriptors	ITcVnImage* [▶ 1863]&		描述符图像（如果不需要，则设为 0；每行 1 个描述符；如有需要，将创建适当的图像。）
ipMask	ITcVnImage* [▶ 1863]	nullptr	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
stParams	TcVnParamsKAZE [▶ 1713]&	{false, false, 0.001f, 4, 2, DT1_KAZE_PM_G2}	其它专家参数

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.11 KeyPointsAndDescriptorsORB

使用 ORB 算法检测关键点并计算描述符。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT KeyPointsAndDescriptorsORB(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipKeyPoints,
    ITcVnImage*&     ipDescriptors,
    ITcVnImage*      ipMask = nullptr,
    TcVnParamsORB&   stParams = {500, 1.2f, 8, 31, 0, 2, FST_ORB_HARRIS, 31, 20}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipKeyPoints	ITcVnContainer* [▶ 1827]&		返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipDescriptors	ITcVnImage* [▶ 1863]&		描述符图像（如果不需要，则设为 0；每行 1 个描述符；如有需要，将创建适当的图像。）
ipMask	ITcVnImage* [▶ 1863]	nullptr	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
stParams	TcVnParamsORB [▶ 1714]&	{500, 1.2f, 8, 31, 0, 2, FST_ORB_HARRIS, 31, 20}	其它专家参数

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.12 KeyPointsFAST

使用 FAST 算法检测关键点。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT KeyPointsFAST(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipKeyPoints,
```

```
    ITcVnImage*       ipMask = nullptr,
    TcVnParamsFAST&   stParams = {10, true, KDT_FAST_9_16}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipKeyPoints	ITcVnContainer* [▶ 1827]&		返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipMask	ITcVnImage* [▶ 1863]	nullptr	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
stParams	TcVnParamsFAST [▶ 1712]&	{10, true, KDT_FAST_9_16}	其它专家参数

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.13 KeyPointsGFTT

使用 GFTT 算法检测关键点，该算法可检测强角点。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT KeyPointsGFTT(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipKeyPoints,
    ITcVnImage*      ipMask = nullptr,
    TcVnParamsGFTT&  stParams = {1000, 0.01, 1, 3, false, 0.04}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipKeyPoints	ITcVnContainer* [▶ 1827]&		返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipMask	ITcVnImage* [▶ 1863]	nullptr	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
stParams	TcVnParamsGFTT [▶ 1712]&	{1000, 0.01, 1, 3, false, 0.04}	其它专家参数

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.14 KeyPointsMSER

使用 MSER 算法检测关键点。

语法

定义：

```
HRESULT KeyPointsMSER(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipKeyPoints,
    ITcVnImage*      ipMask = nullptr,
    TcVnParamsMSER&  stParams = {5, 60, 14400, 0.25, 0.2, 200, 1.01, 0.003, 5}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像

名称	类型	默认值	描述
ipKeyPoints	ITcVnContainer* [▶ 1827]&		返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）
ipMask	ITcVnImage* [▶ 1863]	nullptr	该掩膜用于指定在哪里查找关键点（如果不需要掩膜，则设为 0）
stParams	TcVnParamsMSER [▶ 1713]&	{5, 60, 14400, 0.25, 0.2, 200, 1.01, 0.003, 5}	其它专家参数

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.15 KeyPointsSB

使用简单斑点检测方法检测关键点 - 多次迭代对源图像应用不同的阈值 - 检测连通分量（斑点） - 斑点的中心和半径作为关键点返回
可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
HRESULT KeyPointsSB(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipKeyPoints,
    ITcVnImage*      ipMask = nullptr,
    TcVnParamsSB&    stParams = {true, false, false, false, false, 25, 15000, 0, 1, 255, 0, 1, 0, 1,
5, 2, 30, 225, 10}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipKeyPoints	ITcVnContainer* [▶ 1827]&		返回一个以关键点填充的容器（ContainerType_Vector_TcVnKeyPoint；重复使用非零接口指针。）

名称	类型	默认值	描述
ipMask	ITcVnImage* [▶ 1863]	nullptr	该掩膜用于指定在哪里查找关键点（可选，如果不需要，则设为 0）
stParams	TcVnParamsSB [▶ 1715]&	{true, false, false, false, false, 25, 15000, 0, 1, 255, 0, 1, 0, 1, 5, 2, 30, 225, 10}	用于过滤检测到的斑点的几个参数

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.16 MatchDescriptorsBF

使用蛮力法（Brute Force）匹配描述符。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT MatchDescriptorsBF(
    HRESULT                hrPrev,
    ITcVnImage*            ipQueryDescriptors,
    ITcVnImage*            ipTrainDescriptors,
    ITcVnContainer*&       ipMatches,
    ITcVnImage*            ipMask = nullptr,
    ETcVnNormalizationType eNormType = NT_L2,
    bool                   bCrossCheck = false
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipQueryDescriptors	ITcVnImage* [▶ 1863]		查询描述符（即参考图像/模板图像的描述符）
ipTrainDescriptors	ITcVnImage* [▶ 1863]		训练描述符（即源图像/输入图像的描述符）
ipMatches	ITcVnContainer* [▶ 1827]&		返回一个以描述符匹配项填充的容器（ContainerType_Vector_TcVnDMatch；重复使用非零接口指针。）

名称	类型	默认值	描述
ipMask	ITcVnImage* [▶ 1863]	nullptr	用于指定允许匹配的掩膜，即仅当 mask[i][j] != 0 时，query[i] 才能与 train[j] 匹配（如果不需要掩膜，则设为 0）
eNormType	ETcVnNormalizationType [▶ 1684]	NT_L2	归一化类型（仅支持 L1、L2、L2SQR、HAMMING、HAMMING2）。如果 ORB nBriefPoints 参数为 3 或 4，则 ORB 描述符应使用 HAMMING2
bCrossCheck	BOOL	false	如果为 true，则只返回一致的匹配结果，即 query->train 和 train->query 检测到的匹配结果相同

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.17 MatchDescriptorsFlannLsh

使用基于 FLANN 的方法与 LSH 索引匹配描述符。

语法

定义：

```
HRESULT MatchDescriptorsFlannLsh(
    HRESULT          hrPrev,
    ITcVnImage*      ipQueryDescriptors,
    ITcVnImage*      ipTrainDescriptors,
    ITcVnContainer*& ipMatches,
    ULONG            nTableNumber = 12,
    ULONG            nKeySize = 20,
    ULONG            nMultiProbeLevel = 2,
    ULONG            nChecks = 32
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipQueryDescriptors	ITcVnImage* [▶ 1863]		查询描述符（即参考图像/模板图像的描述符）

名称	类型	默认值	描述
ipTrainDescriptors	ITcVnImage* [▶ 1863]		训练描述符（即源图像/输入图像的描述符）
ipMatches	ITcVnContainer* [▶ 1827]&		返回一个以描述符匹配项填充的容器（ContainerType_Vector_TcVnDMatch；重复使用非零接口指针。）
nTableNumber	ULONG	12	表格数量
nKeySize	ULONG	20	关键点大小
nMultiProbeLevel	ULONG	2	多探针级别
nChecks	ULONG	32	搜索邻区时访问的最大叶节点数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.18 MatchDescriptorsKnnBF

使用蛮力法（Brute Force）匹配描述符（k-NN 最近邻算法）。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT MatchDescriptorsKnnBF(
    HRESULT                 hrPrev,
    ITcVnImage*             ipQueryDescriptors,
    ITcVnImage*             ipTrainDescriptors,
    ITcVnContainer*&        ipMatches,
    ULONG                   nK,
    ITcVnImage*             ipMask = nullptr,
    bool                    bCompactResult = false,
    ETcVnNormalizationType  eNormType = NT_L2,
    bool                    bCrossCheck = false
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipQueryDescriptors	ITcVnImage* [▶ 1863]		查询描述符（即参考图像/模板图像的描述符）

名称	类型	默认值	描述
ipTrainDescriptors	ITcVnImage* [▶ 1863]		训练描述符（即源图像/输入图像的描述符）
ipMatches	ITcVnContainer* [▶ 1827]&		返回一个以描述符匹配项填充的容器（ContainerType_Vector_Vector_TcVnDMatch；重复使用非零接口指针。）
nK	ULONG		每个查询描述符所需的最佳训练匹配项数量（即 nk := 2）
ipMask	ITcVnImage* [▶ 1863]	nullptr	用于指定允许匹配的掩膜，即仅当 mask[i][j] != 0 时，query[i] 才能与 train[j] 匹配（如果不需要掩膜，则设为 0）
bCompactResult	BOOL	false	如果为 true，则匹配向量不包含完全屏蔽的查询描述符条目
eNormType	ETcVnNormalizationType [▶ 1684]	NT_L2	归一化类型（仅支持 L1、L2、L2SQR、HAMMING、HAMMING2）。如果 ORB nBriefPoints 参数为 3 或 4，则 ORB 描述符应使用 HAMMING2
bCrossCheck	BOOL	false	如果为 true，则只返回一致的匹配结果，即 query->train 和 train->query 检测到的匹配结果相同（仅在 nK = 1 时使用）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.19 MatchDescriptorsKnnFlannLsh

使用基于 FLANN 的方法与 LSH 索引匹配描述符（k-NN 最近邻算法）。

语法

定义：

```
HRESULT MatchDescriptorsKnnFlannLsh(
    HRESULT          hrPrev,
    ITcVnImage*      ipQueryDescriptors,
```

```
    ITcVnImage*      ipTrainDescriptors,
    ITcVnContainer*& ipMatches,
    ULONG            nK,
    ULONG            nTableNumber = 12,
    ULONG            nKeySize = 20,
    ULONG            nMultiProbeLevel = 2,
    ULONG            nChecks = 32
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipQueryDescriptors	ITcVnImage* [▶ 1863]		查询描述符（即参考图像/模板图像的描述符）
ipTrainDescriptors	ITcVnImage* [▶ 1863]		训练描述符（即源图像/输入图像的描述符）
ipMatches	ITcVnContainer* [▶ 1827]&		返回一个以描述符匹配项填充的容器（ContainerType_Vector_Vector_TcVnDMatch；重复使用非零接口指针。）
nK	ULONG		每个查询描述符所需的最佳训练匹配项数量（即 nk := 2）
nTableNumber	ULONG	12	表格数量
nKeySize	ULONG	20	关键点大小
nMultiProbeLevel	ULONG	2	多探针级别
nChecks	ULONG	32	搜索邻区时访问的最大叶节点数量

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.19.20 RegionsMSER

使用 MSER 算法检测区域。

语法

定义：

```
HRESULT RegionsMSER(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnContainer*& ipRegions,
    ITcVnContainer*& ipBoundingBoxes,
    TcVnParamsMSER&  stParams = {5, 60, 14400, 0.25, 0.2, 200, 1.01, 0.003, 5}
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像
ipRegions	ITcVnContainer* [▶ 1827]&		返回一个以区域点填充的容器（ContainerType_Vector_Vector_TcVnPoint2_DINT；重复使用非零接口指针。）
ipBoundingBoxes	ITcVnContainer* [▶ 1827]&		返回一个以区域边界框填充的容器（ContainerType_Vector_TcVnRectangle_DINT；重复使用非零接口指针。）
stParams	TcVnParamsMSER [▶ 1713]&	{5, 60, 14400, 0.25, 0.2, 200, 1.01, 0.003, 5}	其它专家参数

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20 机器学习

该组包含用于创建、培训和应用机器学习模型的函数，以及用于特征预处理的函数。关于机器学习 [▶ 138]的可能应用的简要概述和进一步说明可在软件概念 [▶ 132]下找到。

模型创建函数：

- CreateBoostClassifier [▶ 2511]
- CreateKmppModel [▶ 2512]
- CreateKnnModel [▶ 2513]
- CreateLbgModel [▶ 2513]
- CreateLdaTransform [▶ 2515]

- CreateLdaTransformViaComponentNum [▶ 2515]
- CreateNbcModel [▶ 2516]
- CreatePcaTransform [▶ 2517]
- CreatePcaTransformViaComponentNum [▶ 2518]
- CreatePcaTransformViaVariance [▶ 2519]
- CreateRTreesModel [▶ 2519]
- CreateStaModel [▶ 2521]
- CreateSvmModel [▶ 2522]
- CreateSvmSgdClassifier [▶ 2524]

模型训练函数：

- TrainBatch [▶ 2540]
- TrainBatchClusters [▶ 2541]
- TrainSample [▶ 2542]
- TrainSampleClass [▶ 2542]
- TrainSampleCluster [▶ 2543]
- TrainSampleScalar [▶ 2544]
- TrainSampleVector [▶ 2544]

用于获取聚类模型上信息的函数：

- GetClusterCenter [▶ 2529]
- GetClusterNum [▶ 2530]

将模型应用于样本的函数：

- GetSampleCluster [▶ 2531]
- GetSampleNovelty [▶ 2532]
- PredictSampleClass [▶ 2537]
- PredictSampleScalar [▶ 2538]
- PredictSampleVector [▶ 2539]

将模型应用于批处理的函数：

- GetBatchClusters [▶ 2527]
- GetBatchNovelties [▶ 2528]
- PredictBatch [▶ 2536]

将模型应用于功能转换/降维的函数：

- FeatureTransform [▶ 2526]
- InverseFeatureTransform [▶ 2535]

功能标准化函数：

- GetFeatureScales [▶ 2530]
- FeatureScaling [▶ 2525]
- InverseFeatureScaling [▶ 2533]
- InverseFeatureScaling_REAL [▶ 2534]

还请参阅有关此

GetTransformParameters [▶ 2545]

6.2.3.20.1 CreateBoostClassifier

创建 Boost 分类器。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。Boost 分类器只适用于二分类问题。它通过逐步添加弱分类器来改进分类结果，从而学会区分贴有用户定义的两个类别标签的样本。此类模型既不支持在线训练（逐样本训练），也不支持再训练。

语法

定义：

```
HRESULT CreateBoostClassifier(
    HRESULT                 hrPrev,
    ITcVnMlModel*&          ipMlModel,
    ETcVnBoostClassifierType eType = BCT_REAL,
    ULONG                   nMaxDepth = 1,
    ULONG                   nMinSamples = 10,
    ULONG                   nWeakClassifiers = 100,
    double                  fWeightTrimRate = 0.95,
    ITcVnContainer*         ipClassPriors = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]&		返回创建的模型（重复使用非零接口指针。）
eType	ETcVnBoostClassifierType [▶ 1647]	BCT_REAL	学习算法类型（默认类型：TCVN_BCT_REAL）
nMaxDepth	ULONG	1	树模型最大深度（默认值：1）
nMinSamples	ULONG	10	节点内需要分割的最小样本数（默认值：10）
nWeakClassifiers	ULONG	100	弱分类器数量（默认：100）
fWeightTrimRate	double	0.95	训练时使用的权重阈值（关闭：0；默认：0.95）。
ipClassPriors	ITcVnContainer* [▶ 1827]	nullptr	类优先级（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL；仅适用于分类器；可选，如果不需要或不允许，则设为 0；默认值：0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.2 CreateKmppModel

创建指定类型的 k-means++ 模型。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。此类模型既不支持在线训练（逐样本训练），也不支持再训练。可选择提供预测的异常程度。

语法

定义：

```
HRESULT CreateKmppModel(
    HRESULT                 hrPrev,
    ITcVnMlModel*&          ipMlModel,
    ETcVnPrototypeClusterer eKmppType,
    ULONG                   nK,
    bool                    bDoublePrecision = false,
    ULONG                   nMaxIterations = 0,
    double                  fEpsilon = 0.0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]&		返回创建的模型（重复使用非零接口指针。）
eKmppType	ETcVnPrototypeClusterer [▶ 1688]		k-means++ 模型类型
nK	ULONG		参数 k（聚类数）
bDoublePrecision	BOOL	false	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成模型。（默认设置：FALSE）
nMaxIterations	ULONG	0	最大迭代次数（如果等于 0，则使用默认值 10）
fEpsilon	double	0.0	两次连续迭代之间允许的最大误差差值（如果差值等于 0，则使用默认值 0.001）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.3 CreateKnnModel

创建指定类型的 k-NN 模型。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。这类模型支持在线训练（逐样本）和再训练。预测值只能是标量。可选择 k 近邻模型提供预测的异常程度。

语法

定义：

```
HRESULT CreateKnnModel(
    HRESULT         hrPrev,
    ITcVnMlModel*& ipMlModel,
    ETcVnKnn        eKnnType,
    ULONG           nK
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]&	返回创建的模型（重复使用非零接口指针。）
eKnnType	ETcVnKnn [▶ 1681]	k-NN 模型类型
nK	ULONG	用于预测的参数 k（考虑的邻区数量）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.4 CreateLbgModel

创建指定类型的 LBG 模型。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。此类模型既不支持在线训练（逐样本训练），也不支持再训练。可选择提供预测的异常程度。

语法

定义：

```
HRESULT CreateLbgModel(
    HRESULT                 hrPrev,
    ITcVnMlModel*&          ipMlModel,
    ETcVnPrototypeClusterer eLbgType,
    ULONG                   nMaxClusters,
```

```
    double                  fMaxClusterRadius,
    bool                    bSingleSplitSteps,
    bool                    bDoublePrecision = false,
    ULONG                   nMaxIterations = 0,
    double                  fEpsilon = 0.0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]&		返回创建的模型（重复使用非零接口指针。）
eLbgType	ETcVnPrototypeClusterer [▶ 1688]		LBG 模型类型
nMaxClusters	ULONG		最大聚类数
fMaxClusterRadius	double		单个聚类允许的最大半径（L2 范数），即聚类半径越大，被分割成的聚类越小，直到达到 nMaxClusters 的全局数量。
bSingleSplitSteps	BOOL		如果为 true，则总是在拆分单个聚类后运行全局优化。如果为 false，则会在应用全局优化之前，在同一步骤中拆分多个聚类。减少全局优化的应用次数会加快速度，但可能会导致结果不理想，尤其是有 2 个可由 1 表示的邻近聚类时。
bDoublePrecision	BOOL	false	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成模型。（默认设置：FALSE）
nMaxIterations	ULONG	0	最大迭代次数（如果等于 0，则使用默认值 10）
fEpsilon	double	0.0	两次连续迭代之间允许的最大误差差值（如果差值等于 0，则使用默认值 0.001）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.5 CreateLdaTransform

根据提供的数据创建基于 LDA 的特征变换。样本数必须 >= 功能数，而且类别数必须 >= 2。除此以外，如果创建并保留新模型，则初始引用计数被设置为 1。

语法

定义：

```
HRESULT CreateLdaTransform(
    HRESULT          hrPrev,
    ITcVnMlModel*&   ipMlModel,
    ITcUnknown*      ipSamples,
    ITcVnContainer* ipClasses
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]&	返回已创建的特征变换（重复使用非零接口指针。）
ipSamples	ITcUnknown* [▶ 1877]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
ipClasses	ITcVnContainer* [▶ 1827]	与输入样本相对应的类标签（ContainerType_Vector_DINT）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.6 CreateLdaTransformViaComponentNum

根据提供的数据创建基于 LDA 的特征变换，其中使用的分量是预定义的。样本数必须 >= 功能数，而且类别数必须 >= 2。除此以外，如果创建并保留新模型，则初始引用计数被设置为 1。

语法

定义：

```
HRESULT CreateLdaTransformViaComponentNum(
    HRESULT          hrPrev,
    ITcVnMlModel*&   ipMlModel,
    ITcUnknown*      ipSamples,
    ITcVnContainer*  ipClasses,
    ULONG            nComponentNum
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]&	返回已创建的特征变换（重复使用非零接口指针。）
ipSamples	ITcUnknown* [▶ 1877]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
ipClasses	ITcVnContainer* [▶ 1827]	与输入样本相对应的类标签（ContainerType_Vector_DINT）
nComponentNum	ULONG	分量数（nComponentNum 必须小于类的数量。如果数值为 0，则自动计算最大分量数。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.7 CreateNbcModel

创建指定类型的普通贝叶斯（Bayes）分类器。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。如要训练普通贝叶斯（Bayes）分类器，每个类别都需要足够多的样本。这取决于特征的数量和数据的分布。因此，需要对每种应用进行测试。此类模型不支持在线训练（逐样本）。要对此类分类器模型进行再训练，显示的类集必须与之前的学习步骤完全相同， 否则会出现异常。可选择普通贝叶斯分类器提供预测的异常程度。

语法

定义：

```
HRESULT CreateNbcModel(
    HRESULT          hrPrev,
    ITcVnMlModel*& ipMlModel,
    ETcVnNbc         eNbcType
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]&	返回创建的模型（重复使用非零接口指针。）
eNbcType	ETcVnNbc [▶ 1683]	普通贝叶斯（Bayes）分类器类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.8 CreatePcaTransform

根据提供的数据创建基于 PCA 的特征变换。可计算的主成分的最大数等于样本数和功能数的最小值。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。

语法

定义：

```
HRESULT CreatePcaTransform(
    HRESULT        hrPrev,
    ITcVnMlModel*& ipMlModel,
    ITcUnknown*    ipSamples
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]&	返回已创建的特征变换（重复使用非零接口指针。）
ipSamples	ITcUnknown* [▶ 1877]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.9 CreatePcaTransformViaComponentNum

根据提供的数据创建基于 PCA 的特征变换，其中使用的主分量是预定义的。可计算的主成分的最大数等于样本数和功能数的最小值。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。

语法

定义：

```
HRESULT CreatePcaTransformViaComponentNum(
    HRESULT         hrPrev,
    ITcVnMlModel*& ipMlModel,
    ITcUnknown*     ipSamples,
    ULONG           nComponentNum
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]&	返回已创建的特征变换（重复使用非零接口指针。）
ipSamples	ITcUnknown* [▶ 1877]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
nComponentNum	ULONG	主分量数（如果数值为 0，则自动计算最大分量数。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.10 CreatePcaTransformViaVariance

根据需要保留的给定方差比例，从提供的数据中创建基于 PCA 的特征变换。可计算的主成分的最大数等于样本数和功能数的最小值。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。

语法

定义：

```
HRESULT CreatePcaTransformViaVariance(
    HRESULT         hrPrev,
    ITcVnMlModel*&  ipMlModel,
    ITcUnknown*     ipSamples,
    double          fRetainedVariance
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]&	返回已创建的特征变换（重复使用非零接口指针。）
ipSamples	ITcUnknown* [▶ 1877]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
fRetainedVariance	double	PCA 保留的方差比例（1.0 表示 100%。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.11 CreateRTreesModel

创建指定类型的随机树模型。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。此类模型既不支持在线训练（逐样本训练），也不支持再训练。预测值只能是标量。

语法

定义：

```
HRESULT CreateRTreesModel(
    HRESULT         hrPrev,
    ITcVnMlModel*&  ipMlModel,
    ETcVnRTrees     eRTreesType,
    ULONG           nMaxDepth = 5,
    ULONG           nMinSamples = 10,
    ULONG           nActiveVariables = 0,
    ULONG           nMaxIterations = 0,
    double          fEpsilon = 0.0,
```

```
    float           fRegressionAccuracy = 0,
    ITcVnContainer* ipClassPriors = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]&		返回创建的模型（重复使用非零接口指针。）
eRTreesType	ETcVnRTrees [▶ 1690]		随机树模型类型
nMaxDepth	ULONG	5	树模型最大深度（默认值：5）
nMinSamples	ULONG	10	节点内需要分割的最小样本数（默认值：10）
nActiveVariables	ULONG	0	分割时考虑的变量数（0 表示 sqrt（变量总数）；默认值：0）
nMaxIterations	ULONG	0	最大迭代次数（如果等于 0 且 fEpsilon 不为 0.0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 50）
fEpsilon	double	0.0	两次连续迭代之间允许的最大误差差值（如果差值等于 0.0 且 nMaxIterations 不为 0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 0.1）
fRegressionAccuracy	float	0	回归子的终止标准（仅适用于回归子；如果不允许，则设为默认值；默认值：0.0）
ipClassPriors	ITcVnContainer* [▶ 1827]	nullptr	类优先级（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL；仅适用于分类器；可选，如果不需要或不允许，则设为 0；默认值：0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.12 CreateStaModel

创建指定类型的简化 TopoART 神经网络。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。此类模型支持在线训练（逐样本）、再训练以及标量和向量预测。它要求除类标签外的所有输入（即样本和训练输出/预测）都位于区间 [0.0，1.0]。在使用前，需要将回归子的预测值从区间 [0.0，1.0] 重新调整至其各自的值范围。可选择简化 TopoART 神经网络提供预测的异常程度。根据参数设置和可用训练样本的数量，使用相同数据进行重复训练可以改善结果。与其它基于自适应共振理论（ART）的神经网络一样，简化 TopoART 神经网络不易发生灾难性推理，尤其非常适合增量学习任务。（附加专家函数，提供微调和降噪参数）

语法

定义：

```
HRESULT CreateStaModel(
    HRESULT        hrPrev,
    ITcVnMlModel*& ipMlModel,
    ETcVnSta       eStaType,
    double         fRho,
    ULONG          nNu = 3,
    bool           bDoublePrecision = false,
    double         fBetaSbm = 0.0,
    ULONG          nPhi = 1,
    ULONG          nTau = 100
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]&		返回创建的模型（重复使用非零接口指针。）
eStaType	ETcVnSta [▶ 1691]		简化 TopoART 模型类型
fRho	double		警戒参数（控制插入的神经元数量和所形成类别的大小上限；有效范围：[0.0，1.0]；建议范围：[0.8，1.0]）
nNu	ULONG	3	用于分类和预测的神经元数量（必须大于或等于 1；建议范围：[1，10]；默认值：3）
bDoublePrecision	bool	false	如果为 true，将使用双精度（LREAL）而不是单精度（REAL）生成模型。（默认设置：FALSE）
fBetaSbm	double	0.0	第二最佳匹配神经元的学习率（学习第二最佳匹配神经元可使相关类别更接近，并可能改善结果，但可能需要更多神经元并运

名称	类型	默认值	描述
			行更多训练；值为 0.0 时禁用第二最佳匹配神经元的学习；有效范围：[0.0，1.0]；建议范围：[0.0，0.5]）；默认值：0.0
nPhi	ULONG	1	一个神经元为了成为永久性神经元而必须学习的样本数量（降噪所需；与 nTau 配合使用；必须大于或等于 1；数值越大，降噪效果越强；数值为 1 时，禁止删除候选神经元；默认值：1）
nTau	ULONG	100	移除节点后执行的学习步骤数量（降噪所需；与 nPhi 配合使用；必须足够大，以便在移除节点之前，让代表非噪声样本的候选神经元成为最佳匹配神经元至少 nPhi 次；默认值：100）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.13 CreateSvmModel

创建指定类型的 SVM 模型。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。此类模型既不支持在线训练（逐样本训练），也不支持再训练。预测值只能是标量。（用于 C 支持向量分类器的附加专家函数）

语法

定义：

```
HRESULT CreateSvmModel(
    HRESULT             hrPrev,
    ITcVnMlModel*&      ipMlModel,
    ETcVnSvm            eSvmType,
    double              fC,
    double              fNu,
    double              fP,
    ETcVnSvmKernelType  eKernelType,
    double              fGamma,
    double              fCoef0,
    double              fDegree,
    ULONG               nMaxIterations = 0,
    double              fEpsilon = 0.0,
    ITcVnContainer*     ipClassWeights = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]&		返回创建的模型（重复使用非零接口指针。）
eSvmType	ETcVnSvm [▶ 1692]		SVM 模型类型
fC	double		参数 C（对于 TCVN_SVM_C_CLASSIFIER、TCVN_SVM_EPS_REGRESSOR 和 TCVN_SVM_NU_REGRESSOR 为必需参数；其它情况可忽略）
fNu	double		参数 nu（对于 TCVN_SVM_NU_CLASSIFIER、TCVN_SVM_NOVELTY_DETECTOR 和 TCVN_SVM_NU_REGRESSOR 为必需参数；其它情况可忽略）
fP	double		参数 p（对于 TCVN_SVM_EPS_REGRESSOR 为必需参数，其它情况可忽略）
eKernelType	ETcVnSvmKernelType [▶ 1692]		内核类型
fGamma	double		参数 gamma（多项式核、RBF 核、S 形核和卡方核使用；其它情况下忽略不计）
fCoef0	double		参数 coef0（多项式核和 S 形核使用；其它情况下忽略不计）
fDegree	double		度数（多项式核使用，其它情况下可忽略）
nMaxIterations	ULONG	0	最大迭代次数（如果等于 0 且 fEpsilon 不为 0.0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 100000）
fEpsilon	double	0.0	两次连续迭代之间允许的最大误差差值（如果差值等于 0.0 且 nMaxIterations 不为 0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 0.00001）

名称	类型	默认值	描述
ipClassWeights	ITcVnContainer* [▶ 1827]	nullptr	类权重（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL；仅当 eSvmType 等于 TCVN_SVM_C_CLASSIFIER 时有效；可选，如果不需要或不允许，则设为 0；默认值：0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.14 CreateSvmSgdClassifier

创建一个线性 SVM 分类器，使用随机梯度下降算法进行训练。如果创建了一个新模型，初始引用计数将设为 1，如未创建，保持原计数不变。这种 SVM 分类器只适用于二分类问题。它习得标签为 -1 的类和标签为 1 的类之间的一个分离超平面。这些类标签是预定义的。在训练中，任何正类标签都会被映射为 1，而任何负类标签都会被映射为 -1。此类模型既不支持在线训练（逐样本训练），也不支持再训练。

语法

定义：

```
HRESULT CreateSvmSgdClassifier(
    HRESULT                          hrPrev,
    ITcVnMlModel*&                   ipMlModel,
    ETcVnSvmSgdClassifierType        eType = SSCT_ASGD,
    ETcVnSvmSgdClassifierMarginType  eMarginType = SSCMT_SOFT_MARGIN,
    float                            fMarginRegularization = 0.00001f,
    float                            fInitialStepSize = 0.05f,
    float                            fStepDecreasingPower = 0.75f,
    ULONG                            nMaxIterations = 0,
    double                           fEpsilon = 0.0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]&		返回创建的模型（重复使用非零接口指针。）

名称	类型	默认值	描述
eType	ETcVnSvmSgdClassifierType [▶ 1693]	SSCT_ASGD	学习算法类型（默认类型：TCVN_SSCT_ASGD）
eMarginType	ETcVnSvmSgdClassifierMarginType [▶ 1693]	SSCMT_SOFT_MARGIN	边际类型（默认类型：TCVN_SSCMT_SOFT_MARGIN）
fMarginRegularization	float	0.00001f	边际正则化参数（默认值：0.00001）
fInitialStepSize	float	0.05f	初始步长（默认值：0.05）
fStepDecreasingPower	float	0.75f	幂参数（默认值：0.75）
nMaxIterations	ULONG	0	最大迭代次数（如果等于 0 且 fEpsilon 不为 0.0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 100000）
fEpsilon	double	0.0	两次连续迭代之间允许的最大误差差值（如果差值等于 0.0 且 nMaxIterations 不为 0，则禁用；如果 nMaxIterations 和 fEpsilon 等于 0，则使用默认值 0.00001）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.15 FeatureScaling

对一个或多个样本进行特征缩放，如果是 TCVN_FST1_MINMAX，则输出范围在 [fA，fB] 之间。

语法

定义：

```
HRESULT FeatureScaling(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSamples,
    ITcVnContainer*  ipScales,
    ITcVnContainer*& ipScaledSamples,
    double           fA = 0.0,
    double           fB = 1.0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSamples	ITcVnContainer* [▶ 1827]		包含一个或多个输入样本的容器（ContainerType_Vector_REAL、ContainerType_Vector_LREAL、ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_LREAL）
ipScales	ITcVnContainer* [▶ 1827]		包含每个特征的缩放参数的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）。
ipScaledSamples	ITcVnContainer* [▶ 1827]&		使用与 ipSamples 相同的类型返回缩放后的样本。如果使用同一个容器 ipSamples，源数据将被替换。
fA	double	0.0	表示 TCVN_FST1_MINMAX 范围的下限
fB	double	1.0	表示 TCVN_FST1_MINMAX 范围的上限

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.16 FeatureTransform

对一个或多个样本进行特征变换。

语法

定义：

```
HRESULT FeatureTransform(
    HRESULT       hrPrev,
    ITcVnMlModel* ipFeatureTransform,
    ITcUnknown*   ipSamples,
    ITcUnknown**  pipTransformed
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipFeatureTransform	ITcVnMlModel* [▶ 1871]	要使用的特征变换实例
ipSamples	ITcUnknown* [▶ 1877]	包含一个或多个输入样本的容器（ContainerType_Vector_REAL、ContainerType_Vector_LREAL、ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_LREAL）
pipTransformed	ITcUnknown* [▶ 1877]*	使用与 ipSamples 相同的类型返回变换后的样本

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.17 GetBatchClusters

获取一批样本中匹配度最高的聚类的 ID。

语法

定义：

```
HRESULT GetBatchClusters(
    HRESULT          hrPrev,
    ITcVnMlModel*    ipClusterer,
    ITcUnknown*      ipSamples,
    ITcVnContainer*& ipClusters,
    ITcVnContainer** pipNovelties = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipClusterer	ITcVnMlModel* [▶ 1871]		要使用的聚类器
ipSamples	ITcUnknown* [▶ 1877]		包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
ipClusters	ITcVnContainer* [▶ 1827]&		返回聚类 ID（ContainerType_Vector_DINT）
pipNovelties	ITcVnContainer* [▶ 1827]*	nullptr	返回每个样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）（ContainerType_Vector_REAL；可选，如果不需要，则设为 0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.18 GetBatchNovelties

获取多个样本的异常程度。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT GetBatchNovelties(
    HRESULT          hrPrev,
    ITcVnMlModel*    ipNoveltyDetector,
    ITcUnknown*      ipSamples,
    ITcVnContainer*& ipNovelties
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipNoveltyDetector	ITcVnMlModel* [▶ 1871]	要使用异常探测器
ipSamples	ITcUnknown* [▶ 1877]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
ipNovelties	ITcVnContainer* [▶ 1827]&	返回每个样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）（ContainerType_Vector_REAL）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.19 GetClusterCenter

获取聚类的中心。

语法

定义：

```
HRESULT GetClusterCenter(
    HRESULT          hrPrev,
    ITcVnMlModel*    ipClusterer,
    LONG             nCluster,
    ITcVnContainer*& ipCenter
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipClusterer	ITcVnMlModel* [▶ 1871]	要使用的聚类器
nCluster	LONG	聚类 ID，需要该聚类的中心
ipCenter	ITcVnContainer* [▶ 1827]&	返回聚类中心（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL，取决于聚类器内部使用的浮点类型）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.20 GetClusterNum

获取该聚类器使用的聚类数目。

语法

定义：

```
HRESULT GetClusterNum(
    HRESULT        hrPrev,
    ITcVnMlModel* ipClusterer,
    ULONGLONG&     nClusterNum
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipClusterer	ITcVnMlModel* [▶ 1871]	要使用的聚类器
nClusterNum	ULONGLONG&	返回聚类数目

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.21 GetFeatureScales

根据缩放类型计算输入样本中每个特征的缩放参数。

语法

定义：

```
HRESULT GetFeatureScales(
    HRESULT                 hrPrev,
    ITcVnContainer*         ipSamples,
    ITcVnContainer*&        ipScales,
    ETcVnFeatureScalingType eFeatureScalingType
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSamples	ITcVnContainer* [▶ 1827]	包含输入样本的容器（ContainerType_Vector_REAL、ContainerType_Vector_LREAL、ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_LREAL）
ipScales	ITcVnContainer* [▶ 1827]&	返回一个容器，其中包含每个特征的缩放参数（Vector_REAL 或 Vector_LREAL，取决于 ipSamples 的类型）。
eFeatureScalingType	ETcVnFeatureScalingType [▶ 1676]	特征缩放类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.22 GetSampleCluster

获取单个样本的最佳匹配聚类 ID。

语法

定义：

```
HRESULT GetSampleCluster(
    HRESULT       hrPrev,
    ITcVnMlModel* ipClusterer,
    ITcUnknown*   ipSample,
    LONG&         nCluster,
    float*        pNovelty = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipClusterer	ITcVnMlModel* [▶ 1871]		要使用的聚类器
ipSample	ITcUnknown* [▶ 1877]		包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
nCluster	LONG&		返回样本所属聚类的 ID
pNovelty	float*	nullptr	返回所显示样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）（可选，如果不需要，则设为 0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.23 GetSampleNovelty

获取单个样本的异常程度。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT GetSampleNovelty(
    HRESULT       hrPrev,
    ITcVnMlModel* ipNoveltyDetector,
    ITcUnknown*   ipSample,
    float&        fNovelty
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipNoveltyDetector	ITcVnMlModel* [▶ 1871]	要使用异常探测器

名称	类型	描述
ipSample	ITcUnknown* [▶ 1877]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
fNovelty	float&	返回所显示样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.24 InverseFeatureScaling

对一个或多个样本进行特征缩放的逆变换。如果使用 TCVN_FST1_MINMAX（预定输出范围）进行缩放，则应使用 [fA，fB]。

语法

定义：

```
HRESULT InverseFeatureScaling(
    HRESULT          hrPrev,
    ITcVnContainer*  ipSamples,
    ITcVnContainer*  ipScales,
    ITcVnContainer*& ipInverseSamples,
    double           fA = 0.0,
    double           fB = 1.0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSamples	ITcVnContainer* [▶ 1827]		包含一个或多个输入样本的容器（ContainerType_Vector_REAL、ContainerType_Vector_LREAL、ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_LREAL）

名称	类型	默认值	描述
ipScales	ITcVnContainer* [▶ 1827]		包含每个特征的缩放参数的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）。
ipInverseSamples	ITcVnContainer* [▶ 1827]&		使用 ipScales 的缩放类型和参数返回逆样本。它与 ipSamples 类型相同。如果使用同一个容器 ipSamples，源数据将被替换。
fA	double	0.0	表示 TCVN_FST1_MINMAX 范围的下限
fB	double	1.0	表示 TCVN_FST1_MINMAX 范围的上限

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.25 InverseFeatureScaling (float)

对单个值进行特征缩放的逆变换。如果使用 TCVN_FST1_MINMAX（预定输出范围）进行缩放，则应使用 [fA，fB]。

语法

定义：

```
HRESULT InverseFeatureScaling(
    HRESULT         hrPrev,
    float           fSample,
    ITcVnContainer* ipScales,
    float&          fInverseSample,
    double          fA = 0.0,
    double          fB = 1.0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	默认值	描述
fSample	float		样本值
ipScales	ITcVnContainer* [▶ 1827]		包含每个特征的缩放参数的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）。
fInverseSample	float&		返回逆样本。
fA	double	0.0	表示 TCVN_FST1_MINMAX 范围的下限
fB	double	1.0	表示 TCVN_FST1_MINMAX 范围的上限

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.26 InverseFeatureTransform

对一个或多个样本进行特征逆变换。

语法

定义：

```
HRESULT InverseFeatureTransform(
    HRESULT       hrPrev,
    ITcVnMlModel* ipFeatureTransform,
    ITcUnknown*   ipSamples,
    ITcUnknown**  pipTransformed
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipFeatureTransform	ITcVnMlModel* [▶ 1871]	要使用的特征变换实例
ipSamples	ITcUnknown* [▶ 1877]	包含一个或多个输入样本的容器（ContainerType_Vector_REAL、ContainerType_Vector_LREAL、ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_LREAL）

名称	类型	描述
pipTransformed	ITcUnknown* [▶ 1877]*	使用与 ipSamples 相同的类型返回变换后的样本

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.27 PredictBatch

计算一批样本的预测值。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT PredictBatch(
    HRESULT           hrPrev,
    ITcVnMlModel*     ipMlModel,
    ITcUnknown*       ipSamples,
    ITcVnContainer*& ipPredictions,
    ITcVnContainer** pipNovelties = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]		要使用的分类器或回归子
ipSamples	ITcUnknown* [▶ 1877]		包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
ipPredictions	ITcVnContainer* [▶ 1827]&		返回预测输出（取决于 ipSamples；类标签（用于分类为 ContainerType_Vector_DINT）或实值预测（用于标量输出回归为 ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL；用于矢量输出回归

名称	类型	默认值	描述
			为 ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL））
pipNovelties	ITcVnContainer* [▶ 1827]*	nullptr	返回每个样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）（ContainerType_Vector_REAL；可选，如果不需要，则设为 0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.28 PredictSampleClass

对单个样本进行分类。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT PredictSampleClass(
    HRESULT       hrPrev,
    ITcVnMlModel* ipClassifier,
    ITcUnknown*   ipSample,
    LONG&         nClass,
    float*        pNovelty = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipClassifier	ITcVnMlModel* [▶ 1871]		要使用的分类器
ipSample	ITcUnknown* [▶ 1877]		包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
nClass	LONG&		返回分类结果

名称	类型	默认值	描述
pNovelty	float*	nullptr	返回所显示样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）（可选，如果不需要，则设为 0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.29 PredictSampleScalar

计算单个样本的标量预测值。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT PredictSampleScalar(
    HRESULT       hrPrev,
    ITcVnMlModel* ipRegressor,
    ITcUnknown*   ipSample,
    float&        fPrediction,
    float*        pNovelty = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipRegressor	ITcVnMlModel* [▶ 1871]		要使用的回归子
ipSample	ITcUnknown* [▶ 1877]		包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
fPrediction	float&		返回预测的输出
pNovelty	float*	nullptr	返回所显示样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）（可选，如果不需要，则设为 0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.30 PredictSampleVector

计算单个样本的向量预测值。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT PredictSampleVector(
    HRESULT          hrPrev,
    ITcVnMlModel*    ipRegressor,
    ITcUnknown*      ipSample,
    ITcVnContainer*& ipPrediction,
    float*           pNovelty = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipRegressor	ITcVnMlModel* [▶ 1871]		要使用的回归子
ipSample	ITcUnknown* [▶ 1877]		包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
ipPrediction	ITcVnContainer* [▶ 1827]&		返回预测输出（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL，取决于 ipSample）
pNovelty	float*	nullptr	返回所显示样本的异常程度（如果样本完全已知，为 0.0；否则 > 0.0）（可选，如果不需要，则设为 0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.31 TrainBatch

使用一批样本训练分类器、聚类器、异常检测器或回归子。可在线训练模型对每个样本训练一次。根据可用训练样本的应用和数量，使用相同数据进行重复训练可以改善结果。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT TrainBatch(
    HRESULT         hrPrev,
    ITcVnMlModel*   ipMlModel,
    ITcUnknown*     ipSamples,
    ITcVnContainer* ipOutputs
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]	要使用的分类器、聚类器、异常检测器或回归子
ipSamples	ITcUnknown* [▶ 1877]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
ipOutputs	ITcVnContainer* [▶ 1827]	要学习的类标签（用于分类为 ContainerType_Vector_DINT）或目标输出（用于标量输出回归为 ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL；用于向量输出回归为 ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）（对于聚类器和异常检测器，必须为 0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.32 TrainBatchClusters

使用一批样本训练聚类器，并根据要求返回样本所属聚类的 ID。可在线训练聚类器对每个样本训练一次。根据可用训练样本的应用和数量，使用相同数据进行重复训练可以改善结果。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT TrainBatchClusters(
    HRESULT          hrPrev,
    ITcVnMlModel*    ipClusterer,
    ITcUnknown*      ipSamples,
    ITcVnContainer*& ipClusters
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipClusterer	ITcVnMlModel* [▶ 1871]	要使用的聚类器
ipSamples	ITcUnknown* [▶ 1877]	包含一批输入样本的容器（ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_Vector_LREAL）
ipClusters	ITcVnContainer* [▶ 1827]&	返回样本所属聚类的 ID（ContainerType_Vector_DINT；可选，如果不需要，则设为 0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.33 TrainSample

使用单个样本训练聚类器或异常检测器。

语法

定义：

```
HRESULT TrainSample(
    HRESULT       hrPrev,
    ITcVnMlModel* ipMlModel,
    ITcUnknown*   ipSample
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipMlModel	ITcVnMlModel* [▶ 1871]	要使用聚类器或异常检测器
ipSample	ITcUnknown* [▶ 1877]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.34 TrainSampleClass

用单个样本训练分类器。

语法

定义：

```
HRESULT TrainSampleClass(
    HRESULT       hrPrev,
    ITcVnMlModel* ipClassifier,
    ITcUnknown*   ipSample,
    LONG          nClass
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipClassifier	ITcVnMlModel* [▶ 1871]	要使用的分类器

名称	类型	描述
ipSample	ITcUnknown* [▶ 1877]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
nClass	LONG	要学习的类标签

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.35 TrainSampleCluster

使用单个样本训练聚类器，并根据要求返回样本所属聚类的 ID。聚类器必须可以在线训练。

语法

定义：

```
HRESULT TrainSampleCluster(
    HRESULT        hrPrev,
    ITcVnMlModel*  ipClusterer,
    ITcUnknown*    ipSample,
    LONG*          pCluster
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipClusterer	ITcVnMlModel* [▶ 1871]	要使用的聚类器
ipSample	ITcUnknown* [▶ 1877]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
pCluster	LONG*	返回样本所属聚类的 ID（如果指针为 0，则不需要聚类 ID。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.36 TrainSampleScalar

使用单个样本和标量输出来训练回归子。

语法

定义：

```
HRESULT TrainSampleScalar(
    HRESULT       hrPrev,
    ITcVnMlModel* ipRegressor,
    ITcUnknown*   ipSample,
    float         fOutput
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipRegressor	ITcVnMlModel* [▶ 1871]	要使用的回归子
ipSample	ITcUnknown* [▶ 1877]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
fOutput	float	要学习的标量输出

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.37 TrainSampleVector

使用单一样本和向量输出来训练回归子。

语法

定义：

```
HRESULT TrainSampleVector(
    HRESULT         hrPrev,
    ITcVnMlModel*   ipRegressor,
    ITcUnknown*     ipSample,
    ITcVnContainer* ipOutput
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipRegressor	ITcVnMlModel* [▶ 1871]	要使用的回归子
ipSample	ITcUnknown* [▶ 1877]	包含单个输入样本的容器（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）
ipOutput	ITcVnContainer* [▶ 1827]	要学习的矢量输出（ContainerType_Vector_REAL 或 ContainerType_Vector_LREAL）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.20.38 GetTransformParameters

Get the learned parameters of a feature transform model (e.g., PCA or LDA). Returns the component scores, the projection matrix, and optionally the mean vector. For PCA, the scores are eigenvalues representing the variance captured by each component. For LDA, the scores are eigenvalues representing the discriminative power of each component. Components are ordered by decreasing importance.

Syntax

Definition:

```
HRESULT GetTransformParameters(
    HRESULT          hrPrev,
    ITcVnMlModel*    ipModel,
    ITcVnContainer*& ipScores,
    ITcVnContainer*& ipProjectionVectors,
    ITcVnContainer** pipMean = nullptr
)
```

Parameters

Name	Type	Default	Description
hrPrev	HRESULT [▶ 135]		HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)
ipModel	ITcVnMlModel* [▶ 1871]		Trained feature transform model (e.g., PCA or LDA).

Name	Type	Default	Description
ipScores	ITcVnContainer* [▶ 1827]&		Container holding the component scores, sorted in descending order of importance. For PCA and LDA, these represent the eigenvalues. The container type depends on the model (ContainerType_Vector_REAL for PCA, ContainerType_Vector_LREAL for LDA).
ipProjectionVectors	ITcVnContainer* [▶ 1827]&		Container holding the projection vectors, where each inner vector represents one component direction, ordered consistently with the scores. For PCA and LDA, these represent the eigenvectors. The container type depends on the model (ContainerType_Vector_Vector_REAL for PCA, ContainerType_Vector_Vector_LREAL for LDA).
pipMean	ITcVnContainer* [▶ 1827]*	nullptr	Container holding the mean vector used for centering. The container type depends on the model (ContainerType_Vector_REAL for PCA). For LDA, the mean is not applicable and the container will be set to 0. Optional, set to 0 if not required.

Return value

HRESULT [▶ 135]

Required License

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.2.3.21 测量

该组包含用于测量图像中对象的函数。

函数

边缘定位

- LocateCircularArc [▶ 2552]，用于定位比例圆弧
- LocateEdge [▶ 2554]，用于定位线性边缘
- LocateEdges [▶ 2556]，用于定位几个线性边缘
- LocateEllipse [▶ 2558]，用于定位完整椭圆

几何特征

- ClosestPointsBF [▶ 2551]，用于计算两个点云中最近的点
- MeasureAngleBetweenEdges [▶ 2560]，用于测量两条边之间的角度
- MeasureEdgeDistance [▶ 2562]，用于测量两条边之间的平均距离
- MeasureMinEdgeDistance [▶ 2564]，用于测量两条边之间的最小距离

其它

- AdjustSearchWindowOrientationToLinearEdge [▶ 2550]，用于将搜索栏与边缘对齐

边缘定位参数

图像对象的边界往往不完全位于两个像素之间的边界上。相反，它们与像素网格独立运行。因此，有必要使用亚像素精度进行边缘定位。这不仅考虑两个像素之间的转换，还根据对象边界的整体强度梯度来确定像素内的边缘位置。

注意

测量前不要处理图像

在调用测量函数之前，不要处理图像。处理过程可能会导致信息丢失，并相应降低准确性。

所有测量函数都使用边缘定位这一基本功能。这些函数的区别主要在于执行边缘定位的几何排列（例如：LocateEllipse [▶ 2558]，用于圆形物体）和从中计算出的属性（例如：MeasureAngleBetweenEdges [▶ 2560]，用于角度测量）。

必须为每个使用边缘定位的函数定义某些参数。其中包括：

- 搜索区域的定义：在哪寻找边缘?
- 边缘对比的定义：什么会被检测为边缘?
- 测量算法的定义：如何确定边缘的准确位置。

这些参数如下所述。对于专家参数，还指定了标准值。

搜索区域

参数	描述
搜索线的位置	搜索线的位置根据待定位特征定义。 线性边缘的搜索线位置定义为带有 `aStartPoint` 起点和 `aEndPoint` 终点的单线。 椭圆的搜索线位置由 `aCenter` 中心点和 `fSearchRadius` 半径定义。 角搜索线的位置由一个 `aInnerPoint` 内点以及两个 `aOuterPoint1` 和 `aOuterPoint2` 外点定义。

参数	描述
	搜索线的位置应始终设置为搜索线切割待正交定位的边缘。矩形搜索区域可通过 AdjustSearchWindowOrientationToLinearEdge [▶ 2550] 函数调整为与边缘正交。
搜索线数 nSearchLines	这指定要用于边缘定位的搜索线数量。最简单的边缘定位只有一条搜索线，必须在这条搜索线上找到边缘。根据需要确定的几何特征，在局部分布多条搜索线，以便对多个边缘转换进行定位。 在以下情况下可能有必要这样做： • 不仅需要位置，还需要边缘渐变或旋转。 • 几何特征（如椭圆）需要多条搜索线。 • 需要一个平均值。 在定位线性边缘时，搜索线对称排列在中心搜索线的两侧，中心搜索线包含在 nSearchLines 中。因此，总是需要奇数条搜索线。
搜索线之间的距离 fSearchlineDist （专家） 默认：1	搜索线距离以像素为单位定义相邻搜索线之间的距离（> 0，默认为 1）。这样就可以灵活平衡精度和计算时间。
反转搜索方向 bInvertSearchDirection （专家） 默认：false	定位算法始终从搜索线的一端到另一端搜索边缘，例如，从 aStartPoint 到 aEndPoint 进行搜索。如果由于图像属性的原因需要反转搜索方向（即从搜索线的另一端开始），该参数会有所帮助。

边缘对比

参数	描述
边缘方向 eEdgeDirection	ETcVnEdgeDirection 边缘方向决定应找到从亮到暗还是从暗到亮的边缘过渡。为此，有以下值可供选择： • ED_DARK_TO_LIGHT • ED_LIGHT_TO_DARK
最低强度 fMinStrength	最低强度定义边缘想要被找到所需的最小内部强度差。 这样就可以控制边缘为被找到所必须要具备的差异程度。
最大厚度 nMaxThickness （专家） 默认：10	最大厚度指定 fMinStrength 内部必须达到的像素数。 这样就可以控制是只寻找锋利边缘，还是也寻找非常模糊的边缘。

算法

参数	描述
算法 eAlgorithm （专家） 默认：EDA_INTERPOLATION	ETcVnEdgeDetectionAlgorithm 参数定义所使用的定位算法。提供以下选项： • EDA_INTERPOLATION • EDA_APPROX_ERF • EDA_APPROX_GAUSSIAN

参数	描述
	EDA_APPROX_ERF 和 EDA_APPROX_GAUSSIAN 比 EDA_INTERPOLATION 标准算法慢，但可以实现更高的精度。不过，这样的前提条件是边缘的强度梯度也尽可能符合指定模型。erf 误差函数或高斯函数被用作模型。 如果从亮到暗的过渡跨越多个像素，并且在亮区或暗区两侧分别保持近似恒定，则 erf 函数就非常适合边缘。  高斯函数非常适用于线，即暗-亮-暗的过渡，反之亦然。不过，亮度高原可能只有几个像素宽，否则应使用 erf 函数。 
子像素/迭代 nSubpixelsIterations （专家） 默认：10	该参数的含义取决于所使用的定位算法。 如为 EDA_INTERPOLATION： 所考虑的子像素数定义搜索线评估底层像素强度所使用的细粒度。例如，值 10 指定在 1 像素的长度上取 10 个测量值。 建议：5 — 10 如为 EDA_APPROX_ERF 和 EDA_APPROX_GAUSSIAN： 迭代数是一个终止标准，定义针对像素强度对边缘模型进行近似所使用的最大迭代数。更多的迭代会增加运行时；如果迭代次过多，则应使用插值法。 建议：50 — 100
近似精度 fApproxPrecision （专家） 默认：0.001	近似精度是边缘定位的另一个终止标准。它只有在定位算法为 EDA_APPROX_ERF 或 EDA_APPROX_GAUSSIAN 的情况下才能被观察到。 如果达到 nSubPixelIterations 的迭代数，或者连续两次迭代的像素相对偏差小于 fApproximationPrecision，边缘或线的拟合就会中止 建议：0.01 — 0.0001
边缘宽度的限制 fRangeCutoffFactor （专家） 默认：0.135	相对边缘强度的系数 -0.9 至 0.9 用于限制参与拟合的像素的强度范围。参数只有在定位算法为 TCVN_EDA_APPROX_ERF 或 TCVN_EDA_APPROX_GAUSSIAN 的情况下才能被观察到。 允许使用负数，以便考虑噪声，即另一个边缘方向上的强度差异。不过，负值可能会导致结果变差，尤其是当两条边缘彼此靠近时。 在两个方向上最多考虑 nMaxThickness 个像素。

HRESULT 的解释

HRESULT [▶ 135] 返回值对测量函数的意义如下：

代码	名称	描述
0x000	S_OK	函数成功执行，并在所有搜索线上定位了一条边。
0x001	S_FALSE	函数成功执行，但没有在每条搜索线上定位到边。
0x256	s_watchdogtimeout	函数被 Watchdog（看门狗）中止。在这种情况下，所有测量函数都可以返回部分结果。
0x70C	NOTFOUND	函数成功执行，但在任何搜索行上都未找到边。假设只有在确保图像包含具有参数设置边缘属性的对象时才会调用测量函数。如果没有找到任何搜索线，则表明参数设置不正确或函数调用不合适。因此，在这种情况下会输出错误代码。
0x7xx	除 NOTFOUND 外的所有错误代码	函数未成功执行。请参见：ADS 返回代码 [▶ 2857]

例如，可以使用下方查询来处理返回值：

```
if (SUCCEEDED(hr))
{
    // Process results
    if (hr != S_OK)
    {
        // Emit warning that not all search lines were found
    }
}
else if (hr == ADS_E_NOTFOUND)
{
    // Emit warning for wrong parametrization or missing piece
}
else
{
    // Error handling
}
```

缩短执行时间

固定搜索窗口的执行时间根据组件在图像中的位置而略有不同。如果找到的边离起点较近，执行时间就会比离起点较远的边短。

两种近似算法会出现较大的波动，因为这些算法需要不同的迭代，以便利用周围的像素强度近似模型参数。

一般来说，如果搜索窗口的搜索线尽可能正交地击中边缘，则需要更少的迭代。如果图像中物体的位置和方向波动很小或没有波动，可以相应地调整搜索窗口以实现更短的执行时间。此外，可以通过减少最大迭代次数来减少所需的最大时间，但这可能会导致结果不太准确。

此外，可以使用外部看门狗限制时间。如有必要，看门狗会终止函数的执行并返回现有的部分结果。

还请参阅有关此

- LocateAxisAlignedEdges [▶ 2566]

6.2.3.21.1 AdjustSearchWindowOrientationToLinearEdge

调整搜索窗口，使其与线性边缘正交。在调用依赖搜索窗口的边缘定位和距离测量函数之前可能会有所帮助。可使用 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT AdjustSearchWindowOrientationToLinearEdge(
    HRESULT            hrPrev,
    ITcVnImage*        ipSrcImage,
    TcVnPoint2_REAL&   aStartPoint,
    TcVnPoint2_REAL&   aEndPoint,
    ETcVnEdgeDirection eEdgeDirection,
    float              fMinStrength,
    ULONG              nMaxThickness,
    ULONG              nSearchLines
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
aStartPoint	TcVnPoint2_REAL [▶ 1640]&	搜索过程的起始位置（沿 aEndPoint 方向），该位置不变
aEndPoint	TcVnPoint2_REAL [▶ 1640]&	搜索过程的结束位置，由该函数调整
eEdgeDirection	ETcVnEdgeDirection [▶ 1673]	指定要搜索的边缘方向
fMinStrength	float	指定要搜索的边缘的最小强度（强度差）
nMaxThickness	ULONG	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
nSearchLines	ULONG	搜索窗口的宽度，即搜索行数（3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.21.2 ClosestPointsBF

用蛮力法找出两个相同类型的二维点集之间的最近距离。

语法

定义：

```
HRESULT ClosestPointsBF(
    HRESULT           hrPrev,
    ITcVnContainer*   ipPointSet1,
    ITcVnContainer*   ipPointSet2,
    double&           fMinDist,
    TcVnPoint2_LREAL& aPoint1,
    TcVnPoint2_LREAL& aPoint2
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipPointSet1	ITcVnContainer* [▶ 1827]	包含第一组二维点的容器

名称	类型	描述
ipPointSet2	ITcVnContainer* [▶ 1827]	包含第二组二维点的容器
fMinDist	double&	返回不同点集的两点之间的最小距离
aPoint1	TcVnPoint2_LREAL [▶ 1640]&	返回 ipPointSet1 中达到 fMinDist 的点
aPoint2	TcVnPoint2_LREAL [▶ 1640]&	返回 ipPointSet2 中达到 fMinDist 的点

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.21.3 LocateCircularArc

定位圆弧。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT LocateCircularArc(
    HRESULT                     hrPrev,
    ITcVnImage*                 ipSrcImage,
    TcVnCircularArc&            stCircularArc,
    TcVnPoint2_REAL&            aCenter,
    float                       fSearchRadius,
    double                      fArcDirectionRad,
    ETcVnEdgeDirection          eEdgeDirection,
    float                       fMinStrength,
    double                      fAngleStepRad = 0.1,
    ULONG                       nMaxThickness = 10,
    bool                        bInvertSearchDirection = false,
    ULONG                       nSubpixelsIterations = 10,
    float                       fApproxPrecision = 0.001f,
    ETcVnEdgeDetectionAlgorithm eAlgorithm = EDA_INTERPOLATION,
    ITcVnContainer**            pipContourPoints = nullptr,
    float                       fRangeCutoffFactor = 0.135f
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（1 通道）
stCircularArc	TcVnCircularArc [▶ 1703]&		返回圆弧参数

名称	类型	默认值	描述
aCenter	TcVnPoint2_REAL [▶ 1640]&		输入估算的圆心（仅用作搜索圆轮廓的起点，不用于圆心估算）
fSearchRadius	float		输入搜索半径（从 aCenter 开始，应大于实际圆半径，但 aCenter + fSearchRadius 应在图像边界范围内）
fArcDirectionRad	double		输入以弧度为单位的搜索起始方向。有效范围为 [-pi，+pi]，其中右侧为 0 rad，顶部为 -pi/2 rad，底部为 +pi/2 rad。圆弧至少应在 fArcDirectionRad +- 4 * fAngleStepRad 范围内有效
eEdgeDirection	ETcVnEdgeDirection [▶ 1673]		指定要搜索的边缘方向
fMinStrength	float		指定要搜索的边缘的最小强度（强度差）
fAngleStepRad	double	0.1	搜索步长，以弧度为单位（选择时应确保整个弧线有大约 20 - 60 个步长。在大多数情况下，0.1 弧度（5.7 度）是良好值）
nMaxThickness	ULONG	10	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
bInvertSearchDirection	bool	false	如果为 true，则搜索从圆弧外沿圆心方向开始
nSubpixelsIterations	ULONG	10	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	float	0.001f	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 1673]	EDA_INTERPOLATION	选择边缘检测算法
pipContourPoints	ITcVnContainer* [▶ 1827]*	nullptr	返回亚像素精确轮廓（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL；重复使用非零接口指针。）

名称	类型	默认值	描述
fRangeCutoffFactor	float	0.135f	用于剔除参与拟合的像素范围的相对边缘强度系数。负数表示包含噪声，即另一个方向上的强度差异。每个方向最多使用 nMaxThickness 个像素。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.21.4 LocateEdge

以亚像素精度定位指定搜索窗口内首次出现的边缘的点。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT LocateEdge(
    HRESULT                     hrPrev,
    ITcVnImage*                 ipSrcImage,
    ITcVnContainer*&            ipEdgePoints,
    TcVnPoint2_REAL&            aStartPoint,
    TcVnPoint2_REAL&            aEndPoint,
    ETcVnEdgeDirection          eEdgeDirection,
    float                       fMinStrength,
    ULONG                       nSearchLines,
    float&                      fAvgStrength,
    float                       fSearchLineDist = 1,
    ULONG                       nMaxThickness = 10,
    ULONG                       nSubpixelsIterations = 10,
    float                       fApproxPrecision = 0.001f,
    ETcVnEdgeDetectionAlgorithm eAlgorithm = EDA_INTERPOLATION,
    float                       fRangeCutoffFactor = 0.135f,
    ITcVnContainer**            pipEdgeStrengths = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（1 通道）
ipEdgePoints	ITcVnContainer* [▶ 1827]&		返回检测到的边缘点（ContainerType_Vector_TcVnPoint2_REAL）

名称	类型	默认值	描述
aStartPoint	TcVnPoint2_REAL [▶ 1640]&		搜索过程的起始位置（沿 aEndPoint 方向）
aEndPoint	TcVnPoint2_REAL [▶ 1640]&		搜索过程的结束位置
eEdgeDirection	ETcVnEdgeDirection [▶ 1673]		指定要搜索的边缘方向
fMinStrength	float		指定要搜索的边缘的最小强度（强度差）
nSearchLines	ULONG		搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
fAvgStrength	float&		返回检测到的边缘的平均强度
fSearchLineDist	float	1	搜索线之间的距离，以像素为单位（> 0）
nMaxThickness	ULONG	10	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
nSubpixelsIterations	ULONG	10	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	float	0.001f	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 1673]	EDA_INTERPOLATION	选择边缘检测算法
fRangeCutoffFactor	float	0.135f	用于剔除参与拟合的像素范围的相对边缘强度系数。负数表示包含噪声，即另一个方向上的强度差异。每个方向最多使用 nMaxThickness 个像素。
pipEdgeStrengths	ITcVnContainer* [▶ 1827]*	nullptr	返回检测到的边缘强度（可选，如果不需要，则设为 0；ContainerType_Vector_REAL）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.21.5 LocateEdges

以亚像素精度在指定搜索窗口内定位多次出现的边缘的点。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT LocateEdges(
    HRESULT                     hrPrev,
    ITcVnImage*                 ipSrcImage,
    ITcVnContainer*&            ipEdgePoints,
    TcVnPoint2_REAL&            aStartPoint,
    TcVnPoint2_REAL&            aEndPoint,
    ULONG                       nNumEdges,
    ETcVnEdgeDirection          eEdgeDirection,
    bool                        bAlternateDirection,
    float                       fMinStrength,
    ULONG                       nSearchLines,
    float&                      fAvgStrength,
    float                       fSearchLineDist = 1,
    ULONG                       nMaxThickness = 10,
    ULONG                       nSubpixelsIterations = 10,
    float                       fApproxPrecision = 0.001f,
    ETcVnEdgeDetectionAlgorithm eAlgorithm = EDA_INTERPOLATION,
    float                       fRangeCutoffFactor = 0.135f,
    ITcVnContainer**            pipEdgeStrengths = nullptr,
    bool                        bAvgEdgeStrengths = false
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（1 通道）
ipEdgePoints	ITcVnContainer* [▶ 1827]&		返回检测到的边缘点（ContainerType_Vector_Vector_TcVnPoint2_REAL）
aStartPoint	TcVnPoint2_REAL [▶ 1640]&		搜索过程的起始位置（沿 aEndPoint 方向）
aEndPoint	TcVnPoint2_REAL [▶ 1640]&		搜索过程的结束位置
nNumEdges	ULONG		要搜索的边缘的（最大）数量
eEdgeDirection	ETcVnEdgeDirection [▶ 1673]		指定要搜索的边缘方向

名称	类型	默认值	描述
bAlternateDirection	bool		如果为 true，eEdgeDirection 会在每次检测到边缘后交替出现。否则，仅搜索具有 eEdgeDirection 的边缘。
fMinStrength	float		指定要搜索的边缘的最小强度（强度差）
nSearchLines	ULONG		搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
fAvgStrength	float&		返回检测到的边缘的平均强度
fSearchLineDist	float	1	搜索线之间的距离，以像素为单位（> 0）
nMaxThickness	ULONG	10	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
nSubpixelsIterations	ULONG	10	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	float	0.001f	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 1673]	EDA_INTERPOLATION	选择边缘检测算法
fRangeCutoffFactor	float	0.135f	用于剔除参与拟合的像素范围的相对边缘强度系数。负数表示包含噪声，即另一个方向上的强度差异。每个方向最多使用 nMaxThickness 个像素。
pipEdgeStrengths	ITcVnContainer* [▶ 1827]*	nullptr	返回检测到的边缘强度（可选，如果不需要，则设为 0；ContainerType_Vector_Vector_REAL 或 ContainerType_Vector_REAL，取决于 bAvgEdgeStrengths）

名称	类型	默认值	描述
bAvgEdgeStrengths	bool	false	如果为 true，ipEdgeStrengths 将返回每条边缘的平均强度，而不是每条搜索线的单独强度

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.21.6 LocateEllipse

定位一个椭圆。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT LocateEllipse(
    HRESULT                     hrPrev,
    ITcVnImage*                 ipSrcImage,
    TcVnRotatedRectangle&       stEllipse,
    TcVnPoint2_REAL&            aCenter,
    float                       fSearchRadius,
    ETcVnEdgeDirection          eEdgeDirection,
    float                       fMinStrength,
    ULONG                       nMaxThickness = 10,
    bool                        bInvertSearchDirection = false,
    float                       fMinSearchRadius = 0,
    ULONG                       nSubpixelsIterations = 10,
    ULONG                       nSearchLines = 92,
    float                       fApproxPrecision = 0.001f,
    ETcVnEdgeDetectionAlgorithm eAlgorithm = EDA_INTERPOLATION,
    ITcVnContainer**            pipContourPoints = nullptr,
    float                       fRangeCutoffFactor = 0.135f
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（1 通道）
stEllipse	TcVnRotatedRectangle [▶ 1718]&		返回检测到的椭圆
aCenter	TcVnPoint2_REAL [▶ 1640]&		预期的椭圆中心

名称	类型	默认值	描述
fSearchRadius	float		围绕 aCenter 搜索边缘的半径
eEdgeDirection	ETcVnEdgeDirection [▶ 1673]		指定要搜索的边缘方向（从中心到椭圆外侧，或如果 bInvertSearchDirection 为 true，则从椭圆外侧到中心）
fMinStrength	float		指定要搜索的边缘的最小强度（强度差）
nMaxThickness	ULONG	10	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
bInvertSearchDirection	bool	false	如果为 true，则搜索从椭圆外沿中心方向开始
fMinSearchRadius	float	0	在开始搜索边缘之前要跳过的 aCenter 周围半径（例如，为了节省时间或中心包含应忽略的边缘）
nSubpixelsIterations	ULONG	10	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
nSearchLines	ULONG	92	指定搜索线的数量，在所有方向上平均分配（必须 >= 8 且是 4 的倍数）
fApproxPrecision	float	0.001f	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 1673]	EDA_INTERPOLATION	选择边缘检测算法
pipContourPoints	ITcVnContainer* [▶ 1827]*	nullptr	返回亚像素精确椭圆轮廓（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL；重复使用非零接口指针。）
fRangeCutoffFactor	float	0.135f	用于剔除参与拟合的像素范围的相对边缘强度系数。负数表示包含噪声，即另一个方向上的强度差异。每个方向最多使用 nMaxThickness 个像素。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.21.7 MeasureAngleBetweenEdges

测量 2 个边缘之间的角度。
可使用 TwinCAT Job Task 并行执行函数代码实现加速。
被看门狗取消时，可返回部分结果。

语法

定义：

```
HRESULT MeasureAngleBetweenEdges(
    HRESULT                       hrPrev,
    ITcVnImage*                   ipSrcImage,
    float&                        fAngle,
    TcVnPoint2_REAL&              aInnerPoint,
    TcVnPoint2_REAL&              aOuterPoint1,
    TcVnPoint2_REAL&              aOuterPoint2,
    ETcVnEdgeDirection            eEdgeDirection,
    float                         fMinStrength,
    ULONG                         nSearchLines,
    float                         fSearchLineDist = 1,
    ULONG                         nMaxThickness = 10,
    bool                          bInvertSearchDirection = false,
    ULONG                         nSubpixelsIterations = 10,
    bool                          bAngleInDegrees = false,
    float                         fApproxPrecision = 0.001f,
    ETcVnEdgeDetectionAlgorithm eAlgorithm = EDA_INTERPOLATION,
    ITcVnContainer**              pipEdgePoints1 = nullptr,
    ITcVnContainer**              pipEdgePoints2 = nullptr,
    float                         fRangeCutoffFactor = 0.135f
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（1 通道）
fAngle	float&		返回检测到的边缘之间的角度（弧度或度，取决于 bAngleInDegrees）
aInnerPoint	TcVnPoint2_REAL [▶ 1640]&		要测量的角内位置，从该位置开始边缘搜索过程（沿 aOuterPoint1 和 aOuterPoint2 方向）
aOuterPoint1	TcVnPoint2_REAL [▶ 1640]&		边缘 1 搜索过程的结束位置

名称	类型	默认值	描述
aOuterPoint2	TcVnPoint2_REAL [▶ 1640]&		边缘 2 搜索过程的结束位置
eEdgeDirection	ETcVnEdgeDirection [▶ 1673]		指定要搜索的从 aInnerPoint 到 aOuterPoint1 和 aOuterPoint2 的边缘方向
fMinStrength	float		指定要搜索的边缘的最小强度（强度差）
nSearchLines	ULONG		搜索窗口的宽度，即搜索行数（3、5、7.....），以 aInnerPoint 和 aOuterPoint1、aOuterPoint2 指定的行为中心
fSearchLineDist	float	1	搜索线之间的距离，以像素为单位（> 0）
nMaxThickness	ULONG	10	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
bInvertSearchDirection	bool	false	如果为 true，则从每个 aOuterPoint 开始沿 aInnerPoint 方向搜索
nSubpixelsIterations	ULONG	10	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
bAngleInDegrees	bool	false	如果为 true，fAngle 以度为单位
fApproxPrecision	float	0.001f	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 1673]	EDA_INTERPOLATION	选择边缘检测算法
pipEdgePoints1	ITcVnContainer* [▶ 1827]*	nullptr	返回在 aInnerPoint 和 aOuterPoint1 之间检测到的边缘点（ContainerType_Vector_TcVnPoint2_REAL）
pipEdgePoints2	ITcVnContainer* [▶ 1827]*	nullptr	返回在 aInnerPoint 和 aOuterPoint2 之间检测到的边缘点（ContainerType_Vector_TcVnPoint2_REAL）
fRangeCutoffFactor	float	0.135f	用于剔除参与拟合的像素范围的相对边缘强度系数。负数表示包含噪声，

名称	类型	默认值	描述
			即另一个方向上的强度差异。每个方向最多使用 nMaxThickness 个像素。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.21.8 MeasureEdgeDistance

测量两条平行边缘之间的距离。aStartPoint 和 aEndPoint 之间的中点必须位于两条边缘之间。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT MeasureEdgeDistance(
    HRESULT                     hrPrev,
    ITcVnImage*                 ipSrcImage,
    float&                      fAvgDistance,
    TcVnPoint2_REAL&            aStartPoint,
    TcVnPoint2_REAL&            aEndPoint,
    ETcVnEdgeDirection          eEdgeDirection,
    float                       fMinStrength,
    ULONG                       nSearchLines,
    float                       fSearchLineDist = 1,
    ULONG                       nMaxThickness = 10,
    bool                        bInvertSearchDirection = false,
    float                       fSearchGap = 0,
    ULONG                       nSubpixelsIterations = 10,
    float                       fApproxPrecision = 0.001f,
    ETcVnEdgeDetectionAlgorithm eAlgorithm = EDA_INTERPOLATION,
    ITcVnContainer**            pipEdgePoints1 = nullptr,
    ITcVnContainer**            pipEdgePoints2 = nullptr,
    ITcVnContainer**            pipDistances = nullptr,
    float                       fRangeCutoffFactor = 0.135f
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（1 通道）
fAvgDistance	float&		返回检测到的边缘之间的平均距离
aStartPoint	TcVnPoint2_REAL [▶ 1640]&		搜索过程的起始位置（沿 aEndPoint 方向）

名称	类型	默认值	描述
aEndPoint	TcVnPoint2_REAL [▶ 1640]&		搜索过程的结束位置
eEdgeDirection	ETcVnEdgeDirection [▶ 1673]		指定要搜索的从 aStartPoint 到 aEndPoint 的边缘方向
fMinStrength	float		指定要搜索的边缘的最小强度（强度差）
nSearchLines	ULONG		搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
fSearchLineDist	float	1	搜索线之间的距离，以像素为单位（> 0）
nMaxThickness	ULONG	10	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
bInvertSearchDirection	bool	false	如果为 true，则搜索从 aStartPoint 和 aEndPoint 之间的中心点沿两个方向开始
fSearchGap	float	0	可选的间隙宽度（>= 0，以 aStartPoint 和 aEndPoint 之间为中心），搜索边缘时忽略该间隙（可缩短执行时间）。要搜索的 2 个边缘必须位于间隙的不同侧。
nSubpixelsIterations	ULONG	10	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）
fApproxPrecision	float	0.001f	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 1673]	EDA_INTERPOLATION	选择边缘检测算法
pipEdgePoints1	ITcVnContainer* [▶ 1827]*	nullptr	返回检测到的靠近 aStartPoint 的边缘点（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL）
pipEdgePoints2	ITcVnContainer* [▶ 1827]*	nullptr	返回检测到的靠近 aEndPoint 的边缘点（可选，如果不需要，则设为

名称	类型	默认值	描述
			0; ContainerType_Vector_TcVnPoint2_REAL)
pipDistances	ITcVnContainer* [▶ 1827]*	nullptr	返回检测到的边缘点之间的距离（可选，如果不需要，则设为 0; ContainerType_Vector_REAL)
fRangeCutoffFactor	float	0.135f	用于剔除参与拟合的像素范围的相对边缘强度系数。负数表示包含噪声，即另一个方向上的强度差异。每个方向最多使用 nMaxThickness 个像素。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.21.9 MeasureMinEdgeDistance

在指定搜索窗口内测量两条边缘之间的最小距离。aStartPoint 和 aEndPoint 之间的中点必须位于两条边缘之间。
可以使用可用的 TwinCAT Job Tasks 来执行并行编码区域。
被 Watchdog（看门狗）取消时，可返回部分结果。

语法

定义：

```
HRESULT MeasureMinEdgeDistance(
    HRESULT                       hrPrev,
    ITcVnImage*                   ipSrcImage,
    float&                        fMinDistance,
    TcVnPoint2_REAL&              aStartPoint,
    TcVnPoint2_REAL&              aEndPoint,
    ETcVnEdgeDirection            eEdgeDirection,
    float                         fMinStrength,
    ULONG                         nSearchLines,
    TcVnPoint2_REAL&              aPoint1,
    TcVnPoint2_REAL&              aPoint2,
    float                         fSearchLineDist = 1,
    ULONG                         nMaxThickness = 10,
    bool                          bInvertSearchDirection = false,
    float                         fSearchGap = 0,
    ULONG                         nSubpixelsIterations = 10,
    float                         fApproxPrecision = 0.001f,
    ETcVnEdgeDetectionAlgorithm eAlgorithm = EDA_INTERPOLATION,
    ITcVnContainer**              pipEdgePoints1 = nullptr,
    ITcVnContainer**              pipEdgePoints2 = nullptr,
    float                         fRangeCutoffFactor = 0.135f
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（1 通道）
fMinDistance	float&		返回检测到的边缘之间的最小距离
aStartPoint	TcVnPoint2_REAL [▶ 1640]&		搜索过程的起始位置（沿 aEndPoint 方向）
aEndPoint	TcVnPoint2_REAL [▶ 1640]&		搜索过程的结束位置
eEdgeDirection	ETcVnEdgeDirection [▶ 1673]		指定要搜索的从 aStartPoint 到 aEndPoint 的边缘方向
fMinStrength	float		指定要搜索的边缘的最小强度（强度差）
nSearchLines	ULONG		搜索窗口的宽度，即搜索行数（1、3、5、7.....），以 aStartPoint 和 aEndPoint 指定的行为中心
aPoint1	TcVnPoint2_REAL [▶ 1640]&		返回靠近 aStartPoint 的边缘上达到最小距离的点
aPoint2	TcVnPoint2_REAL [▶ 1640]&		返回靠近 aEndPoint 的边缘上达到最小距离的点
fSearchLineDist	float	1	搜索线之间的距离，以像素为单位（> 0）
nMaxThickness	ULONG	10	指定要搜索的边缘的最大厚度，即必须在 nMaxThickness 像素范围内达到 fMinStrength
bInvertSearchDirection	bool	false	如果为 true，则搜索从 aStartPoint 和 aEndPoint 之间的中心点沿两个方向开始
fSearchGap	float	0	可选的间隙宽度（>= 0，以 aStartPoint 和 aEndPoint 之间为中心），搜索边缘时忽略该间隙（可缩短执行时间）。要搜索的 2 个边缘必须位于间隙的不同侧。
nSubpixelsIterations	ULONG	10	指定亚像素数量（对于 INTERPOLATION，通常 10 - 20 个即可）或用于优化参数的最大迭代次数（对于 APPROX_ERF 和 APPROX_GAUSSIAN，通常 50 - 100 次即可）

名称	类型	默认值	描述
fApproxPrecision	float	0.001f	指定 APPROX_ERF 和 APPROX_GAUSSIAN 的近似精度（通常 0.001 即可，对于 INTERPOLATION 不使用）
eAlgorithm	ETcVnEdgeDetectionAlgorithm [▶ 1673]	EDA_INTERPOLATION	选择边缘检测算法
pipEdgePoints1	ITcVnContainer* [▶ 1827]*	nullptr	返回检测到的靠近 aStartPoint 的边缘点（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL）
pipEdgePoints2	ITcVnContainer* [▶ 1827]*	nullptr	返回检测到的靠近 aEndPoint 的边缘点（可选，如果不需要，则设为 0；ContainerType_Vector_TcVnPoint2_REAL）
fRangeCutoffFactor	float	0.135f	用于剔除参与拟合的像素范围的相对边缘强度系数。负数表示包含噪声，即另一个方向上的强度差异。每个方向最多使用 nMaxThickness 个像素。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.21.10 LocateAxisAlignedEdges

Locate the position of multiple occurring edges inside a specified search window. The edges inside the search window must be straight and aligned to one of the image axes. They are located by interpolating the peaks of the first Gaussian derivative of the pixel intensities.

Syntax

Definition:

```
HRESULT LocateAxisAlignedEdges(
    HRESULT             hrPrev,
    ITcVnImage*         ipSrcImage,
    ITcVnContainer*&    ipEdgePoints,
    TcVnPoint2_DINT&    aStartPoint,
    TcVnPoint2_DINT&    aEndPoint,
    ULONG               nSearchWindowHeight,
```

```
    ULONG               nKernelSize,
    ULONG               nNumEdges,
    ETcVnEdgeSelection  eEdgeSelection,
    float               fMinStrength,
    float&              fAvgStrength,
    float               fSigma = 0,
    ITcVnContainer**    pipEdgeStrengths = nullptr,
    ITcVnImage**        pipDerivative = nullptr
)
```

Parameters

Name	Type	Default	Description
hrPrev	HRESULT [▶ 135]		HRESULT indicating the result of previous operations (If SUCCEEDED(hrPrev) equals false, no operation is executed.)
ipSrcImage	ITcVnImage* [▶ 1863]		Source image (1 channel of type ET_USINT, ET_UINT, ET_INT or ET_REAL)
ipEdgePoints	ITcVnContainer* [▶ 1827]&		Returns the detected edge points (ContainerType_Vector_TcVnPoint2_REAL)
aStartPoint	TcVnPoint2_DINT [▶ 1640]&		Position from which to start the search process (in the direction of aEndPoint)
aEndPoint	TcVnPoint2_DINT [▶ 1640]&		Position where the search process ends. As the search direction must be parallel to one of the images axes, aEndPoint must have the same x- or y-coordinate as aStartPoint.
nSearchWindowHeight	ULONG		Specifies the height of the search window, i.e. the number of rows for horizontal or columns for vertical search directions, centered around the line between aStartPoint and aEndPoint. Depending on the orientation, the rows or columns inside that window are averaged and the edges are located on that averaged row or column. Therefore it is important that the edges inside the defined search window are straight and perpendicular to the search direction.

Name	Type	Default	Description
nKernelSize	ULONG		The size of the Gaussian derivative kernel, that is used to locate the edges. The size must be odd and >= 3 (5 or 7 are usually a good choice for sharp edges).
nNumEdges	ULONG		The (maximum) number of edges to search for
eEdgeSelection	ETcVnEdgeSelection [▶ 1642]		Selects the edge directions to search for
fMinStrength	float		Specification of the minimum strength (absolute peak height of the Gaussian derivative of the pixel intensities) of the edge to search for. This value depends on the edge sharpness and nKernelSize, but is generally smaller or equal to the difference of the pixel intensities of the edge. Sharp edges have a higher peak than unsharp ones. Initially, you might choose a relatively small value (e.g. 30) and then increase it depending on the fAvgStrength and ipEdgeStrengths output.
fAvgStrength	float&		Returns the average strength of the detected edges
fSigma	float	0	The sigma parameter for the Gaussian derivative function. If set to 0, a suitable value is automatically chosen depending on nKernelSize. For manually defined sigma, the value should always be chosen depending on nKernelSize (e.g. fSigma = (nKernelSize - 1) / factor, with factor in a range of 4 to 8).
pipEdgeStrengths	ITcVnContainer* [▶ 1827]*	nullptr	Returns the individual edge strengths, where dark to light edge strengths are positive and light to dark edge strengths are negative (optional, set to 0 if not

Name	Type	Default	Description
			required; ContainerType_Vector_REAL).
pipDerivative	ITcVnImage* [▶ 1863]*	nullptr	Returns the Gaussian derivative of the pixel intensities on the search line between aStartPoint and aEndPoint as a single image row, where aStartPoint would be on the left and aEndPoint on the right (optional, set to 0 if not required; 1 channel, ET_REAL).

Return value

HRESULT [▶ 135]

Required License

TC3 Vision Metrology 2D

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4026 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel 四核 Atom CPU	Tc3_Vision

6.2.3.22 其它

该组包含可跨组使用的其他函数。

函数

几何辅助函数

- ClipLineToBoundary [▶ 2571]
- ClipLineToBoundary_ITcVnImage [▶ 2572]
- ClipLineToBoundary_TcVnRectangle_DINT [▶ 2572]
- LineIntersectionPoint [▶ 2579]
- LineIntersectionPointAndAngle [▶ 2580]
- RotatedRectangleCorners [▶ 2581]
- RotatedRectangleIntersection [▶ 2582]
- OrthogonalVector_TcVnVector2_DINT [▶ 2584]
- OrthogonalVector_TcVnVector2_LREAL [▶ 2585]
- OrthogonalVector_TcVnVector2_REAL [▶ 2585]
- PointToLineDistance_TcVnPoint2_DINT [▶ 2586]
- PointToLineDistance_TcVnPoint2_LREAL [▶ 2587]
- PointToLineDistance_TcVnPoint2_REAL [▶ 2587]
- PointToPointDirection_TcVnPoint2_DINT [▶ 2588]
- PointToPointDirection_TcVnPoint2_LREAL [▶ 2589]

- PointToPointDirection_TcVnPoint2_REAL [▶ 2589]
- PointToPointDistance_TcVnPoint2_DINT [▶ 2590]
- PointToPointDistance_TcVnPoint2_LREAL [▶ 2591]
- PointToPointDistance_TcVnPoint2_REAL [▶ 2591]

时间戳

- GetTimestamp [▶ 2576]
- UpdateTimestamp [▶ 2583]

矩阵辅助函数

- InitMatrixStruct [▶ 2578]
- InvertMatrix3x3 [▶ 2578]
- MultiplyMatrices [▶ 2581]

转换接口

- ConvertITcUnknownToITcVnBitmapExport [▶ 2573]
- ConvertITcUnknownToITcVnContainer [▶ 2574]
- ConvertITcUnknownToITcVnImage [▶ 2574]
- ConvertITcUnknownToITcVnMlModel [▶ 2575]

初始化函数

- CheckFunctionInitialization [▶ 2570]
- DeinitializeFunction [▶ 2576]

其他

- HuMomentInvariants [▶ 2577]
- SetRngSeed [▶ 2583]

6.2.3.22.1 CheckFunctionInitialization

检查函数是否已使用指定选项初始化。

语法

定义：

```
HRESULT CheckFunctionInitialization(
    HRESULT                   hrPrev,
    ETcVnInitializableFunction eFunction,
    ULONGLONG                 nOptions
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
eFunction	ETcVnInitializableFunction [▶ 1679]	可初始化函数
nOptions	ULONGLONG	函数的初始化选项

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.2 ClipLineToBoundary

将一条直线剪切到矩形边界上。

语法

定义：

```
HRESULT ClipLineToBoundary(
    HRESULT             hrPrev,
    TcVnVector4_LREAL&  aLine,
    LONG                nX1,
    LONG                nY1,
    LONG                nX2,
    LONG                nY2,
    TcVnPoint2_REAL&    aStartPoint,
    TcVnPoint2_REAL&    aEndPoint
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aLine	TcVnVector4_LREAL [▶ 1641]&	第一个和第二个元素描述了方向向量的 x 和 y 分量。第三个和第四个元素描述位置向量的 x 和 y 分量。
nX1	LONG	左上角矩形点的 x
nY1	LONG	左上角矩形点的 y
nX2	LONG	右下角矩形点的 x
nY2	LONG	右下角矩形点的 Y
aStartPoint	TcVnPoint2_REAL [▶ 1640]&	返回直线的剪切起点。
aEndPoint	TcVnPoint2_REAL [▶ 1640]&	返回直线的剪切终点。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.3 ClipLineToBoundary (ITcVnImage*)

将一条直线剪切到图像边界上。

语法

定义：

```
HRESULT ClipLineToBoundary(
    HRESULT             hrPrev,
    TcVnVector4_LREAL&  aLine,
    ITcVnImage*         ipImage,
    TcVnPoint2_REAL&    aStartPoint,
    TcVnPoint2_REAL&    aEndPoint
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aLine	TcVnVector4_LREAL [▶ 1641]&	第一个和第二个元素描述了方向向量的 x 和 y 分量。第三个和第四个元素描述位置向量的 x 和 y 分量。
ipImage	ITcVnImage* [▶ 1863]	从中提取边界的图像
aStartPoint	TcVnPoint2_REAL [▶ 1640]&	返回直线的剪切起点。
aEndPoint	TcVnPoint2_REAL [▶ 1640]&	返回直线的剪切终点。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.4 ClipLineToBoundary (TcVnRectangle_DINT)

将一条直线剪切到矩形边界上。

语法

定义：

```
HRESULT ClipLineToBoundary(
    HRESULT             hrPrev,
    TcVnVector4_LREAL&  aLine,
    TcVnRectangle_DINT& stRectangle,
    TcVnPoint2_REAL&    aStartPoint,
    TcVnPoint2_REAL&    aEndPoint
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aLine	TcVnVector4_LREAL [▶ 1641]&	第一个和第二个元素描述了方向向量的 x 和 y 分量。第三个和第四个元素描述位置向量的 x 和 y 分量。
stRectangle	TcVnRectangle_DINT [▶ 1717]&	矩形边界
aStartPoint	TcVnPoint2_REAL [▶ 1640]&	返回直线的剪切起点。
aEndPoint	TcVnPoint2_REAL [▶ 1640]&	返回直线的剪切终点。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.5 ConvertITcUnknownToITcVnBitmapExport

将 ITcUnknown 接口指针转换为 ITcVnBitmapExport 接口指针。目标指针将被释放（如果存在），并设置为源 指针。源指针将被设置为 0。

语法

定义：

```
HRESULT ConvertITcUnknownToITcVnBitmapExport(
    HRESULT             hrPrev,
    ITcUnknown*&        ipSrc,
    ITcVnBitmapExport*& ipDest
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrc	ITcUnknown* [▶ 1877]&	源指针
ipDest	ITcVnBitmapExport* [▶ 1846]&	目标指针

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.6 ConvertITcUnknownToITcVnContainer

将 ITcUnknown 接口指针转换为 ITcVnContainer 接口指针。目标指针将被释放（如果存在），并设置为源指针。源指针将被设置为 0。

语法

定义：

```
HRESULT ConvertITcUnknownToITcVnContainer(
    HRESULT          hrPrev,
    ITcUnknown*&     ipSrc,
    ITcVnContainer*& ipDest
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrc	ITcUnknown* [▶ 1877]&	源指针
ipDest	ITcVnContainer* [▶ 1827]&	目标指针

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.7 ConvertITcUnknownToITcVnImage

将 ITcUnknown 接口指针转换为 ITcVnImage 接口指针。目标指针将被释放（如果存在），并设置为源 指针。源指针将被设置为 0。

语法

定义：

```
HRESULT ConvertITcUnknownToITcVnImage(
    HRESULT      hrPrev,
    ITcUnknown*& ipSrc,
    ITcVnImage*& ipDest
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrc	ITcUnknown* [▶ 1877]&	源指针
ipDest	ITcVnImage* [▶ 1863]&	目标指针

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.8 ConvertITcUnknownToITcVnMlModel

将 ITcUnknown 接口指针转换为 ITcVnMlModel 接口指针。目标指针将被释放（如果存在），并设置为源 指针。源指针将被设置为 0。

语法

定义：

```
HRESULT ConvertITcUnknownToITcVnMlModel(
    HRESULT         hrPrev,
    ITcUnknown*&    ipSrc,
    ITcVnMlModel*& ipDest
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrc	ITcUnknown* [▶ 1877]&	源指针
ipDest	ITcVnMlModel* [▶ 1871]&	目标指针

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.9 DeinitializeFunction

取消函数指定选项的初始化。

语法

定义：

```
HRESULT DeinitializeFunction(
    HRESULT                    hrPrev,
    ETcVnInitializableFunction eFunction,
    ULONGLONG                  nOptions
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
eFunction	ETcVnInitializableFunction [▶ 1679]	可初始化函数
nOptions	ULONGLONG	函数的初始化选项

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.10 GetTimestamp

获取对象（如 ITcVnImage 或 ITcVnContainer）上次更改的时间戳（实际 DC 时间）

语法

定义：

```
HRESULT GetTimestamp(
    HRESULT     hrPrev,
    ITcUnknown* ipUnknown,
    LONGLONG&   nTimestamp
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipUnknown	ITcUnknown* [▶ 1877]	实现 ITcUnknown 和 ITcVnTimestamp 的对象（例如 ITcVnImage 或 ITcVnContainer）
nTimestamp	LONGLONG&	返回时间戳

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.11 HuMomentInvariants

计算 Hu 不变矩。

语法

定义：

```
HRESULT HuMomentInvariants(
    HRESULT           hrPrev,
    TcVnMoments&      stMoments,
    TcVnArray7_LREAL& aHuInvariants
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stMoments	TcVnMoments [▶ 1707]&	基于此矩计算 Hu 不变矩
aHuInvariants	TcVnArray7_LREAL [▶ 1641]&	返回 Hu 不变矩

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.12 InitMatrixStruct

初始化一个 TcVnMatrix 类型的结构体，扩展一个包含元信息的缓冲区，以便将其用作矩阵。

语法

定义：

```
HRESULT InitMatrixStruct(
    HRESULT          hrPrev,
    PVOID            pSrcBuffer,
    TcVnMatrix&      stDestMatrix,
    ULONG            nRows,
    ULONG            nCols,
    ETcVnElementType eElementType
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
pSrcBuffer	PVOID	源缓冲区
stDestMatrix	TcVnMatrix [▶ 1706]&	返回完成的矩阵结构体
nRows	ULONG	矩阵行
nCols	ULONG	矩阵列
eElementType	ETcVnElementType [▶ 1674]	矩阵元素类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.13 InvertMatrix3x3

反转 3x3 矩阵。

语法

定义：

```
HRESULT InvertMatrix3x3(
    HRESULT               hrPrev,
    TcVnMatrix3x3_LREAL&  aSrcMatrix,
    TcVnMatrix3x3_LREAL&  aInvertedMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aSrcMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	源矩阵
aInvertedMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	返回反转的矩阵

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.14 LineIntersectionPoint

计算两条直线的交点（如果所提供的直线平行，则返回 S_FALSE）。

语法

定义：

```
HRESULT LineIntersectionPoint(
    HRESULT            hrPrev,
    TcVnVector4_LREAL& aLine1,
    TcVnVector4_LREAL& aLine2,
    TcVnPoint2_REAL&   aIntersecPoint
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aLine1	TcVnVector4_LREAL [▶ 1641]&	第一行。第一个和第二个元素描述的是该直线共线向量的 x 和 y 分量。第三个和第四个元素描述该直线上一个点的 x 和 y 分量。
aLine2	TcVnVector4_LREAL [▶ 1641]&	第二行。第一个和第二个元素描述的是该直线共线向量的 x 和 y 分量。第三个和第四个元素描述该直线上一个点的 x 和 y 分量。
aIntersecPoint	TcVnPoint2_REAL [▶ 1640]&	返回 aLine1 和 aLine2 的交点。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.15 LineIntersectionPointAndAngle

计算两条直线的交点和夹角（如果所提供的直线平行，则返回 S_FALSE）。

语法

定义：

```
HRESULT LineIntersectionPointAndAngle(
    HRESULT            hrPrev,
    TcVnVector4_LREAL& aLine1,
    TcVnVector4_LREAL& aLine2,
    TcVnPoint2_REAL&   aIntersecPoint,
    float&             fAngle,
    bool               bAngleInDegrees
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aLine1	TcVnVector4_LREAL [▶ 1641]&	第一行。第一个和第二个元素描述的是该直线共线向量的 x 和 y 分量。第三个和第四个元素描述该直线上一个点的 x 和 y 分量。
aLine2	TcVnVector4_LREAL [▶ 1641]&	第二行。第一个和第二个元素描述的是该直线共线向量的 x 和 y 分量。第三个和第四个元素描述该直线上一个点的 x 和 y 分量。
aIntersecPoint	TcVnPoint2_REAL [▶ 1640]&	返回 aLine1 和 aLine2 的交点。
fAngle	float&	返回 aLine1 和 aLine2 的夹角。
bAngleInDegrees	BOOL	如果为 TRUE，fAngle 以度为单位。如果为 FALSE，则 fAngle 以弧度为单位。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.16 MultiplyMatrices

两个矩阵的矩阵乘法（A * B = C）。所有矩阵均由结构体表示，这些结构体中包含指向适当大小数组的指针。

语法

定义：

```
HRESULT MultiplyMatrices(
    HRESULT     hrPrev,
    TcVnMatrix& stSrcMatrix1,
    TcVnMatrix& stSrcMatrix2,
    TcVnMatrix& stDestMatrix
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stSrcMatrix1	TcVnMatrix [▶ 1706]&	第一源矩阵（A）
stSrcMatrix2	TcVnMatrix [▶ 1706]&	第二源矩阵（B）
stDestMatrix	TcVnMatrix [▶ 1706]&	目标矩阵（C）（目标矩阵由该函数填充，但需要提供所需的内存。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.17 RotatedRectangleCorners

计算旋转矩形的 4 个角点。

语法

定义：

```
HRESULT RotatedRectangleCorners(
    HRESULT                hrPrev,
    TcVnRotatedRectangle&  stRect,
    TcVnArray4_Point2_REAL& aCorners
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stRect	TcVnRotatedRectangle [▶ 1718]&	旋转矩形

名称	类型	描述
aCorners	TcVnArray4_Point2_REAL [▶ 1641]&	返回 4 个角点

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.18 RotatedRectangleIntersection

查找 2 个旋转矩形的交点。

语法

定义：

```
HRESULT RotatedRectangleIntersection(
    HRESULT                     hrPrev,
    TcVnRotatedRectangle&       stRect1,
    TcVnRotatedRectangle&       stRect2,
    ITcVnContainer*&            ipIntersection,
    ETcVnRectangleIntersection& eIntersection
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
stRect1	TcVnRotatedRectangle [▶ 1718]&	旋转矩形 1
stRect2	TcVnRotatedRectangle [▶ 1718]&	旋转矩形 2
ipIntersection	ITcVnContainer* [▶ 1827]&	返回交点（ContainerType_Vector_TcVnPoint2_REAL，重复使用非零接口指针）
eIntersection	ETcVnRectangleIntersection [▶ 1688]&	返回交点类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.19 SetRngSeed

设置内部伪随机数生成器种子（仅供测试）。

语法

定义：

```
HRESULT SetRngSeed(
    HRESULT hrPrev,
    LONG    nSeed
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
nSeed	LONG	种子（0 会使发生器回到初始状态）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.20 UpdateTimestamp

将对象的时间戳更新为实际 DC 时间（如 ITcVnImage 或 ITcVnContainer）

语法

定义：

```
HRESULT UpdateTimestamp(
    HRESULT     hrPrev,
    ITcUnknown* ipUnknown
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	描述
ipUnknown	ITcUnknown* [▶ 1877]	实现 ITcUnknown 和 ITcVnTimestamp 的对象（例如 ITcVnImage 或 ITcVnContainer）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.21 OrthogonalVector (TcVnVector2_DINT)

计算给定源向量的正交向量。

语法

定义：

```
HRESULT OrthogonalVector(
    HRESULT             hrPrev,
    TcVnVector2_DINT&   aSrc,
    TcVnVector2_LREAL&  aOrthogonal,
    bool                bNormalize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aSrc	TcVnVector2_DINT [▶ 1641]&	源向量
aOrthogonal	TcVnVector2_LREAL [▶ 1641]&	返回与 aSrc 正交的向量
bNormalize	BOOL	如果为 true，则 aOrthogonal 长度被归一化为 1，否则 aOrthogonal 的长度与 aSrc 相同

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.22 OrthogonalVector (TcVnVector2_LREAL)

计算给定源向量的正交向量。

语法

定义：

```
HRESULT OrthogonalVector(
    HRESULT             hrPrev,
    TcVnVector2_LREAL& aSrc,
    TcVnVector2_LREAL& aOrthogonal,
    bool                bNormalize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aSrc	TcVnVector2_LREAL [▶ 1641]&	源向量
aOrthogonal	TcVnVector2_LREAL [▶ 1641]&	返回与 aSrc 正交的向量
bNormalize	BOOL	如果为 true，则 aOrthogonal 长度被归一化为 1，否则 aOrthogonal 的长度与 aSrc 相同

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.23 OrthogonalVector (TcVnVector2_REAL)

计算给定源向量的正交向量。

语法

定义：

```
HRESULT OrthogonalVector(
    HRESULT            hrPrev,
    TcVnVector2_REAL& aSrc,
    TcVnVector2_REAL& aOrthogonal,
    bool               bNormalize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aSrc	TcVnVector2_REAL [▶ 1641]&	源向量

名称	类型	描述
aOrthogonal	TcVnVector2_REAL [▶ 1641]&	返回与 aSrc 正交的向量
bNormalize	BOOL	如果为 true，则 aOrthogonal 长度被归一化为 1，否则 aOrthogonal 的长度与 aSrc 相同

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.24 PointToLineDistance (TcVnPoint2_DINT)

计算点与直线之间的最短距离。

语法

定义：

```
HRESULT PointToLineDistance(
    HRESULT          hrPrev,
    TcVnPoint2_DINT& aPoint,
    TcVnPoint2_DINT& aLinePoint1,
    TcVnPoint2_DINT& aLinePoint2,
    double&          fDistance
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aPoint	TcVnPoint2_DINT [▶ 1640]&	点
aLinePoint1	TcVnPoint2_DINT [▶ 1640]&	线点 1
aLinePoint2	TcVnPoint2_DINT [▶ 1640]&	线点 2
fDistance	double&	返回点与直线之间的距离

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.25 PointToLineDistance (TcVnPoint2_LREAL)

计算点与直线之间的最短距离。

语法

定义：

```
HRESULT PointToLineDistance(
    HRESULT           hrPrev,
    TcVnPoint2_LREAL& aPoint,
    TcVnPoint2_LREAL& aLinePoint1,
    TcVnPoint2_LREAL& aLinePoint2,
    double&           fDistance
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aPoint	TcVnPoint2_LREAL [▶ 1640]&	点
aLinePoint1	TcVnPoint2_LREAL [▶ 1640]&	线点 1
aLinePoint2	TcVnPoint2_LREAL [▶ 1640]&	线点 2
fDistance	double&	返回点与直线之间的距离

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.26 PointToLineDistance (TcVnPoint2_REAL)

计算点与直线之间的最短距离。

语法

定义：

```
HRESULT PointToLineDistance(
    HRESULT          hrPrev,
    TcVnPoint2_REAL& aPoint,
    TcVnPoint2_REAL& aLinePoint1,
    TcVnPoint2_REAL& aLinePoint2,
    double&          fDistance
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）

名称	类型	描述
aPoint	TcVnPoint2_REAL [▶ 1640]&	点
aLinePoint1	TcVnPoint2_REAL [▶ 1640]&	线点 1
aLinePoint2	TcVnPoint2_REAL [▶ 1640]&	线点 2
fDistance	double&	返回点与直线之间的距离

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.27 PointToPointDirection (TcVnPoint2_DINT)

计算 2 个点之间的方向向量。

语法

定义：

```
HRESULT PointToPointDirection(
    HRESULT            hrPrev,
    TcVnPoint2_DINT&   aPoint1,
    TcVnPoint2_DINT&   aPoint2,
    TcVnVector2_LREAL& aDirection,
    bool               bNormalize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aPoint1	TcVnPoint2_DINT [▶ 1640]&	点 1
aPoint2	TcVnPoint2_DINT [▶ 1640]&	点 2
aDirection	TcVnVector2_LREAL [▶ 1641]&	返回从 aPoint1 到 aPoint2 的方向向量
bNormalize	BOOL	如果为 true，则 aDirection 长度被归一化为 1，否则 aDirection 长度为 aPoint1 与 aPoint2 之间的距离

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.28 PointToPointDirection (TcVnPoint2_LREAL)

计算 2 个点之间的方向向量。

语法

定义：

```
HRESULT PointToPointDirection(
    HRESULT            hrPrev,
    TcVnPoint2_LREAL&  aPoint1,
    TcVnPoint2_LREAL&  aPoint2,
    TcVnVector2_LREAL& aDirection,
    bool               bNormalize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aPoint1	TcVnPoint2_LREAL [▶ 1640]&	点 1
aPoint2	TcVnPoint2_LREAL [▶ 1640]&	点 2
aDirection	TcVnVector2_LREAL [▶ 1641]&	返回从 aPoint1 到 aPoint2 的方向向量
bNormalize	BOOL	如果为 true，则 aDirection 长度被归一化为 1，否则 aDirection 长度为 aPoint1 与 aPoint2 之间的距离

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.29 PointToPointDirection (TcVnPoint2_REAL)

计算 2 个点之间的方向向量。

语法

定义：

```
HRESULT PointToPointDirection(
    HRESULT            hrPrev,
    TcVnPoint2_REAL&   aPoint1,
    TcVnPoint2_REAL&   aPoint2,
```

```
    TcVnVector2_REAL& aDirection,
    bool              bNormalize
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aPoint1	TcVnPoint2_REAL [▶ 1640]&	点 1
aPoint2	TcVnPoint2_REAL [▶ 1640]&	点 2
aDirection	TcVnVector2_REAL [▶ 1641]&	返回从 aPoint1 到 aPoint2 的方向向量
bNormalize	BOOL	如果为 true，则 aDirection 长度被归一化为 1，否则 aDirection 长度为 aPoint1 与 aPoint2 之间的距离

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.30 PointToPointDistance (TcVnPoint2_DINT)

计算 2 个点之间的距离。

语法

定义：

```
HRESULT PointToPointDistance(
    HRESULT          hrPrev,
    TcVnPoint2_DINT& aPoint1,
    TcVnPoint2_DINT& aPoint2,
    double&          fDistance
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aPoint1	TcVnPoint2_DINT [▶ 1640]&	点 1
aPoint2	TcVnPoint2_DINT [▶ 1640]&	点 2
fDistance	double&	返回各点之间的距离

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.31 PointToPointDistance (TcVnPoint2_LREAL)

计算 2 个点之间的距离。

语法

定义：

```
HRESULT PointToPointDistance(
    HRESULT           hrPrev,
    TcVnPoint2_LREAL& aPoint1,
    TcVnPoint2_LREAL& aPoint2,
    double&           fDistance
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aPoint1	TcVnPoint2_LREAL [▶ 1640]&	点 1
aPoint2	TcVnPoint2_LREAL [▶ 1640]&	点 2
fDistance	double&	返回各点之间的距离

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.22.32 PointToPointDistance (TcVnPoint2_REAL)

计算 2 个点之间的距离。

语法

定义：

```
HRESULT PointToPointDistance(
    HRESULT          hrPrev,
    TcVnPoint2_REAL& aPoint1,
    TcVnPoint2_REAL& aPoint2,
    double&          fDistance
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
aPoint1	TcVnPoint2_REAL [▶ 1640]&	点 1
aPoint2	TcVnPoint2_REAL [▶ 1640]&	点 2
fDistance	double&	返回各点之间的距离

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23 神经网络

该组包含用于预处理和后处理以及执行神经网络模型的函数。

函数

预处理和后处理

- ConvertDataLayout [▶ 2598]
- ReshapeImage [▶ 2605]

执行神经网络

- ExecuteNeuralNetwork [▶ 2599]
  具有一个输入和一个输出。
- ExecuteNeuralNetwork_MI [▶ 2593]
  具有多个输入 (MI) 和一个输出。
- ExecuteNeuralNetwork_MIMO_String [▶ 2594]
  具有多个输入和输出 (MIMO)。
- ExecuteNeuralNetwork_MISO [▶ 2594]
  具有多个输入和单个指定输出 (MISO)。
- ExecuteNeuralNetwork_MO [▶ 2595]
  一个输入和多个输出 (MO)。
- ExecuteNeuralNetwork_SO [▶ 2596]
  具有一个输入和单个指定输出 (SO)。

后处理

- Sigmoid [▶ 2606]
- SoftMax [▶ 2607]
- NonMaxSuppression [▶ 2603]
- Reduce [▶ 2604]

- ReduceArg [▶ 2605]

获取图层名称

- GetNeuralNetworkLayerNames [▶ 2602]

6.2.3.23.1 ExecuteNeuralNetwork (MI)

使用提供的图像，执行具有多个输入（MI）的神经网络。确保所有模型输入名称均在 sInputNames 中列出，且图像和名称顺序一致。返回一个输出层的结果。该函数不执行任何预处理或转换。
可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
HRESULT ExecuteNeuralNetwork(
    HRESULT             hrPrev,
    ITcVnNeuralNetwork* ipNeuralNetwork,
    PVOID               pSrcImages,
    USHORT              nSrcArraySize,
    PCCH                sInputNames,
    ITcVnImage*&        ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipNeuralNetwork	ITcVnNeuralNetwork* [▶ 1879]	神经网络模型
pSrcImages	PVOID	指向源输入图像数组的指针
nSrcArraySize	USHORT	pSrcImages 引用的数组的大小。pSrcImages 必须引用一个大小等于模型输入数的数组。
sInputNames	PCCH	以“;”分隔的输入名称
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（根据输出层创建适当的目标图像。要处理目标图像，必要时可使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.2 ExecuteNeuralNetwork (MIMO_String)

使用提供的图像，执行具有多个输入和多个输出（MIMO）的神经网络。确保所有模型输入名称均在 sInputNames 中列出，且图像和名称顺序一致。返回指定输出层的结果。该函数不执行任何预处理或转换。可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
HRESULT ExecuteNeuralNetwork(
    HRESULT              hrPrev,
    ITcVnNeuralNetwork* ipNeuralNetwork,
    PVOID                pSrcImages,
    USHORT               nSrcArraySize,
    PCCH                 sInputNames,
    ITcVnContainer*&     ipDestImages,
    PCCH                 sOutputNames
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipNeuralNetwork	ITcVnNeuralNetwork* [▶ 1879]	神经网络模型
pSrcImages	PVOID	指向源输入图像数组的指针
nSrcArraySize	USHORT	pSrcImages 引用的数组的大小。pSrcImages 必须引用一个大小等于模型输入数的数组。
sInputNames	PCCH	以“;”分隔的输入名称
ipDestImages	ITcVnContainer* [▶ 1827]&	包含指定层的输出的图像的容器。
sOutputNames	PCCH	需要输出的输出层的名称，以“;”分隔。如果为空，则提供所有未连接输出层的输出。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.3 ExecuteNeuralNetwork (MISO)

使用提供的图像，执行具有多个输入和单个输出（MISO）的神经网络。确保所有模型输入名称均在 sInputNames 中列出，且图像和名称顺序一致。返回指定输出层的结果。该函数不执行任何预处理或转换。可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
HRESULT ExecuteNeuralNetwork(
    HRESULT              hrPrev,
    ITcVnNeuralNetwork* ipNeuralNetwork,
    PVOID                pSrcImages,
    USHORT               nSrcArraySize,
    PCCH                 sInputNames,
    ITcVnImage*&         ipDestImage,
    PCCH                 sOutputName
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipNeuralNetwork	ITcVnNeuralNetwork* [▶ 1879]	神经网络模型
pSrcImages	PVOID	指向源输入图像数组的指针
nSrcArraySize	USHORT	pSrcImages 引用的数组的大小。pSrcImages 必须引用一个大小等于模型输入数的数组。
sInputNames	PCCH	以“;”分隔的输入名称
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（根据输出层创建适当的目标图像。要处理目标图像，必要时可使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数。
sOutputName	PCCH	指定需要输出的层的名称。如果为空，则提供一个输出层的输出。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.4 ExecuteNeuralNetwork (MO)

使用提供的图像作为输入来执行神经网络，并提供指定输出层的输出。该函数不执行任何预处理或转换。可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
HRESULT ExecuteNeuralNetwork(
    HRESULT              hrPrev,
    ITcVnNeuralNetwork* ipNeuralNetwork,
    ITcVnImage*          ipSrcImage,
    ITcVnContainer*&     ipDestImages,
    ITcVnContainer*      ipOutputNames
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipNeuralNetwork	ITcVnNeuralNetwork* [▶ 1879]	神经网络模型
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）
ipDestImages	ITcVnContainer* [▶ 1827]&	包含指定层的输出的图像的容器。
ipOutputNames	ITcVnContainer* [▶ 1827]	要返回输出的层名称的容器（ContainerType_Vector_String_SINT）。如果设置为零，则将返回所有未连接输出层的输出。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.5 ExecuteNeuralNetwork (SO)

使用提供的图像作为输入来执行神经网络，并提供指定输出层的输出。该函数不执行任何预处理或转换。可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
HRESULT ExecuteNeuralNetwork(
    HRESULT             hrPrev,
    ITcVnNeuralNetwork* ipNeuralNetwork,
    ITcVnImage*         ipSrcImage,
    ITcVnImage*&        ipDestImage,
    PCCH                sOutputName
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipNeuralNetwork	ITcVnNeuralNetwork* [▶ 1879]	神经网络模型

名称	类型	描述
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（根据输出层创建适当的目标图像。要处理目标图像，必要时可使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数。
sOutputName	PCCH	指定需要输出的层的名称。如果为空，则提供一个输出层的输出。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.6 GetNeuralNetworkLayerNames (Exp)

获取神经网络模型中模型层的名称。

语法

定义：

```
HRESULT GetNeuralNetworkLayerNames(
    HRESULT             hrPrev,
    ITcVnNeuralNetwork* ipNeuralNetwork,
    ITcVnContainer*&    ipLayerNames,
    ETcVnLayerNameType  eLayerNameType
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipNeuralNetwork	ITcVnNeuralNetwork* [▶ 1879]	神经网络模型
ipLayerNames	ITcVnContainer* [▶ 1827]&	带有层名称的容器（ContainerType_Vector_String_SINT）
eLayerNameType	ETcVnLayerNameType	层名称类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.7 GetNeuralNetworkLayerNames (String)

获取神经网络模型中模型层的名称。

语法

定义：

```
HRESULT GetNeuralNetworkLayerNames(
    HRESULT             hrPrev,
    ITcVnNeuralNetwork* ipNeuralNetwork,
    PCCH                sLayerNames,
    ULONGLONG           nMaxLength,
    ETcVnLayerNameType  eLayerNameType
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipNeuralNetwork	ITcVnNeuralNetwork* [▶ 1879]	神经网络模型
sLayerNames	PCCH	以“;”分隔的层名称。确保选择足够的字符串大小！
nMaxLength	ULONGLONG	要导出的字符串的最大长度（包括 0 结尾）。如果容器内容较长，则在 nMaxLength - 1 处截断字符串，并添加 0 结尾。在这种情况下，将返回 S_FALSE。
eLayerNameType	ETcVnLayerNameType	层名称类型

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.8 ConvertDataLayout

将数据布局（输入数据中的维度排列）转换为另一种数据布局。

语法

定义：

```
HRESULT ConvertDataLayout(
    HRESULT       hrPrev,
    ITcVnImage*   ipSrcImage,
    ITcVnImage*&  ipDestImage,
    ULONGLONG     nSrcDataLayout,
    ULONGLONG     nDestDataLayout
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的目标图像。）
nSrcDataLayout	ULONGLONG	源图像的数据布局（在枚举 ETcVnDataLayout 中提供常用数据布局。如果 nSrcDataLayout 被设置为 0 或 TCVN_DL_DEFAULT，则 DataLayout 将根据 ipSrcImage 的维度自动确定：0x12 表示 2D，0x123 表示 3D，0x1234 表示 4D，以此类推。）
nDestDataLayout	ULONGLONG	所需的输出数据布局（在枚举 ETcVnDataLayout 中提供常用数据布局。nDestDataLayout 不得等于 0 或 TCVN_DL_DEFAULT。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.9 ExecuteNeuralNetwork

使用提供的图像作为输入来执行神经网络，并提供一个输出层的输出。该函数不执行任何预处理或转换。可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
HRESULT ExecuteNeuralNetwork(
    HRESULT             hrPrev,
    ITcVnNeuralNetwork* ipNeuralNetwork,
    ITcVnImage*         ipSrcImage,
    ITcVnImage*&        ipDestImage
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipNeuralNetwork	ITcVnNeuralNetwork* [▶ 1879]	神经网络模型
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（根据输出层创建适当的目标图像。要处理目标图像，必要时可使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.10 ExecuteNeuralNetwork_MultiLayerOutput

该函数已被弃用，请使用 F_VN_ExecuteNeuralNetwork_MO。使用提供的图像作为输入来执行神经网络，并提供指定输出层的输出。该函数不执行任何预处理或转换。
可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
HRESULT ExecuteNeuralNetwork_MultiLayerOutput(
    HRESULT             hrPrev,
    ITcVnNeuralNetwork* ipNeuralNetwork,
    ITcVnImage*         ipSrcImage,
    ITcVnContainer*&    ipDestImages,
    ITcVnContainer*     ipLayerNames
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipNeuralNetwork	ITcVnNeuralNetwork* [▶ 1879]	神经网络模型

名称	类型	描述
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）
ipDestImages	ITcVnContainer* [▶ 1827]&	包含指定层的输出的图像的容器。
ipLayerNames	ITcVnContainer* [▶ 1827]	要返回输出的层名称的容器（ContainerType_Vector_String_SINT）。如果设置为零，则将返回所有未连接输出层的输出。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.11 ExecuteNeuralNetwork_SingleLayerOutput

该函数已被弃用，请使用 F_VN_ExecuteNeuralNetwork_SO。使用提供的图像作为输入来执行神经网络，并提供指定输出层的输出。该函数不执行任何预处理或转换。
可以使用可用的 TwinCAT Job Tasks 并行执行函数代码实现加速。

语法

定义：

```
HRESULT ExecuteNeuralNetwork_SingleLayerOutput(
    HRESULT              hrPrev,
    ITcVnNeuralNetwork* ipNeuralNetwork,
    ITcVnImage*          ipSrcImage,
    ITcVnImage*&         ipDestImage,
    PCCH                 sLayerName
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipNeuralNetwork	ITcVnNeuralNetwork* [▶ 1879]	神经网络模型
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（必要时必须使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数调整图像布局，使其与模型的预期输入格式相匹配。）

名称	类型	描述
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（根据输出层创建适当的目标图像。要处理目标图像，必要时可使用 F_VN_ConvertDataLayout 或 F_VN_ReshapeImage 等函数。
sLayerName	PCCH	指定需要输出的层的名称。如果为空，则提供一个输出层的输出。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.12 GetNeuralNetworkLayerNames

获取神经网络模型中所有层的名称。如果 bOnlyOuput 标志被设置为 true，则仅返回未连接输出层的名称。

语法

定义：

```
HRESULT GetNeuralNetworkLayerNames(
    HRESULT             hrPrev,
    ITcVnNeuralNetwork* ipNeuralNetwork,
    ITcVnContainer*&    ipLayerNames,
    bool                bOnlyOutputs
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipNeuralNetwork	ITcVnNeuralNetwork* [▶ 1879]	神经网络模型
ipLayerNames	ITcVnContainer* [▶ 1827]&	带有层名称的容器（ContainerType_Vector_String_SINT）
bOnlyOutputs	BOOL	如果为 true，则仅返回未连接输出层的名称。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Neural Network Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.13 NonMaxSuppression

对给定的边界框和相应分数执行非极大值抑制。

语法

定义：

```
HRESULT NonMaxSuppression(
    HRESULT          hrPrev,
    ITcVnContainer*  ipBoundingBoxes,
    ITcVnContainer*  ipScores,
    ITcVnContainer*& ipDestIndcies,
    float            fScoreThreshold,
    float            fOverlapThreshold,
    ITcVnContainer*  ipClassIndcies = nullptr,
    float            fEta = 1.0f,
    LONG             nTopK = 0
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipBoundingBoxes	ITcVnContainer* [▶ 1827]		应用非极大值抑制的边界框的容器（ContainerType_Vector_TcVnRectangle_DINT）
ipScores	ITcVnContainer* [▶ 1827]		每个边界框的相应分数的容器（ContainerType_Vector_REAL）
ipDestIndcies	ITcVnContainer* [▶ 1827]&		应用非极大值抑制后所保留索引的容器（ContainerType_Vector_DINT）
fScoreThreshold	float		用于按分数筛选边界框的阈值 [0.0 至 1.0]
fOverlapThreshold	float		用于确定边界框被视为重叠所需的最小交并比度（IoU）的阈值 [0.0 至 1.0]
ipClassIndcies	ITcVnContainer* [▶ 1827]	nullptr	相应类索引集合的容器（ContainerType_Vector_DINT）
fEta	float	1.0f	在处理过程中用于动态降低重叠阈值的可调阈值 [0.0 至 1.0]。将其设置为 1 可保持重叠阈值的常量值

名称	类型	默认值	描述
nTopK	LONG	0	采用抑制算法的最高分数的边界框的最大数量（必须 >= 0）。将其设置为 0 可将算法应用于所有边界框

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.14 Reduce

通过在行/列之间应用指定操作来减少 ipSrcImage，以获得具有结果值的单行/列/通道。

语法

定义：

```
HRESULT Reduce(
    HRESULT          hrPrev,
    ITcVnImage*      ipSrcImage,
    ITcVnImage*&     ipDestImage,
    LONG             nDim,
    ETcVnReduceType eReduceType
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（USINT 或 REAL 元素，1 通道）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像。如果 ipSrcImage 的类型为 USINT，且减少类型为 SUM 或 AVG，则 ipDestImage 的类型为 REAL；否则，它将保持与 ipSrcImage 相同的类型（如果需要，将创建适当的单通道目标图像。）
nDim	LONG	减少 ITcVnImage 的索引：0 将矩阵减少为单行，1 将其减少为单列，2 将其减少为单通道
eReduceType	ETcVnReduceType [▶ 1689]	减少操作

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.15 ReduceArg

沿提供的轴查找最大/最小元素的索引，并将 ipSrcImage 减少为单行/列/通道。

语法

定义：

```
HRESULT ReduceArg(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    LONG         nDim,
    LONG         eReduceType
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像
ipDestImage	ITcVnImage* [▶ 1863]&	DINT 类型的目标图像（如果需要，将创建适当的单通道目标图像。）
nDim	LONG	减少 ipSrcImage 的索引：0 将矩阵减少为单行，1 将其减少为单列，2 将其减少为单通道
eReduceType	LONG	减少操作 REDUCE_MAX 或 REDUCE_MIN

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.16 ReshapeImage

改变 ITcVnImage 的形状。新形状中元素的总数必须等于 ipSrcImage 中元素的总数。

语法

定义：

```
HRESULT ReshapeImage(
    HRESULT            hrPrev,
    ITcVnImage*        ipSrcImage,
    ITcVnImage*&       ipDestImage,
    TcVnArray33_UDINT& aNewShape
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（1 通道）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的单通道目标图像。）
aNewShape	TcVnArray33_UDINT [▶ 1641]&	指定 ipDestImage 的形状。第一个条目是维数（最多 32），然后是每个维度的大小。如果一个维度被设置为零，其值将根据 ipSrcImage 的元素总数自动计算（仅能有一个维度被设置为零。）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.17 Sigmoid

将逻辑函数 1 / (1 + exp(-fSteepness * (x - fShift))) 逐元素应用到单通道或多通道 REAL 图像中。

语法

定义：

```
HRESULT Sigmoid(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    float        fShift = 0.0f,
    float        fSteepness = 1.0f
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（REAL）
ipDestImage	ITcVnImage* [▶ 1863]&		目标图像（如果需要，将创建适当的目标图像。）
fShift	float	0.0f	确定逻辑函数转换的强度级别。
fSteepness	float	1.0f	控制转换的清晰度。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.23.18 SoftMax

将 softmax 函数逐元素应用到单通道 REAL 图像中。

语法

定义：

```
HRESULT SoftMax(
    HRESULT      hrPrev,
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipDestImage,
    bool         bFlatten
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（REAL，1 通道）
ipDestImage	ITcVnImage* [▶ 1863]&	目标图像（如果需要，将创建适当的单通道目标图像。）
bFlatten	BOOL	如果为 true，生成的图像将被重塑为单行。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.24 OCR

该组包含**光学**字符**识别**功能。

OCR 功能 [▶ 2610]识别图像中的字符，并将识别出的字符作为字符串返回。分类基于经典 Machine Learning（机器学习）模型。这些模型以训练形式提供，因此可以直接使用功能。无需额外设置或训练。支持的字符和字体取决于训练中使用的数据。因此，有多种模型，其功能范围以及一般前提条件和要求如下所述。

对于图像中识别出的所有字符，模型会将每个字符分配到匹配度最高的已知类别。因此，结果总是包含所使用模型的已知字符。不包括拒绝或删除未知字符。

函数仅支持单行字符串；必须将多行字符串分为单独的 ROI，并通过多次函数调用读取。

识别字符的一般要求

- 字符高度最小为 20 像素
- 线宽最小为 3 像素
- 点最小为 3 x 3 像素
- 线最小为 3 x 6 像素
- 字符间距/分隔最小为 4 像素
- 字符不得重叠
- 仅水平对齐/字符排列最大 ± 6° 偏差
- 字符线/轮廓不得中断
- 字符不得镜像或颠倒

图像的一般要求：

- ROI 只包含待识别字符和字符周围的无干扰区
- 在 ROI 内，字符不得被其他轮廓（如矩形）包围
- 字符与背景要对比鲜明
- 背景均匀、无噪音或干扰、不透明

对字体的要求：

- 只能使用字宽相等的比例字体
- 较大的间隙始终会被识别为单个空格
- 仅使用无衬线字体，如 Arial、Tahoma、Courier、Univers、Frutiger、Verdana、OCR-B 等
- 无混合字体
- 无圆点打印字体或斜体字体

模型：

枚举 ETcVnOcrModelType [▶ 1685] 允许访问以下模型：

- `OMT_NUMBERS`
  - 支持数字分类
  - 包含 `0-9` 的字符

- OMT_NUMBERS_SC
  - 支持数字和特殊字符分类
  - 包含从 0-9 开始的字符和六个特殊字符 . / - : = +
- OMT_UCLETTERS
  - 支持大写字母分类
  - 包含 A-Z 的字符
- OMT_NUMBERS_SC_UCLETTERS
  - 支持数字、特殊字符和大写字母分类
  - 包含从 0-9 开始的字符、六个特殊字符 . / - : = + 和 A-Z

在使用组合模型（OMT_NUMBERS_SC_UCLETTERS）时，由于某些字符非常相似，可能会出现混淆。例如 O 和 0、S 和 5 以及 B 和 8。

如果数字或字母的位置已知，则应使用可选格式规格（sPattern）和单独模型的组合作为替代。另外，也可以单独分析结果字符串（ipCharacters）的字符，这样就可以单独决定是否接受 0 而非 O。

初始化功能

为了能够将 OCR 功能与一个或多个模型一起使用，必须首先使用功能块 InitializeFunction [▶ 2626] 初始化该功能以及相应的模型。

加载模型所需的时间取决于所使用 IPC 的性能和模型大小，可能需要几百毫秒甚至几秒钟。因此，在任务循环时间较短的情况下，预计会出现循环时间超限。

如果要使用组合模型 OMT_NUMBERS_SC_UCLETTERS，由于文件大小的原因，路由器内存必须至少设置为 512 MB。

具体大小取决于路由器内存的其他用途和任何现有分段储存。例如，如果功能块 InitializeFunction 返回返回代码 0x80004005，则表示路由器内存不足。因此，应首先检查路由器内存，必要时增加内存。加载模型后，大部分内存会重新释放，供应用使用。

HRESULT 的解释

- 如果识别到图像上的字符，标准函数返回 S_OK。使用可选参数时，这取决于是否指定了 sPattern。如果没有定义格式规格，则返回对应于标准函数。如果已传递格式规格，只会在识别字符与模板规格相匹配的情况下才会返回 S_OK。
- 如果标准函数执行成功，但未找到任何字符，或者找到的字符与可选模板规格不匹配，则返回 S_FALSE。
- 如果可选 sPattern 规格缺少传输模型，则返回 HRESULT = ADSERR_DEVICE_NOTINIT。
- 如果向标准函数传递了多个模型，则始终返回 HRESULT = ADSERR_DEVICE_INVALIDPARM，因为该函数仅支持每次调用一个模型。使用可选参数时，这取决于是否指定了 sPattern。如果 sPattern 为空，则只使用 TCVN_OMT_NUMBERS_SC_UCLETTERS 模型。如果没有传递该模型，则返回 HRESULT = ADSERR_DEVICE_INVALIDPARM。

使用 sPattern 的示例

在使用可选参数 sPattern 和 eOcrOptions 时，还可能出现其他情况。根据输入图像上的字符组合、来自 sPattern 和 eOcrOptions 的信息以及识别的字符，如果出现不匹配，则返回 S_FALSE。然后，还可以对结果字符串（ipCharacters）中的字符进行单独分析，例如，如果长度不同，也会返回 S_FALSE。这些样本涉及 OMT_NUMBERS_SC_UCLETTERS 模型的使用。

ipSrcImage 上的字符	sPattern	eOcrOptions	ipCharacters	HRESULT
12/34	dd.dd		12/34	S_OK
12534	dd.dd		12534	S_OK
12/34	dd!dd		1234	S_OK
12 34	dd!d		124	S_OK
12534	dd!dd		1234	S_OK

ipSrcImage 上的字符	sPattern	eOcrOptions	ipCharacters	HRESULT
12 34	dd_dd		12 34	S_OK
AB12/	uudd#		AB12/	S_OK
12 34	dddd	WITHBLANKS	12 34	S_OK
12/4	dd#d	WITHBLANKS	12/4	S_OK
12 34	dd_dd	WITHBLANKS	12 34	S_OK
12 34 56	dd_dddd	WITHBLANKS	12 34 56	S_OK
12 3 4	dd.dd	WITHBLANKS	12 3 4	S_FALSE
12 34	dd!dd		124	S_FALSE
12 34	dd.dd		1234	S_FALSE
1234	!dddd		234	S_FALSE
12 34	dd_dd		12.34	S_FALSE
12 3	dd_dd		123	S_FALSE
12 3 4	dd_d_d		123 4	S_FALSE
AB12/	uddd#		AB12/	S_FALSE
AB12/	uuddd		AB12/	S_FALSE

6.2.3.24.1 OCR

检测并识别二值图像中的字符（黑底白字）。
可使用可用的 TwinCAT Job Task 并行执行函数代码实现加速。

语法

定义：

```
HRESULT OCR(
    HRESULT           hrPrev,
    ITcVnImage*       ipSrcImage,
    ULONGLONG         eModel,
    ITcVnContainer*& ipCharacters,
    float&            fMinConfidence,
    PCCH              sPattern = "",
    ULONG             eOcrOptions = OO_NONE,
    ITcVnContainer** pipBoundingBoxes = nullptr,
    ITcVnContainer** pipConfidences = nullptr
)
```

参数

名称	类型	默认值	描述
hrPrev	HRESULT [▶ 135]		表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipSrcImage	ITcVnImage* [▶ 1863]		源图像（USINT，1 通道，二值）
eModel	ULONGLONG		选择 OCR 模型类型（ETcVnOcrModelType）
ipCharacters	ITcVnContainer* [▶ 1827]&		返回识别的字符（ContainerType_Vector_String_SINT）
fMinConfidence	float&		返回置信度的最小值
sPattern	PCCH	""	包含字符显示格式的字符串模式

名称	类型	默认值	描述
eOcrOptions	ULONG	OO_NONE	指定必须对函数应用的选项（ETcVnOcrOptions）
pipBoundingBoxes	ITcVnContainer* [▶ 1827]*	nullptr	返回已识别字符的相应边界框（ContainerType_Vector_TcVnRectangle_DINT，可选，如果不需要，则设为 0）
pipConfidences	ITcVnContainer* [▶ 1827]*	nullptr	返回已识别字符的相应分类置信度（ContainerType_Vector_REAL，可选，如果不需要，则设为 0）

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Vision OCR

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.25 纹理分析

该组包含用于分析图像纹理的函数。

函数

粒度属性

- Granulometry [▶ 2611]

Haralick 特征

- HaralickFeatures [▶ 2612]

6.2.3.25.1 Granulometry

使用形态学运算计算纹理图像的粒度。

语法

定义：

```
HRESULT Granulometry(
    HRESULT                     hrPrev,
    ITcVnImage*                 ipImage,
    ITcVnContainer*&            ipIntensityDiffs,
    ETcVnMorphologicalOperator  eMorphType,
    ULONG                       nMinSize,
    ULONG                       nMaxSize,
    ULONG                       nStep
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]	源图像（1 通道）
ipIntensityDiffs	ITcVnContainer* [▶ 1827]&	返回连续结构元素大小（CTcVnContainer_Vector_LREAL）之间的强度差异，可用作纹理描述的特征。
eMorphType	ETcVnMorphologicalOperator [▶ 1682]	仅支持开运算或闭运算。
nMinSize	ULONG	最小结构元素大小（奇数，>= 3）
nMaxSize	ULONG	最大结构元素大小（>= nMinSize）
nStep	ULONG	步长，nMinSize 按此步长递增，直至超过 nMaxSize（偶数，>= 2）。

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.3.25.2 HaralickFeatures

计算灰度图像的哈拉利克（Haralick）特征，该特征描述纹理。

语法

定义：

```
HRESULT HaralickFeatures(
    HRESULT          hrPrev,
    ITcVnImage*      ipImage,
    ITcVnContainer*& ipFeatures,
    ULONG            nDist
)
```

参数

名称	类型	描述
hrPrev	HRESULT [▶ 135]	表示先前运算结果的 HRESULT（如果 SUCCEED（hrPrev）等于 false，则不执行任何操作。）
ipImage	ITcVnImage* [▶ 1863]	源图像（1 通道，USINT）
ipFeatures	ITcVnContainer* [▶ 1827]&	返回计算出的哈拉利克（Haralick）特征（CTcVnContainer_Vector_LREAL）。返回的特征集包含前 13 个哈拉

名称	类型	描述
		利克（Haralick）特征中每个特征的平均值和标准偏差，分别按 0、45、90 和 135 度计算。因此，总共将有 26 个特征（f1 平均值、f1 标准偏差、f2 平均值、f2 标准偏差、...、f13 平均值、f13 标准偏差）。
nDist	ULONG	Distance

返回值

HRESULT [▶ 135]

所需的许可证

TC3 Machine Learning Realtime Inference

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4 功能块

与 GigE Vision 相机或文件系统的通信以及复杂图像处理函数通过功能块进行映射。

功能块分组

TwinCAT Vision 库中的所有功能块按主题分为以下几组：

- 相机寄存器访问 [▶ 2614]
- 文件访问 [▶ 2625]
- 图像采集 [▶ 2638]
- 图像处理 [▶ 2654]

6.2.4.1 许可证概述

以下列表显示了 TwinCAT Vision 功能块与可用许可证的分配情况（见许可证模型 [▶ 12]）。它显示了某些功能块的应用需要哪些许可证。

始终需要基本许可证 TC3 Vision Base（TF7100）。

TC3 Vision Base

FileSourceControl [▶ 2639]

GevCameraControl [▶ 2642]

InitializeFunction [▶ 2626]

ReadCalibrationPattern [▶ 2627]

ReadCalibrationResult [▶ 2628]

ReadContainer [▶ 2629]

ReadImage [▶ 2631]

ReadMemory [▶ 2615]

ReadMlModel [▶ 2631]

ReadNeuralNetwork [▶ 2637]

ReadRegister_REAL [▶ 2617]

ReadRegister_UDINT [▶ 2618]

ReadRegister_ULINT [▶ 2619]

SimpleCameraControl [▶ 2652]

WriteCalibrationResult [▶ 2632]

WriteContainer [▶ 2634]

WriteImage [▶ 2635]

WriteMemory [▶ 2620]

WriteMlModel [▶ 2636]

WriteRegister_REAL [▶ 2621]

WriteRegister_UDINT [▶ 2622]

WriteRegister_ULINT [▶ 2623]

WriteRegisters_UDINT [▶ 2624]

TC3 Vision Matching

GeneralizedHoughBallard [▶ 2655]

SSIM [▶ 2658]

6.2.4.2 相机寄存器访问

该组包含用于读写相机寄存器或内存区域的功能块。

功能块

UDINT 寄存器（4 字节）

- ReadRegister_UDINT [▶ 2618]
- WriteRegister_UDINT [▶ 2622]
- WriteRegisters_UDINT [▶ 2624]

ULINT 寄存器（8 字节）

- ReadRegister_ULINT [▶ 2619]
- WriteRegister_ULINT [▶ 2623]

REAL 寄存器（4 字节）

- ReadRegister_REAL [▶ 2617]
- WriteRegister_REAL [▶ 2621]

更大的内存区域

- ReadMemory [▶ 2615]
- WriteMemory [▶ 2620]

参数

以下参数与本组中所有功能块的工作方式相似：

地址

nAddress 地址指定要读取或保存的相机寄存器。GVCP_REGISTER_ADDRESS 类型只是 UDINT 的别名；因此，地址长度必须为 4 字节（或 32 位）。对于相机的较大内存区域（请参见 ReadMemory [▶ 2615]），地址指向要读取或保存区域的起始位置。

按顺序调用功能块

请注意，每次调用这些功能块中的某一个功能块时，您必须等待来自相机的相应响应。因此，同一时间只能有一个功能块处于活动状态。在进行下一次调用或使用另一个功能块之前，您必须等待当前活动的功能块不再为 bBusy 状态。因此，如果要使用多个功能块或者读取或写入寄存器值，我们建议采用顺序编程的方式。

执行函数

写入和读取命令的触发器分别是 bWrite 和 bRead。它们由一个上升边缘触发（从 false 更改为 true）。相关输入参数在内部保存，因此在执行过程中参数的改变不会产生任何影响。触发器的下降边缘也同样没有影响。

调用功能块

请注意，功能块的参数描述仅在实际调用功能块时才会生效。

超时

为了在出现未知错误时不必长时间等待功能块的执行，可以通过 nTimeout 设置允许的最大执行时间。如果在触发命令触发器后超过这一时间仍无成功结果，执行将被中止。然后，超时错误可由两个输出 bError 和 nErrorId 识别。

工作状态

输出 bBusy 指示功能块是否繁忙。如果为 bBusy == true，则不能触发新命令。

通过适当的 if 查询，可以确定功能块的执行是否完成：

```
if (!fbReadOrWrite.bBusy)
{
    // Function block execution is complete or error occurred.
    // Next write-command can be issued.
}
```

错误状态

如果适用，错误状态通过 bError 以二进制方式表示错误，而 nErrorId 输出错误代码（ADS 返回代码 [▶ 2857]）。一旦有新命令被触发，错误状态就会被重置。如果功能块繁忙（bBusy == true），错误代码为 PENDING (0x71E)。

错误代码的十六进制值可以如下提取：

```
if (fbReadOrWrite.bError)
{
    // Show relevant error code for debugging purposes:
    nErrorCode = fbReadOrWrite.nErrorId & 0xFFF;
}
```

6.2.4.2.1 ReadMemory

该 FB 从相机读取连续内存位置。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()

语法

构造函数：

```
ReadMemory(
    OTCID oidITcVnGevImageProvider
)
```

执行主程序：

```
FB_Main()

FB_Main(
    GVCP_REGISTER_ADDRESS nAddress,
    USHORT                nLength,
    PVOID                 pBuffer,
    bool                  bRead
)

FB_Main(
    bool bRead
)
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 1640]		从相机读取的第一个字节的地址（必须 32 位对齐）
nLength	USHORT		从 nAddress 开始读取的字节数（必须是 4 的倍数，不超过 536）
pBuffer	PVOID		该指针指向读取的内存内容写入的缓冲区（缓冲区的大小必须至少为 nLength 字节！）
bRead	BOOL		读取内存由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.2.2 ReadRegister_REAL

该 FB 从相机读取一个 REAL 寄存器。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()

语法

构造函数：

```
ReadRegister_REAL(
    OTCID oidITcVnGevImageProvider
)
```

执行主程序：

```
FB_Main()

FB_Main(
    GVCP_REGISTER_ADDRESS nAddress,
    bool                  bRead
)

FB_Main(
    bool bRead
)
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 1640]		应读取的寄存器的地址
nEndian	USHORT		可选：寄存器的字节序。0 = 大，1 = 小。默认值：0
bRead	BOOL		读取寄存器由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
fValue	float	读取的值
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.2.3 ReadRegister_UDINT

该 FB 从相机读取一个 DINT 寄存器。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()

语法

构造函数：

```
ReadRegister_UDINT(
    OTCID oidITcVnGevImageProvider
)
```

执行主程序：

```
FB_Main()

FB_Main(
    GVCP_REGISTER_ADDRESS nAddress,
    bool                  bRead
)

FB_Main(
    bool bRead
)
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 1640]		应读取的寄存器的地址
nEndian	USHORT		可选：寄存器的字节序。0 = 大，1 = 小。默认值：0
bRead	BOOL		读取寄存器由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
nValue	ULONG	读取的值
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.2.4 ReadRegister_ULINT

该 FB 从相机读取一个 ULINT 寄存器。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()

语法

构造函数：

```
ReadRegister_ULINT(
    OTCID oidITcVnGevImageProvider
)
```

执行主程序：

```
FB_Main()

FB_Main(
    GVCP_REGISTER_ADDRESS nAddress,
    bool                  bRead
)

FB_Main(
    bool bRead
)
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 1640]		应读取的寄存器的地址
nEndian	USHORT		可选：寄存器的字节序。0 = 大，1 = 小。默认值：0
bRead	BOOL		读取寄存器由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
nValue	ULONGLONG	读取的值
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.2.5 WriteMemory

该 FB 将数据写入相机的连续内存位置。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()

语法

构造函数：

```
WriteMemory(
    OTCID oidITcVnGevImageProvider
)
```

执行主程序：

```
FB_Main()

FB_Main(
    GVCP_REGISTER_ADDRESS nAddress,
    USHORT                nLength,
    PVOID                 pData,
    bool                  bWrite
)

FB_Main(
    bool bWrite
)
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 1640]		要写入相机的第一个字节的地址（必须 32 位对齐）
nLength	USHORT		从 nAddress 开始写入的字节数（必须是 4 的倍数，不超过 536）
pData	PVOID		该指针指向应写入相机的数据（大小必须至少为 nLength 字节！）
bWrite	BOOL		写入内存由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。
nBytesWritten	USHORT	已成功写入相机的字节数。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.2.6 WriteRegister_REAL

该 FB 将一个 REAL 值写入相机的寄存器。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()

语法

构造函数：

```
WriteRegister_REAL(
    OTCID oidITcVnGevImageProvider
)
```

执行主程序：

```
FB_Main()

FB_Main(
    GVCP_REGISTER_ADDRESS nAddress,
    float                 fValue,
    bool                  bWrite
)

FB_Main(
    bool bWrite
)
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 1640]		应写入的寄存器的地址
fValue	float		要写入的值
nEndian	USHORT		可选：寄存器的字节序。0 = 大，1 = 小。默认值：0
bWrite	BOOL		写入寄存器由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.2.7 WriteRegister_UDINT

该 FB 将一个 DINT 值写入相机的寄存器。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()

语法

构造函数：

```
WriteRegister_UDINT(
    OTCID oidTTcVnGevImageProvider
)
```

执行主程序：

```
FB_Main()

FB_Main(
    GVCP_REGISTER_ADDRESS nAddress,
    ULONG                 nValue,
    bool                  bWrite
)

FB_Main(
    bool bWrite
)
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 1640]		应写入的寄存器的地址
nValue	ULONG		要写入的值
nEndian	USHORT		可选：寄存器的字节序。0 = 大，1 = 小。默认值：0
bWrite	BOOL		写入寄存器由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.2.8 WriteRegister_ULINT

该 FB 将一个 ULINT 值写入相机的寄存器。需要开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()

语法

构造函数：

```
WriteRegister_ULINT(
    OTCID oidITcVnGevImageProvider
)
```

执行主程序：

```
FB_Main()

FB_Main(
    GVCP_REGISTER_ADDRESS nAddress,
    ULONGLONG             nValue,
    bool                  bWrite
)

FB_Main(
    bool bWrite
)
```

输入

名称	类型	默认值	描述
nAddress	GVCP_REGISTER_ADDRESS [▶ 1640]		应写入的寄存器的地址
nValue	ULONGLONG		要写入的值
nEndian	USHORT		可选：寄存器的字节序。0 = 大，1 = 小。默认值：0
bWrite	BOOL		写入寄存器由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.2.9 WriteRegisters_UDINT

该 FB 在相机上写入多个 UDINT 寄存器。需要指定的寄存器具有相同的字节序，以及开放的控制通道（例如，在此之前调用 FB_VN_GevCameraControl.OpenCamera()）。

语法

构造函数：

```
WriteRegisters_UDINT(
    OTCID oidITcVnGevImageProvider
)
```

执行主程序：

```
FB_Main()

FB_Main(
    GVCP_ADDRESS_VALUE_PAIR* pAddressValuePairs,
    USHORT                   nArraySize,
    bool                     bWrite
)

FB_Main(
    bool bWrite
)
```

输入

名称	类型	默认值	描述
pAddressValuePairs	GVCP_ADDRESS_VALUE_PAIR [▶ 1700]*		一个数组，包含应写入的寄存器的地址/值对
nArraySize	USHORT		pAddressValuePairs 中地址/值对的数量
nEndian	USHORT		可选：寄存器的字节序。0 = 大，1 = 小。默认：0
bWrite	bool		写入寄存器由该输入端的上升沿触发。
nTimeout	unsigned int	VISION_ADS_TIMEOUT	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	bool	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	bool	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

6.2.4.3 文件访问

该组包含用于在文件系统上读写数据的功能块。

功能块

图像

- ReadImage [▶ 2631]
- WriteImage [▶ 2635]

容器

- ReadContainer [▶ 2629]
- WriteContainer [▶ 2634]

机器学习模型

- ReadMlModel [▶ 2631]
- WriteMlModel [▶ 2636]

标定板

- ReadCalibrationPattern [▶ 2627]
- ReadCalibrationResult [▶ 2628]
- WriteCalibrationResult [▶ 2632]

函数初始化

- InitializeFunction [▶ 2626]

神经网络

- ReadNeuralNetwork [▶ 2637]

通用

这些功能块异步工作；从第一次执行到所需结果的时间跨越几个循环。每次读/写操作的实际持续时间取决于多个因素，如图像的大小和操作系统的行为。以下参数对所有功能块均一样，因此在此只集中描述一次。

参数

文件路径

文件路径 `sFilePath` 指定保存文件的位置或加载哪个文件。可以使用绝对和相对路径。在相对路径的情况下，服务配置 [▶ 68]中设置的默认路径被作为基础。

文件路径在写入命令 `bWrite` 的上升边缘上进行内部存储，因此，在以后的循环中更改路径没有影响。

执行函数

写入和读取命令的触发器分别是 `bWrite` 和 `bRead`。它们由一个上升边缘触发（从 `false` 更改为 `true`）。相关输入参数在内部保存，因此在执行过程中参数的改变不会产生任何影响。触发器的下降边缘也同样没有影响。

调用功能块

请注意，功能块的参数描述仅在实际调用功能块时才会生效。

超时

为了在出现未知错误时不必长时间等待功能块的执行，可以通过 `nTimeout` 设置允许的最大执行时间。如果在触发命令触发器后超过这一时间仍无成功结果，执行将被中止。然后，超时错误可由两个输出 `bError` 和 `nErrorId` 识别。

工作状态

输出 bBusy 指示功能块是否繁忙。如果为 bBusy == true，则不能触发新命令。

通过适当的 if 查询，可以确定功能块的执行是否完成：

```
if (!fbReadOrWrite.bBusy)
{
    // Function block execution is complete or error occurred.
    // Next write-command can be issued.
}
```

错误状态

如果适用，错误状态通过 bError 以二进制方式表示错误，而 nErrorId 输出错误代码（ADS 返回代码 [▶ 2857]）。一旦有新命令被触发，错误状态就会被重置。如果功能块繁忙（bBusy == true），错误代码为 PENDING (0x71E)。

错误代码的十六进制值可以如下提取：

```
if (fbReadOrWrite.bError)
{
    // Show relevant error code for debugging purposes:
    nErrorCode = fbReadOrWrite.nErrorId & 0xFFF;
}
```

最常见的错误代码包括：

Hex	Dec	名称	含义
0x006	6	TARGETPORTNOTFOUND	ADS 服务器无法访问。基本上是在 TwinCAT Vision Service 没有启动时发生。请参见 视觉服务不启动 [▶ 2843]。
0x700	1792	ERROR	保存或加载失败时出现一般错误。这通常表明指定的文件格式不受支持或者文件路径无法访问。 该错误通常发生在 TwinCAT Vision Service [▶ 68] 中。
0x70B	1803	INVALIDPARM	无效参数。这通常表明接口指针 [▶ 132]无效。
0x719	1817	TIMEOUT	超时。加载或保存已超过最大执行时间 nTimeout。一般来说，主要发生在 ADS 负载较重的情况下。
0x71E	1822	PENDING	并非真正的错误，因此这里没有设置输出 bError。该代码发出信号，表明功能块处于繁忙状态。如果在功能块繁忙时记录到新上升边缘，其将与输出 bBusy 同时发生。

6.2.4.3.1 InitializeFunction

该 FB 将初始化选定的函数。

语法

构造函数：

```
InitializeFunction()
```

执行主程序：

```
FB_Main()

FB_Main(
    ETcVnInitializableFunction eFunction,
    ULONGLONG                  nOptions,
    bool                       bStart
)

FB_Main(
    bool bStart
)
```

输入

名称	类型	默认值	描述
eFunction	ETcVnInitializableFunction [▶ 1679]		选择要初始化的函数
nOptions	ULONGLONG	0	为函数选择初始化选项
bStart	BOOL		函数初始化由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.3.2 ReadCalibrationPattern

该 FB 可从 targte PC 上的 xml 文件中读取标定板参考点。

语法

构造函数：

```
ReadCalibrationPattern()
```

执行主程序：

```
FB_Main()

FB_Main(
    PCCH             sFilePath,
    ITcVnContainer** pipDestContainer,
    bool             bRead
)

FB_Main(
    bool bRead
)
```

输入

名称	类型	默认值	描述
sFilePath	PCCH		文件的完整路径或目标计算机上默认目录的相对路径
pipDestContainer	ITcVnContainer* [▶ 1827]*		返回包含标定板点的容器（ContainerType_Vector_TcVnPoint3_REAL）
bRead	BOOL		读取文件由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.3.3 ReadCalibrationResult

该 FB 可从目标计算机上的 json 文件读取标定结果。

语法

构造函数：

```
ReadCalibrationResult()
```

执行主程序：

```
FB_Main()

FB_Main(
    PCCH                  sFilePath,
    TcVnMatrix3x3_LREAL*  paCameraMatrix,
    TcVnArray8_LREAL*     paDistortionCoefficients,
    TcVnMatrix3x3_LREAL*  paRotationMatrix,
    TcVnVector3_LREAL*    paTranslationVector,
    bool                  bRead
)

FB_Main(
    bool bRead
)
```

输入

名称	类型	默认值	描述
sFilePath	PCCH		文件的完整路径或目标计算机上默认目录的相对路径
paCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]*		返回相机矩阵
paDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]*		返回畸变系数
paRotationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]*		返回旋转矩阵
paTranslationVector	TcVnVector3_LREAL [▶ 1641]*		返回平移向量
bRead	BOOL		读取文件由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.3.4 ReadContainer

该 FB 从目标计算机上的 xml 文件读取容器。

语法

构造函数：

```
ReadContainer()
```

执行主程序：

```
FB_Main()

FB_Main(
    PCCH              sFilePath,
    ITcVnContainer** pipDestContainer,
    bool              bRead
)

FB_Main(
    PCCH              sFilePath,
    ITcVnContainer** pipDestContainer,
    GUID              nDestTypeGuid,
    bool              bRead
)

FB_Main(
    bool bRead
)
```

输入

名称	类型	默认值	描述
sFilePath	PCCH		文件的完整路径或目标计算机上默认目录的相对路径
pipDestContainer	ITcVnContainer* [▶ 1827]*		返回包含已加载内容的容器
nDestTypeGuid	GUID		要返回的容器的类型 ID。如果未设置该参数，将自动确定合适的容器类型（csv 文件除外）。
bRead	BOOL		读取文件由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.3.5 ReadImage

该 FB 从目标计算机上的文件读取图像。

语法

构造函数：

```
ReadImage()
```

执行主程序：

```
FB_Main()

FB_Main(
    PCCH          sFilePath,
    ITcVnImage** pipDestImage,
    bool          bRead
)

FB_Main(
    bool bRead
)
```

输入

名称	类型	默认值	描述
sFilePath	PCCH		文件的完整路径或目标计算机上默认图像目录的相对路径
pipDestImage	ITcVnImage* [▶ 1863]*		返回包含已加载内容的图像
bRead	BOOL		读取文件由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.3.6 ReadMlModel

该 FB 可从目标计算机上的文件读取机器学习模型。

语法

构造函数：

```
ReadMlModel()
```

执行主程序：

```
FB_Main()

FB_Main(
    PCCH            sFilePath,
    ITcVnMlModel** pipDestMlModel,
    bool            bRead
)

FB_Main(
    bool bRead
)
```

输入

名称	类型	默认值	描述
sFilePath	PCCH		文件的完整路径或目标计算机上默认目录的相对路径
pipDestMlModel	ITcVnMlModel* [▶ 1871]*		返回已加载的机器学习模型
bRead	BOOL		读取文件由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.3.7 WriteCalibrationResult

该 FB 将标定结果以 json 文件的形式写入硬盘。

语法

构造函数：

```
WriteCalibrationResult()
```

执行主程序：

```
FB_Main()

FB_Main(
    PCCH                  sFilePath,
    TcVnMatrix3x3_LREAL*  paCameraMatrix,
    TcVnArray8_LREAL*     paDistortionCoefficients,
    TcVnMatrix3x3_LREAL*  paRotationMatrix,
    TcVnVector3_LREAL*    paTranslationVector,
    bool                  bWrite
)

FB_Main(
    bool bWrite
)
```

输入

名称	类型	默认值	描述
sFilePath	PCCH		文件的完整路径或目标计算机上默认目录的相对路径（例如，只有文件名）。如果字符串为空，将生成一个文件名（包含当前时间）。
paCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]*		返回相机矩阵
paDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]*		返回畸变系数
paRotationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]*		返回旋转矩阵
paTranslationVector	TcVnVector3_LREAL [▶ 1641]*		返回平移向量
bWrite	BOOL		写入容器由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.3.8 WriteContainer

该 FB 将容器作为 xml 文件写入硬盘。

语法

构造函数：

```
WriteContainer()
```

执行主程序：

```
FB_Main()

FB_Main(
    ITcVnContainer**           pipContainer,
    PCCH                       sFilePath,
    ETcVnContainerExportFormat eExportFormat,
    bool                       bWrite
)

FB_Main(
    bool bWrite
)
```

输入

名称	类型	默认值	描述
pipContainer	ITcVnContainer* [▶ 1827]*		要写入的容器
sFilePath	PCCH		文件的完整路径或目标计算机上默认目录的相对路径（例如，只有文件名）。如果字符串为空，将生成一个文件名（包含当前时间）。
eExportFormat	ETcVnContainerExportFormat [▶ 1666]	CEF_XML	容器导出格式（默认为可人工读取的 xml 格式）
bWrite	BOOL		写入容器由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.3.9 WriteImage

该 FB 将图像写入硬盘。

语法

构造函数：

```
WriteImage()
```

执行主程序：

```
FB_Main()

FB_Main(
    ITcVnImage** pipImage,
    PCCH         sFilePath,
    bool         bWrite
)

FB_Main(
    bool bWrite
)
```

输入

名称	类型	默认值	描述
pipImage	ITcVnImage* [▶ 1863]*		要写入的图像
sFilePath	PCCH		文件的完整路径或目标计算机上默认目录的相对路径（例如，只有文件名）。如果字符串为空，将生成一个文件名（包含当前时间）。
bWrite	BOOL		写入图像由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.3.10 WriteMlModel

该 FB 将机器学习模型写入硬盘。

语法

构造函数：

```
WriteMlModel()
```

执行主程序：

```
FB_Main()

FB_Main(
    ITcVnMlModel** pipMlModel,
    PCCH           sFilePath,
    bool           bWrite
)

FB_Main(
    bool bWrite
)
```

输入

名称	类型	默认值	描述
pipMlModel	ITcVnMlModel* [▶ 1871]*		要写入的机器学习模型
sFilePath	PCCH		文件的完整路径或目标计算机上默认目录的相对路径（例如，只有文件名）。如果字符串为空，将生成一个文件名（包含当前时间）。
bWrite	BOOL		写入机器学习模型由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.3.11 ReadNeuralNetwork

该 FB 从目标计算上的文件读取神经网络。

语法

构造函数：

```
ReadNeuralNetwork()
```

执行主程序：

```
FB_Main()

FB_Main(
    PCCH                 sFilePath,
    ITcVnNeuralNetwork** pipDestNeuralNetwork,
    bool                 bRead
)

FB_Main(
    PCCH                 sFilePath,
    ITcVnNeuralNetwork** pipDestNeuralNetwork,
    TcVnArray33_UDINT*   paInputShape,
    bool                 bRead
)

FB_Main(
    bool bRead
)
```

输入

名称	类型	默认值	描述
sFilePath	PCCH		文件的完整路径或目标计算机上默认目录的相对路径
pipDestNeuralNetwork	ITcVnNeuralNetwork* [▶ 1879]*		返回神经网络
paInputShape	TcVnArray33_UDINT [▶ 1641]*		用于测试神经网络模型的执行情况的可选输入形状。如果执行失败，将返回错误代码 INCOMPATIBLE。设置 aInputShape = 0 或 aInputShape[0] = 0 将禁用执行测试。
bRead	BOOL		读取文件由该输入端的上升沿触发。
nTimeout	unsigned int	vision_ads_timeout	表示函数被取消前的时间。

输出

名称	类型	描述
bBusy	BOOL	在功能块执行命令之前，该输出保持为 TRUE，但最长持续时间为“nTimeout”输入所提供的时间。当 bBusy = TRUE 时，输入端将不接受任何新命令。
bError	BOOL	一旦在执行命令过程中出错，该输出将切换为 TRUE。特定命令的错误代码包含在“nErrorId”中。如果功能块出现超时错误，“Error”为 TRUE，“nErrorId”为 1861（十六进制 0x745）。在输入端执行命令后重置为 FALSE。
nErrorId	ULONG	包含 ADS 错误代码或最后执行的命令的特定错误代码。在输入端执行命令后重置为 0。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.4 图像采集

该组包含用于通过 Vision 设备（GigE Vision 相机 [▶ 71]和文件源 [▶ 123]对象）采集图像的功能块

功能块

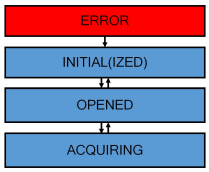
- SimpleCameraControl [▶ 2652]，用于相机和文件源
- GevCameraControl [▶ 2642]，用于 GigE Vision 相机
- FileSourceControl [▶ 2639]，用于文件源

通用

该组中的功能块代表图像提供程序 TcCOM 对象的代理。因此，它们不需要循环调用。相反，TcCOM 对象通过功能块进行控制。

状态机

所有图像采集功能块均包括一个状态机，用于管理所代表设备的状态。基本上，功能块必须置于 ACQUIRING（采集中）状态，才能连续接收图像。INITIAL（初始）状态代表重启或复位后的起始点，可触发 INITIALIZING（初始化）。如为 OPENED（已打开）状态，与设备的通信通道会被打开，以读取或写入寄存器值。



所有状态都由枚举 ETcVnCameraState [▶ 1650] 表示，并在下文中加以说明：

主要状态

主要的状态是静止状态。它们基本上表明是否有与 Vision 设备的连接（`OPENED`），该设备是否正在捕捉图像（`ACQUIRING`）以及与该设备的连接是否有错误（`ERROR`）。你可以通过 FB 的方法在这些状态之间切换。一般来说，两个主要状态之间的转换不会立即发生，而是至少经过一个周期。在这个中间时间，视觉设备处于中间状态 [▶ 2638]。

状态	描述
ERROR	错误状态，在这个状态下，每个设备只能通过调用方法`Reset`重新激活。
INITIAL	每个设备的初始状态。
INITIALIZED	与 INITIAL 类似，手动初始化相机（例如使用 Force-IP 和初始化命令）
OPENED	存在与相机的连接，可以触发单个图像，且可以读写相机寄存器。 **这个状态可以通过 StartAcquisition 方法跳过。**
ACQUIRING	设备处于采集状态，且正在发送图像，具体取决于设置。

中间状态

中间状态十分必要，因为大多数行动不能立即完成（如图像采集）。如果功能块处于这些中间状态之一，必须再次调用相应的方法，以完成向相应的主状态的过渡。

状态	描述
INITIALIZING	INITIAL > INITIALIZED
OPENING	INITIAL > OPENED 且 INITIALIZED > OPENED
STARTACQUISITION	OPENED > ACQUIRING.
STOPACQUISITION	ACQUIRING > OPENED
RESETTINGFEATURES	OPENED > OPENED
TRIGGERING	OPENED > OPENED 且 ACQUIRING > ACQUIRING
CLOSING	OPENED > INITIALIZED

必须在中间状态下调用方法！

在中间状态中也必须调用各自的状态转换方法。在相应的方法调用后使用`GetState`方法检测到静止状态时，状态转换才算完成，且不再需要调用该方法。触发一次方法，而随后的等待并不会导致所需的状态转换。

然后，C++ 中的状态机既可以用 `switch` 结构实现，也可以用 `if … else if …` 结构实现。

6.2.4.4.1 FileSourceControl

该 FB 可访问文件源实例发送的图像。

语法

构造函数：

```
FileSourceControl(
    OTCID oidITcVnFileImageProvider
)
```

方法

名称	描述
GetCurrentImage [▶ 2640]	获取当前可用的图像（如果有）。
GetCurrentImageAndFileName [▶ 2640]	获取当前可用的图像（如果有）和相应的文件名。
StartAcquisition [▶ 2640]	激活文件源。如果不在触发模式下，将开始发送图像。
StopAcquisition [▶ 2641]	停用文件源。将停止发送图像。
Reset [▶ 2641]	将控制器重置为初始状态（根据当前状态，可能需要多次调用，直至返回 S_OK）
GetState [▶ 2641]	获取内部相机控制状态机的当前状态。
TriggerImage [▶ 2641]	如果处于触发模式，该函数将触发列表中的下一个图像。
TriggerImageExp [▶ 2641]	如果处于触发模式，该函数将触发列表中的特定图像。
TriggerImageByName [▶ 2642]	如果处于触发模式，此函数会按名称触发列表中的特定图像。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.4.1.1 GetCurrentImage

获取当前可用的图像（如果有）。

语法

定义：

```
HRESULT GetCurrentImage(
    ITcVnImage*& ipImage
)
```

参数

名称	类型	描述
ipImage	ITcVnImage* [▶ 1863]&	返回图像

返回值

HRESULT [▶ 135]

6.2.4.4.1.2 GetCurrentImageAndFileName

获取当前可用的图像（如果有）和相应的文件名。

语法

定义：

```
HRESULT GetCurrentImageAndFileName(
    ITcVnImage*& ipImage,
    PCHAR        sFileName,
    USHORT       nMaxLen
)
```

参数

名称	类型	描述
ipImage	ITcVnImage* [▶ 1863]&	返回图像
sFileName	PCHAR	返回文件名
nMaxLen	USHORT	定义文件名的最大允许长度

返回值

HRESULT [▶ 135]

6.2.4.4.1.3 StartAcquisition

激活文件源。如果不在触发模式下，将开始发送图像。

语法

定义：

```
HRESULT StartAcquisition()
```

返回值

HRESULT [▶ 135]

6.2.4.4.1.4 StopAcquisition

停用文件源。将停止发送图像。

语法

定义：

```
HRESULT StopAcquisition()
```

返回值

HRESULT [▶ 135]

6.2.4.4.1.5 重置

将控制器重置为初始状态（根据当前状态，可能需要多次调用，直至返回 S_OK）

语法

定义：

```
HRESULT Reset()
```

返回值

HRESULT [▶ 135]

6.2.4.4.1.6 GetState

获取内部相机控制状态机的当前状态。

语法

定义：

```
ETcVnCameraState GetState()
```

返回值

ETcVnCameraState [▶ 1650]

6.2.4.4.1.7 TriggerImage

如果处于触发模式，该函数将触发列表中的下一个图像。

语法

定义：

```
HRESULT TriggerImage()
```

返回值

HRESULT [▶ 135]

6.2.4.4.1.8 TriggerImageExp

如果处于触发模式，该函数将触发列表中的特定图像。

语法

定义：

```
HRESULT TriggerImageExp(
    LONG nSkipImages
)
```

参数

名称	类型	描述
nSkipImages	LONG	要跳过的图像的数量（相对于当前图像数量）

返回值

HRESULT [▶ 135]

6.2.4.4.1.9 TriggerImageByName

如果处于触发模式，此函数会按名称触发列表中的特定图像。

语法

定义：

```
HRESULT TriggerImageByName(
    PCCH sImageName
)
```

参数

名称	类型	描述
sImageName	PCCH	要触发的图像名称

返回值

HRESULT [▶ 135]

6.2.4.4.2 GevCameraControl

该 FB 提供控制 GigE Vision 相机和访问其标定数据的基本功能。

语法

构造函数：

```
GevCameraControl(
    OTCID oidITcVnGevImageProvider
)
```

方法

名称	描述
ClearImageQueue [▶ 2644]	删除 TcVnGevImageProvider TcCOM 模块接收队列中的所有图像，并重置相应的被遗漏的图像计数。
CloseCamera [▶ 2644]	关闭相机的控制通道。
GetCalibPatternRef [▶ 2644]	从 TcVnGevImageProvider TcCOM 模块获取标定板参考点（可通过标定助手设置）

名称	描述
GetCameraMatrix [▶ 2645]	从 TcVnGevImageProvider TcCOM 模块获取相机矩阵（需要固有标定参数，例如由标定助手设置的参数）
GetCurrentImage [▶ 2645]	获取当前可用的图像（接收队列中的第一个）。
GetCurrentImageUndistorted [▶ 2645]	获取当前可用的去畸变图像（接收队列中的第一个）。与包含编码（例如，贝尔（Bayer）图）的像素格式不兼容。需要 TcVnGevImageProvider TcCOM 模块中的固有标定参数，例如由标定助手设置的参数。
GetCurrentImageWithGvspInfo [▶ 2646]	获取当前可用的图像（接收队列中的第一个）及其相应的 GVSP 信息。
GetCurrentImageWithGvspInfoUndistorted [▶ 2646]	获取当前可用的去畸变图像（接收队列中的第一个）及其相应的 GVSP 信息。与包含编码（例如，贝尔（Bayer）图）的像素格式不兼容。需要 TcVnGevImageProvider TcCOM 模块中的固有标定参数，例如由标定助手设置的参数。
GetCurrentImageWithTimestamps [▶ 2646]	获取当前可用的图像（接收队列中的第一个）和相应的时间戳。
GetDistortionCoefficients [▶ 2647]	从 TcVnGevImageProvider TcCOM 模块获取几何畸变系数（需要固有标定参数，例如由标定助手设置的参数）
GetImageQueueSize [▶ 2652]	获取接收队列中当前的图像数量。
GetLastImageFromQueue [▶ 2647]	获取队列中最后收到的图像。
GetOmittedImagesNum [▶ 2648]	获取自上次调用 ClearImageQueue() 起遗漏的图片数量。如果 TcVnGevImageProvider TcCOM 模块中的图像接收队列已满，而又有新图像到达，队列中的第一个图像将被删除，遗漏的计数也会增加。
GetRotationMatrix [▶ 2648]	从 TcVnGevImageProvider TcCOM 模块获取旋转矩阵（需要外部标定参数，例如由标定助手设置的参数）
GetState [▶ 2648]	获取内部相机控制状态机的当前状态。
GetTranslationVector [▶ 2649]	从 TcVnGevImageProvider TcCOM 模块获取平移向量（需要外部标定参数，例如由标定助手设置的参数）
InitializeCamera [▶ 2649]	将相机初始化到预期状态（包括向相机发送 GevImageAcquisition TcCOM 模块中定义的“InitCameraCommands”）。
OpenCamera [▶ 2649]	打开相机的控制通道。
Reset [▶ 2649]	将相机控制器重置为初始状态（根据当前状态可能需要多次调用，直至返回 S_OK）
ResetCameraFeatures [▶ 2650]	将相机特征重置为初始状态（向相机发送 GevImageAcquisition TcCOM 模块中定义的“InitCameraCommands”）。
SetCameraMatrix [▶ 2650]	将相机矩阵设置为 TcVnGevImageProvider TcCOM 模块
SetDistortionCoefficients [▶ 2650]	为 TcVnGevImageProvider TcCOM 模块设置畸变系数
SetRotationMatrix [▶ 2651]	为 TcVnGevImageProvider TcCOM 模块设置旋转矩阵
SetTranslationVector [▶ 2651]	为 TcVnGevImageProvider TcCOM 模块设置平移向量
StartAcquisition [▶ 2651]	向相机发送 TcVnGevImageProvider TcCOM 模块中定义的“StartAcquisitionCommands”。
StopAcquisition [▶ 2651]	向相机发送 TcVnGevImageProvider TcCOM 模块中定义的“StopAcquisitionCommands”。
TriggerImage [▶ 2652]	向相机发送 TcVnGevImageProvider TcCOM 模块中定义的“SoftwareTriggerCommands”。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.4.2.1 ClearImageQueue

删除 TcVnGevImageProvider TcCOM 模块接收队列中的所有图像，并重置相应的被遗漏的图像计数。

语法

定义：

```
HRESULT ClearImageQueue()
```

返回值

HRESULT [▶ 135]

6.2.4.4.2.2 CloseCamera

关闭相机的控制通道。

语法

定义：

```
HRESULT CloseCamera()
```

返回值

HRESULT [▶ 135]

6.2.4.4.2.3 GetCalibPatternRef

从 TcVnGevImageProvider TcCOM 模块获取标定板参考点（可通过标定助手设置）

语法

定义：

```
HRESULT GetCalibPatternRef(
    ITcVnContainer*& ipCalibPatternRef
)
```

参数

名称	类型	描述
ipCalibPatternRef	ITcVnContainer* [▶ 1827]&	返回标定板参考点（ContainerType_Vector_TcVnPoint3_REAL）

返回值

HRESULT [▶ 135]

6.2.4.4.2.4 GetCameraMatrix

从 TcVnGevImageProvider TcCOM 模块获取相机矩阵（需要固有标定参数，例如由标定助手设置的参数）

语法

定义：

```
HRESULT GetCameraMatrix(
    TcVnMatrix3x3_LREAL& aCameraMatrix
)
```

参数

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	返回相机矩阵

返回值

HRESULT [▶ 135]

6.2.4.4.2.5 GetCurrentImage

获取当前可用的图像（接收队列中的第一个）。

语法

定义：

```
HRESULT GetCurrentImage(
    ITcVnImage*& ipImage
)
```

参数

名称	类型	描述
ipImage	ITcVnImage* [▶ 1863]&	返回图像

返回值

HRESULT [▶ 135]

6.2.4.4.2.6 GetCurrentImageUndistorted

获取当前可用的去畸变图像（接收队列中的第一个）。与包含编码（例如，贝尔（Bayer）图）的像素格式不兼容。需要 TcVnGevImageProvider TcCOM 模块中的固有标定参数，例如由标定助手设置的参数。

语法

定义：

```
HRESULT GetCurrentImageUndistorted(
    ITcVnImage*& ipImage
)
```

参数

名称	类型	描述
ipImage	ITcVnImage* [▶ 1863]&	返回去畸变图像

返回值

HRESULT [▶ 135]

6.2.4.4.2.7 GetCurrentImageWithGvspInfo

获取当前可用的图像（接收队列中的第一个）及其相应的 GVSP 信息。

语法

定义：

```
HRESULT GetCurrentImageWithGvspInfo(
    ITcVnImage*&     ipImage,
    GVSP_IMAGE_INFO& stGvspInfo
)
```

参数

名称	类型	描述
ipImage	ITcVnImage* [▶ 1863]&	返回图像
stGvspInfo	GVSP_IMAGE_INFO [▶ 1700]&	返回 GVSP 信息

返回值

HRESULT [▶ 135]

6.2.4.4.2.8 GetCurrentImageWithGvspInfoUndistorted

获取当前可用的去畸变图像（接收队列中的第一个）及其相应的 GVSP 信息。与包含编码（例如，贝尔（Bayer）图）的像素格式不兼容。需要 TcVnGevImageProvider TcCOM 模块中的固有标定参数，例如由标定助手设置的参数。

语法

定义：

```
HRESULT GetCurrentImageWithGvspInfoUndistorted(
    ITcVnImage*&     ipImage,
    GVSP_IMAGE_INFO& stGvspInfo
)
```

参数

名称	类型	描述
ipImage	ITcVnImage* [▶ 1863]&	返回去畸变图像
stGvspInfo	GVSP_IMAGE_INFO [▶ 1700]&	返回 GVSP 信息

返回值

HRESULT [▶ 135]

6.2.4.4.2.9 GetCurrentImageWithTimestamps

获取当前可用的图像（接收队列中的第一个）和相应的时间戳。

语法

定义：

```
HRESULT GetCurrentImageWithTimestamps(
    ITcVnImage*&    ipImage,
    ULONGLONG       nTimestamps,
    LONGLONG*       pTimestamps,
    GVSP_IMAGE_INFO& stGvspInfo
)
```

参数

名称	类型	描述
ipImage	ITcVnImage* [▶ 1863]&	返回图像
nTimestamps	ULONGLONG	选择要返回的时间戳（使用按位 OR 组合多个 ETcVnTimestamp 值）
pTimestamps	LONGLONG*	返回请求的时间戳（确保提供足够大小的数组）。
stGvspInfo	GVSP_IMAGE_INFO [▶ 1700]&	返回 GVSP 信息

返回值

HRESULT [▶ 135]

6.2.4.4.2.10 GetDistortionCoefficients

从 TcVnGevImageProvider TcCOM 模块获取畸变系数（需要固有标定参数，例如由标定助手设置的参数）

语法

定义：

```
HRESULT GetDistortionCoefficients(
    TcVnArray8_LREAL& aDistortionCoefficients
)
```

参数

名称	类型	描述
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&	返回畸变系数 [k1、k2、p1、p2、k3、k4、k5、k6]

返回值

HRESULT [▶ 135]

6.2.4.4.2.11 GetLastImageFromQueue

获取队列中最后收到的图像。

语法

定义：

```
HRESULT GetLastImageFromQueue(
    ITcVnImage*& ipImage
)
```

参数

名称	类型	描述
ipImage	ITcVnImage* [▶ 1863]&	返回图像

返回值

HRESULT [▶ 135]

6.2.4.4.2.12 GetOmittedImagesNum

获取自上次调用 ClearImageQueue() 起遗漏的图片数量。如果 TcVnGevImageProvider TcCOM 模块中的图像接收队列已满，而又有新图像到达，队列中的第一个图像将被删除，遗漏的计数也会增加。

语法

定义：

```
HRESULT GetOmittedImagesNum(
    ULONGLONG& nOmitted
)
```

参数

名称	类型	描述
nOmitted	ULONGLONG&	返回遗漏图像的数量

返回值

HRESULT [▶ 135]

6.2.4.4.2.13 GetRotationMatrix

从 TcVnGevImageProvider TcCOM 模块获取旋转矩阵（需要外部标定参数，例如由标定助手设置的参数）

语法

定义：

```
HRESULT GetRotationMatrix(
    TcVnMatrix3x3_LREAL& aRotationMatrix
)
```

参数

名称	类型	描述
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	返回旋转矩阵

返回值

HRESULT [▶ 135]

6.2.4.4.2.14 GetState

获取内部相机控制状态机的当前状态。

语法

定义：

```
ETcVnCameraState GetState()
```

返回值

ETcVnCameraState [▶ 1650]

6.2.4.4.2.15 GetTranslationVector

从 TcVnGevImageProvider TcCOM 模块获取平移向量（需要外部标定参数，例如由标定助手设置的参数）

语法

定义：

```
HRESULT GetTranslationVector(
    TcVnVector3_LREAL& aTranslationVector
)
```

参数

名称	类型	描述
aTranslationVector	TcVnVector3_LREAL [▶ 1641]&	返回平移向量

返回值

HRESULT [▶ 135]

6.2.4.4.2.16 InitializeCamera

将相机初始化到预期状态（包括向相机发送 GevImageAcquisition TcCOM 模块中定义的“InitCameraCommands”）。

语法

定义：

```
HRESULT InitializeCamera()
```

返回值

HRESULT [▶ 135]

6.2.4.4.2.17 OpenCamera

打开相机的控制通道。

语法

定义：

```
HRESULT OpenCamera()
```

返回值

HRESULT [▶ 135]

6.2.4.4.2.18 重置

将相机控制器重置为初始状态（根据当前状态可能需要多次调用，直至返回 S_OK）

语法

定义：

```
HRESULT Reset()
```

返回值

HRESULT [▶ 135]

6.2.4.4.2.19 ResetCameraFeatures

将相机特征重置为初始状态（向相机发送 GevImageAcquisition TcCOM 模块中定义的“InitCameraCommands”）。

语法

定义：

```
HRESULT ResetCameraFeatures()
```

返回值

HRESULT [▶ 135]

6.2.4.4.2.20 SetCameraMatrix

将相机矩阵设置为 TcVnGevImageProvider TcCOM 模块

语法

定义：

```
HRESULT SetCameraMatrix(
    TcVnMatrix3x3_LREAL& aCameraMatrix
)
```

参数

名称	类型	描述
aCameraMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	要复制到 TcCOM 模块的相机矩阵

返回值

HRESULT [▶ 135]

6.2.4.4.2.21 SetDistortionCoefficients

为 TcVnGevImageProvider TcCOM 模块设置畸变系数

语法

定义：

```
HRESULT SetDistortionCoefficients(
    TcVnArray8_LREAL& aDistortionCoefficients
)
```

参数

名称	类型	描述
aDistortionCoefficients	TcVnArray8_LREAL [▶ 1641]&	要复制到 TcCOM 模块的畸变系数

返回值

HRESULT [▶ 135]

6.2.4.4.2.22 SetRotationMatrix

为 TcVnGevImageProvider TcCOM 模块设置旋转矩阵

语法

定义：

```
HRESULT SetRotationMatrix(
    TcVnMatrix3x3_LREAL& aRotationMatrix
)
```

参数

名称	类型	描述
aRotationMatrix	TcVnMatrix3x3_LREAL [▶ 1641]&	要复制到 TcCOM 模块的旋转矩阵

返回值

HRESULT [▶ 135]

6.2.4.4.2.23 SetTranslationVector

为 TcVnGevImageProvider TcCOM 模块设置平移向量

语法

定义：

```
HRESULT SetTranslationVector(
    TcVnVector3_LREAL& aTranslationVector
)
```

参数

名称	类型	描述
aTranslationVector	TcVnVector3_LREAL [▶ 1641]&	要复制到 TcCOM 模块的平移向量

返回值

HRESULT [▶ 135]

6.2.4.4.2.24 StartAcquisition

向相机发送 TcVnGevImageProvider TcCOM 模块中定义的“StartAcquisitionCommands”。

语法

定义：

```
HRESULT StartAcquisition()
```

返回值

HRESULT [▶ 135]

6.2.4.4.2.25 StopAcquisition

向相机发送 TcVnGevImageProvider TcCOM 模块中定义的“StopAcquisitionCommands”。

语法

定义：

```
HRESULT StopAcquisition()
```

返回值

HRESULT [▶ 135]

6.2.4.4.2.26 TriggerImage

向相机发送 TcVnGevImageProvider TcCOM 模块中定义的“SoftwareTriggerCommands”。

语法

定义：

```
HRESULT TriggerImage()
```

返回值

HRESULT [▶ 135]

6.2.4.4.2.27 GetImageQueueSize

获取接收队列中当前的图像数量。

语法

定义：

```
HRESULT GetImageQueueSize(
    ULONGLONG& nSize
)
```

参数

名称	类型	描述
nSize	ULONGLONG&	返回接收队列中的图像数量

返回值

HRESULT [▶ 135]

6.2.4.4.3 SimpleCameraControl

该 FB 提供控制相机或文件源的基本功能。

语法

构造函数：

```
SimpleCameraControl(
    OTCID oidITcVnImageProvider
)
```

方法

名称	描述
GetCurrentImage [▶ 2653]	获取当前可用的图像（如果有）。

名称	描述
GetState [▶ 2653]	获取内部相机控制状态机的当前状态。
Reset [▶ 2653]	将相机重置为初始状态（根据当前状态可能需要多次调用，直至返回 S_OK）
StartAcquisition [▶ 2654]	开始图像采集。
StopAcquisition [▶ 2654]	停止图像采集。
TriggerImage [▶ 2654]	触发下一个图像。

所需的许可证

TC3 Vision Base

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.4.3.1 GetCurrentImage

获取当前可用的图像（如果有）。

语法

定义：

```
HRESULT GetCurrentImage(
    ITcVnImage*& ipImage
)
```

参数

名称	类型	描述
ipImage	ITcVnImage* [▶ 1863]&	返回图像

返回值

HRESULT [▶ 135]

6.2.4.4.3.2 GetState

获取内部相机控制状态机的当前状态。

语法

定义：

```
ETcVnCameraState GetState()
```

返回值

ETcVnCameraState [▶ 1650]

6.2.4.4.3.3 重置

将相机重置为初始状态（根据当前状态可能需要多次调用，直至返回 S_OK）

语法

定义：

```
HRESULT Reset()
```

返回值

HRESULT [▶ 135]

6.2.4.4.3.4 StartAcquisition

开始图像采集。

语法

定义：

```
HRESULT StartAcquisition()
```

返回值

HRESULT [▶ 135]

6.2.4.4.3.5 StopAcquisition

停止图像采集。

语法

定义：

```
HRESULT StopAcquisition()
```

返回值

HRESULT [▶ 135]

6.2.4.4.3.6 TriggerImage

触发下一个图像。

语法

定义：

```
HRESULT TriggerImage()
```

返回值

HRESULT [▶ 135]

6.2.4.5 图像处理

该组包含用于图像处理的功能块。在这些综合功能块中，设定值在执行后会被保留。此外，功能块还可以包含多个方法。

函数 [▶ 1882]章节中包含的函数则没有内部状态信息，也不保留变量值。

功能块

- GeneralizedHoughBallard [▶ 2655]
- SSIM [▶ 2658]

6.2.4.5.1 GeneralizedHoughBallard

该 FB 提供广义霍夫变换功能。首先，通过相应的方法设置参数（可选，否则使用默认值）。然后，设置模板（必填）。随后即可执行检测方法。

语法

构造函数：

```
GeneralizedHoughBallard()
```

方法

名称	描述
Detect [▶ 2655]	检测图像中的模板（之前必须调用过 SetTemplate）。
SetCannyThreshold [▶ 2656]	设置 Canny 边缘检测阈值。
SetInvAccuRatio [▶ 2656]	设置累加器大小与源图像大小的反比（例如，值为 2 表示大小双向减半。）
SetLevels [▶ 2656]	设置 R 表的级数。
SetMinDist [▶ 2657]	设置不同对象中心之间的最小距离。
SetTemplate [▶ 2657]	设置检测方法中要搜索的模板图像。
SetVotesThreshold [▶ 2658]	设置累加器阈值，即检测匹配所需的最终评估数（数值太小会导致误检测）。

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.5.1.1 检测

检测图像中的模板（之前必须调用过 SetTemplate）。

语法

定义：

```
HRESULT Detect(
    ITcVnImage*       ipImage,
    ITcVnContainer*& ipPositions
)
```

参数

名称	类型	描述
ipImage	ITcVnImage* [▶ 1863]	源图像（1 通道，USINT）

名称	类型	描述
ipPositions	ITcVnContainer* [▶ 1827]&	返回检测到的模板位置的中心（ContainerType_Vector_TcVnPoint2_REAL；重复使用非零接口指针。）

返回值

HRESULT [▶ 135]

6.2.4.5.1.2 SetCannyThreshold

设置 Canny 边缘检测阈值。

语法

定义：

```
HRESULT SetCannyThreshold(
    LONG nLow = 50,
    LONG nHigh = 100
)
```

参数

名称	类型	描述
nLow	LONG	低阈值（> 0）
nHigh	LONG	高阈值（> nLow，通常为 2 至 3 * nLow）

返回值

HRESULT [▶ 135]

6.2.4.5.1.3 SetInvAccuRatio

设置累加器大小与源图像大小的反比（例如，值为 2 表示大小双向减半。）

语法

定义：

```
HRESULT SetInvAccuRatio(
    double fInvAccuRatio = 1.0
)
```

参数

名称	类型	描述
fInvAccuRatio	double	反向累加器比率（> 0）

返回值

HRESULT [▶ 135]

6.2.4.5.1.4 SetLevels

设置 R 表的级数。

语法

定义：

```
HRESULT SetLevels(
    LONG nLevels = 360
)
```

参数

名称	类型	描述
nLevels	LONG	R 表级别数（> 0）

返回值

HRESULT [▶ 135]

6.2.4.5.1.5 SetMinDist

设置不同对象中心之间的最小距离。

语法

定义：

```
HRESULT SetMinDist(
    double fMinDist = 1.0
)
```

参数

名称	类型	描述
fMinDist	double	最小距离（> 0）

返回值

HRESULT [▶ 135]

6.2.4.5.1.6 SetTemplate

设置检测方法中要搜索的模板图像。

语法

定义：

```
HRESULT SetTemplate(
    ITcVnImage* ipImage
)
```

参数

名称	类型	描述
ipImage	ITcVnImage* [▶ 1863]	模板图像（1 通道，USINT）

返回值

HRESULT [▶ 135]

6.2.4.5.1.7 SetVotesThreshold

设置累加器阈值，即检测匹配所需的最终评估数（数值太小会导致误检测）。

语法

定义：

```
HRESULT SetVotesThreshold(
    LONG nVotes = 100
)
```

参数

名称	类型	描述
nVotes	LONG	检测匹配所需的最终评估数（> 0）

返回值

HRESULT [▶ 135]

6.2.4.5.2 SSIM

该 FB 提供结构相似性（SSIM）功能。首先，设置参考图像。随后即可执行计算方法。

语法

构造函数：

```
SSIM()
```

方法

名称	描述
Compute [▶ 2658]	计算结构相似性（之前必须调用过 SetReference）。
SetReference [▶ 2659]	设置 SSIM 方法使用的参考图像。

所需的许可证

TC3 Vision Matching

系统要求

开发环境	目标平台	要包括的 PLC 库
TwinCAT V3.1.4024.59 或更高版本	PC 或 CX（x64），最低配置为 PL50，例如 Intel Atom® 四核 CPU	Tc3_Vision

6.2.4.5.2.1 计算

计算结构相似性（之前必须调用过 SetReference）。

语法

定义：

```
HRESULT Compute(
    ITcVnImage*  ipSrcImage,
    ITcVnImage*& ipSSIM
)
```

参数

名称	类型	描述
ipSrcImage	ITcVnImage* [▶ 1863]	源图像（USINT）
ipSSIM	ITcVnImage* [▶ 1863]&	返回 SSIM（REAL，数值范围 0 至 1）

返回值

HRESULT [▶ 135]

6.2.4.5.2.2 SetReference

设置 SSIM 方法使用的参考图像。

语法

定义：

```
HRESULT SetReference(
    ITcVnImage* ipImage
)
```

参数

名称	类型	描述
ipImage	ITcVnImage* [▶ 1863]	参考图像（USINT）

返回值

HRESULT [▶ 135]

7 TwinCAT HMI 软件包

本章会介绍 TwinCAT Vision 与 TwinCAT HMI 的交互。其中包含一个服务器扩展包 [▶ 2660]，通过它可以在 TwinCAT HMI 中显示来自 PLC 的图像 [▶ 143]，以及一个用于图像和颜色的可视化和交互的控制包 [▶ 2665]。

集成软件包

在 TwinCAT HMI 1.12 以上版本中，软件包的集成是通过软件包管理 NuGet 进行的。在 TwinCAT HMI 项目中，右键单击 **References（引用）**并选择 **Manage NuGet Packages...（管理 NuGet 软件包.....）**。然后点击 **Browse（浏览）**，选择“TwinCAT HMI Official”或“Nuget.org”作为 **Package source（软件包源）**。现在您将看到可用的 TwinCAT HMI 软件包列表，您可以从中选择 **Beckhoff.TwinCAT.HMI.Vision** 或 **Beckhoff.TwinCAT.HMI.Vision.Controls** 等软件包。

从 TwinCAT HMI 1.14 版开始，可通过 TwinCAT Package Manager（TwinCAT 软件包管理器）安装 **TwinCAT.HMI.Vision** 和 **TwinCAT.HMI.Vision.Controls** 软件包。然后必须如上所述，通过“Beckhoff Offline Packages（倍福离线软件包）”**软件包源**将软件包集成到 HMI 项目中。

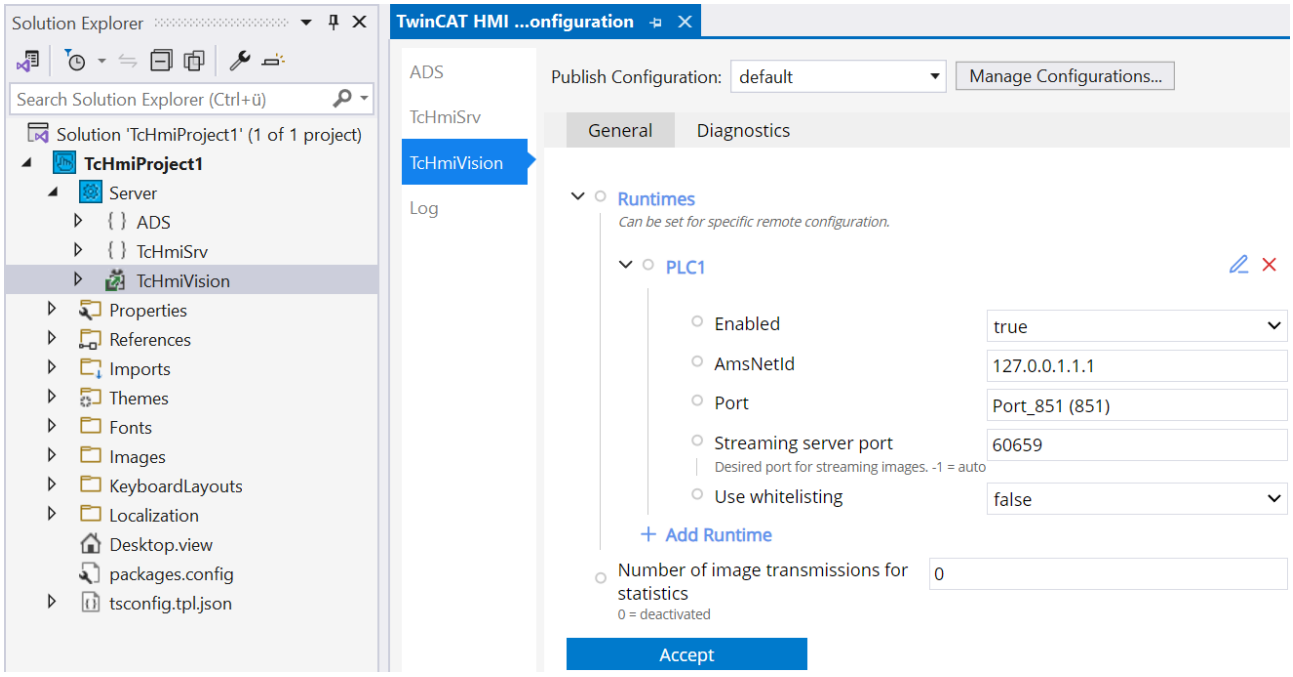
可在 TwinCAT HMI 文档中找到一般信息，例如软件包管理的详细说明。

7.1 服务器扩展

NuGet 软件包 Beckhoff.TwinCAT.HMI.Vision（TwinCAT 软件包“TwinCAT.HMI.Vision”）包含 `TcHmiVision` HMI 服务器扩展包，该扩展包可用于从 PLC 中检索图像数据并将其显示在 TwinCAT HMI 中。为此，来自配置运行时的所有 ITcVnDisplayableImage [▶ 407] 类型的 ADS 符号均会列出，并可以各种格式与 HMI 符号进行链接。有关所需的 .NET 版本，请参见 系统要求 [▶ 14]。

配置

软件包成功集成到 HMI 项目后，请在 **Server > TcHmiVision > General（服务器 > TcHmiVision > 常规）**下输入输入目标 PLC 的 **AmsNetId** 和**端口**：



此外，还可通过 **Enabled（启用）**设置启用或禁用单个目标系统。

流媒体服务器端口会确定用于发送配置为`Stream`作为传输格式的图像的流媒体端口。

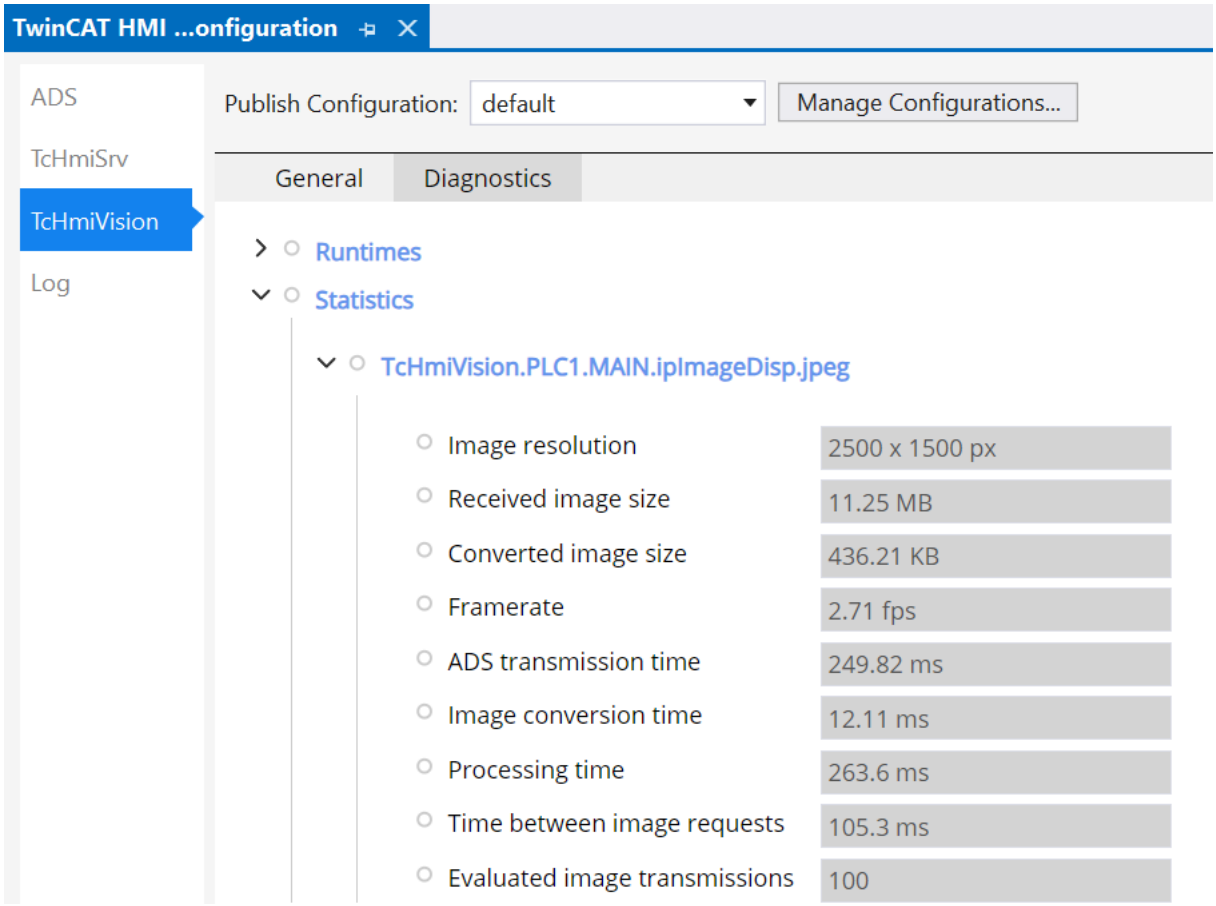
如果扩展包不列出来自 PLC 的单个符号，则可使用 `TcHmiSymbol.Hide` PLC 属性将其隐藏。另一方面，如果只列出某些符号，则必须启用 **Use Whitelisting（使用白名单）**设置。然后便可以使用 `TcHmiSymbol.Show` PLC 属性标记相应的变量。

```
VAR
    { attribute 'TcHmiSymbol.Hide' } // or: TcHmiSymbol.Show
    ipDisplayableImage: ITcVnDisplayableImage;
END_VAR
```

统计信息

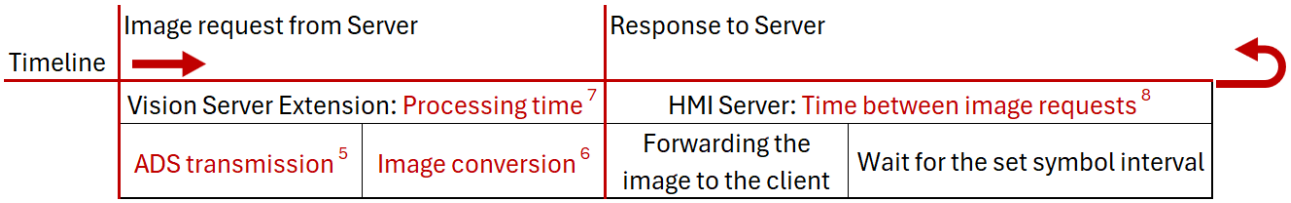
在 **Server（服务器） > TcHmiVision > General（常规）** 下，将 `Number of image transmissions for statistics` 设置为大于 0 的值，以启用统计数据收集。例如，如果将该值设为 100，则最后的 100 次图像传输会包含在统计信息中。要在 Engineering 中重置统计数据，请先输入 0，然后重新输入之前使用的数字。

统计数据列于 **Server（服务器） > TcHmiVision > Diagnostics（诊断） > Statistics（统计信息）** 下。为每个配置的图像符号计算单独的值。

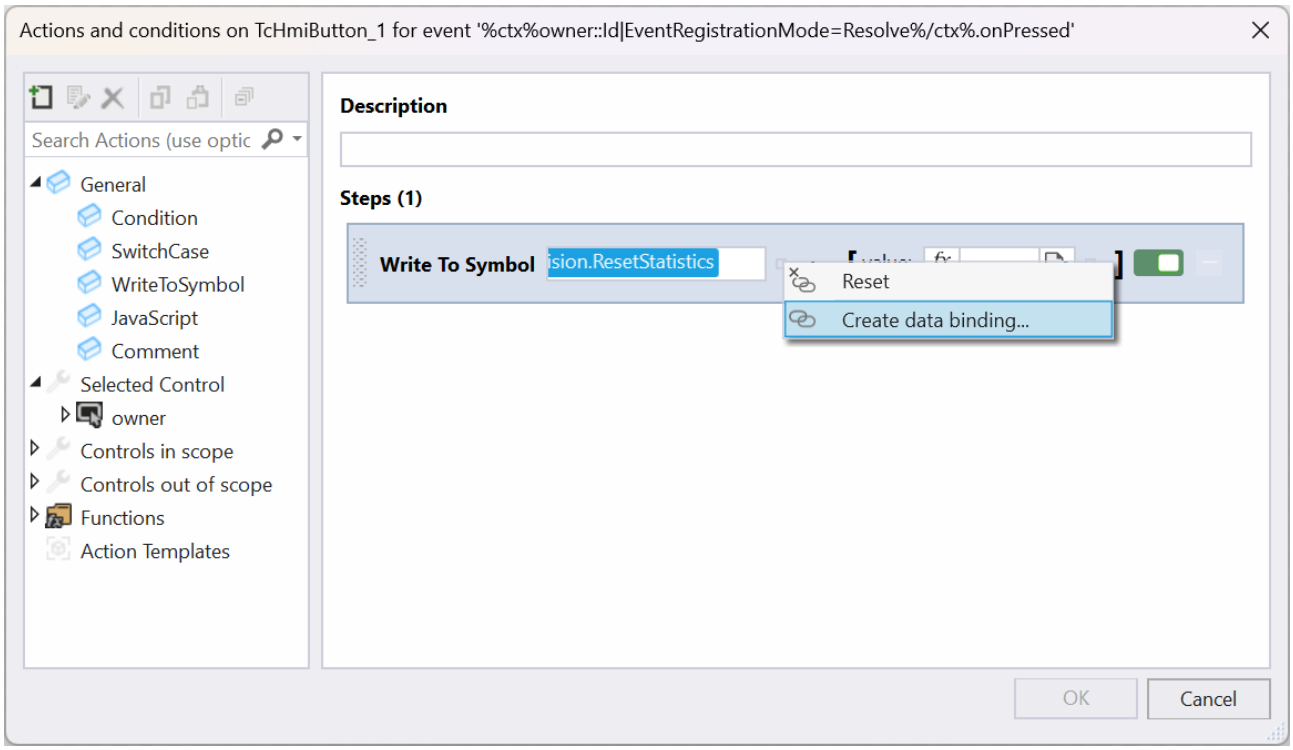


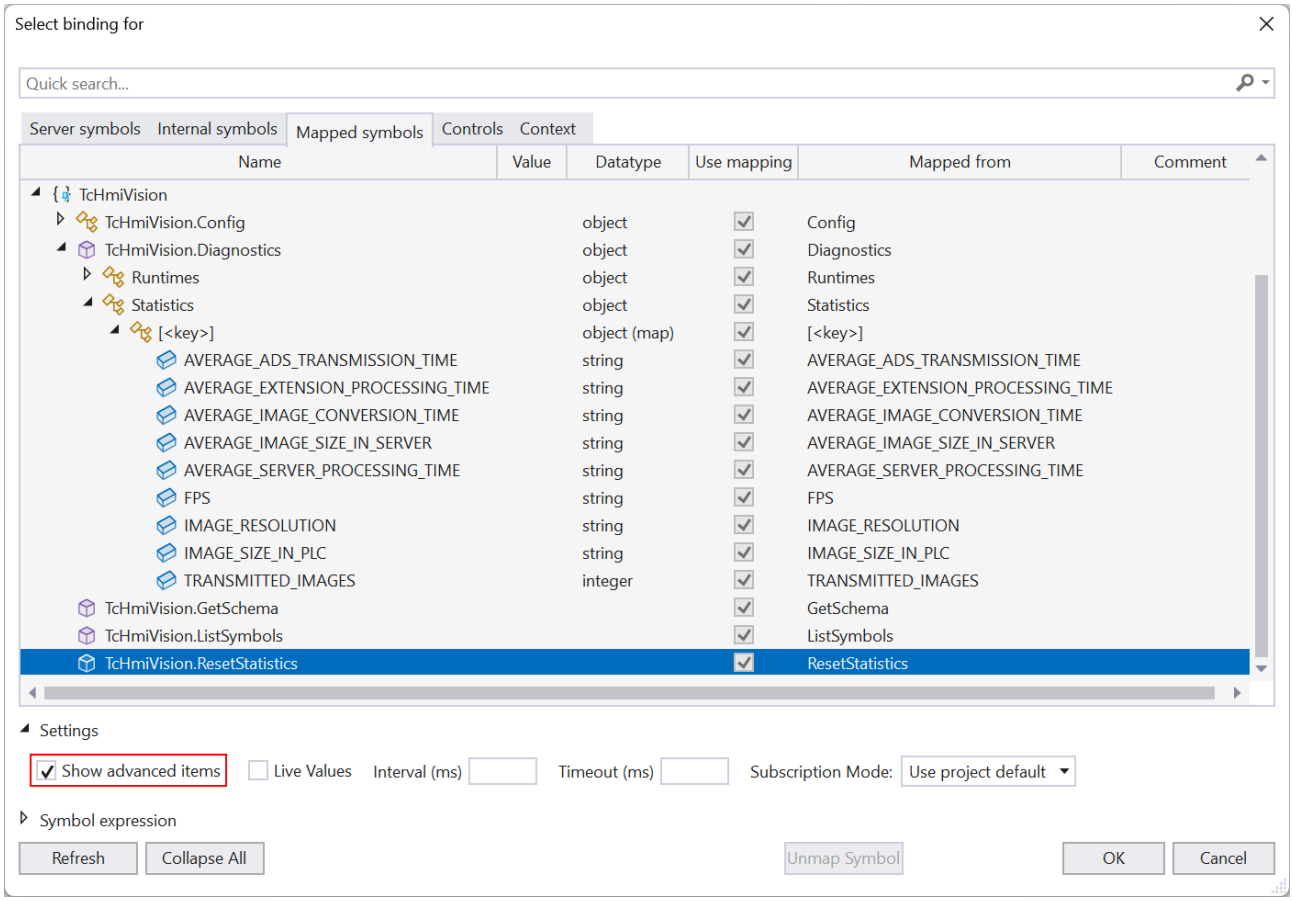
1. 图像分辨率：图像分辨率（像素）
2. 接收图像大小：接收图像的图像大小
3. 转换后的图像大小：转换后的图像大小，例如根据配置的图像格式转换成 Jpeg 或 Png 格式，然后再转换成 Base64 字符串
4. 帧频：每秒图像传输速率
5. ADS 传输时间：通过 ADS 传输图像的时长
6. 图像转换时间：图像转换时长
7. 处理时间：扩展包传输图像（包括转换）所需的时间。
8. 图像请求间隔时间：扩展包发出响应与服务器发出下一个图像请求之间的时间间隔。在这一阶段，服务器等待设定的符号间隔。如果图像传输暂停，该值可能会失真。为防止数值被伪造，可按以下说明重置统计信息。或者，您也可以等到已传输更多图像并调整统计信息之后再进行操作。此外，如果同一符号存在多个数据绑定，该值可能会有所不同。
9. 经评估的图像传输：目前纳入统计信息的图像传输数量

图像传输序列的图示，统计值用红色标记：



要从应用程序重置统计数据，必须向 `TcHmiVision.ResetStatistics` 符号写入一个空值。例如，将 **Write To（写入）符号**拖动到事件的 Actions（操作）和 Conditions（条件）中。在左侧字段中，点击 **Create data binding...（创建数据绑定.....）** 要列出符号，必须在窗口设置中启用 **Show advanced items（显示高级项目）**。现在将相应的符号关联到 **Mapped symbols（映射符号） > TcHmiVision > ResetStatistics** 下。

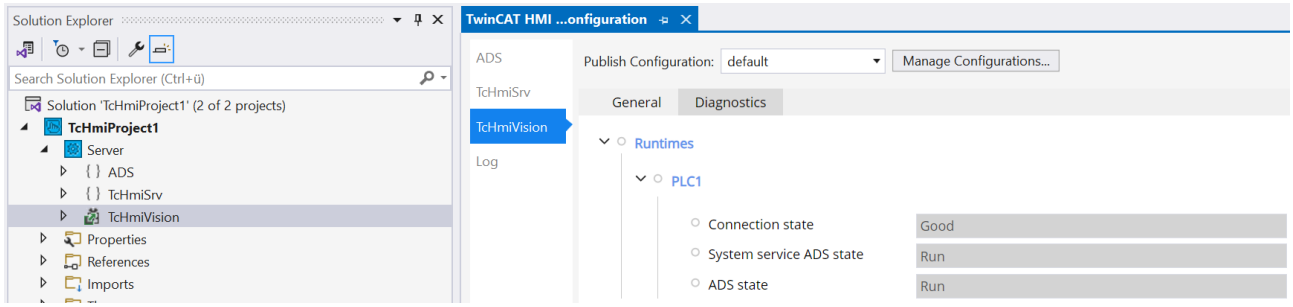




要在应用程序中显示统计数据，各个值可以通过符号进行映射，这些符号同样列在 Show advanced items（显示高级项目）中。

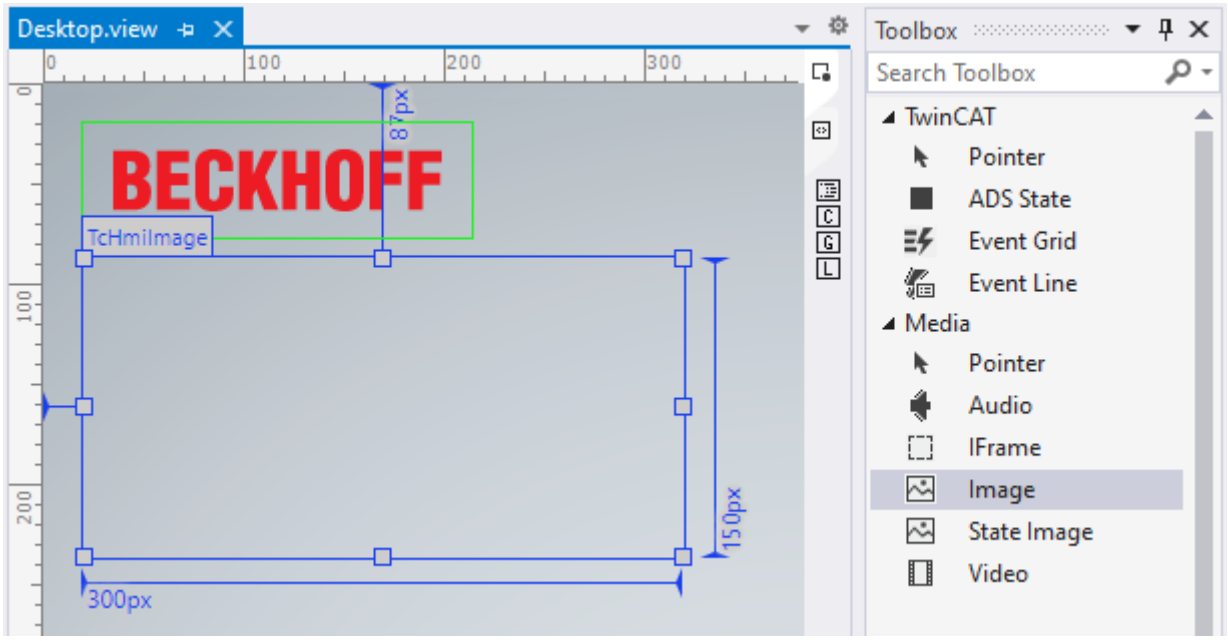
诊断

在 **Server（服务器） > TcHmiVision > Diagnostics（诊断）** 下，可以获取已配置目标系统的连接状态信息：

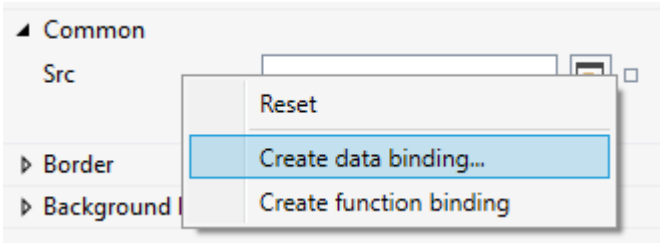


应用

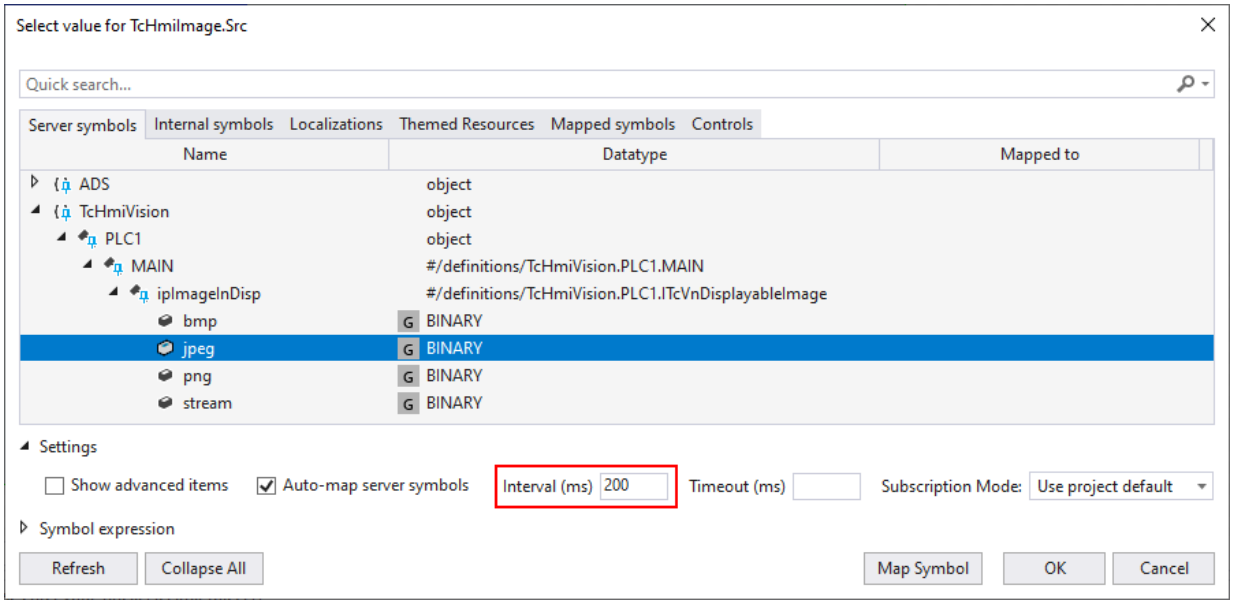
1. 导航至 HMI 视图并添加 **Image（图像）**控制器。



2. 打开控制元件的属性并点击参数 **Src...** 下的 **Create data binding...（创建数据绑定…）**



3. 在以下位置将图像控制器的 Src 参数链接到相应的图像符号：**Server symbols（服务器符号） > TcHmiVision > PLC > MAIN > ipImage > <image format>（<图像格式>）**



关联图像

只有当配置目标系统的过程图象包含相应图像变量时才能关联图像。

间隔时间

从 PLC 检索图像的速度取决于系统、图像大小和工作量。因此，我们建议在 **Settings（设置）**的 Binding（绑定）对话框中指定一个固定的间隔时间，以防止系统超载。

如果一个页面上要显示多个图像，也建议使用不同的时间间隔，如 200 ms 和 201 ms。这样，图像之间的检索分开进行，这对于数据量较大的情况有利。

传输格式

服务器扩展包含来自 PLC 的 BMP 格式图像数据，并将其作为 Base64 字符串转发至 TwinCAT HMI 客户端。为了减少传输到客户端的数据量，可以在服务器扩展中对图像进行压缩。提供下列图像格式：

- **BMP**：未压缩
- **JPEG**：有损压缩，适用于自然图像
- **PNG**：压缩（无损），适用于人工创建图像
- **流**：适用于连续图像流（例如实时相机图像，不推荐！）

要以数据流形式传输，必须以管理员身份启动 HMI 服务器。否则无法传输图像。

7.2 控制

NuGet 软件包 Beckhoff.TwinCAT.HMI.Vision.Controls（TwinCAT 软件包 “TwinCAT.HMI.Vision.Controls”）提供了一些用于显示和操作 Vision 数据的控制器。本文档假定 TwinCAT HMI 概念的基本知识参见 TE2000 并参考它们。在那里还可以找到关于使用控制器的详细说明。

控制器

该软件包包含以下控制器，可通过 HMI Toolbox（HMI 工具箱）添加到 HMI 页面：

控制	类别	类型	描述
图像查看 [▶ 2666]	Vision	`TcHmiVnImage`	• 显示图像 • 缩放和拖动选项 • 提取图像信息 • 用不同形状标记图像区域 • 多图像选择 • 叠加图形 • 具有操作功能的工具栏 • 用于显示图像和形状信息的信息栏
颜色 [▶ 2689]	Vision	`TcHmiVnColor`	显示和设置颜色
历史 [▶ 2700]	Vision	`TcHmiVnHistory`	Image Watch Control（图像查看控制器）的图像历史功能的显示与控制。
Histogram [▶ 2707]	Vision	`TcHmiVnHistogram`	根据图像生成并显示直方图，以分析颜色信息。
矩形（交互式） [▶ 2714]	Shapes（形状）	`TcHmiVnRectangle`	交互式显示和绘制矩形。
多边形（交互式） [▶ 2716]	Shapes（形状）	`TcHmiVnPolygon`	交互式显示和绘制多边形。

形状控制器

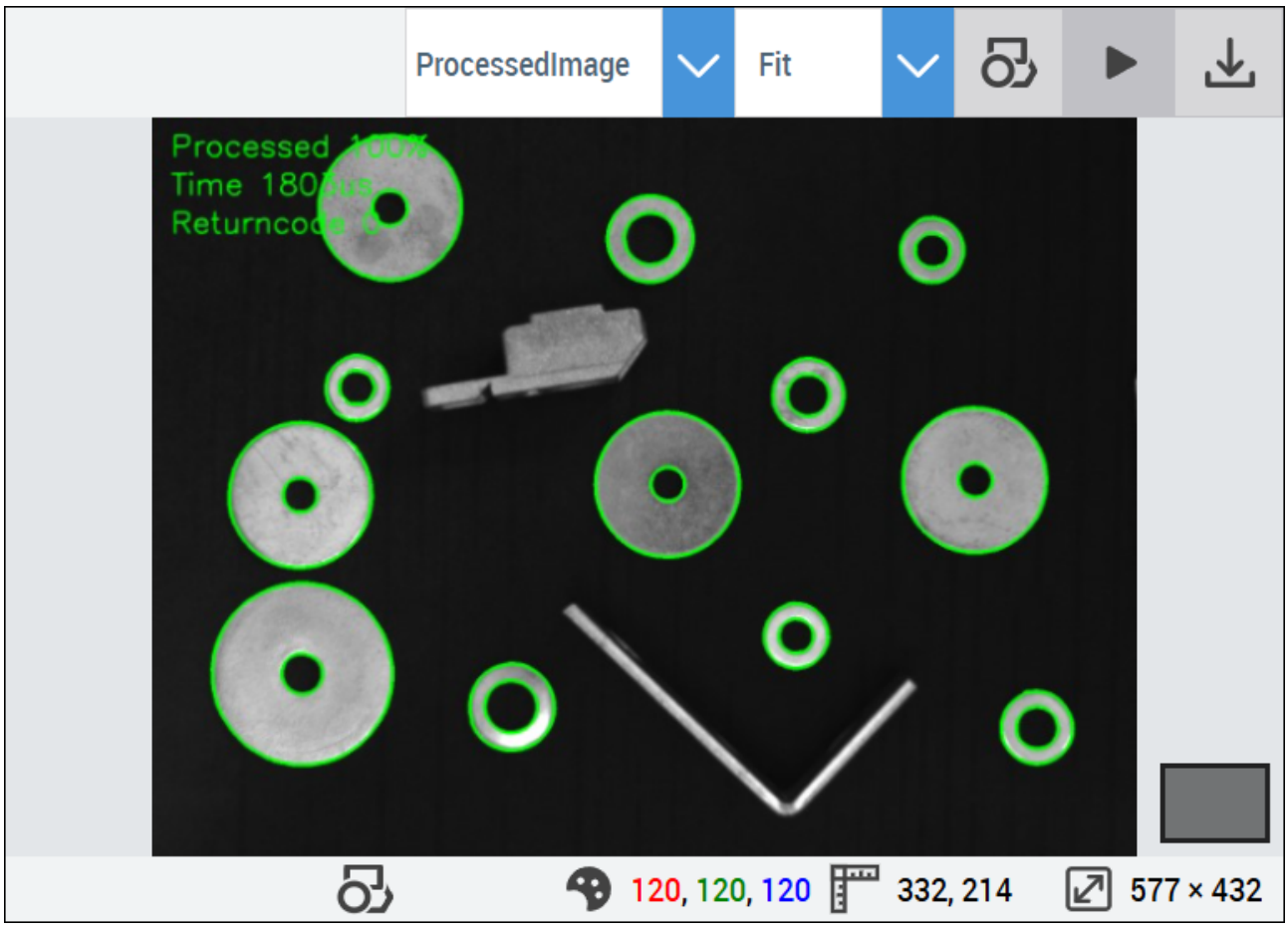
矩形（交互式）和多边形（交互式）控制器主要是辅助控制器，可在 Image Watch（图像查看）控制器中重复使用。必要时，也可单独使用。此外，这些控制器还可以借助其属性，用于点、线、圆形、椭圆形或正方形等派生形状。

此外，该软件包还包含以下辅助函数：

函数	类别	描述
BoxColorConverter [▶ 2717]	Vision	转换颜色 [▶ 2689]控制器的 `ColorValue` 属性，以便在颜色框中显示，反之亦然。
ConvertColor [▶ 2717]	Vision	在不同格式之间转换颜色值。
PixelColorFormatter [▶ 2718]	格式化	将 Image Watch（图像查看） [▶ 2666]控件的 `PixelColor` 属性转为标准化文本，以便在 Toolbar（工具栏） [▶ 2678]中显示。
ShapeValueFormatter [▶ 2719]	格式化	将 Image Watch（图像查看） [▶ 2666]控件的 `ShapeValue` 属性转为标准化文本，以便在 Toolbar（工具栏） [▶ 2678]中显示。
ToRotatedRectangle [▶ 2719]	数据转换	将 TcVnRectangle [▶ 2721] 类型的矩形转换为 TcVnRotatedRectangle [▶ 2720] 类型。
ToUprightRectangle [▶ 2719]	数据转换	将 TcVnRotatedRectangle [▶ 2720] 类型的矩形转换为 TcVnRectangle [▶ 2721] 类型。

7.2.1 图像查看

除显示图像外，**Image Watch（图像查看）**控件 `TcHmiVnImage` 还可提供许多其他工具和信息显示。有关各区域和组件的概述，请参见控制结构 [▶ 2673]分章。每个配置选项的说明可在本页的以下类别中找到。每个类别都有分章，提供更多信息、细节和应用示例。为了从 PLC 直接显示 TwinCAT Vision 图像，还需要适用于 HMI 服务器的 Vision 扩展 [▶ 2660]。



对于以下所有属性，控件有一个 getter 和一个 setter 方法。例如，属性 `BarColor` 存在两个方法：`getBarColor(): SolidColor` 和 `setBarColor(value: SolidColor): void`。属性的使用将在链接分章中进行更详细的说明。

Controls（控件）基类所提供的所有其他属性、事件和权限均基于 TwinCAT HMI 文档。

类别：颜色

属性	类型	描述
`BarColor`	`SolidColor`	工具栏和信息栏颜色（请参见 **Toolbar（工具栏）**和 **Infobar（信息栏）**类别）。
`ShapeStrokeColor`	`SolidColor`	可在图像上绘制的形状轮廓的颜色（请参见 **Shapes（形状）**类别）。
`ShapeHandleColor`	`SolidColor`	可在图像上绘制的形状标识符的颜色（请参见 **Shapes（形状）**类别）。
`ThumbnailBackgroundColor`	`SolidColor`	缩略图的背景颜色。
`ThumbnailStrokeColor`	`SolidColor`	缩略图标记的颜色。

类别：常用

以下属性指定了要显示图像的加载资源 [▶ 2674]。

属性	类型	描述
Image	String	图像显示资源。可以是图像文件的路径，也可以是数据 URI。
ImageList	ImageList	已命名图像资源的列表。如果列表不为空，Image 的绑定将被 ImageIndex 所选的资源替换。
ImageIndex	Number	如果 ImageList 不为空，选择将显示的列表图像资源。
Alt	String	如果当前图像资源不可用，将显示替代文本。
DisplayedImage (只读)	String	始终包含当前显示的图像。例如，可用于与直方图控件 [▶ 2707]进行同步。

函数	描述
downloadImage(fileName: string): void	触发以原始尺寸对当前显示图像进行浏览器特定下载。 也可通过 Toolbar（工具栏） [▶ 2678]中的用户界面元素触发。

类别：视图

以下属性和函数决定了显示图像的部分 [▶ 2676]以及如何对其进行更改。

属性	类型	描述
ImageFreeze	Boolean	如果激活，当前显示的图像将固定，直到 ImageFreeze 禁用后才会更新。
Offset	[Number, Number]	图像左上角在控制器坐标系中的位置。
Scale	Number (%)	根据 ScaleReference 按百分比缩放图像。
ScaleReference	ScaleReference	如为 ImageSize，100% 对应原始图像大小。 如为 ControlWidth，100% 对应控制器的宽度。 如为 ControlHeight，100% 对应控制器的高度。 如为 ControlSize，100% 要么对应控制器的宽度，要么对应控制器的高度，具体取决于哪个先限制图像的大小。
FitImage	Boolean	如果激活，图像缩放将处于“自适应”模式。在此过程中，Offset 和 Scale 会进行调整，使图像在控件中居中显示并填充控制器。如果 Offset 或 Scale 在运行时在其他地方被更改，该属性将自动设置为 false。不过，它随时可以通过 true 进行重置。
Draggable	Boolean	如果激活，则可以使用鼠标或触控移动图像。在此过程中，Offset 会自动调整。

属性	类型	描述
Scalable	Boolean	如果激活，则可以使用鼠标滚轮或多点触控缩放图像。在此过程中，Offset 和 Scale 会自动进行调整。
ScaleMin	Number	通过鼠标滚轮或多点触控缩放时的最小 Scale。
ScaleMax	Number	通过鼠标滚轮或多点触控缩放时的最大 Scale。
ScalingCenter	ScalingCenter	通过 Scale 属性 缩放时保持固定的点。 如为 TopLeftCorner，图像的左上角保持固定。 如为 ImageCenter，图像的中心点保持固定。 如为 ControlCenter，控制器的中心保持固定。 使用鼠标滚轮缩放时，光标的位置始终保持固定。
View (只读)	Number[4]	以感兴趣区域的形式指定图像的显示部分。格式为：[X, Y, Breite, Höhe]。

函数	描述
centerImageToControl(): void	调整 Offset，使当前图像位于控件的中心。
fitImageToControl(center: boolean): void	已废弃，请使用 FitImage 属性。 调整 Offset 和 Scale，使当前图像在控制器中居中并填充控制器。
setView(view: number[]): void	更改 Offset 和 Scale，以便显示图像的指定部分。如果剪切的长宽比与控制器不匹配，则不一定要与视图匹配。

类别：工具栏

以下属性可用于自定义控件的工具栏 [▶ 2678]。

属性	类型	描述
ToolbarVisible	Boolean	工具栏显示。
ToolbarItems	ToolbarItems	选择要在工具栏中显示的元素。
ScaleOptions	Number[]	除“Fit（适合）”选项外，还可在 ScaleSelection 组合框中显示和选择缩放值。
ToolbarHeight	Number (px)	工具栏的高度。
ImageSelectionWidth	Number (px)	ImageSelection 组合框的宽度。
ScaleSelectionWidth	Number (px)	ScaleSelection 组合框的宽度。
QueueIndexWidth	Number (px)	QueueIndex 输入字段的宽度和条目总数。
ScaleSelectionEditable	Boolean	如果激活，可将值手动输入 ScaleSelection。如果未启用，则只能从提供的 ScaleOptions 中进行选择。

类别：信息栏

以下属性可用于自定义控件的信息栏 [▶ 2679]。

属性	类型	描述
InfobarVisible	Boolean	显示信息栏。
InfobarItems	InfobarItems	选择要在信息栏中显示的元素。
PixelColorFormat	Function	根据 PixelColor 属性的值为信息栏生成文本的函数。
ShapeValueFormat	Function	根据 ShapeValue 属性的值为信息栏生成文本的函数。
InfobarHeight	Number (px)	信息栏的高度。

类别：形状

以下属性有助于在图像上绘制形状 [▶ 2682]。

属性	类型	描述
ShapeSelectionActive	Boolean	如果激活，则可以在图像上绘制、显示和编辑形状（基于 ShapeType）。
ShapeType	ShapeType	形状类型。可能为： • 点状 • 线 • 矩形/正方形 • 椭圆形/圆形 • 多边形
ShapeIsRotatable	Boolean	如果激活，可以使用用户界面中的标识符旋转合适的图形。（线条、矩形、正方形、椭圆形、多边形）
ShapeData	ShapeData	形状的几何描述。可用于查询绘制形状的数据或显示特定的预定义形状。
ShapeAutoClear	Boolean	如果启用，ShapeData 将在 ShapeType 被停用时重置为相应 ShapeSelectionActive 的基准值。
ShapeStrokeThickness	Number (px)	显示形状线的粗细。
ShapeHandleSize	Number (px)	显示形状标识符的大小。
ShapeClickableSize	Number (px)	显示形状标识符可点击区域的大小。可用于简化在触控面板上编辑形状的过程，而不会显示不必要的大标识符。
ShapeAngleInterval	Number (°)	可以改变形状旋转的步长。例如，当值为 90 时，只有四种可能的排列方式。如为 0，任意旋转。
ShapeIsInImageArea (只读)	Boolean	表示显示形状是否完全位于图像区域内。可用于检查 ShapeData 对某些 PLC 功能的有效性。计算时只使用角点。
ShapeLimitToImage	Boolean	如果激活，则只能在图像中绘制或编辑形状。

属性	类型	描述
ShapeDataUpdate	DataUpdate	指定 ShapeData 应在 (onChanged) 期间更新还是在图形交互结束 (onConfirmed) 时更新。

函数	描述
clearShape(): void	删除显示的形状，同时将 ShapeData 重置为相应 ShapeType 的基准值。

类别：缩略图

以下属性决定缩略图的显示。缩略图 [▶ 2686]显示图像可见部分的概况。

属性	类型	描述
ThumbnailVisible	Boolean	显示缩略图。
ThumbnailPosition	ThumbnailPosition	缩略图的位置。
ThumbnailSize	Number (px)	缩略图的大小。确定缩略图的宽度；高度根据长宽比自动计算。

类别：叠加

以下属性允许在图像上叠加对象 [▶ 2686]（例如作为定位或标记辅助）。

属性	类型	描述
OverlayVisible	Boolean	显示叠加。
OverlayImage	String	叠加在显示图像上的图像。
OverlayElements	OverlayElements	在显示的图像上叠加了几何元素。

类别：信息

以下属性可用于获取有关图像或图像特定像素的信息 [▶ 2687]。

属性	类型	描述
PixelInfoUpdate	PixelInfoUpdate	决定何时更新 PixelColor 和 PixelPosition。 • 如为 Never，它们从未更新。 • 如为 AtClick，它们会在您点击图像的特定部分时更新。 • 如为 AtCursor，它们会在每次在图像上移动光标时更新。此操作会始终显示当前光标位置的值。
PixelColor (只读)	[Number, Number, Number, Number]	所选图像像素的颜色。
PixelPosition (只读)	[Number, Number]	所选图像像素在图像坐标中的位置。
ImageSize (只读)	[Number, Number]	显示图像的原始尺寸。
ImageDisplaySize (只读)	[Number, Number]	显示图像的显示尺寸。

类别：历史

以下属性可用于自定义控件的历史 [▶ 2688]功能。

属性	类型	描述
QueueIndex	Number	设置要显示的历史图像的索引。0 对应最后接收到的图像。仅在 Freeze（冻结）激活时有效。
QueueSize	Number	限定应临时保存多少个最后接收到的图像。根据 FIFO（先进先出）原则，最旧的图像将首先被删除。
ImageTimestamp (只读)	Number	当 ImageFreeze 激活时，所显示的历史图像的时间戳（以毫秒为单位）。

函数	描述
clearHistory(): void	删除整个历史队列。

类别：统计

以下属性可用于自定义控件的统计 [▶ 2689]功能。

属性	类型	描述
NumberOfImageTransmissions	Number	限定统计中应包含多少个最后接收到的图像。
ImageRenderTime	Number	在浏览器中渲染图像的平均时间（以毫秒为单位）。

函数	描述
resetStatistic(): void	重置整个统计数据。

事件

所有事件均列在“Vision”类别中。

事件	描述
.onPixelInfo	如果 PixelInfoUpdate <> Never 且用户点击图像中的某个位置或将鼠标指针移至该位置以选择某个像素，则会触发该事件。 在内部，事件会更新 PixelColor 和 PixelPosition 属性的值。
.onImageReceived	控制器接收到图像时会触发该事件。
.onShapeChanged	一旦 ShapeData 属性发生变化，就会触发该事件。
.onShapeConfirmed	设置形状或完成与形状的交互时会触发该事件。

主题资源

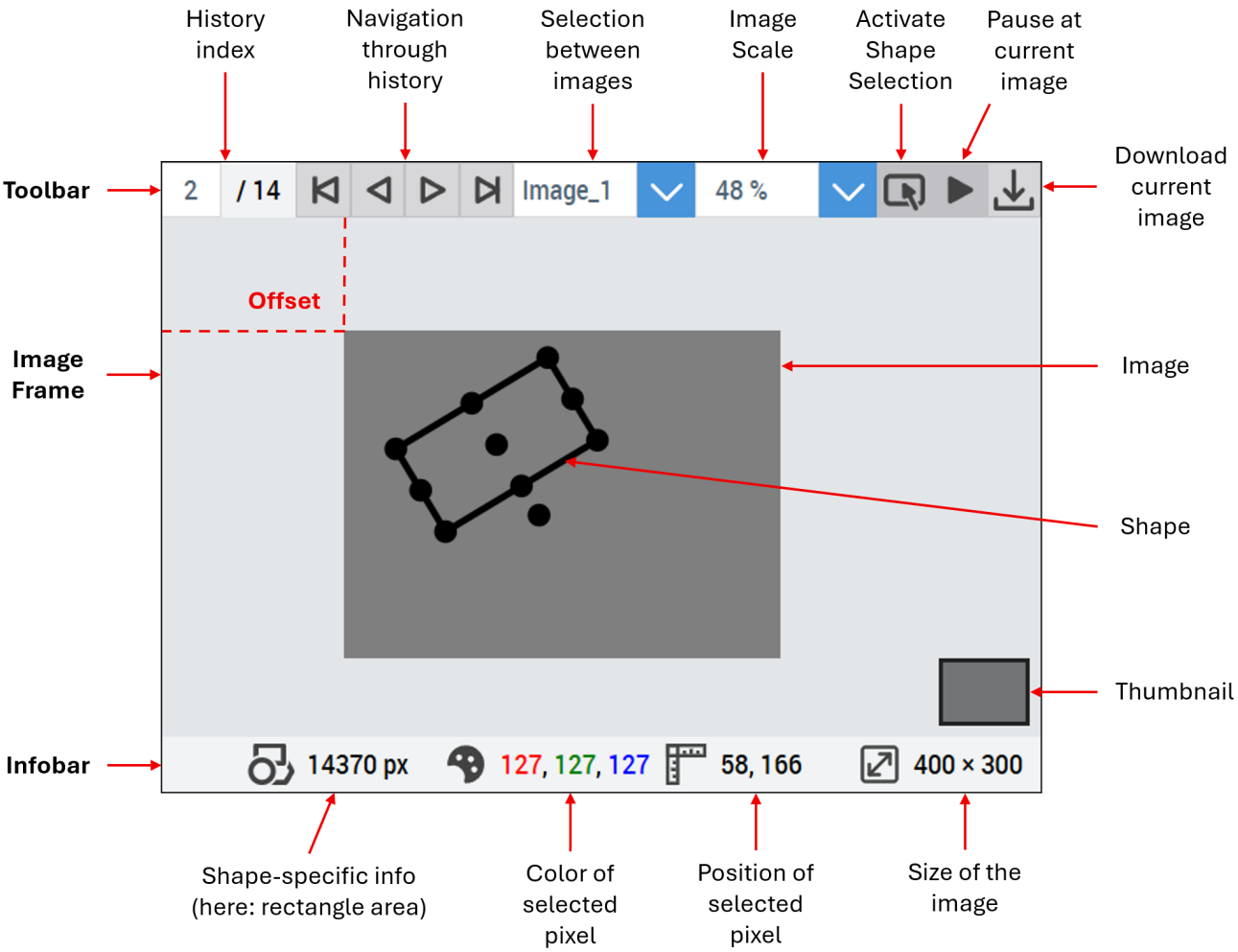
如为 Themes（主题）设置，可以使用以下属性更改子控制器（工具栏中的组合框）的外观。关于各个属性的详细说明，请参见标准 HMI 文档。

属性	类型
ImageSelection_ContentPadding	Padding
ImageSelection_DataHeight	DataHeight
ImageSelection_DropDownBackgroundColor	Color
ImageSelection_DropDownFontFamily	FontFamily
ImageSelection_DropDownFontSize	MeasurementValue
ImageSelection_DropDownFontStyle	FontStyle
ImageSelection_DropDownFontWeight	FontWeight
ImageSelection_DropDownHorizontalAlignment	HorizontalAlignment

属性	类型
ImageSelection_DropDownTextColor	SolidColor
ImageSelection_DropDownVerticalAlignment	VerticalAlignment
ImageSelection_DropDownStyle	TcHmiCombobox.DropDownStyle
ImageSelection_FontFamily	FontFamily
ImageSelection_FontSize	Number (px)
ImageSelection_FontStyle	FontStyle
ImageSelection_FontWeight	FontWeight
ImageSelection_HorizontalAlignment	HorizontalAlignment
ImageSelection_MaxListHeight	Number (px)
ImageSelection_TextColor	SolidColor
QueueIndex_BackgroundColor	Color
QueueIndex_ContentPadding	Padding
QueueIndex_FontFamily	FontFamily
QueueIndex_FontSize	Number (px)
QueueIndex_FontStyle	FontStyle
QueueIndex_FontWeight	FontWeight
QueueIndex_HorizontalAlignment	HorizontalAlignment
QueueIndex_TextColor	SolidColor
QueueIndex_VerticalAlignment	VerticalAlignment
ScaleSelection_ContentPadding	Padding
ScaleSelection_DataHeight	DataHeight
ScaleSelection_DropDownBackgroundColor	Color
ScaleSelection_DropDownFontFamily	FontFamily
ScaleSelection_DropDownFontSize	MeasurementValue
ScaleSelection_DropDownFontStyle	FontStyle
ScaleSelection_DropDownFontWeight	FontWeight
ScaleSelection_DropDownHorizontalAlignment	HorizontalAlignment
ScaleSelection_DropDownStyle	TcHmiCombobox.DropDownStyle
ScaleSelection_DropDownTextColor	SolidColor
ScaleSelection_DropDownVerticalAlignment	VerticalAlignment
ScaleSelection_FontFamily	FontFamily
ScaleSelection_FontSize	Number (px)
ScaleSelection_FontStyle	FontStyle
ScaleSelection_FontWeight	FontWeight
ScaleSelection_HorizontalAlignment	HorizontalAlignment
ScaleSelection_MaxListHeight	Number (px)
ScaleSelection_TextColor	SolidColor

7.2.1.1 控件结构

Image Watch（图像查看）控制器由显示工具和信息的区域（Toolbar（工具栏） [▶ 2678]和 Infobar（信息栏） [▶ 2679]）以及显示图像的主区域组成。在此区域内，可以通过编程、鼠标或触控来移动和缩放图像。缩略图或形状选择等其他功能可根据具体应用使用。



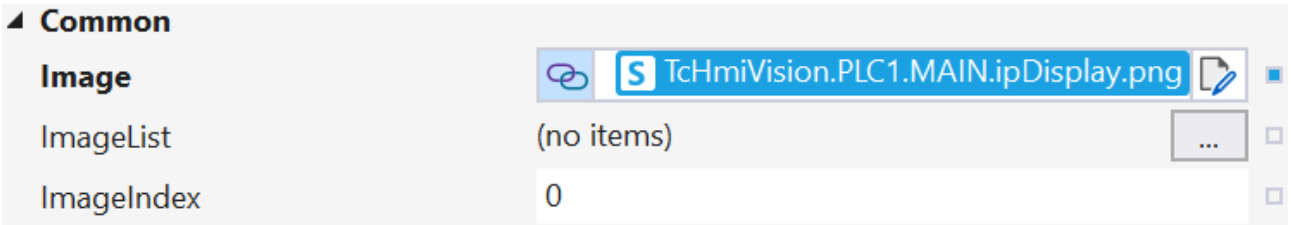
7.2.1.2 图片源

本页解释的属性和函数载于 **Common（常用）**类别中。

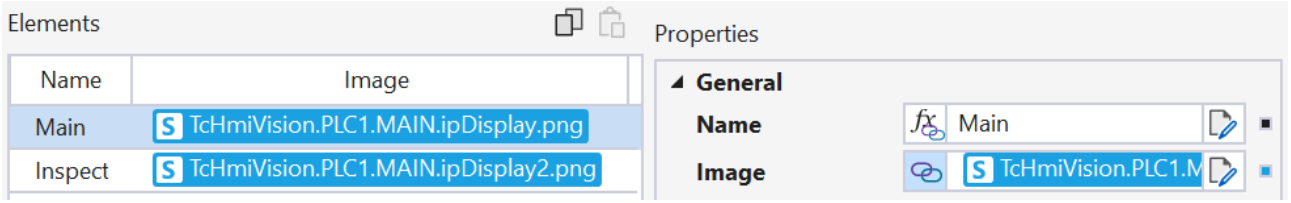
图像资源

图像资源可确定将在 Image Viewer（图像查看器）中显示的图像。预期用途是使用 HMI.Vision 扩展包 [▶ 2660]将 `ITcVnDisplayableImage` 符号作为图像资源进行链接。不过，一般来说，图像资源只需要是一个有效的图像 URL。这可以是项目中的直接路径、外部服务器的 URL 或数据 URL。其作用类似于标准图像控制器的 Src 属性。

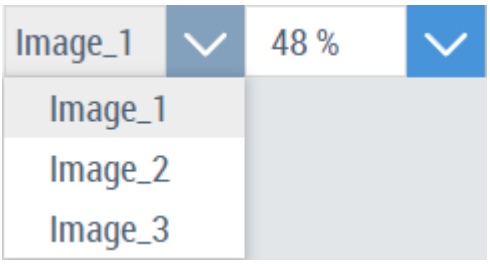
最简单的情况是，通过 `Image` 属性指定图像资源：



此外，还可以在 `ImageList` 属性中存储多个图像资源：



`ImageIndex` 属性可用于选择实际使用的资源。例如，这可用于轻松切换来自 PLC 的各图像符号。也可以通过 Toolbar（工具栏） [▶ 2678]中的 `ImageSelection` 组合框以图表形式进行切换。为此，在 `ImageList` 中不仅可以分配相应的图像资源，还可以分配相关图像名称。



无法同时使用图像和图像列表

在内部，使用 `ImageList` 属性会将所选 `ImageList` 元素的相应图像符号绑定到 `Image` 符号。因此，使用 `Image` 和 `ImageList` 互不兼容；如果已经使用 `ImageList`，则不应将图像符号与 `Image` 属性绑定。

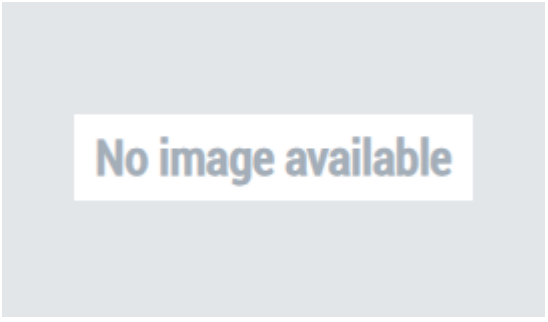
如果当前图像流处于暂停状态（请参见 View（视图） [▶ 2676]类别中的属性 `ImageFreeze`）或者显示的是过去的图像（请参见 History（历史） [▶ 2688]类别），那么通过 `Image` 或 `ImageList` 定义的图像资源不一定与显示的图像相对应。例如，为了将显示的图像与在直方图控件中进一步处理的图像进行同步，可以使用属性 `DisplayedImage`。它始终包含实际显示的图像，并且可以直接链接到直方图控件的 `Image` 属性。

接收事件

每次控件接收新图像时，都会触发 `onImageReceived` 事件。

替代文本

如果未指定图像资源或无法从指定资源加载有效图像，则会显示替代文本。这可通过 `Alt` 属性进行更改，默认情况下为 "无可用图像”。



图像下载

当前显示的图像可通过浏览器特定下载本地保存到客户端设备上。下载可通过 `downloadImage` 函数或 Toolbar（工具栏） [▶ 2678]上的下载按钮触发。

图像以原始大小保存。

文件名可在程序调用中作为参数指定。无需指定文件扩展名，因为它会自动添加。如果参数为空，也会自动创建文件名。对于符号，首先使用符号名；对于数据 URL，则相应地使用控制器名：

<符号名/控制器名>_<日期>_<时间>-<MS>.<文件扩展名>

对于图像路径，使用指定路径中的文件名和文件扩展名，不附加时间戳。

在控制器内无法选择保存路径，但它对应于所使用浏览器的下载路径设置。通常的下载路径是 **C:\User\<User>\Downloads**。

图像下载要求和限制

由于图像下载通过浏览器特定功能完成，因此可能性、设置和功能取决于 TwinCAT HMI 客户端所使用的网络浏览器。

`downloadImage` 函数需要对 TwinCAT HMI 用户管理的**操作**授权。

该函数不能用于作为数据流链接的图像，也不能在 HMI 工程的实时视图中使用。

7.2.1.3 图像视图

本页解释的属性和函数载于 **View（视图）**类别中。

在内部，显示图像的大小和位置由 `Offset`、`Scale` 和 `ScaleReference` 属性精确定义。不过，改变大小和位置的典型用例使用触摸控和鼠标手势等交互功能。因此，这些手势会自动在内部更改上述属性。

属性： `Offset`

`Offset` 属性指定图像相对于控制器本身的位置。左上角被定义为原点，X 轴向右，Y 轴向下。图像偏移也是从图像的左上角开始测量。偏移坐标以像素为单位指定。

如前所述，`Offset` 属性由控制器本身更改，例如，当用户使用触控或鼠标手势与图像交互的情况。

默认情况下，偏移属性与 `TwoWay` 绑定相关联。

属性： `Scale` **和** `ScaleReference`

`Scale` 属性以百分比为单位指定缩放系数，用于计算图像在屏幕上显示的像素大小。根据 `ScaleReference` 属性的值，大小总是根据原始图像大小或控件尺寸计算。

`ScaleReference` 可以是以下值之一：

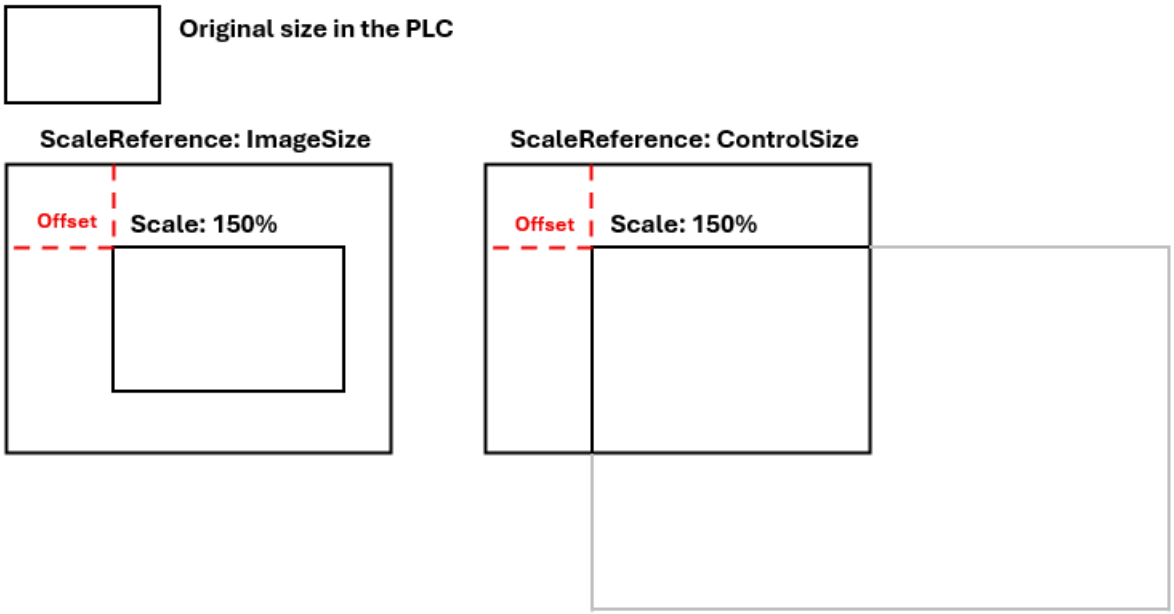
ImageSize	显示图像的大小等于原始图像大小乘以 `Scale` 属性的值。 例如：图像原始大小为 300x200，`Scale` 属性为 150%。最终显示图像大小为 450x300。
ControlSizeWidth	显示图像的大小等于控制器的宽度（或更精确地说，控制器内的图像框架）乘以 `Scale` 属性的值。
ControlSizeHeight	显示图像的大小等于控制器的高度（更精确地说，控制器内图像框架的高度）乘以 `Scale` 属性的值。
ControlSize	显示图像的大小等于控制器的大小（或更精确地说，控制器内的图像框架）乘以 `Scale` 属性的值。如果控制器的长宽比与图像长宽比不匹配，则会使用高度或宽度作为参考尺寸，具体取决于哪个尺寸相差更多。

`ImageSize` 设置的优势在于显示尺寸与图像原始尺寸之间的直接关系。PLC 中尺寸不同的两个图像在 HMI 中总是有不同的尺寸。`ControlSize` 设置的优势在于根据控制器自动调整图像。无论图像在 PLC 中的实际多大如何，其大小始终是一样的。

长宽比

始终计算图像的高度和宽度以保持原始图像的长宽比。

示例： `Offset`、`Scale` **和** `ScaleReference`



属性： `FitImage`

要自动将图像居中并调整其适应控制器尺寸，可以激活 `FitImage` 属性。此时，自适应模式处于激活状态，因此后续接收的或历史中的图像即使分辨率不同，也会按相应的缩放和位置进行显示。如果在运行时通过任何其他方式（例如鼠标拖拽、滚动或程序设置）更改了 `Scale` 或 `Offset`，则会将 `FitImage` 自动设置为 `false`。自适应模式随即结束。此操作也可以手动完成，并且可以通过设置为 `true` 随时重新激活自适应模式。

函数： `FitImageToControl` （已废弃，请改用 `FitImage`）

要在图像查看器中将当前显示图像最大化并居中，可以调用 FitImageToControl 函数。如果要从一开始就这样做，则应将该函数与 .onAttached 事件相关联。一旦激活该函数，所有后续图像也将居中并最大化，直到视图在其他地方被修改，从而取消该模式。

属性： `ScalingCenter`

`ScalingCenter` 提供调整缩放行为的可能性。`Scale` 属性的单独变化始终会导致图像右下角发生变化，而左上角则保持不变。要改变这一点并固定其他点，可以使用 `ScalingCenter` 属性。在这种情况下，`Offset` 属性会自动进行调整。提供以下选项：

ScaleCenter 值	描述
`TopLeftCorner`	缩放时图像左上角保持居中。
`ImageCenter`	缩放时图像中心保持居中。
`ControlCenter`	缩放时控制器中心保持居中。

手势

用户可以通过以下手势/交互来改变图像的大小和位置：

- 在图像框内任意位置按住鼠标左键，拖动鼠标移动图像，从而改变 `Offset` 属性。`Scale` 属性不受影响。
- 用一根手指触摸图像框，然后移动手指以移动图像，从而改变 `Offset` 属性。`Scale` 属性不受影响。
- 滚动鼠标滚轮以改变图像大小，从而改变 `Scale` 属性。缩放的中心将始终是鼠标指针的位置。因此，`Offset` 属性也会相应改变。鼠标指针必须始终位于控制器上方。
- 用两根手指触摸图框，然后朝一个方向移动两根手指，相向或相离，即可改变图片的位置和大小。`Offset` 和 `Scale` 属性均进行了调整。

为防止这些UI交互操作，可将 `Draggable` 和 `Scalable` 属性设置为 `false`，以防止UI交互操作引起的位置或尺寸改变。始终有可能更改值（例如写入符号），这不受两个属性的影响。

属性： `ScaleMin` 和 `ScaleMax`

控制定位和缩放系统应用限制规则，以防止用户创建混乱或失向图像视图。

`ScaleMin` 和 `ScaleMax` 属性表示对 `Scale` 属性设置的UI交互限制。

此外，控制器会自动将图像限制在具有一定外填充（目前为 10px）的边界内。在此限值之外的位置是无用的，因为看不到图像。

属性： `View` **（只读）**

`Offset` 和 `Scale` 属性有利于图像视图的内部处理，但对UI操作来说不一定直观（例如定义图像区域）。人们通常倾向于从感兴趣区域或类似表征的角度来思考问题。因此，在改变图像的大小或位置后，会计算并通过只读 `View` 属性提供图像的当前可见区域。

内部计算总是将坐标原点设置为图像的左上角，并由此计算左上角位置以及图像框的宽度和高度。

`View` 属性以 `[X, Y, Breite, Höhe]` 形式的数组提供。

要使用视图描述设置位置和大小，请参见下一节中的 `SetView` 函数。

函数： `SetView`

`SetView` 函数允许UI操作显示图像的特定区域。调用此函数时，由参数指定的图像区域将被最大化，并在图像框内居中。

区域必须在原始图像坐标中指定为 `[x, y, Breite, Höhe]`。

不同的视图

根据指定区域和图像框的长宽比，请求的视图可能与实际视图不同。这反过来又可以通过只读 `View` 属性进行检索。

单独行动

视图功能不是通过属性访问，而是通过一个只读属性和一个 setter 方法访问。其中的原因是上面提到的有序视图和实际视图之间的区别，而且在实现哪个视图的问题上，**View（视图）**属性会与 `Offset` 和 `Scale` 属性相冲突。由于视图 setter 是一次性操作，而不是绑定属性，因此无法做到这一点。

停止图像流

如果 `ImageFreeze` 属性为 true，即使收到新图像，当前显示图像也不会改变。预期用例是冻结当前图像，以供进一步审查，而无需在 PLC 中添加额外的逻辑。

如果 `ImageFreeze` 属性从 `true` 变为 `false`，控制器会立即再次显示最后一个图像，并忽略中间图像。

`ImageFreeze` 属性可由工具栏中的 Freeze（冻结）按钮控制（请参见 Toolbar（工具栏） [▶ 2678]类别中的 `FreezeButtonVisible` 属性）。

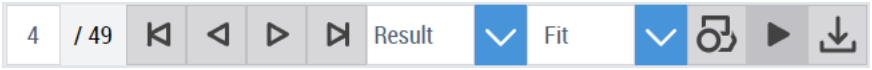
7.2.1.4 工具栏

本页介绍的属性和函数均可在 **Toolbar（工具栏）**类别中找到。

工具栏中的控制器可用于交互式执行各种操作，否则只能通过属性访问。每个元素都可以显示或隐藏。此外，整个工具栏的高度可以调整，也可以完全隐藏。

工具栏中元素的位置是固定的，无法自定义。如果仍有必要进行此类调整，您可以使用 `ToolbarItems` 属性隐藏相应的组件，通过自己的控件自行实现这些调整，然后将其放置在 Image Control（图像控件）工具栏上。

可用的工具栏组件分为 General（常规）和 History（历史）组。下面对每个组中的各个组件从左到右进行说明，General（常规）组以显示的 `Result` 图像选择开始：



类别：常规

ImageSelection	用于从图像列表中选择要显示的图像的组合框（请参见 **Common（常用）**类别中的 `ImageList` 属性）。仅在图像列表至少包含一个元素时显示。在图像之间进行切换时，加载时间可能较短。
ScaleSelection	用于选择特定的缩放值来调整当前图像的组合框。可通过 `ScaleOptions` 属性调整可选择的缩放值。 如果在设置用户自定义比例时停用 `ScaleSelectionEditable`，那么组合框不会再显示数值。此为组合框控制器的默认行为。
ShapeButton	用于设置 `ScaleSelectionActive` 属性的值的按钮。如果激活，则会显示现有形状并可进行编辑。如果未显示形状，则可以绘制新的形状。
FreezeButton	用于设置 `ImageFreeze` 属性的值的按钮。如果激活，则当前显示的图像会固定，不会更新。
DownloadButton	用于下载当前图像的按钮，带默认名称（请参见 Image download（图像下载） [▶ 2674] 下的 `downloadImage` 函数）。若要指定不同的文件名，客户可执行此按钮，然后用自定义名称调用函数。

类别：历史

QueueIndex	当 `ImageFreeze` 激活时，用于选择或显示指定历史图像的索引输入值。0 对应最后接收到的图像。
NavigationButtons	当 `ImageFreeze` 激活时，用于在图像历史中向前或向后移动一个图像条目的按钮。
NavigationEndButtons	当 `ImageFreeze` 激活时，用于跳转到历史中接收到的第一个或最后一个图像的按钮。

也可通过编程访问的函数

通过工具栏的界面元素可以访问的操作，也可以通过控制器的属性以编程方式访问。

尺寸设置

工具栏提供了多种自定义选项，例如，可以通过 `ToolbarHeight` 属性调整高度。此外，各个元素的宽度可以通过 `ImageSelectionWidth`、`ScaleSelectionWidth` 和 `QueueIndexWidth` 属性进行调整。所有按钮的宽度会根据工具栏的高度自动调整，使其呈正方形。

无法调整组合框下拉菜单按钮的宽度，因为标准组合框控制器中不具备这一功能。

在内部，`ToolbarHeight` 属性的最大限值为 150 个像素。

背景颜色

工具栏的背景颜色默认为半透明白色 `"rgba(255, 255, 255, 0.5)"` 或 R：255、G：255、B：255、A：127，也可使用 **Colors（颜色）**类别中的 `BarColor` 属性进行自定义。

高级设置

其他图形设置（如组合框的字体）可通过 Themed Resources（主题资源）进行更改。基本上，提供了组合框、文本块和数值输入的属性。可在 `TcHmiVnImage > Themed Resources` 的主题编辑器中自定义 Themed Resources（主题资源）。这些设置会应用于 Image Viewer（图像查看器）控制器的所有实例。

7.2.1.5 信息栏

本页解释的属性和函数均可在 **Infobar（信息栏）**类别中找到。

信息栏中的显示元素可用于显示有关形状值、像素值或图像大小的各种信息。每个元素都可以显示或隐藏。此外，整个信息栏的高度可以调整，也可以完全隐藏起来。

信息栏中元素的位置是固定的，无法自定义。如果仍有必要进行此类调整，您可以使用 **InfobarItems** 属性隐藏相应的元素，并通过自己的控制器自行实现这些调整，然后将其放置在 Image Control（图像控制器）信息栏上。可用的信息栏元素从左到右依次为：



ShapeValue	当前显示形状的计算值。该值由绑定到 `ShapeValueFormat` 属性的函数计算。 `ShapeValueFormatter` 标准格式化函数 [▶ 2719]会根据所选形状类型返回一个特定值： • 点的坐标 • 线的长度 • 正方形、矩形、圆形或椭圆形的面积 • 多边形的点数 本例 [▶ 2680]介绍了自定义格式化函数的实现和使用方法。
PixelColor	所选像素的颜色值。该值由绑定到 `PixelColorFormat` 属性的函数计算。 `PixelColorFormatter` 标准格式化函数 [▶ 2718]会根据所选颜色空间返回一个特定值。可选择的颜色格式有：`Gray`, `RGB`, `RGBa`, `HSV`, `HSV FULL`, `HLS`, `HLS FULL`, `Lab`, `YUV`, `YCrCb`。您还可以设置是否以及如何进行四舍五入。可在 0-3 `Decimals` 和 `Unrounded` 之间进行选择。还可以设置 PiyelType。此处的可能值为 `8-Bit (USINT)` 和 `16-Bit (UINT)`。 本例 [▶ 2681]介绍了自定义格式化函数的实现和使用方法。
PixelPosition	所选像素的位置。
ImageSize	当前显示图像的原始尺寸。

尺寸设置

信息栏通过 `InfobarHeight` 属性为自身高度提供自定义选项，其内部限值最大为 150 个像素。

背景颜色

信息栏的背景颜色默认为半透明白色 `"rgba(255, 255, 255, 0.5)"` 或 R：255、G：255、B：255、A：127，也可使用 **Colors（颜色）**类别中的 `BarColor` 属性进行自定义。

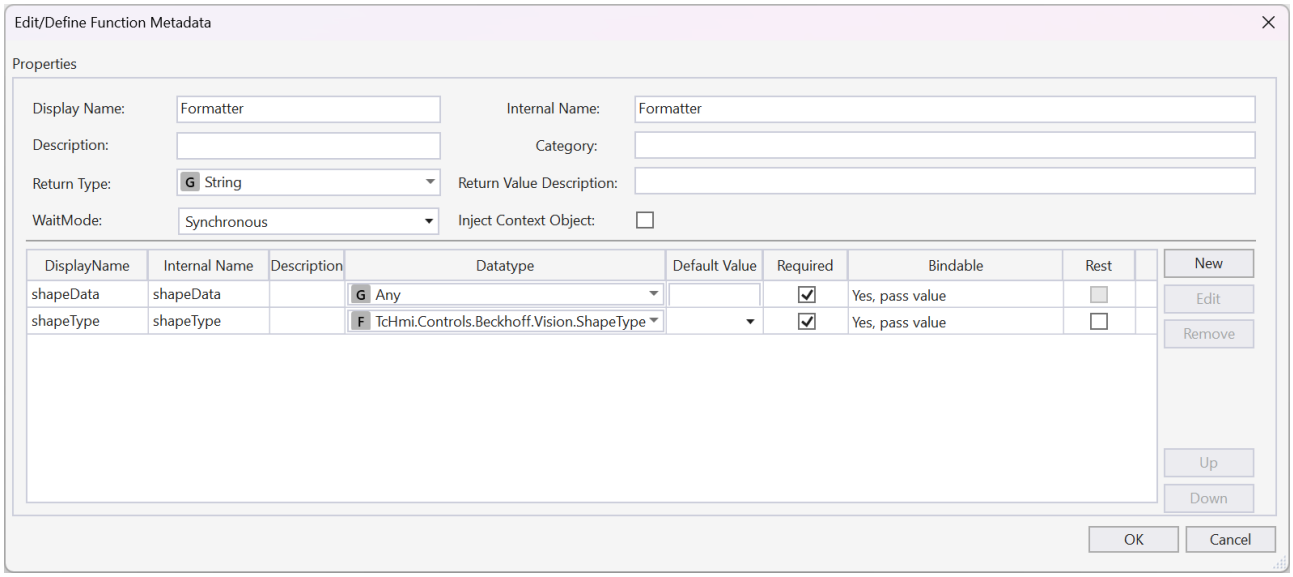
7.2.1.5.1 ShapeValueFormatting 函数

除了 `ShapeValueFormatter` 函数外，还可以创建一个自定义格式化函数，以便在信息栏中显示其他形状信息。下方示例展示了如何创建这样一个函数，以及它必须具备哪些接口。

在 HMI 文档中会说明各函数的一般使用方法。

创建 TypeScript 函数后，必须创建两个参数。第一个参数是以与 ShapeData 属性相同的方式获取传递的 Shape（形状）说明，因为可从控制器中获取。该参数必须创建为 `Any` 类型。第二个参数传递类似于 ShapeType 属性的形状类型。必须将该参数指定为 `TcHmi.Controls.Beckhoff.Vision.ShapeType` 类型。需要注意的是，由于时序原因，这些传递的参数可能与函数执行时相应控制属性的值不同。

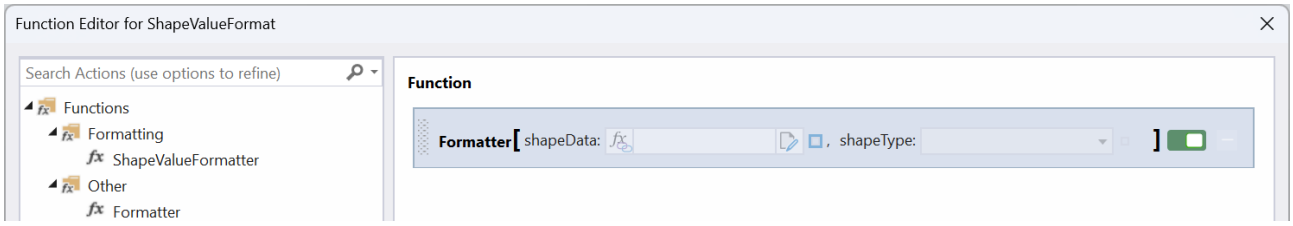
该函数必须返回一个 `String` 类型的值，然后显示在信息栏中。每当形状说明发生变化，就会调用该函数，因此需要更新信息栏中的字符串。



在以下代码示例中，如果形状数据中包含对象，则会返回宽度和高度：

```
namespace TcHmi {
    export namespace Functions {
        export namespace TcHmiProject {
            export function Formatter(shapeData: any, shapeType: any) {
                if (shapeData.stSize != null && shapeData.stSize.fWidth != null &&
                    shapeData.stSize.fHeight != null) {
                    return 'w: ' + shapeData.stSize.fWidth.toFixed(0) + ' h: ' +
                        shapeData.stSize.fHeight.toFixed(0);
                }
                else {
                    return '';
                }
            }
        }
    }
}
TcHmi.Functions.registerFunctionEx('Formatter', 'TcHmi.Functions.TcHmiProject',
TcHmi.Functions.TcHmiProject.Formatter);
```

要使用您创建的格式化函数，请点击 `ShapeValueFormat` 属性上的 Select（选择）按钮，然后在 Function Editor（函数编辑器）中选择您刚刚创建的功能。



7.2.1.5.2 PixelColorFormatting 函数

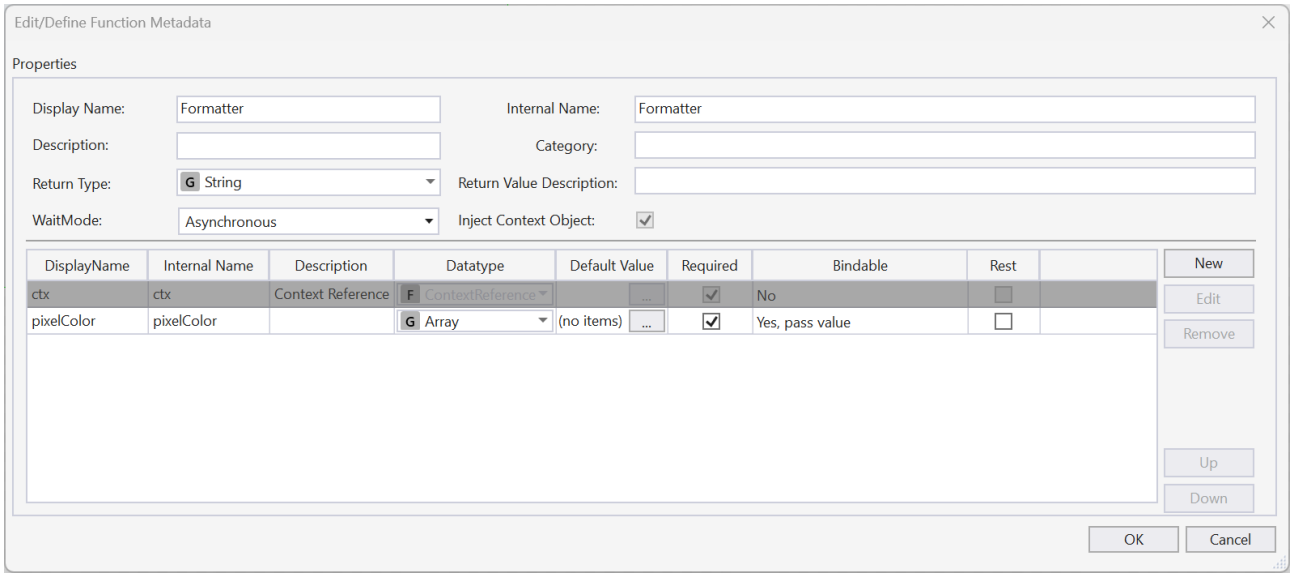
作为 `PixelColorFormatter` 函数的替代方案，可以创建一个用户定义的格式化函数，以不同的方式对所选像素的颜色进行格式化。该函数会获取像素的颜色值并返回一个字符串，该字符串显示在信息栏中。下方示例展示了如何创建这样一个函数，以及它必须具备哪些接口。



在 HMI 文档中会说明各函数的一般使用方法。

创建 TypeScript 函数后，必须将 WaitMode 改为 `Asynchronous`。这样就创建了返回所需的参数“ctx”。然后必须再创建一个“Array（数组）”类型的参数，该参数以数组的形式接收颜色值，接收方式与控制器的 PixelColor 属性相同。需要注意的是，由于时序原因，传递的参数可能与函数执行时相应控制属性的值不同。

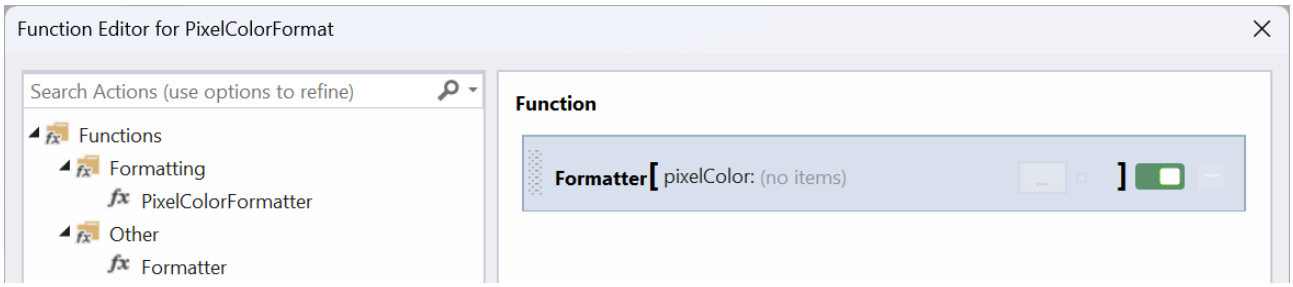
函数必须返回一个 `String` 类型的值，以便在信息栏中显示。当 `PixelColor` 属性的值发生变化以及信息栏中的字符串需要更新时，便会进行调用。



以下方代码块为例，只要 pixelColor 中至少有 3 个元素，就会返回红、绿、蓝 3 种颜色值。数值乘以 256，转换为 16 位：

```
namespace TcHmi {
    export namespace Functions {
        export namespace TcHmiProject {
            export function Formatter(ctx: any, pixelColor: number[]) {
                if (pixelColor.length >= 3) {
                    ctx.success(`<span style="color:red;">${pixelColor[0] * 256}</span>, ` +
                                `<span style="color:green;">${pixelColor[1] * 256}</span>, ` +
                                `<span style="color:blue;">${pixelColor[2] * 256}</span>`);
                }
                else {
                    ctx.success('');
                }
            }
        }
    }
}
TcHmi.Functions.registerFunctionEx('Formatter', 'TcHmi.Functions.TcHmiProject',
TcHmi.Functions.TcHmiProject.Formatter);
```

要使用您创建的格式化函数，请点击 `PixelColorFormat` 属性上的 Select（选择）按钮，然后在 Function Editor（函数编辑器）中选择您刚刚创建的功能。



7.2.1.6 形状

本页解释的属性和函数载于 **Shapes（形状）**类别中。

一个常见的用例是用户选择某些图像区域。可以是感兴趣的区域，也可以是物体的位置。另一个相关用例是用户自定义测量图像距离或其面积。

为了适应所有这些用例，**Shapes（形状）**类别中的属性提供了灵活的功能，用于在图像上绘制和编辑不同的形状，并检索其各自的几何描述。

属性：ShapeSelectionActive

可以通过 ShapeSelectionActive 属性或 Toolbar（工具栏） [▶ 2678]中的 **Shape（形状）**按钮激活形状选择。只要该属性已激活，就可以通过点击图像表面绘制形状，并通过拖放角点或形状本身进行编辑。如果该属性设置为 false，并且 ShapeAutoClear 属性已激活，那么当前形状会被重置并丢失，因此必须先提取所需数据。

可通过 ShapeData 属性获取绘制形状的几何描述。根据 ShapeType 属性（见下文）的不同，形状数据的格式也不同。由于符号直接绑定到内部使用的形状控制器，因此您可以期待与交互式控制器本身（例如 TcHmiVnPolygon 和 TcHmiVnRectangle）中使用的格式完全相同。ShapeLimitToImage 属性可用于将形状限制在图像区域内。如果情况（默认值）并非如此，请注意形状也可以取负值。
shapeIsInImageArea 函数可用于查询形状是否位于图像内。

属性：ShapeType und ShapeData

根据两种不同的数据类型，可提供不同类型的形状。您可以使用 ShapeType 属性选择以下形状之一：

ShapeType	ShapeData	PLC 声明
点	[[x, y]]	ARRAY[0..0] OF TcVnPoint2_REAL or ARRAY[0..0] OF TcVnPoint2_LREAL
线	[[x1, y1], [x2, y2]]	ARRAY[0..1] OF TcVnPoint2_REAL or ARRAY[0..1] OF TcVnPoint2_LREAL

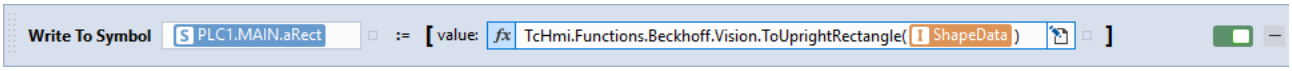
正方形、
矩形、
圆形、
椭圆
	{
	fAngle：角度、
	aCenter：[cx, cy]、
	stSize：{
	fWidth：宽度、
	fHeight：高度、
	},
	}
		TcVnRotatedRectangle

多边形	[[x1，y1]，[x2，y2]，…，…，[xn，yn]]	ARRAY[0..n] OF TcVnPoint2_REAL or ARRAY[0..n] OF TcVnPoint2_LREAL

ShapeData 中的值指的是原始图像的图像坐标。

根据定义，由于正方形和圆形的高度和宽度始终是相同的，因此，如果长宽比不同，那么在点击第二个位置时，这两种形状不会被精确绘制到该端点。在这种情况下，可以通过移动相应的角来到达所需位置。

ToUprightRectangle [▶ 2719] 转换函数可用来将 TcVnRotatedRectangle [▶ 2720] 类型的 Square（正方形）和 Rectangle（矩形）的形状类型数据直接传输到 TcVnRectangle_UDINT [▶ 236] 或 TcVnRectangle [▶ 2721] 类型的 PLC 变量中，并将其用作不带旋转角度的直立外接矩形。对于相反的情况，还有 ToRotatedRectangle [▶ 2719] 函数。



ShapeData 属性可通过 Java 脚本进行如下编程更改：

```
let control = TcHmi.Controls.get("TcHmiVnImage")
// Polygon with 4 Points
control.setShapeData([[10, 10], [10, 100], [150, 150], [100, 10]])
```

```
// Rectangle
control.setShapeData({"aCenter":[200,200],"fAngle":15,"stSize":{"fWidth":150,"fHeight":150}})
```

在第一个示例中，写入了一个包含 4 个点的多边形数组。要写入一个点，只需第一个元素，而要写入一条线，则需要前两个元素。在第二个示例中，写的是一个矩形。由于形状类型正方形、圆形和椭圆形基于相同的数据类型，因此这些类型也可以使用相同的符号。

也可通过 WriteToSymbol 操作将数据写入 ShapeData。为此，请复制 setShapeData() 函数圆括号内的部分，并将其粘贴为对象值。

属性：ShapeAutoClear

如果 ShapeAutoClear 属性已激活，并且在 ShapeSelectionActive 为 true 时更改了 ShapeType，则当前形状会被重置，并可绘制所选形状类型的新形状。如果 ShapeAutoClear 未激活，并且在同一基本控制器的形状类型之间切换，则在切换 ShapeType 时会保留数据，并直接显示新形状。点、线和多边形类型属于一组，使用相同的基本控制器，而正方形、矩形、圆形和椭圆形类型属于另一组。

外观

通过调整轮廓的颜色和大小/粗细，可以改变形状和控制点的外观：

- 通过 ShapeStrokeThickness，您可以设置线条的粗细（单位：像素）。
- 通过 ShapeHandleSize，您可以设置显示的形状控制点的大小（单位：像素）
- 通过 ShapeClickableSize ，可设置控制点可点击区域的大小（单位：像素）。这样可以简化触控面板上的操作，而不必显示不必要的大手柄。
- 通过 ShapeStrokeColor 和 ShapeHandleColor，可以独立设置线条和控制点的颜色。

属性：ShapeIsRotatable

如果 ShapeIsRotatable 已激活，则会为合适的形状（线条、矩形、正方形、椭圆形、多边形）显示一个额外的手柄，通过该手柄可以旋转形状。使用 ShapeAngleInterval 属性，可以设置可改变形状旋转的步长（单位：度）。默认值为 0°，可根据需要旋转形状。例如，如果形状只能放置在 4 个位置，则输入 90° 作为数值。

属性：ShapeLimitToImage und ShapeIsInImageArea

通常情况下，一个形状必须完全位于图像中，例如可以设置一个感兴趣的区域。如果 ShapeLimitToImage 属性已激活，则只能在图像内绘制和移动形状。这一限制不会影响形状的程序设置，只会影响图形交互。还需要注意的是，如果已设置 ShapeIsRotatable 和 ShapeAngleInterval，那么形状的旋转仍不受限制。

如果 ShapeLimitToImage 未激活，则可以通过 ShapeIsInImageArea 属性查询形状的定位。只有在当前存在有效的 ShapeData 且形状的所有角点都位于图像区域内时，才会返回 true。角点还用于检查圆形和椭圆形的形状。为了能够区分不存在的形状和图像中没有的形状，还应该检查是否已设置 ShapeSelectionActive，以及 ShapeData 是否有效。

属性：ShapeDataUpdate

ShapeDataUpdate 属性可用于指定应在何时更新 ShapeData 属性。如果已选择 onChanged，则 ShapeData 会在图形交互过程中不断更新。另一方面，如果已选择 onConfirmed，则 ShapeData 只会在交互结束时更新。

属性：ShapePolygonNumPoints und ShapePolygonClosed

如果将 ShapeType 属性设置为 Polygon，则可通过 ShapePolygonNumPoints 属性定义多边形的完整点数。例如，如果指定了三个点，那么在绘制第三个点时，多边形就会自动完成。如果使用默认值 0，则点的数量是任意的。只有点击第一个点时，多边形才会完成。ShapePolygonClosed 属性还可用于指定多边形的首尾点是否应相连，即多边形是否应显示为封闭状态。

函数：clearShape()

可以通过调用 clearShape 函数，以编程方式删除绘制的形状。

显示元素

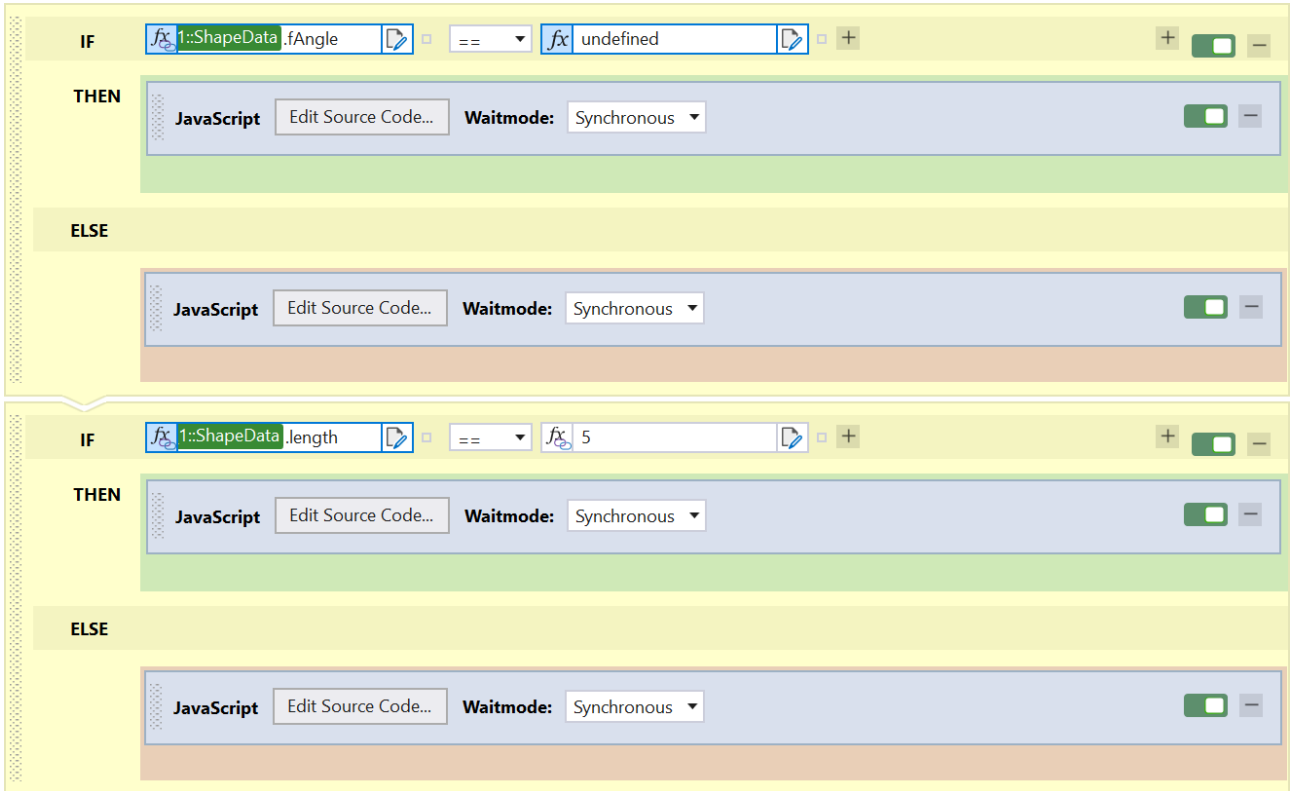
为了支持在图像中测量距离或面积的用例，Infobar（信息栏） [▶ 2679]提供了一个元素，用于显示代表当前绘制形状的文本。默认情况下，“Point（点）”形状类型会显示坐标，“Line（线）”形状类型会显示线长，“Polygon（多边形）”形状类型会显示点数，“Square（正方形）”、“Rectangle（矩形）”、“Circle（圆形）”和“Ellipse（椭圆形）”形状类型会显示面积。如果需要显示更多信息或其他信息，可以使用单独的 ShapeValueFormatting 功能 [▶ 2680]来实现。

查询形状类型和数据值

根据所选的形状类型，ShapeData 中的数据可以有不同的格式。要确定存在哪种格式，可以查询当前的 ShapeType，也可以通过数据值来确定，例如是否定义了 fAngle 属性。如果是这样，就会出现一个 TcVnRotatedRectangle 类型，否则就是一个点数组。要查询数组中包含多少个点，可以查询长度。然后便可以用这个数字来确定是一个点、一条线还是一个多边形，有多少个点。

此外，还可以通过查询各个值来检查是否存在有效数据。删除或重置形状数据时，单个值会写入 0，表示无效值。

这些查询可用于确保各种形状类型分别写入相同类型的指定 PLC 变量，并确保只有有效数据才能传输到控制程序。



事件：onShapeChanged und onShapeConfirmed

onShapeChanged 和 onShapeConfirmed 事件均可用于通知您形状的变化。

一旦 ShapeData 发生变化，就会触发 onShapeChanged 事件。在以编程方式设置 ShapeData 时、调用 clearShape() 函数法时以及图形交互过程中，都可能发生变化。该事件用于跟踪 ShapeData 实时值的变化，前提是将 ShapeDataUpdate 设置为 onChanged 值。

当以编程方式设置形状或完成与形状的图形交互后，会触发 onShapeConfirmed 事件。该事件会在 ShapeData 有效的条件下触发。

形状的高级使用

Shape（形状）功能是通过集成到 Image Watch（图像查看）控制器中的单独控制实现的。这些形状控制器具有一些 Image Watch（图像查看）控制器无法直接提供的属性。例如，其中包括矩形的精确长宽比。这些属性可以通过编程或单独使用形状控制器来访问。

7.2.1.7 缩略图

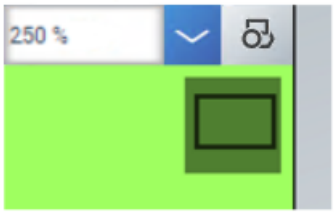
本页解释的属性和函数载于 **Thumbnail（缩略图）**类别。

缩略图表示在控制器中可以看到的图像部分。与图像的 **View（视图）**密切相关（请参见 **View（视图）**类别中的只读属性 `View`），但在某些方面有所不同。

缩略图是否可见由 `ThumbnailVisbile` 属性决定。其一般位置和大小由属性 `ThumbnailPosition` 和 `ThumbnailSize` 决定，并且始终是正方形。如果图像的长宽比不是 1:1，那么缩略图的尺寸就是图像的全宽，而高度则会根据长宽比进行调整。

当整个图像框都被图像覆盖时，缩略图显示与只读属性 `View` 几乎相同。否则，它的不同之处在于图像之外的区域会包含在 `View` 中，但不会包含在缩略图中。

Colors（颜色）类别中的属性 `ThumbnailBackgroundColor` 和 `ThumbnailStrokeColor` 可用于调整背景颜色和缩略图标记的颜色。



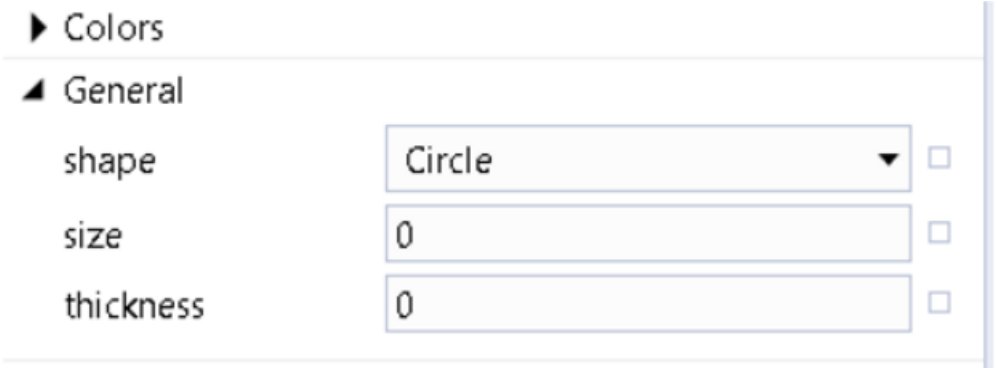
7.2.1.8 覆盖物

本页解释的属性和函数载于 **Overlay（叠加）**类别。

例如，叠加可用于帮助设置相机或定位图像中的对象。

预定义几何形状

控制器提供可以在图像中心显示的预定义几何对象（如圆形、矩形、十字形、框架）。例如，这些形状可用作十字或图像中特定位置的框架和标记。还可以同时显示多个对象（例如一个圆和一个十字光标）。这通过数组属性 `OverlayElements` 实现。对象配置如下所示：



对象在图像中始终居中。它们具有由**尺寸**定义的固定比例，不会随图像缩放。

叠加图像

如果这种功能对于特定应用来说不够灵活，可以使用 `OverlaySource` 属性指定叠加图像。显示的位置和大小与主图像完全相同，也就是说，叠加图像会被拉伸/缩放以适应主图像。因此，我们建议叠加图像的长宽比至少与主图像相同，大小最好也与主图像相同，并且除相关部分外均透明。

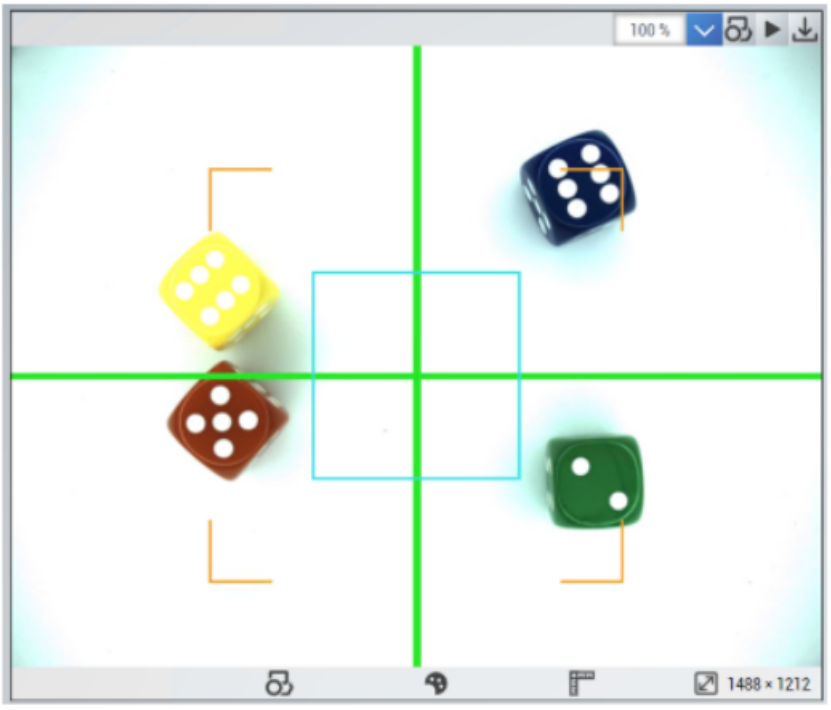
例如，您可以在透明图像上绘制一个带有附加附件的用户定义十字光标，并将其用作叠加。要创建透明图像，必须在背景颜色的第四个通道处创建颜色值为 0 的四通道图像。然后，要画入的光标线在第四通道处获得颜色值 255，这样这些元素就可以覆盖背景图像。

视图

例如，叠加了矩形和文本的图像可能会产生以下结果：



在下方示例中，来自 PLC 的叠加图像（绿线）与控制器中定义的两个对象相结合：



请注意，在传输带有“TwinCAT.HMI.Vision”服务器扩展包的透明图像时，必须使用 PNG 或 BMP 图像格式，因为 JPEG 和 Stream 不支持带有 alpha 通道的透明图像内容。

7.2.1.9 信息

本页解释的属性和函数载于 **Info（信息）**类别中。

您可以使用以下属性查询所显示图像或该图像中特定像素的信息。

ImageSize	显示图像的原始尺寸。这与 PLC 中的图像尺寸相对应。

ImageDisplaySize	图像的显示尺寸（单位：像素）。
PixelColor	特定选定像素的颜色（详见下文）。颜色是由四个元素组成的数组，分别代表颜色通道 [R、G、B、A]。
PixelPosition	特定选定像素的位置（详见下文）。位置指定为一个数组，包含两个元素 [X, Y]，以原始图像坐标表示，而不是以显示坐标表示。

PixelColor 和 **PixelPosition** 是为特定像素计算的。通过点击、触摸或光标在显示图像上的位置来选择该像素。

属性 **PixelInfoUpdate** 决定是否执行计算，或由哪个事件触发计算。

从不	**PixelColor** 和 **PixelPosition** 完全不会更新。
AtClick	当用户点击或触摸特定像素时，会更新 **PixelColor** 和 **PixelPosition**。当用户点击或触摸当前图像外的图像框上的某一点时，这些值会被重置。
AtCursor	当用户在图像上移动鼠标指针时，**PixelColor** 和 **PixelPosition** 会不断更新。当鼠标指针不在图像上时，这些值会被重置。

每次更新像素颜色和位置时，都会触发 **onPixelInfo** 事件。

显示屏

当前的 PixelColor、PixelPosition 和 ImageSize 信息可以显示在信息栏中（请参见 **Infobar（信息栏）**类别中的属性 `InfobarItems`）。



7.2.1.10 历史

本页解释的属性和函数载于 **History（历史）**类别中。

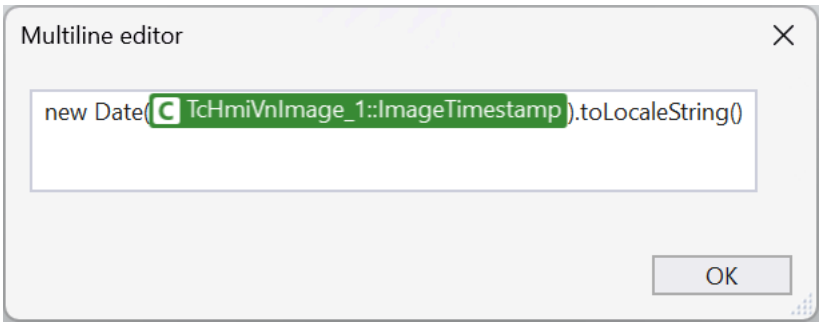
Image Watch（图像查看）控制器提供了临时保存一定数量接收到的图像的选项。要使用历史功能并能够选择并显示这些图像，必须首先激活 `ImageFreeze` 属性作为前提条件。

属性：`QueueSize`

`QueueSize` 属性限定了最近接收到的图像需要保存的数量，从而也相应地确定了历史队列的大小。通过 `0` 可以停用图像历史功能。如果队列已满且又接收到一个新图像，则根据 FIFO（先进先出）原则删除最旧的图像。要访问历史记录，必须激活 `ImageFreeze`。然后，您可以使用 `QueueIndex` 来确定应显示哪个图像。`0` 条目始终与队列顶部且最后接收到的最新图像相对应。

属性：`ImageTimestamp`（只读）

当前所选历史图像的时间戳可以通过 `ImageTimestamp` 属性以毫秒格式读出。要对其进行进一步处理（例如用于显示），可以使用 JavaScript `Date` 类，如下例所示：




```
new Date(/**TimestampSymbol**/).toLocaleString()
// output: "4/8/2025, 1:45:13 PM"
```

函数：clearHistory

clearHistory 函数可删除整个历史队列。

临时数据存储

历史队列中的图像数据是临时存储的，当相应的 ImageWatch-Control 实例被释放时，这些数据将被删除。例如，在离开页面或更改内容时，就会发生这种情况。

性能

在配置 QueueSize 时，应特别注意图像大小。这取决于图像分辨率、通道数量以及传输格式。转换后的图像大小可以从服务器扩展 [▶ 2660]中的统计数据获取。对于 BMP 传输格式，图像数据未经压缩，因此数据量较大。在与较大的 QueueSize 连接时，这可能会导致浏览器出现“内存溢出异常”。因此，我们建议不要将 QueueSize 选得过大，并建议使用压缩后的 JPEG 或 PNG 传输格式。

7.2.1.11 统计信息

本页解释的属性和函数载于 **Statistic（统计）**类别中。

Image Watch Control（图像查看控制器）提供了记录图像渲染时长统计数据的选项。要获取关于图像大小以及通过 ADS 进行图像传输的更多统计信息，可以读取 TcHmiVision [▶ 2660] 人机界面扩展的统计信息。

属性：NumberOfImageTransmissions

要分析浏览器处理接收到的图像所需的耗时，可以使用 NumberOfImageTransmissions 属性指定应将最近接收到的多少个纳入统计数据中。通过 0 可以停用统计功能。

属性：ImageRenderTime（只读）

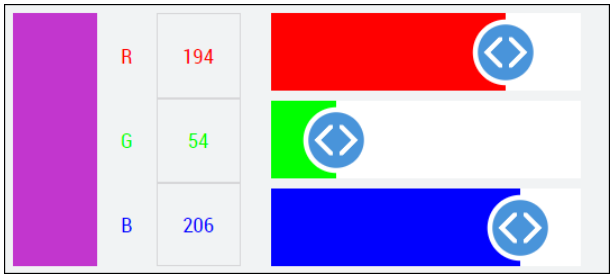
图像在浏览器中渲染所需的平均时间可以通过 ImageRenderTime 以毫秒为单位读出。

函数：resetStatistic

resetStatistic 函数可重置统计数据。

7.2.2 颜色

Color Control（颜色控制器）TcHmiVnColor 会显示颜色，并在必要时对颜色进行调整。其中包含各种表示颜色值的显示元素，根据配置的不同，其也可用于设置颜色值。有关组件和设置的概述，请参见 显示元素 [▶ 2691] 分章。每个配置选项的说明可在本页的以下类别中找到。每个类别都有分章，提供更多信息、细节和应用示例。



对于以下所有属性，控件有一个 getter 和一个 setter 方法。例如，属性 Orientation 存在两个方法：getOrientation 和 setOrientation。属性的使用将在链接分章中进行更详细的说明。

Controls（控件）基类所提供的所有其他属性、事件和权限均基于 TwinCAT HMI 文档。

类别：常用

以下属性表示控制器颜色值的数据接口 [▶ 2693]。它一方面用于设置控制器的显示颜色，另一方面用于检索用户输入的颜色值。

属性	类型	描述
ColorSymbol	ColorSymbol [▶ 2720]	包含当前颜色值的 HMI 符号。
ColorChannels	TcHmiVnColor.ColorChannels [▶ 2720]	定义颜色的格式。其中包括： • 通道数量 • 显示颜色通道 • 通道的最大值 • 整数或浮点数 • 通道的标签
BoxColorConversion	Function	可将相应的颜色值转换为可由浏览器解释的字符串的函数，反之亦然。如果每个颜色通道与默认的 RGB 格式不同，则需要告诉控制器每个颜色通道的含义。
ColorValue (只读)	ColorValue [▶ 2720]	颜色值。这可以在没有符号被链接为 ColorSymbol 的情况下检索。

ColorValue 属性始终与映射的 PLC 变量的数据类型相适应。默认为四通道数组 (TcVnVector4_LREAL)，也可在不进行映射的情况下使用。有关更多选项，请参见 与 PLC 连接 [▶ 2693]。

函数	描述
setColorValue(colorValue: ColorValue [▶ 2720]): void	设置颜色值。可以在没有符号被链接为 ColorSymbol 的情况下调用。

类别：用途

以下属性指定了控制器的行为。

属性	类型	描述
BoxUsage	UsageMode [▶ 2721]	确定是否显示颜色框以及其是否可以编辑。
ValuesUsage	UsageMode [▶ 2721]	确定是否显示单个颜色值以及是否可对它们进行编辑。
SliderUsage	UsageMode [▶ 2721]	确定是否以滑块的形式显示单个颜色值以及是否可对它们进行编辑。
SliderValueUpdate	DataUpdate	指定是在 onChanged 期间更新颜色符号，还是在与滑块交互 onConfirmed 结束时更新颜色符号。

类别：外观

以下属性决定了控制器的外观 [▶ 2691]。

属性	类型	描述
Orientation	定向 [▶ 2720]	控制器的定向；类似于 LinearGauge 控制器的定向。Horizontal 或 Vertical。
BoxSize	Number (px)	根据定向（定向值）确定色框的宽度或高度。

属性	类型	描述
LabelsSize	Number (px)	根据朝向（朝向值）确定显示颜色名称的宽度或高度。
ValuesSize	Number (px)	根据定向（定向值）确定显示颜色值的宽度或高度。

事件

所有事件均列在“Vision”类别中。

事件	描述
.onValueChanged	一旦 ColorValue 属性发生变化，就会触发该事件。

主题资源

如为 Themes（主题）设置，可以使用以下属性来更改子控件（颜色值的文本框和通道名称）的外观。关于各个属性的详细说明，请参见标准 HMI 文档。

属性	类型
TextFontFamily	FontFamily
TextFontSize	MeasurementValue
TextFontSizeUnit	MeasurementValue
TextFontStyle	FontStyle
TextFontWeight	FontWeight
TextPadding	Padding

7.2.2.1 显示元素

元素

颜色控制器中可以显示以下元素。在每种情况下，相应的使用属性（BoxUsage, ValuesUsage und SliderUsage）决定元素是隐藏（Hide）还是显示（ViewOnly）以及用户是否可以借助元素（Editable）改变颜色。



名称	描述
颜色框	这是一个显示当前颜色的矩形区域。如果 BoxUsage 属性设置为“可编辑”，则点击 ColorBox 会打开用于选择颜色的浏览器特定对话框。根据浏览器的不同，这可能看起来像一个单独的窗口或叠加，也可能什么都不显示。 由于每个浏览器都控制着对话框本身的外观和功能，因此我们对此没有任何影响。

名称	描述
	例如，颜色吸管会对屏幕进行截图，因此它无法处理不断变化的图像。更多信息载于 不同浏览器中的颜色框 [▶ 2692] 章节。
ColorLabels	该区域包含各个颜色通道的标签。这些数值有多少个，以及它们的文字颜色，都you由 **ColorChannels** 属性决定。
颜色值	该区域包括用于查看和编辑控制器中存储颜色值的文本字段。这些数值有多少个，以及它们的文字颜色，都you由 **ColorChannels** 属性决定。
颜色滑块	该区域包括与 **ColorValues** 相对应的线性滑块。

排列

显示元素的排列取决于 `Orientation` 属性。“水平”排列的元素彼此相邻，“垂直”排列的元素彼此相叠。根据朝向的不同，`BoxSize` 和 `ValuesSize` 属性定义相应元素的宽度或高度。滑块的宽度或高度会自动与可用空间的其余部分相对应。

注意

小尺寸

如果所选定向上的控制器非常小，则可能无法完整且正确地显示所有元素。这主要影响滑块。超过一定尺寸时，JavaScript 控制台中会显示警告，说明尺寸可能会影响控制器的功能。

编号

显示颜色值和滑块的数量始终与设定颜色格式 [▶ 2695]的通道数量对应。颜色格式通过 `ColorChannels` 属性设置。

文本

可以使用主题编辑器中的主题资源通过各种字体属性自定义颜色值的文本框。

7.2.2.1.1 不同浏览器中的颜色框

如果 `BoxUsage` 属性设置为 `Editable`，则可以通过点击颜色框打开另一个颜色选择对话框。此对话框取决于浏览器。以下是一些最常用浏览器的对话框：

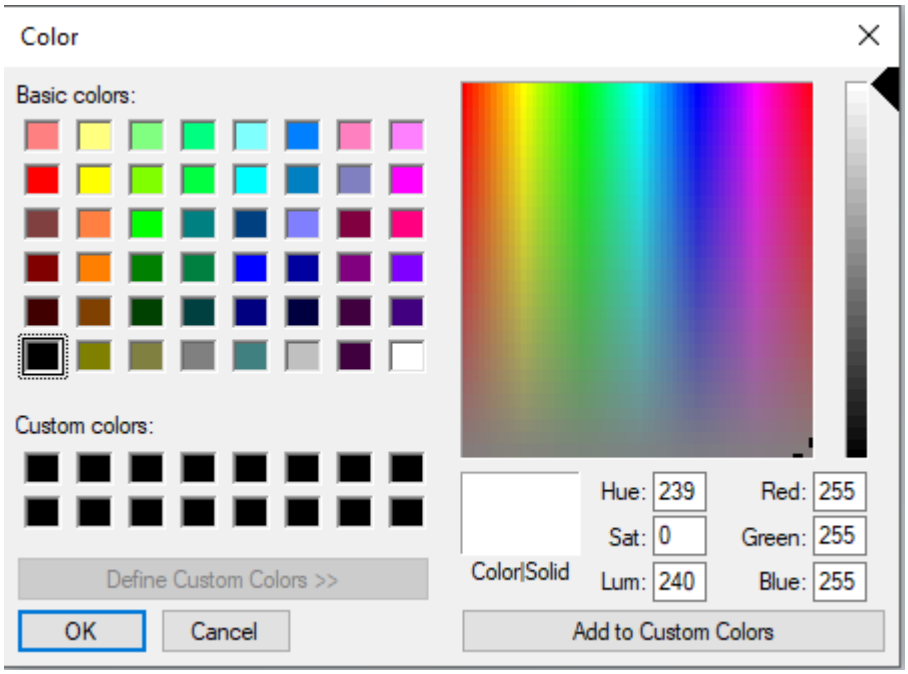
Google™ Chrome 和 Microsoft Edge



取色器

基于 Chromium™ 的浏览器中的取色器会在选择工具时对浏览器窗口进行截图。然后，所选颜色就来自该截图。这意味着当图像发生变化时，所选颜色将不是点击时显示的颜色。因此，建议在使用吸管功能前通过 `ImageFreeze` 属性停止滚动。

Mozilla Firefox



7.2.2.2 颜色数据

`ColorSymbol` 属性可与不同的符号绑定。为避免输出颜色值与符号的数据类型不一致，Color Control（颜色控制器）会读取符号的类型，并尝试自动推导出正确的返回类型。

例如，如果要显示三个颜色通道，但在 PLC 中链接了一个类型为 `TcVnVector4_LREAL` 的数组，那么这些类型就不匹配。Color Control（颜色控制器）会自动识别，并将一个带有四个值（而不是三个值）的数组写入符号中，这样便不会出错。通道的其余部分留空。

只读 `ColorValue` 属性用于在无需将符号绑定到 `ColorSymbol` 属性的情况下以编程方式访问颜色。除此之外，没有任何区别。两个版本可以同时使用。请注意，`ColorValue` 也会输出绑定到 `ColorSymbol` 的符号数据类型。

`SliderValueUpdate` 属性用于指定是在 `onChanged` 期间还是在与滑块的交互 `onConfirmed` 结束时更新颜色符号。

与 `ColorValue` 属性类似，`SetColorValue` 函数提供了一种可设置该控件颜色值的方法，而无需将符号与 `ColorSymbol` 属性相关联。

下方示例会解释访问颜色值的两种方法：

- 与 PLC 关联 [▶ 2693]
- 计划使用 [▶ 2694]

转换为 HMI 颜色对象 [▶ 2694] 示例展示了如何将颜色数据用于其他场景。

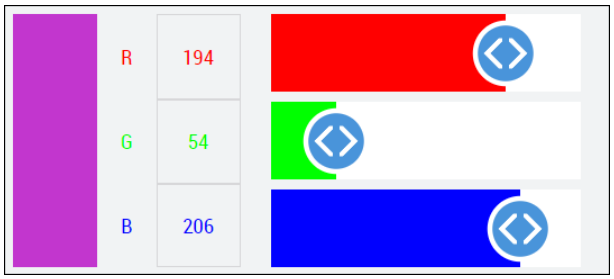
7.2.2.2.1 与 PLC 连接

在最简单的情况下，Color Control（颜色控制器）可用来表示来自 PLC 的颜色变量。例如，为此可将 `TcVnVector4_LREAL` 类型的 PLC 变量与 `ColorSymbol` 属性相关联。此外，还支持所有 `Vector3` 和 `Vector4` 数据类型以及 `INT`, `DINT`, `SINT`, `UINT`, `USINT` 和 `LREAL`。



现在，PLC 变量和控制器彼此同步：

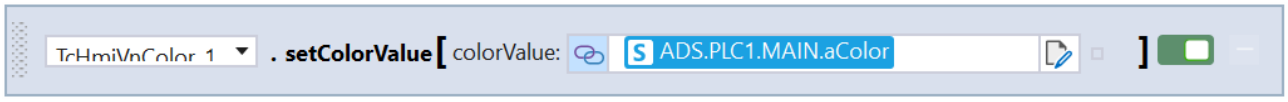
Expression	Type	Value
aColor	TcVnVector4_LREAL	
aColor[0]	LREAL	194
aColor[1]	LREAL	54
aColor[2]	LREAL	206
aColor[3]	LREAL	0



PLC 数组的第四个元素会被忽略，因为默认情况下只有三个通道在控制器中显示为 RGB。可以通过颜色格式 [▶ 2695]对此进行设置。

7.2.2.2.2 编程使用

如果不将 Color Control（颜色控制器）的颜色值直接与符号关联，则可以通过编程方式访问。要在控制器中设置颜色，可通过 JavaScript 代码或 **Actions and conditions（操作和条件）**对话框使用 `setColorValue` 函数：




```
TcHmi.Controls.get("Background").setColorValue([0, 0, 255])
```

要查询控件的颜色，请使用只读 `ColorValue` 属性：



在 JavaScript 中，必须使用 `getColorValue` 函数来检索属性：

```
const color = TcHmi.Controls.get("Background").getColorValue()
```

7.2.2.2.3 转换为 HMI 颜色对象

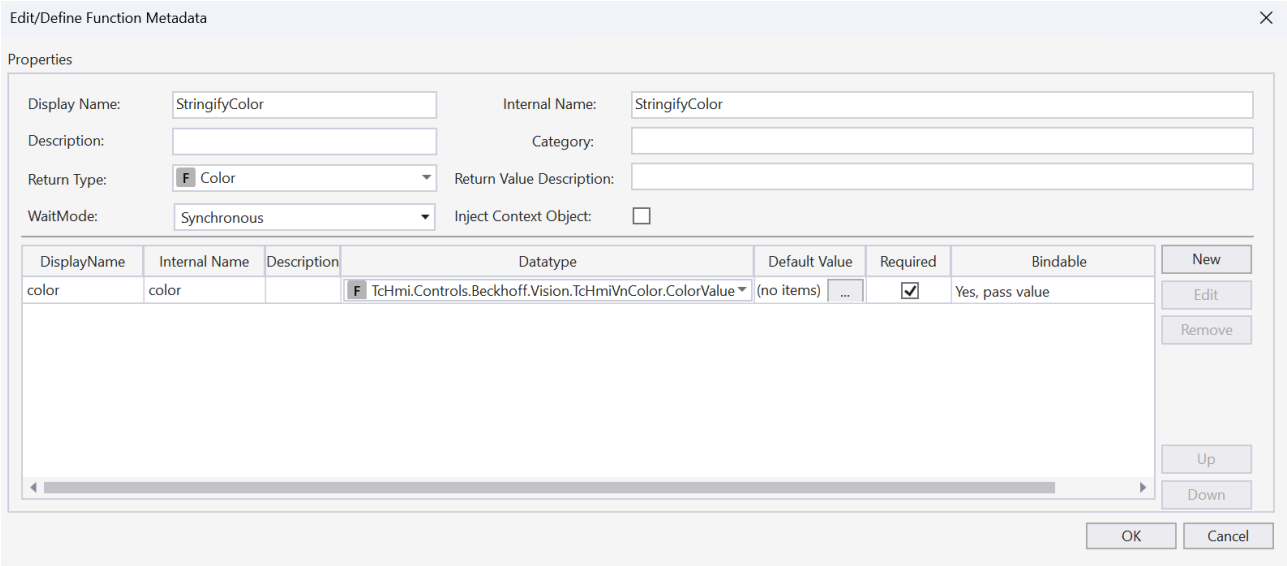
要在 TwinCAT HMI 中使用一种颜色，例如作为背景或用于文本或框架等其他属性，必须将其转换为 TcHmi.Color 类型的对象。下方示例展示了如何创建这样一个函数，以及它必须具备哪些接口。



在 HMI 文档中会说明各函数的一般使用方法。

创建 TypeScript 函数后，必须创建一个参数。例如，这会从图像控制的 PixelColor 属性或颜色控制的 ColorValue 中获取颜色值，因此必须创建 `Array` 类型。需要注意的是，由于时序原因，传递的参数可能与函数执行时相应控制属性的值不同。

函数必须返回一个 `Color` 类型的值，该值然后可用于 HMI 中的其他属性。



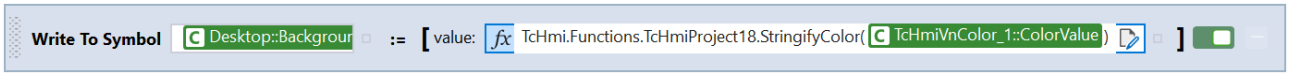
以下代码示例显示 RGB 颜色值的转换：

```
namespace TcHmi {
    export namespace Functions {
        export namespace TcHmiProject {
            export function StringifyColor(color: any){
                if (color.length >= 3){
                    return { color: `rgb(${color[0]}, ${color[1]}, ${color[2]})` };
                }
                throw new Error("Could not stringify color.");
            }
        }
    }
}
TcHmi.Functions.registerFunctionEx('StringifyColor', 'TcHmi.Functions.TcHmiProject',
TcHmi.Functions.TcHmiProject.StringifyColor);
```

返回值必须是 `Color` 类型且结构如下，才能作为颜色对象被接受：

```
{
    color: 'rgb(<red>, <green>, <blue>)'
}
```

例如，可以使用这样的函数转换 TcHmiVnColor 控制器的颜色值，然后将其设置为另一个用户界面元素的背景颜色。



7.2.2.3 颜色格式

颜色表示方法有很多种。诸如单色（一通道灰度值）、RGB 颜色空间（三通道颜色值）、HSV / HLS 颜色空间（三通道颜色值）等最常见格式将在下文详细介绍。不过，该控制器提供了许多配置选项，因此也可以设置和使用特殊颜色格式。

设置颜色格式

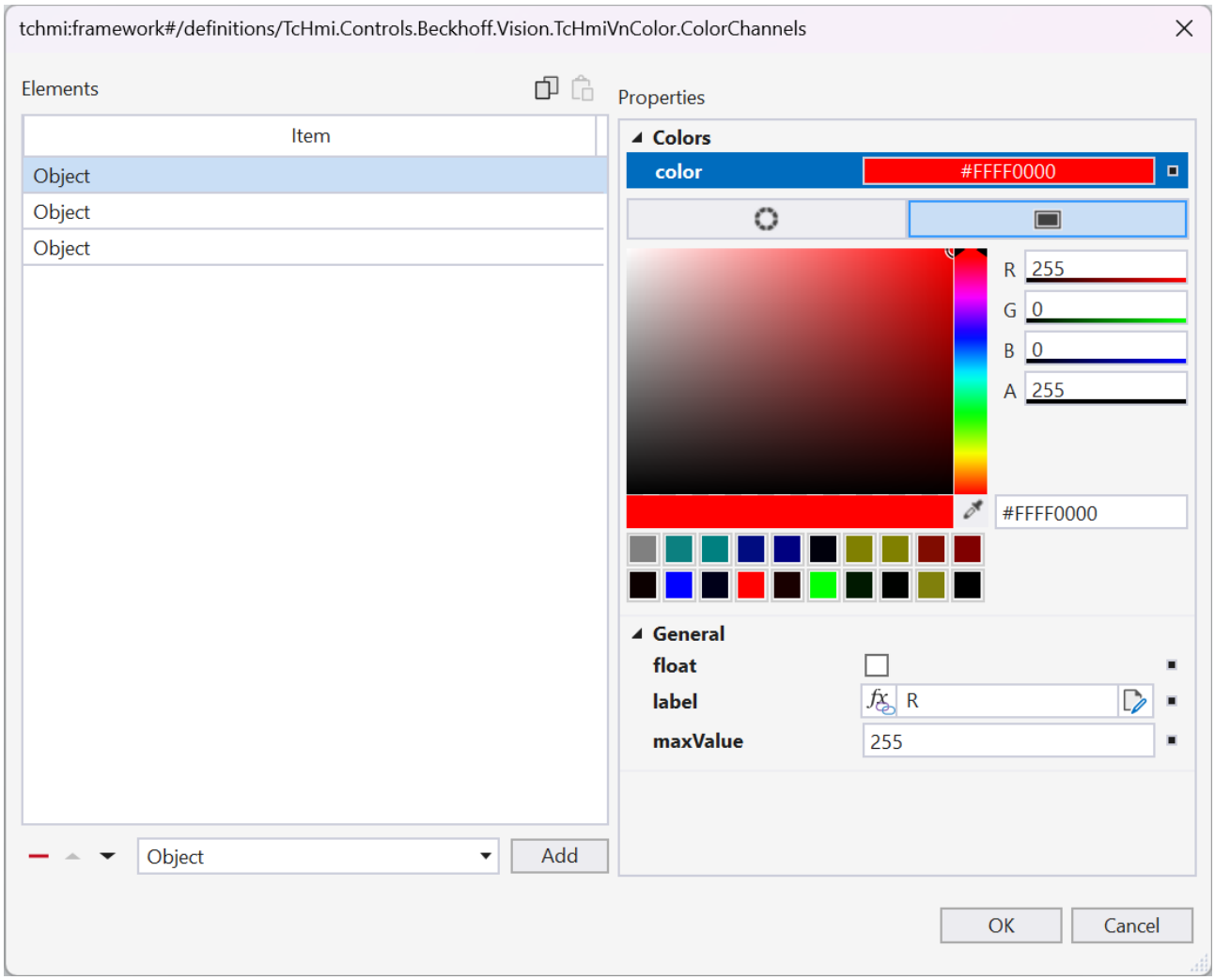
为了使控制器适应任何颜色格式，可以详细设置一般条件：

- 通道数量（对象）
- 显示每个通道的颜色。这将用作相应滑块的颜色和显示的颜色值。
- 每个通道的数据类型（浮点数或整数）
- 每个通道的标签
- 每个通道的最大值



如果所有颜色通道的标签属性均为空，则颜色名称将被隐藏，其他元素之间d的间距也会相应缩短。

可通过 `ColorChannels` 属性的对话框进行调整。RGB 格式为默认预设值。例如，要设置单色八格式，必须先删除现有三个通道（对象）中的两个。之后，可以将 R、G 和 B 颜色通道中的颜色表示调整为 128 这样的平均灰度值。



针对以下颜色格式提供配置示例：

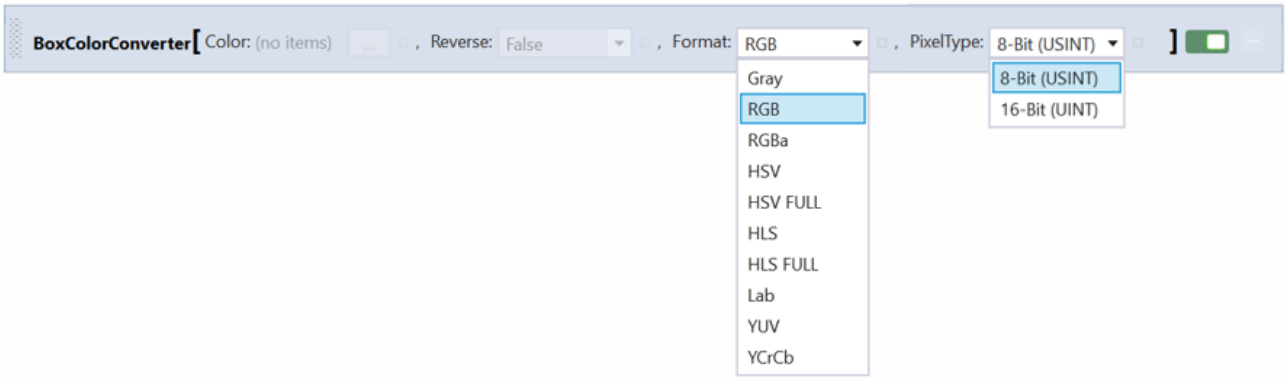
- RGB(a)（标准） [▶ 2697]
- HSV / HLS（值调整） [▶ 2698]
- 百分比 RGB（自定义格式） [▶ 2699]

颜色框转换

本节仅适用于使用颜色框且颜色不符合标准 RGB 格式的情况。

通过数字和滑块显示颜色值时，不需要对颜色格式有更深入的了解，因为只显示数值。不过，为了在颜色框中显示颜色，必须定义如何解释单个颜色通道及其值。例如，RGB 值的显示方式必须与 HSV 值不同。

由于浏览器特定颜色输入元素默认处理 RGB 颜色，因此所有其他颜色格式都必须首先转换为 RGB，才能在颜色框中显示。这种转换可通过 `BoxColorConversion` 属性进行设置。Gray（灰色）、RGBa、HSV、HSV FULL、HLS、HLS FULL、Luv、YUV 和 YCrCb 格式以及 8 位和 16 位数据类型的转换函数已经存在。



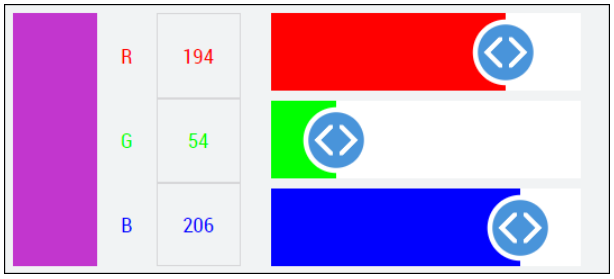
通用颜色转换器

除了这种针对颜色框的特殊转换函数外，还有一个通用颜色转换器，其具有同一套颜色格式，适用于 HMI 中的一般用途，也以 ConvertColor [▶ 2717] 的形式提供。

对于完全不同的颜色格式或具有不同缩放比例的颜色格式，可以创建用户定义函数 [▶ 2699]。

7.2.2.3.1 标准 RGB(a) 颜色格式

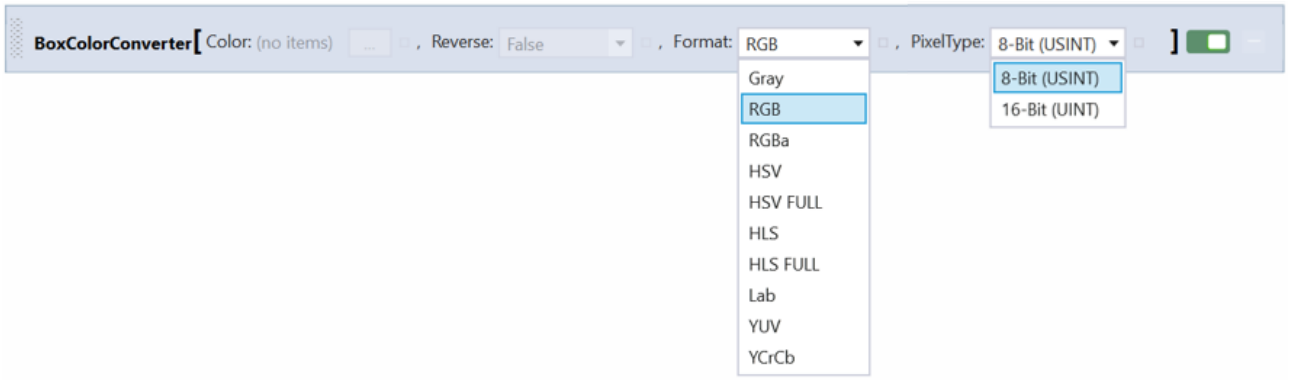
默认情况下，定义三个通道，所有通道均配置为最大值为 255 的整数，且显示颜色对应于 RGB 颜色格式。



`ColorChannels` 属性的值可通过脚本如下写入：

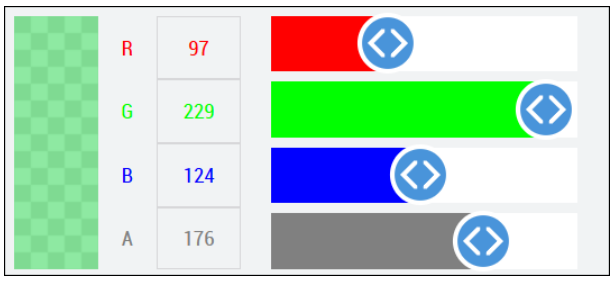
```
[
  {
    "color": {
      "color": "rgba(255, 0, 0, 1)"
    },
    "float": false,
    "label": "R",
    "maxValue": 255.0
  },
  {
    "color": {
      "color": "rgba(0, 255, 0, 1)"
    },
    "float": false,
    "label": "G",
    "maxValue": 255.0
  },
  {
    "color": {
      "color": "rgba(0, 0, 255, 1)"
    },
    "float": false,
    "label": "B",
    "maxValue": 255.0
  }
]
```

`BoxColorConversion` 属性设置为供给 `BoxColorConverter` 函数，配置格式为“RGB”，像素类型为“8 位 (USINT)”。



RGBa 颜色格式

为了能够显示透明的颜色，必须在 `ColorChannels` 属性中添加额外的颜色通道。建议使用显示颜色“rgba(128, 128, 128, 1)”进行配置。由于 `BoxColorConverter` 已有 RGBa 转换功能，因此必须选择 RGBa 作为颜色格式。

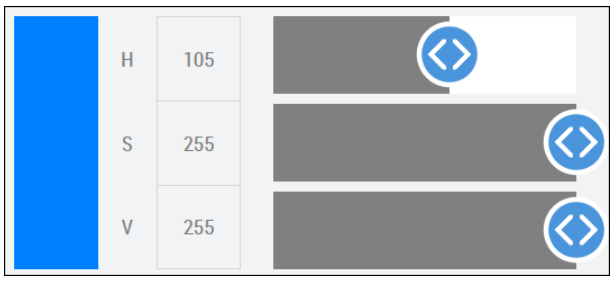


7.2.2.3.2 HSV/ HLS (FULL) 颜色格式

如果要使用 Color Control（颜色控制器）以 HSV 格式显示颜色，则需要进行一些调整。“Hue（色调）”通道的数值范围仅定义为 0 至 180，因此第一个通道的 `maxValue` 必须减小到 180。其他两个通道的刻度范围为 0 至 255。由于无法将 HSV 格式的各个颜色通道分配给特定的颜色，因此应将显示滑块和数值等的颜色设置为统一的灰度值，这样就不会产生混淆。

HLS 颜色空间可以使用相同的设置和数值范围，因为只有 L 和 S 通道的计算方法不同。

使用“HSV FULL”和“HLS FULL”颜色格式时，所有颜色通道的定义范围为 0 至 255。

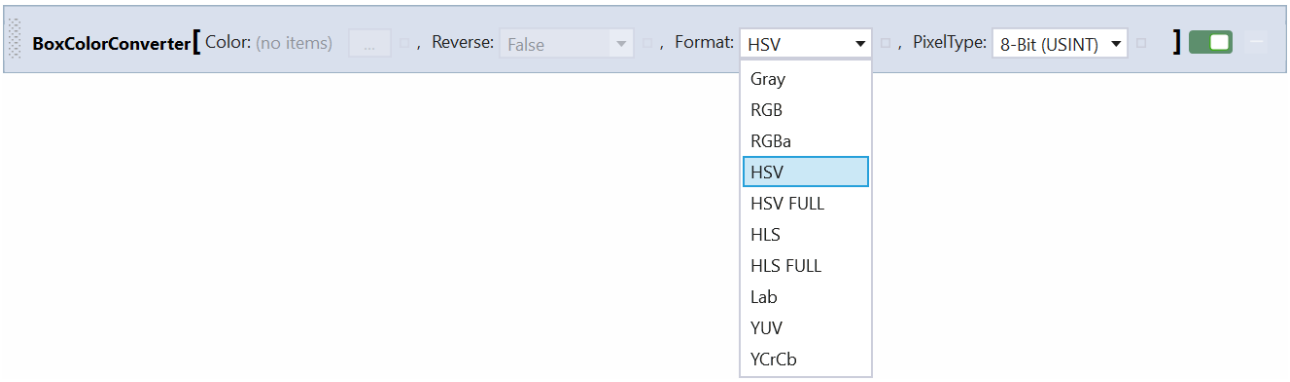


`ColorChannels` 属性的值可通过脚本如下写入：

```
[
  {
    "color": {
      "color": "rgba(90, 90, 90, 1)"
    },
    "float": false,
    "label": "H",
    "maxValue": 180.0
  },
  {
    "color": {
      "color": "rgba(90, 90, 90, 1)"
    },
    "float": false,
    "label": "S",
    "maxValue": 255.0
  },
  {
```

```
    "color": {
      "color": "rgba(90, 90, 90, 1)"
    },
    "float": false,
    "label": "V",
    "maxValue": 255.0
  }
]
```

最后，必须调整 `BoxColorConversion` 属性的颜色框的转换。为此，必须在 `BoxColorConverter` 函数中选择相应的颜色格式：



7.2.2.3.3 用户定义的颜色格式

作为 Color Control（颜色控制器）配置的另一个示例，考虑使用范围在 0 和 1 之间的值的 RGB 颜色格式。要在“Color Control（颜色控制器）”中这样显示，必须在 `ColorChannel` 属性中将通道设置为 `float`，且 `maxValue` 为 `1`。



`ColorChannels` 属性的值可通过脚本如下写入：

```
[
  {
    "color": {
      "color": "rgba(255, 0, 0, 1)"
    },
    "float": true,
    "label": "R",
    "maxValue": 1.0
  },
  {
    "color": {
      "color": "rgba(0, 255, 0, 1)"
    },
    "float": true,
    "label": "G"
    "maxValue": 1.0
  },
  {
    "color": {
      "color": "rgba(0, 0, 255, 1)"
    },
    "float": true,
    "label": "B",
    "maxValue": 1.0
  }
]
```

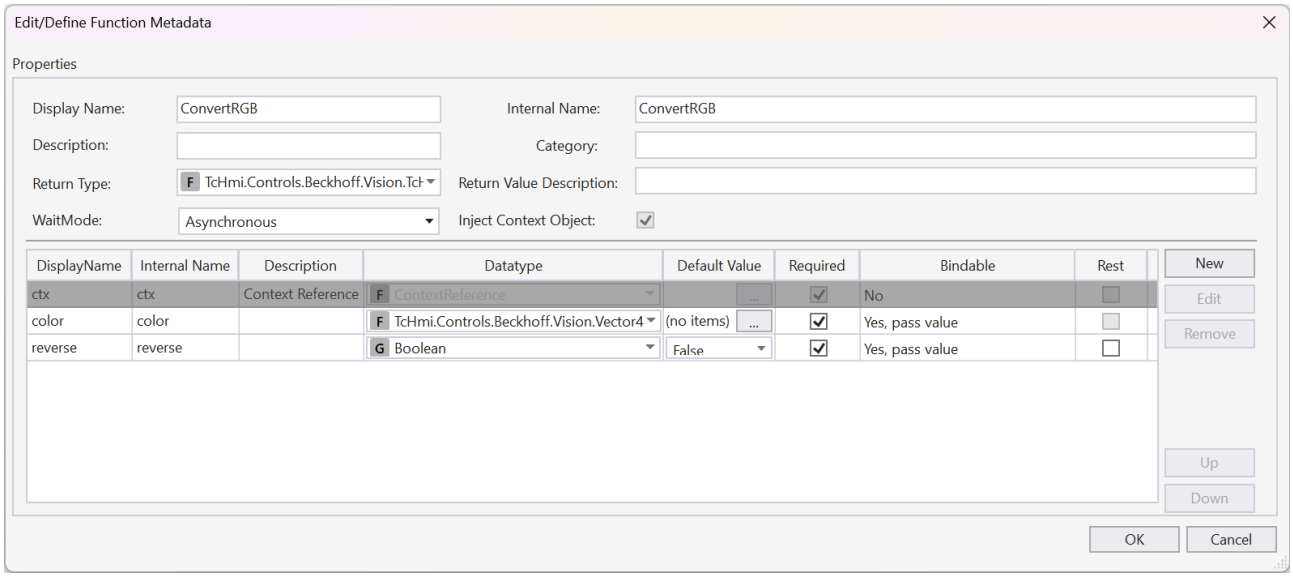
ColorBoxFormatting 函数

为了使颜色框在值范围改变后仍能显示正确的颜色，必须调整 `BoxColorConversion` 属性。与标准转换不同，没有适用于此的现成函数。相反，有必要创建一个单独的函数。下方示例展示了如何为上述用例创建这样一个函数，以及它必须具备哪些接口。

在 HMI 文档中会说明各函数的一般使用方法。

创建 TypeScript 函数后，必须将 WaitMode 更改为 Asynchronous（异步）。这样就创建了返回所需的参数“ctx”。然后必须创建两个参数。第一个参数获取要转换的颜色值，因此必须以 `TcHmi.Controls.Beckhoff.Vision.Vector4` 类型创建。第二个参数指定转换方向，必须创建为 `Boolean`。如为 `True`，值转换为 RGB，如为 `False`，则从 RGB 转换回用户定义的颜色格式。需要注意的是，由于时序原因，这些传递的参数可能与函数执行时相应控制属性的值不同。

该函数必须返回一个 `TcHmi.Controls.Beckhoff.Vision.TcHmiVnColor.ColorValue` 类型的值，然后该值显示在颜色框中。



在下方程序示例中，会调整颜色值的缩放。如果参数为 `reverse = false`，则颜色值会以用户定义的格式从控制器传输到函数中。然后必须将颜色格式转换为 RGB(a)，才能在 ColorBox 中显示。“颜色”参数中传输的颜色值始终为四通道。如果 ColorBox 中不显示透明颜色，则返回的颜色通道值必须为三通道。另外，第四个通道也可以设置为 255。

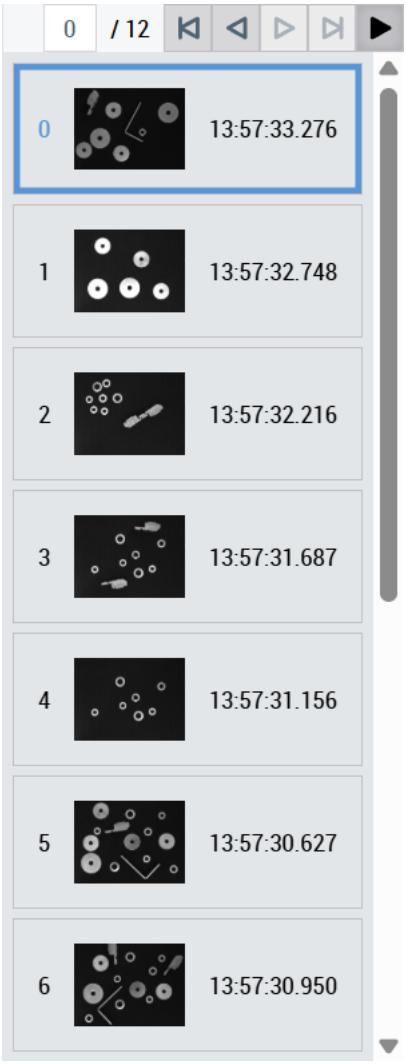
如果参数为 `reverse = true`，则颜色选择对话框中的颜色值会以 RGBa 格式传输到函数中。然后必须将颜色格式转换为用户定义的颜色格式。第四个通道取自控制器的颜色值，因为颜色选择对话框不支持透明度。

```
namespace TcHmi {
    export namespace Functions {
        export namespace TcHmiProject {
            export function ConvertRGB(ctx: any, color: any, reverse: any) {
                // Important: color has 4 channels. If the ColorBox should be RGB, return a 3
channel color or set the fourth channel (a) to 255
                if (!reverse) {
                    ctx.success(color.slice(0, 3).map((value: number) => value * 255));
                }
                else {
                    ctx.success(color.slice(0, 3).map((value: number) => value / 255));
                }
            }
        }
    }
}
TcHmi.Functions.registerFunctionEx('ConvertRGB', 'TcHmi.Functions.TcHmiProject',
TcHmi.Functions.TcHmiProject.ConvertRGB);
```

这可以确保颜色框显示正确的颜色，并且手动更改颜色框中的颜色可以获得正确的颜色值。

7.2.3 历史

`TcHmiVnHistory` 历史控件通过工具栏控制 ImageWatch [▶ 2666] 控件的历史功能，并提供历史图像的图形列表（图像列表）。每个配置选项的说明可在本页的以下类别中找到。每个类别都有分章，提供更多信息。该控件只能与 ImageWatch 控件合使用，并且仅提供历史功能的显示与操作。数据和相关配置 [▶ 2688]可在 ImageWatch 控件中找到。



对于以下所有属性，控件有一个 getter 和一个 setter 方法。例如，属性 `BarColor` 存在两个方法：`getBarColor(): SolidColor` 和 `setBarColor(value: SolidColor): void`。属性的使用将在链接分章中进行更详细的说明。

Controls（控件）基类所提供的所有其他属性、事件和权限均基于 TwinCAT HMI 文档。

类别：颜色

属性	类型	描述
`BarColor`	`SolidColor`	工具栏的颜色（请参见 **Toolbar（工具栏）**类别）。
`IndexColor`	`SolidColor`	列表中索引的颜色。
`IndexStrokeColor`	`SolidColor`	列表中索引边框的颜色。仅在 `FitImage` 已停用的情况下有效。
`IndexBackgroundColor`	`SolidColor`	列表中索引背景的颜色。
`TimestampColor`	`SolidColor`	列表中时间戳的颜色。
`TimestampStrokeColor`	`SolidColor`	列表中时间戳边框的颜色。仅在 `FitImage` 已停用的情况下有效。
`TimestampBackgroundColor`	`SolidColor`	列表中时间戳背景颜色。
`ElementBackgroundColor`	`SolidColor`	列表中元素背景颜色。
`ElementSelectionColor`	`SolidColor`	所选元素的高亮颜色。

类别：常用

以下属性可用于指定指向 ImageWatch 控件的链接。

属性	类型	描述
ImageWatchControl	TcHmiVnImage	指向要控制的 ImageWatch 控件实例的链接。

类别：工具栏

以下属性可用于自定义控件的工具栏 [▶ 2703]。

属性	类型	描述
ToolbarVisible	Boolean	工具栏显示。
QueueIndexVisible	Boolean	当 ImageFreeze 激活时，用于选择或显示指定历史图像的索引输入值。0 对应最后接收到的图像。
NavigationButtonsVisible	Boolean	当 ImageFreeze 激活时，用于在图像历史中向前或向后移动一个图像条目的按钮。
NavigationEndButtonsVisible	Boolean	当 ImageFreeze 激活时，用于跳转到历史中接收到的第一个或最后一个图像的按钮。
FreezeButtonVisible	Boolean	用于暂停已链接的 ImageWatch 的图像更新的按钮。历史功能仅在 ImageFreeze 已激活的情况下才能使用。
DownloadButtonVisible	Boolean	用于触发以原始尺寸对当前显示图像进行浏览器特定下载的按钮，请参见图像下载 [▶ 2674]。
ToolbarHeight	Number (px)	工具栏的高度。
QueueIndexWidth	Number (px)	QueueIndex 输入字段的宽度和条目总数。

类别：列表

以下属性可用于自定义控件的图像列表 [▶ 2703]。

属性	类型	描述
ImageListVisible	Boolean	图像列表的显示。
Orientation	Orientation	控件的方向，水平或垂直。
IndexVisible	Boolean	索引的显示。
ImageVisible	Boolean	图像的显示。
TimestampVisible	Boolean	时间戳的显示。
FitImage	Boolean	最大化图像尺寸，以便索引和时间戳在图像上显示。仅在 ImageVisible 已激活的情况下有效。
ImageSize	Number	在垂直视图中，它决定列表中的图像高度；在水平视图中，它决定列表中的图像宽度。仅在 ImageVisible 已激活且 FitImage 未激活的情况下有效。
TimestampFormat（时间戳格式）	函数	用于根据时间戳生成字符串并在列表中显示的函数。

属性	类型	描述
IndexFontFamily	FontFamily	列表中索引的字体。
IndexFontSize	MeasurementValue	列表中索引的字体大小。
IndexFontStyle	FontStyle	列表中索引的字体样式。
IndexFontWeight	FontWeight	列表中索引的字体粗细。
TimestampFontFamily	FontFamily	列表中时间戳的字体。
TimestampFontSize	MeasurementValue	列表中时间戳的字体大小。
TimestampFontStyle	FontStyle	列表中时间戳的字体样式。
TimestampFontWeight	FontWeight	列表中时间戳的字体粗细。

7.2.3.1 工具栏

本页介绍的属性和函数均可在 **Toolbar（工具栏）**类别中找到。

工具栏中的控制器可用于交互式执行各种操作，否则只能通过属性访问。每个元素都可以显示或隐藏。此外，整个工具栏的高度可以调整，也可以完全隐藏。

工具栏中元素的位置是固定的，无法自定义。如果仍有必要进行此类调整，您可以隐藏相应的元素，通过自己的控件自行实现这些调整，然后将其放置在 History Control（历史控件）工具栏上。

可用的工具栏元素从左到右依次为：



QueueIndex	当 ImageFreeze 激活时，用于选择或显示指定历史图像的索引输入值。0 对应最后接收到的图像。
NavigationButtons	当 ImageFreeze 激活时，用于在图像历史中向前或向后移动一个图像条目的按钮。
NavigationEndButtons	当 ImageFreeze 激活时，用于跳转到历史中接收到的第一个或最后一个图像的按钮。
FreezeButton	用于暂停已链接的 ImageWatch 的图像更新的按钮。历史功能仅在 ImageFreeze 已激活的情况下才能使用。
DownloadButton	用于触发以原始尺寸对当前显示图像进行浏览器特定下载的按钮，请参见图像下载 [▶ 2674]。

也可通过编程访问的函数

通过工具栏的界面元素可以访问的操作，也可以通过已链接的 ImageWatch 控件的属性以编程方式访问。

尺寸设置

工具栏提供了通过 ToolbarHeight 属性自定义自身高度以及通过 QueueIndexWidth 自定义索引输入字段宽度的选项。如果图像列表被隐藏，则工具栏的高度始终与控件的高度相同。所有按钮的宽度会根据工具栏的高度自动调整，使其呈正方形。

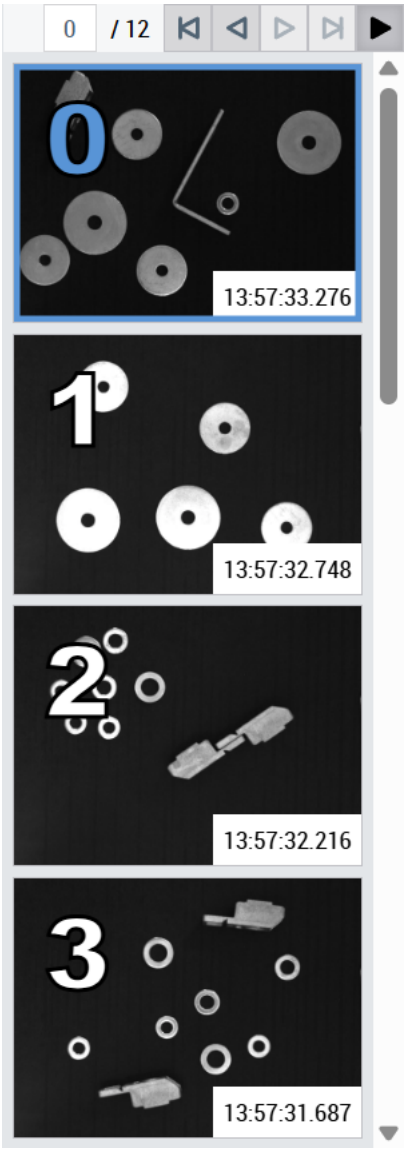
在内部，ToolbarHeight 属性的最大限值为 150 个像素。

背景颜色

工具栏的背景颜色默认为半透明白色 "rgba(255, 255, 255, 0.5)" 或 R：255、G：255、B：255、A：127，也可使用 **Colors（颜色）**类别中的 BarColor 属性进行自定义。

7.2.3.2 图像列表

本页解释的属性和函数载于 **List（列表）**类别中。



属性：ImageListVisible

`ImageListVisible` 属性可用于隐藏和显示图像列表。如果列表被隐藏，则工具栏将占据控件的整个区域。由于控件以纵向格式初始化，因此在这种情况下应调整高度。

图像列表

如果已链接的 ImageWatch 控件的 `ImageFreeze` 已激活，图像列表将显示所有临时保存的历史图像的条目。在列表中为每个图像显示了索引编号、相应的图像以及时间戳。时间戳是 ImageWatch 控件接收到相应图像的时刻。前提条件是 ImageWatch 控件的 `QueueSize` 必须大于 0。

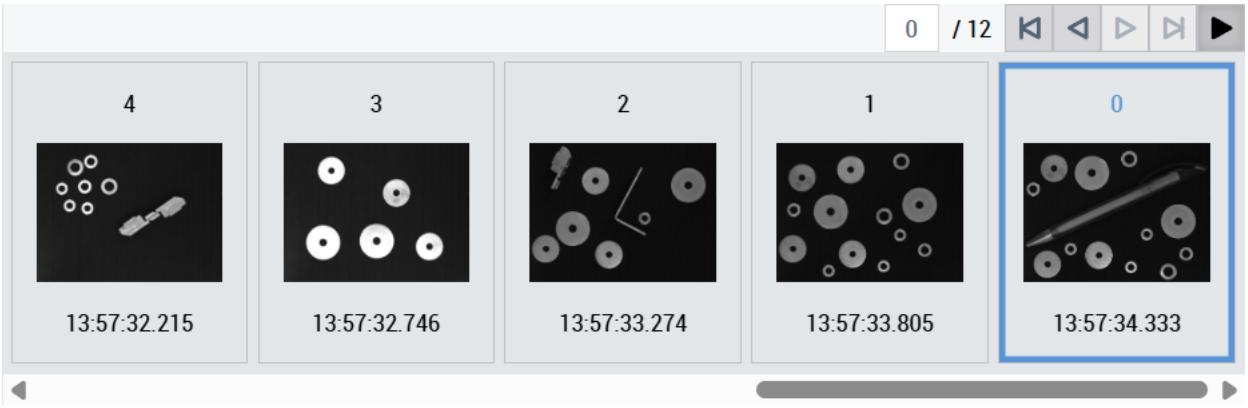
在以下情况下，将显示替代文本，而非图像列表：

- 如果没有链接 ImageWatch 控件：“请链接一个 ImageWatch 控件，以使用图像历史”
- 如果 ImageFreeze 未激活：“请激活冻结，以显示图像历史”
- 如果历史队列为空：“图像历史队列为空”

根据朝向不同，顶部或右侧的列表条目始终代表当前接收到的图像。点击条目，即可选择图像。在列表中所选的图像会高亮显示，并在已链接的 ImageWatch 控件中显示。

属性：Orientation

`Orientation` 属性用于定义图像列表中条目的排列方式。`vertikal` 表示从上到下，`horizontal` 表示从右到左。这意味着最后添加的索引为 0 的图像始终位于图像列表的顶部或右侧。

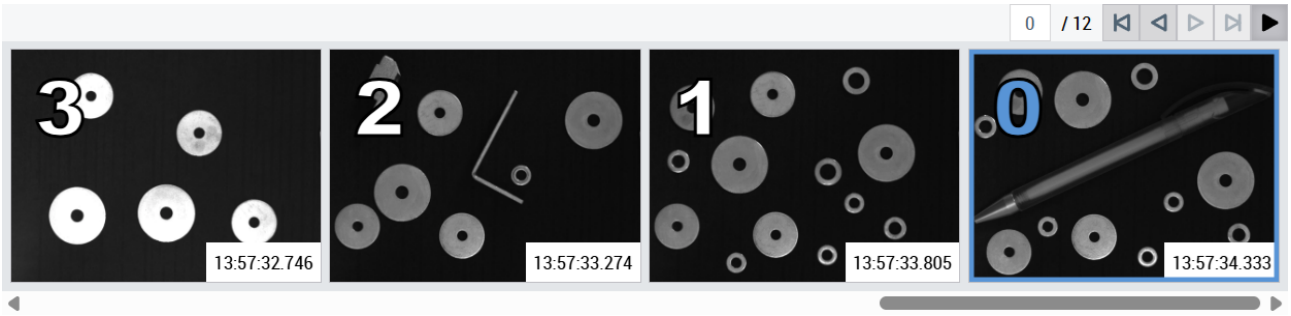


可见性

列表条目的内容可以根据需要显示或隐藏。为此使用属性 `IndexVisible`、`ImageVisible` 和 `TimestampVisible`。内容始终按照以下顺序显示：索引、图像、时间戳。如果列表朝向为垂直，则内容从左到右排列；如果朝向为水平，则内容从上到下排列。

属性：FitImage

如果属性 `FitImage` 已激活，则列表中的图像将以最大化方式显示。索引移动到图像的左上角，时间戳移动到图像的右下角。为确保两者清晰可见，索引使用了较大的字体并带有文本边框。另一方面，时间戳采用单色背景。



这种标准配置可以使用 Colors（颜色）类别中的属性 `IndexColor`、`IndexStrokeColor`、`IndexBackgroundColor`、`TimestampColor`、`TimestampStrokeColor` 和 `TimestampBackgroundColor` 进行自定义。例如，也可以为索引设置单色背景，为时间戳设置文本边框。或者，通过设置透明颜色，可以移除时间戳的单色背景以及索引的文本边框。

属性：ImageSize

使用 `ImageSize` 属性可以指定列表中图像的所需图像大小。对于垂直朝向，该属性指的是图像高度。对于水平朝向，该属性指的是图像宽度。如果按照设置的图像大小，图像会占用其他内容的空间，则其大小会被自动缩减。如果不希望这样，则应根据朝向扩展整个控制器的宽度（垂直）或高度（水平）。

性能

当 ImageWatch 控件的 `ImageFreeze` 首次激活时，将创建历史控件的图像列表。对于较大的 `QueueSize` 值，根据系统性能，这可能需要更多时间。每次激活 `ImageFreeze` 时，仅更新列表条目（前提是条目数量保持不变），这样会快得多。因此，我们建议不要将 `QueueSize` 选得过大。

用于自定义字体的属性

`ListFontFamily`、`ListFontSize`、`ListFontStyle` 和 `ListFontWeight` 属性可用于自定义列表样式。

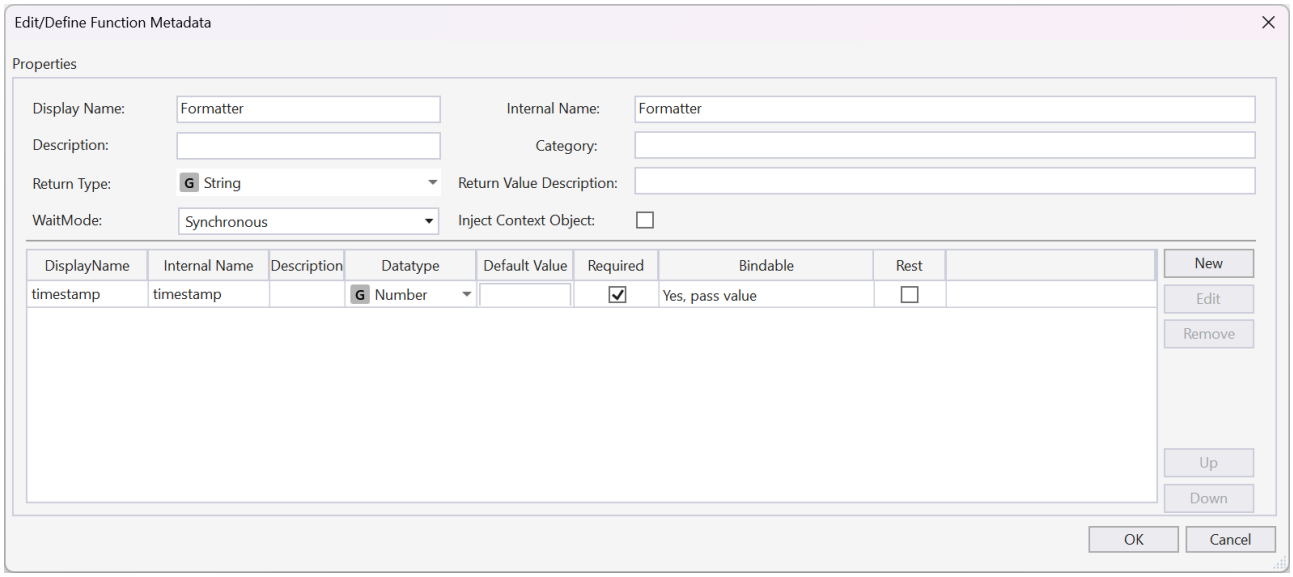
7.2.3.2.1 TimestampFormatting 函数

作为内部格式化函数的替代方案，可以创建一个用户定义的格式化函数，对时间戳进行格式化。该函数会接收图像的时间戳并返回一个格式化后的字符串，该字符串显示在图像列表中。下方示例展示了如何创建这样一个函数，以及它必须具备哪些接口。

在 HMI 文档中会说明各函数的一般使用方法。

创建 TypeScript 函数后，必须创建一个类型为“数字”的参数。该参数接收与 ImageWatch 控件的 `ImageTimestamp` 属性类似的时间戳。

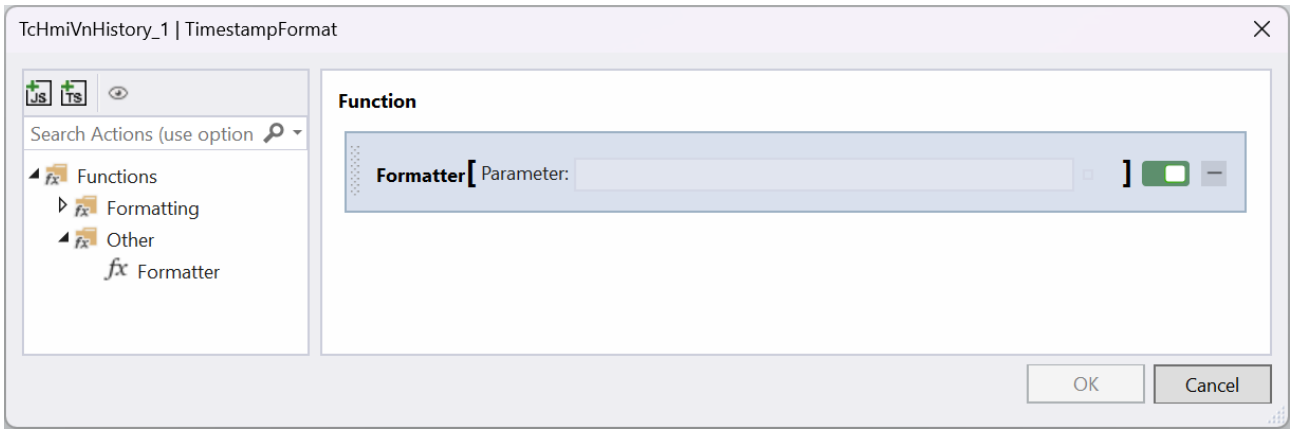
该函数必须返回一个类型为“字符串”的值，然后显示在图像列表中。每当需要显示新的时间戳时，就会针对每个列表条目调用该函数。



在以下程序示例中，时间戳被转换为带有时间的本地日期格式并返回。

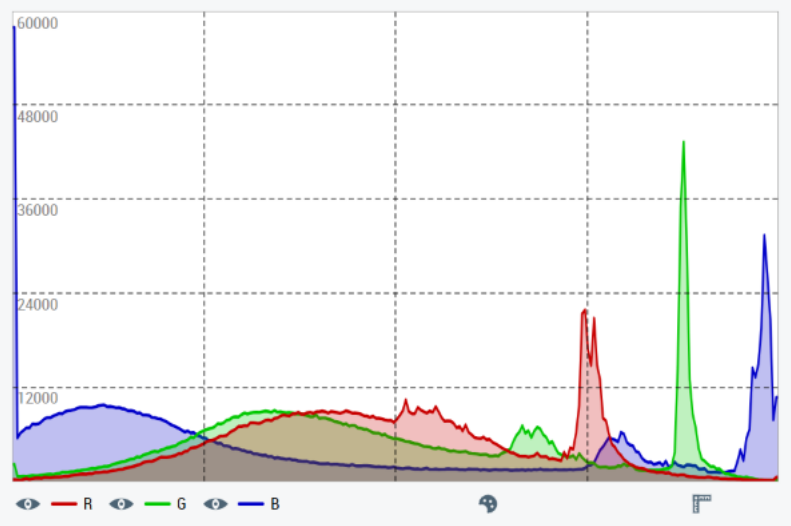
```
namespace TcHmi {
    export namespace Functions {
        export namespace TcHmiProject {
            export function Formatter(timestamp: number) {
                return new Date(timestamp).toLocaleString();
            }
        }
    }
}
TcHmi.Functions.registerFunctionEx('Formatter', 'TcHmi.Functions.TcHmiProject',
TcHmi.Functions.TcHmiProject.Formatter);
```

要使用您创建的格式化函数，请点击 `TimestampFormat` 属性上的 Select（选择）按钮，然后在 Function Editor（函数编辑器）中选择您刚刚创建的功能。



7.2.4 直方图

直方图控件 `TcHmiVnHistogram` 根据图像生成并显示直方图。它还提供了在生成直方图之前将图像从颜色格式 `RGB` 转换为另一种格式的选项。直方图可以显示为折线图或条形图。每个配置选项的说明可在本页的以下类别中找到。每个类别都有分章，提供更多信息、细节和应用示例。为了从 PLC 直接处理 TwinCAT Vision 图像，还需要适用于 HMI 服务器的 Vision 扩展 [▶ 2660]。



对于以下所有属性，控件有一个 getter 和一个 setter 方法。例如，属性 `LabelColor` 存在两个方法：`getLabelColor(): SolidColor` 和 `setLabelColor(value: SolidColor): void`。属性的使用将在链接分章中进行更详细的说明。

Controls（控件）基类所提供的所有其他属性、事件和权限均基于 TwinCAT HMI 文档。

类别：颜色

属性	类型	描述
`HistogramBackgroundColor`	`Color`	直方图的背景颜色。
`SelectionColor`	`Color`	直方图上选择标记的颜色。
`GridLineColor`	`Color`	直方图上网格线的颜色。
`LabelColor`	`Color`	用于以频数值标记网格线的标签的颜色。
`InfobarForegroundColor`	`Color`	信息栏的前景颜色。

类别：常用

以下属性可限定直方图的资源 [▶ 2710]和视图 [▶ 2710]。

属性	类型	描述
`Image`	`String`	图像显示资源。可以是图像文件的路径，也可以是数据 URI。
`ImageConversion`	`ImageConversion`	用于直方图计算的图像转换格式。输入图像始终被解释为 RGB。
`ColorChannels`	`ColorChannels`	定义颜色通道的显示方式。其中包括： • 通道数量 • 显示颜色通道 • 通道的标签 • 通道的可见性

属性	类型	描述
ChartType	ChartType	直方图类型。可能为： • Line 用于折线图视图 • Bar Overlay 用于带有通道透明叠加效果的条形图视图 • Bar Mix 用于带有通道叠加颜色混合效果的条形图视图。
Histogram (只读)	Histogram	当前图像计算得到的直方图数据，以二维数值数组的形式表示。该数组包含每个颜色通道的频数值和颜色值。数组的第一个维度对应颜色通道，第二个维度对应颜色值。

类别：选择

以下属性和函数与颜色值的选择 [▶ 2712]有关。

属性	类型	描述
SelectionMode	SelectionMode	确定是否可以选择多个连续的颜色值（Range）、只选择单个颜色值（Single）、同时选择两个选项（Single or Range）或完全不选择（Disabled）。
SelectionLowerBound (只读)	Number	当前选择的最低颜色值。对于单选，这是所选的颜色值。
SelectionUpperBound (只读)	Number	当前选择的最高颜色值。对于单选，这是所选的颜色值。
SelectionFrequencies (只读)	Frequencies	若为单次选择，展示当前所选图像各通道的频数值；若为范围选择，则展示其平均值。

函数	描述
clearSelection(): void	取消当前的颜色值选择。

类别：信息栏

以下属性可用于自定义控件的信息栏 [▶ 2712]。

属性	类型	描述
ShowInfobar	Boolean	确定是否显示信息栏。
InfobarShowLegend	Boolean	限定在信息栏中是否显示图例。
InfobarShowFrequencies	Boolean	确定在信息栏中是否显示当前选择的频数值。
InfobarShowBin	Boolean	确定在信息栏中是否显示与当前选择相关的亮度范围（条柱）。
InfobarFontFamily	FontFamily	信息栏的字体。
InfobarFontSize	MeasurementValue	信息栏的字体大小。
InfobarFontStyle	FontStyle	信息栏的字体样式。
InfobarFontWeight	FontWeight	信息栏的字体粗细。

类别：网格

以下属性可用于自定义控件的网格线 [▶ 2713]。

属性	类型	描述
ShowGrid	Boolean	限定在直方图上是否显示网格线。
GridShowHorizontalLines	Boolean	限定在直方图上是否显示水平网格线。
GridHorizontalLineCount	Number	水平网格线的数量。
GridShowVerticalLines	Boolean	限定在直方图上是否显示垂直网格线。
GridVerticalLineCount	Number	垂直网格线的数量。
GridLineWidth	Number	网格线的粗细。
GridLineStyle	ChartLineStyle	网格线的样式。可能为： • Dashed • Dotted • Solid

类别：子网格

以下属性可用于调整控件的子网格线 [▶ 2713]。

属性	类型	描述
SubgridShowHorizontalLines	Boolean	限定在直方图上是否显示垂直子网格线。
SubgridHorizontalLineCount	Number	两条水平网格线之间的水平子网格线的数量。
SubgridLineWidth	Number	子网格线的粗细。
SubgridLineStyle	ChartLineStyle	子网格线的样式。可能为： • Dashed • Dotted • Solid

类别：标签

用于以频数值标记网格线的标签 [▶ 2713]，可以通过以下属性进行自定义。

属性	类型	描述
ShowLabels	Boolean	限定是否在每条水平网格线处显示带有相应频数值的标签。
LabelPosition	LabelPosition	标签的显示位置。在直方图旁边的 Left outside 和 Right outside 处，以及在直方图内部相应网格线正下方的 Left inside 和 Right inside 处。
LabelFontFamily	FontFamily	标签的字体。
LabelFontSize	MeasurementValue	标签的字体大小。
LabelFontStyle	FontStyle	标签的字体样式。
LabelFontWeight	FontWeight	标签的字体粗细。

事件

所有事件均列在 **Vision（视觉）** 类别中。

事件	描述
.onImageReceived	控制器接收到图像时会触发该事件。

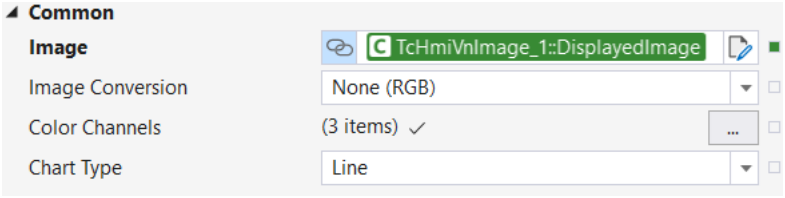
7.2.4.1 直方图资源

本页解释的属性载于 **Common（常用）**类别中。

图像资源

图像资源定义了用于计算直方图数据的图像。预期用途是将 ImageWatch [▶ 2666] 控件的 `DisplayedImage` 属性绑定为图像资源，或者使用 HMI.Vision 扩展 [▶ 2660]链接来自 PLC 的 `ITcVnDisplayableImage` 符号。不过，一般来说，图像资源只需要是一个有效的图像 URL。这可以是项目中的直接路径、外部服务器的 URL 或数据 URL。其作用类似于标准图像控制器的 Src 属性。

使用 `Image` 属性指定图像资源：



例如，为了将直方图与 ImageWatch 控件的显示图像进行同步，可以将 ImageWatch 控件的 `DisplayedImage` 属性通过数据绑定直接绑定到 `Image` 属性，如截图所示。

颜色格式转换

通常情况下，传入图像的三个通道会被解释为 RGB 值，并直接用于生成直方图数据。不过，如果要根据不同的颜色格式来计算直方图，有两种方法可以实现。

首先，`ImageConversion` 属性可以用于设置一种颜色格式，在生成直方图之前会将图像转换为该格式。可能的颜色格式有：`None(RGB)`、`Gray`、`HSV`、`HSV FULL`、`HLS`、`HLS FULL`、`Lab`、`YUV` 和 `YCrCb`。使用默认值 `None(RGB)` 时，图像不进行转换。设置的颜色格式会影响直方图中颜色通道的显示颜色。有关此方面的更多信息，请参见直方图视图 [▶ 2710]页面。

或者，可以事先在 PLC 中使用 F_VN_ConvertColorSpace [▶ 1248] 函数对图像进行转换，然后将其作为伪彩色图像传输到直方图控件。在这种情况下，`ImageConversion` 应保持为 `None(RGB)` 不变，标签和颜色可以通过 **Common（常用）**类别中的 `ColorChannels` 进行手动调整。

ImageConversion 的限制

使用 `ImageConversion` 时，传入图像始终被解释为 RGB。对于非 RGB 格式的图像，这种内部转换会导致直方图数据失真。

计算得到的直方图

计算得到的直方图值可以通过 `Histogram` 属性以二维数组的形式完全访问。数组的第一个维度对应颜色通道，第二个维度对应颜色值。例如，索引 `[1][120]` 可以用于查询第二个颜色通道中有多少像素的颜色值为 120。

接收事件

每次控件接收新图像时，都会触发 `onImageReceived` 事件。

7.2.4.2 直方图视图

本页解释的属性和函数载于 **Common（常用）**和 **Colors（颜色）**类别中。

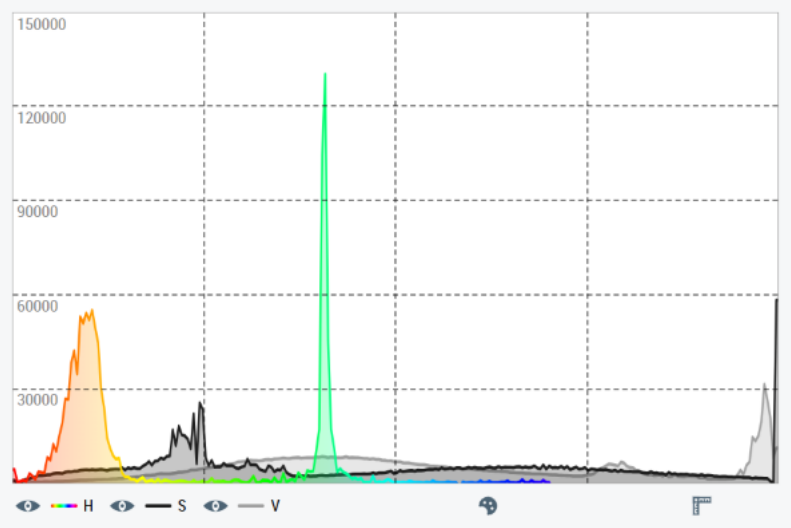
属性： `ColorChannels`

`ColorChannels` 属性是一个数组，可用于定义各个颜色通道的显示方式。该数组可以有 1-3 个元素，每个条目包含一个颜色通道的定义。具体来说，可以为每个颜色通道设置以下详细信息：

- `Color`：显示颜色（也可使用颜色渐变）。

- `Label`：图例中的标签。
- `Visible`：是否显示通道。

如果未设置通道的整个属性或单个值，则会使用内部默认值，该默认值取决于在 `ImageConversion` 和 `ChartType` 中设置的值。例如，当 `ImageConversion` = `HSV` 时，使用标签 H-S-V。此外，还会为 H 通道设置相应的颜色渐变，并为 S 和 V 通道设置灰度值作为颜色：

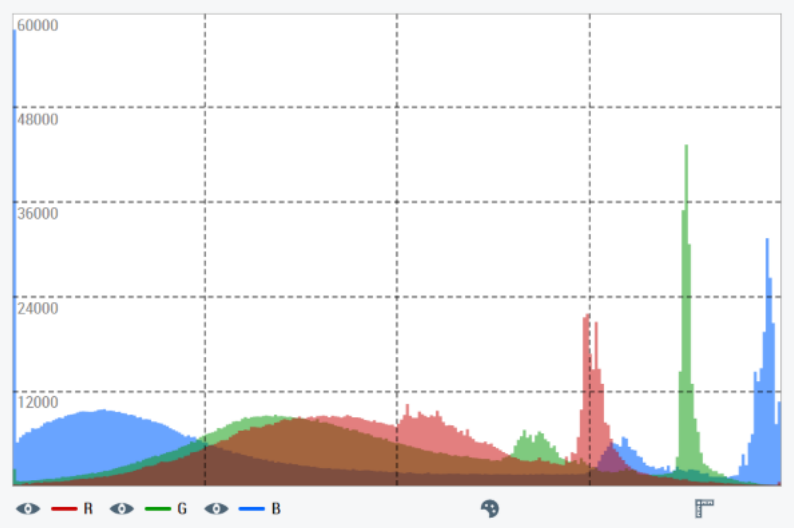


对于单色图像，仅处理第一个颜色通道是合适的。为此，必须 在`ColorChannels` 属性中删除两个通道。如果没有为剩余通道设置手动标签和/或手动颜色，则将自动使用灰度值渐变作为显示颜色，并使用标签“G”（灰色的缩写）。

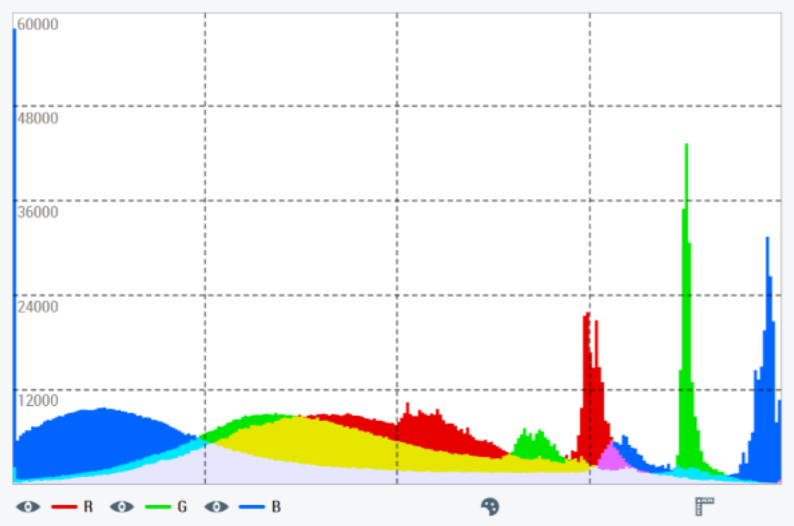
属性： `ChartType`

`ChartType` 属性可用于指定直方图要显示为折线图（`Line`）还是条形图（`Bar Overlay` 和 `Bar Mix`）。在 `Bar Overlay` 中，颜色通道以叠加方式显示。如果使用此设置手动设置显示颜色，则应注意确保透明度，以便所有通道保持可见。不过，使用 `Bar Mix` 时，颜色通道的显示颜色在叠加时会进行混合。

`Bar Overlay` 的示例：



`Bar Mix` 的示例：

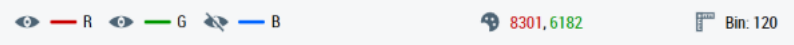


属性： `HistogramBackgroundColor`

使用 `HistogramBackgroundColor` 属性可以调整直方图的背景颜色。尤其是在使用 `Bar Overlay` 图表类型时，调整背景颜色有助于保持所有通道清晰可见。由于数据行在此模式下以透明方式叠加，背景会透过颜色通道显现出来，从而影响它们的对比度。

7.2.4.3 信息栏

本页解释的属性载于 **Infobar（信息栏）**类别中。信息栏显示在控件的底部，并分为两个区域。



颜色通道的图例位于左侧。每个条目包含一个用于显示和隐藏相应通道的图标、显示颜色的预览以及一个标签。这三个部分由 `ColorChannels` 属性定义（请参见直方图视图 [▶ 2710]）。如果通过图例中的图标直接显示或隐藏某个通道，则 `ColorChannels` 属性的相应可见性属性将自动更新。

当前选择的相关信息会在旁边显示。对于简单选择，显示的是所选颜色值（条柱）及其每个颜色通道的频率值。对于范围选择，显示的是所选颜色值范围的平均频率值的最小值和最大值。

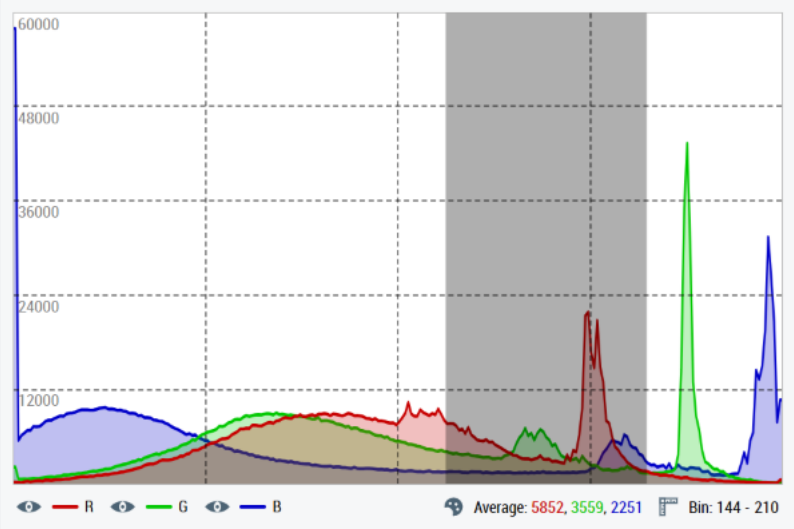
属性

`ShowInfobar` 属性可用于显示和隐藏整个信息栏。此外，还可以使用属性 `InfobarShowLegend`、`InfobarShowFrequencies` 和 `InfobarShowBin` 分别显示和隐藏图例、频率值和所选条柱的显示。

属性 `InfobarForegroundColor`、`InfobarFontFamily`、`InfobarFontSize`、`InfobarFontStyle` 和 `InfobarFontWeight` 可用于自定义信息栏文本的样式。

7.2.4.4 选择

本页解释的属性载于 **Selection（选择）**类别中。



属性：SelectionMode

SelectionMode 属性可用于设置是否以及如何在直方图上选择颜色值：

- 使用 Single 时，只能选择单个颜色值。
- 在 Range 中，可以通过点击、按住并拖动鼠标来选择多个连续的颜色值。
- Single or Range 将这两种选项结合起来。
- Disabled 完全禁用颜色值的选择。

对于单选，所选颜色值（条柱）以及每个颜色通道相应的频率值会显示在信息栏中。选择范围时，会显示每个颜色通道的颜色值范围和该范围内的平均频率值。

属性：SelectionLowerBound 和 SelectionUpperBound

对于范围选择，SelectionLowerBound 属性包含所选的最低颜色值（条柱）。因此，SelectionUpperBound 属性包含所选的最高颜色值。对于单选，两个属性都包含完全相同的所选颜色值。

属性：SelectionFrequencies

SelectionFrequencies 属性包含一个长度为 1-3 的数值数组，具体取决于已定义的颜色通道的数量。对于单选，它包含每个通道所选颜色值的频率值。对于范围选择，则会计算每个通道的平均值。

属性：SelectionColor

SelectionColor 属性用于设置直方图上选择区域的颜色。理想情况下，该颜色应为部分透明（带 α 值），以便标记区域下方的直方图数据系列仍能清晰可见。

函数：clearSelection()

调用 clearSelection() 函数会完全清除当前选择。

事件：onSelectionChanged

一旦选择发生变化（重新选择或取消选择一个或多个颜色值），就会触发 onSelectionChanged 事件。

7.2.4.5 网格线和标签

本页解释的属性载于 **Grid（网格）**、**Subgrid（子网格）**和 **Labels（标签）**类别中。

网格线的可见性

ShowGrid 属性决定是否显示网格线。如果启用 ShowGrid，则可使用 GridShowHorizontalLines 和 GridShowVerticalLines 属性分别指定是否显示水平和/或垂直网格线。

如果水平网格线（或泛指所有网格线）被禁用，则水平子网格线也会自动隐藏。否则，可以使用 SubgridShowHorizontalLines 属性来单独控制是否应显示水平子网格线。

网格线的数量

GridHorizontalLineCount 和 GridVerticalLineCount 属性决定所显示的水平网格线和垂直网格线的总数。使用四条水平线和三条垂直线（默认值）时，直方图会被划分为五个水平区域和四个垂直区域。此外，还会为水平网格线生成相应的数字标签，以表示各自的频率值。

每个水平区域内的子网格线的数量通过 SubgridHorizontalLineCount 属性单独定义。

网格线的标签

使用 ShowLabels 属性可以显示和隐藏标签。此外，LabelPosition 属性还提供了调整标签对齐方式的选项：

- LeftInside（默认值）：在直方图左侧网格线的正下方显示标签。
- RightInside：标签的位置相同，但在直方图内采用右对齐方式。
- LeftOutside 和 RightOutside：标签位于实际的直方图之外（左侧或右侧），并与水平网格线保持相同高度，以避免遮挡任何数据系列。

网格线的样式

GridLineColor 属性用于定义网格线的颜色。GridLineWidth 和 SubgridLineWidth 属性可用于为网格线和子网格线设置不同的线条粗细。此外，GridLineStyle 和 SubgridLineStyle 可用于指定是否将线条显示为虚线（Dashed）、点线（Dotted）或实线（Solid）。

7.2.5 矩形（交互式）

矩形（交互式）TcHmiVnRectangle 控件提供了一个通过点击来显示或绘制矩形的环境。主要是在 图像查看 [▶ 2666] 控制器内使用，但也可以独立使用。因此，举例来说，可以选择图像区域来标记物体或定义 ROI。除矩形外，您还可以绘制正方形、圆形或椭圆形。为此，必须相应调整 笔画形状 和 长宽比 属性。



对于以下所有属性，控件有一个 getter 和一个 setter 方法。**Controls（控件）**基类所提供的所有其他属性、事件和权限均基于 TwinCAT HMI 文档。

类别：颜色

属性	类型	描述
StrokeColor	SolidColor	绘制的矩形轮廓的颜色。
HandleColor	SolidColor	绘制的矩形的手柄颜色。

类别：常用

属性	类型	描述
ShapeData	TcVnRotatedRectangle	包含对矩形的描述。既可用于检索绘制的矩形，也可用于设置特定的矩形。
AspectRatio	Number	指定绘制矩形时执行的长宽比。0 值并不强制使用特定的长宽比。

函数	描述
clear	从 ShapeData 属性中删除绘制的矩形和描述。

类别：用途

属性	类型	描述
InitialSelection	InitialSelection	指定绘制矩形时是双击还是扫动，或者同时使用这两个选项。
CenterHandle	Boolean	在矩形中心显示元素，以移动它。
CornerHandles	Boolean	显示矩形四角的元素，以改变大小。
EdgeHandles	Boolean	显示矩形边缘的元素，以调整大小。
RotationHandle	Boolean	在矩形外侧显示元素，以旋转它。
ShapeDragging	Boolean	确定是否可以移动矩形的边缘。
AngleInterval	Number	指定旋转矩形的步骤。在 0，矩形可按需要旋转。
Boundary	Boundary	定义一个区域，在该区域内可以使用 X, Y, Width, Height 和 IsActive 这五个属性绘制和编辑形状。形状不能离开该区域。
IsShapeInBoundary	Boolean	表示当前形状的所有角点是否都在所设置的 Boundary 范围内。
DataUpdate	DataUpdate	指定 ShapeData 应在 onChanged 期间更新还是在图形交互结束 onConfirmed 时更新。

类别：外观

属性	类型	描述
StrokeThickness	MeasurementValue	矩形的线条粗细。
HandleSize	MeasurementValue	移动和更改矩形的元素大小。
StrokeShape	StrokeShape	指定绘制的是矩形还是椭圆形。二者的描述完全相同。要获得正方形或圆形，可将 长宽比 设置为 1。
ClickableSize	MeasurementValue	上述元素周围可点击区域的大小，用于移动和更改矩形。这样可以在不改变外观的情况下简化触摸屏的使用。
Offset	Vector2	指定显示形状的偏移量。
Scale	Number	指定显示形状的缩放比例。1 值对应于 100%。

事件

事件	描述
.onShapeChanged	一旦 ShapeData 属性发生变化，就会触发该事件。
.onShapeConfirmed	设置形状或完成与形状的交互时会触发该事件。

7.2.6 多边形（交互式）

多边形（交互式）TcHmiVnPolygon 控件提供了一个通过点击来显示或绘制多边形的环境。该控制器主要在 Image Watch（图像查看）[▶ 2666]控制器中使用，但也可以独立使用。除多边形外，还可以显示或绘制线条或点。为此，必须相应地将 NumPoints 属性设置为 1 或 2。

对于以下所有属性，控件有一个 getter 和一个 setter 方法。**Controls（控件）**基类所提供的所有其他属性、事件和权限均基于 TwinCAT HMI 文档。

类别：颜色

属性	类型	描述
StrokeColor	SolidColor	绘制多边形轮廓的颜色。
HandleColor	SolidColor	绘制的多边形手柄的颜色。

类别：常用

属性	类型	描述
ShapeData	PolygonData	包含对多边形的描述。既可用于检索绘制的多边形，也可用于设置特定的多边形。
NumPoints	Number	指定完整多边形的点数。在 0 点时，任何数量的点都有可能出现。要将多边形作为完整多边形处理，这一规范非常重要。
Closed	Boolean	表示多边形是封闭的还是折线。

函数	描述
clear	从 ShapeData 属性中删除绘制的多边形和描述。

类别：用途

属性	类型	描述
RotationHandle	Boolean	在多边形外侧显示元素以旋转多边形。
形状拖动	Boolean	限定多边形是否可沿线条方向移动。
AngleInterval	Number	指定旋转多边形的步骤。在 0，多边形可以按需要旋转。
Boundary	Boundary	定义一个区域，在该区域内可以使用 X, Y, Width, Height 和 IsActive 这五个属性绘制和编辑形状。形状不能离开该区域。
IsShapeInBoundary	Boolean	表示当前形状的所有角点是否都在所设置的 Boundary 范围内。
DataUpdate	DataUpdate	指定 ShapeData 应在 onChanged 期间更新还是在图形交互结束 onConfirmed 时更新。

类别：外观

属性	类型	描述
StrokeThickness	MeasurementValue	多边形的线条粗细。
HandleSize	MeasurementValue	移动和改变多边形的元素大小。

属性	类型	描述
ClickableSize	MeasurementValue	上述元素周围可点击区域的大小，用于移动和更改多边形。这样可以在不改变外观的情况下简化触摸屏的使用。
Offset	Vector2	指定显示形状的偏移量。
Scale	Number	指定显示形状的缩放比例。1 值对应于 100%。

事件

事件	描述
.onShapeChanged	一旦 ShapeData 属性发生变化，就会触发该事件。
.onShapeConfirmed	设置形状或完成与形状的交互时会触发该事件。

7.2.7 帮助功能

以下帮助函数被某些控件用作默认行为，或用于确保与 PLC 数据类型的互操作性。

BoxColorConverter

函数 BoxColorConverter 列于 Vision 类别中。它被用作 Color Control（颜色控制器） [▶ 2689] BoxColorConversion 属性的默认值。它在 RGB 和 Format 参数指定的颜色格式之间进行转换。

参数	数据类型	描述
ctx	SelectableRequired<TcHmi.Context, 'success' | 'error'> | null	用于 HMI 内部接收异步返回。在 JavaScript 中调用时可设置为 null。
Color	Vector4	要转换的颜色值。
Reverse	Boolean	转换方向： false：format 至 RGBa true：RGBa 至 format
Format	ColorSpace	转换格式： Gray, RGB, HSV, HLS, RGBa, HSV FULL, HLS FULL, Lab, YUV, YCrCb
PixelType	ElementType	颜色值的缩放： 8-Bit (USINT), 16-Bit (UINT)

由于函数是异步运行的，因此它将转换后的颜色作为未来值返回，即 Promise<ColorValue | null>。有关处理 Promise，请参见 ConvertColor。转换过程中的错误会导致返回 null。

ConvertColor

函数 ConvertColor 列于 Vision 类别中。它能在灰色、RGB、RGBa、HSV、HSV FULL 和 HLS、HLS FULL、Lab、YUV 和 YCrCb 格式之间转换颜色阵列。

参数	数据类型	描述
ctx	SelectableRequired<TcHmi.Context, 'success' | 'error'> | null	用于 HMI 内部接收异步返回。在 JavaScript 中调用时可设置为 null。
Color	Vector4	要转换的颜色值。
SrcFormat	ColorSpace	传入颜色值的格式：

参数	数据类型	描述
		Gray, RGB, HSV, HLS, RGBa, HSV FULL, HLS FULL, Lab, YUV, YCrCb
DestFormat	ColorSpace	转换成的格式： Gray, RGB, HSV, HLS, RGBa, HSV FULL, HLS FULL, Lab, YUV, YCrCb
Precision	Precision	指定四舍五入的小数位数。不对 Unrounded 四舍五入。
SrcPixelType	ElementType	对传入颜色值进行缩放： 8-Bit (USINT), 16-Bit (UINT)
DestPixelType	ElementType	对转换后的颜色值进行缩放： 8-Bit (USINT), 16-Bit (UINT)

转换过程中的错误会导致返回 0。由于该函数是异步函数，它将转换后的颜色作为未来值返回，即 Promise<ColorValue | null>。

例如，要将 RGB 格式的红色转换为 HLS 格式，可以使用下面的函数调用：

```
let rgb = [255, 0, 0];
let promise = TcHmi.Functions.Beckhoff.Vision.ConvertColor(null, rgb, "RGB", "HLS", "2 Decimals",
"8-Bit (USINT)", "8-Bit (USINT)");
promise.then(hls => console.log(hls.toString()));

//console output
//0,127.5,255
//console output with DestPixelType "16-Bit (UINT)"
//0,32640,65280
```

为了能够处理未来的 ColorValue，可以通过 await 等待。或者，如本样本所示，函数 then 可用于传递匿名函数，如 hls => console.log(hls.toString()) 。这会在转换完成后接收 ColorValue。然后 ColorValue 会在控制台中以 string 的形式输出。

PixelColorFormatter

函数 PixelColorFormatter 列于 Formatting 类别中。此为 Image Watch Control（图像查看控制器）[▶ 2666] PixelColorFormat 属性的默认值。它会将 PixelColor 值转换为可通过 HTML 显示的字符串。具体来说，各个通道的颜色值相互附加，并用对于某些颜色格式也以颜色突出显示：



参数	数据类型	描述
ctx	SelectableRequired<TcHmi.Context, 'success' | 'error'> | null	用于 HMI 内部接收异步返回。在 JavaScript 中调用时可设置为 null。
PixelColor	Vector4	要转换的颜色值。
Format	ColorSpace	转换格式： Gray, RGB, HSV, HLS, RGBa, HSV FULL, HLS FULL, Lab, YUV, YCrCb
Precision	Precision	指定四舍五入的小数位数。使用“非四舍五入”时，不进行四舍五入。
PixelType	ElementType	颜色值的缩放： 8-Bit (USINT), 16-Bit (UINT)

该函数可返回 `String` HTML 代码的 `Promise<string>`。有关处理 Promise，请参见 `ConvertColor`。

ShapeValueFormatter

`ShapeValueFormatter` 函数列于 `Formatting` 类别中。此为 Image Watch Control（图像查看控制器） [▶ 2666] `ShapeValueFormat` 属性的默认值。它会将 `ShapeValue` 值转换为可通过 HTML 显示的字符串。例如，对于矩形，面积以像素为单位计算，并以字符串形式返回：



参数	数据类型	描述
`ShapeData`	`Any`	要转换的形状数据。
`ShapeType`	ShapeType [▶ 2720]	传递的形状类型。

该函数返回一个 HTML 代码的 `String`。

旋转矩形

`ToRotatedRectangle`函数列在数据转换类别中。它将一个`UprightRectangle`转换为一个`RotatedRectangle`，角度为`0`。

参数	数据类型	描述
`rectangle`	TcVnRectangle [▶ 2721]	待转换矩形。

该函数以TcVnRotatedRectangle [▶ 2720]的形式返回转换后的矩形，并可按如下方式应用：

```
let upright = {
    nWidth: 100,
    nHeight: 60,
    nX: 200,
    nY: 100,
}
let rotated = TcHmi.Functions.Beckhoff.Vision.ToRotatedRectangle(upright);
console.log(rotated);

//console output
// {
//     aCenter: [250, 130],
//     fAngle: 0,
//     stSize: { fWidth: 100, fHeight: 60 },
// }
```

直立矩形

`ToUprightRectangle`函数列在数据转换类别中。它将一个`RotatedRectangle`转换为一个`UprightRectangle`，放弃了角度。

参数	数据类型	描述
`rectangle`	TcVnRotatedRectangle [▶ 2720]	待转换矩形。

该函数以TcVnRectangle [▶ 2721]的形式返回转换后的矩形，并可按如下方式应用：

```
let rotated = {
    aCenter: [250, 130],
    fAngle: 0,
    stSize: { fWidth: 100, fHeight: 60 },
}
let upright = TcHmi.Functions.Beckhoff.Vision.ToUprightRectangle(rotated);
console.log(upright);

//console output
// {
//     nWidth: 100,
//     nHeight: 60,
//     nX: 200,
//     nY: 100,
// }
```

7.2.8 数据类型

TcHmiVnColor.ColorChannels

定义颜色格式的通道。它是一个具有以下属性的对象数组，每个对象都描述一个颜色通道：

Property（属性）	数据类型	描述
color	SolidColor	显示通道的颜色。
maxValue	Number	允许值范围的上限。下限始终为 0。
float	Boolean	定义通道是用浮点数还是整数来描述。
label	String	通道的标签

例如，在色彩控件 [▶ 2695]中使用。

颜色符号

定义一个具有 ColorValue 兼容数据类型的符号。

颜色值

定义了一个可以作为颜色进行解释的值。是一个描述灰度值的数字，或者是一个通过相应通道数描述颜色的 3 或 4 元素数组。

例如，有效值为：

- 0
- 128
- [0, 0, 255]
- [0, 0, 255, 255]

方向

定义了显示元素的方向。可能的值为“水平”和“垂直”。

TcVnRotatedRectangle

代表可以旋转的矩形。是一个具有以下属性的对象：

属性	数据类型	描述
aCenter	TcVnPoint2_REAL	矩形中心。
fAngle	DINT	旋转矩形的角度。
stSize	TcVnSize2_REAL	围绕中心点的矩形尺寸。

形状类型

在图像查看控件 [▶ 2666]中定义了可选择的形状类型。是一个枚举函数，附带以下值：

- 点
- 线
- 方形
- 矩形
- 圆形
- 椭圆
- 多边形

TcVnRectangle

代表一个轴对齐的矩形，且是一个具有以下属性的对象：

属性	数据类型	描述
nHeight	DINT	矩形高度。
nWidth	DINT	矩形宽度。
nX	DINT	右上角的 X 坐标。
nY	DINT	右上角的 Y 坐标。

使用模式

定义了显示元素的显示和可编辑性。可能的值：

数值	描述
隐藏	元素未显示。
仅查看	元素已显示，但不能编辑。
可编辑	元素已显示且可编辑。

主要用于颜色控制 [▶ 2691]。

8 例程和工具

在本章中，您可以找到 PLC 中许多函数和功能块以及向导和相机配置的应用示例。在开始查看 PLC 例程之前，请访问 第一步 [▶ 27]。

各例程相互独立。每个示例都有一个重点，在下方会对其进行分组。以函数或功能块为重点的例程根据 API 结构列出。

您还可以找到其他支持工具：

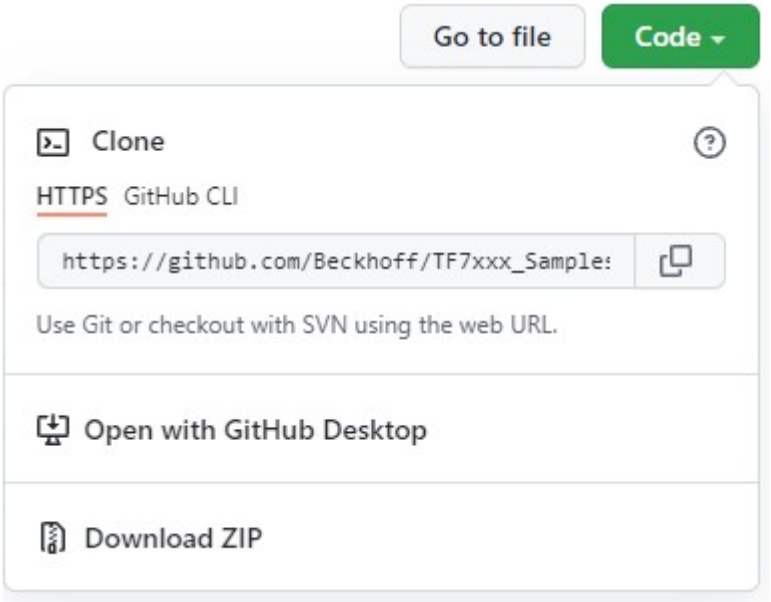
- 神经网络 (ONNX) 兼容性检查工具 [▶ 2837]

下载

例程可从 GitHub 上的存储库获取：https://github.com/Beckhoff/TF7xxx_Samples。您可以选择克隆存储库或下载包含所有例程的 ZIP 文件。

必须单独下载例程图像

GitHub 存储库只包含带有相应程序的 TwinCAT 项目。必须从倍福网站单独下载图像数据“TF7xxx_Sample_Images.zip”。链接和更多信息也可在 GitHub 存储库的 Readme 文件中找到。



项目资源

每个示例都按照大纲顺序和章节标题名称放在一个单独文件夹中。其中有一个 TwinCAT 项目和相应的 PLC 代码。

例如，包含例程图像的独立 zip 文件夹可以解压到存储库中的 Images（图像）文件夹中。文件夹名称以及例程图像的存储结构与项目结构相同。因此，只需通过具有相同文件夹名称的路径即可分配图像。

例如，注意通往存储库或 TwinCAT 项目的短路径

由于操作系统可能将路径长度限制为 255 个字符，如果文件夹路径过长，在执行 TwinCAT 项目时可能会出现问题。为了便于理解，带有章节标题的文件夹结构已经很长，因此应为整个存储库或单个 TwinCAT 项目选择一个较短的路径，例如 C:\TF7xxx_Samples/。

在执行项目之前，请根据目标系统调整系统配置 [▶ 31]，尤其要注意 CPU 核、路由器内存和 PLC 循环时间的配置。此外，请按照 版本概览 [▶ 15] 中的说明检查相机或文件源的 TcCOM 对象以及 PLC 库的版本，必要时调整版本。

在 GitHub 存储库中可以找到一个 PowerShell 脚本 (SetTcCOMModuleVersion.ps1)，其可用于调整 PLC 文件夹中所有项目的版本。为此，请通过 PowerShell 控制台运行脚本，如下指定所需版本：

PS C:\TF7xxx_Samples> .\SetTcCOMModuleVersion.ps1 4.0.4.8 (5.9.19.0)

如果需要图像或数据流，可通过 文件源控制 [▶ 124] 或相机对象的 记录/回放 [▶ 101] 选项卡对它们进行整合。

8.1 函数样本

以下所有示例均聚焦于单一功能，并按照 TwinCAT Vision API 参考文档进行排序。每个示例都配有相应的图像，这些图像可通过 File Source（文件源）加载到 PLC 中。此外，相应的示例说明还显示了需要修改哪些函数参数才能看到特定的效果。

启动功能示例的一般步骤如下：

1. 在文件源控制器中添加所附的示例图像。
   使用您自己的图像时，请注意预期的图像格式，这可能因示例而异。
2. 激活配置并将 TwinCAT 和 PLC 切换至运行模式。

⇨ 现在，您可以通过 **View > Other Windows > ADS Image Watch（视图 > 其他窗口 > ADS 图像查看）**查看来自 PLC 的图像。
- `ipImageInDisp` 始终显示输入图像，
- `ipImageResDisp` 始终显示结果图像
（当然，您所使用的图像名称可能与此不同，我们在示例中始终将变量命名为相同的名称，以便于识别，并帮助您在任何项目中快速找到所需内容。）
- 如有需要，其他图像会显示中间步骤的结果或更多信息。
详细信息可在相应的示例说明中找到。

在示例说明中，您还可以找到在登录后的运行模式下通过**准备值**写入的函数参数，然后点击 **写入值**。然后，您可以在 Ads Image Watch（Ads 图像查看）中显示的图像中再次看到这些效果。

示例

- 基本图像运算 [▶ 2725]
- 编码读取 [▶ 2731]
- 轮廓分析 [▶ 2747]
- 图像分析 [▶ 2755]
- 图像颜色和对比处理 [▶ 2781]
- 图像分割 [▶ 2790]
- 测量 [▶ 2793]

编写自定义函数的示例

如果您在自己的应用场景中反复使用某些 TwinCAT Vision API 函数序列，最好将它们封装在一个函数中。以下示例说明了在此过程中您应该注意的事项：

示例：自写入函数 [▶ 2814]

8.1.1 容器基本操作

8.1.1.1 容器元素选择

在本示例中，借助用户自定义的条件对容器元素进行筛选。为此可以使用函数 F_VN_CopyContainerElementsConditional_ITcVnContainer [▶ 749]，因为容器元素本身也是容器。如果容器元素是迭代器，也可以使用 F_VN_CopyContainerElementsConditional_ITcVnForwardIterator [▶ 750] 函数。

说明

如果要对多维容器中的元素进行筛选，由于要访问其子元素，这可能会迅速导致大量的编程工作。为了解决这个问题，TwinCAT Vision 专门提供了一个函数，其用法示例如下：

创建一个功能块，该功能块可实现 ITcVnCustomElementCondition_ITcVnContainer 接口。该接口拥有 Condition 方法，该方法接受 ITcVnContainer 类型的元素。要应用于元素的筛选条件在该方法的实现中定义。然后，将带有筛选条件的功能块传递给 F_VN_CopyContainerElementsConditional_ITcVnContainer 函数，以便对容器中的所有元素执行筛选操作。在本示例中，筛选条件为元素是否拥有超过五个子元素。

应用

举例来说，对一个容器中的子容器进行筛选，筛选条件为每个子容器所拥有的元素数量。具体而言，只有那些拥有超过五个元素的子容器才会被纳入一个新容器中。通过 aContainerStructure 数组可以改变容器的结构。这些元素表示每个子容器所包含的子元素数量。如果子容器本身的数量也发生变化，则必须相应调整 cNumberOfSubContainers 常量以与之匹配。

程序

主要

首先定义一个整型常量，以便能够简单地改变要筛选的子容器：

```
VAR CONSTANT
    cNumberOfSubContainers  :   INT := 5;
END_VAR
```

然后声明所有必要的变量：

```
VAR
    aContainerStructure     :   ARRAY [0..(cNumberOfSubContainers-1)] OF INT := [2, 7, 3, 5, 12];
    ipHelper                :   ITcVnContainer;
    ipContainerBase         :   ITcVnContainer;
    ipContainerFiltered     :   ITcVnContainer;
    fbCondition             :   FB_ConditionMoreThanFive;
    nSelectedContainers     :   ULINT;
    hr                      :   HRESULT;
    i                       :   INT;
END_VAR
```

aContainerStructure	容器结构的定义。例如，一个长度为 5 且初始化值为 [2, 7, 3, 5, 12] 的数组意味着在示例中将创建一个包含 5 个子容器的容器。第一个子容器有 2 个元素，第二个子容器有 7 个元素，以此类推。 数组的长度由 cNumberOfSubContainers 常量定义。
ipHelper	辅助接口指针，用于为 ipContainer 主容器填充子容器。
ipContainerBase	容器，用于创建在 aNumberOfElement 中定义的子容器。
fbCondition	包含筛选条件的功能块。
ipContainerFiltered	容器，最终仅包含 ipContainer 中满足筛选条件的子容器。
nSelectedContainers	整数，指示实际满足筛选条件的子容器的数量。
hr	类型 HRESULT [▶ 135] 的状态变量。
i	辅助变量

在程序部分，创建一个容器，并用上述定义的子容器进行填充。调用 F_VN_ReserveContainerMemory [▶ 768] 是可选的，仅为了获得更好的性能。

```
hr := F_VN_CreateContainer(ipContainerBase, ContainerType_Vector_Vector_REAL, 0, hr);
hr := F_VN_ReserveContainerMemory(ipContainerBase, TO_ULINT(cNumberOfSubContainers), hr);
FOR i:=0 TO (cNumberOfSubContainers-1) DO
    hr := F_VN_CreateContainer(ipHelper, ContainerType_Vector_REAL,
TO_ULINT(aContainerStructure[i]), hr);
    hr := F_VN_AppendToContainer_ITcVnContainer(ipHelper, ipContainerBase, hr);
END_FOR
```

然后，对刚刚创建的容器进行筛选。将包含自写入条件的功能块指定为筛选条件。

```
hr := F_VN_CopyContainerElementsConditional_ITcVnContainer(
  ipSrcContainer    := ipContainerBase,
  ipDestContainer   := ipContainerFiltered,
  ipConditionFB     := fbCondition,
  hr);
```

为了理解该函数的功能，对满足所述条件的元素进行计数。现在，nSelectedContainers 变量会根据子容器和筛选条件而变化。

```
hr := F_VN_GetNumberOfElements(ipContainerFiltered, nSelectedContainers, hr);
```

在这种情况下，数字表示有多少个子容器拥有超过五个子元素。

包含筛选条件的功能块

筛选条件以包含 Condition 方法的功能块形式进行传输。为了能够将自写入功能块传输到该函数，它必须实现 ITcVnCustomElementCondition_ITcVnContainer [▶ 240]接口：

```
FUNCTION_BLOCK FB_ConditionMoreThanFive IMPLEMENTS ITcVnCustomElementCondition_ITcVnContainer
```

Condition 方法定义了筛选条件。每个子容器都会单独调用它。根据返回值是 TRUE 还是 FALSE，决定相应的子容器是否被纳入新容器中。

```
METHOD Condition : BOOL
```

每次调用该方法时，当前子容器会作为输入参数进行传输：

```
VAR_INPUT
    ipElement           :   ITcVnContainer;
END_VAR
```

在这种情况下，如果一个元素拥有超过五个子元素，则将其纳入。为此，计算子元素的数量，并根据计算结果返回 TRUE 或 FALSE 值。

```
hr := F_VN_GetNumberOfElements(ipElement, nNumberOfElements, hr);
IF FAILED(hr) OR nNumberOfElements <= 5 THEN
    Condition := FALSE;
ELSE
    Condition := TRUE;
END_IF
```

所需变量声明如下：

```
VAR
    nNumberOfElements   :   ULINT;
    hr                  :   HRESULT;
END_VAR
```

功能块本身和 Condition 方法都必须包含 {attribute 'c++_compatible'} 编译器属性。

8.1.2 基本图像操作

8.1.2.1 融合多个图像

在本例程中，使用 F_VN_FuseImagesArray [▶ 797] 函数将五个图像连接起来。该功能在与线扫描相机和连续物料流的无缝记录相结合时尤为实用。

说明

F_VN_FuseImagesArray [▶ 797] 函数接受一个图像数组（最大为十），并将它们连接成一个新图像。在下方样本程序中，最初创建了 5 个不同颜色、高度为 10px 的图像。数量可以通过变量 nImages 更改，高度可以通过 nHeight 更改。请确保该数字仅在 1-10 范围内变化。

使用参数 nFirstLine 和 nNumLines 指定图像线，即可确定结果图像的大小。这样就可以裁剪第一和最后一个图像。

如果只需要连接两个图像，也可以使用函数 F_VN_FuseImages [▶ 796]。

程序

声明了以下变量：

```
VAR
    hr                  :  HRESULT;
    aImages       : ARRAY[0..9] OF ITcVnImage;
    ipImageFused     : ITcVnImage;
    ipImageFusedDisp    :  ITcVnDisplayableImage;
    nImages             :  INT := 5;
    nHeight             :  UDINT := 10;
    aColor              :  ARRAY[0..9] OF TcVnVector4_LREAL := [[255,0,0], [200,50,0], [150,100,0],
[100,150,0], [50,200,0], [0,250,0], [0,200,50], [0,150,100], [0,100,150], [0,50,200]];
    i                   :  INT;
VAR
```

hr	类型 HRESULT [▶ 135] 的状态变量。
aImages	包含多个要连接在一起的图像之数组。
ipImageFused	将来自 aImages 的多个图像连接在一起的结果图像。
ipImageFusedDisp	可显示的结果图像。
nImages	要连接的图像数量。
aColor	十种不同的颜色，以便能够区分 aImages 中的图像。
i	任意变量。

在程序中，首先对 `aImages` 数组中的图像进行初始化和着色。为了能够直观地区分图像，使用十种不同的 `aColor` 颜色。

```
FOR i := 0 TO nImages-1 DO
     hr := F_VN_CreateImageAndSetPixels(aImages[i], 100, nHeight, ETcVnElementType.TCVN_ET_USINT, 3,
aColor[i], hr);
END_FOR
```

然后，将来自 `aImages` 的定义数量 `nImages` 图像垂直连接到图像`ipImageFused` 中：

```
hr := F_VN_FuseImagesArray(aImages, nImages, ipImageFused, 0, nImages * nHeight, hr);
```

最后显示图像：

```
hr := F_VN_TransformIntoDisplayableImage(ipImageFused, ipImageFusedDisp, hr);
```

结果

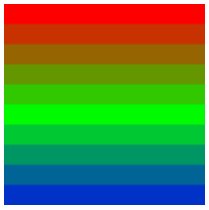
nImages = 2：



nImages = 5：



nImages = 10：



类似例程

复制图像区域 [▶ 2726]

8.1.2.2 复制图像区域

在本示例中，使用不同方法将两个图像复制到了另一图像：

- 使用 F_VN_CopyImageRegionToRegion [▶ 780] 函数，
- 选择 ROI 并复制图像，然后

- 使用 F_VN_AddImages [▶ 506] 函数选择一个 ROI 并添加两个图像。

说明

下图所示方法可用于在图像之间交换矩形图像区域。如果需要，也可以借助算术图像运算以部分透明的方式来实现。

F_VN_CopyImageRegionToRegion [▶ 780] 函数接收两个图像和两个图像区域，并将其中一个图像区域的内容复制到另一图像区域。两个图像区域的大小相同，并且必须完全呈现在图像中。示例中使用该函数将红色图像复制到混合图像。不过，要复制蓝色图像，需要使用具有相同功能的程序化版本。该版本的优点是不需要复制也可以将蓝色和混合图像混合。

在该示例中，可以使用 `aPositionRed` 和 `aPositionBlue` 变量改变两个复制图像的位置。`bCopyBlue` 参数决定是复制还是混合蓝色图像。

变量

```
hr                :   HRESULT;

ipImageMerge      :   ITcVnImage;
ipImageRed        :   ITcVnImage;
ipImageBlue       :   ITcVnImage;
ipImageMergeDisp  :   ITcVnDisplayableImage;

aBlack            :   TcVnVector4_LREAL := [0, 0, 0];
aRed              :   TcVnVector4_LREAL := [255, 0, 0];
aBlue             :   TcVnVector4_LREAL := [0, 0, 255];

aPositionRed      :   TcVnPoint := [20, 20];
aPositionBlue     :   TcVnPoint := [480, 480];
bCopyBlue         :   BOOL := FALSE;
```

程序

```
// create images & set colors
hr := F_VN_CreateImageAndSetPixels(ipImageMerge, 1000, 1000, TCVN_ET_USINT, 3, aBlack, hr);
hr := F_VN_CreateImageAndSetPixels(ipImageRed, 500, 500, TCVN_ET_USINT, 3, aRed, hr);
hr := F_VN_CreateImageAndSetPixels(ipImageBlue, 500, 500, TCVN_ET_USINT, 3, aBlue, hr);

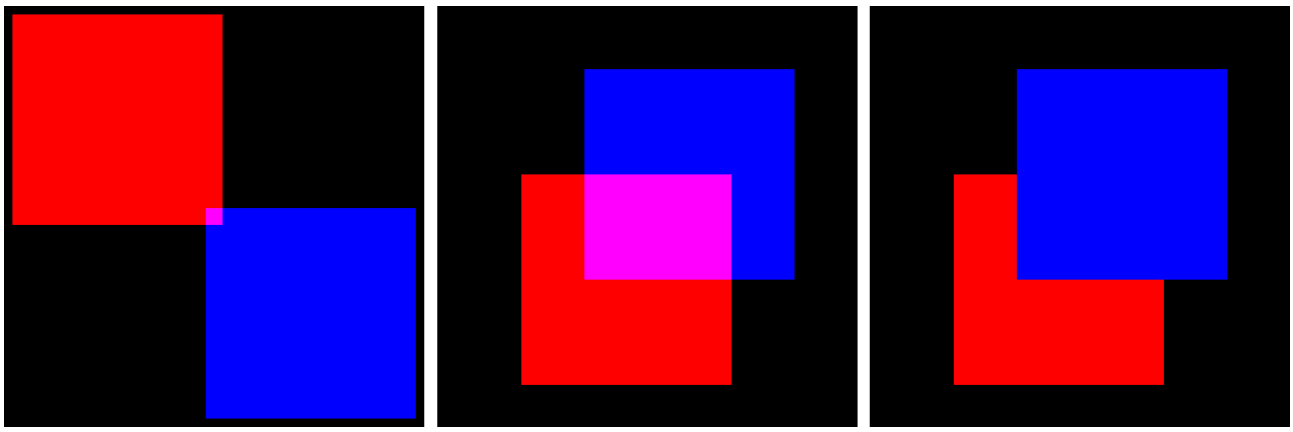
// Copy red image to selected region in merge-image
hr := F_VN_CopyImageRegionToRegion(
    ipSrcImage  :=  ipImageRed,
    nXSrc       :=  0,      nYSrc   :=  0,
    nWidth      :=  500,    nHeight :=  500,
    ipDestImage :=  ipImageMerge,
    nXDest      :=  TO_UDINT(aPositionRed[0]),
    nYDest      :=  TO_UDINT(aPositionRed[1]),
    hrPrev      :=  hr
);

// Copy or mix blue image to selected region in merge-image
hr := F_VN_SetRoi(TO_UDINT(aPositionBlue[0]), TO_UDINT(aPositionBlue[1]), 500, 500, ipImageMerge,
hr);
IF bCopyBlue THEN
    hr := F_VN_CopyImage(ipImageBlue, ipImageMerge, hr);
ELSE
    hr := F_VN_AddImages(ipImageBlue, ipImageMerge, ipImageMerge, hr);
END_IF
hr := F_VN_ResetRoi(ipImageMerge, hr);

hr := F_VN_TransformIntoDisplayableImage(ipImageMerge, ipImageMergeDisp, hr);
```

结果

以下是程序结果例程。在左边和中间的图像中，添加了蓝色方块，在右边图像中，复制了蓝色方块。此外，方块的目标位置也不同。



类似例程

融合多个图像 [▶ 2725]

8.1.3 编码质量

本示例展示了如何根据 ISO/IEC 标准评估各种一维条码和二维码的质量。为此，执行以下操作：

- 函数 F_VN_GradeBarcodeExp [▶ 824]、F_VN_GradeDataMatrixCodeExp [▶ 829] 或 F_VN_GradeQRCodeExp [▶ 834] 用于对每个图像中的一个编码进行评估。
- 函数的执行时间通过 Watchdog（看门狗）监控。
- 系统会生成各种（可选的）可视化结果，以便进行快速或详细分析。

说明

编码分级是根据国际标准对一维条码和二维码的打印质量进行评估的过程。该流程用于确保不同的读取器在不同的条件下都能够可靠地解码。质量评估遵循特定的 ISO 标准：一维条码依据 ISO/IEC 15416:2016 标准，二维码依据 ISO/IEC 15415:2011 标准。评估采用 0 至 4 的等级评分，其中 0 表示编码质量极差，4 表示编码质量极好。

要获得符合标准且具有可比性的分级结果，必须满足 ISO 标准中规定的条件。这包括对光源、光圈、分辨率和测量距离的特定要求。不过，分级函数也可用于在非严格标准条件下进行质量检查。在这种情况下，必须确保输入图像清晰、光照均匀，编码位于图像中央且完整，并尽可能避免透视几何畸变。尽管此时的结果不再具有标准意义上的可比性，但仍可用于评估编码质量，例如在生产环节中检查是否达到所需的最低质量要求。

分级函数会在指定的图像区域内搜索单个编码，并根据相应标准中定义的准则评估质量。此外，它们还会返回已找到的编码的轮廓和旋转角度。

务必向这些函数传输一个单通道 8 位灰度图像。此外，编码必须位于所传输图像区域的中心位置。如果编码不在图像中心，则必须事先使用其他函数确定其位置，并通过设置感兴趣区域 [▶ 150]（ROI）进行相应调整。

对于二维码函数，`fModuleWidth` 参数定义了预期的最小模块宽度（以像素为单位）。该值应尽可能与图像中的实际尺寸相符。要求绝对最小尺寸为 3 个像素。为获得最佳效果，建议使用 5 至 8 个像素的范围。与实际模块宽度存在较大偏差会降低识别率，并显著增加函数的执行时间。

分级

在本示例中，编码类型根据文件名中找到的关键词（如“Code128”、“EAN13”、“DM”或“QR”）进行识别。随后会调用相应的分级函数。使用分级函数的专家变体 `F_VN_GradeBarcodeExp`、`F_VN_GradeDataMatrixCodeExp` 和 `F_VN_GradeQRCodeExp`。这些函数提供了额外的详细信息及可视化结果，例如扫描线、模块评估和编码轮廓。参数 `ipResultVisualization`、`ipScanLineInfo` 和 `ipMarginGrades` 为可选参数，可设置为 0，从而缩短函数的处理时间。函数的执行由 Watchdog（看门狗）监控，以避免循环超时。

```
hrWD := F_VN_StartRelWatchdog(tStop, S_OK);
// Grade code based on code type
CASE eCodeType OF
    BARCODE:
```

```
        // Grade 1D barcode
        hrGrade := F_VN_GradeBarcodeExp(
                        ipSrcImage              :=  ipImageIn,
                        ipDecodedData           :=  ipDecodedData,
                        stCodeGrades            :=  stCodeGrades1D,
                        eBarcodeType            :=  eBarcodeType,
                        eSearchDirection        :=  eSearchDirection,
                        ipResultVisualization   :=  ipResultVisualization,
                        ipScanLineInfo          :=  ipScanLineInfo,
                        ipContour               :=  ipContour,
                        hrPrev                  :=  hr,
                        fAngleDeg               =>  fRotationAngleDeg);

        // Get overall result grade
        fGradeOverallResult := stCodeGrades1D.fOverall;

    DATAMATRIX:
        // Set module with according to image resoulution
        IF INT_TO_BOOL(Find(sFileName, '1')) OR INT_TO_BOOL(Find(sFileName, '2')) THEN
            fModuleWidth := 4;
        ELSE
            fModuleWidth := 9;
        END_IF

        // Grade 2D data matrix code
        hrGrade := F_VN_GradeDataMatrixCodeExp(
                        ipSrcImage              :=  ipImageIn,
                        ipDecodedData           :=  ipDecodedData,
                        stCodeGrades            :=  stCodeGradesDM,
                        fModuleWidth            :=  fModuleWidth,
                        ipResultVisualization   :=  ipResultVisualization,
                        ipMarginGrades          :=  ipMarginGrades,
                        ipContour               :=  ipContour,
                        hrPrev                  :=  hr,
                        fAngleDeg               =>  fRotationAngleDeg);

        // Get overall result grade
        fGradeOverallResult := USINT_TO_REAL(stCodeGradesDM.nOverall);

    QRCODE:
        // Set module with according to image resoulution
        fModuleWidth := 4;

        // Grade 2D QR code
        hrGrade := F_VN_GradeQRCodeExp(
                        ipSrcImage              :=  ipImageIn,
                        ipDecodedData           :=  ipDecodedData,
                        stCodeGrades            :=  stCodeGradesQR,
                        fModuleWidth            :=  fModuleWidth,
                        ipResultVisualization   :=  ipResultVisualization,
                        ipMarginGrades          :=  ipMarginGrades,
                        ipContour               :=  ipContour,
                        hrPrev                  :=  hr,
                        fAngleDeg               =>  fRotationAngleDeg);
        // Get overall result grade
        fGradeOverallResult := USINT_TO_REAL(stCodeGradesQR.nOverall);

    ELSE
        // Unknown code type, skip grading
        hrGrade := Tc2_System.E_HRESULTAdsErr.NOTFOUND;
END_CASE
hrWD := F_VN_StopWatchdog(hrWD, tRest => tRest);
```

评估和可视化

函数调用成功后，即可对分级结果进行评估。使用参数 `ipDecodedData` 和 `ipContour`，可以像读码 [▶ 2731]一样读取编码并显示编码轮廓。`fAngleDeg` 可选择指定编码在图像中的旋转。

```
IF SUCCEEDED(hrGrade) THEN
    // Export decoded data to string
    hr := F_VN_ExportContainer_String(ipDecodedData, sCodeResult, 255, hr);

    // Create code result image with contour and rotation angle
    hr := F_VN_ConvertColorSpace(ipImageIn, ipImageCodeResult, TCVN_CST_GRAY_TO_RGB, hr);
    hr := F_VN_DrawContours(ipContour, 0, ipImageCodeResult, aColorGreen, 2, hr);

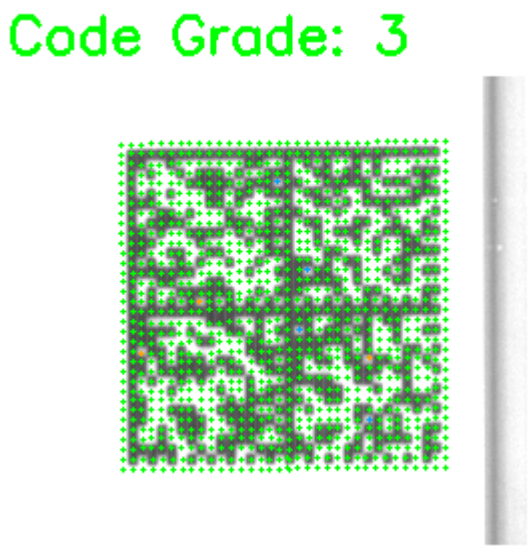
    sText := CONCAT('Rotation angle: ', REAL_TO_STRING(fRotationAngleDeg));
    hr := F_VN_PutTextExp(sText, ipImageCodeResult, 5, 10, TCVN_FT_HERSHEY_SIMPLEX, 0.4, aColorBlue,
1, TCVN_LT_8_CONNECTED, FALSE, hr);

    // Further processing …
END_IF
```



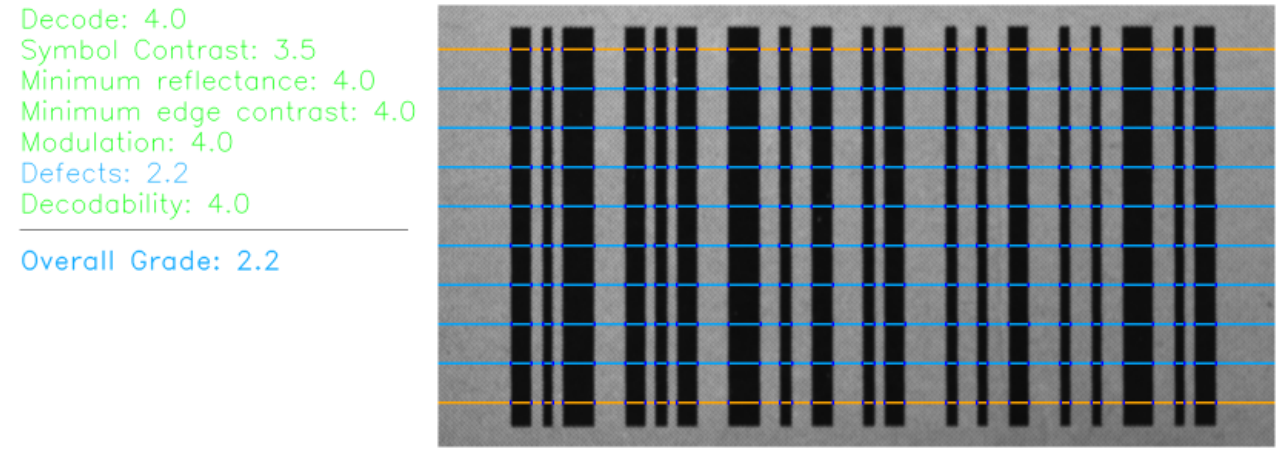
分级结果保存在结构 `stCodeGrades1D`、`stCodeGradesDM` 或 `stCodeGradesQR` 中，包含总体评分以及所有单项评分。

本示例包含各种辅助函数，用于在 POU/Visualization 下可视化结果。默认情况下，辅助函数 `F_VisualizeGradingResult` 用于生成结果图像 `ipImageGradeResult`，该图像在分级函数生成的图像 `ipResultVisualization` 上方显示总体评估结果：



分级函数的输出图像 `ipResultVisualization` 以颜色代码形式显示各区域的评估结果：绿色（≥ 3.0）、蓝色（≥ 2.0）、橙色（≥ 1.0）和红色（< 1.0）。对于一维条码，各条扫描线是彩色的；对于二维码，各个模块是彩色的。这样可以在编码上对问题区域（例如缺陷、调制不良或对比度不足）进行空间定位。

参数 `bEnableDetailedResultImage` 可用于切换到更详细的视图。辅助函数 `F_VisualizeDetailedGradingResults` 用于生成一个图像，左侧显示所有分级参数的单个数值，右侧显示图像 `ipResultVisualization`：



为了进行更深入的分析，还可以对分级函数生成的 `ipScanLineInfo` 和 `ipMarginGrades` 容器进行评估。

在所示示例中，对于一维条码，`F_VisualizeScanLineIntensityProfile` 辅助函数用于生成所选搜索线沿扫描方向的强度分布图，并将其显示在 `ipImageCustomVisualization` 中：



该强度分布图代表之前显示的条形码的第一条搜索线。这解释了缺陷等级被降级为 2.2 的原因。强度分布图显示，由于背景杂乱，白色区域内存在强烈的噪声。与符号对比度相比，该噪声强度过高，导致分级结果大幅降级，从而拉低了总体评分。

对于二维码，`ipImageCustomVisualization` 中的 `F_VisualizeMarginGrades` 辅助函数会将所有等级低于 3 的编码模块进行可视化：



8.1.4 代码读取

8.1.4.1 编码检测

该示例展示了使用神经网络对一维条码和二维码进行定位和分类的过程。它演示了以下函数的使用：

- F_VN_DetectCodesNeuralNetwork [▶ 839] 用于检测一维条码和/或二维码。

- F_VN_ResizeRegion_TcVnRectangle_DINT [▶ 1155] 用于调整所找到区域的大小，以进行后续处理步骤。
- 对所找到的编码区域和编码类型进行可视化。

说明

F_VN_DetectCodesNeuralNetwork 函数用于定位一维条码和二维码，并以矩形形式返回可能的编码区域以及相关的编码类型。这种方法在背景不稳定、光源发生变化或图像中存在多个编码的情况下尤为有利，因为后续的读码函数只需在已识别的部分区域上执行即可。由于需要检查的像素数量减少，这通常会降低读取函数在寻找编码时所需的计算量。

在示例中，参数 eModelType 可用于在运行时切换不同的神经网络模型（ETcVnCodeDetectionModel [▶ 164]）。由于这些模型在输入分辨率和检测范围上有所不同，因此可以对识别精度与执行速度进行取舍评估。不过，如果图像中的结构（尤其是在缩放后的输入图像中）与编码相似，也可能会导致混淆。

这些模型从两个维度进行系统划分：

- 识别范围（前缀）：
  - TCVN_CDM_DETECT_**1**_x（一维条码和二维码 [QR、DATAMATRIX 和 DOTCODE]）
  - TCVN_CDM_DETECT_**2**_x（仅一维条码）
  - TCVN_CDM_DETECT_**3**_x（仅二维码）
- 输入分辨率（后缀）：
  - TCVN_CDM_DETECT_x_**1**（224×224 像素，速度最快）
  - TCVN_CDM_DETECT_x_**2**（352×256 像素，平衡）
  - TCVN_CDM_DETECT_x_**3**（416×352 像素，对小编码的识别精度最高）

该函数会在内部将输入图像自动缩放至相应的目标分辨率。

此外，变量 eCodeTypeSelection 还可用于定义在所选模型范围内应搜索哪些特定的编码类型。要同时筛选多个选定类型，可以通过按位 OR 运算（例如 eCodeTypeSelection := TCVN_CT_QR OR TCVN_CT_DATAMATRIX）将这些枚举连接起来。

模型初始化

在执行编码检测之前，必须先加载模型。使用功能块 FB_VN_InitializeFunction [▶ 1584]. 启动程序时，可执行一次该操作

如果模型要在运行时更改或不再需要，可以使用 F_VN_DeinitializeFunction [▶ 1522] 对模型进行去初始化，以便再次释放内存。

变量

```
bInitialized    :   BOOL := FALSE;
eModelType      :   ETcVnCodeDetectionModel := TCVN_CDM_DETECT_1_1;
fbInit          :   FB_VN_InitializeFunction;
nInitReturnCode :   UDINT;
```

代码

```
// Load code detection model
IF NOT bInitialized THEN
    fbInit(eFunction := TCVN_IF_CODEDETECTION, nOptions := eModelType, bStart := TRUE);

    IF NOT fbInit.bBusy THEN
        fbInit(bStart := FALSE);
        IF NOT fbInit.bError THEN
            bInitialized := TRUE;
            nInitReturnCode := fbInit.nErrorId AND 16#FFF;
        ELSE
            nInitReturnCode := fbInit.nErrorId AND 16#FFF;
        END_IF
    END_IF
END_IF
```

编码区域的定位

初始化成功后，使用函数 F_VN_DetectCodesNeuralNetwork 对当前输入图像执行循环检测。该函数返回两个容器。一方面，ipCodeRegions 包含所找到区域（以矩形表示）的坐标；另一方面，ipCodeTypes 包含相应的已识别编码类型。

变量：

```
ipCodeRegions        :   ITcVnContainer;
ipCodeTypeResult     :   ITcVnContainer;
eCodeTypeSelection   :   ETcVnCodeType := TCVN_CT_ANY;
```

代码：

```
hr := F_VN_DetectCodesNeuralNetwork(
        ipSrcImage      := ipImageIn,
        ipCodeRegions   := ipCodeRegions,
        ipCodeTypes     := ipCodeTypeResult,
        eModelType      := eModelType,
        nCodeType       := eCodeTypeSelection,
        hrPrev          := hr);
```

区域的评估与调整

如果函数执行成功且至少已找到一个编码，则遍历所有检测到的元素，以检索各个位置和类型。

由于模型输出的条码区域通常非常精确且紧贴编码边缘，因此会对结果应用函数 F_VN_ResizeRegion_TcVnRectangle_DINT。这样可以通过一个可调整的系数（fWidthRatio、fHeightRatio）和一个绝对最小像素偏移量（nMinWidthOffset、nMinHeightOffset）来扩大矩形区域。这对于后续的读码函数至关重要，可以确保所需的安静区完全包含在图像部分中。

最后，CASE 语句对所找到的编码类型进行评估，以便根据类型分配特定的颜色和标签用于可视化展示，如示例中所示。在此处或后续程序中，可以利用这些信息添加相应的读码函数。

变量：

```
fWidthRatio          :   REAL := 1.15;
fHeightRatio         :   REAL := 1.15;
nMinWidthOffset      :   DINT := 10;
nMinHeightOffse      :   DINT := 10;
hr                   :   HRESULT;
ipImageIn            :   ITcVnImage;
eCodeTypeResult      :   ETcVnCodeType;
stRectangle          :   TcVnRectangle;
nDetectedCodes       :   ULINT;
i                    :   ULINT;
```

代码：

```
IF hr = S_OK THEN
    // Get number of detecded codes
    hr := F_VN_GetNumberOfElements(ipCodeRegions, nDetectedCodes, hr);

    IF nDetectedCodes > 0 THEN
        FOR i := 0 TO nDetectedCodes - 1 DO
            // Get code type and region from detection results
            hr := F_VN_GetAt_ULINT(ipCodeTypeResult, eCodeTypeResult, i, hr);
            hr := F_VN_GetAt_TcVnRectangle_DINT(ipCodeRegions, stRectangle, i, hr);

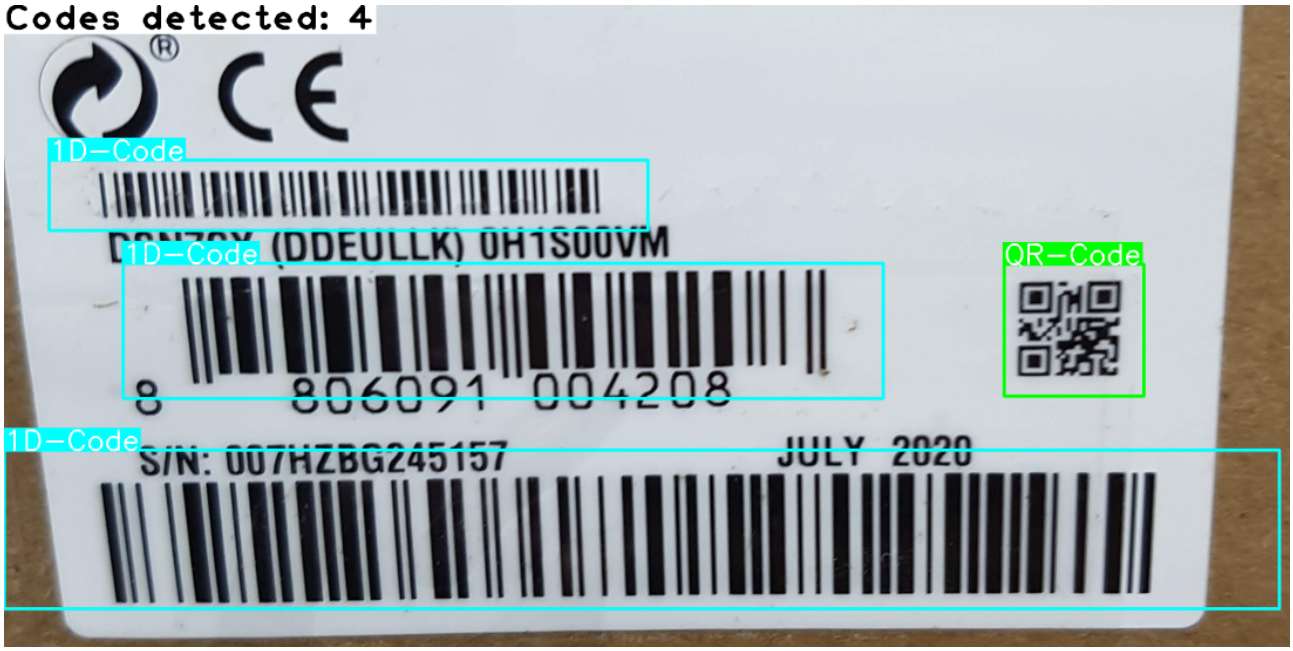
            // Enlarge detection region to meet the respective code requirements for the quiet zone
            hr := F_VN_ResizeRegion_TcVnRectangle_DINT(
                    stSrcRect           := stRectangle,
                    stDestRect          := stRectangle,
                    fWidthRatio         := fWidthRatio,
                    fHeightRatio        := fHeightRatio,
                    nMinWidthOffset     := nMinWidthOffset,
                    nMinHeightOffset    := nMinHeightOffset,
                    ipImage             := ipImageIn,
                    hrPrev              := hr);

            // Evaluate the recognized code type and its region
            CASE eCodeTypeResult OF
                TCVN_CT_1D:
                    // ...
                TCVN_CT_2D:
                    // ...
                TCVN_CT_QR
                    // ...
                TCVN_CT_DATAMATRIX:
                    // ...
                TCVN_CT_DOTCODE:
                    // ...
            END_CASE
        END_FOR
```

```
    ELSE
        // No code detected
    END_IF
ELSE
    IF hr = S_FALSE THEN
        // No code detected
    ELSE
        // Error: See HRESULT for more details
    END_IF
END_IF
```

结果

定位结果在 ipImageResultDisp 图像中以彩色的矩形以及可选的类型标签显示。此外，左上角会显示状态文本，例如“Codes detected: n（检测到编码：n）”或“No code detected（未检测到编码）”，如果出现错误，则显示 HRESULT 代码。所示图像使用 TCVN_CDM_DETECT_1_2 模型进行分析：



为了便于调试，该示例还会生成 ipImageModelInputDisp 图像。它显示了输入图像在经过函数内部缩放至所选模型分辨率时的情况。这使您能够直观地看到神经网络作为输入实际处理的细节和长宽比。

默认情况下，该示例使用 TCVN_CDM_DETECT_1_1 模型（224×224 像素）。在编码非常小或输入图像的长宽比不利的情况下，内部缩小至该模型尺寸可能会导致细小结构丢失。因此，可能无法检测到编码或编码分类不正确。例如，当尺寸大幅缩小时，几乎无法从视觉上区分QR码和数据矩阵码，这在某些情况下会导致分类错误。

为了在这些情况下优化检测和分类精度，有两种可行的解决方案（也可组合使用）：

- 更高的模型分辨率：将参数 eModelType 更改为具有更大输入尺寸的模型（例如 TCVN_CDM_DETECT_1_2 或 TCVN_CDM_DETECT_1_3），可为神经网络提供更多细节。这会使类型的正确分类更加可靠，但同时也会增加执行时间。
- 使用 ROI [▶ 150]：在检测之前在输入图像上设置 ROI，可以缩小图像部分。这可以使长宽比更接近模型输入的格式，同时也能提高模型的有效像素分辨率。理想情况下，应根据模型所使用的长宽比来设置 ROI。这样可以避免或至少最大限度地减少编码的几何畸变或压缩。这通常可以提高识别率，而无需使用计算量更大的模型。

8.1.4.2 EAN-13 条形码读取

该样本说明了以下内容

- 使用函数F_VN_ReadBarcodeExp [▶ 842]，每个图像读取一个 EAN13 代码
- 函数的执行时间通过看门狗 [▶ 140]进行监控。

说明

- 函数F_VN_ReadBarcode [▶ 842]/F_VN_ReadBarcodeExp [▶ 842]可以搜索和读取图像中不同类型的条形码。如需加快处理时间，可以指定条形码类型（eBarcodeType）。在这个例子中，这些为TCVN_BT_EAN13。此外，处理时间可以通过看门狗进行监控。样本带看门狗监控的斑点检测 [▶ 2755]提供了关于看门狗的更全面的信息。
- 与F_VN_ReadBarcode [▶ 842]相比，本示例中使用的函数F_VN_ReadBarcodeExp [▶ 842]具有以下特点：
  - 除了条形码类型外，还可以指定搜索方向（eSearchDirection）：
    - TCVN_BSD_ANY 首先在水平方向搜索，然后在垂直方向搜索
    - TCVN_BSD_HORIZONTAL 只在水平方向搜索
    - TCVN_BSD_VERTICAL 只在垂直方向搜索
  - 可以返回进行读取的代码部分。

变量

```
hr                          :   HRESULT;

ipImageIn                   :   ITcVnImage;
ipImageInDsply              :   ITcVnDisplayableImage;

ipImageRes                  :   ITcVnImage;
ipImageResDsply             :   ITcVnDisplayableImage;

// Barcode
ipCodeDecodedList           :   ITcVnContainer;
ipCodeContourList           :   ITcVnContainer;
sCodeAsString               :   STRING(255);
eBarcodeSearchDirection     :   ETcVnBarcodeSearchDirection := TCVN_BSD_ANY;
eBarcodeType                :   ETcVnBarcodeType := TCVN_BT_EAN13;

// Watchdog
hrWD                        :   HRESULT;
tStop                       :   DINT := 50000;
tRest                       :   DINT;

// Output
sText                       :   STRING;

// Color
aColorRed                   :   TcVnVector4_LREAL := [255, 0, 0];
```

代码

```
// Execute the Barcode Reading Function with EAN13 selected monitored by the Watchdog-Function
// ------------------------------------------------------------
hrWD := F_VN_StartRelWatchdog(tStop, S_OK);
    hr := F_VN_ReadBarcodeExp(
                ipSrcImage          :=  ipImageIn,
                ipDecodedData       :=  ipCodeDecodedList,
                ipContours          :=  ipCodeContourList,
                eBarcodeType        :=  eBarcodeType,
                nCodeNumber         :=  1,
                eSearchDirection    :=  eBarcodeSearchDirection,
                hrPrev              :=  hr);
hrWD := F_VN_StopWatchdog(hrWD, tRest => tRest);

// Check if the function was executed successfully
IF hr = S_OK THEN
    // Export Code into String
    hr := F_VN_ExportSubContainer_String(ipCodeDecodedList, 0, sCodeAsString, 255, hr);

    // Write Code into Result Image
    hr := F_VN_PutTextExp(sCodeAsString, ipImageRes, 50, 100, ETcVnFontType.TCVN_FT_HERSHEY_PLAIN,
5, aColorRed,3, TCVN_LT_4_CONNECTED, FALSE, hr);

    // Draw Code Contour into Result Image
    hr := F_VN_DrawContours(ipCodeContourList, 0, ipImageRes, aColorRed, 3, hr);
ELSE
    // Write HRESULT into Result Image
    sText := CONCAT('Returncode ', DINT_TO_STRING(hr));
    hr := F_VN_PutTextExp(sText, ipImageRes, 50, 100, ETcVnFontType.TCVN_FT_HERSHEY_PLAIN, 5,
aColorRed,3, TCVN_LT_4_CONNECTED, FALSE, hr);
END_IF

// Write Code Reading proceeded time into Result Image
sText := CONCAT(CONCAT('Time: ', DINT_TO_STRING(tStop - tRest)), 'us');
hr := F_VN_PutTextExp(sText, ipImageRes, 50, 200, ETcVnFontType.TCVN_FT_HERSHEY_PLAIN, 5,
aColorRed,3, TCVN_LT_4_CONNECTED, FALSE, hr);
```

结果

如果选择 TCVN_BSD_ANY 为搜索方向，水平和垂直 EAN13 代码都将被检测到。然而，当比较看门狗的时间时，就会发现水平 EAN13 码（第一张图片）的识别速度比垂直 EAN13 码（第二张图片）快得多，因为代码搜索最初在水平方向进行。





如果搜索方向设置为 TCVN_BSD_VERTICAL，则垂直 EAN13 码的识别速度会快得多（第三张图片）。然而，在这种情况下，没有搜索水平码。



类似样本

- 数据矩阵码读取 [▶ 2737]
- 药品编码读取 [▶ 2739]
- QR 码读取 [▶ 2741]
- 读码过程中的结果评估 [▶ 2745]

8.1.4.3 数据矩阵码读取

该样本说明了以下内容

- 对于每张图片，使用函数F_VN_ReadDataMatrixCodeExp [▶ 852]读取数据矩阵码。
- 函数的执行时间通过看门狗 [▶ 140]进行监控。

说明

函数F_VN_ReadDataMatrixCodeExp [▶ 852]在指定的图像中搜索数据矩阵码并进行读取。与F_VN_ReadDataMatrixCode [▶ 850]函数相比，可以指定每个图像需要搜索的代码数量。在这个样本中，每个图像一个代码。此外，还会返回找到的代码轮廓。TCVN_CSS_ONLY_NOT_FLIPPED被选为搜索策略，因为该图像没有被镜像。

如果图像中存在与代码相似的区域，或者图像中根本没有代码，这对执行时间会产生负面影响。为了防止周期时间超限，在这个样本中，函数F_VN_ReadDataMatrixCodeExp [▶ 852]通过看门狗 [▶ 140]进行监控（见带看门狗监控的斑点检测 [▶ 2755]上的样本）。

变量

```
hr                  :    HRESULT;

ipImageIn           :    ITcVnImage;
ipImageInDisp       :    ITcVnDisplayableImage;

ipImageRes          :    ITcVnImage;
ipImageResDisp      :    ITcVnDisplayableImage;

// Data Matrix Code
ipCodeDecodedList   :  ITcVnContainer;
```

```
ipCodeContourList    :  ITcVnContainer;
sCodeAsString        :  STRING(255);

// Watchdog
hrWD                 :    HRESULT;
tStop                :    DINT := 20000;
tRest                :    DINT;

// Output
sText                :    STRING;

// Color
aColorRed            :    TcVnVector4_LREAL := [255, 0, 0];
```

代码

```
// Execute the DMC Code Reading Function monitored by the Watchdog function
// ------------------------------------------------------------------
hrWD := F_VN_StartRelWatchdog(tStop, S_OK);
    hr := F_VN_ReadDataMatrixCodeExp(
    ipSrcImage    := ipImageIn,
    ipDecodedData  := ipCodeDecodedList,
    ipContours    := ipCodeContourList,
    nCodeNumber    := 1,
    eSearchStrategy := TCVN_CSS_ONLY_NOT_FLIPPED,
    hrPrev      := hr
  );
hrWD := F_VN_StopWatchdog(hrWD, tRest => tRest);

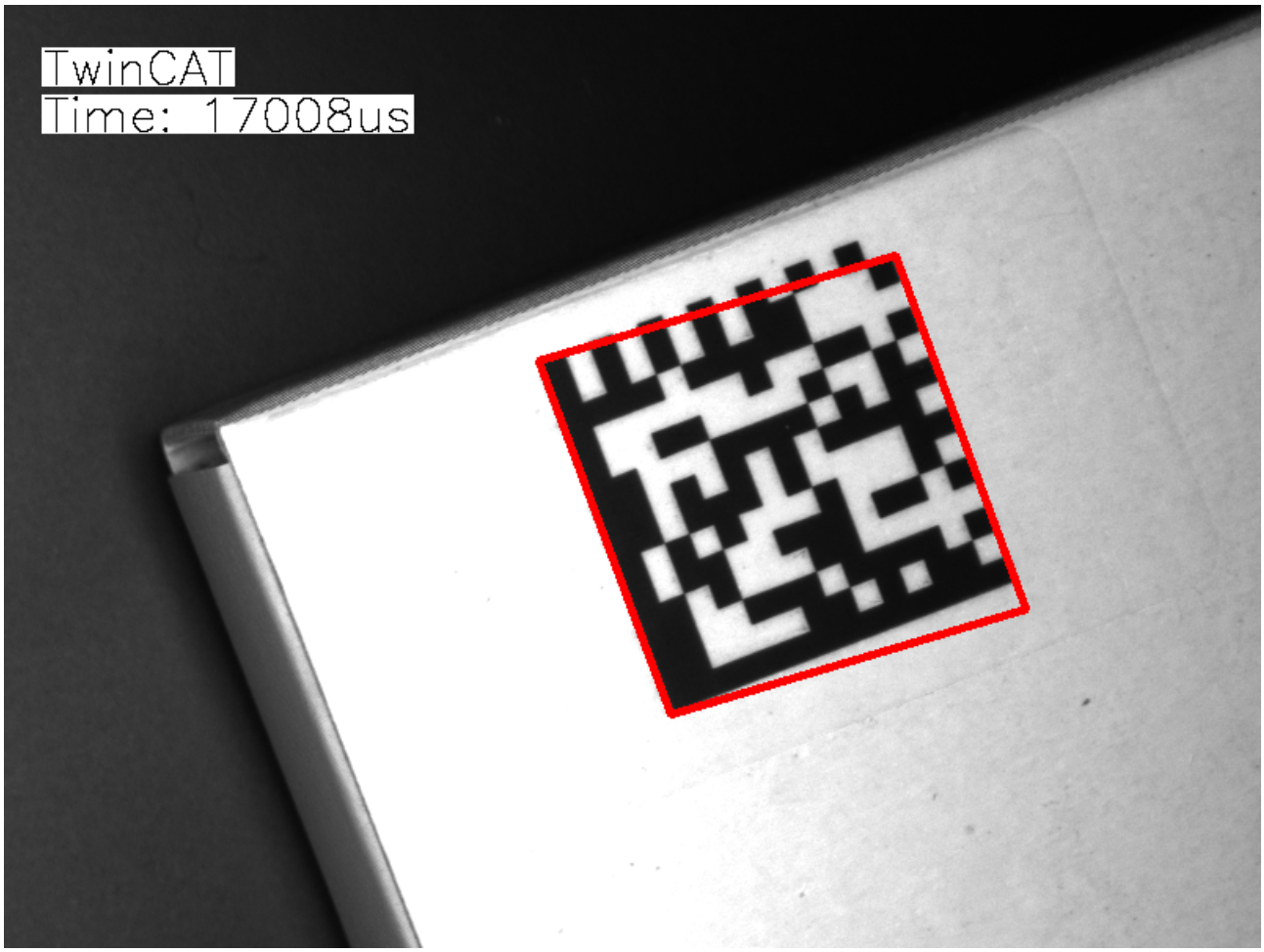
// Check if the function was executed successfully
IF hr = S_OK THEN
    // Export Code into String
    hr := F_VN_ExportSubContainer_String(ipCodeDecodedList, 0, sCodeAsString, 255, hr);

    // Write Code into Result Image
    hr := F_VN_PutText(sCodeAsString, ipImageRes, 25, 50, ETcVnFontType.TCVN_FT_HERSHEY_PLAIN, 2,
aColorRed, hr);

    // Draw Code Contour into Result Image
    hr := F_VN_DrawContours(ipCodeContourList, 0, ipImageRes, aColorRed, 3, hr);
ELSE
    // Write HRESULT into Result Image
    sText := CONCAT('Returncode ', DINT_TO_STRING(hr));
    hr := F_VN_PutLabelExp(sText, ipImageRes, 25, 50, 2, 2, ETcVnFontType.TCVN_FT_HERSHEY_PLAIN,
aColorRed, aColorWhite, ETcVnLineType.TCVN_LT_8_CONNECTED, hr);
END_IF

// Write Code Reading proceeded time into Result Image
sText := CONCAT(CONCAT('Time: ', DINT_TO_STRING(tStop - tRest)), 'us');
hr := F_VN_PutText(sText, ipImageRes, 25, 80, ETcVnFontType.TCVN_FT_HERSHEY_PLAIN, 2, aColorRed,
hr);
```

结果



类似样本

- EAN-13 条形码读取 [▶ 2734]
- 药品编码读取 [▶ 2739]
- QR 码读取 [▶ 2741]
- 读码过程中的结果评估 [▶ 2745]

8.1.4.4 药品编码读取

该样本说明了以下内容

- 通过函数F_VN_ReadPharmaCodeExp [▶ 861]，每个图像读取一个药品代码。
- 函数的执行时间通过看门狗 [▶ 140]进行监控。

说明

- 函数F_VN_ReadPharmaCodeExp [▶ 861]在指定的图像中搜索药品代码并进行读取。与函数 F_VN_ReadPharmaCode [▶ 861]相比，还会返回代码位置。
- 类似代码的结构对执行时间有负面的影响。为了防止周期时间超限，在样本中，函数 F_VN_ReadPharmaCodeExp [▶ 861]通过看门狗 [▶ 140]进行监控。样本带看门狗监控的斑点检测 [▶ 2755]解释了如何使用看门狗 [▶ 140]。

变量

```
hr                  :    HRESULT;

ipImageIn           :    ITcVnImage;
ipImageInDisp       :    ITcVnDisplayableImage;
```

```
ipImageRes          :    ITcVnImage;
ipImageResDisp      :    ITcVnDisplayableImage;

// Pharma-Code
ipCodeDecodedList   :  ITcVnContainer;
ipCodeContourList   :  ITcVnContainer;
sCodeAsString       :  STRING(255);
aCodeContour    :    ARRAY[0..3] OF TcVnPoint2_DINT;

// Watchdog
hrWD                :    HRESULT;
tStop               :    DINT := 5000;
tRest               :    DINT;

// Output
sText               :    STRING;

// Color
aColorRed           :    TcVnVector4_LREAL := [255, 0, 0];
```

代码

```
// Execute the Pharma-Code Reading Function monitored by the Watchdog-Function
// ---------------------------------------------------------------
hrWD := F_VN_StartRelWatchdog(tStop, S_OK);
  hr := F_VN_ReadPharmaCodeExp(
          ipSrcImage    := ipImageIn,
          ipDecodedData  := ipCodeDecodedList,
          ipContours    := ipCodeContourList,
          nCodeNumber   := 1,
          nMinBarNumber  := 4,
          hrPrev      := hr);
hrWD := F_VN_StopWatchdog(hrWD, tRest => tRest);

// Check if the function was executed successfully
IF hr = S_OK THEN
    // Export Code into String
    hr := F_VN_ExportSubContainer_String(ipCodeDecodedList, 0, sCodeAsString, 255, hr);

    // Write Code into Result Image
    sText := CONCAT('Code: ', sCodeAsString);
    hr := F_VN_PutTextExp(sText, ipImageRes, 25, 50, ETcVnFontType.TCVN_FT_HERSHEY_PLAIN, 3,
aColorRed, 2, TCVN_LT_4_CONNECTED, FALSE, hr);

    // Draw Code Contour into Result Image
    hr := F_VN_DrawContours(ipCodeContourList, 0, ipImageRes, aColorRed, 3, hr);
ELSE
    // Write HRESULT into Result Image
    sText := CONCAT('Returncode ', DINT_TO_STRING(hr));
    hr := F_VN_PutTextExp(sText, ipImageRes, 25, 50, ETcVnFontType.TCVN_FT_HERSHEY_PLAIN, 3,
aColorRed,2, TCVN_LT_4_CONNECTED, FALSE, hr);
END_IF

// Write Code Reading proceeded time into Result Image
sText := CONCAT(CONCAT('Time: ', DINT_TO_STRING(tStop - tRest)), 'us');
hr := F_VN_PutTextExp(sText, ipImageRes, 25, 100, ETcVnFontType.TCVN_FT_HERSHEY_PLAIN, 3,
aColorRed,2, TCVN_LT_4_CONNECTED, FALSE, hr);
```

结果



类似样本

- 数据矩阵码读取 [▶ 2737]
- EAN-13 条形码读取 [▶ 2734]
- QR 码读取 [▶ 2741]
- 读码过程中的结果评估 [▶ 2745]

8.1.4.5 QR 码读取

该样本说明了以下内容

- 通过函数F_VN_ReadQRCodeExp，每个图像读取一个 QR 码。
- 函数的执行时间通过看门狗 [▶ 140]进行监控。

说明

函数F_VN_ReadQRCodeExp在指定的图像中搜索 QR 码并进行读取。与函数F_VN_ReadQRCode [▶ 864]相比，还会返回找到代码的轮廓。

如果图像中存在与代码相似的区域，或者图像中根本没有代码，这对执行时间会产生负面影响。为了防止周期时间超限，在这个样本中，函数F_VN_ReadQRCodeExp [▶ 866]通过看门狗 [▶ 140]进行监控（见带看门狗监控的斑点检测 [▶ 2755]上的样本）。

变量

```
hr                  :    HRESULT;

ipImageIn           :    ITcVnImage;
ipImageInDisp       :    ITcVnDisplayableImage;
```

```
ipImageRes          :    ITcVnImage;
ipImageResDisp      :    ITcVnDisplayableImage;

// QR Code
ipCodeDecodedList   :  ITcVnContainer;
ipCodeContourList   :  ITcVnContainer;
sCodeAsString       :  STRING(255);

// Watchdog
hrWD                :    HRESULT;
tStop               :    DINT := 20000;
tRest               :    DINT;

// Output
sText               :    STRING;

// Color
aColorRed           :    TcVnVector4_LREAL := [255, 0, 0];
```

代码

```
// Execute the QR Code Reading Function monitored by the Watchdog-Function
// ---------------------------------------------------------------
hrWD := F_VN_StartRelWatchdog(tStop, S_OK);
  hr := F_VN_ReadQRCodeExp(
      ipSrcImage    := ipImageIn,
      ipDecodedData  := ipCodeDecodedList,
      ipContours    := ipCodeContourList,
      nCodeNumber   := 1,
      eSearchStrategy := TCVN_CSS_ONLY_NOT_INVERTED + TCVN_CSS_ONLY_NOT_FLIPPED,
      hrPrev      := hr
  );
hrWD := F_VN_StopWatchdog(hrWD, tRest => tRest);

// Check if the function was executed successfully
IF hr = S_OK THEN
    // Export Code into String
    hr := F_VN_ExportSubContainer_String(ipCodeDecodedList, 0, sCodeAsString, 255, hr);

    // Write Code into Result Image
    hr := F_VN_PutTextExp(sCodeAsString, ipImageRes, 50, 100, ETcVnFontType.TCVN_FT_HERSHEY_PLAIN,
5, aColorRed,3, TCVN_LT_4_CONNECTED, FALSE, hr);

    // Draw Code Contour into Result Image
    hr := F_VN_DrawContours(ipCodeContourList, 0, ipImageRes, aColorRed, 3, hr);
ELSE
    // Write HRESULT into Result Image
    sText := CONCAT('Returncode ', DINT_TO_STRING(hr));
    hr := F_VN_PutTextExp(sText, ipImageRes, 50, 100, ETcVnFontType.TCVN_FT_HERSHEY_PLAIN, 5,
aColorRed,3, TCVN_LT_4_CONNECTED, FALSE, hr);
END_IF

// Write Code Reading proceeded time into Result Image
sText := CONCAT(CONCAT('Time: ', DINT_TO_STRING(tStop - tRest)), 'us');
hr := F_VN_PutTextExp(sText, ipImageRes, 50, 200, ETcVnFontType.TCVN_FT_HERSHEY_PLAIN, 5,
aColorRed,3, TCVN_LT_4_CONNECTED, FALSE, hr);
```

结果



类似样本

- 数据矩阵码读取 [▶ 2737]
- EAN-13 条形码读取 [▶ 2734]
- 药品编码读取 [▶ 2739]
- 读码过程中的结果评估 [▶ 2745]

8.1.4.6 DotCode（点码）读取

本示例演示以下内容

- 通过函数 F_VN_ReadDotCodeRoiExp [▶ 859] 读取每个图像的单个DotCode（点码）
- 通过 Watchdog（看门狗） [▶ 140]监控函数的执行时间。

说明

`F_VN_ReadDotCodeRoiExp` 函数在指定的图像区域内搜索单个DotCode(点码)并对其进行解码。此外，它还会返回已找到的编码的轮廓和旋转角度。

务必向该函数传输一个单通道 8 位灰度图像。此外，要读取的DotCode(点码)必须位于所传输图像区域的中心位置。如果编码不在图像中心，则必须事先使用其他函数确定其位置，并通过设置感兴趣区域 [▶ 150]（ROI）进行相应调整。

`fModuleWidth` 参数定义了预期的点直径（以像素为单位）。该值应尽可能与图像中的实际直径相符。要求绝对最小尺寸为 3 个像素。为获得最佳效果，建议使用 5 至 8 个像素的范围。与实际模块宽度存在较大偏差会降低识别率，并显著增加函数的执行时间。

此外，使用 `eOptions` 参数（ETcVnDotCodeOptions [▶ 188]）可以根据具体情况调整搜索策略。例如，`TCVN_DCO_BINARY_INPUT` 可用于预二值化图像，`TCVN_DCO_REGULAR_COLOR` / `TCVN_DCO_INVERTED_COLOR` 可用于专门搜索深色或浅色编码。

为了避免循环超时，使用 Watchdog（看门狗）对示例中的函数进行监控。

变量

```
// DotCode reading
fModuleWidth       : REAL := 4.0;
eOptions           : ULINT := TCVN_DCO_DEFAULT;
ipCodeDecoded      : ITcVnContainer;
ipCodeContour      : ITcVnContainer;
sCodeAsString      : STRING(255);
fCodeAngleDeg      : REAL;

// Watchdog
hrWD                        :   HRESULT;
tStop                       :   DINT := 20000;
tRest                       :   DINT;

// Output
sText                       :   STRING;
```

代码

```
// Execute the DotCode reading function monitored by the Watchdog function
hrWD := F_VN_StartRelWatchdog(tStop, S_OK);
  hr := F_VN_ReadDotCodeRoiExp(
        ipSrcImage    := ipImageIn,
        ipDecodedData  := ipCodeDecoded,
        fModuleWidth   := fModuleWidth,
        ipContour     := ipCodeContour,
        eOptions      := eOptions,
        hrPrev        := hr,
        fAngleDeg     => fCodeAngleDeg);
hrWD := F_VN_StopWatchdog(hrWD, tRest => tRest);

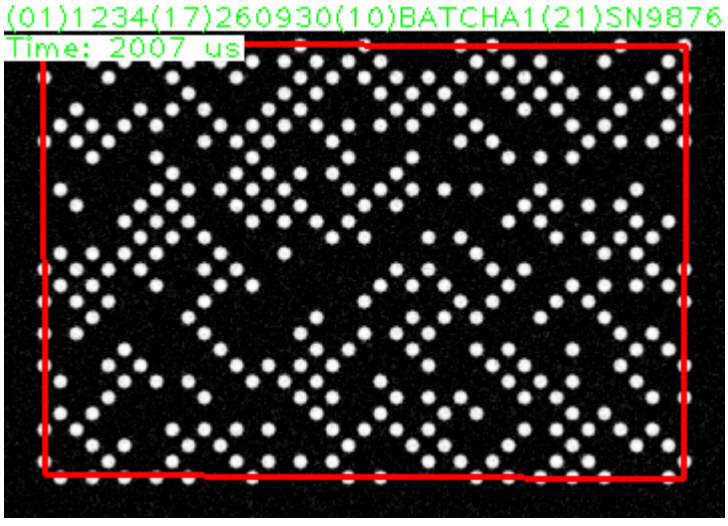
// Check if the function was executed successfully
IF hr = S_OK THEN
    // Export code into string
    hr := F_VN_ExportContainer_String(ipCodeDecoded, sCodeAsString, 255, hr);

    // Draw code contour into result image
    hr := F_VN_DrawContours(ipCodeContour, 0, ipImageRes, aColorRed, 2, hr);

    // Write code into result image
    hr := F_VN_PutLabelExp(sCodeAsString, ipImageRes, 0, 11, 1, 1,
ETcVnFontType.TCVN_FT_HERSHEY_PLAIN, aColorGreen, aColorWhite, ETcVnLineType.TCVN_LT_8_CONNECTED,
S_OK);
ELSE
    // Write HRESULT into result image
    sText := CONCAT('Returncode ', DINT_TO_STRING(hr));
    hr := F_VN_PutLabelExp(sText, ipImageRes, 0, 11, 1, 1, ETcVnFontType.TCVN_FT_HERSHEY_PLAIN,
aColorRed, aColorWhite, ETcVnLineType.TCVN_LT_8_CONNECTED, S_OK);
END_IF

// Write Code Reading proceeded time into Result Image
sText := CONCAT(CONCAT('Time: ', DINT_TO_STRING(tStop - tRest)), ' us');
hr := F_VN_PutLabelExp(sText, ipImageRes, 0, 26, 1, 1, ETcVnFontType.TCVN_FT_HERSHEY_PLAIN,
aColorGreen, aColorWhite, ETcVnLineType.TCVN_LT_8_CONNECTED, S_OK);
```

结果



8.1.4.7 读码过程中的结果评估

本示例演示以下内容

- 使用函数 F_VN_ReadBarcode [▶ 842] 读取每个图像中的一个 EAN13 编码。
- 函数的执行时间通过 Watchdog（看门狗） [▶ 140]监控。
- 函数的不同返回代码的处理方法。

说明

- 函数 F_VN_ReadBarcode [▶ 842] 可以搜索并读取图像中不同类型的条形码。有关该函数使用方法的详细信息，请参见示例 EAN-13 条形码读取 [▶ 2734]。
- 使用相应的查询语句，说明了如何处理函数的返回代码。以下情况之间有所区别：
  - 函数执行成功，找到编码并解码。
  - 函数执行成功，但图像中不存在编码，或者无法找到编码/解码。
  - 函数本身没有错误，但被 Watchdog（看门狗）中止。
  - 函数有其他错误。

有关读码函数的返回代码的详细说明，请参见概述章节 。

变量

```
hr                          :   HRESULT;

// Images
ipImageIn                   :   ITcVnImage;
ipImageInDisp               :   ITcVnDisplayableImage;
ipImageRes                  :   ITcVnImage;
ipImageResDisp              :   ITcVnDisplayableImage;

// Barcode
ipCodeDecodedList           :   ITcVnContainer;
ipCodeContourList           :   ITcVnContainer;
sCodeAsString               :   STRING(255);

// Watchdog
hrWD                        :   HRESULT;
tStop                       :   DINT := 50000;
tRest                       :   DINT;

// Color
aColorGreen                 :   TcVnVector4_LREAL := [0, 255, 0];

// Return code
nReturnCode                 :   DWORD;
sReturnCode                 :   STRING;
sResultText                 :   STRING;
```

代码

```
// Execute the Barcode Reading Function with EAN13 selected monitored by the Watchdog-Function
hrWD := F_VN_StartRelWatchdog(tStop, S_OK);
    hr := F_VN_ReadBarcode(
        ipSrcImage      :=  ipImageIn,
        ipDecodedData   :=  ipCodeDecodedList,
        eBarcodeType    :=  ETcVnBarcodeType.TCVN_BT_EAN13,
        hrPrev          :=  hr
    );
hrWD := F_VN_StopWatchdog(hrWD, tRest => tRest);

// Handle return-code
IF hr = S_OK THEN
    // Prepare code result in string
    hr := F_VN_ExportSubContainer_String(ipCodeDecodedList, 0, sCodeAsString, 255, hr);
    sResultText := CONCAT('Code: ', sCodeAsString);
ELSE
    // Check for succeeded return codes or add specific error handling
    CASE hr OF
        S_FALSE:
            sResultText := 'No code found...';

        S_WATCHDOG_TIMEOUT:
            sResultText := 'Cancelled by watchdog...';
    ELSE
   // Extract error-code from HRESULT & react accordingly
   nReturnCode := DINT_TO_DWORD(hr) AND 16#FFF;
```

```
        sReturnCode := DWORD_TO_HEXSTR(nReturnCode, 3, FALSE);
        sResultText := CONCAT('Returncode ', sReturnCode);
    END_CASE
END_IF

// Draw result image
hr := F_VN_PutTextExp(sResultText, ipImageRes, 50, 100, ETcVnFontType.TCVN_FT_HERSHEY_PLAIN, 4,
aColorGreen, 3, TCVN_LT_4_CONNECTED, FALSE, S_OK);
hr := F_VN_TransformIntoDisplayableImage(ipImageRes, ipImageResDisp, hr);
```

注意

预留 hr

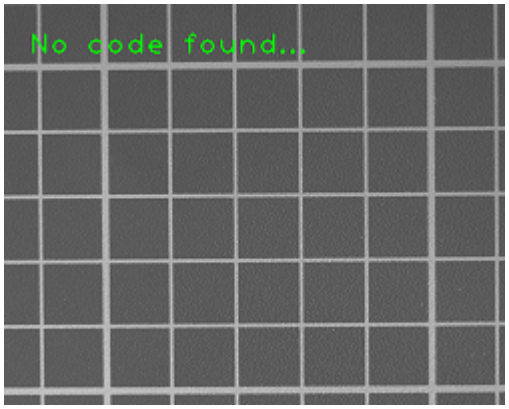
确保在调用读码函数与评估之间不会将变量 hr 用作其他函数的返回值。

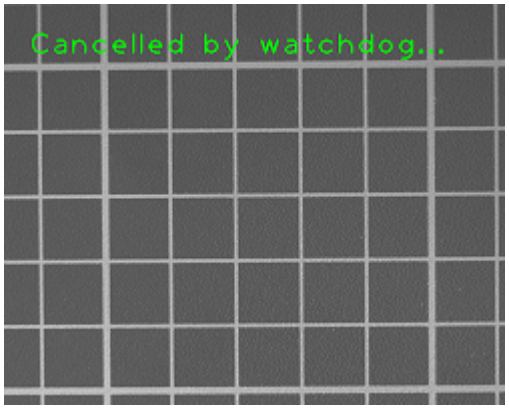
结果

如果按预期找到编码，则为 `hr = S_OK`，并可输出结果。



如果未能找到预期编码，则返回代码为 `S_FALSE`。如果 Watchdog（看门狗） [▶ 140]中止执行，则代码为 `16#256`。系统会输出相应的消息。





如果未正确执行函数，则可输出错误代码。在示例中不会出现这种情况，只有在错误使用读码函数时才会出现这种情况。

类似例程

- 数据矩阵码读取 [▶ 2737]
- EAN-13 条形码读取 [▶ 2734]
- Pharma编码读取 [▶ 2739]
- QR 码读取 [▶ 2741]

8.1.5 轮廓分析

8.1.5.1 匹配轮廓 1vsN（手动形状）

在这个样本中，使用函数F_VN_MatchContours1vsN [▶ 989]，将轮廓与参考轮廓进行比较。其中

- 参考轮廓是理想化的、手动创建的多边形
- 其他轮廓是从图像中提取出来的。

说明

函数F_VN_MatchContours [▶ 988]根据 Hu 矩比较样本中使用的两个轮廓。函数F_VN_MatchContours1vsN [▶ 989]代表一个扩展，可用于同时比较一个轮廓和其他几个轮廓。在这个样本中，这用于根据预定的参考形状对工件进行分类。

参考形状是理想化的、手动创建的多边形，包括等腰三角形、矩形/方形和梯形。由于只指定了多边形的角点，参考形状与测试轮廓在描述性轮廓点的数量上有很大不同。此外，它们在位置、方向、大小和轮廓内的干扰程度上也有不同。

函数F_VN_DetectBlobs [▶ 1171]用于搜索具有一定最小尺寸的轮廓，以便从图像中提取部件的形状。找到的轮廓然后用于描述的匹配。

输入参数

除了待比较轮廓外，只需向函数传递一个参数，描述用于确定 Hu 矩的异同的计算方法（**eComparisonMethod**，类型ETcVnContoursMatchComparisonMethod [▶ 184]）。方法 `TCVN_CMCM_CONTOURS_MATCH_I1`计算各个特征的倒数值的差异之和，而方法 `TCVN_CMCM_CONTOURS_MATCH_I2`计算纯特征的差异之和。与前两种方法不同，第三种方法 `TCVN_CMCM_CONTOURS_MATCH_I3`只计算个体特征之间的最大差异。这三种方法中哪一种最适合于比较两个轮廓取决于使用情况。

变量

仅在本样本中的数据类型

CustomTcVnArray4_Point2_DINT 数据类型并不包含在 TwinCAT Vision 中，而是为该样本明确创建。执行方法可以在该样本的下载中找到。

```
// Indexes representing the different shapes
TRIANGLE              :   USINT := 0;
RECTANGLE             :   USINT := 1;
TRAPEZOID             :   USINT := 2;

// Images
ipImageIn             :   ITcVnImage;
ipImageInDisp         :   ITcVnDisplayableImage;
ipImageRes            :   ITcVnImage;
ipImageResDisp        :   ITcVnDisplayableImage;

// Contours
ipContourList         :   ITcVnContainer;
ipContour             :   ITcVnContainer;
ipIterator            :   ITcVnForwardIterator;
aShapeArrays          :   ARRAY [0..2] OF CustomTcVnArray4_Point2_DINT :=
                              [[[0,0],[40,0],[20, LREAL_TO_DINT(SQRT(1200))], [0,0]],  // Triangle
                               [[0,0],[40,0],[40,40],[0,40]],                          // Rectangle
                               [[0,0],[40,0],[60,22],[20,22]]];                        // Trapezoid;
aShapes               :   ARRAY[0..2] OF ITcVnContainer;
ipShapes              :   ITcVnContainer;

// Matching
```

```
ipMatchIndexes        :   ITcVnContainer;
ipDissimilarities     :   ITcVnContainer;
aMatchIndexes         :   ARRAY [0..2] OF ULINT;
aDissimilarities      :   ARRAY [0..2] OF LREAL;

// Parameters
stBlobParams          :   TcVnParamsBlobDetection;
fThreshold            :   REAL := 170;
fMinArea              :   REAL := 10000;
fMaxDissimilarity     :   LREAL := 0.02;
eComparisonMethod     :   ETcVnContoursMatchComparisonMethod := TCVN_CMCM_CONTOURS_MATCH_I3;

// drawing
aColors               :   ARRAY[0..2] OF TcVnVector4_LREAL :=
                              [[0, 175, 0, 0], [0, 0, 255, 0], [255, 0, 0, 0]]; // green, blue, red
aColorWhite           :   TcVnVector4_LREAL := [255, 255, 255];
aColorBlack           :   TcVnVector4_LREAL := [0, 0, 0];
aTexts                :   ARRAY[0..2] OF STRING := ['Triangle', 'Rectangle', 'Trapezoid'];
sText                 :   STRING(255);
nTopLeftX             :   UDINT;
nTopLeftY             :   UDINT;

// Miscellaneous
aPixelValue           :   TcVnVector4_LREAL;
I                     :   USINT;
stBoundingRectangle   :   TcVnRectangle_UDINT;
aOffsets              :   ARRAY [0..2] OF TcVnPoint := [[20, 50], [20, 100], [20, 160]];
hr                    :   HRESULT;
```

代码

```
// Fill manually defined shapes into one container
hr := F_VN_CreateContainer(ipShapes, ContainerType_Vector_Vector_TcVnPoint2_DINT, 0, hr);
FOR i:=TRIANGLE TO TRAPEZOID DO
    hr := F_VN_CreateContainerFromArray(ADR(aShapeArrays[i]), aShapes[i],
ContainerType_Vector_TcVnPoint2_DINT, 4, hr);
    hr := F_VN_InsertIntoContainer_ITcVnContainer(aShapes[i], ipShapes, i, hr);
END_FOR

// Prepare result image
hr := F_VN_ConvertColorSpace(ipImageIn, ipImageRes, TCVN_CST_GRAY_TO_RGB, hr);

// Check if background is light or dark
hr := F_VN_GetPixel(ipImageIn, aPixelValue, 50, 50, hr);
IF SUCCEEDED(hr) AND_THEN aPixelValue[0] < 128 THEN
    stBlobParams.eThresholdType := TCVN_TT_BINARY;
ELSE
    stBlobParams.eThresholdType := TCVN_TT_BINARY_INV;
END_IF

// Find contours in image
stBlobParams.bFilterByArea := TRUE;
stBlobParams.fMinArea := fMinArea;
stBlobParams.fMinThreshold := fThreshold;
hr := F_VN_DetectBlobs(ipImageIn, ipContourList, stBlobParams, hr);

// Iterate through all found contours
hr := F_VN_GetForwardIterator(ipContourList, ipIterator, hr);
WHILE hr = S_OK AND_THEN ipIterator.CheckIfEnd() <> S_OK DO
    hr := F_VN_GetContainer(ipIterator, ipContour, hr);
    hr := F_VN_IncrementIterator(ipIterator, hr);

  // Match the current contour with the reference shapes
  hr := F_VN_MatchContours1vsN(
    ipRefContour:= ipContour,
    ipContours:= ipShapes,
    ipMatchIndexes:= ipMatchIndexes,
    ipDissimilarities:= ipDissimilarities,
    fDissimilarityThreshold:= 100,
    eComparisonMethod:= eComparisonMethod,
    hrPrev:= hr
  );

    hr := F_VN_ExportContainer(ipMatchIndexes, ADR(aMatchIndexes), SIZEOF(aMatchIndexes), hr);
    hr := F_VN_ExportContainer(ipDissimilarities, ADR(aDissimilarities), SIZEOF(aDissimilarities),
hr);

    // Draw matching results
    IF aDissimilarities[0] < fMaxDissimilarity THEN

        // Calculate position of object
        hr := F_VN_UprightBoundingRectangle(ipContour, stBoundingRectangle, hr);
        nTopLeftX := LREAL_TO_UDINT(stBoundingRectangle.nX + 30);
        nTopLeftY := LREAL_TO_UDINT(stBoundingRectangle.nY + (stBoundingRectangle.nHeight / 2));

        // Draw matching result
        hr := F_VN_DrawContours(ipContour, -1, ipImageRes, aColors[aMatchIndexes[0]], 5, hr);
        hr := F_VN_PutTextExp(aTexts[aMatchIndexes[0]], ipImageRes, nTopLeftX, nTopLeftY,
TCVN_FT_HERSHEY_SIMPLEX, 0.8, aColors[aMatchIndexes[0]], 2, TCVN_LT_8_CONNECTED, FALSE, hr);
        hr := F_VN_DrawRectangle(nTopLeftX, nTopLeftY + 5, nTopLeftX + 200, nTopLeftY + 85,
```

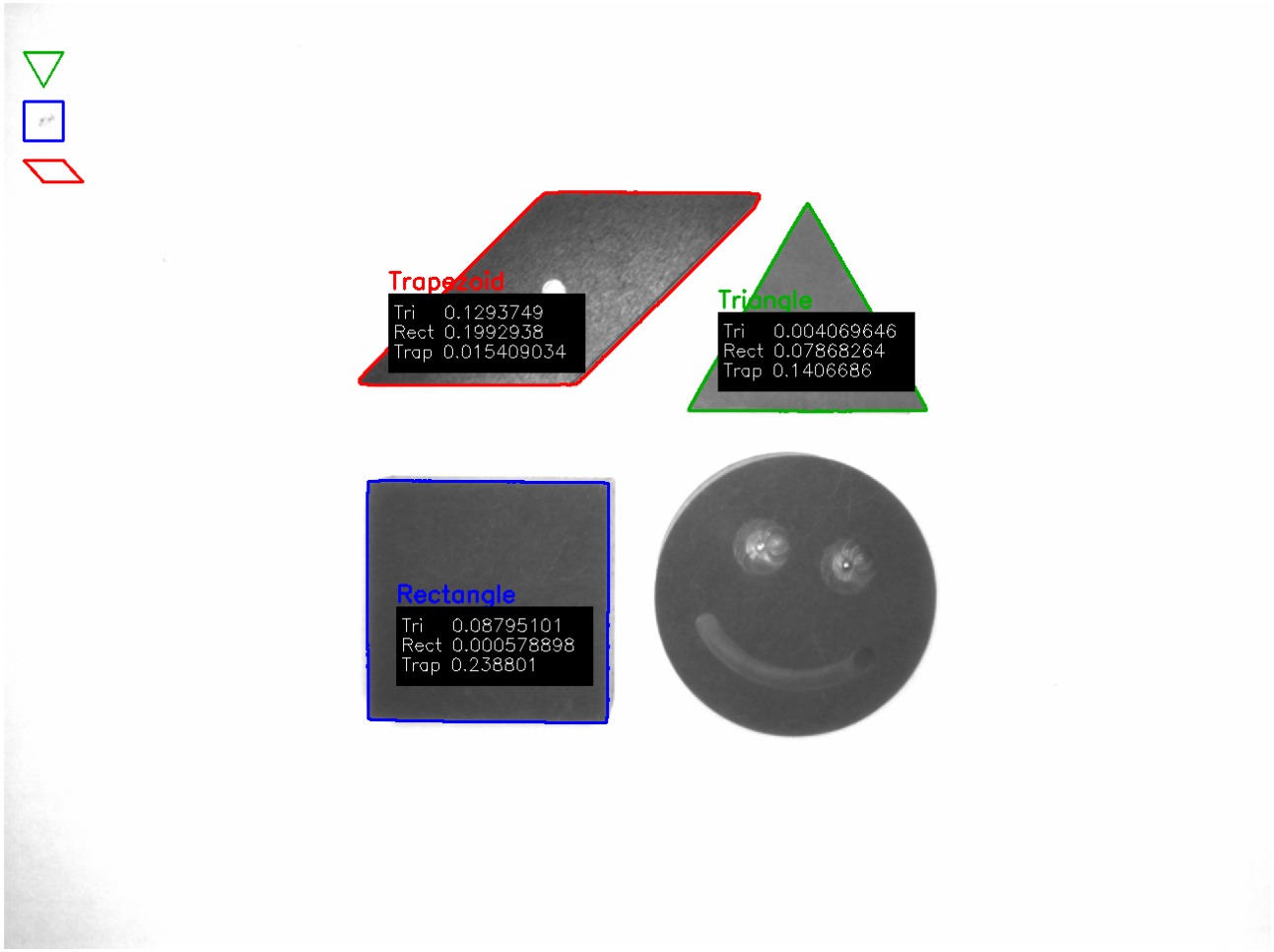
```
ipImageRes, aColorBlack, -1, hr);
        // Draw all dissimilarity values
        FOR i:=0 TO 2 DO
            sText := CONCAT(LEFT(aTexts[aMatchIndexes[i]], 4), CONCAT(' ',
REAL_TO_STRING(LREAL_TO_REAL(aDissimilarities[i]))));
            hr := F_VN_PutTextExp(sText, ipImageRes, nTopLeftX + 5, nTopLeftY + 30 + 20*i,
TCVN_FT_HERSHEY_SIMPLEX, 0.6, aColorWhite, 1, TCVN_LT_8_CONNECTED, FALSE, hr);
        END_FOR

    END_IF
END_WHILE

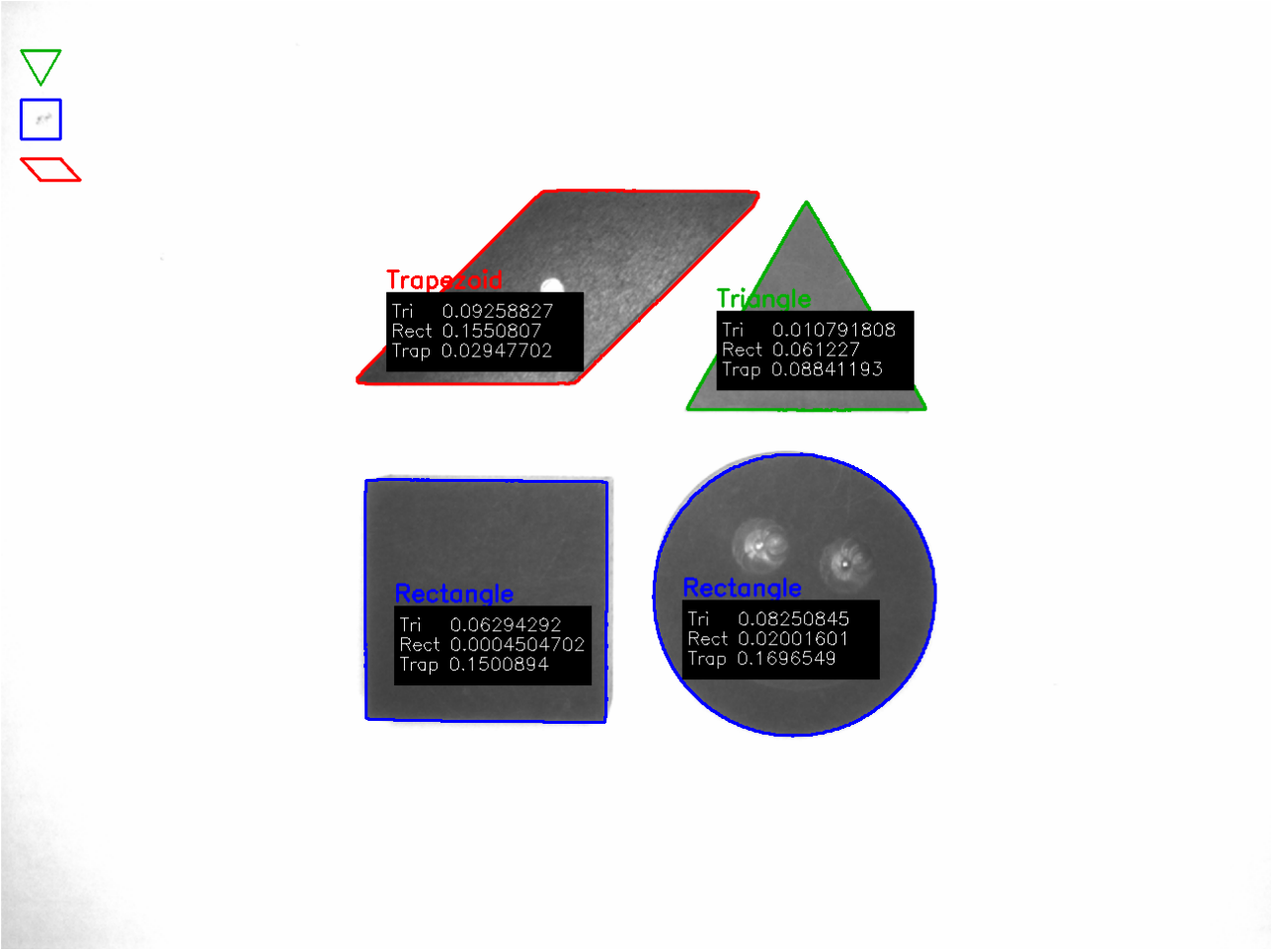
// Draw reference shapes
FOR i:=TRIANGLE TO TRAPEZOID DO
    hr := F_VN_DrawContoursExp(aShapes[i], -1, ipImageRes, aColors[i], 2, TCVN_LT_8_CONNECTED, 0, 0,
aOffsets[i], hr);
END_FOR
hr := F_VN_TransformIntoDisplayableImage(ipImageRes, ipImageResDisp, hr);
hr := F_VN_TransformIntoDisplayableImage(ipImageIn, ipImageInDisp, hr);
```

结果

结果图像显示了参考形状（左边界）和找到的轮廓，这些轮廓与所有参考形状的相异度不超过一个最大值。此外，可视化显示了轮廓与哪个参考形状最相似。相应的描述文本（包括每个参考形状的相异度）被叠加在各自的轮廓上。对于本样本中使用的参数，MatchContours1.bmp 图像的结果如下：



在每种情况下，最低值表示正确的参考轮廓。如果参数 eComparisonMethod 从 `TCVN_CMCM_CONTOURS_MATCH_I3`改为`TCVN_CMCM_CONTOURS_MATCH_I2`，结果会有很大变化。这也意味着必须调整最大接受相异度的参数：



新的数值允许对相应的参考形状进行更不明确的分配，且圆形组件被错误地分配给矩形参考形状。因此，这种计算方法显然不太适合这个样本。

类似样本

- 匹配轮廓（提取的形状） [▶ 2750]

8.1.5.2 匹配轮廓（提取的形状）

在这个样本中，使用函数F_VN_MatchContours [▶ 988]对两个轮廓进行比较。

说明

函数F_VN_MatchContours [▶ 988]根据 Hu 矩比较样本中使用的两个轮廓。在这个样本中，该功能用于在新的情况下识别一个特定的组件。为此，在参考图像中选择一个组件，保存其轮廓，然后与另一图像的轮廓进行比较。通过这种方式，与参考组件相比，低于一定相异度的组件被标记出来。

输入参数

除了待比较轮廓外，只需向函数传递一个参数，描述用于确定 Hu 矩的异同的计算方法（**eComparisonMethod**，类型ETcVnContoursMatchComparisonMethod [▶ 184]）。TCVN_CMCM_CONTOURS_MATCH_I1 方法计算各个特征的倒数值的差异之和，而 TCVN_CMCM_CONTOURS_MATCH_I2 方法计算纯粹特征的差异之和。与前两种方法不同，第三种方法 TCVN_CMCM_CONTOURS_MATCH_I3 只计算各个特征之间的最大差异。这三种方法中哪一种最适合于比较两个轮廓取决于使用情况。

变量

```
hr                      :   HRESULT;
ipImageIn               :   ITcVnImage;
```

```
ipImageInDisp           :   ITcVnDisplayableImage;

ipImageRes              :   ITcVnImage;
ipImageResDisp          :   ITcVnDisplayableImage;
ipIterator              :   ITcVnForwardIterator;

// result
ipContourList           :   ITcVnContainer;
ipContourReference      :   ITcVnContainer;
ipContourCheck          :   ITcVnContainer;
fBestDissimilarity      :   LREAL;
fDissimilarity          :   LREAL;

// parameters
fThreshold              :   REAL := 170;
fMinArea                :   REAL := 10000;
fMaxDissimilarity       :   LREAL := 0.01;
eComparisonMethod       :   ETcVnContoursMatchComparisonMethod := TCVN_CMCM_CONTOURS_MATCH_I3;

// drawing
aColorGreen             :   TcVnVector4_LREAL := [0, 175, 0];
aColorBlue              :   TcVnVector4_LREAL := [0, 0, 255];
aColorRed               :   TcVnVector4_LREAL := [255, 0, 0];
aColorRes               :   TcVnVector4_LREAL;
sText                   :   STRING(255);
sTextReference          :   STRING(255) := 'Reference contour';
sTextCheck              :   STRING(255) := 'Check dissimilarity';

// other
bDarkBackground         :   BOOL;
nContours               :   ULINT;
nCounter                :   UINT := 0;
aPixelValue             :   TcVnVector4_LREAL;
stBoundingRectangle     :   TcVnRectangle_UDINT;
stParams                :   TcVnParamsBlobDetection;
```

代码

```
// Prepare result image
hr := F_VN_ConvertColorSpace(ipImageIn, ipImageRes, TCVN_CST_GRAY_TO_RGB, hr);

// Check if background is dark in order to identify reference image
hr := F_VN_GetPixel(ipImageIn, aPixelValue, 50, 50, hr);
bDarkBackground := SUCCEEDED(hr) AND_THEN aPixelValue[0] < 128;

IF bDarkBackground THEN
    stParams.eThresholdType := TCVN_TT_BINARY;
ELSE
    stParams.eThresholdType := TCVN_TT_BINARY_INV;
END_IF

// Find contours in image
hr := F_VN_DetectBlobs(ipImageIn, ipContourList, stParams, hr);

// Distinguish reference and test image
IF bDarkBackground THEN

    // Select one of the found contours as reference for matching
    hr := F_VN_GetNumberOfElements(ipContourList, nContours, hr);
    hr := F_VN_GetAt_ITcVnContainer(ipContourList, ipContourReference, nCounter MOD nContours, hr);

    // Draw selected contour and text
    hr := F_VN_PutTextExp(sTextReference, ipImageRes, 50, 50, TCVN_FT_HERSHEY_SIMPLEX, 1.3,
aColorBlue, 2, TCVN_LT_8_CONNECTED, FALSE, hr);
    hr := F_VN_DrawContours(ipContourReference, -1, ipImageRes, aColorBlue, 5, hr);

    nCounter := nCounter + 1;

ELSE

    fBestDissimilarity := 10E300;

    // Iterate through all found contours
    hr := F_VN_GetForwardIterator(ipContourList, ipIterator, hr);
    WHILE hr = S_OK AND_THEN ipIterator.CheckIfEnd() <> S_OK DO
        hr := F_VN_GetContainer(ipIterator, ipContourCheck, hr);
        hr := F_VN_IncrementIterator(ipIterator, hr);

        // Match the current contour with the selected reference contour
    hr := F_VN_MatchContours(ipContourReference, ipContourCheck, eComparisonMethod, fDissimilarity, hr);

        // Save best result
        IF fBestDissimilarity > fDissimilarity THEN
            fBestDissimilarity := fDissimilarity;
        END_IF

        // Choose action depending on the dissimilarity of both contours
        IF SUCCEEDED(hr) AND_THEN fDissimilarity < fMaxDissimilarity THEN
            aColorRes := aColorGreen;
        ELSE
            aColorRes := aColorRed;
        END_IF
```

```
        // Draw matching results
        sText := REAL_TO_STRING(LREAL_TO_REAL(fDissimilarity));
        hr := F_VN_UprightBoundingRectangle(ipContourCheck, stBoundingRectangle, hr);
        hr := F_VN_DrawContours(ipContourCheck, -1, ipImageRes, aColorRes, 5, hr);
        hr := F_VN_PutTextExp(sText, ipImageRes, LREAL_TO_UDINT(stBoundingRectangle.nX + 30),
LREAL_TO_UDINT(stBoundingRectangle.nY + (stBoundingRectangle.nHeight / 2)), TCVN_FT_HERSHEY_SIMPLEX,
0.8, aColorRes, 2, TCVN_LT_8_CONNECTED, FALSE, hr);

    END_WHILE

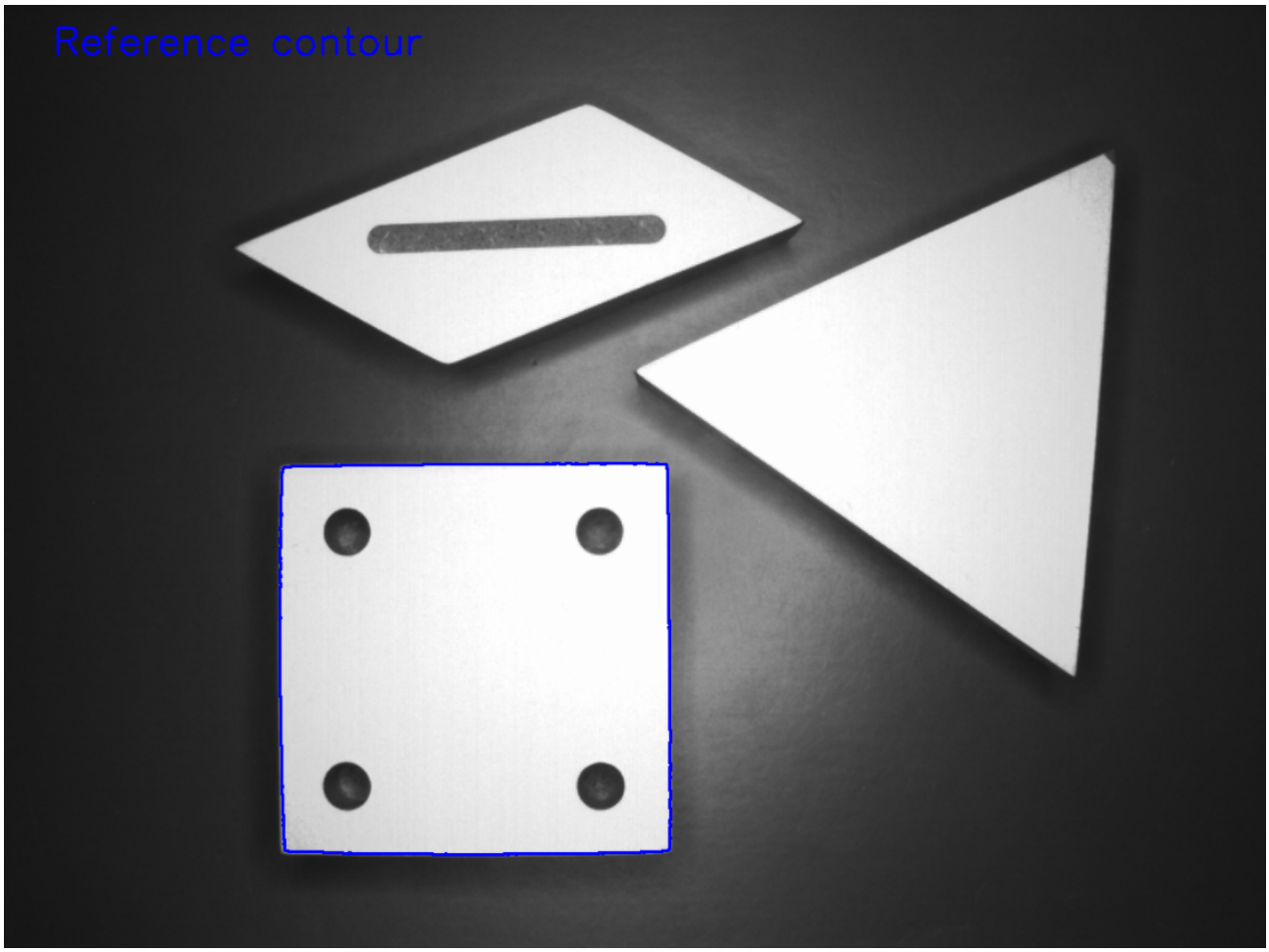
    // Draw text
    hr := F_VN_PutTextExp(sTextCheck, ipImageRes, 50, 50, TCVN_FT_HERSHEY_SIMPLEX, 1.3, aColorGreen,
2, TCVN_LT_8_CONNECTED, FALSE, hr);
    sText := CONCAT('Best match: ', REAL_TO_STRING(LREAL_TO_REAL(fBestDissimilarity)));
    hr := F_VN_PutTextExp(sText, ipImageRes, 50, 100, TCVN_FT_HERSHEY_SIMPLEX, 1.3, aColorGreen, 2,
TCVN_LT_8_CONNECTED, FALSE, hr);

END_IF

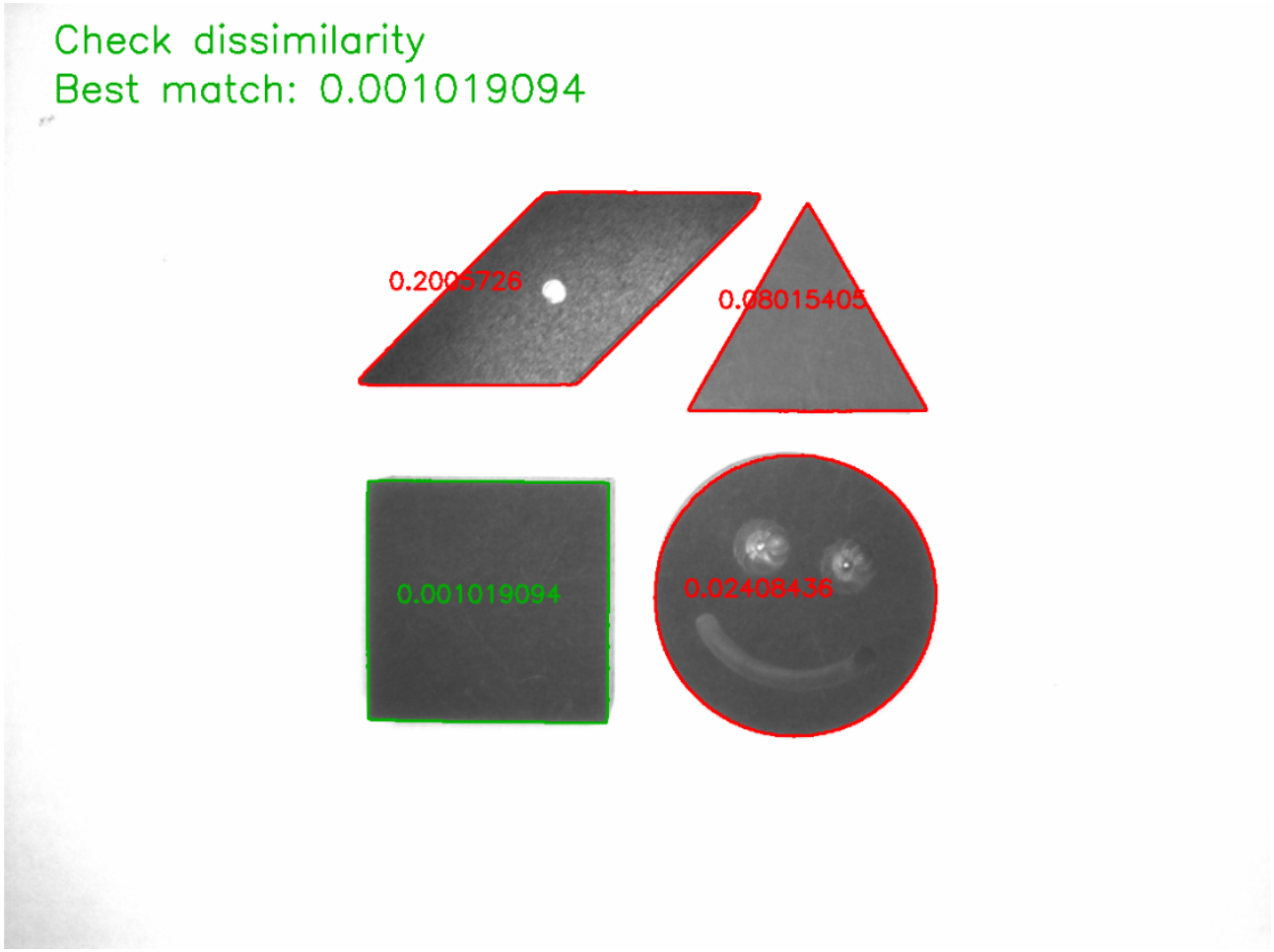
hr := F_VN_TransformIntoDisplayableImage(ipImageRes, ipImageResDisp, hr);
hr := F_VN_TransformIntoDisplayableImage(ipImageIn, ipImageInDisp, hr);
```

结果

当前选定的参考轮廓在参考图像上以蓝色标记：



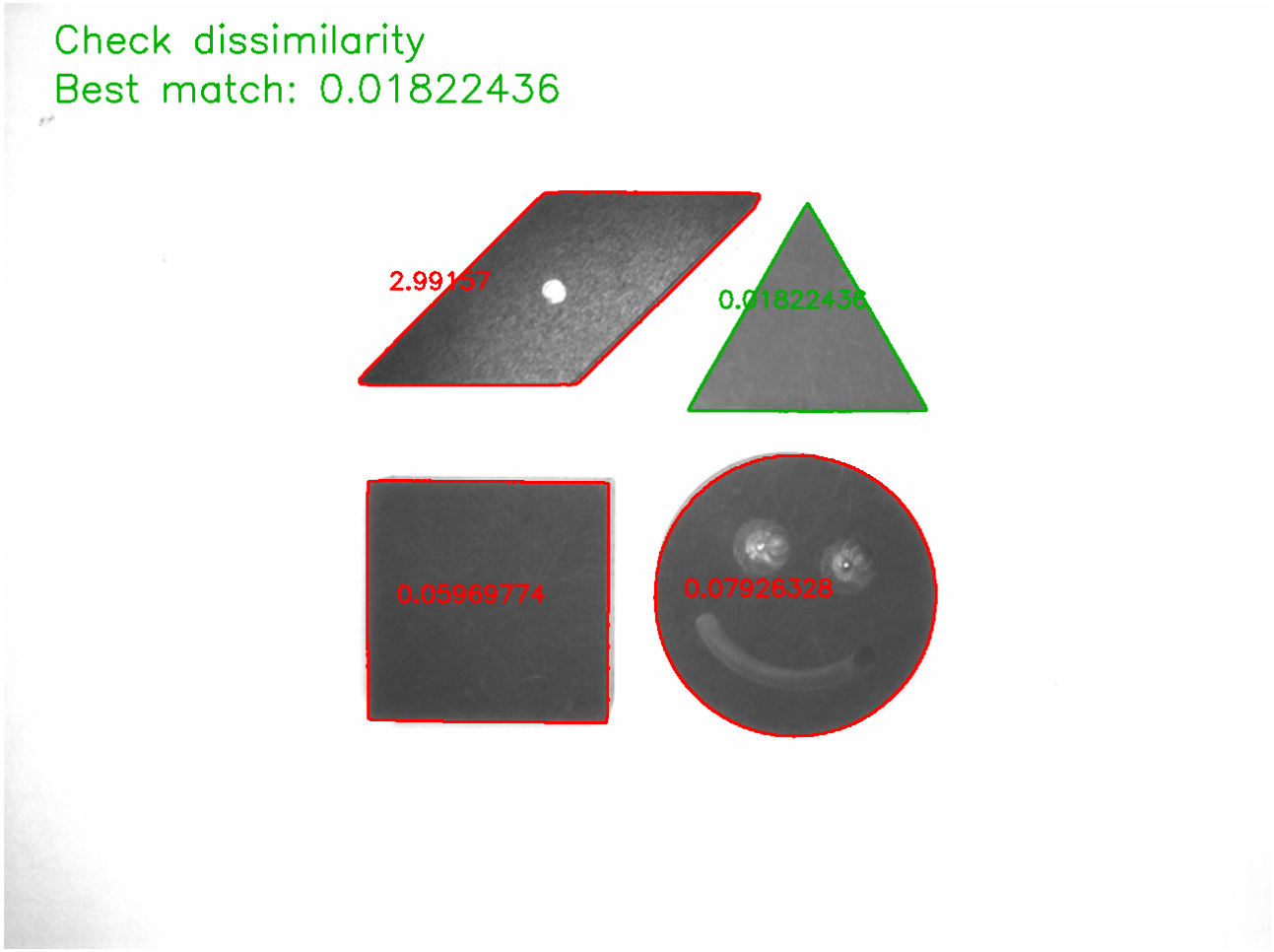
你可以看到轮廓因镜头变形而略有弯曲，并有一些小瑕疵。在测试图案中，相对于参考轮廓的相异度低于设定阈值的轮廓显示为绿色，其他轮廓显示为红色。此外，计算出的相异度被叠加在所有的轮廓上。下面的图示显示了当矩形被选为参考组件时MatchContours1.bmp输入图像的结果：



可以看出，到目前为止，与参考轮廓相比，矩形组件的相异度最小。如果参数 eComparisonMethod 从 TCVN_CMCM_CONTOURS_MATCH_I3 改为TCVN_CMCM_CONTOURS_MATCH_I2，计算出的相异度也会相应改变：



在这种情况下，结果更加清晰，但这并不适用于所有形状。如果选择三角形作为参考组件，那么这个设置的结果就明显不如原始设置的清晰：



这一结果表明，不同的计算方法可以提供最好的结果，具体情况取决于应用。

类似样本

- 匹配轮廓 1vsN（手动形状） [▶ 2747]

8.1.6 图像分析

8.1.6.1 对象检测

8.1.6.1.1 带看门狗监控的斑点检测

本示例演示以下内容

- F_VN_DetectBlobs [▶ 1171] 函数用于查找图像中的圆形结构
- 函数的执行时间通过 Watchdog（看门狗）监控。

说明

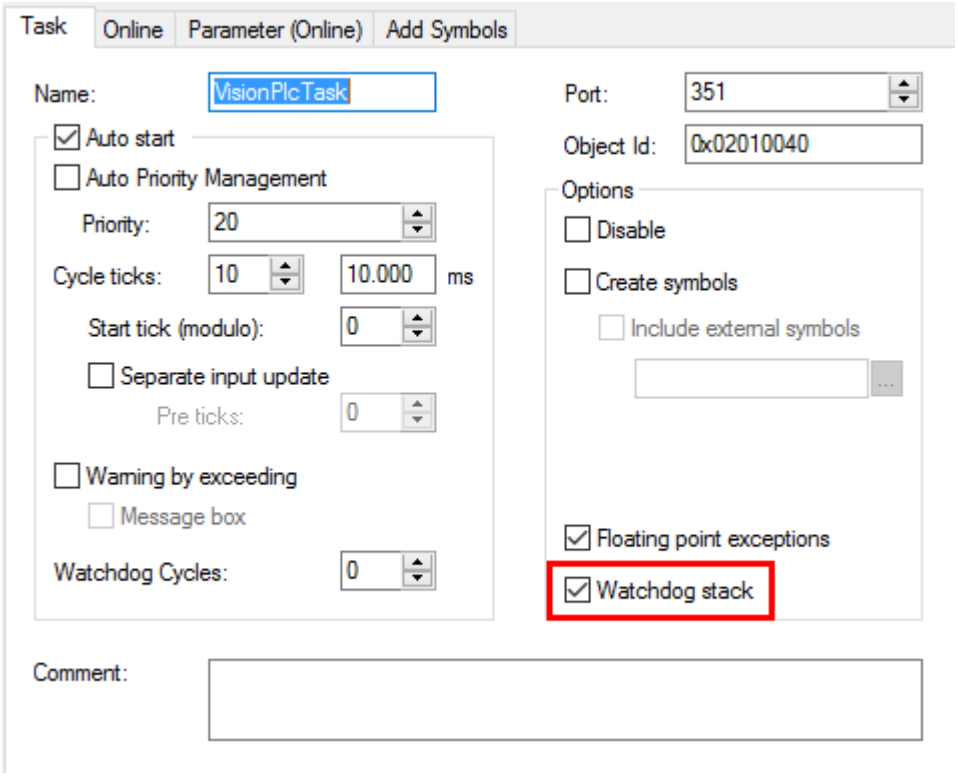
F_VN_DetectBlobs [▶ 1171] 函数便于在图像中搜索类似结构。它在一次调用中将设置阈值与轮廓搜索和滤波相结合。在本shi示例中，函数参数化为返回示例图像中的所有圆圈结构。

由于外部影响，实际图像内容可能与预期图像内容有很大偏差。对于执行时间取决于图像内容的图像处理算法来说，这会导致循环时间超限，从而产生不希望看到的副作用。为了防止出现这种情况，需要 Watchdog（看门狗）。Watchdog（看门狗）可以在指定时间后中止 API 中标有以下句子的函数：“被 Watchdog（看门狗）取消时可返回部分结果”。到此为止所获得的结果可以进一步使用。

F_VN_DetectBlobs 就是可以通过这种方式进行监控的函数之一。通过改变 F_VN_StartRelWatchdog [▶ 996] Watchdog（看门狗）启动函数中允许的 tStop 执行时间，可以观察到函数终止后应用部分结果的情况，并与 F_VN_DetectBlobs 的完全执行情况进行比较。可以通过 F_VN_StopWatchdog [▶ 998] 停止 Watchdog（看门狗）。tRest 中的正返回值表示剩余时间，而 tRest 中的负返回值表示终止所需的额外时间。

任务配置

必须在执行的 PLC 任务时启用 Watchdog（看门狗）监控 [▶ 140]。否则，PLC 中的 Watchdog（看门狗）函数调用将被忽略，启动 Watchdog（看门狗）函数将返回返回值 16#71A: "No Interface"。



变量

```
hr                  :   HRESULT;
hrFunc              :   HRESULT;

ipImageIn           :   ITcVnImage;
ipImageInDisp       :   ITcVnDisplayableImage;

ipImageRes          :   ITcVnImage;
ipImageResDisp      :   ITcVnDisplayableImage;

// Blob Detection
stBlobParams     : TcVnParamsBlobDetection;
ipContourList    : ITcVnContainer;

// Watchdog
hrWD         : HRESULT;
tStop        : DINT := 5000;
tRest        : DINT;
nFraction    : UDINT;

// Output
sText               :   STRING(255);

// Color
aColorGreen         :   TcVnVector4_LREAL := [0, 255, 0, 0];

// Image Infos
stPixelFormat       :   TcVnPixelFormat;
```

代码

```
// Set Parameters
// ---------------------------------------------------------------
// - Set fMaxArea to 100000 detect all circles or set it to 10000
//   to detect only the small circles
```

```
stBlobParams.bFilterByArea          := TRUE;
stBlobParams.fMinArea               := 100;
stBlobParams.fMaxArea               := 100000;

stBlobParams.bFilterByCircularity   := TRUE;
stBlobParams.fMinCircularity        := 0.80;

stBlobParams.fMinThreshold          := 70;
stBlobParams.fThresholdStep         := 0;

// Execute DetectBlobs-Function monitored by Watchdog-Function
// ------------------------------------------------------------
// - Set the Watchdog stop time tStop from 5000us to 1000us to see
//   a Watchdog interrupt and that the interim results can be used

hrWD := F_VN_StartRelWatchdog(tStop, S_OK);
  hrFunc := F_VN_DetectBlobs(
            ipSrcImage          := ipImageIn,
                                   ipBlobContours    := ipContourList,
                                   stParams          := stBlobParams,
                                   hrPrev            := hr);
hrWD := F_VN_StopWatchdog(hrWD, tRest => tRest, nFractionProcessed => nFraction);

// Draw Result Image
// ------------------------------------------------------------
hr := F_VN_DrawContours(ipContourList, -1, ipImageRes, aColorGreen, 3, hr);

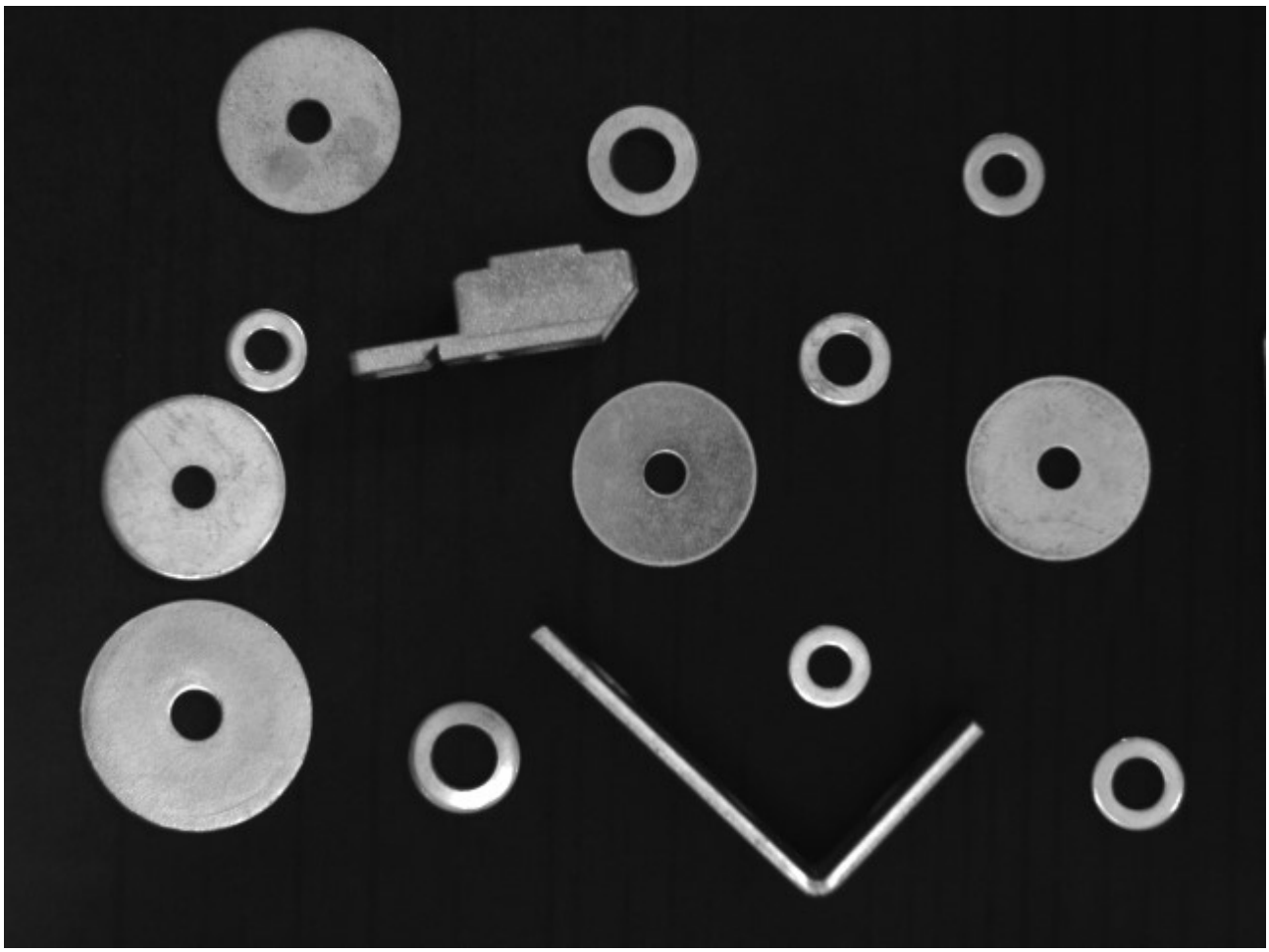
sText := CONCAT(CONCAT('Processed ', UDINT_TO_STRING(nFraction)), '%');
hr := F_VN_PutTextExp(sText, ipImageRes, 25, 50, TCVN_FT_HERSHEY_SIMPLEX, 1.3, aColorGreen, 2,
TCVN_LT_8_CONNECTED, FALSE, hr);

sText := CONCAT(CONCAT('Time ', DINT_TO_STRING(tStop - tRest)), 'us');
hr := F_VN_PutTextExp(sText, ipImageRes, 25, 100, TCVN_FT_HERSHEY_SIMPLEX, 1.3, aColorGreen, 2,
TCVN_LT_8_CONNECTED, FALSE,hr);

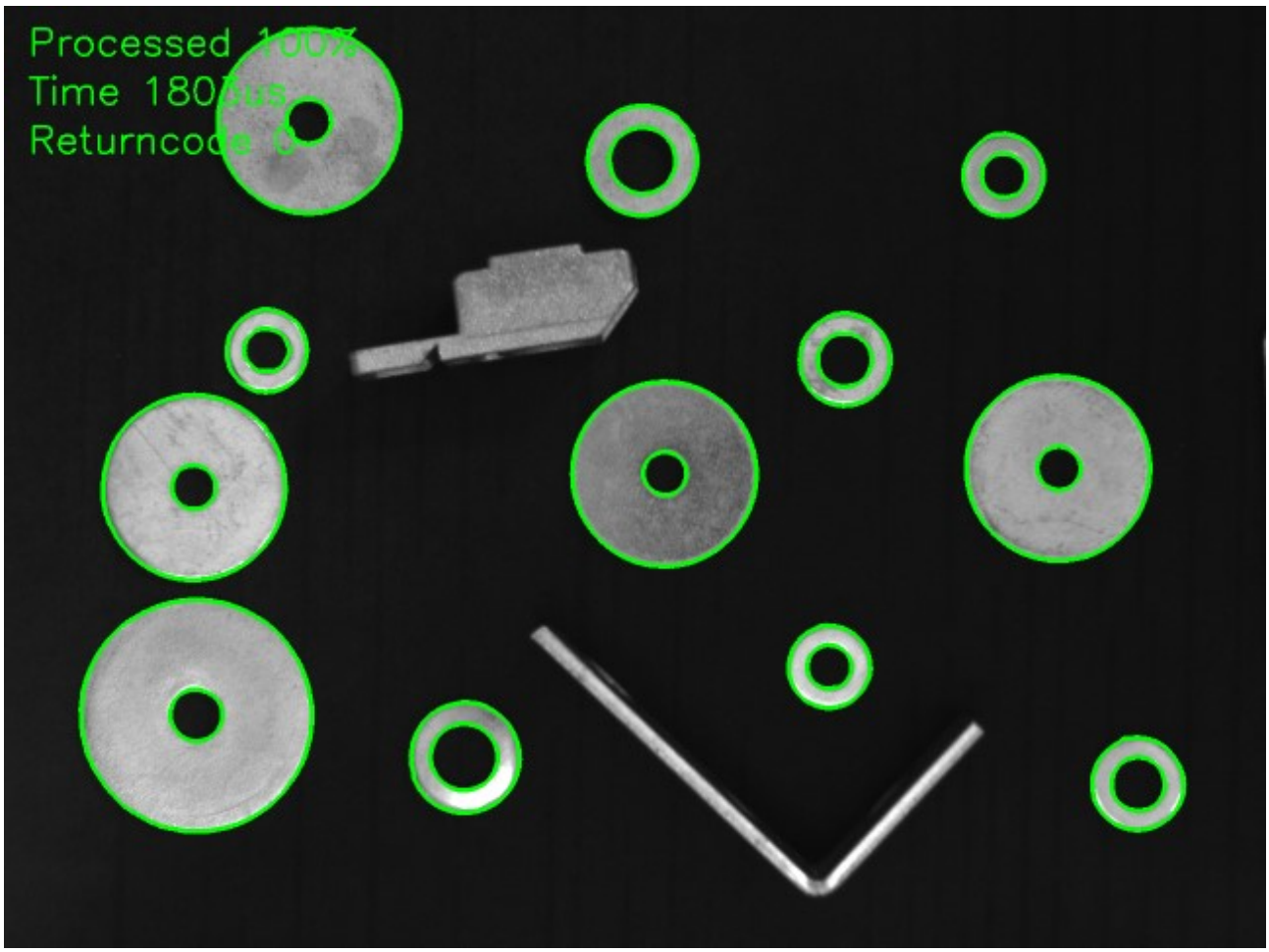
sText := CONCAT('Returncode ', DINT_TO_STRING(hrFunc));
hr := F_VN_PutTextExp(sText, ipImageRes, 25, 150, TCVN_FT_HERSHEY_SIMPLEX, 1.3, aColorGreen, 2,
TCVN_LT_8_CONNECTED, FALSE,hr);
```

结果

输入图像

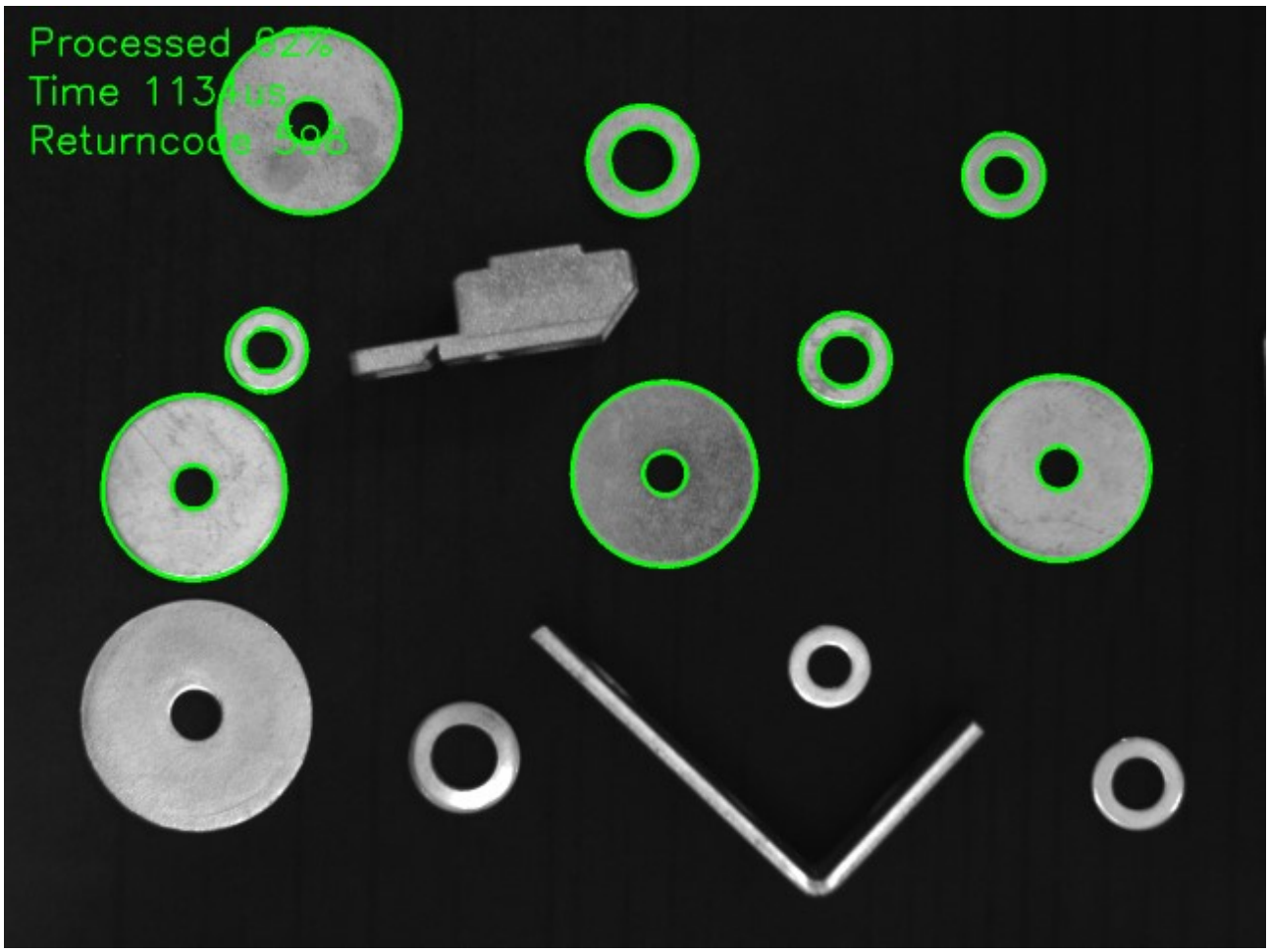


在图像中发现了所有圆圈，并对示例程序进行了参数设置。



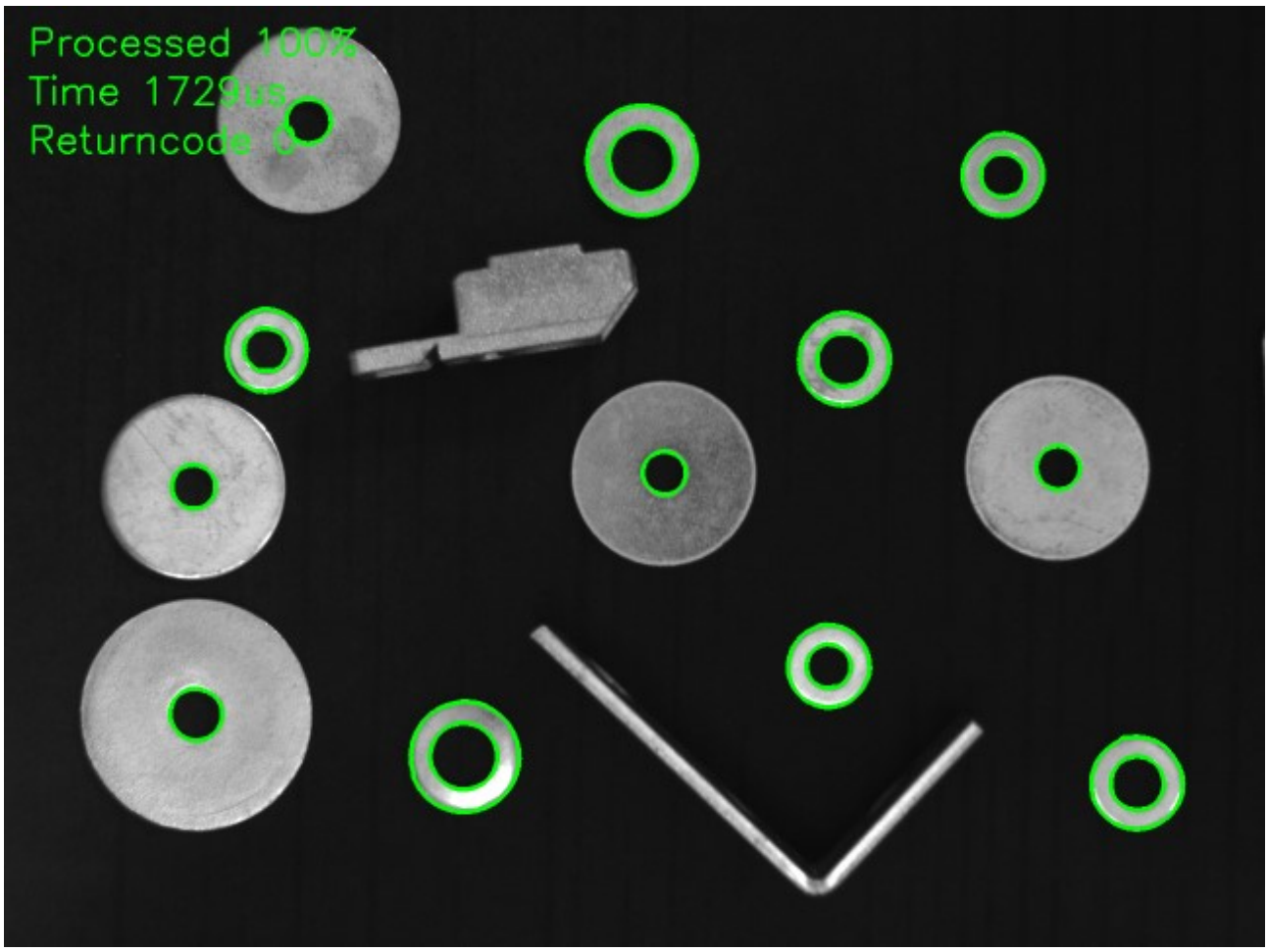
通过在 1 ms 后触发 Watchdog（看门狗），在图像中发现了一些圆圈。由于中止过程需要一些时间，因此实际所需时间大于 1 ms。

```
hrWD := F_VN_StartRelWatchdog(tStop:=1000, WATCHDOG_ACCUMULATION_TYPE_MEAN, S_OK);
```



通过缩小以像素为单位的最大封闭区域，只在图像中找到了小圆圈。

```
stBlobParams.fMinArea := 100;
stBlobParams.fMaxArea := 10000;
```



类似例程

- 查找轮廓而不是斑点检测 [▶ 2761]

8.1.6.1.2 查找轮廓而不是斑点检测

本例程展示了使用 F_VN_FindContoursExp [▶ 1181] 的斑点检测样本 [▶ 2755]的替代解决方案。

说明

函数 F_VN_DetectBlobs [▶ 1171] 将许多单独的图像处理步骤整合到一个函数调用中。不过，内部处理步骤也可以通过单独的函数调用来完成。这样可以更容易地设置参数，因为可以看到中间结果。因此，在该例程中，使用了与斑点检测样本 [▶ 2755]中相同的图像，并搜索了相同的圆圈结构。

变量

```
hr                    : HRESULT;

ipImageIn             : ITcVnImage;
ipImageInDisp         : ITcVnDisplayableImage;
ipImageWork           : ITcVnImage;
ipImageThresholdDisp  : ITcVnDisplayableImage;
ipImageRes            : ITcVnImage;
ipImageResDisp        : ITcVnDisplayableImage;

// Sample Specific Variables
ipContourList         : ITcVnContainer;
ipContourResultList   : ITcVnContainer;
ipContour             : ITcVnContainer;
ipIterator            : ITcVnForwardIterator;
aOffset               : TcVnPoint;

fThreshold            : LREAL := 70;

fArea                 : LREAL;
fAreaMin              : LREAL := 100;
fAreaMax              : LREAL := 100000;
```

```
fCircularity          : LREAL;
fCircularityMin       : LREAL := 0.8;

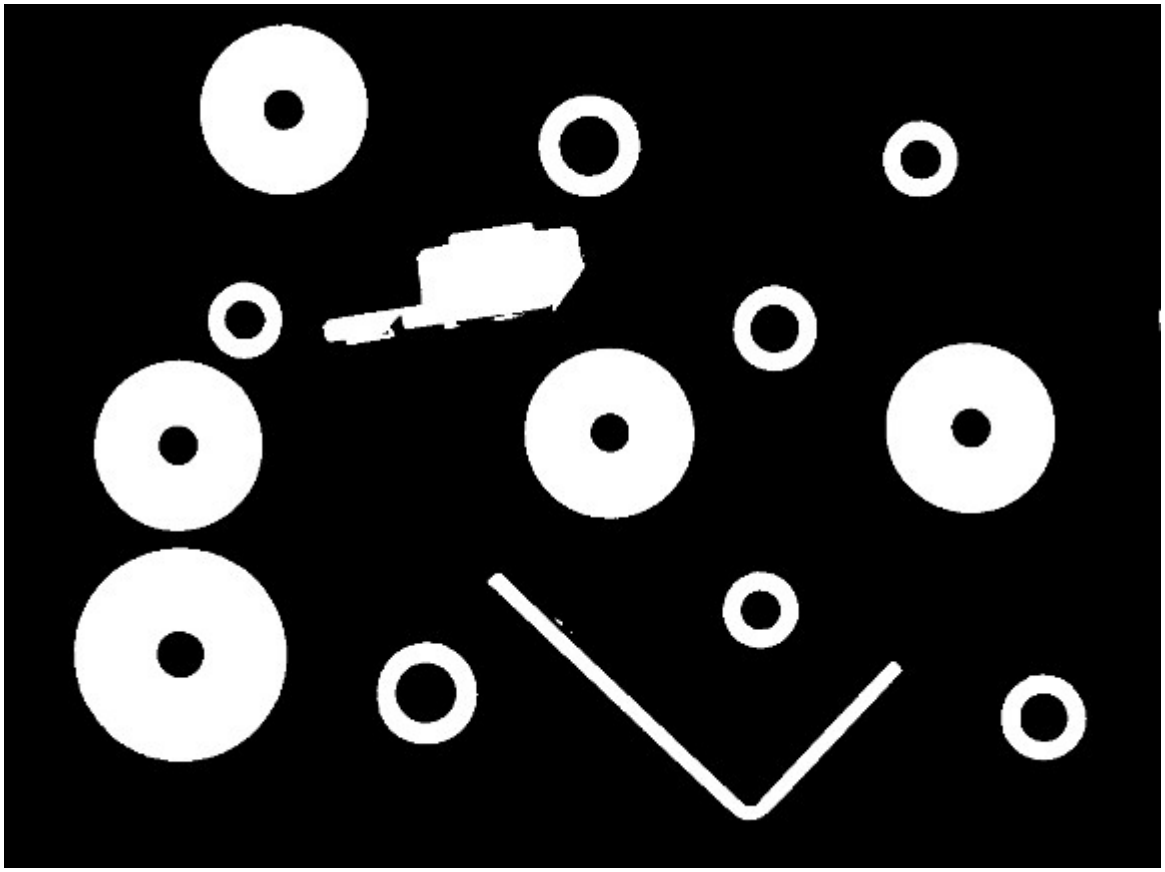
// Image Infos
stPixelFormat         : TcVnPixelFormat;

// COLORS
aColorGreen           : TcVnVector4_LREAL := [0, 255, 0];
```

代码

图像分割预处理

```
// Image Segementation
hr := F_VN_Threshold(ipImageIn, ipImageWork, fThreshold, 255, TCVN_TT_Binary, hr);
```



轮廓搜索

```
// Find Contours
hr := F_VN_FindContoursExp(
ipSrcImage              := ipImageWork,
ipContours              := ipContourList,
eRetrievalMode          := ETcVnContourRetrievalmode.TCVN_CRM_LIST,
eApproximationMethod    := ETcVnContourApproximationMethod.TCVN_CAM_SIMPLE,
aOffset                 := aOffset,
hrPrev                  := hr);
```

根据周边区域以像素和圆度对轮廓进行滤波

```
// Filter Contours
hr := F_VN_GetForwardIterator(ipContourList, ipIterator, hr);
hr := F_VN_CreateContainer(ipContourResultList, ContainerType_Vector_Vector_TcVnPoint2_DINT, 0, hr);

WHILE SUCCEEDED(hr) AND_THEN ipIterator.CheckIfEnd() <> S_OK DO

hr := F_VN_GetContainer(ipIterator, ipContour, hr);
hr := F_VN_IncrementIterator(ipIterator, hr);

// Filter by Area
hr := F_VN_ContourArea(ipContour, fArea, hr);

IF fArea > fAreaMin AND fArea < fAreaMax THEN

// Filter by Circularity
hr := F_VN_ContourCircularity(ipContour, fCircularity, hr);

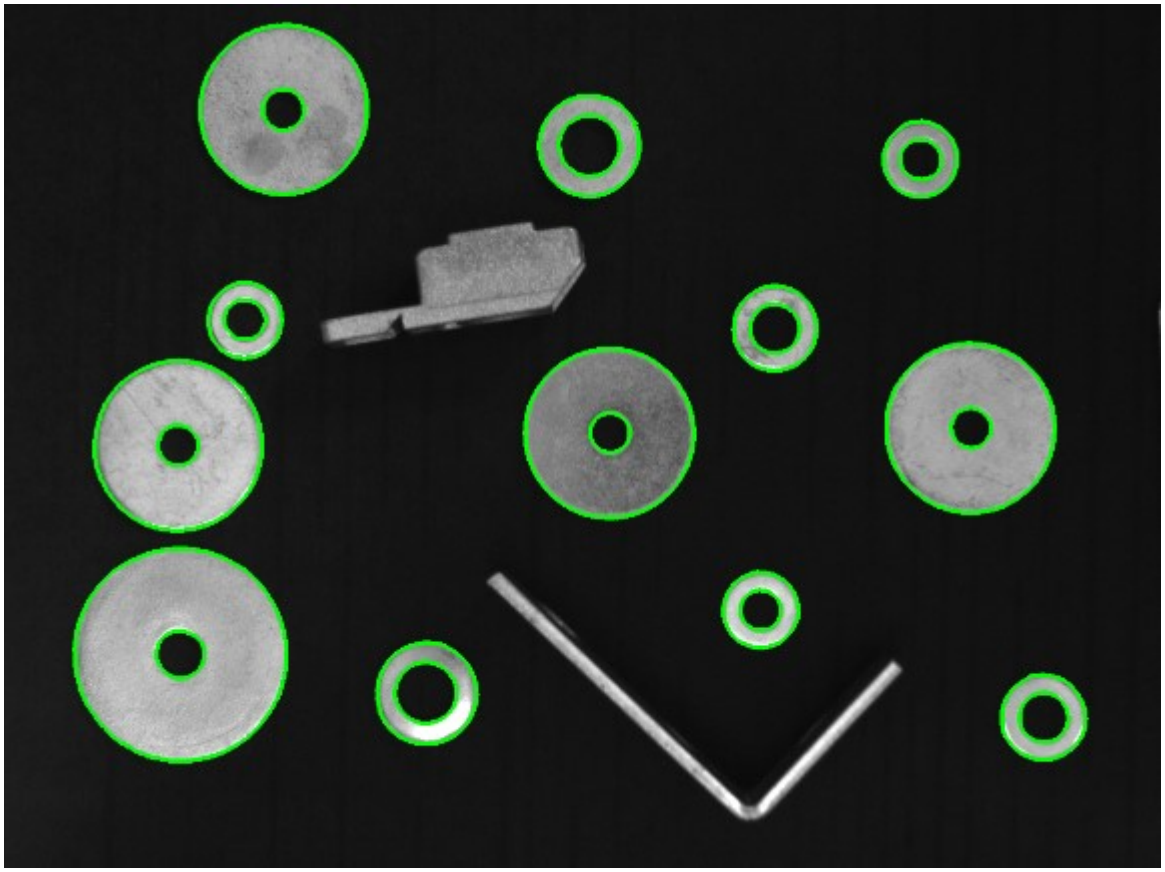
IF fCircularity > fCircularityMin THEN
// Add contour to the result contour container
```

```
hr := F_VN_AppendToContainer_ITcVnContainer(ipContour, ipContourResultList, hr);

END_IF
END_IF
END_WHILE
```

结果输出

```
// Draw contours into the result image and display it
hr := F_VN_DrawContours(ipContourResultList, -1, ipImageRes, aColorGreen, 3, hr);
hr := F_VN_TransformIntoDisplayableImage(ipImageRes, ipImageResDisp, hr);
```



类似例程

- 带看门狗监控的斑点检测 [▶ 2755]

8.1.6.1.3 查找轮廓 - 近似法

本示例涉及轮廓搜索中可能的拟合模式（类型为 ETcVnContourApproximationMethod [▶ 179] 的 **eApproximationMethod**）。

说明

F_VN_FindContours [▶ 1179]、F_VN_FindContoursExp [▶ 1181] 和 F_VN_FindContourByHierarchy [▶ 1175] 函数用于检测图像中的轮廓。轮廓由点来描述。根据所选的拟合模式（类型为 ETcVnContourApproximationMethod [▶ 179] 的 **eApproximationMethod**），将返回每个轮廓的所有点或简化后的点集：

- TCVN_CAM_NONE
  返回轮廓的所有点
- TCVN_CAM_SIMPLE
  采用无损压缩
- TCVN_CAM_TC89_L1 或 TCVN_CAM_TC89_KCOS
  这两个函数均采用 Teh-Chin 链式拟合算法中的一种变体

变量

```
hr                      :   HRESULT;
ipImageIn               :   ITcVnImage;
ipImageInDisp           :   ITcVnDisplayableImage;

ipImageRes              :   ITcVnImage;
ipImageResDisp          :   ITcVnDisplayableImage;

// Sample Specific Variables
ipContourList           :   ITcVnContainer;
ipContour               :   ITcVnContainer;
ipIterator              :   ITcVnForwardIterator;
eRetrievalMode          :   ETcVnContourRetrievalMode := TCVN_CRM_LIST;
eApproximationMethod    :   ETcVnContourApproximationMethod := TCVN_CAM_SIMPLE;
aOffset                 :   TcVnPoint;

// Colors
aColorRed               :   TcVnVector4_LREAL := [255, 0, 0];
```

代码

```
// Create Result Image
hr := F_VN_ConvertColorSpace(ipImageIn, ipImageRes, TCVN_CST_Gray_TO_RGB, hr);

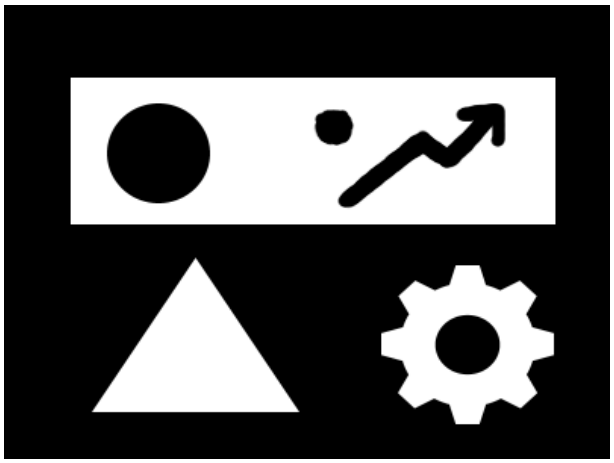
// Find Contours and their Hierarchy
// ---------------------------------
hr := F_VN_FindContoursExp(
        ipSrcImage          := ipImageIn,
        ipContours          := ipContourList,
        eRetrievalMode      := eRetrievalMode,
        eApproximationMethod    := eApproximationMethod,
        aOffset             := aOffset,
        hrPrev              := hr);

// Draw the points of the contour
hr := F_VN_GetForwardIterator(ipContourList, ipIterator, hr);

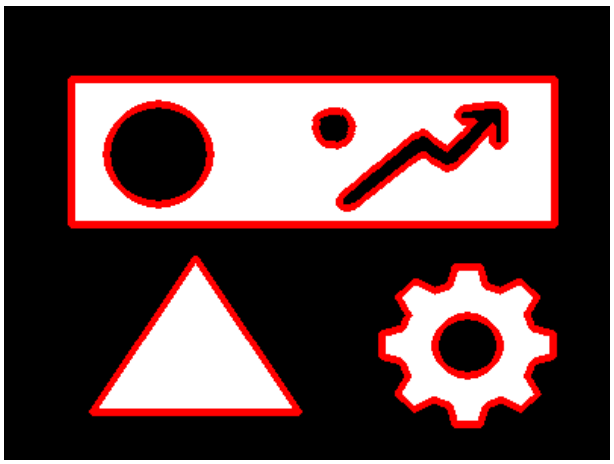
WHILE SUCCEEDED(hr) AND_THEN ipIterator.CheckIfEnd() <> S_OK DO
    hr := F_VN_GetContainer(ipIterator, ipContour, hr);
    hr := F_VN_IncrementIterator(ipIterator, hr);
    hr := F_VN_DrawPoints(ipContour, ipImageRes, ETcVnDrawShape.TCVN_DS_CIRCLE, aColorRed, hr);
END_WHILE
```

结果

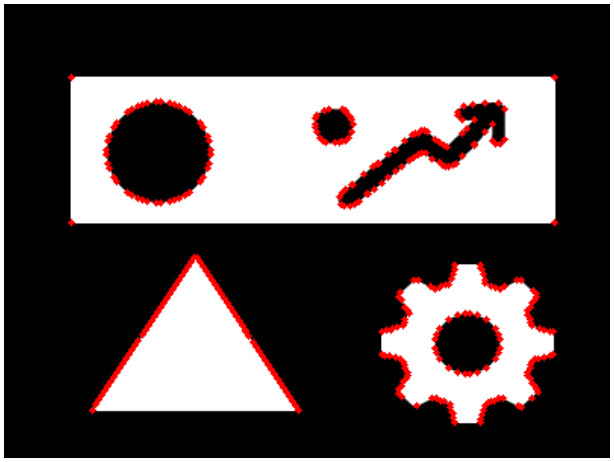
输入图像



根据 ETcVnContourApproximationMethod [▶ 179].TCVN_CAM_NONE 绘制所有点后得到的结果图像



根据 ETcVnContourApproximationMethod [▶ 179].TCVN_CAM_TC89_L1 绘制所有点后得到的结果图像



类似例程

- 查找轮廓 - 层次结构和检索模式 [▶ 2765]
- 查找轮廓而不是斑点检测 [▶ 2761]

8.1.6.1.4 查找轮廓 - 层次结构和检索模式

该示例展示了轮廓搜索（F_VN_FindContoursExp [▶ 1181] 和 F_VN_FindContourHierarchyExp [▶ 1175]）的检索模式（ETcVnContourRetrievalMode [▶ 181]），以及该模式如何影响轮廓结果和返回的层级。

说明

轮廓检索模式（类型为 ETcVnContourRetrievalMode [▶ 181] 的 **eRetrievalMode**）可用于指定在轮廓搜索过程中如何考虑轮廓层级。该示例根据所选模式对结果进行可视化展示。

有关层级结构的详细说明，请参见 F_VN_FindContourHierarchyExp [▶ 1175]。

变量

```
hr                      :   HRESULT;

ipImageIn               :   ITcVnImage;
ipImageInDisp           :   ITcVnDisplayableImage;

ipImageRes              :   ITcVnImage;
ipImageResDisp          :   ITcVnDisplayableImage;

ipImageHierarchy        :   ITcVnImage;
ipImageHierarchyDisp    :   ITcVnDisplayableImage;

// Sample Specific Variables
ipContourList           :   ITcVnContainer;
```

```
ipContour               :   ITcVnContainer;
ipHierarchyList         :   ITcVnContainer;
aHierarchy              :   TcVnVector4_DINT;
aHierarchyTree          :   TcVnVector4_DINT;
eRetrievalMode     : ETcVnContourRetrievalMode := TCVN_CRM_LIST;
eApproximationMethod    :   ETcVnContourApproximationMethod := TCVN_CAM_SIMPLE;
aOffset                 :   TcVnPoint;

nNumOfElem              :   ULINT;
nParents                :   UINT;
i                       :   ULINT;
j                       :   ULINT;
stRect                  :   TcVnRectangle_UDINT;
sText                   :   STRING(80);

// Colors
aColorWhite             :   TcVnVector4_LREAL := [255, 255, 255];
aColorBlack             :   TcVnVector4_LREAL := [0, 0, 0];
aColorList              :   ARRAY[0..4] OF TcVnVector4_LREAL := [ [255,0,0], [0,255,0], [0,0,255], [
200,200,0], [200,0,200]];
```

代码

```
// Create Result Image
hr := F_VN_ConvertColorSpace(ipImageIn, ipImageRes, TCVN_CST_Gray_TO_RGB, hr);
hr := F_VN_CreateImage(ipImageHierarchy, 130, 200, EtcVnElementType.TCVN_ET_USINT, 3, hr);
hr := F_VN_SetPixels(ipImageHierarchy, aColorWhite, hr);

// Find Contours and their Hierarchy
// ----------------------------------
hr := F_VN_FindContourHierarchyExp(

      ipSrcImage        := ipImageIn,
      ipContours        := ipContourList,
      ipHierarchy       := ipHierarchyList,
      eRetrievalMode      := eRetrievalMode,
      eApproximationMethod  := eApproximationMethod,
      aOffset           := aOffset,
      hrPrev            := hr);

hr := F_VN_GetNumberOfElements(ipContourList, nNumOfElem, hr);

IF nNumOfElem > 0 THEN
   FOR i:= 0 TO (nNumOfElem-1) BY 1 DO

      hr := F_VN_GetAt_ITcVnContainer(ipContourList, ipContour, i, hr);
      hr := F_VN_GetAt_TcVnVector4_DINT(ipHierarchyList, aHierarchy, i, hr);

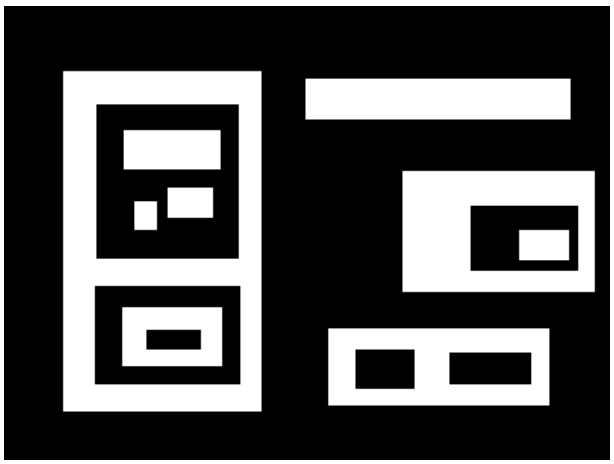
      // Count Parents and set aColor depending on parent number
      nParents := 0;
      aHierarchyTree := aHierarchy;
      WHILE SUCCEEDED(hr) AND_THEN aHierarchyTree[3] >= 0 DO
         nParents := nParents + 1;
         hr := F_VN_GetAt_TcVnVector4_DINT(ipHierarchyList, aHierarchyTree,
TO_ULINT(aHierarchyTree[3]), hr);
      END_WHILE

      // Draw contour and number into the result image
      sText := TO_STRING(i);
      hr := F_VN_UprightBoundingRectangle(ipContour, stRect, hr);
      hr := F_VN_PutText(sText, ipImageRes, stRect.nX +10, stRect.nY +15, TCVN_FT_HERSHEY_PLAIN, 1,
aColorList[(nParents) MOD 4], hr);
      hr := F_VN_DrawContours(ipContour, -1, ipImageRes, aColorList[(nParents) MOD 4], 2, hr);

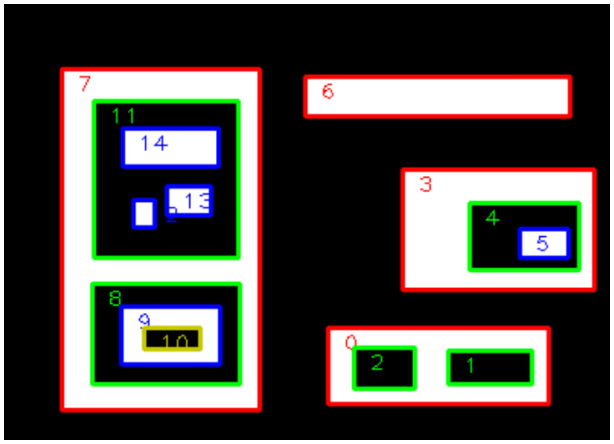
      // Write Hierarchy
      hr := F_VN_PutText(sText, ipImageHierarchy, 10, TO_UDINT(10 + i*10), TCVN_FT_HERSHEY_PLAIN,
0.5, aColorList[(nParents) MOD 4], hr);
      FOR j := 0 TO 3 BY 1 DO
         sText := TO_STRING(aHierarchy[j]);
         hr := F_VN_PutText(sText, ipImageHierarchy, TO_UDINT(10 + (j+1)*22), TO_UDINT(10 + i*10),
TCVN_FT_HERSHEY_PLAIN, 0.5, aColorBlack, hr);
      END_FOR
   END_FOR
END_IF
```

结果

输入图像



使用 TCVN_CRM_TREE 找到的轮廓。相同层级的轮廓以相同的颜色呈现



上述结果图像的层级。每行对应一个轮廓。各列的含义如下：

- 轮廓索引（“彩色列”）
- 相同层级上的下一个轮廓的索引
- 相同层级上的上一个轮廓的索引
- 第一个子轮廓的索引
- 父轮廓的索引

数字 -1 表示不存在相应的轮廓。



类似例程

- 查找轮廓 - 近似法 [▶ 2763]
- 查找轮廓而不是斑点检测 [▶ 2761]

8.1.6.1.5 模板匹配和评估

在本示例中，您将执行模板匹配，以找到印刷电路板上的一个元件。为此，请使用：

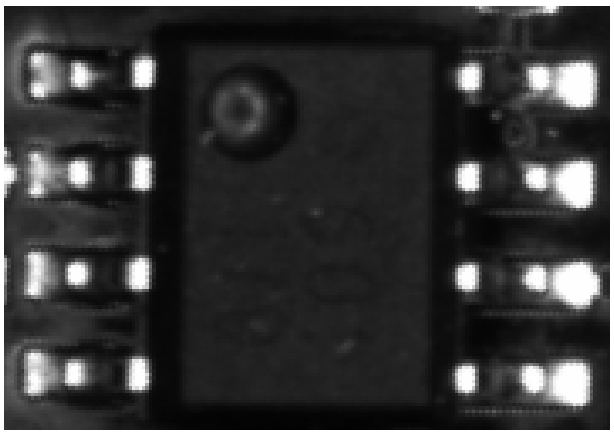
- F_VN_MatchTemplateAndEvaluate [▶ 1203]
- F_VN_DrawRectangle [▶ 1039]

应用

请记得将输入图像和模板图像添加到相应的文件源中。通过 `fMatchThreshold` 阈值，您可以设置一个匹配度下限，只有达到该下限的匹配结果才会被接受。

程序

在本示例中，需要注意的是，除了正常的输入图像外，您还需要一个模板图像，以便在输入图像中搜索与模板相匹配的位置。



如果两个图像均可用，则使用 F_VN_MatchTemplateAndEvaluate [▶ 1203] 函数自动执行模板匹配。

```
hr := F_VN_MatchTemplateAndEvaluate(ipImageIn, ipImageTemplate, ipMatches, fMatchThreshold, hr);
```

在找到的匹配项中，那些相关性程度高于指定的 `fMatchThreshold` 阈值的匹配项现在以点的形式存放在 `ipMatches` 容器中。每个点指定了 `ipImageTemplate` 在 `ipImageIn` 中每个出现位置的左上角。

原则上，您现在已经完成了模板匹配。举例来说，接下来将访问各个匹配项，以便将它们绘制到图像中。为此，请使用容器 [▶ 146]章节中介绍的访问容器元素的结构。

```
hr := F_VN_GetImageHeight(ipImageTemplate, nHeight, hr);
hr := F_VN_GetImageWidth(ipImageTemplate, nWidth, hr);

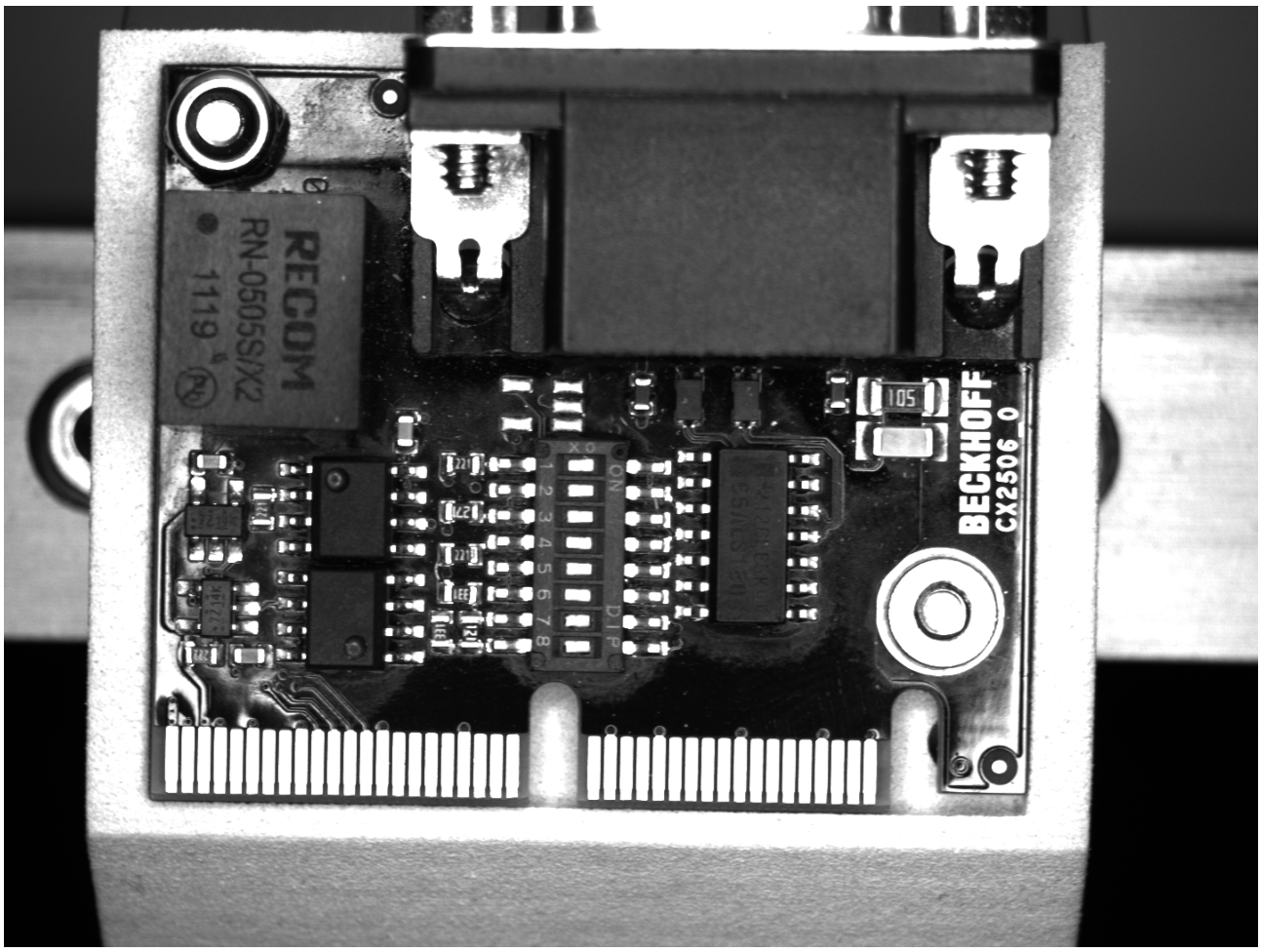
hr := F_VN_GetForwardIterator(ipMatches, ipIterator, hr);
IF SUCCEEDED(hr) AND ipIterator <> 0 THEN
    hr := ipIterator.TcQueryInterface(IID_ITcVnAccess_TcVnPoint2_DINT, ADR(ipAccess));
    IF SUCCEEDED(hr) AND ipAccess <> 0 THEN
        WHILE SUCCEEDED(hr) AND ipIterator.CheckIfEnd() <> S_OK DO
            hr := ipAccess.Get(aPosition);
            hr := F_VN_DrawRectangle(
                DINT_TO_UDINT(aPosition[0]),
                DINT_TO_UDINT(aPosition[1]),
                DINT_TO_UDINT(aPosition[0])+nWidth,
                DINT_TO_UDINT(aPosition[1])+nHeight,
                ipImageRes,
                aGreen,
                5,
                hr);
            hr := F_VN_IncrementIterator(ipIterator, hr);
        END_WHILE
    END_IF
END_IF
```

注意

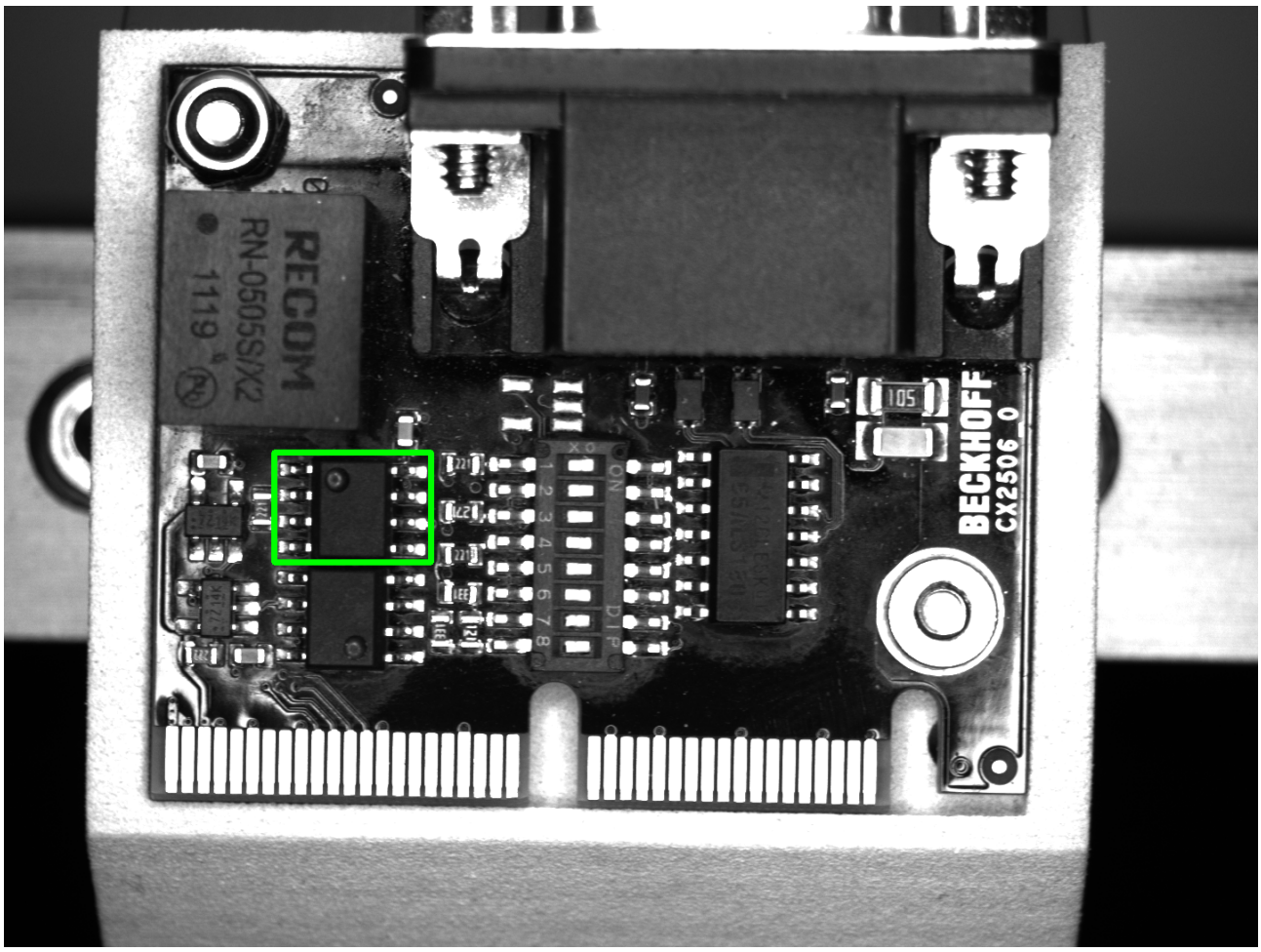
执行时间长

根据输入图像和模板图像的大小，与其他函数相比，模板匹配的执行可能需要很长的时间。确保相应设置循环时间，必要时使用 Watchdog（看门狗）！

原始图像



结果图像



8.1.6.1.6 带旋转的Walsh模板匹配

在本示例中，在执行模板匹配时需要考虑旋转位置，以便定位参考图像在搜索图像中的所有实例，并确定它们的方向。

为此可使用以下函数之一：

- F_VN_FindTemplateWalshExp [▶ 1185] 用于以固定的角度步长在整个 360° 范围内进行搜索。
- F_VN_FindTemplateWalshExp_AngleRange [▶ 1191] 用于限制在指定的角度范围内进行定向搜索，且角度增量可自由调整。

说明

在Walsh变换的基础上，Walsh模板匹配在输入图像中搜索参考图像的实例。由于通常只需少量此类投影即可进行可靠的比较，因此该方法性能很高，从而也能支持在不同旋转角度下进行搜索，以便确定旋转对象的位置和方向。

本示例包含三个不同的模板图像，以供测试使用。通过 `sFolderPath` 和 `sTemplateFileName` 指定要使用的模板。要加载新模板，请将 `bTemplateLoaded` 设为 `FALSE`；然后图像将自动重新加载。更换模板后，应检查匹配参数，并在必要时进行调整。

`bUseRotationSteps` 参数可用于在两种函数之间进行切换：

- 如果设置为 `TRUE`，则执行 `F_VN_FindTemplateWalshExp`。测试整个旋转范围（0° 至 360°）。角度分辨率由参数 `nRotations` 得出，其中仅限数值 1、2、4、8、16 和 32 有效。
- 如果设置为 `FALSE`，则使用 `F_VN_FindTemplateWalshExp_AngleRange`。该函数允许通过 `fStartAngle` 和 `fStopAngle`（例如 ±90°）限制搜索范围，并通过 `nAngles` 设置任意所需的测试步数。

参数 nRotations 和 nAngles 定义了要检查的角度步数，从而决定了搜索的角度分辨率。由于算法会在这些离散的测试步骤之间进行插值，因此也可以在明确的测试角度之间确定旋转值。所选的测试旋转次数对函数的总运行时有很大影响。因此，为了优化性能，建议使用 F_VN_FindTemplateWalshExp_AngleRange 将搜索范围限制为实际相关的角度范围。这样可以减少测试角度的数量，并最大限度地缩短执行时间。

此外，运行时与可达到的匹配质量在很大程度上取决于 nProjections 和 fScaleFactor 之间的相互作用。更多的投影次数可以让更多的图像信息流入处理过程，从而提高匹配的可靠性，但会增加处理时间。同样，缩放系数越大（越接近 1.0），精度越高，但处理时间也越长。由于这两个参数都直接影响相似度值，因此每次参数变更后都必须相应地重新调整阈值 fMatchThreshold。

本章末尾提供了有关此处所述参数的参数设置与优化的更多信息。

此外，还设置了 ROI（stMatchingRoi），将搜索区域限制在相关图像部分。这样可以减少需要分析的像素数量，从而缩短处理时间。不过，只有完全位于 ROI 内的对象才能被找到。

模板匹配的执行

首先，在初始图像中定义 ROI。根据 bUseRotationSteps 参数的不同，随后将执行 360° 搜索（F_VN_FindTemplateWalshExp），或者在指定的角度范围内执行定向搜索（F_VN_FindTemplateWalshExp_AngleRange）。

由于 nRotations 参数被限制为固定值 1、2、4、8、16 和 32，因此一条前置的 CASE 指令可以确保将输入值向下舍入到下一个较低的有效值。如果不进行此项修正，函数将返回 HRESULT 0x70B 错误代码（INVALIDPARM）。

对于这两个搜索函数，标志 TCVN_FTWO_FUSE_MATCHES 和 TCVN_FTWO_ROTATED_RECTANGLE 均会被传输至 eOptions 参数。建议始终使用 TCVN_FTWO_FUSE_MATCHES，因为该选项会将因旋转角度和位置略有不同而产生的冗余匹配项合并为一个匹配项。TCVN_FTWO_ROTATED_RECTANGLE 标志大大简化了后续的评估过程，因为其结果直接以结构化数据类型 TcVnRotatedRectangle 的形式提供。

变量

```
// ROI for the template matching
stMatchingRoi       :   TcVnRectangle_UDINT := (nX := 150, nY := 150, nWidth := 740, nHeight :=
650);

// Template matching parameters
bUseRotationSteps   :   BOOL := TRUE;
bShowMatchValue     :   BOOL := TRUE;
bShowRotationValue  :   BOOL := TRUE;
fMatchThreshold     :   REAL := 0.94;
nProjections        :   UDINT := 15;
nRotations          :   UDINT := 32;
fScaleFactor        :   REAL := 0.225;
eInterpolationType  :   ETcVnInterpolationType := TCVN_IT_BILINEAR;

// Template matching with angle range specific parameters.
nAngles             :   UDINT := 16;
fStartAngle         :   LREAL := -90;
fStopAngle          :   LREAL := 90;

// Template matching results
hrMatch             :   HRESULT;
ipMatches           :   ITcVnContainer;
ipMatchValues       :   ITcVnContainer;
```

代码

```
hr := F_VN_SetRoi_TcVnRectangle_UDINT(stMatchingRoi, ipImageIn, hr);

IF bUseRotationSteps THEN
    // Use when object rotation is not specified to cover up to 360° range

    // Valid values for the number of rotations: 1 (0°), 2 (0° and 180°), 4 (0°, 90°, 180°, and
270°), 8 (0°, 45°, ..., 315°), 16 (0°, 22.5°, ..., 337.5°) and 32 (0°, 11.25°, ..., 348.75°)
    // Set value to the next lower valid rotation step
    CASE nRotations OF
        0..1: nRotations := 1;
        2..3: nRotations := 2;
        4..7: nRotations := 4;
        8..15: nRotations := 8;
        16..31: nRotations := 16;
    ELSE
        nRotations := 32;
    END_CASE

    hrMatch := F_VN_FindTemplateWalshExp(
            ipSrcImage          := ipImageIn,
```

```
            ipTemplateImage     := ipImageTemplate,
            ipMatches           := ipMatches,
            fMatchThreshold     := fMatchThreshold,
            nProjections        := nProjections,
            eOptions            := TCVN_FTWO_FUSE_MATCHES OR TCVN_FTWO_ROTATED_RECTANGLE,
            nRotations          := nRotations,
            fScaleFactor        := fScaleFactor,
            eInterpolationType  := eInterpolationType,
            ipMatchValues       := ipMatchValues,
            hrPrev              := hr);
ELSE
    // Use when rotation range can be limited (e.g. ±45°)
    hrMatch := F_VN_FindTemplateWalshExp_AngleRange(
            ipSrcImage          := ipImageIn,
            ipTemplateImage     := ipImageTemplate,
            ipMatches           := ipMatches,
            fMatchThreshold     := fMatchThreshold,
            nProjections        := nProjections,
            fStartAngle         := fStartAngle,
            fStopAngle          := fStopAngle,
            nAngles             := nAngles,
            eOptions            := TCVN_FTWO_FUSE_MATCHES OR TCVN_FTWO_ROTATED_RECTANGLE,
            fScaleFactor        := fScaleFactor,
            eInterpolationType  := eInterpolationType,
            ipMatchValues       := ipMatchValues,
            hrPrev              := hr);
END_IF
```

结果的评估与绘制

首先，系统会检查函数是否执行成功，以及是否找到了匹配项。遍历找到的匹配项，以读取坐标（带 ROI 偏移）和匹配值，并将其绘制到结果图像中。为此，在项目中提供了一个辅助函数 `F_DrawMatchResult` ，用于绘制一个旋转矩形和一个表示方向的箭头。同时还会显示匹配值和检测到的旋转角度。如果只想显示旋转后的矩形，则可以使用变量 `bShowMatchValue` 和 `bShowRotationValue` 隐藏这些文本。

变量

```
stMatchRectangle    :   TcVnRotatedRectangle;
fMatchValue         :   REAL;
nNumberOfMatches    :   ULINT;
nMatchIdx           :   ULINT;

// Color and Drawing
sStatusText         :   STRING;
aColorGreen         :   TcVnVector4_LREAL := [0, 255, 0];
aColorWhite         :   TcVnVector4_LREAL := [255, 255, 255];
aColorBlack         :   TcVnVector4_LREAL := [0, 0, 0];
aColorOrange        :   TcVnVector4_LREAL := [255, 140, 0];

hr                  :   HRESULT;
```

代码

```
IF hrMatch = S_OK THEN
    hr := F_VN_GetNumberOfElements(ipMatches, nNumberOfMatches, hr);

    IF nNumberOfMatches > 0 THEN
        // Write the number of matches into result image
        sStatusText := CONCAT('Matches: ', ULINT_TO_STRING(nNumberOfMatches));

        FOR nMatchIdx := 0 TO nNumberOfMatches - 1 DO
            // Get match region
            hr := F_VN_GetAt_TcVnRotatedRectangle(ipMatches, stMatchRectangle, nMatchIdx, hr);
            // Add ROI offeset
            stMatchRectangle.aCenter[0] := stMatchRectangle.aCenter[0] +
UDINT_TO_REAL(stMatchingRoi.nX);
            stMatchRectangle.aCenter[1] := stMatchRectangle.aCenter[1] +
UDINT_TO_REAL(stMatchingRoi.nY);

            // Get match value
            hr := F_VN_GetAt_REAL(ipMatchValues, fMatchValue, nMatchIdx, hr);

            // Draw matches on result image
            hr := F_DrawMatchResult(stMatchRectangle, ipImageResult, bShowRotationValue,
bShowMatchValue, fMatchValue, 3, 1.2, aColorGreen, aColorOrange, hr);
        END_FOR
    ELSE
        sStatusText := 'No matches';
    END_IF
ELSIF hrMatch = Tc2_System.E_HRESULTAdsErr.NOTFOUND THEN
    sStatusText := 'No matches';
ELSE
    // Last 3 hex digits of HRESULT
    sStatusText := CONCAT('HRESULT: 0x', DWORD_TO_HEXSTR((TO_DWORD(hrMatch) AND 16#FFF), 3, FALSE));
END_IF

// Draw status text
hr := F_VN_PutLabelExp(sStatusText, ipImageResult, 0, 32, 3, 3, TCVN_FT_HERSHEY_PLAIN, aColorBlack,
```

```
aColorWhite, TCVN_LT_8_CONNECTED, S_OK);
hr := F_VN_ResetRoi(ipImageIn, S_OK);
```

结果

结果图像 `ipImageResultDisp` 对找到的匹配项进行了可视化，这些匹配项由函数通过 `ipMatches` 容器输出。使用的 ROI 以蓝色显示。

使用示例程序（`sTemplateFileName := 'Template01.png'`）中存储的默认值将“CE”标记作为模板进行搜索：



使用相同的参数可以在图像中搜索第二个模板（Template02.png）。



通过 `fExecutionTime_ms` 评估函数调用后的执行时间，可以分析性能，这一过程非常有趣。由于 Template02.png 的整体图像面积几乎增大了一倍，因此在使用相同参数的情况下，其匹配速度更快。

基本而言，模板尺寸和长宽比的选择会影响处理时间：

- 由于算法原因，较大尺寸的模板比非常小的模板处理得更快。
- 为了获得最短的执行时间，正方形模板优于矩形格式的模板。不过，只有当您要查找的对象区域能够被轻松地隔离在一个正方形区域内，且不会有过多的干扰背景像素流入模板时，这种做法才有意义。

作为第二个用例，例如可以使用 Template03.png，通过端子模块上的触点来检查元件是否存在、位于哪个位置以及处于何种方向。例如，如果只接受 ±60° 的方向，则可以使用 `F_VN_FindTemplateWalshExp_AngleRange` 函数有针对性地限制搜索范围：

```
bUseRotationSteps    := FALSE;
fStartAngle          := -60;
fStopAngle           := 60;.
```

ROI 必须进行相应的调整，并预留足够大的尺寸，以确保在允许的 ±60° 范围内，无论元件如何旋转，都能可靠地找到所搜索的触点。同时，它应尽可能小，以免不必要地延长执行时间。此外，由于搜索范围仅限于 120°，因此可以大大减少测试角度的数量，从而最大限度地缩短处理时间。针对此场景，以下参数已被证明是有效的：

```
// Adjusted parameters for Template03.png
stMatchingRoi.nX         := 200;
stMatchingRoi.nY         := 10;
stMatchingRoi.nWidth     := 750;
stMatchingRoi.nHeight    := 500;

fMatchThreshold          := 0.94;
nProjections             := 18;
nAngles                := 16;
fScaleFactor             := 0.2;
```



关于参数设置和优化的说明

要将本示例适配到您自己的应用程序、图像或新模板，必须检查匹配参数，并在必要时进行调整。对于参数设置与错误分析，建议遵循以下步骤：

- **模板图像的选择：**由于Walsh匹配是基于像素的，因此所选模板图像的适用性（结构质量）对识别成功与否具有决定性作用。模板应尽可能以醒目、高对比度且独特的结构（例如，特定的轮廓或图案）框选出您要查找的对象。避免同质、无特征的表面或高度重复的图案。此外，模板的灰度值变化必须尽可能与所搜索对象的图像区域精确匹配。较大的偏差会导致匹配值降低，这意味着使用所选模板可能无法实现可靠的匹配。

- **阈值（fMatchThreshold）：**
  该参数可对结果进行筛选。如果阈值设置得过高，则可能无法找到正确的匹配项。如果设置得过低，则会导致大量错误的匹配项。函数输出的所找到对象的匹配值（fMatchValue）可作为调整阈值（包含公差）的指导，以便可靠地将有效匹配与错误匹配区分开。
- **匹配稳定性：**
  如果仅通过阈值无法有效排除频繁出现的错误匹配，通常是因为匹配过程中包含的图像信息过少。提高投影数量（nProjections）和/或增大缩放系数（fScaleFactor）可以提高准确性和可靠性。由于这两个参数都直接影响得出的相似度值，因此每次参数变更后都必须相应地重新调整阈值。
- **检查缩放后的输入图像：**
  本示例会生成缩放后的调试图像，以便检查设置。这些图像以匹配函数进行内部处理时所使用的精确分辨率（考虑到参数 fScaleFactor 和 eInterpolationType）对输入图像和模板图像进行可视化。这种视觉检查可以确保即使在缩放之后，独特的对象结构仍然足够清晰可见，以实现可靠的识别。
- **旋转角度精度：**
  如果检测到的角度不够精确，则可以增加要检查的角度步数（nRotations 或 nAngles）。不过，角度精度不足也可能源于这样的情况：传输到匹配过程中的图像信息总体上不足以可靠地确定方向。
- **照明影响：**
  如果这些参数调整没有成功，不规律的照明情况也可能是原因所在。通过额外传递标志 TCVN_FTWO_SKIP_DC（到 eOptions 参数），可以忽略第一个Walsh投影，这意味着亮度和对比度的全局差异影响会减小。不过，这通常需要更多的投影数量。或者，应改善实际的找照明情况，或通过适当的预处理提前校正照明不均的问题。

8.1.6.2 任何类型的形状的平均强度

在本例程中，图像的平均强度在人工定义的图像区域内计算。为此，使用了一个圆形图像掩膜 [▶ 150]。

主要使用以下函数：

- F_VN_ImageAverageExp [▶ 1224]
- F_VN_DrawCircle [▶ 1003]

说明

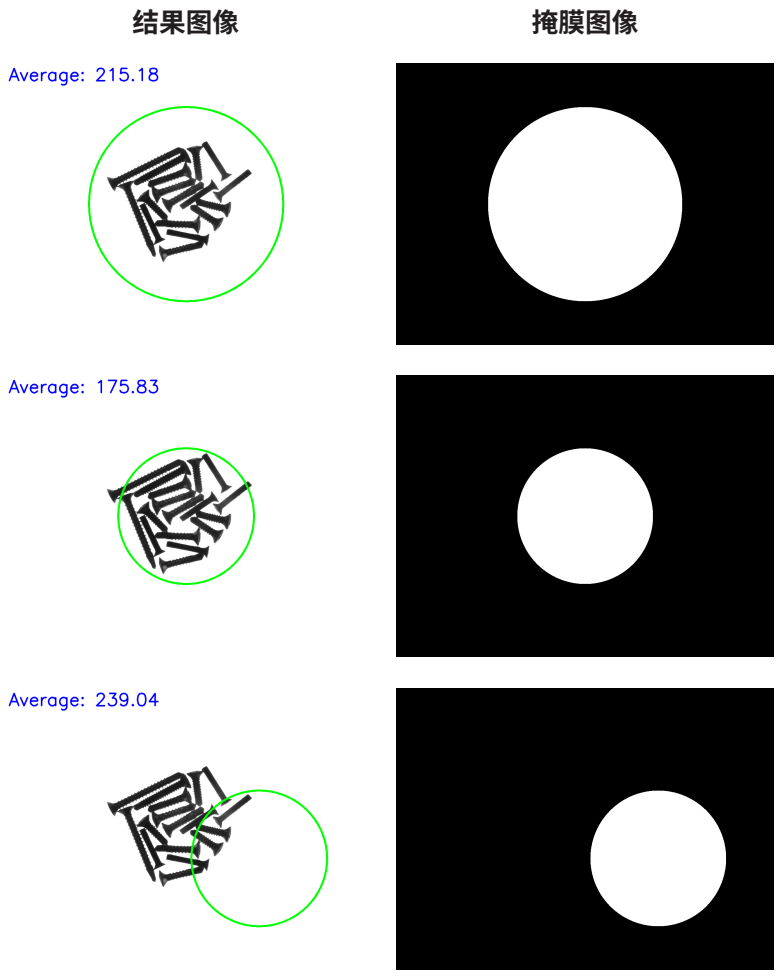
统计学图像特征 [▶ 2778] 例程显示如何计算统计图像特征（如平均值、最大值和最小值）。然而，这些特征往往不是针对整个图像计算，而只是针对特定图像区域计算。如果这些区域是矩形，设置 ROI [▶ 150] 会有所帮助。您必须使用图像掩膜 [▶ 150]才能将计算限制在任何形状的区域内。这些本质上是图像，其强度值描述了原始图像中哪些像素需要考虑。因此，掩膜可以假定为任何形状；本示例显示的是一个圆形掩膜。

对于本示例，将观察到一个带有螺钉集合的拟二元图像。一个可以想象的用例是，例如，根据平均强度值大致确定螺钉的数量。



应用

通过改变参数 nCircleShrink 和 aCenterOffset，可以改变圆形掩膜的大小和位置。根据该掩膜，平均强度值会发生变化：如果掩膜变小，显示的白色背景减少，平均值会降低。如果掩膜移动并显示更多的白色背景，则平均值就会增加。



如有必要，还可以将其他图像加载到本示例中。请注意，随后可能需要调整掩膜参数。

代码

示例图像通过 FileSource 状态机加载为 ipImageWork，并通过以下程序进行处理。

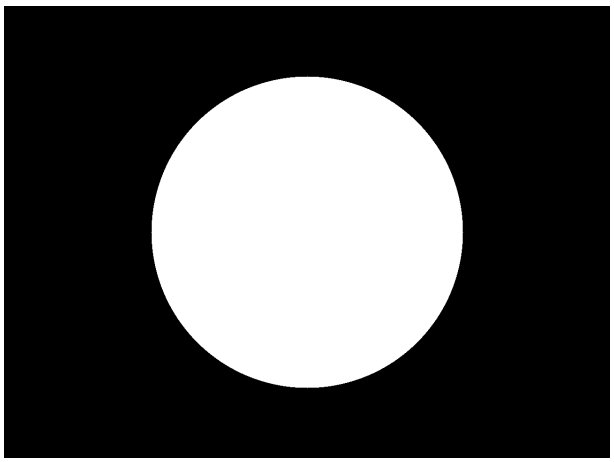
首先定义掩膜的属性。由于掩膜是圆形，它由 aCenter 中心点和 nRadius 半径描述。通过用户可调 aCenterOffset 偏移标定的图像中心点被用作圆圈的中心点。半径计算为从中心点到图像最近边界的距离减去 nCircleShrink 值。

```
hr := F_VN_GetImageWidth(ipImageWork, nWidth, hr);
hr := F_VN_GetImageHeight(ipImageWork, nHeight, hr);
aCenter[0] := (nWidth / 2) + aCenterOffset[0];
aCenter[1] := (nHeight / 2) + aCenterOffset[1];
nRadius := (MIN(nWidth, nHeight) / 2) - nCircleShrink;
```

创建与 ipImageWork 输入图像大小相同的新图像作为掩膜，并在黑色背景上绘制白色圆圈。圆圈包含计算出的属性。选择 nThickness 值作为圆边界的 -1 宽度，以便填满圆。

```
hr := F_VN_CreateImageAndSetPixels(ipImageMask, nWidth, nHeight, TCVN_ET_USINT, 1, aColorBlack, hr);
hr := F_VN_DrawCircle(
    nCenterX    :=  TO_UDINT(aCenter[0]),
    nCenterY    :=  TO_UDINT(aCenter[1]),
    nRadius     :=  nRadius,
    ipDestImage :=  ipImageMask,
    aColor      :=  aColorWhite,
    nThickness  :=  -1,
    hrPrev      :=  hr);
```

结果是现在可以得到一个具有所需圆形的 ipImageMask 掩膜图像：



现在，ipImageWork 输入图像在所定义圆形区域内的平均强度值将借助该 ipImageMask 掩膜计算。F_VN_ImageAverageExp [▶ 1224] 函数返回类型为 TcVnVector4_LREAL 的 aAverage 数组，以便覆盖最大图像通道数。由于本示例中的输入图像只有一个通道，因此第一个数组元素是所需的平均值，并单独保存。

```
hr := F_VN_ImageAverageExp(
    ipSrcImage  :=  ipImageWork,
    aAverage    :=  aAverage,
    ipMask      :=  ipImageMask,
    hrPrev      :=  hr);
fAverageInMask := aAverage[0];
```

最后，在结果图像中绘制掩膜圆圈和计算出的平均值。

```
hr := F_VN_ConvertColorSpace(ipImageWork, ipImageRes, TCVN_CST_GRAY_TO_RGB, hr);
hr := F_VN_DrawCircle(TO_UDINT(aCenter[0]), TO_UDINT(aCenter[1]), nRadius, ipImageRes, aColorGreen,
5, hr);
sText := CONCAT('Average: ', LREAL_TO_FMTSTR(fAverageInMask, 2, TRUE));
hr := F_VN_PutTextExp(sText, ipImageRes, 40, 60, TCVN_FT_HERSHEY_SIMPLEX, 2, aColorBlue, 3,
TCVN_LT_ANTIALIASED, FALSE, hr);
```

LREAL_TO_FMTSTR 函数需要 Tc2_Utilities PLC 库。

8.1.6.3 统计学图像特征

在这个样本中，将计算图像的统计特征（最小、最大、中位数、平均值、标准差）。以下功能用于此目的：

- F_VN_MinPixelValue [▶ 1233]
- F_VN_MaxPixelValue [▶ 1231]
- F_VN_ImageMedian [▶ 1229]
- F_VN_ImageAverageStdDev [▶ 1225]

应用

这个项目包含一个文件源，任何 RGB 图像都可以载入其中。这些在程序中被转换为灰度图像，并对各种统计数据进行分析。结果值被画在图像中，且可以在ADS 图像查看 [▶ 128]的ipImageRes下查看。

程序

```
// Prepare images
hr := F_VN_ConvertColorSpace(ipImageIn, ipImageIn, TCVN_CST_RGB_TO_GRAY, hr);
hr := F_VN_CopyImage(ipImageIn, ipImageRes, hr);

// Calculate image statistics
hr := F_VN_MinPixelValue(
    ipImage     :=  ipImageIn,
    aMinValue   :=  aMin,
    aPosition   :=  aPos,
    hrPrev      :=  hr
);
hr := F_VN_MaxPixelValue(
    ipImage     :=  ipImageIn,
    aMaxValue   :=  aMax,
    aPosition   :=  aPos,
    hrPrev      :=  hr
);
hr := F_VN_ImageMedian(
    ipSrcImage  :=  ipImageIn,
    aMedian     :=  aMedian,
```

```
    hrPrev      :=  hr
);
hr := F_VN_ImageAverageStdDev(
    ipSrcImage  :=  ipImageIn,
    aAverage    :=  aAverage,
    aStdDev     :=  aStdDev,
    hrPrev      :=  hr
);

// Draw results
sText := CONCAT('Min: ', LREAL_TO_FMTSTR(aMin[0], 2, TRUE ));
hr := F_VN_PutLabel(sText, ipImageRes, nTextX, (nTextYBase + (1 * nTextYIncrement)), fFontScale,
hr);
sText := CONCAT('Max: ', LREAL_TO_FMTSTR(aMax[0], 2, TRUE ));
hr := F_VN_PutLabel(sText, ipImageRes, nTextX, (nTextYBase + (2 * nTextYIncrement)), fFontScale,
hr);
sText := CONCAT('Median: ', LREAL_TO_FMTSTR(aMedian[0], 2, TRUE ));
hr := F_VN_PutLabel(sText, ipImageRes, nTextX, (nTextYBase + (3 * nTextYIncrement)), fFontScale,
hr);
sText := CONCAT('Average: ', LREAL_TO_FMTSTR(aAverage[0], 2, TRUE ));
hr := F_VN_PutLabel(sText, ipImageRes, nTextX, (nTextYBase + (4 * nTextYIncrement)), fFontScale,
hr);
sText := CONCAT('StdDev: ', LREAL_TO_FMTSTR(aStdDev[0], 2, TRUE ));
hr := F_VN_PutLabel(sText, ipImageRes, nTextX, (nTextYBase + (5 * nTextYIncrement)), fFontScale,
hr);

hr := F_VN_TransformIntoDisplayableImage(ipImageIn, ipImageInDisp, hr);
hr := F_VN_TransformIntoDisplayableImage(ipImageRes, ipImageResDisp, hr);
```

该函数LREAL_TO_FMTSTR需要 PLC 库Tc2_Utilities。

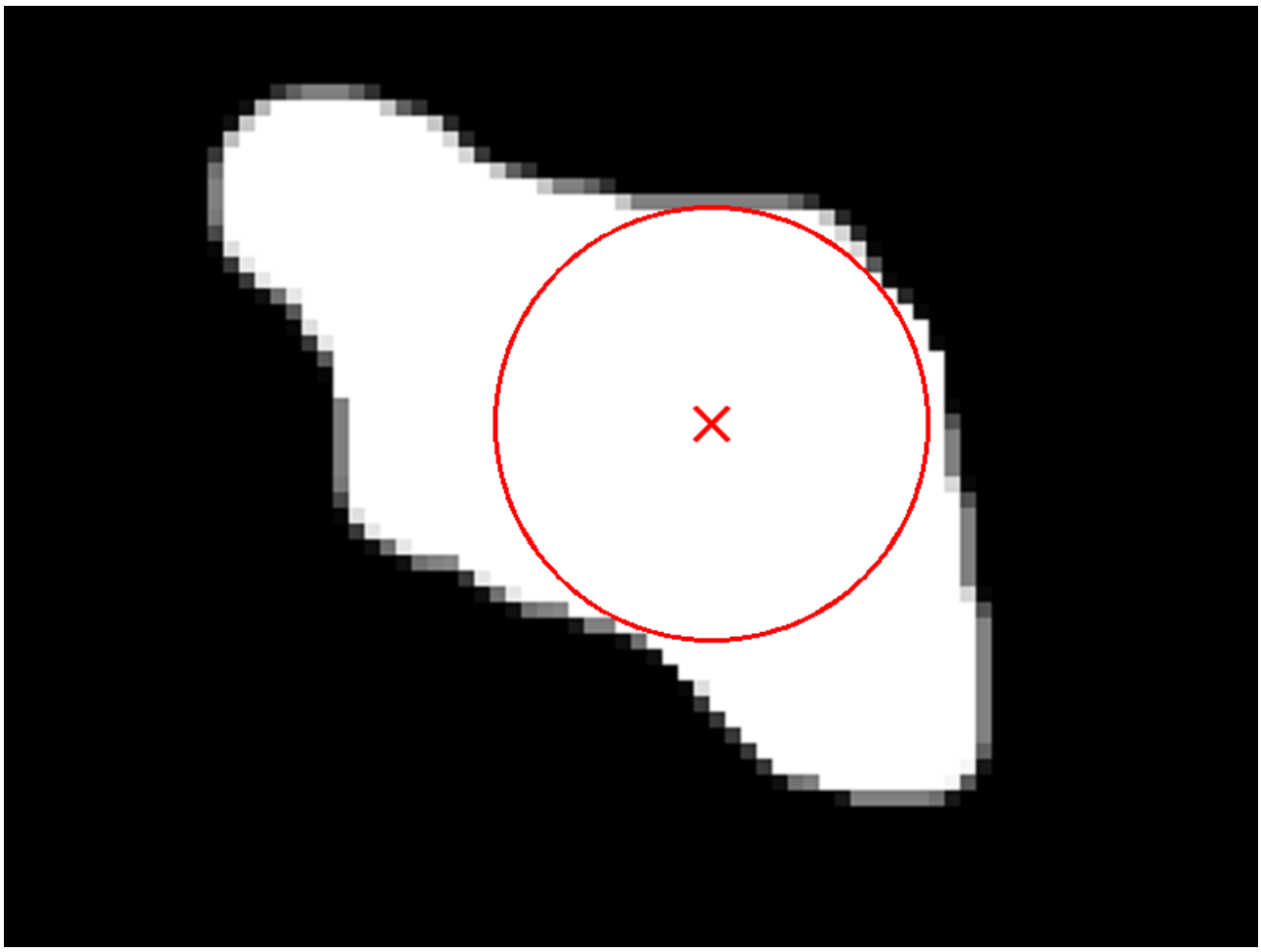
8.1.6.4 通过距离变换实现内圈

在这个样本中，你将通过距离变换的方法找到一个畸形物体的内圈。为此，基本上可以使用以下函数：

- F_VN_DistanceTransformation [▶ 1218]
- F_VN_DrawCircle [▶ 1003]

说明

外部轮廓的近似值，如包围的矩形或凸包络，可以通过适当的 API 功能计算。反之，通过距离变换，可以找到轮廓内最大尺寸的内圈。



为此，通过距离变换确定与物体边缘存在最大欧氏距离的点。然后这个点就是物体内圈的中心点。

如果通过自己的图像使用这个样本，应该通过变量bInvert和fThreshold调整阈值和二值化。通过参数eDistanceType和eMaskSize，可以有目的地改变函数F_VN_DistanceTransformation [▶ 1218]的参数。

程序

第一步是对输入图像进行二值化。这实现了物体和背景之间的清晰区分。二值化对距离变换很重要，因为距离变换只对数值为 0 的像素有反应。如果物体是白色背景上的黑色，必须通过阈值类型BINARY_INV反转。

```
IF bInvert THEN
    eThresholdType := TCVN_TT_BINARY_INV;
ELSE
    eThresholdType := TCVN_TT_BINARY;
END_IF
hr := F_VN_Threshold(ipImageIn, ipImageWork, fThreshold, 255, eThresholdType, hr);
```

通过距离变换，你可以得到一个图像，其中每个像素的值表示与下一个值为 0 的像素的距离。因此，具有最高值的像素是距离最大的像素。如果使用 L2 准则（欧氏距离）作为距离类型eDistanceType，你可以把这个像素解释为一个尽可能填满物体的圆的中心点。因此，通过函数F_VN_MaxPixelValue [▶ 1231]确定具有最大值的像素，并将其位置作为圆的中心点，而其数值作为圆的半径。

```
hr := F_VN_DistanceTransformation(
    ipSrcImage      :=  ipImageWork,
    ipDestImage     :=  ipImageWork,
    eDistanceType   :=  TCVN_DT_L2,
    eMaskSize       :=  TCVN_DTM_5,
    hrPrev          :=  hr
);
hr := F_VN_MaxPixelValue(ipImageWork, aMax, aPos, hr);
fRadius := aMax[0];
aCenter := aPos;
```

掩码大小eMaskSize决定了距离计算的准确性，且对结果影响不大。

最后，在确定的中心点和半径的基础上，在结果图像中画出圆，以便能够验证执行情况。

```
hr := F_VN_DrawCircle(
    nCenterX    :=  TO_UDINT(aCenter[0]),
    nCenterY    :=  TO_UDINT(aCenter[1]),
    nRadius     :=  TO_UDINT(fRadius),
    ipDestImage :=  ipImageRes,
    aColor      :=  aRed,
    nThickness  :=  2,
    hrPrev      :=  hr
);
hr := F_VN_DrawPointExp(TO_UDINT(aCenter[0]), TO_UDINT(aCenter[1]), ipImageRes, TCVN_DS_X, aRed, 10,
2, TCVN_LT_ANTIALIASED, hr);
```

8.1.7 图像颜色和对比度处理

8.1.7.1 与 RGB 参考色的颜色相似度

在这个样本中，函数F_VN_ReferenceColorSimilarityExp_TcVnVector3_LREAL [▶ 1264]计算每个像素与 3 种参考颜色的相似度，并将它们作为灰度图像返回。然后，根据这些灰度图像，检测参考色的物体。

变量

```
hr                  :   HRESULT;

// Images
ipImageIn           :   ITcVnImage;
ipImageInDisp       :   ITcVnDisplayableImage;

ipImageThres        :   ITcVnImage;
ipImageThresDisp    :   ARRAY [0..2] OF ITcVnDisplayableImage;
ipImageWorkCol      :   ITcVnImage;
ipImageWorkColDisp  :   ARRAY [0..2] OF ITcVnDisplayableImage;

ipImageRes          :   ITcVnImage;
ipImageResDisp      :   ITcVnDisplayableImage;

// Color
iColor              :   INT;
aColorTxt           :   ARRAY[0..2] OF STRING := [ 'RED', 'GREEN', 'BLUE' ];
aColor              :   ARRAY[0..2] OF TcVnVector4_LREAL := [ [150, 0, 0], [0, 255, 0],
[0, 0, 255] ];
aColorRef           :   ARRAY[0..2] OF TcVnVector3_LREAL := [ [255, 75, 60], [40, 140, 95],
[40, 140, 190] ];

// Contours
ipContourList       :   ITcVnContainer;
ipIterator          :   ITcVnForwardIterator;
aOffset             :   TcVnPoint;
ipContour           :   ITcVnContainer;
fArea               :   LREAL;
aCenter             :   TcVnPoint2_LREAL;
```

代码

```
// Attention: With other images another color space transformation could be necessary
hr := F_VN_ConvertColorSpace(ipImageIn, ipImageRes, TCVN_CST_Bayer_RG_TO_RGB, hr);

FOR iColor := 0 TO 2 DO

   // Compute the Color Similarity to a Reference Color
   hr := F_VN_ReferenceColorSimilarityExp_TcVnVector3_LREAL(
                   ipSrcImage          := ipImageRes,
                   ipDestImage         := ipImageWorkCol,
                   aRefColor           := aColorRef[iColor],
                   fVariance           := 0.1,
                   fLuminanceWeight    := 0.2,
                   hrPrev              := hr);

   hr := F_VN_Threshold(ipImageWorkCol, ipImageThres, 200, 255, TCVN_TT_Binary, hr);

   // Find all objects / contours in the black and white image
   hr := F_VN_FindContours(ipImageThres, ipContourList, hr);
   hr := F_VN_GetForwardIterator(ipContourList, ipIterator, hr);

   // Filter the objects by size and draw the contours
   WHILE SUCCEEDED(hr) AND_THEN ipIterator.CheckIfEnd() <> S_OK DO
      hr := F_VN_GetContainer(ipIterator, ipContour, hr);
      hr := F_VN_IncrementIterator(ipIterator, hr);

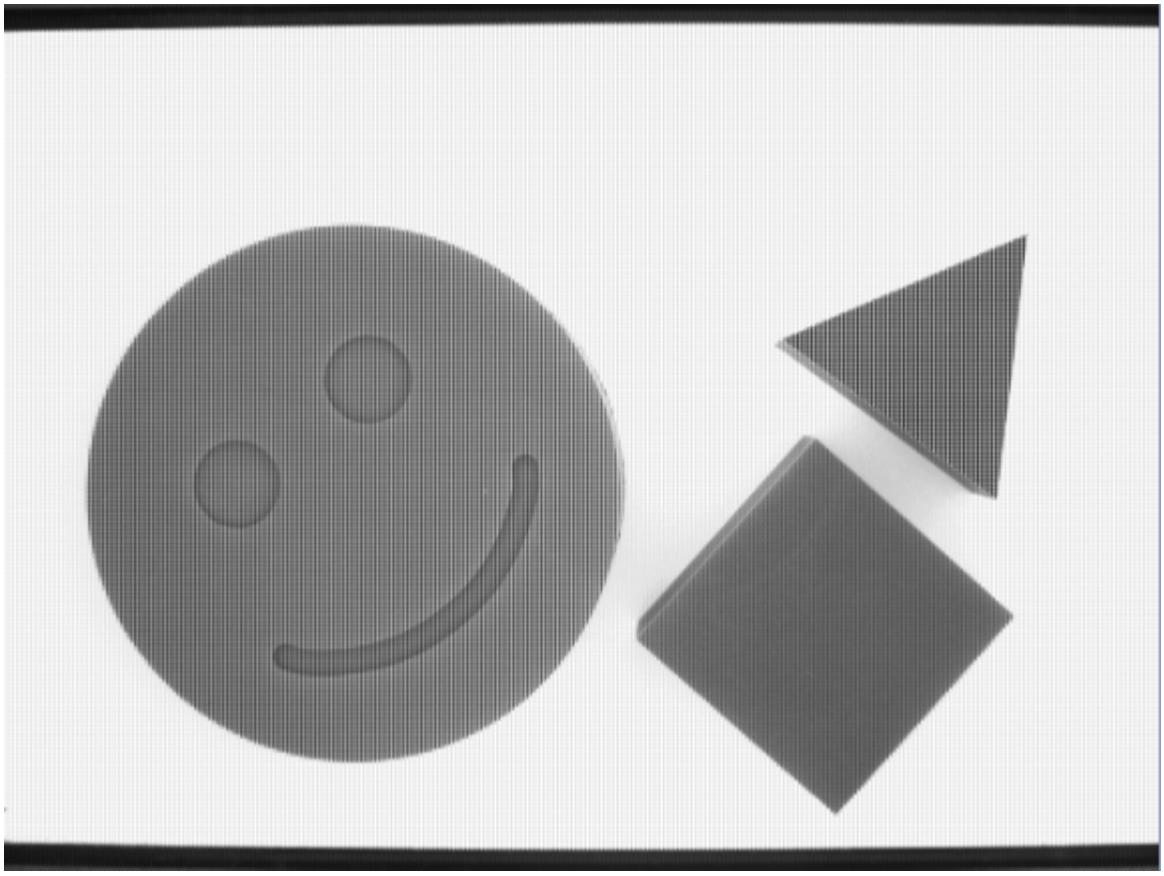
      // Filter contours by size
      hr := F_VN_ContourArea(ipContour, fArea, hr);
      IF fArea > 5000 THEN
         // Draw Results into an Image
```

```
        hr := F_VN_DrawContours(ipContour, -1, ipImageRes, aColor[iColor], 3, hr);
        hr := F_VN_ContourCenterOfMass(ipContour, aCenter, hr);
        hr := F_VN_PutText(aColorTxt[iColor], ipImageRes, LREAL_TO_UDINT(aCenter[0])-30,
LREAL_TO_UDINT(aCenter[1])+10, TCVN_FT_HERSHEY_PLAIN, 2, aColor[iColor],hr);
     END_IF
   END_WHILE

   // Display effect of the Color-Similarity-Function
   hr := F_VN_TransformIntoDisplayableImage(ipImageThres, ipImageThresDisp[iColor], hr);
   hr := F_VN_TransformIntoDisplayableImage(ipImageWorkCol, ipImageWorkColDisp[iColor], hr);
END_FOR
```

结果

Bayer RG 格式的输入图像

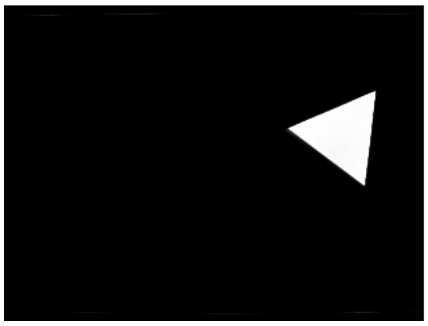


在通过函数F_VN_ConvertColorSpace [▶ 1248]将输入图像转换为 RGB 图像后，F_VN_ReferenceColorSimilarityExp_TcVnVector3_LREAL [▶ 1264]可用于确定每个像素与参考颜色的颜色相似度。以下内容适用于返回的结果图像：

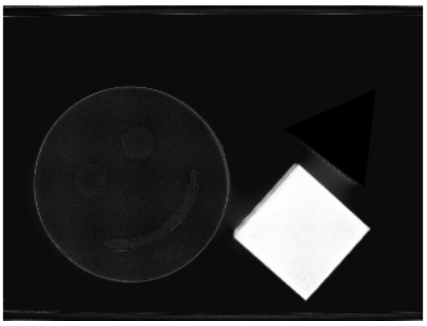
- 像素值 255（白色）对应于与参考颜色的 100% 匹配
- 像素值 0（黑色）对应于与参考颜色的 0% 匹配
- 0 至 255 之间的像素值对应于与参考颜色部分相关。

```
// Compute the Color Similarity to a Reference Color
hr := F_VN_ReferenceColorSimilarityExp_TcVnVector3_LREAL(
              ipSrcImage        := ipImageRes,
              ipDestImage       := ipImageWorkCol,
              aRefColor         := aColorRef[iColor],
              fVariance         := 0.1,
              fLuminanceWeight  := 0.2,
              hrPrev            := hr);
```

aColorRef := [255,75,60]; (RED)



aColorRef := [40,140,95]; (GREEN)



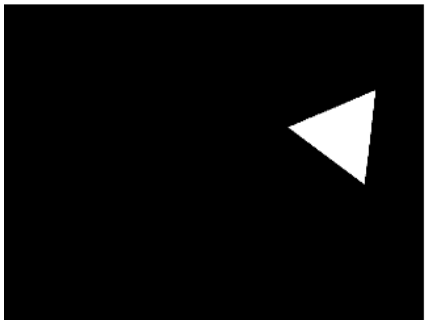
aColorRef := [40,140,190]; (BLUE)



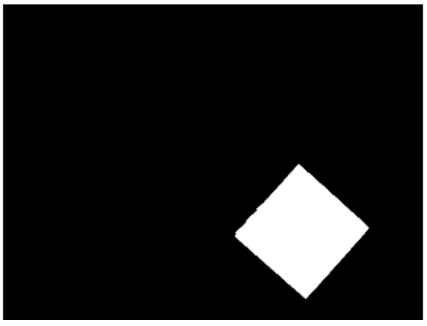
阈值（F_VN_Threshold [▶ 1322]）用于定义与参考颜色的匹配程度，以便将其分配给参考颜色。

```
hr := F_VN_Threshold(ipImageWorkCol, ipImageThres, 200, 255, TCVN_TT_Binary, hr);
```

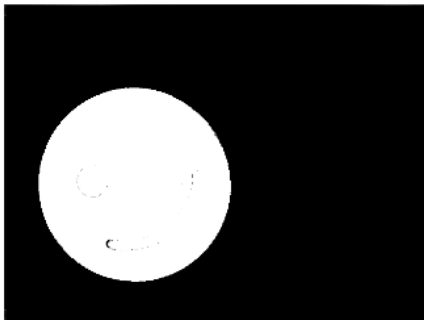
aColorRef := [255,75,60]; (RED)



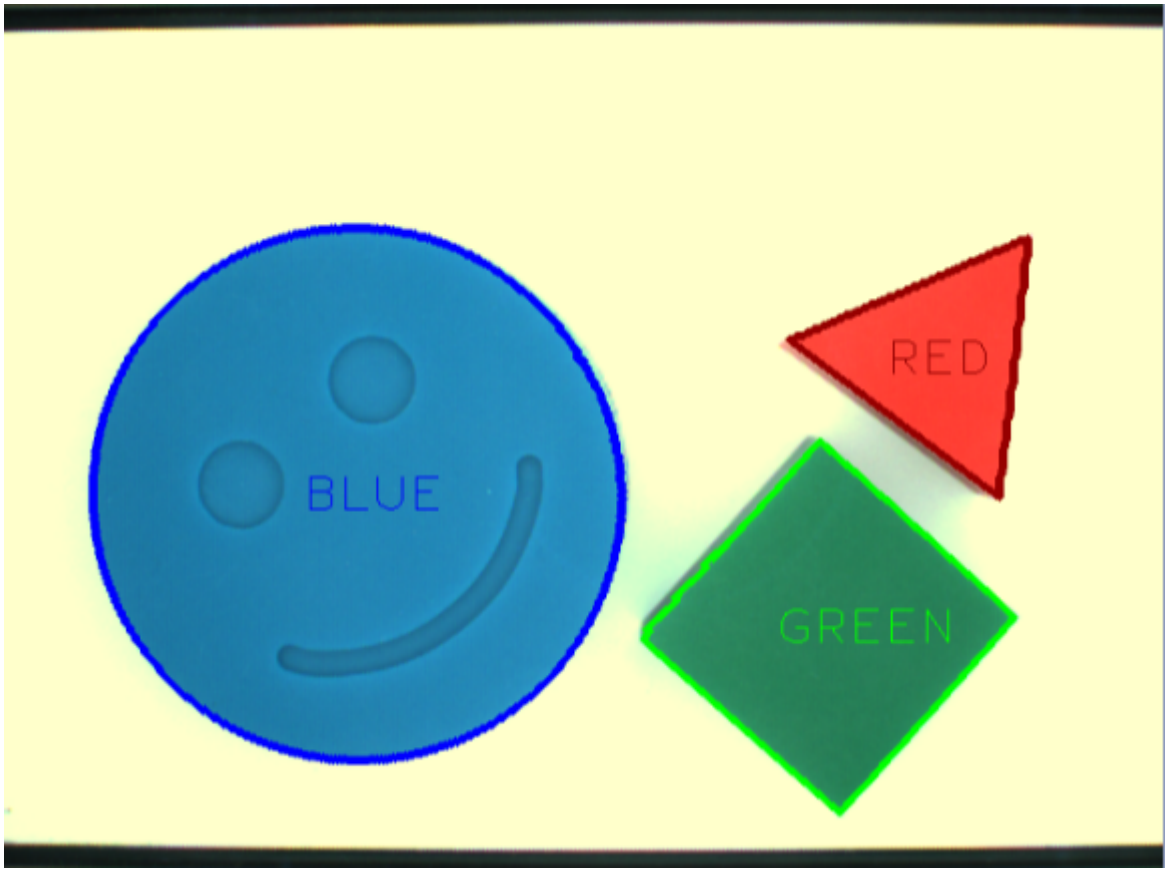
aColorRef := [40,140,95]; (GREEN)



aColorRef := [40,140,190]; (BLUE)



随后检测后的结果图像。



类似样本

- 检查颜色范围 [▶ 2790]

8.1.7.2 通过颜色表显示

在这个样本中，会将颜色表应用于灰度图像。大体上可以使用以下功能：

- F_VN_GenerateColorMap [▶ 1249]
- F_VN_CreateContainerFromArray [▶ 752]
- F_VN_GenerateCustomColorMap [▶ 1250]
- F_VN_ApplyColorMap [▶ 1244]

说明

有几个原因可以说明为什么想在单色图像上应用颜色表。这些包括，例如更令人愉快的外观和对人眼来说更清晰的数值范围的显示。这首先与 16 位图像有关，因为如果数值频谱分布在三个颜色通道上，那么可识别性会大大增强。

根据使用情况，可以使用各种颜色的表格。TwinCAT Vision 通过枚举ETcVnColorMap [▶ 165]提供一些预定义的颜色梯度。此外，你还可以定义你自己的颜色梯度，并用它们创建颜色表。

在这个示例中，你将各种颜色表应用于以下灰度梯度（数值范围为 16 位）。



附图 29: 灰度梯度

当然，你也可以使用自己的图像。请注意，仅可使用单通道图像，否则必须事先使用F_VN_ConvertColorSpace [▶ 1248]进行适当的转换。

程序

在将颜色表应用于图像之前，必须首先创建颜色表。为此，必须知道颜色表用于 8 位还是 16 位图像，因为根据不同的情况，需要不同数量的条目（见颜色表）。因此，必须将类型为ETcVnColorMapSize [▶ 166]的参数`eColorMapSize`设置为`256`（如果是 8 位）和`65536`（如果是 16 位）以与输入图像`ipImageIn`的位深相匹配。通过F_VN_GetImageInfo [▶ 800]查询像素格式的元素大小来实现：

```
hr := F_VN_GetImageInfo(ipImageIn, stImageInfo, hr);
IF stImageInfo.stPixelFormat.nElementSize = 8 THEN
    eColorMapSize := TCVN_CMS_256;
ELSIF stImageInfo.stPixelFormat.nElementSize = 16 THEN
    eColorMapSize := TCVN_CMS_65536;
END_IF
```

根据颜色梯度，生成一个颜色表。可以手动定义颜色梯度或者使用模板中的颜色梯度。这个样本根据参数`bUseCustomColors`处理区分。

```
IF bUseCustomColors THEN
    // create color map from custom-defined colors
    <...>
ELSE
    // create color map from preset
    <...>
END_IF
```

通过创建包含多个条目的 TcVnVector3_REAL 类型容器，可以手动定义颜色梯度。每个条目代表一种颜色。在创建颜色表时，颜色被解释为均匀地分布在定义区域；所有其他数值都是对三个通道中的每一个进行线性插值。此处使用了来自阵列`aColors`的颜色梯度，包括红、绿、蓝三种颜色。然后，使用函数 F_VN_GenerateCustomColorMap 创建颜色表。

```
VAR
    aColors: ARRAY [0..(N_COLORS-1)] OF TcVnVector3_REAL := [
    [255, 0, 0],
    [0, 255, 0],
```

```
    [0, 0, 255]
    ];
END_VAR

hr := F_VN_CreateContainerFromArray(ADR(aColors), ipColors, ContainerType_Vector_TcVnVector3_REAL,
N_COLORS, hr);
hr := F_VN_GenerateCustomColorMap(
    ipColorMap      :=  ipColorMap,
    ipInitialColors :=  ipColors,
    eColorMapSize   :=  eColorMapSize,
    hrPrev          :=  hr
);
```

为了使用预定义的颜色梯度，请使用函数F_VN_GenerateColorMap [▶ 1249]，并从枚举ETcVnColorMap [▶ 165]将参数eColorMap设置为需要的颜色梯度。在这种情况下，最初设置为JET。

```
VAR
    eColorMap: ETcVnColorMap := TCVN_CM_JET;
END_VAR

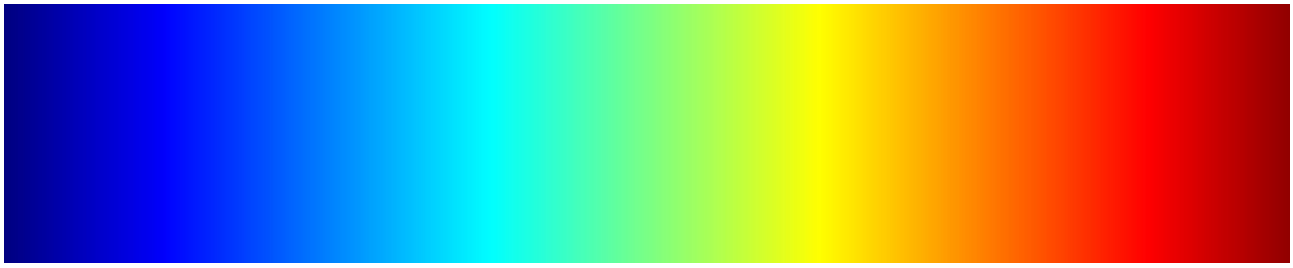
hr := F_VN_GenerateColorMap(
    ipColorMap      :=  ipColorMap,
    eColorMap       :=  eColorMap,
    eColorMapSize   :=  eColorMapSize,
    hrPrev          :=  hr
);
```

创建的颜色表是一个容器 [▶ 146]，其中存储了输入图像中每个可能的值所要发出的颜色。因此，也可以通过定义所有 256 个或 65536 个条目，然后直接从这些条目中创建一个容器，自己创建一个颜色表。然而，在大多数情况下，使用 API 功能更为简单。

最后，将创建的颜色表应用于输入图像，以获得理想的结果。结果图像总是一个 3 通道的 8 位图像。

```
hr := F_VN_ApplyColorMap(
    ipSrcImage      :=  ipImage,
    ipDestImage     :=  ipImage,
    ipColorMap      :=  ipColorMap,
    hrPrev          :=  hr
);
```

如果将预定义的 JET 颜色表应用于上图所示的输入图像，会产生以下结果：



附图 30: 应用 JET 颜色表后的颜色梯度

8.1.8 图像过滤

8.1.8.1 模糊过滤器

在这个样本中，比较了以下模糊过滤器：

- 双边过滤器 [▶ 1278]
- 高斯过滤器 [▶ 1295]
- 中位数过滤器 [▶ 1302]
- 作为自定义过滤器 [▶ 1287]实现的平均数过滤器

说明

在某些情况下，图像需要平滑化或摆脱噪音。对于读码这样的任务，需要同时在图像上获得良好的对比度和良好的边缘，以便下游的算法能够成功地工作。

双边过滤器特别适用于将图像平滑与同时保留边缘相结合。对于每个像素的强度，它计算出周围像素的加权平均值。加权不仅取决于像素的距离有多远，还取决于强度的差异。因此，尖锐的边缘被保留下来。

高斯过滤器通过高斯钟形函数的二维离散近似值平滑图像。这样可以减少图像的噪音。较小的结构会丢失，但另一方面，较粗的结构会保留。因此，与下面两个过滤器相比，边缘的模糊程度较低，但较大的干扰没有被很好地去除。

中位数过滤器将指定掩码的所有像素值按升序排序，然后选择之后相应像素设置的中位数。这种过滤器很适合抑制小于所应用过滤器掩码的结构（干扰），并且对异常值非常稳健。因此，可以最好地去除小的椒盐干扰等，但这样做的缺点是，其他的详细信息也会丢失。

为了形成一个平均值，使用了**自定义过滤器**功能。在这里，每个像素的值都被内核像素的平均值所取代，这导致了模糊化。因此，干扰（噪音）会受到抑制，但随着内核数量的增加，细节和边缘逐渐变得模糊不清。由于自定义过滤器功能为应用自己的过滤器提供了许多不同的选择，为了实现过滤器内核的标准化，还需要`F_VN_InitMatrixStruct`功能。

变量

```
// Bilateral Filter
ipImageBilateral          :   ITcVnImage;
ipImageBilateralDisp      :   ITcVnDisplayableImage;
nBilateral_Diameter       :   DINT  := 7;
fBilateral_SigmaColor     :   LREAL := 100;
fBilateral_SigmaSpace     :   LREAL := 100;

// Gaussian Filter
ipImageGaussian           :   ITcVnImage;
ipImageGaussianDisp       :   ITcVnDisplayableImage;
nFilterWidth              :   UDINT := 7;
nFilterHeight             :   UDINT := 7;

// Median Filter
ipImageMedian             :   ITcVnImage;
ipImageMedianDisp         :   ITcVnDisplayableImage;
nMedian_FilterSize        :   UDINT := 7;

// Custom Filter e.g. Mean
ipImageCustom             :   ITcVnImage;
ipImageCustomDisp         :   ITcVnDisplayableImage;
stKernelMatrix            :   TcVnMatrix;
// 7x7 Mean Filter Kernel with weights of 1/49 ~ 0.0204081632653
aMatrixArray7x7           :   ARRAY [0..6,0..6] OF REAL := [49(0.0204081632653)];
```

代码

```
hr := F_VN_BilateralFilter(
        ipSrcImage      := ipImageIn,
        ipDestImage     := ipImageBilateral,
        nDiameter       := nBilateral_Diameter,
        fSigmaColor     := fBilateral_SigmaColor,
        fSigmaSpace     := fBilateral_SigmaSpace,
        hrPrev          := hr);

hr := F_VN_GaussianFilter(
        ipSrcImage      := ipImageIn,
        ipDestImage     := ipImageGaussian,
        nFilterWidth    := nFilterWidth,
        nFilterHeight   := nFilterHeight,
        hrPrev          := hr);

hr := F_VN_MedianFilter(
        ipSrcImage      := ipImageIn,
        ipDestImage     := ipImageMedian,
        nFilterSize     := nMedian_FilterSize,
        hrPrev          := hr);

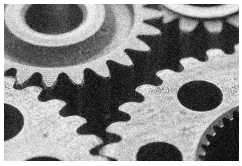
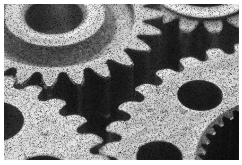
// Mean-Filter with Custom Filter Function
hr := F_VN_InitMatrixStruct(
        pSrcBuffer      := ADR(aMatrixArray7x7),
        stDestMatrix    := stKernelMatrix,
        nRows           := 7,
        nCols           := 7,
        eElementType    := TCVN_ET_REAL,
        hrPrev          := hr);

hr := F_VN_CustomFilter(
        ipSrcImage      := ipImageIn,
        ipDestImage     := ipImageCustom,
        eDestDepth      := TCVN_ET_USINT,
        stKernel        := stKernelMatrix,
        hrPrev          := hr);
```

结果

未经处理的原始图像（第一行）在细节上已经显示出齿轮表面的精细结构。为了展示过滤器的效果，在原始图像中加入了高斯噪声（第二行）和椒盐噪声（第三行）等额外干扰。

为了使不同过滤器的效果尽可能具有可比性，所有的过滤器都使用了 7×7 的内核尺寸。如果使用其他参数值或内核大小以及功能的 EXP 变体的更多设置选项，可以具体地实现其他效果。由于过滤器的效果在很大程度上取决于输入图像的内容，建议通过不同的参数设置测试几个过滤器，以便直接比较并找到最适合使用情况的过滤器。下表包含了结果图像的概览：

原始图像	双边过滤器	高斯过滤器	中位数过滤器	平均数过滤器
				
				
				

注意

执行时间长或波动大

过滤器的执行时间基本上取决于过滤器的类型以及图像和内核的大小。因此，请注意，过滤器操作可能比其他功能需要更长的时间，所以应该相应地调整任务的周期时间。

如果过滤器的参数在运行期间更改，特别是内核大小，建议使用看门狗 [▶ 140]。对于中位数过滤器而言，由于原理的原因，运行时间总是会有一些波动。

此外，工作任务 [▶ 66]可导致总计算时间的减少，这应根据系统的情况进行测试。

8.1.8.2 边缘检测过滤器

在本示例中，对以下边缘检测滤波器进行了比较：

- 索贝尔（Sobel）滤波器 [▶ 1312]
- 沙尔（Scharr）滤波器 [▶ 1306]
- 拉普拉斯（Laplace）滤波器 [▶ 1297]

说明

索贝尔（Sobel）滤波器是一种用于标记图像中梯度的简单线性滤波器。沙尔（Scharr）滤波器与前者非常相似；仅在滤波器核的特性方面有所不同。沙尔（Scharr）滤波器具有更好的各向同性，即边缘的检测与方向无关。

拉普拉斯（Laplace）滤波器根据图像离散二阶导数的零点来检测边缘。相比之下，这种变体容易受到噪声的影响。

变量

```
// Sobel
ipImageSobel            :   ITcVnImage;
ipImageSobelDisp        :   ITcVnDisplayableImage;
```

```
eSobel_DestDepth        :   ETcVnElementType := TCVN_ET_USINT;
aSobel_DerivOrder       :   ARRAY [0..1] OF UDINT := [1, 1];
nSobel_KernelSize       :   UDINT := 3;
fSobel_Scale            :   LREAL := 10;
fSobel_Delta            :   LREAL := 0;
eSobel_BorderExtra      :   ETcVnBorderInterpolationMethod := TCVN_BIM_DEFAULT;

// Scharr
ipImageScharr           :   ITcVnImage;
ipImageScharrDisp       :   ITcVnDisplayableImage;
eScharr_DestDepth       :   ETcVnElementType := TCVN_ET_USINT;
eScharr_FilterDir       :   ETcVnFilterDirection := TCVN_FD_X;
fScharr_Scale           :   LREAL := 1;
fScharr_Delta           :   LREAL := 0;
eScharr_BorderExtra     :   ETcVnBorderInterpolationMethod := TCVN_BIM_DEFAULT;

// Laplace
ipImageLaplace          :   ITcVnImage;
ipImageLaplaceDisp      :   ITcVnDisplayableImage;
eLaplace_DestDepth      :   ETcVnElementType := TCVN_ET_USINT;
nLaplace_KernelSize     :   UDINT := 3;
fLaplace_Scale          :   LREAL := 10;
fLaplace_Delta          :   LREAL := 0;
eLaplace_BorderExtra    :   ETcVnBorderInterpolationMethod := TCVN_BIM_DEFAULT;
```

代码

```
// Execute the Sobel Filter
hr := F_VN_SobelFilterExp(
        ipSrcImage      :=  ipImageIn,
        ipDestImage     :=  ipImageSobel,
        eDestDepth      :=  eSobel_DestDepth,
        nXOrder         :=  aSobel_DerivOrder[0],
        nYOrder         :=  aSobel_DerivOrder[1],
        nKernelSize     :=  nSobel_KernelSize,
        fScale          :=  fSobel_Scale,
        fDelta          :=  fSobel_Delta,
        eBorderType     :=  eSobel_BorderExtra,
        hrPrev          :=  hr);

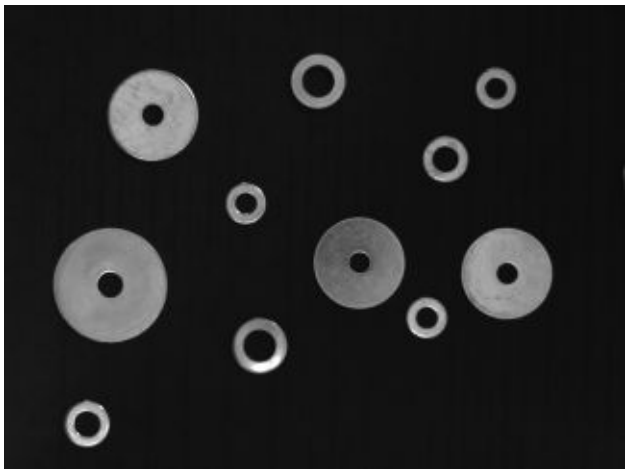
// Execute the Scharr Filter
hr := F_VN_ScharrFilterExp(
        ipSrcImage      :=  ipImageIn,
        ipDestImage     :=  ipImageScharr,
        eDestDepth      :=  eScharr_DestDepth,
        eFilterDirection:=  eScharr_FilterDir,
        fScale          :=  fScharr_Scale,
        fDelta          :=  fScharr_Delta,
        eBorderType     :=  eScharr_BorderExtra,
        hrPrev          :=  hr);

// Execute the Laplacian Filter
hr := F_VN_LaplacianFilterExp(
         ipSrcImage     :=  ipImageIn,
         ipDestImage    :=  ipImageLaplace,
         eDestDepth     :=  eLaplace_DestDepth,
         nKernelSize    :=  nLaplace_KernelSize,
         fScale         :=  fLaplace_Scale,
         fDelta         :=  fLaplace_Delta,
         eBorderType    :=  eLaplace_BorderExtra,
         hrPrev         :=  hr);
```

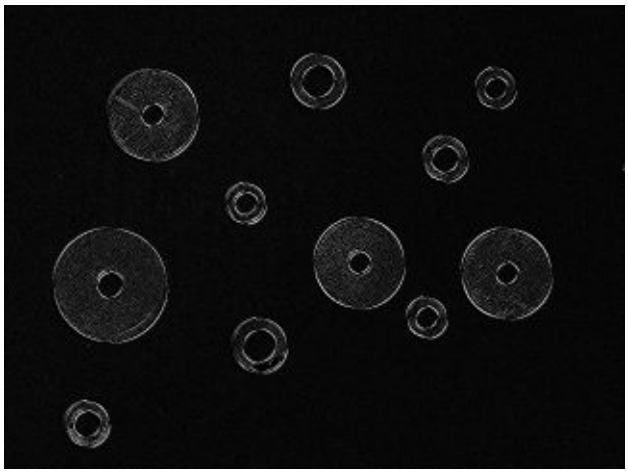
结果

示例项目中包含三个滤波器的函数调用。在本示例中可以测试参数配置及其对示例图像的影响。

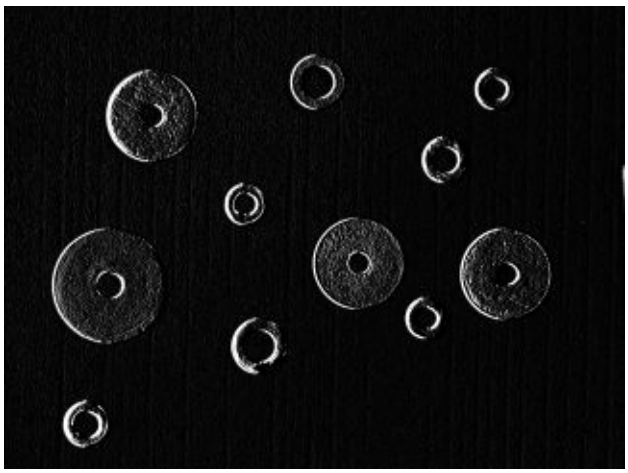
原始图像 **索贝尔（Sobel）滤波器**

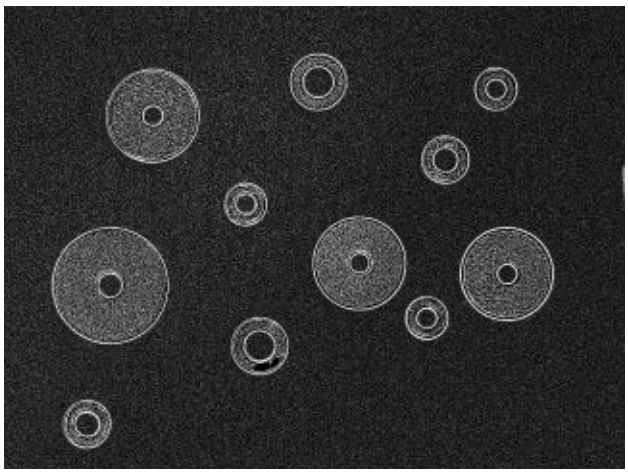


沙尔（Scharr）滤波器



拉普拉斯（Laplace）滤波器





8.1.8.3 形态学算子

在本示例中，您将对图像应用形态学算子。为此，您可以使用：

- F_VN_CreateStructuringElement [▶ 1286]
- F_VN_MorphologicalOperator [▶ 1303]

说明

根据特定规则，利用形态学算子可以改变图像中的对象。形态学算子大多应用于二值图像；但通常也可应用于灰度图像。例如，通过组合算子（如 OPENING 和 CLOSING），可以补全对象中缺失的部分并去除小的异常值。

变量

```
ipElement    :  ITcVnImage;
eSEShape     :  ETcVnStructuringElementShape := TCVN_SES_RECTANGLE;
aSESize      :  ARRAY [0..1] OF UDINT := [29, 29];
eOperator    :  ETcVnMorphologicalOperator := TCVN_MO_OPENING;
```

程序

形态学算子需要结构元素来定义其针对每个像素的影响区域。为此，您可以从枚举 ETcVnStructuringElementShape [▶ 211] 中选择不同的形状和不同的尺寸。

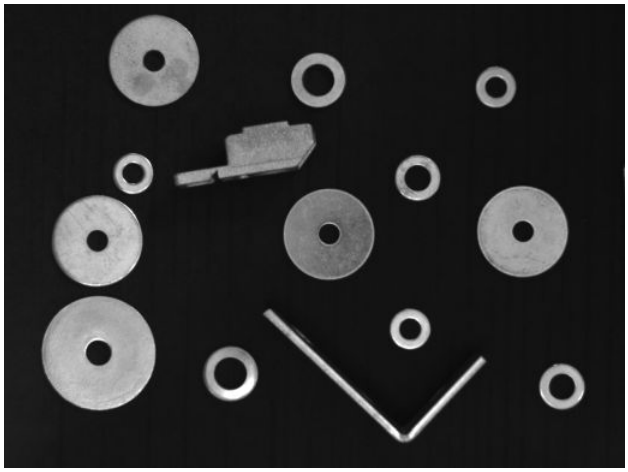
```
hr := F_VN_CreateStructuringElement(
    ipStructuringElement    :=  ipElement,
    eShape                  :=  eSEShape,
    nWidth                  :=  aSESize[0],
    nHeight                 :=  aSESize[1],
    hrPrev                  :=  hr);
```

然后，您可以通过枚举 ETcVnMorphologicalOperator [▶ 202] 定义一种形态学运算，并使用所创建的结构元素将其应用于输入图像。

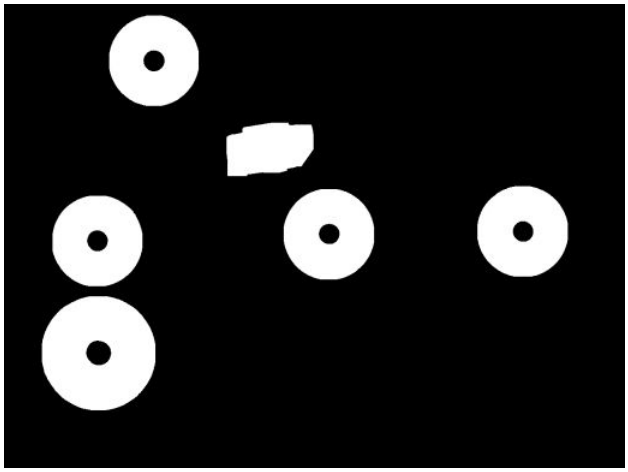
```
hr := F_VN_MorphologicalOperator(
    ipSrcImage              :=  ipImageIn,
    ipDestImage             :=  ipImageRes,
    eOperator               :=  eOperator,
    ipStructuringElement    :=  ipElement,
    hrPrev                  :=  hr);
```

例如，使用尺寸为 [31; 31] 的矩形结构元素并结合先前的阈值进行形态学开运算，将产生如下结果。它会从图像中移除较小或较窄的对象。

输入图像



结果图像



注意

执行时间长

根据输入图像和结构元素的大小，与其他函数相比，形态学算子的执行可能需要很长的时间。确保相应设置循环时间，必要时使用 Watchdog（看门狗）！

8.1.9 图像分割

- 检查颜色范围 [▶ 2790]

8.1.9.1 检查颜色范围

在本示例中，函数 F_VN_CheckColorRange [▶ 1319] 在图像中搜索红色、绿色和蓝色对象，并根据它们的颜色对其进行标注。该函数适用于 RGB 或 HSV 图像。也可以将该函数应用于其他颜色空间中的图像，如 Lab、BGR 或 YCrCb。在这种情况下，必须对图像转换和阈值参数设置进行相应调整。

说明

该程序允许为颜色范围分割选择颜色空间。变量 `eColorSpace` 可用于在 RGB 和 HSV 两种变体之间进行切换。根据所选的颜色空间，可使用不同的阈值数组（`aColorRefLowRGB/aColorRefUpRGB` 适用于 RGB，`aColorRefLowHSV/aColorRefUpHSV` 适用于 HSV）。这些变量包含分别为要检测的三种颜色（红色、绿色、蓝色）定义的下限值和上限值。引用 `refColorLow` 和 `refColorUp` 分别指向当前激活的数组。

输入图像为 Bayer RG 格式，通过 F_VN_ConvertColorSpace [▶ 1248] 函数转换为 RGB 图像。在 RGB 模式下，直接使用转换后的图像；而在 HSV 模式下，则会通过变换 `TCVN_CST_RGB_TO_HSV_FULL` 执行一次从 RGB 到 HSV 的额外颜色空间转换。选择 `HSV_FULL` 变体后，色调通道可以使用完整的 8 位数值范围（0-255），而不是像 `HSV` 那样限制在 0-179，从而能够实现更精细的参数设置。HSV 颜色空间在照明不均匀的条件下通常可以提供更可靠的结果，因为色调值（色调）与色彩饱和度及亮度（值）相分离。在叠加颜色空间RGB中，亮度的变化通常会导致所有 3 个通道的数值发生改变，这就很难定义用于区分颜色的阈值。

最后可以使用 F_VN_CheckColorRange [▶ 1319] 函数和相应的上下限阈值。

变量

```
eColorSpace             :   (RGB, HSV); // Selected color space

hr                      :   HRESULT;

// Images
ipImageIn               :   ITcVnImage;
ipImageInDisp           :   ITcVnDisplayableImage;

ipImageColorSpace       :   ITcVnImage;
ipImageColorSpaceDisp   :   ITcVnDisplayableImage;

ipImageWorkCol          :   ITcVnImage;
ipImageWorkColDisp      :   ARRAY [0..2] OF ITcVnDisplayableImage;

ipImageRes              :   ITcVnImage;
ipImageResDisp          :   ITcVnDisplayableImage;

// Colors
iColor                  :   INT;
aColorTxt               :   ARRAY[0..2] OF STRING := [ 'RED', 'GREEN', 'BLUE' ];
aColor                  :   ARRAY[0..2] OF TcVnVector4_LREAL := [ [150, 0, 0], [0, 255, 0], [0, 0,
255] ];
refColorBoundsLow       :   REFERENCE TO ARRAY[0..2] OF TcVnVector4_LREAL;
refColorBoundsUp        :   REFERENCE TO ARRAY[0..2] OF TcVnVector4_LREAL;

// Bounds values for RGB
aColorBoundsLowRGB      :   ARRAY[0..2] OF TcVnVector4_LREAL := [ [150, 50, 20], [35, 90, 60], [20,
40, 130] ];
aColorBoundsUpRGB       :   ARRAY[0..2] OF TcVnVector4_LREAL := [ [255, 120, 100], [100, 200, 140],
[60, 160, 255] ];

// Bounds values for HSV
aColorBoundsLowHSV      :   ARRAY[0..2] OF TcVnVector4_LREAL := [ [251, 180, 205], [103, 150, 100],
[130, 180, 130] ];
aColorBoundsUpHSV       :   ARRAY[0..2] OF TcVnVector4_LREAL := [ [6, 220, 255], [113, 190, 190],
[149, 230, 210] ];

// Contours
ipContourList           :   ITcVnContainer;
ipIterator              :   ITcVnForwardIterator;
ipContour               :   ITcVnContainer;
fArea                   :   LREAL;
aCenter                 :   TcVnPoint2_LREAL;
```

代码

```
// Attention: With other images another color space transformation could be necessary
hr := F_VN_ConvertColorSpace(ipImageIn, ipImageRes, TCVN_CST_Bayer_RG_TO_RGB, hr);

// Select the color space for segmentation
CASE eColorSpace OF
    RGB:
        hr := F_VN_CopyImage(ipImageRes, ipImageColorSpace, hr);
        refColorBoundsLow REF= aColorBoundsLowRGB;
        refColorBoundsUp REF= aColorBoundsUpRGB;

    HSV:
        hr := F_VN_ConvertColorSpace(ipImageRes, ipImageColorSpace, TCVN_CST_RGB_TO_HSV_FULL, hr);
        refColorBoundsLow REF= aColorBoundsLowHSV;
        refColorBoundsUp REF= aColorBoundsUpHSV;
END_CASE

FOR iColor := 0 TO 2 DO

    // Apply a "Color-Threshold" on the image
    hr := F_VN_CheckColorRange(ipImageColorSpace, ipImageWorkCol, refColorBoundsLow[iColor],
refColorBoundsUp[iColor], hr);

    // Find all objects / contours in the black and white image
    hr := F_VN_FindContours(ipImageWorkCol, ipContourList, hr);
    hr := F_VN_GetForwardIterator(ipContourList, ipIterator, hr);

    // Filter the objects by size and draw the contours
    WHILE SUCCEEDED(hr) AND_THEN ipIterator.CheckIfEnd() <> S_OK DO

        hr := F_VN_GetContainer(ipIterator, ipContour, hr);
        hr := F_VN_IncrementIterator(ipIterator, hr);

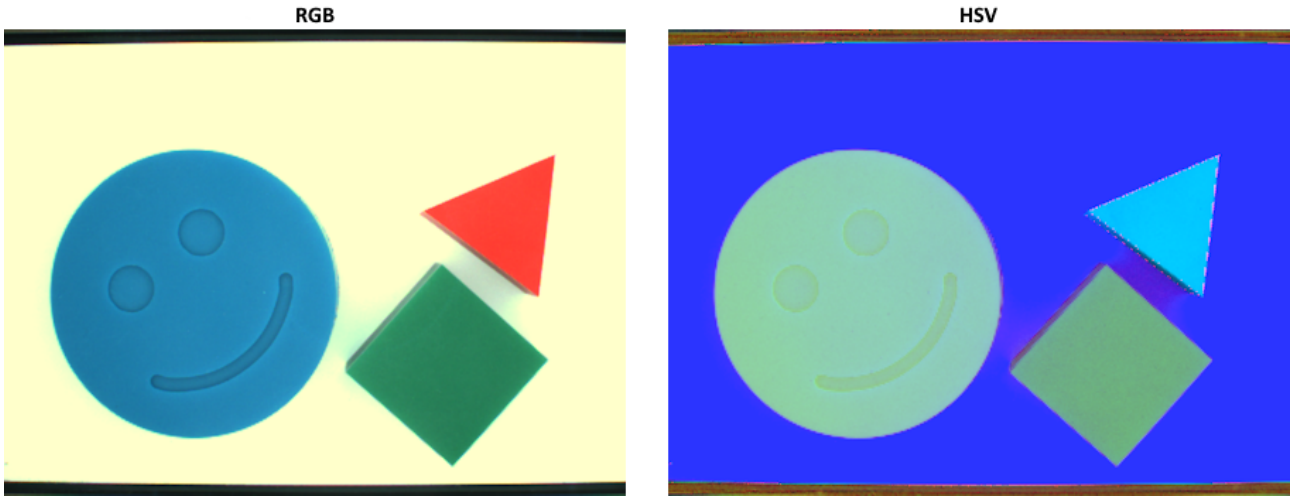
        // Filter contours by size
        hr := F_VN_ContourArea(ipContour, fArea, hr);

        IF fArea > 5000 THEN
            // Draw Results into an Image
            hr := F_VN_DrawContours(ipContour, -1, ipImageRes, aColor[iColor], 3, hr);
            hr := F_VN_ContourCenterOfMass(ipContour, aCenter, hr);
            hr := F_VN_PutText(aColorTxt[iColor], ipImageRes, LREAL_TO_UDINT(aCenter[0])-30,
LREAL_TO_UDINT(aCenter[1])+10, TCVN_FT_HERSHEY_PLAIN, 2, aColor[iColor], hr);
        END_IF
    END_WHILE
```

```
    // Display effect of the CheckColorRange-Function
    hr := F_VN_TransformIntoDisplayableImage(ipImageWorkCol, ipImageWorkColDisp[iColor], hr);
END_FOR
```

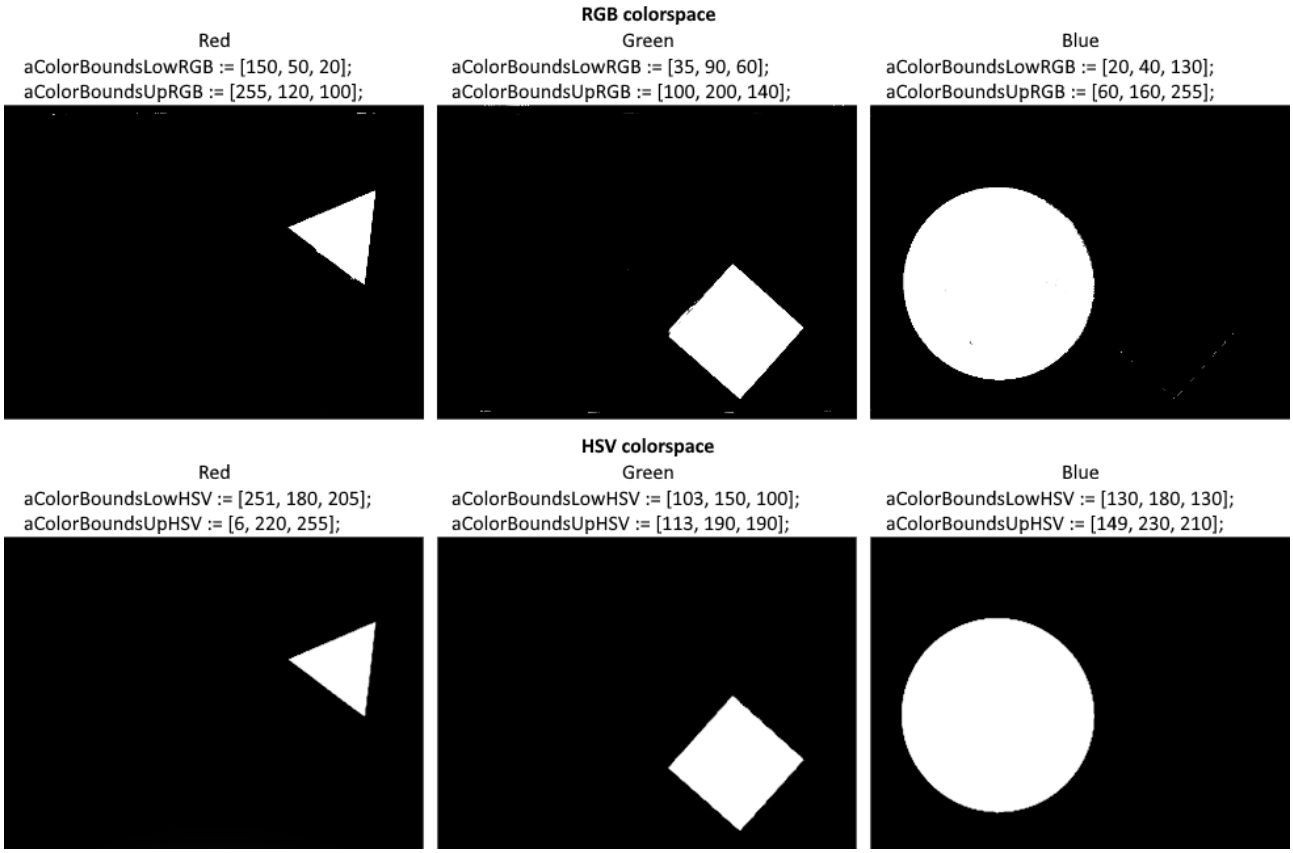
结果

在 `ipImageColorSpaceDisp` 中，所选颜色空间会在 ADS Image Watch（ADS 图像查看）中显示。这样可以分析精确的像素值，从而优化阈值参数设置。HSV 图像以伪彩色进行显示，因为 H、S 和 V 通道被解释为 RGB。

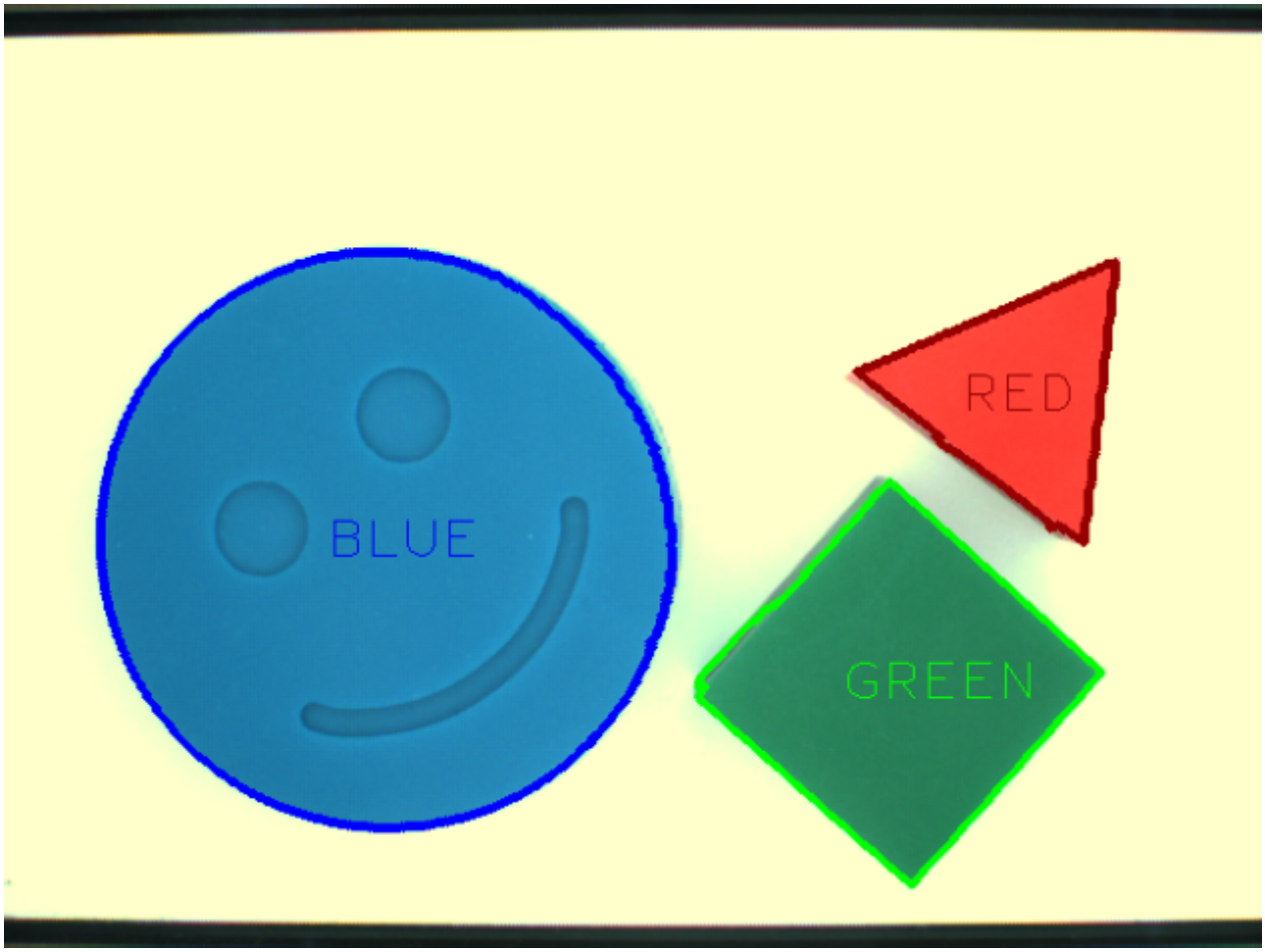


二值图像 `ipImageWorkColDisp[0..2]` 展示了每种颜色的分割结果。如果选择 RGB 作为颜色空间，通常在对象的边缘处可见伪影。这是因为所有三个颜色通道都与亮度相关。因此，对象边缘上的照明梯度和高光可能会使颜色值发生显著变化，从而以不理想的方式落入或超出分割范围。HSV 颜色空间通常可以提供更清晰的分割结果，因为亮度（值）和饱和度的阈值可以与颜色信息（色调）分开设置。

在 HSV 颜色空间中检查红色调的颜色范围时，`F_VN_CheckColorRange` 能够正确处理色调通道的所谓“环绕问题”，因为红色范围在 0° 周围扩展约 +-20°。如果为色调定义的下限值大于上限值（例如 251 至 6），该函数会自动将其识别为跨越零点的圆形区域。这样就无需手动拆分为两个独立的颜色范围和函数调用，并且能够在一个步骤中检测整个颜色范围。



结果图像 `ipImageResDisp` 以彩色轮廓以及与检测颜色相对应的标签来显示检测到的对象。



类似例程

- 与 RGB 参考色的颜色相似度 [▶ 2781]

8.1.10 测量

8.1.10.1 定位圆弧

在本示例中，F_VN_LocateCircularArc [▶ 1464] 函数可用于定位和测量圆弧。该函数返回一个 TcVnCircularArc 结构，其中包含中心、半径以及边界角度作为结果。要测量完整的圆，请改用 F_VN_LocateEllipse [▶ 1485] 函数。

搜索的起始位置应大致对应于圆弧的预期圆心，即相应的示例图像中的 400，300。

为了证明此处大致位置即可满足要求，示例中将相应参数 aCenter 设置为 420，310。该示例表明，可以补偿要检测对象在图像中的微小位置变化。

由于示例图像中的实际半径约为 200 像素，且起始点故意设置得不精确，因此搜索半径 fRadius 被设为 270。在具体示例中，大约 230 像素已足够，但在实践中，预留额外的安全缓冲往往是有益的，例如，当对象的位置偏离超出预期时。

对于示例图像，必须选择搜索方向，以确保在所有图像中都能正确找到对象。一个合适的方向是左下方向，大约 2,356 弧度（135°）。为了说明该方向不必非常精确，示例中将 fDirection 设为 2.1 弧度（120.3°）。

此外，应将从起点开始的明暗过渡指定为搜索标准。还必须指定最小对比度，当超过该值时，函数会将相邻像素之间的强度差检测为边缘。

如果这些参数不足以获得良好结果，F_VN_LocateCircularArcExp [▶ 1465] 专家函数还提供了许多额外的参数，本示例未考虑这些参数。

变量

```
hr                : HRESULT;

ipImageIn         : ITcVnImage;
ipImageInDisp     : ITcVnDisplayableImage;
ipImageRes        : ITcVnImage;
ipImageResDisp    : ITcVnDisplayableImage;

// result
stArc       : TcVnCircularArc;

// input parameters (to specify where to start searching for the circular arc)
aCenter             : TcVnPoint2_REAL := [420, 310];
fRadius             : REAL := 270;
fDirection          : LREAL := 2.1;

// drawing
aColor              : TcVnVector4_LREAL := [0, 175, 0];
sText               : STRING(255);
```

代码

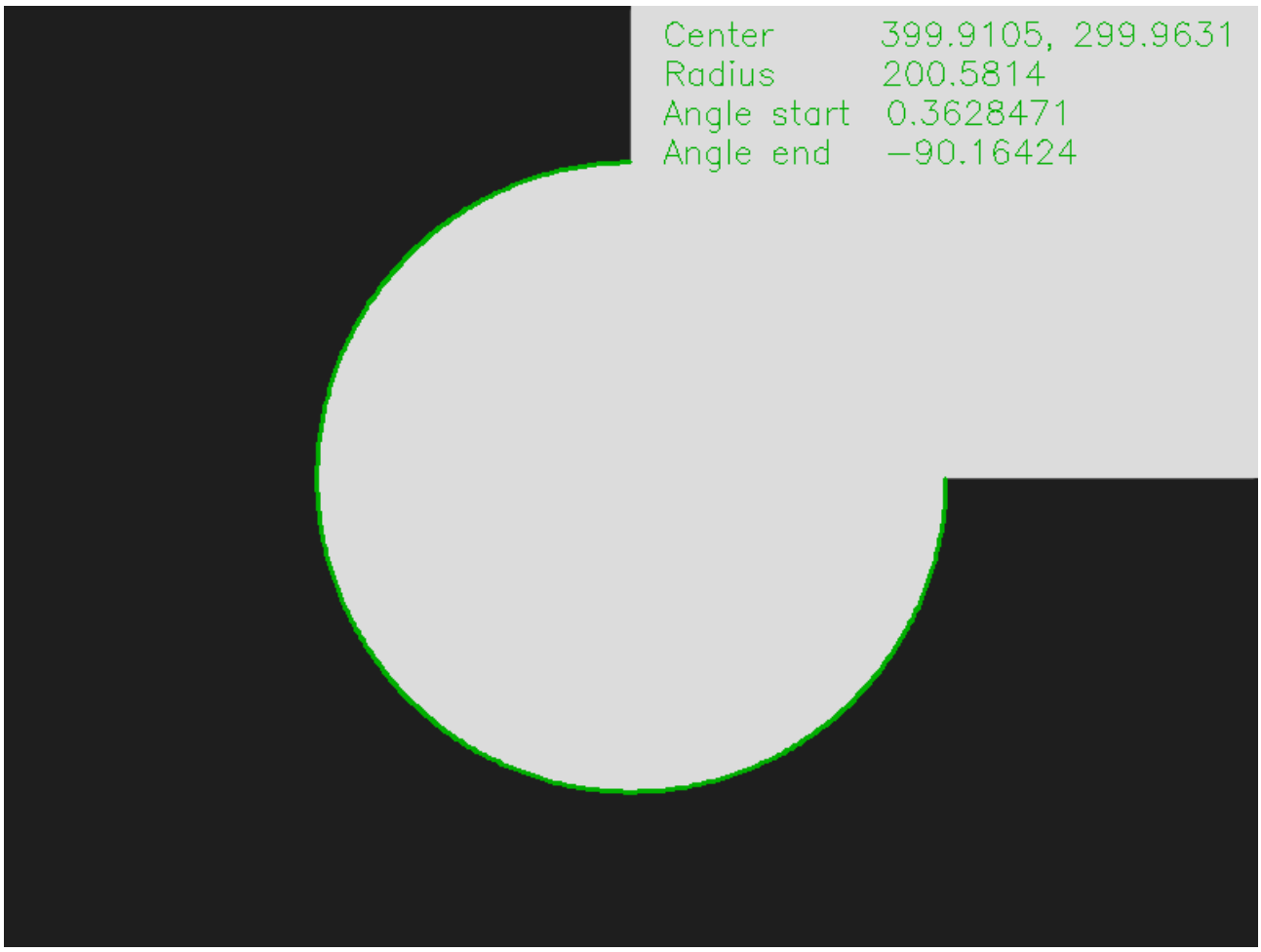
```
hr := F_VN_LocateCircularArc(
                        ipSrcImage          := ipImageIn,
                        stCircularArc       := stArc,
                        aCenter             := aCenter,
                        fSearchRadius       := fRadius,
                        fArcDirectionRad    := fDirection,
                        eEdgeDirection      := TCVN_ED_LIGHT_TO_DARK,
                        fMinStrength        := 100,
                        hrPrev              := hr);

// Draw result for visualization
hr := F_VN_ConvertColorSpace(ipImageIn, ipImageRes, TCVN_CST_GRAY_TO_RGB, hr);
hr := F_VN_DrawCircularArc(stArc, ipImageRes, aColor, 2, hr);
sText := CONCAT(CONCAT(CONCAT('Center ', REAL_TO_STRING(stArc.aCenter[0])), ', '),
REAL_TO_STRING(stArc.aCenter[1]));
hr := F_VN_PutText(sText, ipImageRes, 420, 25, TCVN_FT_HERSHEY_SIMPLEX, 0.7, aColor, hr);
sText := CONCAT('Radius ', REAL_TO_STRING(stArc.fRadius));
hr := F_VN_PutText(sText, ipImageRes, 420, 50, TCVN_FT_HERSHEY_SIMPLEX, 0.7, aColor, hr);
sText := CONCAT('Angle start ', REAL_TO_STRING(stArc.fStartAngle * 180 / LREAL_TO_REAL(PI)));
hr := F_VN_PutText(sText, ipImageRes, 420, 75, TCVN_FT_HERSHEY_SIMPLEX, 0.7, aColor, hr);
sText := CONCAT('Angle end ', REAL_TO_STRING(stArc.fEndAngle * 180 / LREAL_TO_REAL(PI)));
hr := F_VN_PutText(sText, ipImageRes, 420, 100, TCVN_FT_HERSHEY_SIMPLEX, 0.7, aColor, hr);

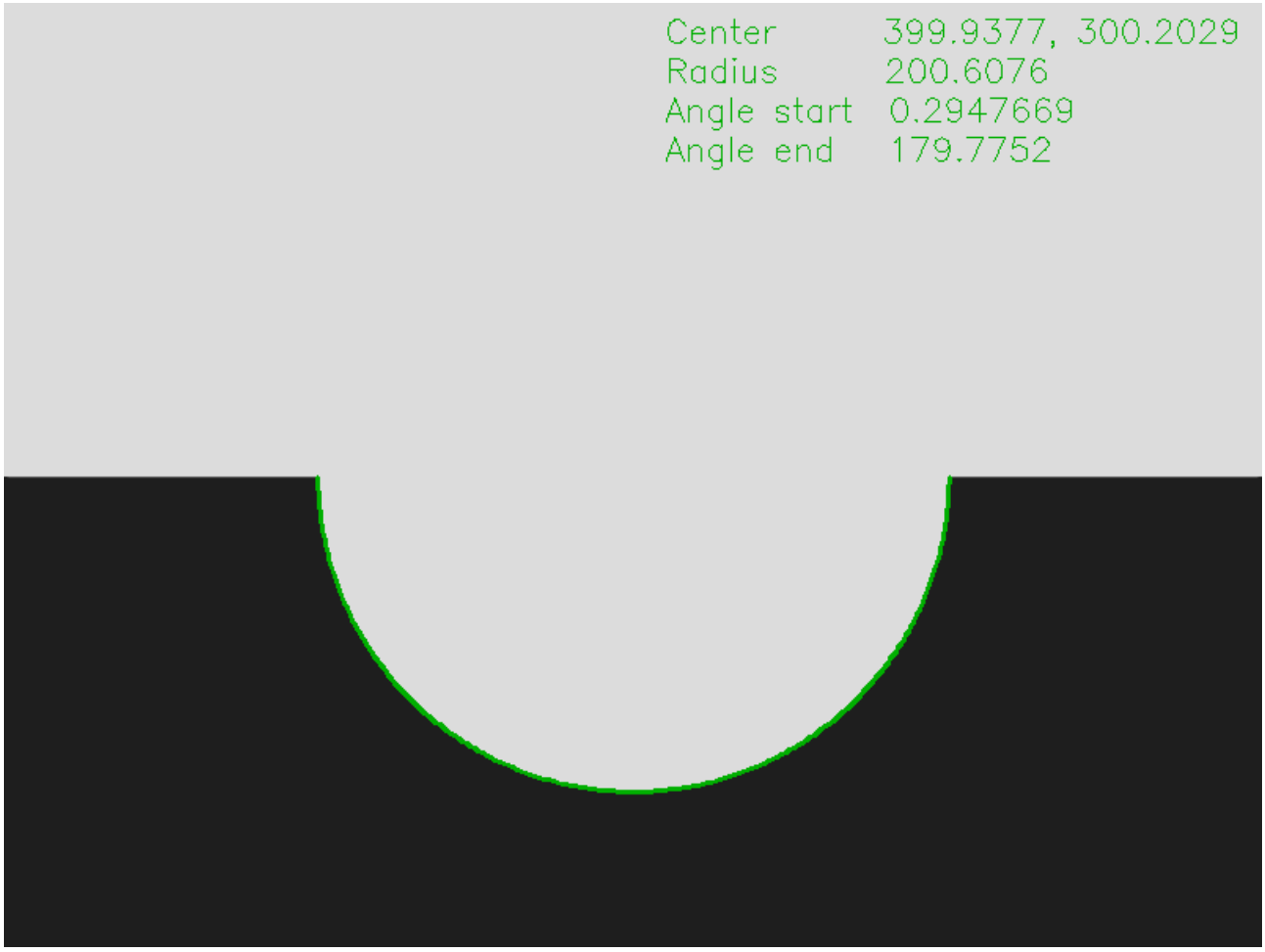
// Display source and result image
hr := F_VN_TransformIntoDisplayableImage(ipImageIn, ipImageInDisp, S_OK);
hr := F_VN_TransformIntoDisplayableImage(ipImageRes, ipImageResDisp, S_OK);
```

结果

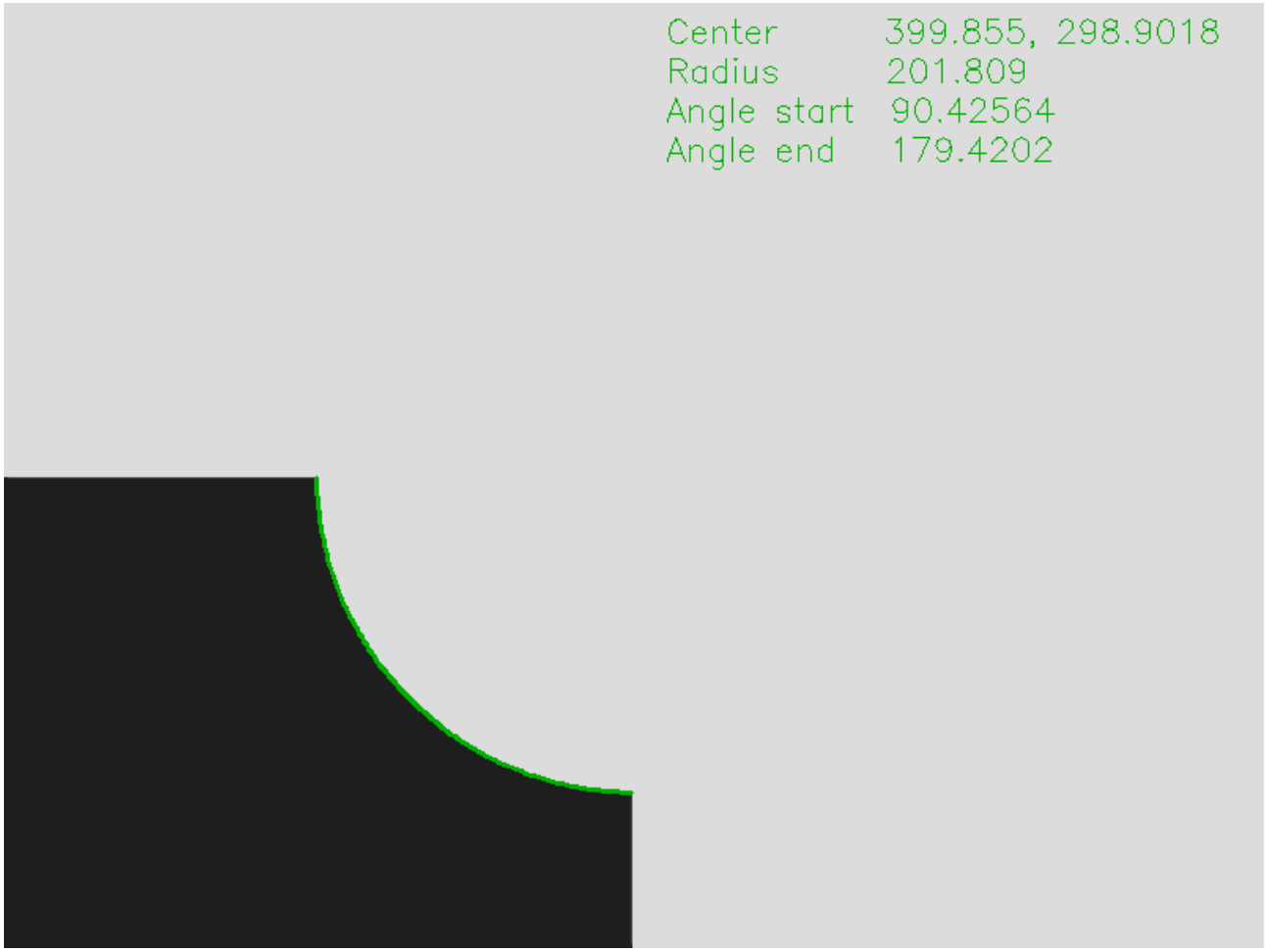
为实现可视化，使用 F_VN_DrawCircularArc [▶ 1008] 函数将找到的圆弧绘制到结果图像中。底层的数值也会以文本形式显示。



附图 31: LocateCircularArc_3/4圆_结果



附图 32: LocateCircularArc_半圆_结果



附图 33: LocateCircularArc_四分之一圆_结果

类似例程

- 定位边缘 [▶ 2797]
- 定位椭圆 [▶ 2802]
- 测量边缘之间的角度 [▶ 2807]

8.1.10.2 定位边缘

在此例程中，

- 函数 F_VN_LocateEdgeExp [▶ 1471] 用于在定义的搜索窗口中定位一条边，
- 并用 Watchdog（看门狗）监控执行时间，必要时对其进行限制。

说明

搜索窗口可用于使用函数 F_VN_LocateEdge [▶ 1470] 定位边缘。为简单起见，某些参数是固定的。或者，此处使用的函数 F_VN_LocateEdgeExp [▶ 1471] 可以完全访问所有参数。本例程旨在说明各个参数。我们建议在标准配置之外尝试更改配置，并考虑对边缘结果和处理时间的影响。在 Results（结果）下可找到部分配置更改，包括说明。

变量

```
hr              : HRESULT;
hrFunc          : HRESULT;

ipImageIn       : ITcVnImage;
ipImageInDisp   : ITcVnDisplayableImage;
ipImageRes      : ITcVnImage;
ipImageResDisp  : ITcVnDisplayableImage;
```

```
// result
ipEdgePoints    : ITcVnContainer;

// parameters
aStartPoint     : TcVnPoint2_REAL := [850, 400];
aEndPoint       : TcVnPoint2_REAL := [550, 400];
eDirection      : ETcVnEdgeDirection := TCVN_ED_DARK_TO_LIGHT;
fMinStrength    : REAL := 50;
nSearchLines    : UDINT := 31;
fSearchLineDist : REAL := 1;
nMaxThickness   : UDINT := 7;
nSubpixIter     : UDINT := 10;
eAlgorithm      : ETcVnEdgeDetectionAlgorithm := TCVN_EDA_INTERPOLATION;
fAvgStrength    : REAL;

// Watchdog
hrWD            : HRESULT;
tStop           : DINT := 15000;
tRest           : DINT;
nFraction       : UDINT;

// drawing
aLine           : TcVnVector4_LREAL;
aColorGreen     : TcVnVector4_LREAL := [0, 175, 0];
aColorBlue      : TcVnVector4_LREAL := [0, 0, 255];
aColorRed       : TcVnVector4_LREAL := [255, 0, 0];
sText           : STRING(255);
```

代码

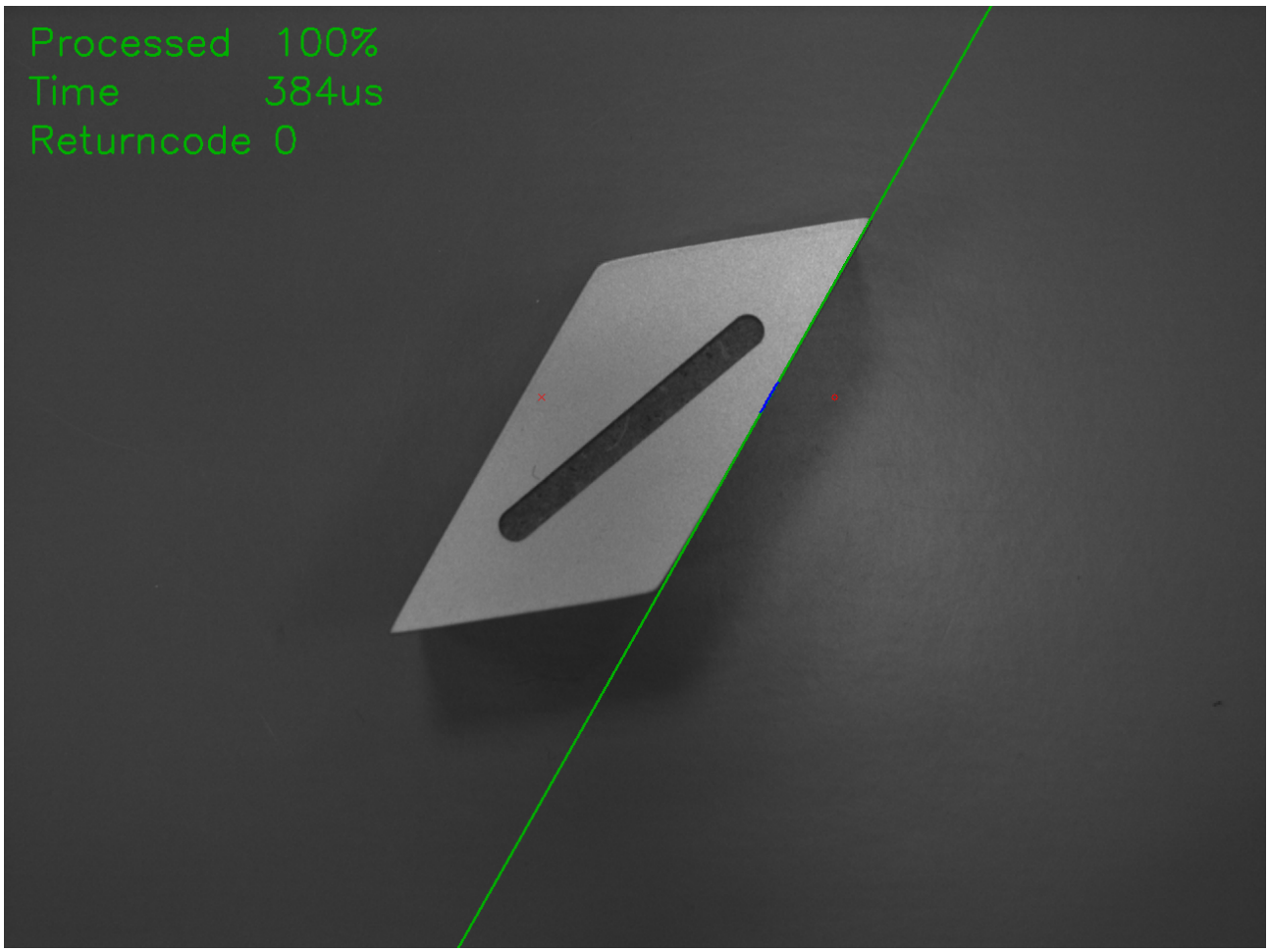
```
hrWD := F_VN_StartRelWatchdog(tStop, hr);
   hrFunc := F_VN_LocateEdgeExp(
                    ipSrcImage              :=   ipImageIn,
                    ipEdgePoints            :=   ipEdgePoints,
                    aStartPoint             :=   aStartPoint,
                    aEndPoint               :=   aEndPoint,
                    eEdgeDirection          :=   eDirection,
                    fMinStrength            :=   fMinStrength,
                    nSearchLines            :=   nSearchLines,
                    fSearchLineDist     :=  fSearchLineDist,
                    nMaxThickness           :=   nMaxThickness,
                    nSubpixelsIterations    :=   nSubpixIter,
                    fApproxPrecision        :=   0.0001,
                    eAlgorithm              :=   eAlgorithm,
                    hrPrev                  :=   hr,
                    fAvgStrength            =>   fAvgStrength);
hrWD := F_VN_StopWatchdog(hrWD, nFractionProcessed=>nFraction, tRest=>tRest);

// Draw result for visualization
hr := F_VN_ConvertColorSpace(ipImageIn, ipImageRes, TCVN_CST_GRAY_TO_RGB, hr);
sText := CONCAT(CONCAT('Processed ', UDINT_TO_STRING(nFraction)),'%');
hr := F_VN_PutTextExp(sText, ipImageRes, 25, 50, TCVN_FT_HERSHEY_SIMPLEX, 1.3, aColorGreen, 2,
TCVN_LT_8_CONNECTED, FALSE, hr);
sText := CONCAT(CONCAT('Time ', DINT_TO_STRING(tStop - tRest)), 'us');
hr := F_VN_PutTextExp(sText, ipImageRes, 25, 100, TCVN_FT_HERSHEY_SIMPLEX, 1.3, aColorGreen, 2,
TCVN_LT_8_CONNECTED, FALSE,hr);
sText := CONCAT('Returncode ', DINT_TO_STRING(hrFunc));
hr := F_VN_PutTextExp(sText, ipImageRes, 25, 150, TCVN_FT_HERSHEY_SIMPLEX, 1.3, aColorGreen, 2,
TCVN_LT_8_CONNECTED, FALSE,hr);
hr := F_VN_DrawPoint(REAL_TO_UDINT(aStartPoint[0]), REAL_TO_UDINT(aStartPoint[1]), ipImageRes,
TCVN_DS_CIRCLE, aColorRed, hr);
hr := F_VN_DrawPoint(REAL_TO_UDINT(aEndPoint[0]), REAL_TO_UDINT(aEndPoint[1]), ipImageRes,
TCVN_DS_X, aColorRed, hr);
hr := F_VN_FitLine(ipEdgePoints, aLine, hr);
hr := F_VN_DrawLine_TcVnVector4_LREAL(aLine, ipImageRes, aColorGreen, 2, hr);
hr := F_VN_DrawPointsExp(ipEdgePoints, ipImageRes, TCVN_DS_PLUS, aColorBlue, 1, 1,
TCVN_LT_8_CONNECTED, hr);

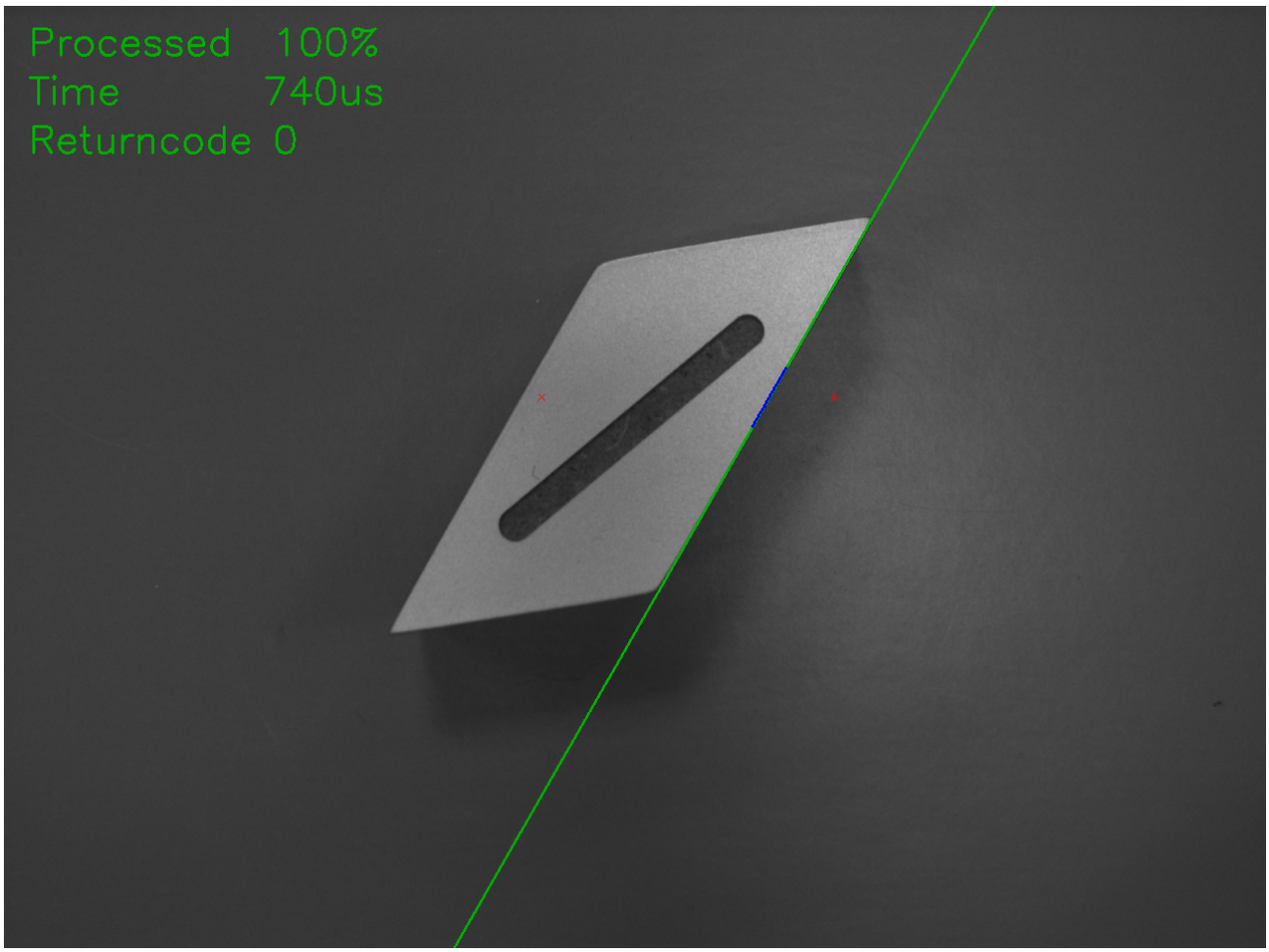
// Display source and result image
hr := F_VN_TransformIntoDisplayableImage(ipImageIn, ipImageInDisp, S_OK);
hr := F_VN_TransformIntoDisplayableImage(ipImageRes, ipImageResDisp, S_OK);
```

结果

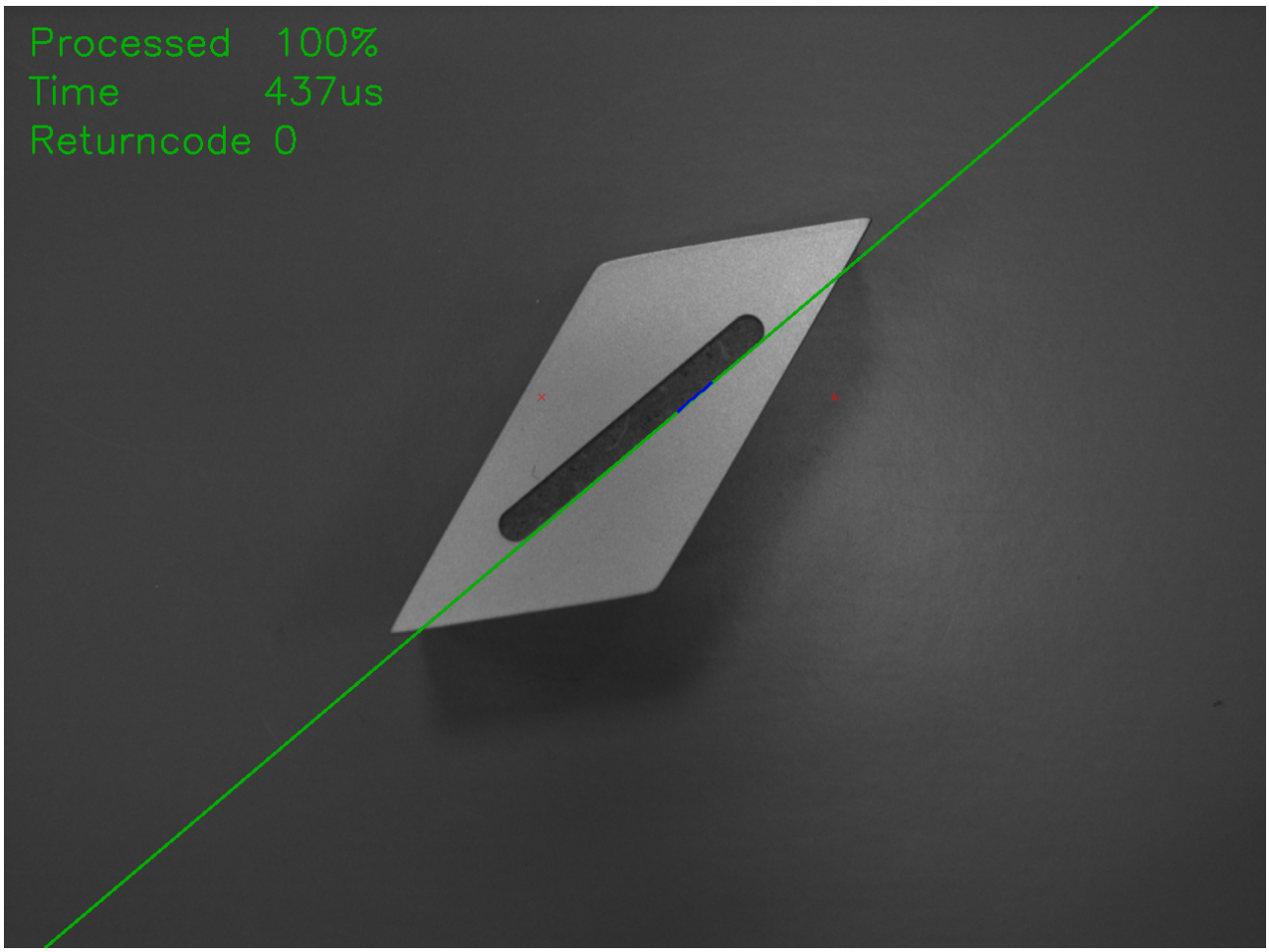
为便于可视化，找到的边缘点以蓝色显示，输出图像中的回归线以绿色绘制。此外，还绘制了起点（红色圆圈）和终点（红色 X）。左上角显示函数的处理百分比、所需执行时间和返回值。
对于本示例中使用的参数，结果如下：



如果使用参数 `nSearchLines` 将搜索行数从 31 行增加到 61 行，那么所需计算时间也会增加一倍。反过来，回归线变得更加精确：



如果想找到组件的铣削边缘而不是外边缘，请将参数 `eDirection` 更改为 `TCVN_ED_LIGHT_TO_DARK`。这样就忽略了从暗到亮的外边缘过渡，而是找到从亮到暗的下边缘：



缩短执行时间

值得注意的是，尽管参数完全相同，但执行时间却比定位外缘时要长。这是因为找到的边缘距离起点更远。

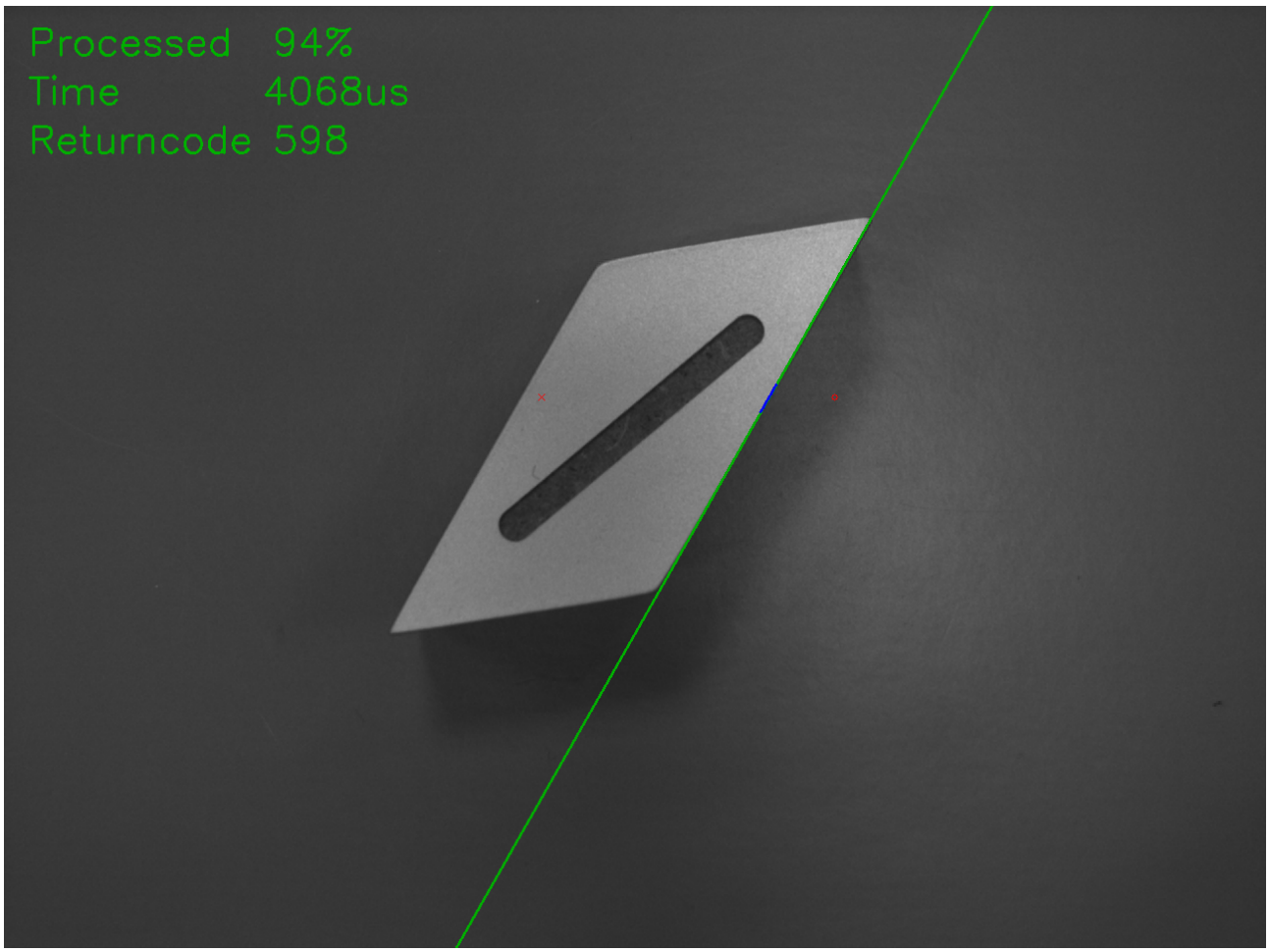
反之，这意味着固定搜索窗口的执行时间会因组件在图像中的位置而略有不同，这一点在三个例程图像中也可以看到。

这两种近似算法可能会出现更大的波动，因为它们需要不同的迭代来利用周围像素的强度近似模型参数。

一般来说，如果搜索窗口的搜索线尽可能正交地击中边缘，所需的迭代次数就会减少。如果图像中物体的位置和方向波动很小或没有波动，则可以相应地调整搜索窗口，以达到缩短执行时间的目的。此外，还可以通过减少最大迭代次数来缩短所需的最长时间，但这可能会导致结果不够精确。

使用 Watchdog（看门狗）

此外，还可以通过外部 Watchdog（看门狗）对时间进行限值，必要时看门狗会终止函数的执行，并返回现有的部分结果。例如，如果将算法改为 `TCVN_EDA_APPROX_ERF` 并将最大迭代次数设为 60，则图像 LocateEdge2.bmp（外边缘与搜索线正交）的执行时间几乎正好为 4 ms（取决于所安装的 CPU），而其他两幅图像的执行时间分别为 4.3 ms 和 4.4 ms。如果将参数 `tStop` 设为 4000，则函数会在 4 ms 后尽快停止执行，并返回到那时为止可用的部分结果。换句话说，图像 LocateEdge2.bmp 上的函数继续进行 100% 处理，而另外两个图像（本例中为 LocateEdge3.bmp）上的函数则提前终止：



由于只能在算法中的某些点终止，并返回部分结果（之前在搜索行上找到的边缘点），因此该函数不会在精确到 4000 µs 后终止，而是需要更长的时间，在选择中止时间时必须考虑到这一点。所需的最长额外时间一般取决于算法和参数设置。

尽管终止了，但部分结果足以实现预期结果，因为只缺少一两条搜索行的结果。

类似例程

- 定位圆弧 [▶ 2793]
- 定位椭圆 [▶ 2802]
- 测量边缘之间的角度 [▶ 2807]

8.1.10.3 定位椭圆

在此例程中，

- 函数 F_VN_LocateEllipseExp [▶ 1486] 用于在定义的搜索区域内定位圆形对象
- 并用 Watchdog（看门狗）监控执行时间，必要时对其进行限制。

说明

如需基本了解测量函数及其参数，请先按照 Locate Edge（定位边缘）例程 [▶ 2797]进行操作。函数 F_VN_LocateEllipseExp [▶ 1486] 与 F_VN_LocateEdgeExp [▶ 1470] 的不同之处在于，它使用的是圆形搜索窗口，而不是矩形窗口。在此例程中，由于圆形物体的位置会发生轻微变化，因此对较大的、不准确的搜索窗口进行参数设置。可以在所有图像中使用相同的搜索窗口确定圆圈。

参数 `bInvertSearchDirection` 在本例程中设置为 `TRUE`，用于定义搜索方向相反，即从外部向 `aCenter` 搜索。这一点很有必要，因为例程图像中也包含系统正在寻找的椭圆形内的边缘。

搜索方向设置为 TCVN_ED_LIGHT_TO_DARK，因为在此例程中，物体的颜色比背景深。

参数 nMaxThickness 定义了 fMinStrength 必须达到的像素数，被设置为“7”。

变量

```
hr               : HRESULT;
hrFunc           : HRESULT;

ipImageIn        : ITcVnImage;
ipImageInDisp    : ITcVnDisplayableImage;
ipImageRes       : ITcVnImage;
ipImageResDisp   : ITcVnDisplayableImage;

// result
stEllipse        : TcVnRotatedRectangle;
ipContourPoints  : ITcVnContainer;

// input parameters
aCenter          : TcVnPoint2_REAL := [650, 400];
fRadius          : REAL := 300;
fMinRadius       : REAL := 100;
fMinStrength     : REAL := 30;
nSubPixIter      : UDINT := 10;
nSearchLines     : UDINT := 40;
eAlgorithm       : ETcVnEdgeDetectionAlgorithm := TCVN_EDA_INTERPOLATION;

// Watchdog
hrWD             : HRESULT;
tStop            : DINT := 15000;
tRest            : DINT;
nFraction        : UDINT;

// drawing
aColorRed        : TcVnVector4_LREAL := [200, 0, 0];
aColorGreen      : TcVnVector4_LREAL := [0, 175, 0];
sText            : STRING(255);
```

代码

```
hrWD := F_VN_StartRelWatchdog(tStop, hr);
    hrFunc := F_VN_LocateEllipseExp(
                ipSrcImage                  :=    ipImageIn,
                stEllipse                   :=    stEllipse,
                aCenter                     :=    aCenter,
                fSearchRadius               :=    fRadius,
                eEdgeDirection              :=    TCVN_ED_LIGHT_TO_DARK,
                fMinStrength                :=    fMinStrength,
                nMaxThickness               :=    7,
                bInvertSearchDirection      :=    TRUE,
                fMinSearchRadius            :=    fMinRadius,
                nSubpixelsIterations        :=    nSubpixIter,
                nSearchLines                :=    nSearchLines,
                fApproxPrecision            :=    0.0001,
                eAlgorithm                  :=    eAlgorithm,
                ipContourPoints             :=    ipContourPoints,
                hrPrev                      :=    hr);
hrWD := F_VN_StopWatchdog(hrWD, nFractionProcessed=>nFraction, tRest=>tRest);

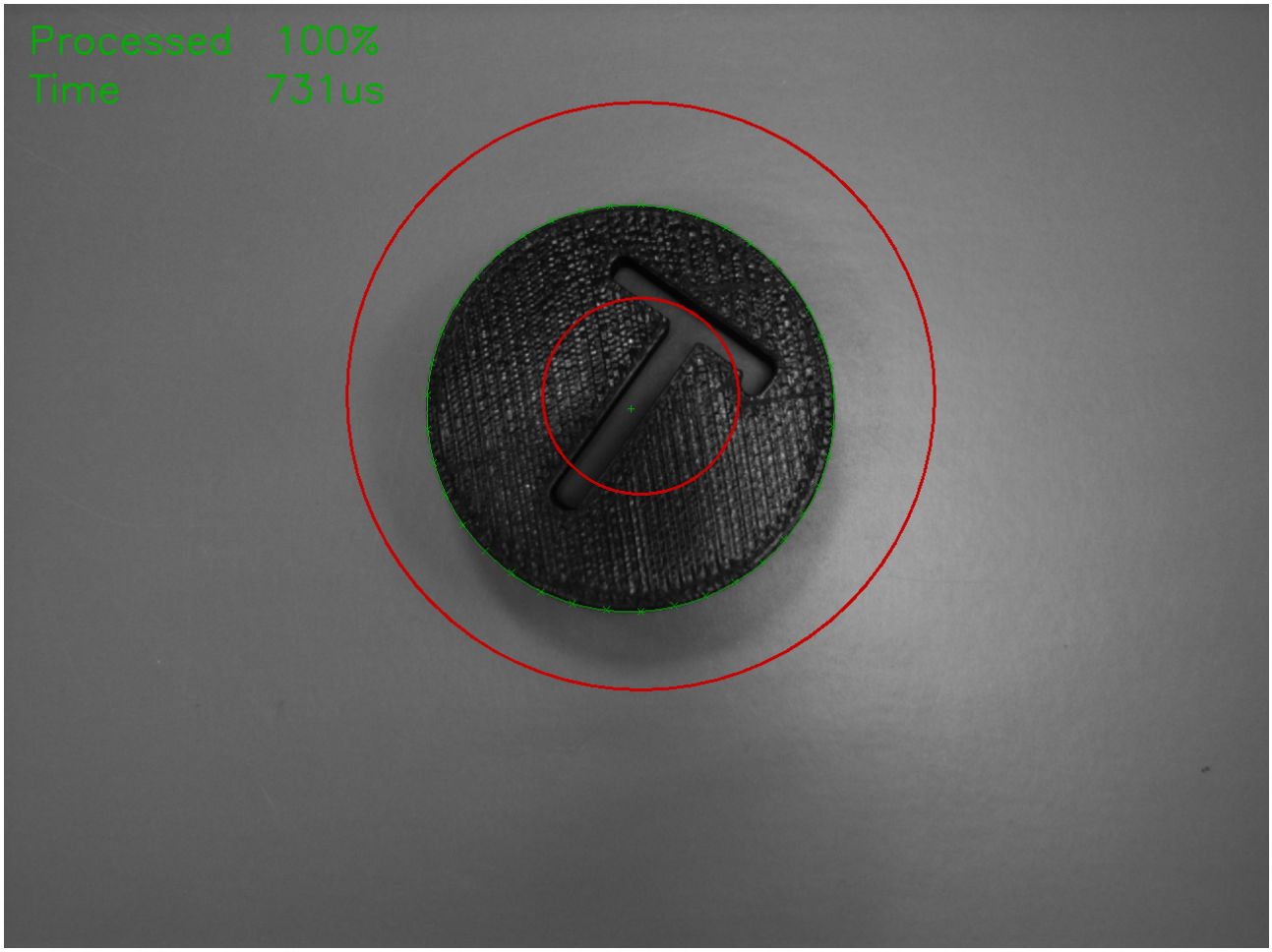
// Draw result for visualization
hr := F_VN_ConvertColorSpace(ipImageIn, ipImageRes, TCVN_CST_GRAY_TO_RGB, hr);
sText := CONCAT(CONCAT('Processed ', UDINT_TO_STRING(nFraction)),'%');
hr := F_VN_PutTextExp(sText, ipImageRes, 25, 50, TCVN_FT_HERSHEY_SIMPLEX, 1.3, aColorGreen, 2,
TCVN_LT_8_CONNECTED, FALSE, hr);
sText := CONCAT(CONCAT('Time ', DINT_TO_STRING(tStop - tRest)), 'us');
hr := F_VN_PutTextExp(sText, ipImageRes, 25, 100, TCVN_FT_HERSHEY_SIMPLEX, 1.3, aColorGreen, 2,
TCVN_LT_8_CONNECTED, FALSE,hr);
hr := F_VN_DrawCircle(REAL_TO_UDINT(aCenter[0]), REAL_TO_UDINT(aCenter[1]),
REAL_TO_UDINT(fMinRadius), ipImageRes, aColorRed, 2, hr);
hr := F_VN_DrawCircle(REAL_TO_UDINT(aCenter[0]), REAL_TO_UDINT(aCenter[1]), REAL_TO_UDINT(fRadius),
ipImageRes, aColorRed, 2, hr);
hr := F_VN_DrawPoints(ipContourPoints, ipImageRes, TCVN_DS_X, aColorGreen, hr);
hr := F_VN_DrawPoint(REAL_TO_UDINT(stEllipse.aCenter[0]), REAL_TO_UDINT(stEllipse.aCenter[1]),
ipImageRes, TCVN_DS_PLUS, aColorGreen, hr);
hr := F_VN_DrawEllipse(stEllipse, ipImageRes, aColorGreen, 1, hr);

// Display source and result image
hr := F_VN_TransformIntoDisplayableImage(ipImageIn, ipImageInDisp, S_OK);
hr := F_VN_TransformIntoDisplayableImage(ipImageRes, ipImageResDisp, S_OK);
```

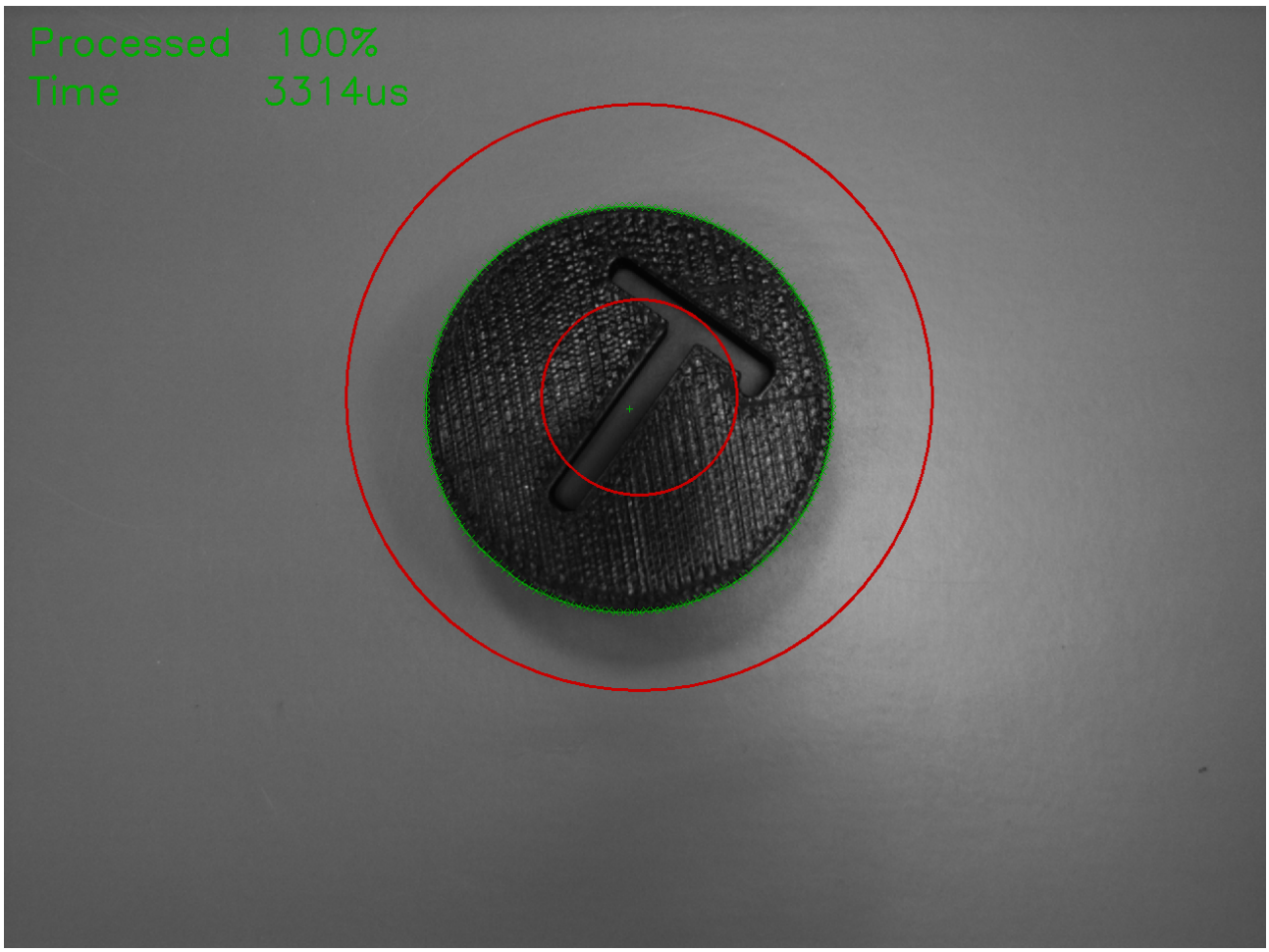
结果

该函数返回 `TcVnRotatedRectangle` 结构 `stEllipse`，该结构通过其中心 `aCenter`、高度和宽度 `stSize.fHeight` / `stSize.fWidth` 以及角度 `fAngle` 定义找到的椭圆形。
作为选项，每个搜索行 `ipContourPoints` 还可以返回一个额外的边缘点，椭圆形就是从这个边缘点逼近的。

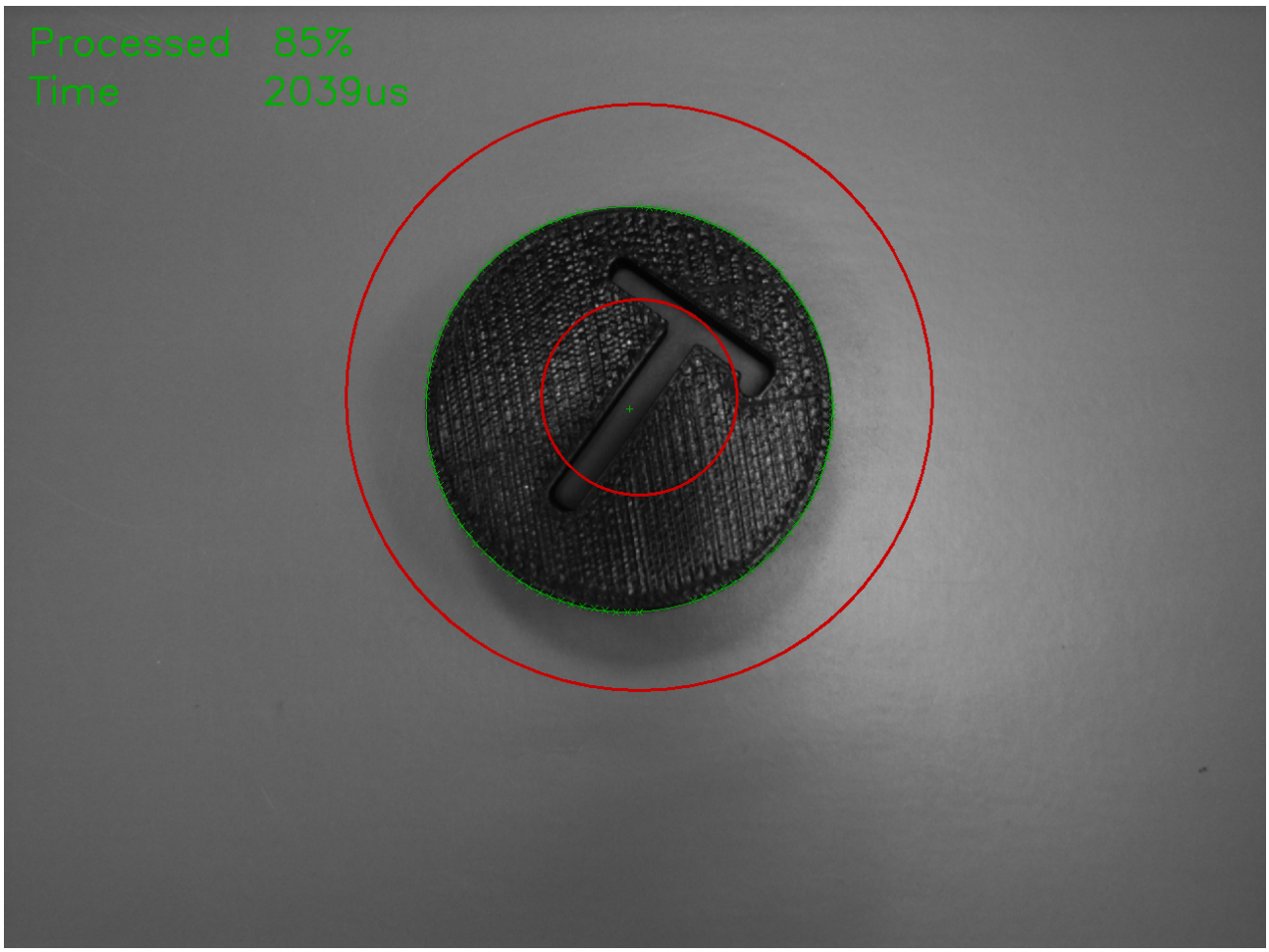
为了实现可视化，搜索区域首先以围绕指定中心点的两个红色圆圈的形式绘制到图像中。找到的椭圆形会被绘制成一条绿线，相应的中心点会被画成绿色的“+”。`ipContourPoints` 中的各个测量点被绘制为绿色 x。此外，左上角还显示处理百分比和所需计算时间。对于本示例中使用的参数，输入图像 LocateEllipse_T.bmp 的结果如下所示：



如果搜索行数从 40 行增加到 180 行，所需计算时间会按预期实现线性增长：

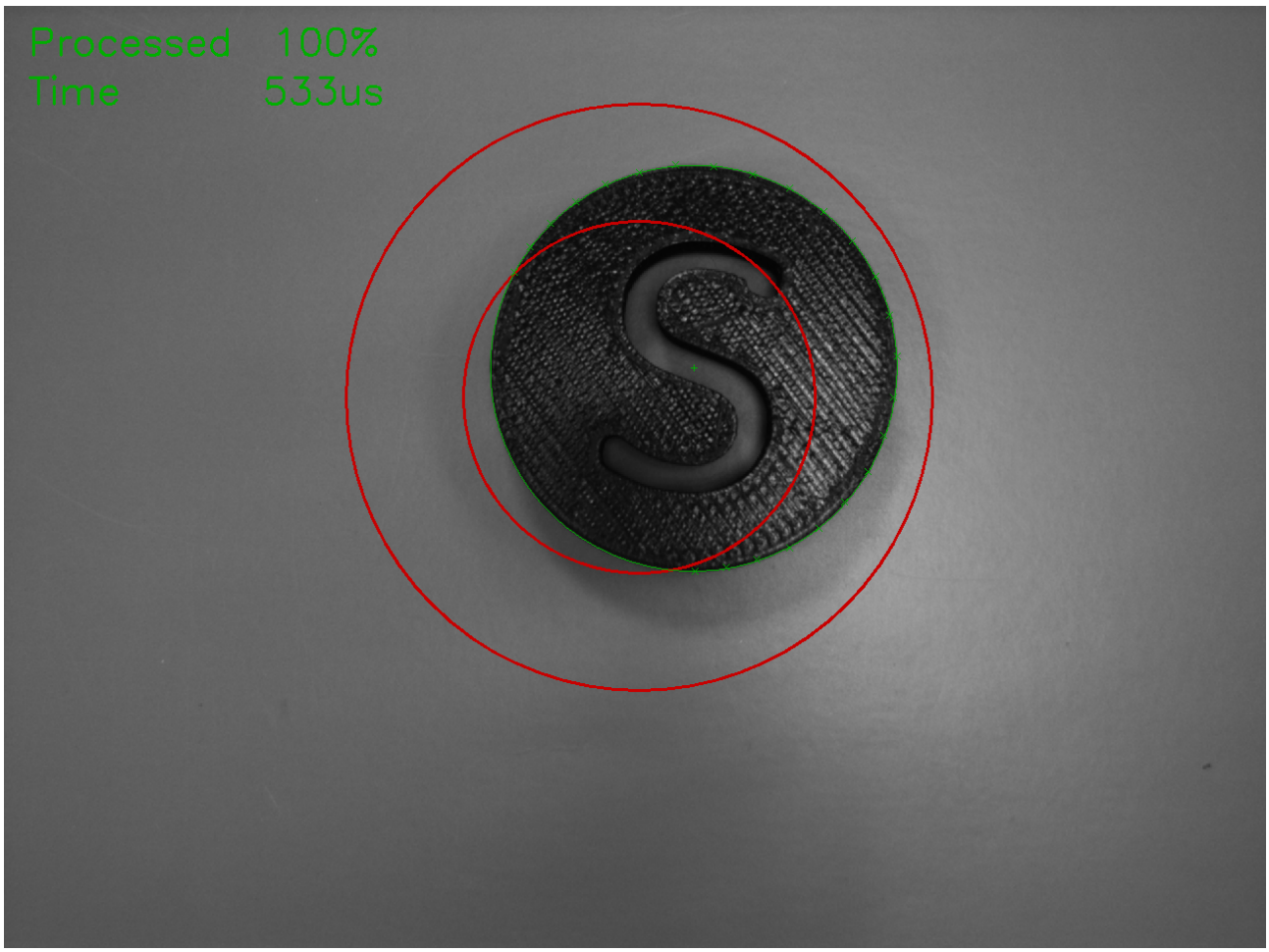


在实际操作中，计算时间会因输入图像的不同而有所差异。因此，在许多情况下，通过 Watchdog（看门狗）来限制计算时间是非常有用的。例如，如果搜索行数设置为 120，Watchdog（看门狗）超时 `tStop` 至 2000（取决于所使用的 CPU），则该函数会在处理时间的 85% 左右终止，并返回此时可用的部分结果：



由于只能在特定时间点中断算法并返回部分结果，因此函数不会在 2000 µs 后准确终止，而是需要更长的时间，在选择终止时间时必须考虑到这一点。所需的最长额外时间一般取决于算法和参数设置。
虽然由于终止（绿色 x 之间有四个相对较大的间隙），一些搜索线的结果缺失，但椭圆形仍可被正确近似，即部分结果可以像函数已被 100% 处理一样使用。

如果物体的位置部分超出了规定的搜索范围，例如因为之前的定位步骤不正确，也可以在一定限值内进行搜索。例如，如果将 `fMinRadius` 设置为 180，那么一些样本图像的部分外轮廓便会超出指定范围。尽管如此，还是有足够的搜索点来近似椭圆形。但在实际操作中，不应依赖这一点，而应将搜索区域设置得足够大：



类似例程

- 定位圆弧 [▶ 2793]
- 定位边缘 [▶ 2797]
- 测量边缘之间的角度 [▶ 2807]

8.1.10.4 测量边缘之间的角度

在本示例中，

- F_VN_MeasureAngleBetweenEdgesExp [▶ 1492] 函数用于测量对象的两个边缘之间的角度
- 并用 Watchdog（看门狗）监控执行时间，必要时对其进行限制。

说明

在定义搜索窗口后，将搜索方向设置为 `TCVN_ED_LIGHT_TO_DARK`，因为在本示例中，从内向外执行搜索，且对象的颜色比背景浅。

`nMaxThickness` 参数定义了 `fMinStrength` 必须达到的像素数，被设置为“7”。

变量

```
hr              : HRESULT;
hrFunc          : HRESULT;

ipImageIn       : ITcVnImage;
ipImageInDisp   : ITcVnDisplayableImage;
ipImageRes      : ITcVnImage;
ipImageResDisp  : ITcVnDisplayableImage;

// result
fAngle          : REAL;
ipEdgePoints1   : ITcVnContainer;
ipEdgePoints2   : ITcVnContainer;
```

```
// parameters
aInnerPoint     : TcVnPoint2_REAL := [635, 350];
aOuterPoint1    : TcVnPoint2_REAL := [530, 280];
aOuterPoint2    : TcVnPoint2_REAL := [790, 280];
fMinStrength    : REAL := 50;
nSearchLines    : UDINT := 41;
fSearchLineDist : REAL := 1;
nSubpixIter     : UDINT := 10;
eAlgorithm      : ETcVnEdgeDetectionAlgorithm := TCVN_EDA_INTERPOLATION;

// Watchdog
hrWD            : HRESULT;
tStop           : DINT := 15000;
tRest           : DINT;
nFraction       : UDINT;

// drawing
aLine1          : TcVnVector4_LREAL;
aLine2          : TcVnVector4_LREAL;
aColorRed       : TcVnVector4_LREAL := [255, 0, 0];
aColorGreen     : TcVnVector4_LREAL := [0, 175, 0];
aColorYellow    : TcVnVector4_LREAL := [255, 255, 0];
sText           : STRING(255);
```

代码

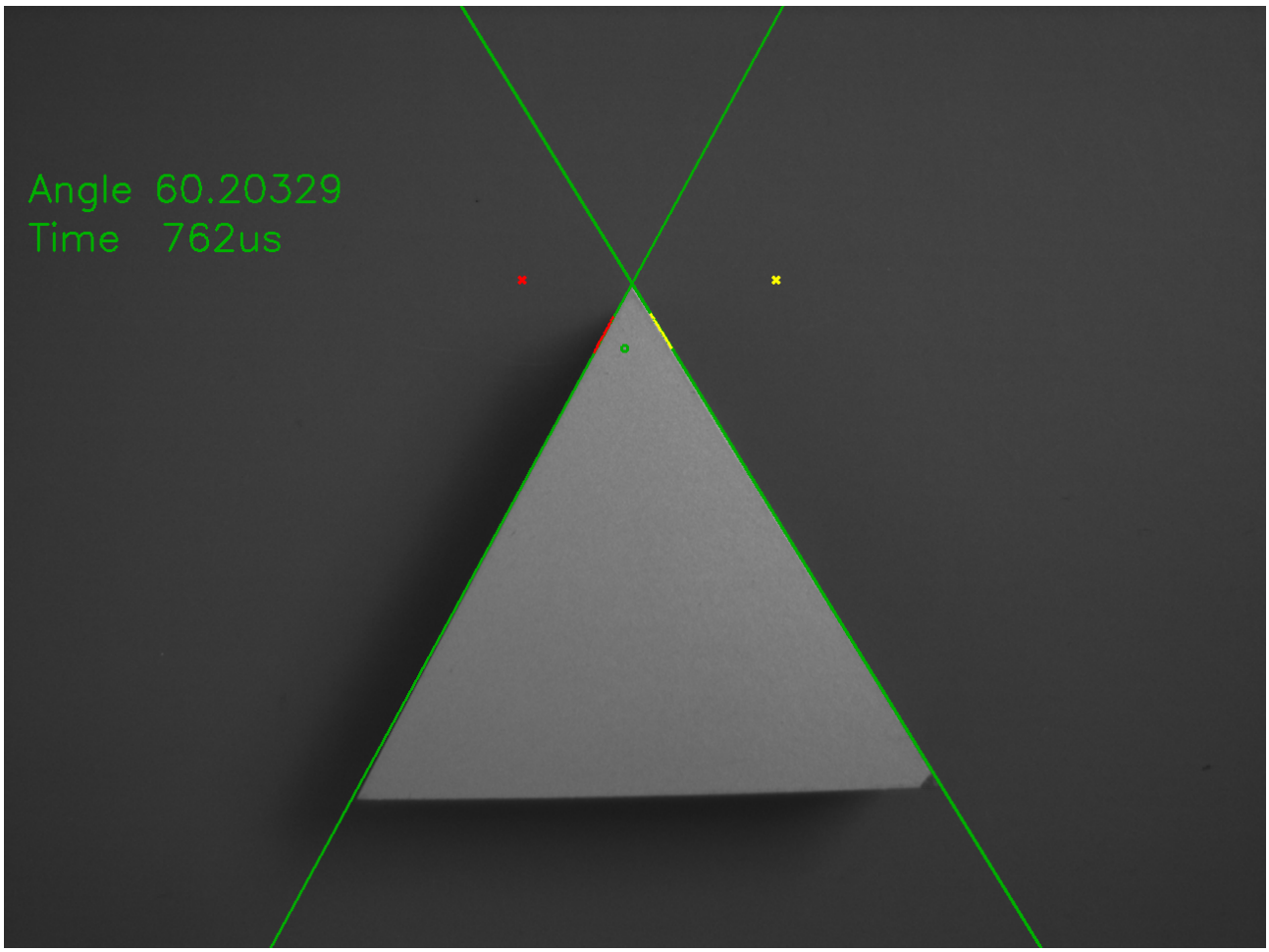
```
hrWD := F_VN_StartRelWatchdog(tStop, hr);
    hrFunc := F_VN_MeasureAngleBetweenEdgesExp(
                    ipSrcImage              := ipImageIn,
                    fAngle                  := fAngle,
                    aInnerPoint             := aInnerPoint,
                    aOuterPoint1            := aOuterPoint1,
                    aOuterPoint2            := aOuterPoint2,
                    eEdgeDirection          := TCVN_ED_LIGHT_TO_DARK,
                    fMinStrength            := fMinStrength,
        nSearchLines        := nSearchLines,
        fSearchLineDist     := fSearchLineDist,
                    nMaxThickness           := 7,
                    bInvertSearchDirection  := FALSE,
                    nSubpixelsIterations    := nSubpixIter,
                    bAngleInDegrees         := TRUE,
                    fApproxPrecision        := 0.0001,
                    eAlgorithm              := eAlgorithm,
                    ipEdgePoints1           := ipEdgePoints1,
                    ipEdgePoints2           := ipEdgePoints2,
                    hrPrev                  := hr);
hrWD := F_VN_StopWatchdog(hrWD, nFractionProcessed=>nFraction, tRest=>tRest);

// Draw result for visualization
hr := F_VN_ConvertColorSpace(ipImageIn, ipImageRes, TCVN_CST_GRAY_TO_RGB, hr);
sText := CONCAT('Angle ', REAL_TO_STRING(fAngle));
hr := F_VN_PutTextExp(sText, ipImageRes, 25, 200, TCVN_FT_HERSHEY_SIMPLEX, 1.3, aColorGreen, 2,
TCVN_LT_8_CONNECTED, FALSE,hr);
sText := CONCAT(CONCAT('Time ', DINT_TO_STRING(tStop - tRest)), 'us');
hr := F_VN_PutTextExp(sText, ipImageRes, 25, 250, TCVN_FT_HERSHEY_SIMPLEX, 1.3, aColorGreen, 2,
TCVN_LT_8_CONNECTED, FALSE,hr);
hr := F_VN_DrawPointExp(REAL_TO_UDINT(aInnerPoint[0]), REAL_TO_UDINT(aInnerPoint[1]), ipImageRes,
TCVN_DS_CIRCLE, aColorGreen, 3, 2, TCVN_LT_8_CONNECTED, hr);
hr := F_VN_DrawPointExp(REAL_TO_UDINT(aOuterPoint1[0]), REAL_TO_UDINT(aOuterPoint1[1]), ipImageRes,
TCVN_DS_X, aColorRed, 3, 2, TCVN_LT_8_CONNECTED, hr);
hr := F_VN_DrawPointExp(REAL_TO_UDINT(aOuterPoint2[0]), REAL_TO_UDINT(aOuterPoint2[1]), ipImageRes,
TCVN_DS_X, aColorYellow, 3, 2, TCVN_LT_8_CONNECTED, hr);
hr := F_VN_FitLine(ipEdgePoints1, aLine1, hr);
hr := F_VN_FitLine(ipEdgePoints2, aLine2, hr);
hr := F_VN_DrawLine_TcVnVector4_LREAL(aLine1, ipImageRes, aColorGreen, 2, hr);
hr := F_VN_DrawLine_TcVnVector4_LREAL(aLine2, ipImageRes, aColorGreen, 2, hr);
hr := F_VN_DrawPointsExp(ipEdgePoints1, ipImageRes, TCVN_DS_PLUS, aColorRed, 1, 1,
TCVN_LT_8_CONNECTED, hr);
hr := F_VN_DrawPointsExp(ipEdgePoints2, ipImageRes, TCVN_DS_PLUS, aColorYellow, 1, 1,
TCVN_LT_8_CONNECTED, hr);

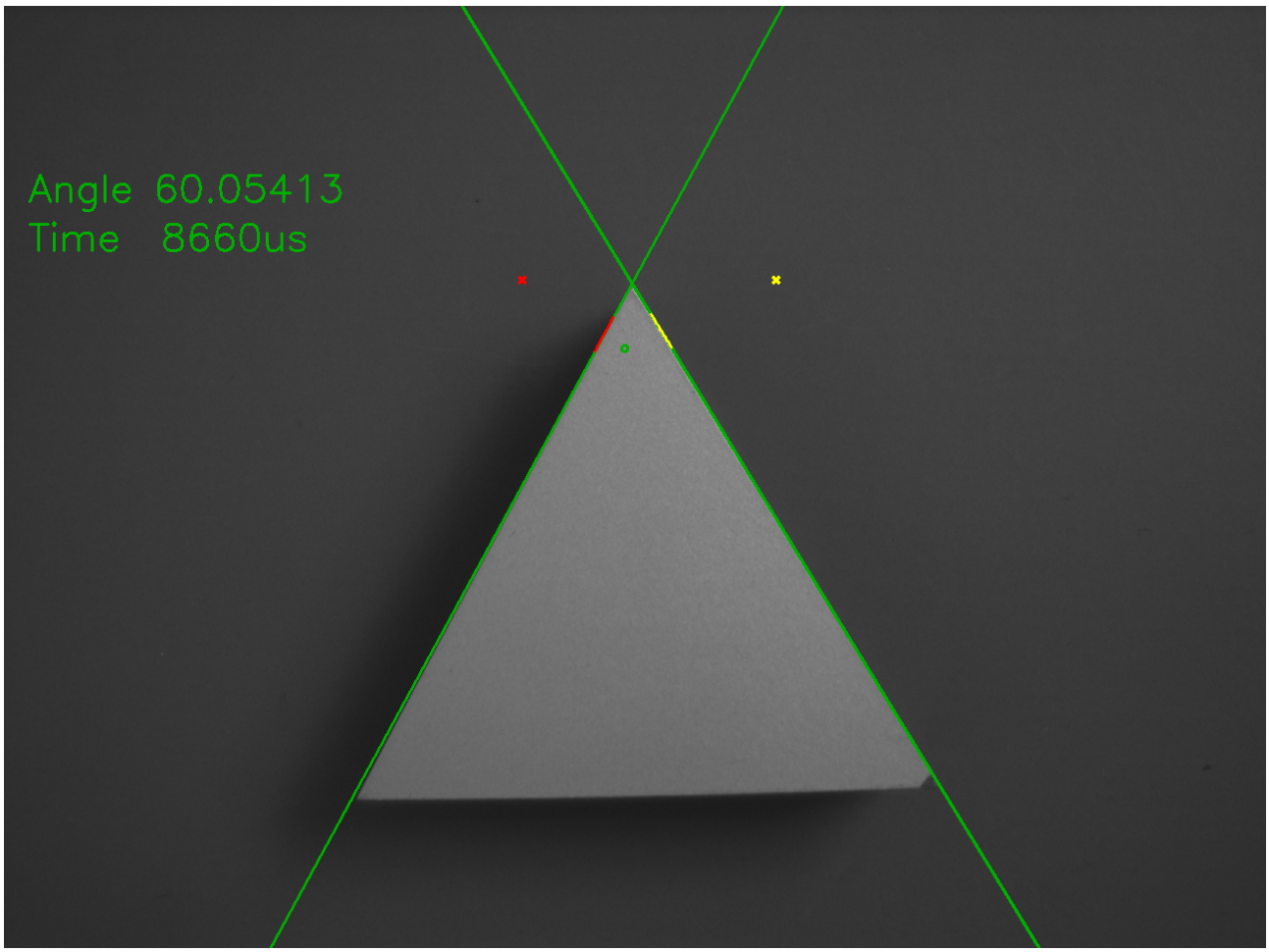
// Display source and result image
hr := F_VN_TransformIntoDisplayableImage(ipImageIn, ipImageInDisp, S_OK);
hr := F_VN_TransformIntoDisplayableImage(ipImageRes, ipImageResDisp, S_OK);
```

结果

出于可视化目的，`aInnerPoint` 首先显示为一个绿色圆圈，`aOuterPoint1` 显示为一个红色 x，`aOuterPoint2` 显示为一个黄色 x。所找到的相应 `ipEdgePoints1` 和 `ipEdgePoints2` 边缘点也以红色或黄色绘制出来。根据边缘点拟合出的线条以绿色绘制。计算得到的 `fAngle` 角度（以度为单位）以及所需的计算时间（以 µs 为单位）显示在图像的左侧。对于本示例中使用的参数，结果如下：



为了获得更准确的结果，可将 `eAlgorithm` 改为 `TCVN_EDA_APPROX_ERF`，并将 `nSubpixIter` 改为 `50`。不过，这也显著增加了所需的计算时间：



类似例程

- 定位圆弧 [▶ 2793]
- 定位边缘 [▶ 2797]
- 定位椭圆 [▶ 2802]

8.1.11 OCR

本示例展示了如何使用 OCR（光学字符识别）功能识别产品包装上的字符，例如日期和批号。该程序演示了 F_VN_OCR [▶ 1563] 标准函数以及 F_VN_OCRExp [▶ 1564] 扩展函数的应用，包括所需的预处理和后处理。

说明

这些函数可识别二值图像（黑底白字）中的字母数字字符。该分类基于具有不同功能和字符范围的 Machine Learning（机器学习）模型。

- ETcVnOcrModelType [▶ 204] 枚举可用于访问各种模型：
  - `TCVN_OMT_NUMBERS`：数字
  - `TCVN_OMT_NUMBERS_SC`：数字 + 特殊字符
  - `TCVN_OMT_UCLETTERS`：大写字母
  - `TCVN_OMT_NUMBERS_SC_UCLETTERS`：组合字符集
- `F_VN_OCRExp` 函数可提供扩展选项：
  - `sPattern` 表示所定义的格式和字符规格（例如“uudd”表示“AB12”）
  - `eOcrOptions` 附加的 ETcVnOcrOptions [▶ 204] 可影响功能和结果输出
  - `ipBoundingBoxes` 表示返回已识别字符的边界框

- ipConfidences 表示每个字符的置信度值

有关完整的说明、要求和限制以及更多信息，请参见 OCR [▶ 1560] 章节以及相应函数下的说明。

模型初始化

对于 OCR 识别，底层字符识别模型必须在首次使用前执行初始化。这是通过 FB_VN_InitializeFunction [▶ 1584] 功能块完成的，其中需要先在 eFunction 处指定要初始化的函数。后续调用 OCR 函数时所使用的模型通过 nOptions 进行定义。只需调用一次函数，就能同时初始化多个模型。

然后便可以使用 F_VN_CheckFunctionInitialization [▶ 1514] 来检查各自的模型是否已被正确初始化。如果模型要在运行时更改或不再需要，可以使用 F_VN_DeinitializeFunction [▶ 1522] 对模型进行去初始化，以便再次释放内存。

变量

```
bInitialized    : BOOL := FALSE;
fbInit          : FB_VN_InitializeFunction;
nReturnCode     : UDINT;
```

代码

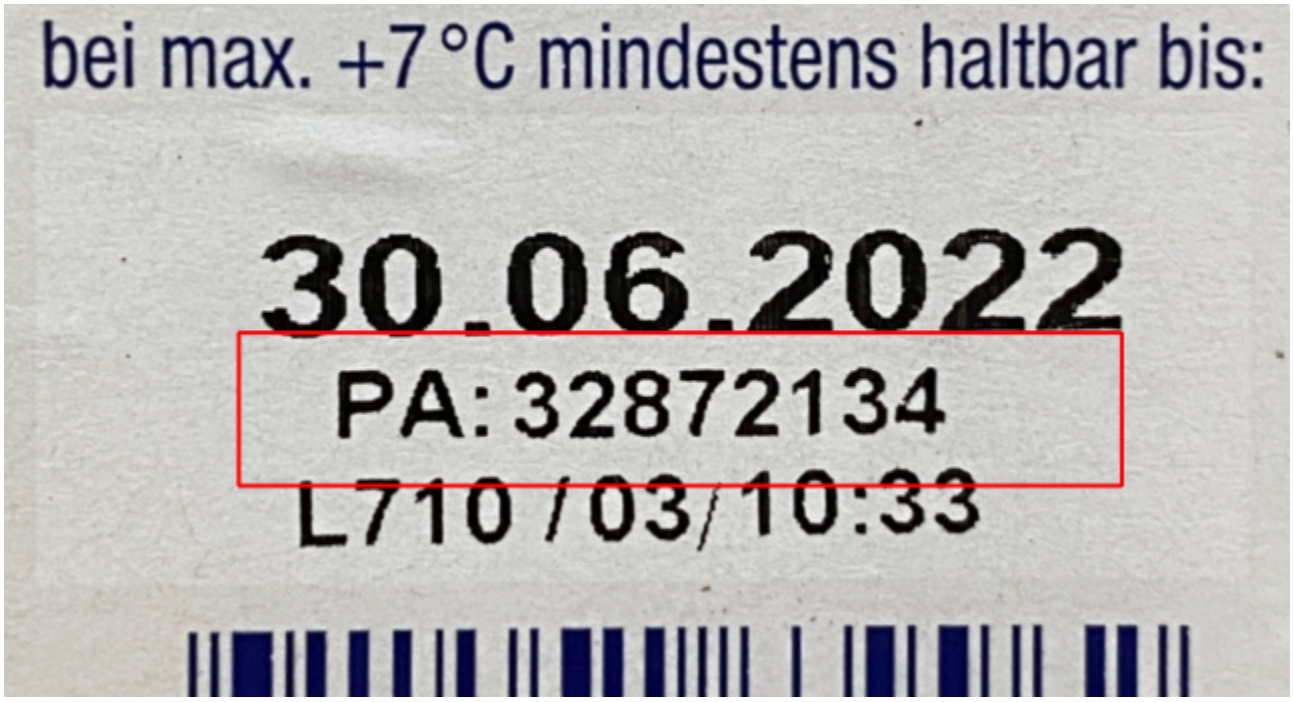
```
IF NOT bInitialized THEN
    fbInit(
        eFunction := TCVN_IF_OCR,
        nOptions  := ETcVnOcrModelType.TCVN_OMT_NUMBERS_SC OR ETcVnOcrModelType.TCVN_OMT_UCLETTERS,
        bStart    := TRUE);

    IF NOT fbInit.bBusy THEN
        fbInit(bStart := FALSE);
        IF NOT fbInit.bError THEN
            bInitialized := TRUE;
            nReturnCode := fbInit.nErrorId AND 16#FFF;
        ELSE
            nReturnCode := fbInit.nErrorId AND 16#FFF;
        END_IF
    END_IF
END_IF
```

预处理

一些预处理步骤对于使用 OCR 功能是必要或有用的。首先，必须定义 ROI 只包含一行中待识别字符和字符周围的无干扰区。此外，必须将该图像区域转换为单通道二值图像，该图像仅包含黑底上的白色待识别字符。在本示例中，使用了 F_VN_Threshold [▶ 1322]，根据背景和字符颜色的不同，该函数以正常模式或反向模式应用。

由于示例项目中使用了不同场景下的不同产品图像，因此诸如 stRoi 和 fThreshold 等参数在 F_GetROI 函数内为每个测试图像单独存储，并通过图像名称进行检索。在实际应用中，不同的参数值通常通过用户界面中的配方进行处理。以下图像用于进一步描述该示例应用，图中以红色显示指定的 ROI。

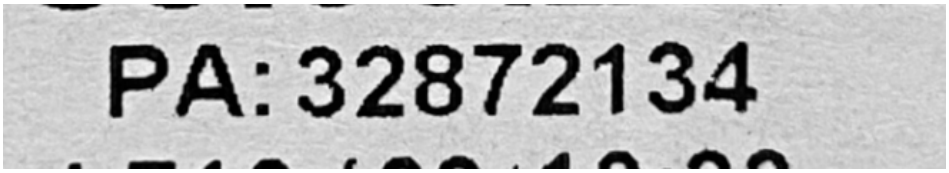


除了示例中所示的必要预处理之外，还可以使用噪声抑制的滤波操作以及开运算或闭运算等形态学运算，以去除微小干扰并对字符形状进行平滑处理或补全。对比度增强函数也经常使用。选择合适的滤波函数和参数在很大程度上取决于图像材料的具体特性，因此，其他函数（例如 ImageColorAndContrastProcessing、ImageFiltering 或 ImageSegmentation 中的函数）也有助于改善结果。

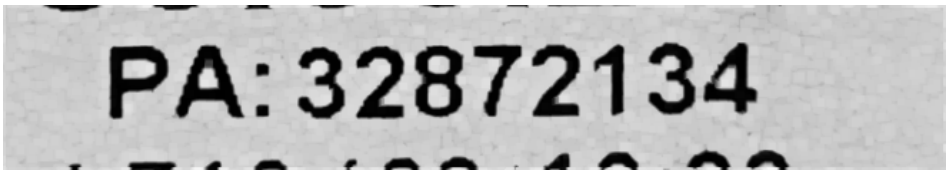
另一个有用的步骤是使用 F_VN_BrightBorderObjects [▶ 1284] 和 F_VN_SubtractImages [▶ 543] 函数删除边框对象。这样可以消除图像区域边缘的干扰对象。这些情况通常发生在 ROI 定义不精确的情况下，例如相邻字符或行的部分内容无意中被包含在部分中，尤其是当它们之间的距离非常小时。

在示例项目中执行的预处理步骤如下：

1. Set ROI



2. Filter image



3. Binarize image



4. Remove border objects



代码

```
// Set ROI
hr := F_VN_SetRoi_TcVnRectangle_UDINT(stRoi, ipBinaryImage, hr);

// Filter image
hr := F_VN_CreateStructuringElement(ipStructElem, ETcVnStructuringElementShape.TCVN_SES_RECTANGLE,
3,3, hr);
```

```
hr := F_VN_MorphologicalOperator(ipBinaryImage, ipBinaryImage,
ETcvnMorphologicalOperator.TCVN_MO_OPENING, ipStructElem, hr);

// Binarize image depending on bright or dark text color
IF bInvertImage THEN
    hr := F_VN_Threshold(ipBinaryImage, ipBinaryImage, fThreshold, 255, TCVN_TT_BINARY_INV, hr);
ELSE
    hr := F_VN_Threshold(ipBinaryImage, ipBinaryImage, fThreshold, 255, TCVN_TT_BINARY, hr);
END_IF

// Remove border objects
hr := F_VN_BrightBorderObjects(ipBinaryImage, ipThreshBorder, hr);
hr := F_VN_SubtractImages(ipBinaryImage, ipThreshBorder, ipBinaryImage, hr);
```

OCR 应用

预处理成功后，即可使用 OCR 功能。在示例项目中，bUseExpFunction 参数可用于在标准和专家版本之间切换。以下描述涉及专家版本的使用以及附加选项。

sPattern 参数用于指定要识别的字符格式。如果事先知道要识别的字符的顺序和类型，该选项尤其有用。然后，您可以为字符串中的每个位置指定期望出现的字符。这为函数提供了每个字符应使用哪个识别模型的信息，从而可以获得更好的结果。特定的模型包含更少的字符，从而降低了混淆的风险。此外，如果识别出的字符与默认值不匹配，函数会直接返回 S_FALSE。

对于所示图像，“uu#ddddddddd”模式用于识别字符串。这意味着 TCVN_OMT_UCLETTERS 模型的前两个字符用于识别大写字母。然后，该函数使用 TCVN_OMT_NUMBERS_SC 模型来处理以下 9 个字符，其中第一个位置预计会出现一个特殊字符，其余 8 个位置每个位置都会出现一个单位数字。

代码

```
hrOCR := F_VN_OCRExp(
            ipSrcImage      := ipBinaryImage,
            eModel          := ETcVnOcrModelType.TCVN_OMT_NUMBERS_SC or
ETcVnOcrModelType.TCVN_OMT_UCLETTERS,
            ipCharacters    := ipOCRResult,
            sPattern        := sPattern,
            eOcrOptions     := eOcrOptions,
            ipBoundingBoxes := ipBoundingBoxes,
            ipConfidences   := ipConfidences,
            hrPrev          := hr,
            fMinConfidence  => fMinConfidence);

// Check if characters were found
hr := F_VN_GetNumberOfElements(ipOCRResult, nNumberOfElements, hrOCR);
IF SUCCEEDED(hr) AND nNumberOfElements > 0 THEN
    // Export character to string
    hr := F_VN_ExportSubContainer_String(ipOCRResult, 0, sText, 255, hr);
    // Write text result to image
    hr := F_VN_PutText(sText, ipOriginalImage, stRoi.nX + 5, stRoi.nY + 25, TCVN_FT_HERSHEY_DUPLEX,
1, aGreenColor, hr);
    // Further processing …
END_IF
```

OCR 结果评估

调用 OCR 函数后，应首先评估并检查 HRESULT 返回值，以确认是否识别出字符以及识别出了多少个字符。如果返回 S_OK，则表示标准函数已识别出字符，而对于专家版本，这取决于是否传递了 sPattern 默认值。如果存在默认值，则仅当识别出的字符与其匹配时才会返回 S_OK。否则，将返回 S_False，您可以自行检查已识别出的字符。

如果识别出字符，则可将它们导出到字符串，例如绘制到图像中，以便进行视觉反馈。此外，还可以将字符数、整个字符串或单个字符与存储的预期值进行比较。由于这些评估方式因具体应用而异，而且可以通过标准 PLC 功能实现，因此本示例中未使用它们。

使用专家版本时，可以导出、查看已识别字符的各个分类置信度，并将其用作验收标准，以便进行更详细的分析。

为了能够分配边界、重叠或可能错误识别的字符，可以调用各个已识别字符的边界框模块，对其进行评估并将其绘制到图像中。进行计算和绘制时，可能需要考虑先前设置的 ROI。

代码

```
// Get number of Confidence elements and export them if array is large enough
hr := F_VN_GetNumberOfElements(ipConfidences, nNumberOfElements, hr);

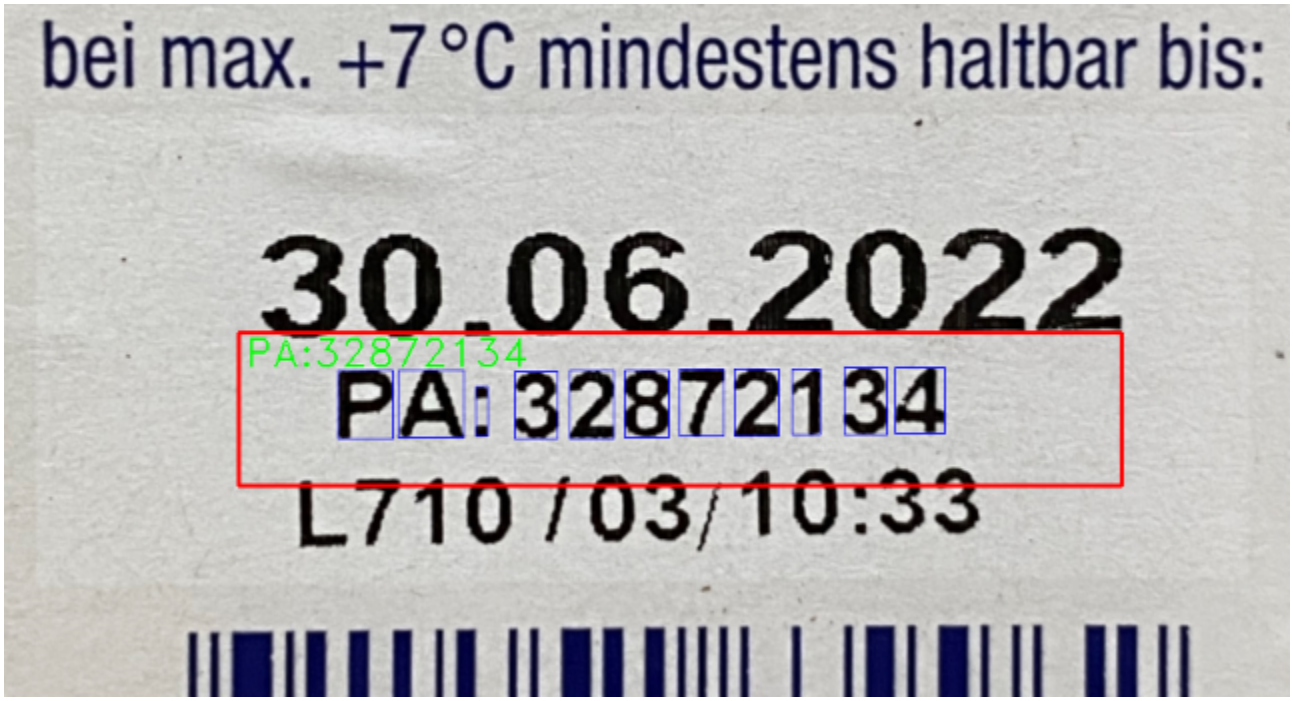
// Check if number of elements fits to array size
IF nNumberOfElements > 0 AND nNumberOfElements <= 12 THEN
    hr := F_VN_ExportContainer(ipConfidences, ADR(aConfidences), SIZEOF(aConfidences), hr);
```

```
END_IF

// Get bounding box rectangle and draw it to filtered and original image
hr := F_VN_GetNumberOfElements(ipBoundingBoxes, nNumberOfElements, hr);
IF nNumberOfElements > 0 THEN
    FOR nIterator := 0 TO nNumberOfElements -1 DO
        hr := F_VN_GetAt_TcVnRectangle_DINT(ipBoundingBoxes, stRectangle, nIterator, hr);
        hr := F_VN_DrawRectangle_TcVnRectangle_DINT(stRectangle, ipBinaryImage, aWhiteColor, 1, hr);
        // Add ROI Offset
        stRectangle.nX := stRectangle.nX + UDINT_TO_DINT(stRoi.nX);
        stRectangle.nY := stRectangle.nY + UDINT_TO_DINT(stRoi.nY);
        hr := F_VN_DrawRectangle_TcVnRectangle_DINT(stRectangle, ipOriginalImage, aBlueColor, 1,
hr);
    END_FOR
END_IF
```

结果呈现

以下图像展示了专家函数的结果显示。已识别字符以绿色显示，可选的边界框模块以蓝色显示。



每个已识别的字符的置信度值可在 `aConfidences` 数组中用于进一步评估。

Expression	Type	Value
aConfidences	ARRAY [0..11] OF REAL	
aConfidences[0]	REAL	1
aConfidences[1]	REAL	1
aConfidences[2]	REAL	0.813590169
aConfidences[3]	REAL	1
aConfidences[4]	REAL	1
aConfidences[5]	REAL	0.9643206
aConfidences[6]	REAL	1
aConfidences[7]	REAL	1
aConfidences[8]	REAL	1
aConfidences[9]	REAL	1
aConfidences[10]	REAL	0.9882314
aConfidences[11]	REAL	0

8.1.12 自行编写的函数

本示例说明了为 Vision 应用编写自定义函数时需要注意的事项。主要涉及以下方面：

- HRESULT [▶ 135] 的处理
- 接口指针 [▶ 132]和引用的处理

其他结构同样如此

该函数实现基于 TwinCAT Vision API 函数的结构。其他结构也是可行的，但请注意此处说明的要点。

方法同样如此

本示例描述了一个函数。不过，它同样适用于方法。

为此，我们以对象计数与绘制的自写入函数为例进行说明：

```
FUNCTION F_CountAndDrawObjects : HRESULT
```

声明部分

该函数的属性在 VAR_INPUT 部分中声明如下。原始图像作为 ITcVnImage 直接传输。由于在函数或方法中不存储变量值，因此在这种情况下无需注意其他事项。另一方面，结果图像作为 REFERENCE TO ITcVnDisplayableImage 传递，因为它在函数内部创建，并且需要返回外部，以便进行进一步处理。在这种情况下，由于引用的存在，不会创建接口指针的副本。

```
VAR_INPUT
    ipSrcImage          :   ITcVnImage;
    ipDestImage         :   REFERENCE TO ITcVnImage;
    aColor              :   TcVnVector4_LREAL;
    nNumberOfObjects    :   REFERENCE TO ULINT;
    hrPrev              :   HRESULT;
END_VAR
```

此外，函数内部所需的一些辅助变量在 VAR 中声明。

```
VAR
    hr                  :   HRESULT;
    ipContours          :   ITcVnContainer;
    stParams            :   TcVnParamsBlobDetection;
END_VAR
```

输入检查

在函数的起始处，会执行一些输入检查，以确保函数能够正确执行。

最初，FAILED(hrPrev) 用于检查在先前的处理链中是否发生了错误。如果发生了错误，则显示该错误并直接结束函数。

```
IF FAILED(hrPrev) THEN
    F_CountAndDrawObjects := hrPrev;
    RETURN;
END_IF
```

然后，程序会检查作为 VAR_INPUT 传递给函数的所有接口指针和引用是否有效。对于接口指针，通过与 0 进行比较来执行检查；对于引用，则通过 __ISVALIDREF() 进行检查。如果至少有一个参数无效，则该函数无法正常工作并返回 INVALIDPARM。

```
IF ipSrcImage = 0 OR NOT __ISVALIDREF(ipDestImage) OR NOT __ISVALIDREF(nNumberOfObjects) THEN
    F_CountAndDrawObjects := Tc2_System.E_HRESULTAdsErr.INVALIDPARM;
    RETURN;
END_IF
```

如果所有输入检查均成功，则必须增加作为 VAR_INPUT 传递的所有接口指针的引用计数器，这是因为在传递接口指针时会创建一个副本，该副本在功能块执行后仍被保留，因此必须在引用计数器中进行映射。此处可以直接执行方法 TcAddRef，而无需事先进行 <> 0 检查，因为该项检查已在输入检查阶段完成。正如声明部分所述，对于函数和方法而言，增加引用计数器并非必要，因为变量值（如本例中的指针地址）在执行后不会被保留。

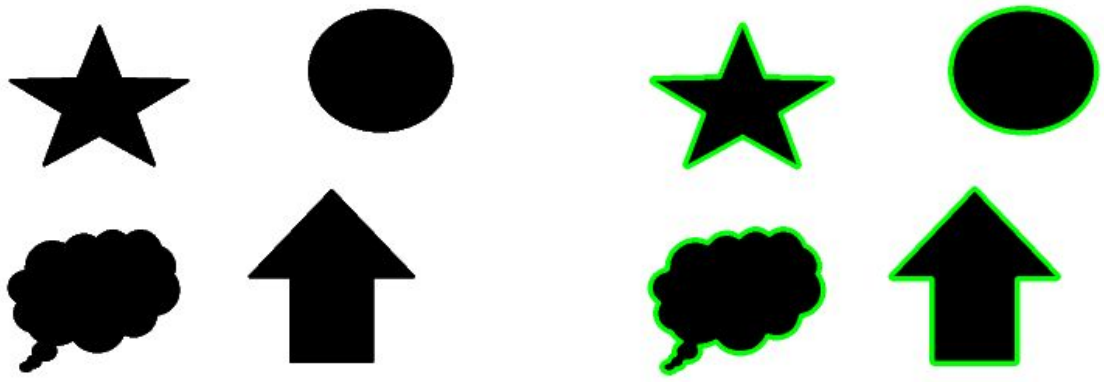
主要处理

实际图像处理顺序位于函数的主要部分。此处仅为示例，不再进一步展开说明。
F_VN_ConvertColorSpace 的调用表明，为了返回结果而作为 REFERENCE TO 传递的接口指针可以由 TwinCAT Vision API 函数直接写入。

```
hr := F_VN_ConvertColorSpace(ipSrcImage, ipDestImage, TCVN_CST_GRAY_TO_RGB, hr);
stParams.bFilterByArea := TRUE;
stParams.fMinArea := 10_000;
stParams.fMaxArea := 100_000;
hr := F_VN_DetectBlobs(ipSrcImage, ipContours, stParams, hr);
hr := F_VN_GetNumberOfElements(ipContours, nNumberOfObjects, hr);
hr := F_VN_DrawContours(ipContours, -1, ipDestImage, aColor, 5, hr);
```

通过该图像处理顺序，对象会被绘制在结果图像中，且其数量以 ULINT 的形式返回：

原始图像 **结果图像**



找到的对象数量：4

清理与返回

在实际图像处理顺序之后，函数内部声明的接口指针会被再次释放。之所以会出现这种情况，是因为函数中声明的所有变量（即接口指针）在函数结束后都会被删除。如果后台仍有尚未释放的数据，则会导致内存泄漏。

```
FW_SafeRelease(ADR(ipContours));
```

最后，将 HRESULT 作为返回值赋给函数，以便向外部报告任何错误。为了确保其不被篡改，FW_SafeRelease 函数的返回值不应赋值给 HRESULT。

```
F_CountAndDrawObjects := hr;
```

8.2 功能块样本

本节包含功能块的样本。

8.2.1 相机寄存器访问

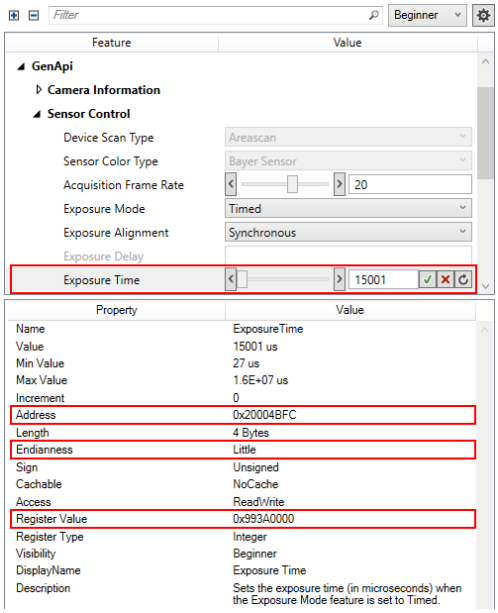
8.2.1.1 读/写寄存器

所有可以通过配置助手在配置模式下读写的相机参数也可以通过功能块FB_VN_ReadRegister_UDINT [▶ 1573]和FB_VN_ReadRegister_REAL [▶ 1572]从 PLC 读取或通过FB_VN_WriteRegister_UDINT [▶ 1579]和FB_VN_WriteRegister_REAL [▶ 1578]写入。所需功能块取决于参数的数据类型，且可以在寄存器类型的属性列表中读取。在这个样本中，曝光时间以整数读取和写入。

```
fbReadValue    :    FB_VN_ReadRegister_UDINT;
fbWriteValue   :    FB_VN_WriteRegister_UDINT;
```

两个功能块都需要参数的地址和字节序列的指示（字节序列）。此外，写入功能块还要求待写入相机的参数值。由于在数值下显示或输入的值不需要直接符合相机内部的处理方式，例如在枚举或其他字节序列的情况下，可以在**数值**寄存器下读取。

这些信息可以从配置助手中读取：



注意

相机特有的参数

此处显示的参数名称、属性和值特定于供应商或相机和固件，因此可能有所不同。

变量

```
bTriggerReadValue  :   BOOL;
nReadValue         :   UDINT;
fbReadValue        :   FB_VN_ReadRegister_UDINT;
nReturnCodeRead    :   UDINT;

bTriggerWriteValue :   BOOL;
nWriteValue        :   UDINT := 5000;
fbWriteValue       :   FB_VN_WriteRegister_UDINT;
nReturnCodeWrite   :   UDINT;
```

代码

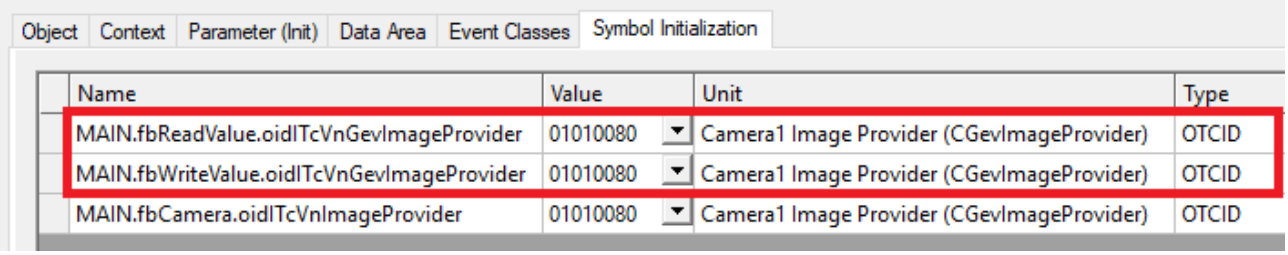
```
fbReadValue(
    nAddress    :=   16#20004BFC, // Check the Address of the Camera Parameter
    nEndian     :=   1,           // 0 = Big, 1 = Little
    bRead       :=   bTriggerReadValue,
    nTimeout    :=   T#5S,
    nValue      =>   nReadValue);

fbWriteValue(
    nAddress    :=   16#20004BFC, // Check the Address of the Camera Parameter
    nValue      :=   nWriteValue, // Check the right Input Format
    nEndian     :=   1,           // 0 = Big, 1 = Little
    bWrite      :=   bTriggerWriteValue,
    nTimeout    :=   T#5S);
```

读取和写入参数的条件

一般来说，仅在打开相机连接的情况下才能读写参数（`ETcVnCameraState >= TCVN_CS_OPENED`）。关于具体条件，请参考相机的相关供应商文件。

两个功能块必须通过相机的**图像提供者**进行初始化：



附图 34: 将功能块与相机实例联系起来

8.2.2 文件访问

8.2.2.1 保存 PLC 的图像

在这个样本中，来自 PLC 的图像被保存到本地驱动器。功能块FB_VN_WriteImage [▶ 1596]用于此目的。

保存过程在几个周期内异步进行。

保存过程的时间

保存过程的持续时间取决于各种因素，因此不能笼统地说。

变量

```
hr                      :   HRESULT;

fbCamera                :   FB_VN_SimpleCameraControl;

ipImageIn               :   ITcVnImage;
ipImageInDisp           :   ITcVnDisplayableImage;

// Sample Specific Variables
fbWriteImage            :   FB_VN_WriteImage := (nTimeout := T#500MS);
sFilePath               :   STRING(255) := '';
bWriteImageTrigger      :   BOOL;
bWriteImageWaitResult:  :   BOOL;
bWriteImageDone         :   BOOL;
nReturnCode             :   UDINT;
```

在这个样本中，通过将触发变量bWriteImageTrigger设置为TRUE，可以保存一个图像。

代码

在第一次调用fbWriteImage且上升沿处于bWrite时，已经完全接受了转移的ipImage，因此可以在之后立即释放，例如使用F_VN_TransformIntoDisplayableImage。此后，fbWriteImage必须继续循环调用，直到bBusy输出再次为FALSE。这可能是一个错误的情况或者第一次调用的情况，例如，如果没有指定文件扩展名或指定了不正确的扩展名。在下面的 IF 块中，将评估写入是否成功或是否发生错误。如果成功，bWriteImageDone设定且保持不变，直到下次设置bWriteImageTrigger。

```
IF SUCCEEDED(hr) AND ipImageIn <> 0 THEN

    IF NOT fbWriteImage.bBusy AND bWriteImageTrigger THEN
        bWriteImageTrigger := FALSE;
        bWriteImageDone := FALSE;
        bWriteImageWaitResult := TRUE;

        // With setting sFilePath:= '' to an empty string the images are saved under the default
path
        // The default path can be set at the service configuration tab of the Vision Node
        // If sFilePath is set, it must contain the full path, image name and type
        // for example: sFilePath := 'C:\WriteImage\ImageName.bmp'
        fbWriteImage(ipImage := ipImageIn, sFilePath := sFilePath, bWrite := TRUE);
    END_IF

    hr := F_VN_TransformIntoDisplayableImage(ipImageIn, ipImageInDisp, hr);
END_IF

fbWriteImage(sFilepath:= '', bWrite := FALSE);
```

```
IF bWriteImageWaitResult AND NOT fbWriteImage.bBusy THEN
    bWriteImageWaitResult := FALSE;
    // Check if fbWriteImage was finished successfully or implement error handling
    IF fbWriteImage.bError THEN
        nReturnCode := fbWriteImage.nErrorId AND 16#FFF;
    ELSE
        bWriteImageDone := TRUE;
        nReturnCode := 0;
    END_IF
END_IF
```

8.2.2.2 连续加载图像到 PLC

在这个样本中，将以连续方式向 PLC 加载图像。为此，使用：

- FB_VN_ReadImage [▶ 1590]

这个样本同样适用于通过FB_VN_ReadContainer [▶ 1589]加载容器。

说明

一个经常使用的情况是，参考图像被保存，以便与控制器中的实时图像进行比较。这些参考图像应在操作过程中被替换，例如，当引入新的参考时。图像可以被重新加载任何次数和任何时间间隔。

为了不需要为每一个新的参考图像通知 PLC 文件名，定义一个命名惯例。例如，这可以是 `Image<Index>.png`，其中`<Index>`将被一个连续的整数所取代。然后，为每个新的参考图像提供一个文件名，索引递增 1。因此，PLC 知道下一个参考图像的名称是什么，并可以有目的地尝试加载。

注意

文件损坏的风险

切勿替换图像文件的内容；相反，仅在文件系统中创建新的图像！首先，由于视觉服务的内部缓存机制，被替换的图像未寄存。其次，几个进程同时访问一个图像会导致文件损坏。缓存设置可以交替调整。

应用

6 张彩色图片（名称为`image<Index>.png`）和空文件夹`load`附在这个样本上。将`load`文件夹的绝对路径定义为`sBasePath`。启动样本 PLC 后，按顺序将图像（从`Index=0`开始）移动到`load`文件夹，并在 ADS 图像查看中观察图像`ipImageRefDisp`的相应变化。

程序

在循环程序中，根据索引`nIndex`，最初计算待加载文件的路径。另外，调整变量声明中的基本路径 `sBasePath`，使其与你的图像的内存位置相匹配。

```
sPath := CONCAT(CONCAT(sBasePath, TO_STRING(nIndex)), '.png');
```

通过计算出的路径，循环调用类型为FB_VN_ReadImage [▶ 1590]的功能块 fbReadImage。

```
fbReadImage(
    sFilePath   :=  sPath,
    ipDestImage :=  ipImageIn,
    bRead       :=  TRUE,
    nTimeout    :=  T#1S
);
```

除了调用功能块外，还必须在它准备好时做出反应。一方面，通过调用`bRead:=FALSE`重置功能块；另一方面，如果执行成功，增加索引并将图像转发到所需的接口指针。必须查询是否发生了错误，因为只有在当前图像已经被成功加载的情况下，才可以增加索引。

```
IF NOT fbReadImage.bBusy THEN
    IF NOT fbReadImage.bError THEN
        nReturnCode := 0;
        nIndex := nIndex + 1;

        IF ipImageIn <> 0 THEN
            FW_SafeRelease(ADR(ipImageRef));
            ipImageRef := ipImageIn;
            ipImageIn.TcAddRef();
            FW_SafeRelease(ADR(ipImageIn));
        END_IF
    ELSE
```

```
        nReturnCode := fbReadImage.nErrorId AND 16#FFF;
    END_IF
    fbReadImage(sFilePath:='', bRead:=FALSE);
END_IF
```

为了节省内存和计算时间，此处的图像转发通过接口有序转移而不是通过拷贝进行。

8.2.3 图像采集

8.2.3.1 通过文件名触发图像

在这个样本中，根据功能块FB_VN_FileSourceControl [▶ 1601]中的文件名，特意触发了图像。

应用

该样本交替显示两个文件名为Image1.bmp和Image2.bmp的图像（来自ADS Image Watch [▶ 128]的文件源）。样本包括带有相应文件名的图像。你可以用自己的图片替换样本图像，或者另外添加图像。为此，在`sFileName`中更改文件名或添加到文件名中，如有必要，调整元素的数量。注意，所有待显示的图像必须添加在文件源控制 [▶ 124]中。通过设置`bLoop`为`false`，可以停止图像的自动切换，然后可以通过设置`nImageIndex`选择待触发的图像。

程序

```
hr              :   HRESULT;
sFileName       :   ARRAY [0..1] OF STRING := ['Image1.bmp', 'Image2.bmp'];
fbFileSource : FB_VN_FileSourceControl;
eState          :   ETcVnCameraState;
ipImageIn       :   ITcVnImage;
ipImageInDisp   :   ITcVnDisplayableImage;
bTrigger        :   BOOL := TRUE;
bLoop           :   BOOL := TRUE;
nImageIndex     :   UINT := 0;
```

fbFileSource	用于控制类型FB_VN_FileSourceControl [▶ 1601]的文件源 [▶ 123]的功能块。
eState	文件源的状态
ipImageIn	输入图像
ipImageInDisp	可显示的输入图像
bTrigger	触发器标记
sFileName	文件名 - 更改这个变量，以便加载不同的图像。然而，这个图像必须在添加在文件源控制 [▶ 124]中。
hr	类型为HRESULT [▶ 135]的状态变量。

一般来说，通常的状态机被用来操作文件源：

```
eState := fbFileSource.GetState();
CASE eState OF

<...>

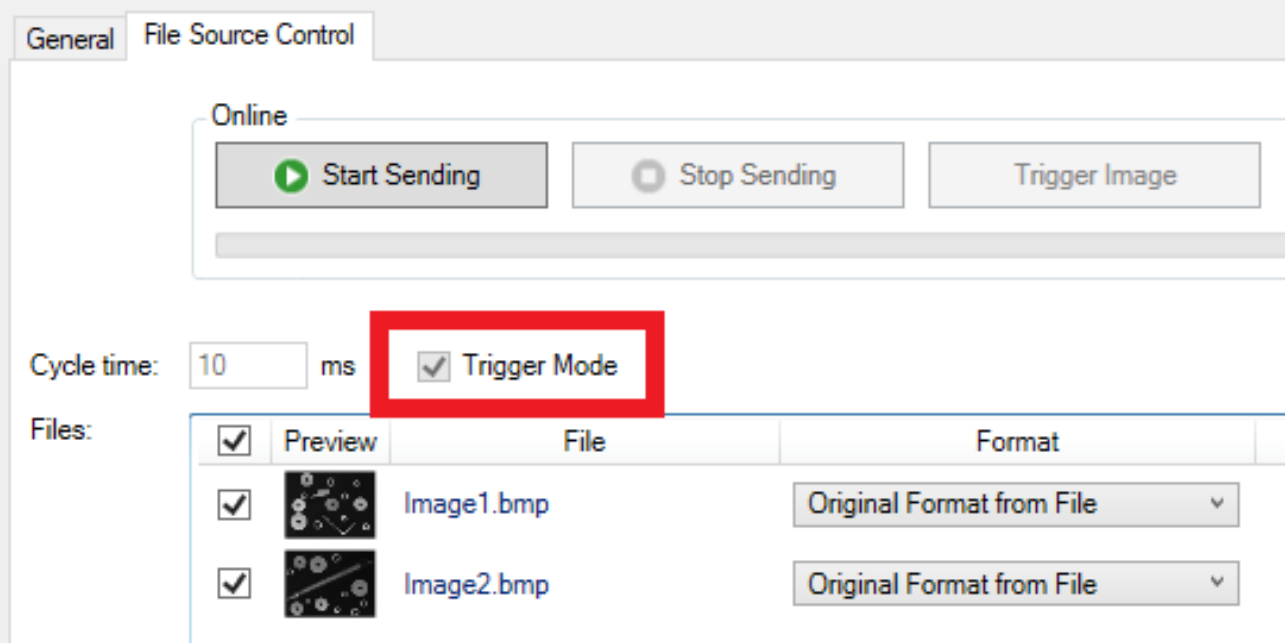
END_CASE
```

出错时的复位和图像采集的开始都是在相应的状态下正常进行：

```
TCVN_CS_INITIAL, TCVN_CS_INITIALIZING, TCVN_CS_INITIALIZED, TCVN_CS_OPENING, TCVN_CS_OPENED,
TCVN_CS_STARTACQUISITION:
    fbFileSource.StartAcquisition();

TCVN_CS_ERROR:
    fbFileSource.Reset();
```

然而，为了使文件源在开始采集图像时不立即加载图像，必须在文件源控制中激活**触发模式**：



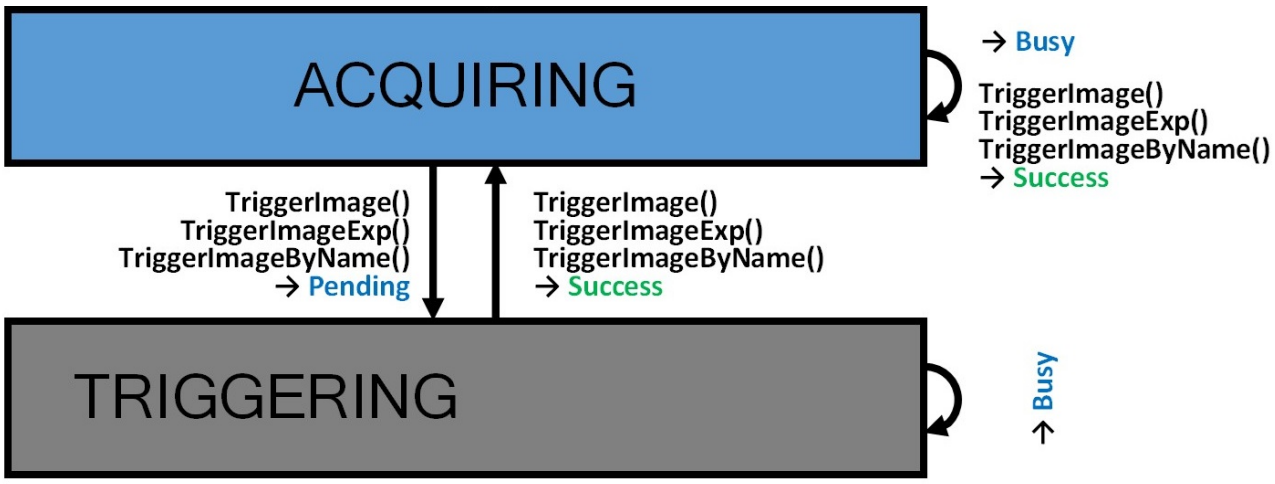
在ACQUIRING状态下，现在将交替地触发和接收具有相应文件名的图像。这里明确使用了TriggerImageByName [▶ 1609]的方法来触发图像。这个方法只存在于功能块FB_VN_FileSourceControl [▶ 1601]中，而不存在于FB_VN_SimpleCameraControl [▶ 1630]中。

```
TCVN_CS_ACQUIRING:
    IF bTrigger THEN
    hr := fbFileSource.TriggerImageByName(sFileName[nImageIndex]);
        IF SUCCEEDED(hr) THEN
            bTrigger := FALSE;

            IF bLoop THEN
                nImageIndex := nImageIndex + 1;
                IF nImageIndex >= 2 THEN
                    nImageIndex := 0;
                END_IF
            END_IF
        END_IF
    ELSE
        hr := fbFileSource.GetCurrentImage(ipImageIn);
        IF SUCCEEDED(hr) AND ipImageIn <> 0 THEN
            bTrigger := TRUE;
            hr := F_VN_TransformIntoDisplayableImage(ipImageIn, ipImageInDisp, hr);
        END_IF
    END_IF
```

由于文件源在触发图像时暂时进入TRIGGERING状态，因此必须在此状态下继续触发。这一直持续到文件源完全加载了图像，从而将其状态改回ACQUIRING。图像的文件名不需要为进一步触发而再次传输；只需调用方法TriggerImage [▶ 1608]即可。

```
TCVN_CS_TRIGGERING:
    fbFileSource.TriggerImage();
```



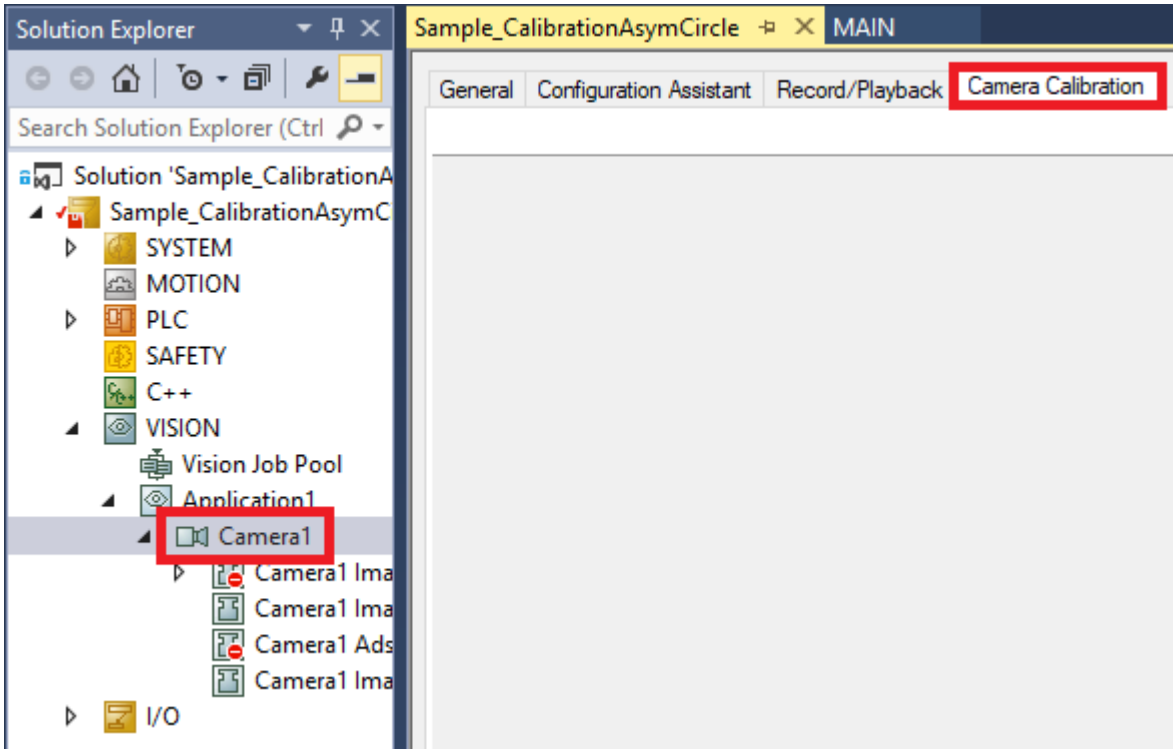
在这个样本中，变量bTrigger被自动设置为TRUE，以在图像成功加载时进行演示。在实际应用中，触发器将被设置在一个在序列上有意义的点上。

8.3 助手样本

8.3.1 校准助手

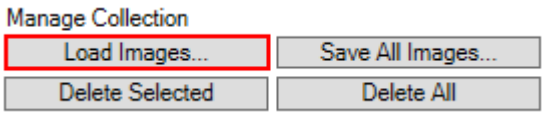
需要几何相机标定以将图像点转换为世界点，在此过程中确定所需的参数。该示例展示使用不对称圆圈模板和标定助手对相机进行几何标定。其他标定模板也可采取类似步骤（请参见 相机校准 [▶ 106]）。只有第 2 步需要做出相应调整。

要运行此示例，首先打开已创建相机的项目，然后打开相机实例的标定助手。



步骤 1 — 图像采集

首先，必须将待标定相机拍摄的标定模板图像加载到助手。这可以通过相机直接采集图像或加载现有图像来实现。对于本示例，您可以在相应的图像文件夹中找到一个不对称圆圈模板的四个合适图像ImgCalibAsymCirclesx.bmp。通过 **Load Images....（加载图像……）** 将这些图像加载到标定助手。

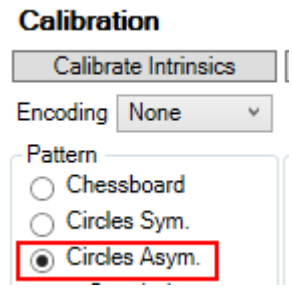


步骤 2 — 确定不对称圆圈模板

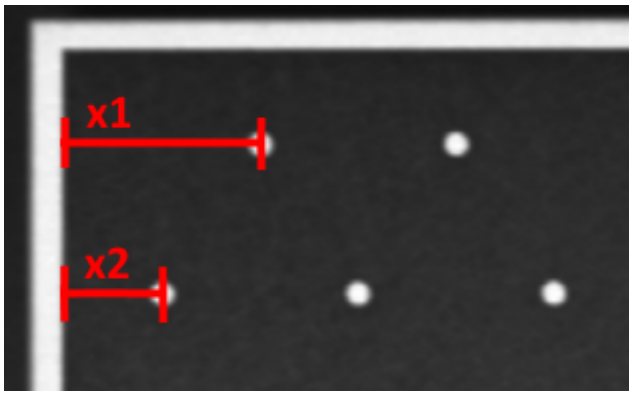
在本例程中，使用了非对称圆圈模板作为标定模板。要使用其他模板类型，请遵守相应定义：

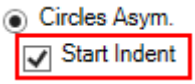
- 棋盘图案 [▶ 114]
- 对称圆形图案 [▶ 116]
- 不对称的圆圈图案 [▶ 116]
- 单个圆圈图案 [▶ 118]

必须在标定助手中为加载的示例图像选择不对称圆圈模板：

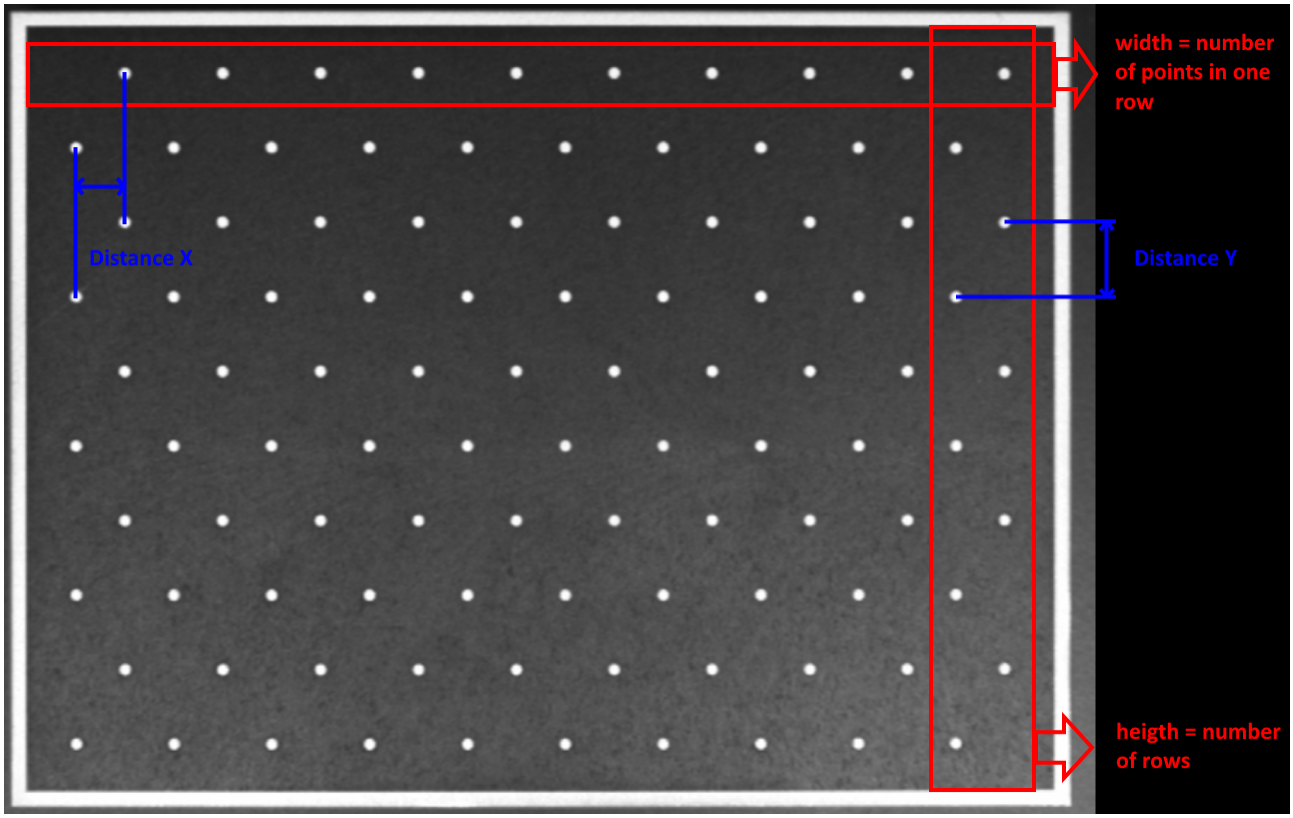


由于不对称，圆圈模板的第一行（x1）或第二行（x2）均缩进。在这种情况下，第一行 (x1) 缩进，因此应选择 **Start Indent（开始缩进）**。

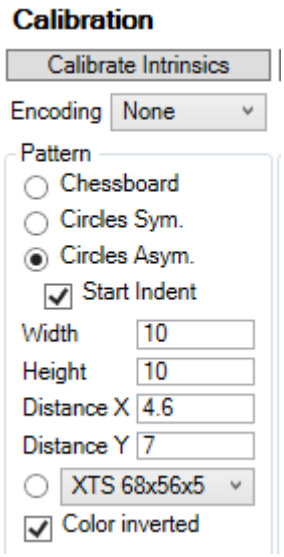




宽度模板的宽度与行中点/圆圈的数量相对应，而**高度**则与行数相对应。**距离 X**用于指定两个圆圈在 X 方向上的最短距离。在不对称模板中，这是下一行/上一行到下一个圆的距离。注意图像中的标记。两行之间的距离在**距离 Y** 中指定。

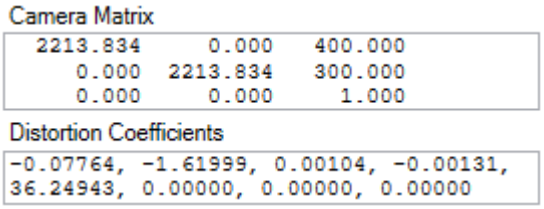


助手希望模板为白底黑圈。由于本示例使用黑底白圈模板，因此必须通过选择**颜色反转**来反转颜色。



步骤 3 — 计算内部参数

标定模板完全确定后，可以使用 **Calibrate Intrinsics（标定内部）**按钮计算内部参数。**再投影 误差**应介于 0 和 1 之间，并应确定**相机矩阵**和**几何畸变系数**。

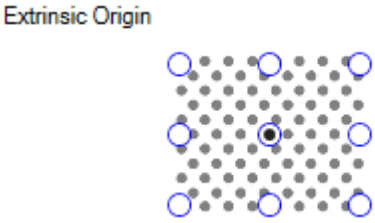


如果不是这种情况，**再投影 误差**工具提示可提供有关可能错误的信息。此外，C:\ProgramData\Beckhoff\Vision_CalibrationAssistantOutput 下还会创建一个图像系列，提供有关识别模板的进一步信息。该目录需要写入权限。

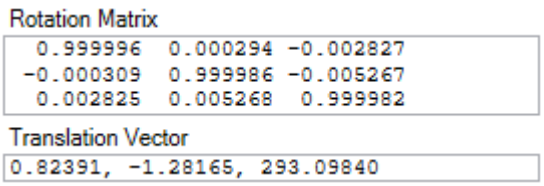
步骤 4 — 计算外部参数

在计算出内部参数后，可以计算外部参数。为此，只需使用一个图像，即模板定位在稍后测量组件等平面上的图像。这是第一个示例图像。因此，点击第一个加载图像，使其出现在图像预览中。

还可以指定**外源**模板的零点。默认情况下，它位于模板的中心。

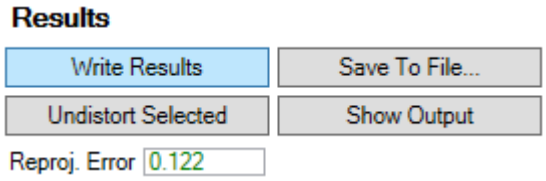


然后，**Calibrate Extrinsics（标定外部）**按钮会计算**旋转矩阵**和**平移向量**。结果可在 **Results（结果）**下查看。

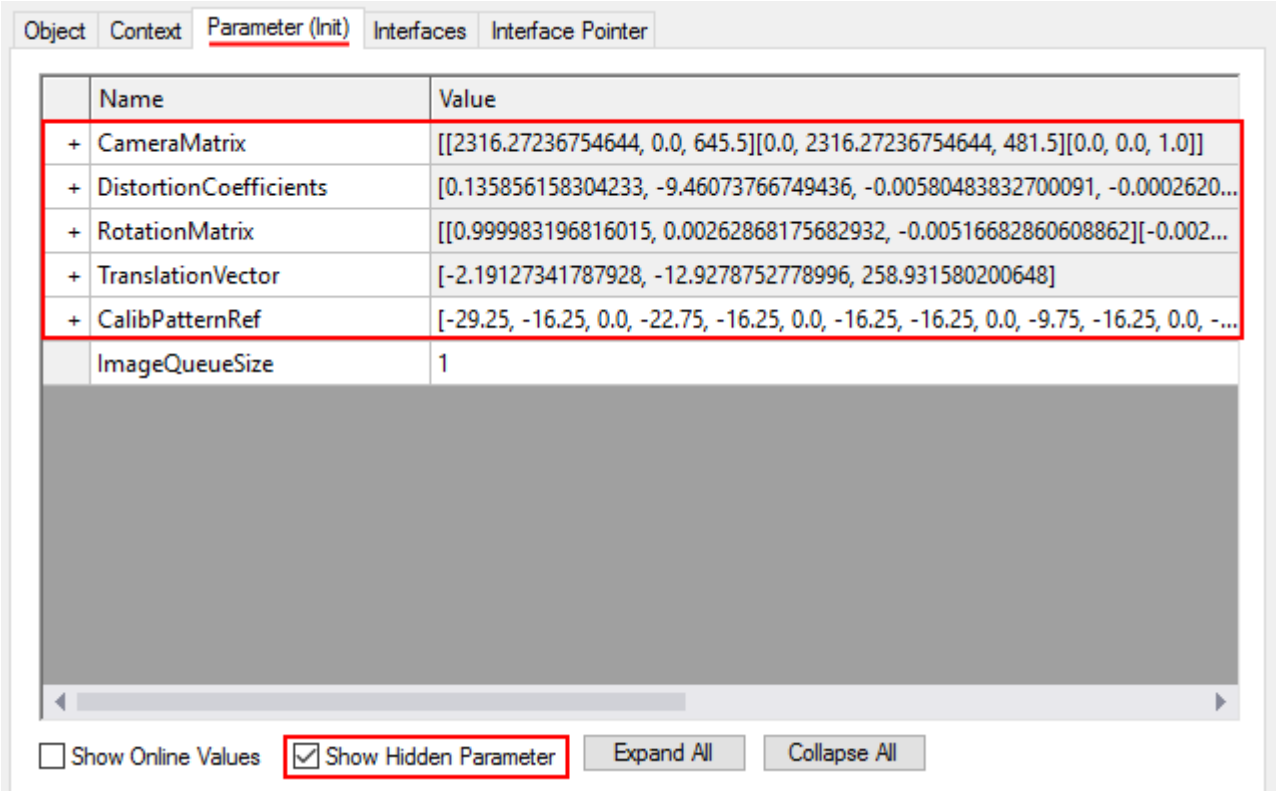


步骤 5 — 将结果写入相机的图像提供程序

如果标定成功，可通过 **Write Results（写入结果）**将结果写入相机的图像提供程序。这一步骤是必要的，这样 PLC 中的 FB_VN_GevCameraControl [▶ 1609] 相机功能块才能访问标定结果。



在图像提供程序中，如果选择了 **Show Hidden Parameters（显示隐藏参数）**，标定结果可以在 **Parameters （参数）（Init）**下找到。



完成此步骤后，请务必点击  激活配置。否则 PLC 将无法提供标定结果。

步骤 6 — 将标定结果载入 PLC

FB_VN_GevCameraControl [▶ 1609] 功能块可用于将标定参数加载到 PLC：

变量

```
hr                          :   HRESULT;
fbCamera                    :   FB_VN_GevCameraControl;
aCameraMatrix               :   TcVnMatrix3x3_LREAL;
aDistortionCoefficients     :   TcVnArray8_LREAL;
aRotationMatrix             :   TcVnMatrix3x3_LREAL;
aTranslationVector          :   TcVnVector3_LREAL;
```

代码

```
hr := fbCamera.GetCameraMatrix(aCameraMatrix);
hr := fbCamera.GetDistortionCoefficients(aDistortionCoefficients);
hr := fbCamera.GetRotationMatrix(aRotationMatrix);
hr := fbCamera.GetTranslationVector(aTranslationVector);
```

步骤 7 — 将像素转换为世界坐标

然后使用 F_VN_TransformCoordinatesImageToWorld_Points [▶ 1105] 函数确定现实世界中图像中的点相对于标定模板零点的坐标，例如：

变量

```
aPointImage                 :   TcVnPoint2_LREAL;
aPointWorld                 :   TcVnPoint3_LREAL
```

代码

```
hr := F_VN_TransformCoordinatesImageToWorld_Point(
        aSrcPoint                   :=  aPointImage,
        aDestPoint                  :=  aPointWorld,
        aCameraMatrix               :=  aCameraMatrix,
        aDistortionCoefficients     :=  aDistortionCoefficients,
        aRotationMatrix             :=  aRotationMatrix,
        aTranslationVector          :=  aTranslationVector,
        fZ                          :=  0,
        hrPrev                      :=  hr);
```

步骤 8 — 将世界点转换为图像坐标。

这种转换可以通过 F_VN_TransformCoordinatesWorldToImage_Points [▶ 1108] 函数进行反转，从而将世界坐标系中的点进行转换，使其能够显示在图像中的正确位置。这对于显示辅助线以便在图像中更好地定向非常有用。下方程序显示了图像中的世界坐标系。

变量

```
i                           :   INT;
aLine                       :   TcVnVector4_DINT;
aCoordinatesWorld           :   ARRAY[0..4] OF TcVnPoint3_LREAL := [ [0,0,0], [50,0,0], [-50, 0, 0],
[0, 50, 0], [0, -50, 0] ];
aCoordinatesImage           :   ARRAY[0..4] OF TcVnPoint2_REAL;
```

代码

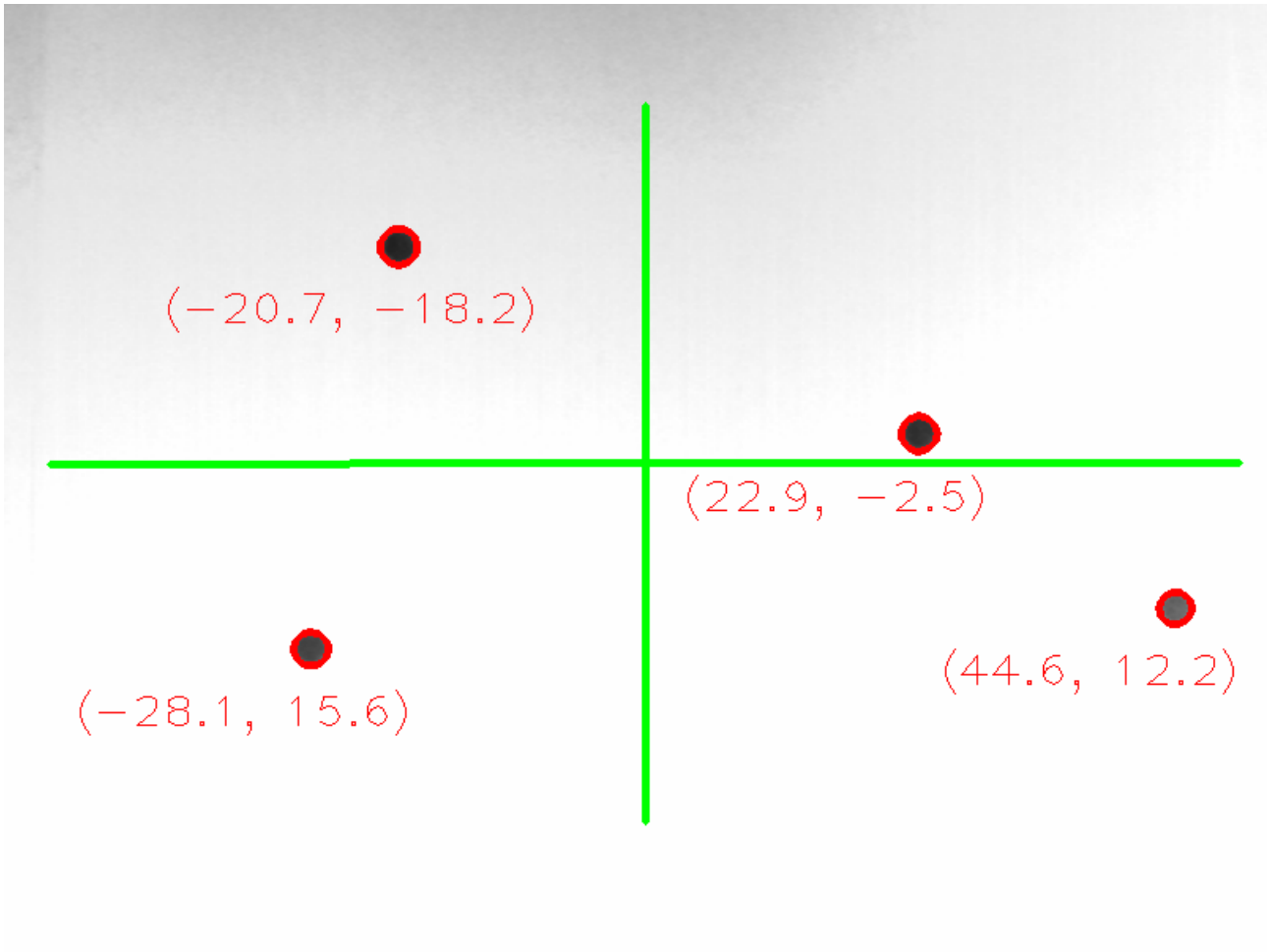
```
FOR i:= 0 TO 4 DO
    hr := F_VN_TransformCoordinatesWorldToImage_Point(
            aSrcPoint                   :=  aCoordinatesWorld[i],
            aDestPoint                  :=  aCoordinatesImage[i],
            aCameraMatrix               :=  aCameraMatrix,
            aDistortionCoefficients     :=  aDistortionCoefficients,
            aRotationMatrix             :=  aRotationMatrix,
            aTranslationVector          :=  aTranslationVector,
            hrPrev                      :=  hr);

    IF i > 0 THEN
        aLine[0] := REAL_TO_DINT(aCoordinatesImage[0][0]);
        aLine[1] := REAL_TO_DINT(aCoordinatesImage[0][1]);
        aLine[2] := REAL_TO_DINT(aCoordinatesImage[i][0]);
        aLine[3] := REAL_TO_DINT(aCoordinatesImage[i][1]);
    hr := F_VN_DrawLine_TcVnVector4_DINT(aLine , ipImageRes, aColorGreen, 3, hr);
    END_IF
END_FOR
```

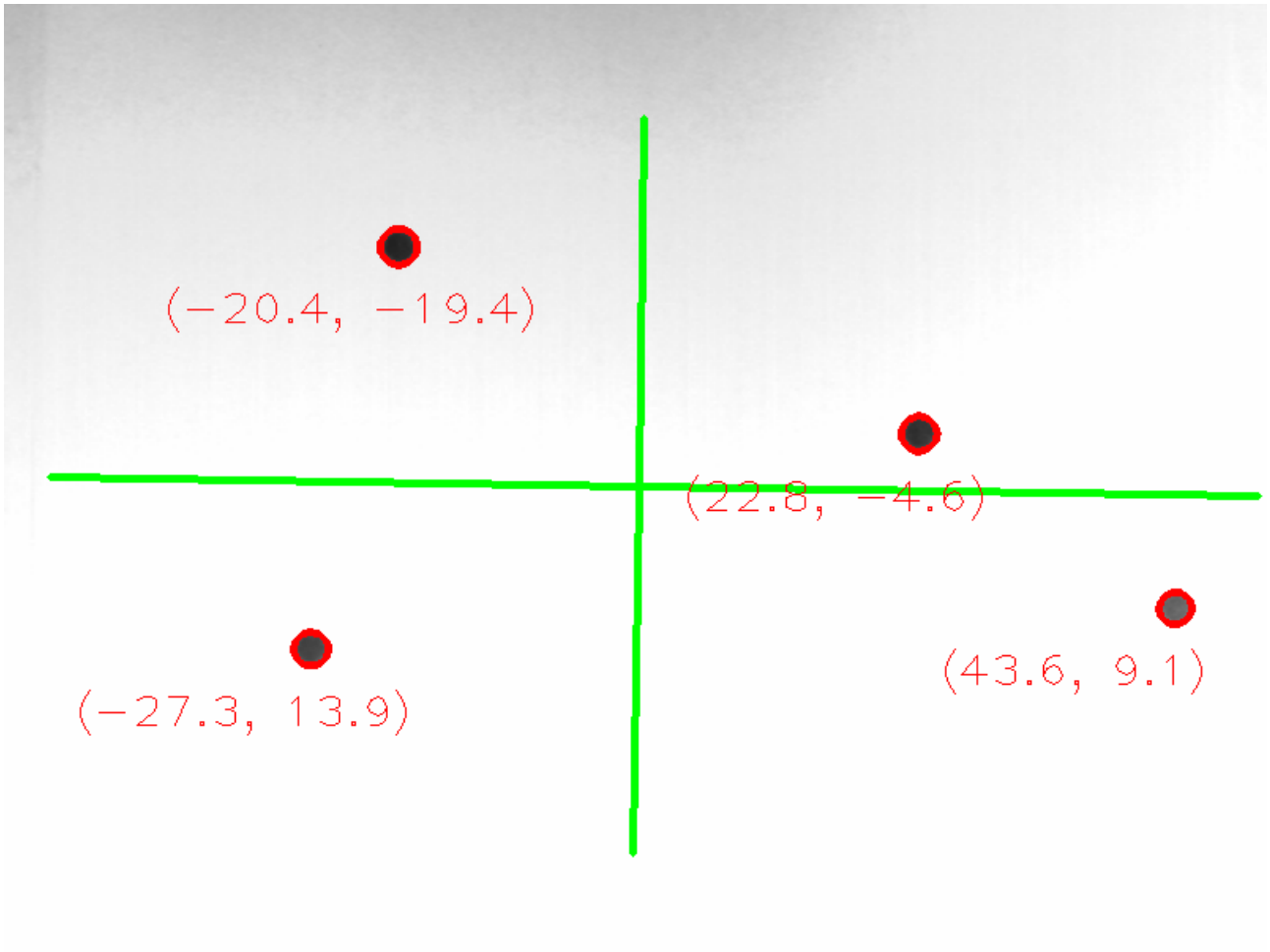
步骤 9 — 查看结果

执行程序时，必须先开始回放相机数据流 [▶ 104]。为此，请打开相机对象的 **Record/Playback（记录/回放）**选项卡，选择相应图像文件夹中的 CameraSimulation.tcs 文件并开始回放。

生成的图像可在 ADS Image Watch [▶ 128] 中进行评估。可以看到图像上绘制的坐标轴和确定的各个点的世界坐标。



如果在步骤 4 中使用不同的图像或坐标原点进行外部标定，结果将有所不同。坐标轴会发生改变，像素的世界坐标也会随之调整。



8.4 相机配置样本

由于每个 GigE Vision 相机的参数可能不同，因此不可能提供完整的描述。下面部分概述了经常使用的相机参数以及如何配置这些参数。关于 GenAPI 参数的配置说明，可以查看配置 [▶ 81]一章。

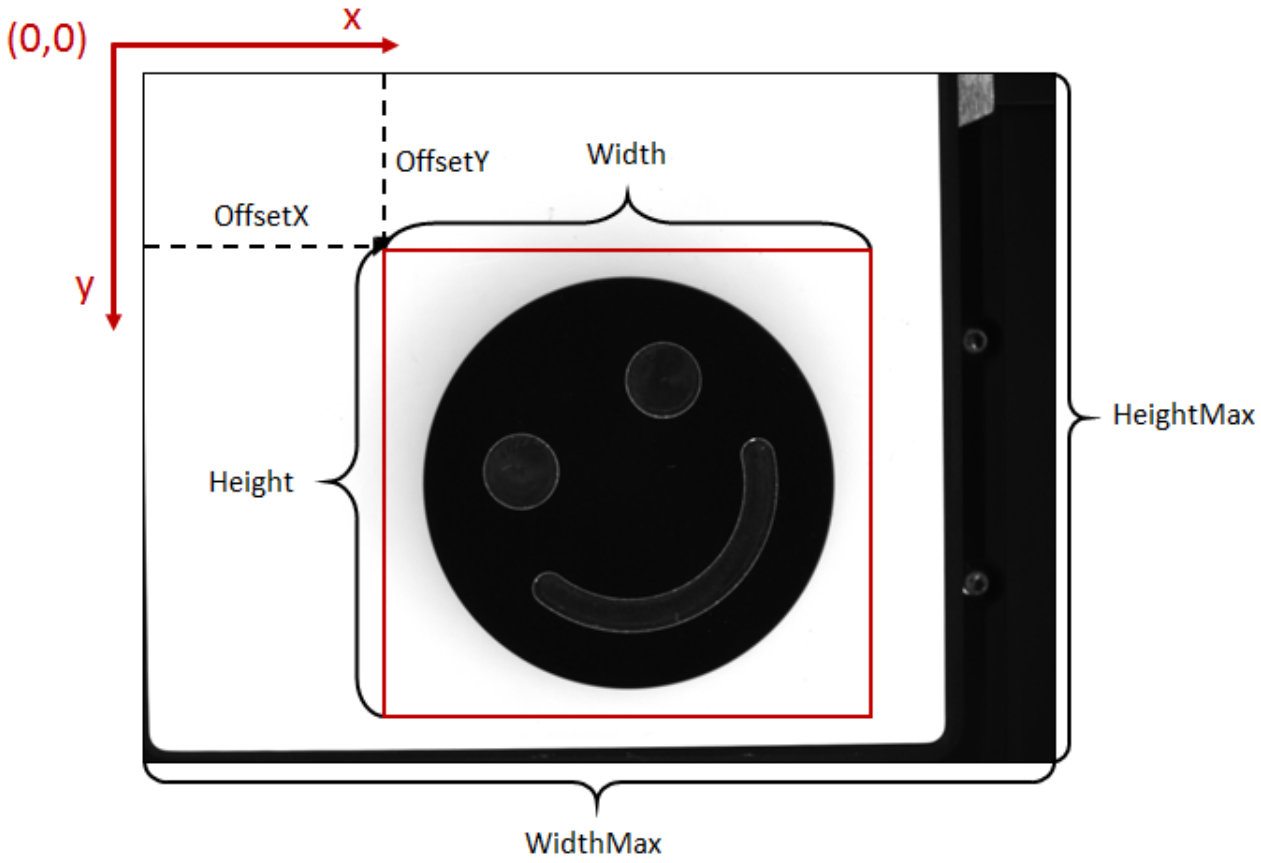
8.4.1 感兴趣区域（ROI）

如果相机传感器记录的视场比需要的大，建议在实际需要的视场上放置感兴趣区域（ROI）：

- 传感器可以更快地读出，具体取决于相机传感器和所选范围。
- 需要传输的数据量减少，也减少了数据传输时间。
- 同时，在整个图像上进行的所有图像处理算法的处理时间也会减少。
- 此外，可以减少图像噪音，使图像处理更加稳健和容易。

检查相机的功能

请参考相机手册，了解相机是否支持相应的功能。这同样适用于实际使用的功能名称。根据 GenICam 标准功能命名公约（2.4 版），使用功能名称的程序如下。



设置 ROI 的功能可以在ImageFormatControl或相机的相应类别下找到：

- WidthMax和HeightMax指定图像的最大宽度和高度，单位：像素。这些值可能因其他功能而不同（如像素合并 [▶ 2829]）。
- Width和Height指定了 ROI 的宽度和高度（即待查看的视场）且必须进行相应设置。
- OffsetX和OffsetY用于定位 ROI 或视场。起点是图像的左上角。

在设置功能时，请注意：

- OffsetX + Width <= WidthMax
- OffsetY + Height <= HeightMax

8.4.2 像素合并

像素合并是将几个像素值相加或平均变成一个。

它可以用于：

- 增加光的敏感性，
- 改善对比度，
- 在相同的视场下降低图像分辨率，使帧率和处理时间更快。

如果在图像分辨率降低时无法再达到所要求的精度，就不应该使用。

在对像素进行相加或平均时，将区分水平和垂直像素合并。

检查相机的功能

请参考相机手册，了解相机是否支持相应的功能。这同样适用于实际使用的功能名称。根据 GenICam 标准功能命名公约（2.4 版），使用功能名称的程序如下。

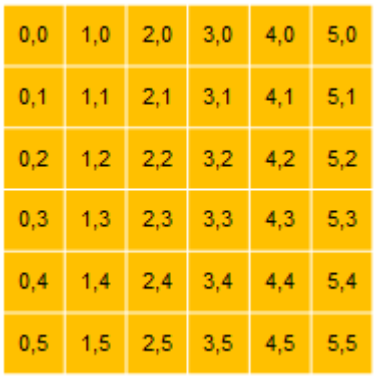
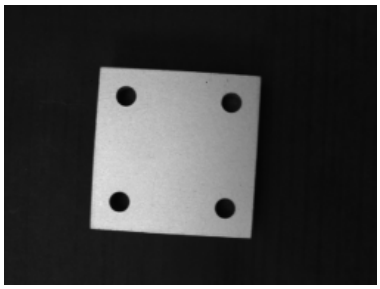
关于设置像素合并的功能，可以在ImageFormatControl下或相机的相应类别下找到：

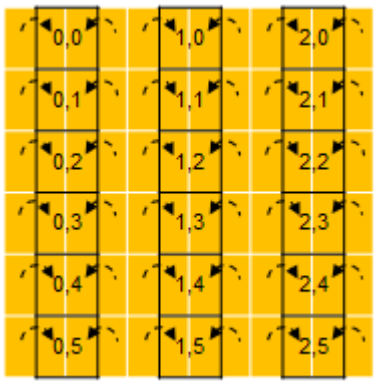
- `BinningSelector`表示像素合并是适用于整个`sensor`，还是适用于ROI [▶ 2828]。
- `BinningHorizontalMode`和`BinningVerticalMode`表示是否应将像素相加或平均。

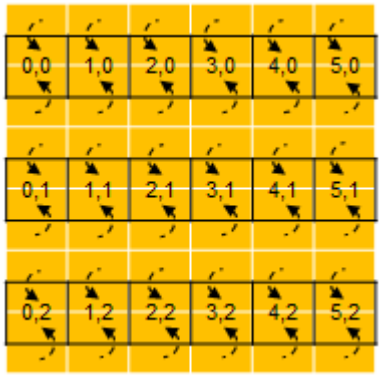
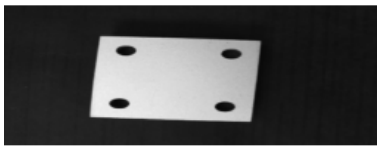
`BinningHorizontal`和`BinningVertical`表示在水平和垂直方向上应合并多少个像素。值为 1 表示在相应的方向上没有像素合并。

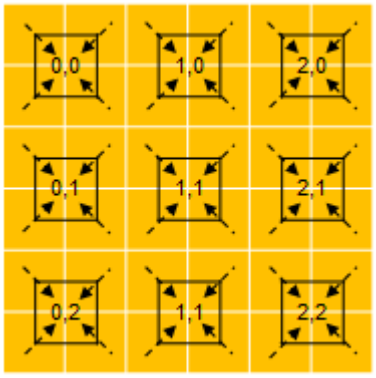
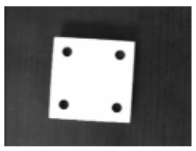
通过单色相机进行像素合并

举例展示单色相机的四种可能配置（显示了像素的坐标）：

`BinningHorizontal = 1`
`BinningVertical = 1`
⇒ 无“像素合并”
		

`BinningHorizontal = 2`
`BinningVertical = 1`
`BinningHorizontalMode = sum`
		

`BinningHorizontal = 1`
`BinningVertical = 2`
`BinningVerticalMode = sum`
		

BinningHorizontal = 2 BinningVertical = 2 BinningHorizontalMode = sum BinningVerticalMode = sum		

通过彩色相机进行像素合并

相比单色相机，彩色相机的像素合并支持更少。它可以通过不同的方式实施。详情请参考相机用户手册。

8.4.3 图像采集和触发

采集和触发功能相互关联，因此将在下文中一起说明。术语Acquisition基本上描述了可以拍摄图像的相机状态，见ImageAcquisition [▶ 1599]。是否以及何时实际拍摄和发送图像由Trigger参数的配置决定。

关于各种配置选项以及采集和触发设置之间的关系，将在下面的章节中进行更详细的描述（章节按照AcquisitionMode布置）。

目前为止，最常用的图像采集选项是：

- 连续图像采集 [▶ 2833] - 相机流，它在没有任何其他触发的情况下以固定的时间间隔捕捉图像。
- 单一图像触发，带连续图像采集 [▶ 2833] - 相机在触发信号（硬件或软件）后正好拍摄一张图像。相机一直处于ACQUIRING状态，因此在每次录像前不必重复写入AcquisitionStart命令。

检查相机的功能

请参考相机手册，了解相机是否支持相应的功能。这同样适用于实际使用的功能名称。根据 GenICam 标准功能命名公约（2.4 版），使用功能名称的程序如下。

采集模式

AcquisitionMode功能主要指定在采集过程中拍摄多少张图像，并指相机状态机的TCVN_CS_ACQUIRING状态。如需结束记录，必须通过AcquisitionStop命令退出状态，然后才能通过AcquisitionStart命令再次开始下一次记录。因此，Continuous模式是最经常选择的设置，因为这样可以永久地保持在采集状态。

- Continuous
  - 图像采集通过AcquisitionStart命令开始，且仅由AcquisitionStop命令终止。在这之间，根据触发器的设置，连续捕捉图像。
  - 连续的图像采集 [▶ 2832]
- MultiFrame
  - 采集图像，直到达到AcquisitionFrameCount中指定的图像数量。对于每个图像采集序列，都需要一个AcquisitionStart命令。
  - 多图像采集 [▶ 2834]
- SingleFrame
  - 拍摄到单个图像。对于每次图像采集，都需要一个AcquisitionStart命令。
  - 单一图像采集 [▶ 2836]

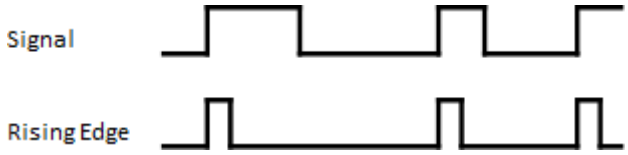
触发源

TriggerSource功能指定哪个内部或物理信号解释为触发器。可能来源清单很长，并且根据相机的不同而不同。下面提供一些样本：

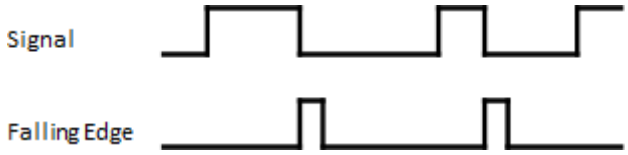
- Software
  一个软件命令设置了触发器。PLC 功能块FB_VN_GevCameraControl [▶ 1609]和FB_VN_SimpleCameraControl [▶ 1630]提供了发送命令的方法TriggerImage [▶ 1629]。
- Line0, Line1, ...
  物理相机的输入是触发源。注意与之相连的信号源的必要边缘陡度。这可能因相机的不同而不同。
- Encoder0, Encoder1, ...
  线扫描相机通常与一个编码器连接，以触发图像采集。如果没有物理编码器，但 TwinCAT 中有相应的轴值，则可以通过 EL2521 或 EL2522 输出编码器信号。

触发器的激活

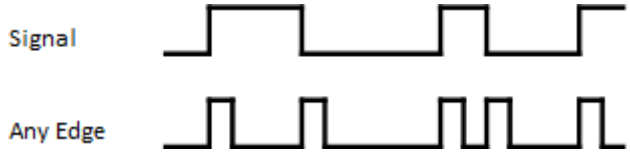
- RisingEdge
  触发器通过TriggerSource的上升沿激活。



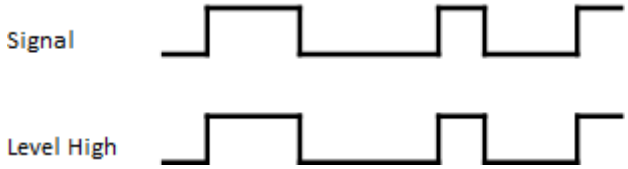
- FallingEdge
  触发器通过TriggerSource的下降沿激活。



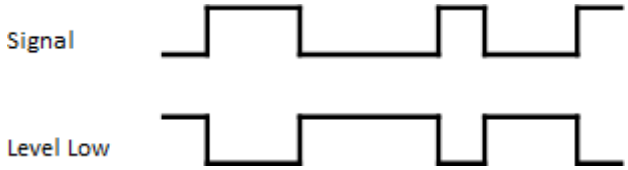
- AnyEdge
  触发器通过TriggerSource的上升或下降沿激活。



- LevelHigh
  只要有信号出现在TriggerSource，触发器就会被激活。



- LevelLow
  只要在TriggerSource没有信号出现，触发器就会被激活。



8.4.3.1 连续的图像采集

默认情况下，图像采集和触发的设置处于互锁状态。下面的解释部分假设AcquisitionMode设置为Continuous。

其他情况描述如下：

- 多图像采集 [▶ 2834]
- 单一图像采集 [▶ 2836]

检查相机的功能

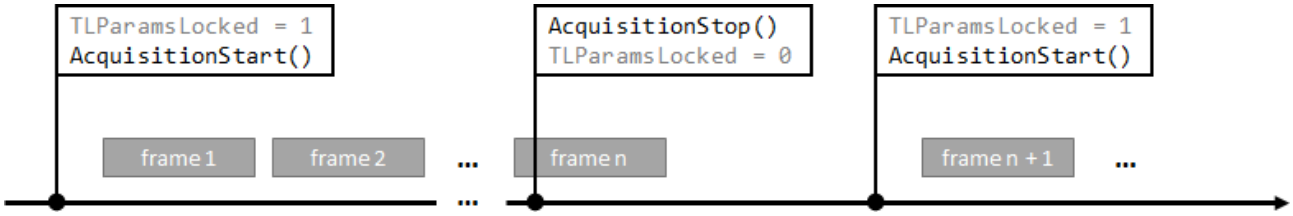
请参考相机手册，了解相机是否支持相应的功能。这同样适用于实际使用的功能名称。根据 GenICam 标准功能命名公约（2.4 版），使用功能名称的程序如下。

连续的图像采集

如果相机需要进行流式处理，即以均匀的时间间隔拍摄图像而不需要进一步触发，则需要进行以下配置：

- `AcquisitionMode = Continuous`
- `TriggerSelector = AcquisitionStart`
- `TriggerMode = Off`

然后，`AcquisitionStart`命令启动数据流，且`AcquisitionStop`命令会结束数据流。当前图像的记录已经完成，且图像已传输。

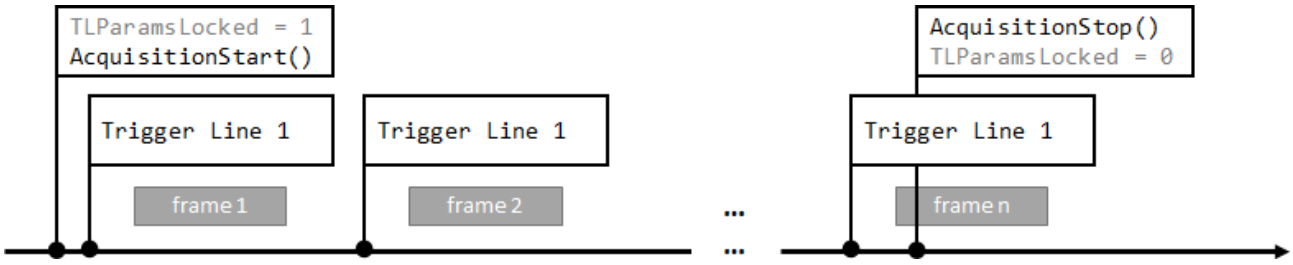


用于连续图像采集的单一图像触发器

如果相机要在触发信号后仅拍摄一幅图像，则需要进行以下配置：

- `AcquisitionMode = Continuous`
- `TriggerSelector = FrameStart`
- `TriggerMode = On`
- `TriggerSource = Line1 (exemplarisch)`

与其他情况一样，在`AcquisitionStart`命令之后，相机即可拍摄图像。每章图像都是在选定的信号源发出触发信号后记录。触发信号的详细信息在`TriggerActivation`中指定。

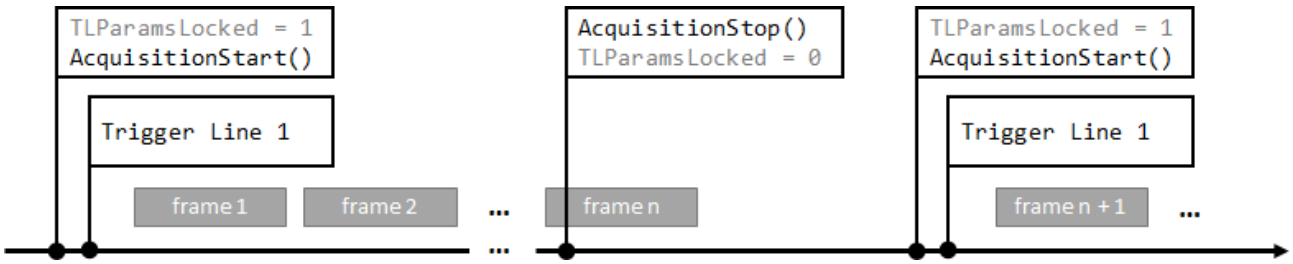


触发启动信号后连续图像采集

如果要求相机在触发信号后以均匀的时间间隔连续拍出图像，则需要进行以下配置：

- `AcquisitionMode = Continuous`
- `TriggerSelector = AcquisitionStart`
- `TriggerMode = On`
- `TriggerSource = Line1 (exemplarisch)`

在`AcquisitionStart`命令之后，相机即可捕捉和传输图像。然而，只有在触发信号之后才开始记录，在样本中是在线路 1 的硬件触发信号之后。数据流通过`AcquisitionStop`命令结束；但是，当前图像的记录仍将完成并传输。

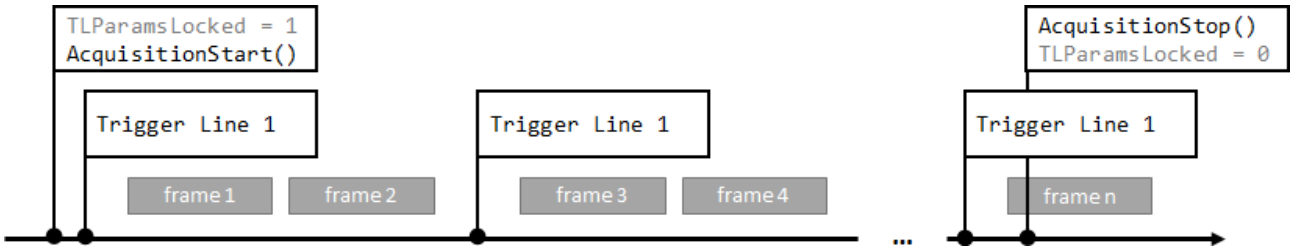


用于连续图像采集的多图像触发器

如果相机要在触发信号后记录一定数量的图像，需要进行以下配置。例如，该数字设置为 2：

- AcquisitionMode = Continuous
- AcquisitionBurstFrameCount = 2 (exemplarisch)
- TriggerSelector = FrameBurstStart
- TriggerMode = On
- TriggerSource = Line1 (exemplarisch)

与其他情况一样，在AcquisitionStart命令之后，相机即可拍摄图像。在选定源的触发信号后，记录定义的图像数量。触发信号的详细信息在TriggerActivation中指定。如果在图像采集/传输过程中发出AcquisitionStop命令，已经开始的图像会完成，但任何缺失的图像都不再创建。



8.4.3.2 多图像采集

默认情况下，图像采集和触发的设置处于互锁状态。下面的解释部分假设AcquisitionMode设置为MultiFrame。

其他情况描述如下：

- 连续的图像采集 [▶ 2832]
- 单一图像采集 [▶ 2836]

检查相机的功能

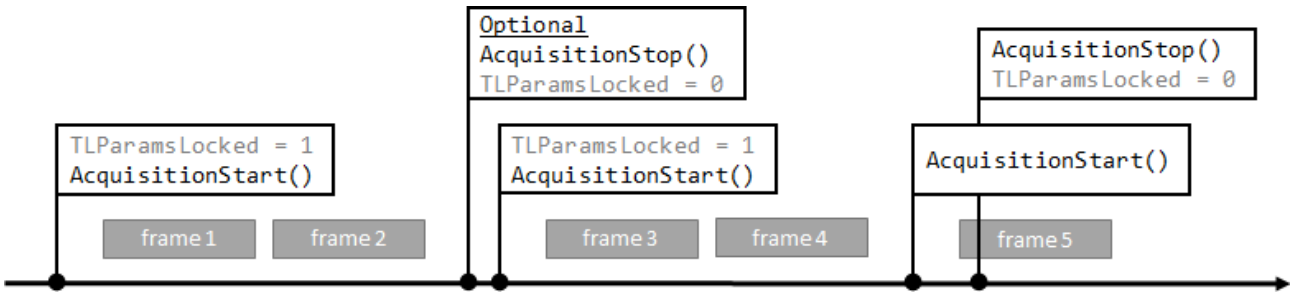
请参考相机手册，了解相机是否支持相应的功能。这同样适用于实际使用的功能名称。根据 GenICam 标准功能命名公约（2.4 版），使用功能名称的程序如下。

多图像采集

如果相机要在不使用触发器的情况下拍出一定数量的图像，需要进行以下配置：

- AcquisitionMode = MultiFrame
- AcquisitionFrameCount = 2 (example)
- TriggerSelector = AcquisitionStart
- TriggerMode = Off

AcquisitionStart命令发出后，数据流开始。在达到指定的帧数（AcquisitionFrameCount）时，将会停止。根据选择，然后发出AcquisitionStop命令，并取消锁定。这使得在下一个AcquisitionStart命令之前可以改变相机的特征。如果在图像采集/传输过程中发出AcquisitionStop命令，已经开始的图像会完成，但任何缺失的图像都不再创建。

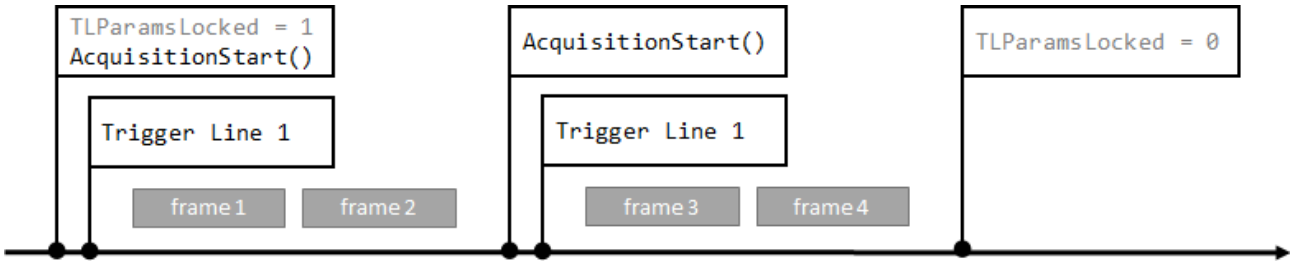


触发启动信号后的多图像采集

如果相机要在触发信号后拍出一定数量的图像，需要进行以下配置：

- AcquisitionMode = MultiFrame
- AcquisitionFrameCount = 2 (example)
- TriggerSelector = AcquisitionStart
- TriggerMode = On
- TriggerSource = Line1 (example)

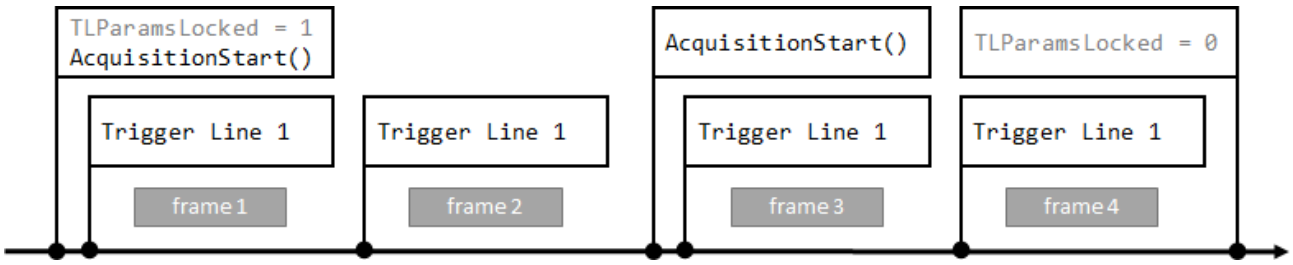
AcquisitionStart命令激活数据流。然而，在触发信号之后，才开始记录指定图像，在样本中为线路 1 上的硬件触发信号之后。与多图像采集（无触发器）一样，AcquisitionStop命令和取消锁定可以选择在达到指定的图像数量后进行。



带单一图像触发的多图像采集

配置：

- AcquisitionMode = MultiFrame
- AcquisitionFrameCount = 2 (example)
- TriggerSelector = FrameStart
- TriggerMode = On
- TriggerSource = Line1 (example)

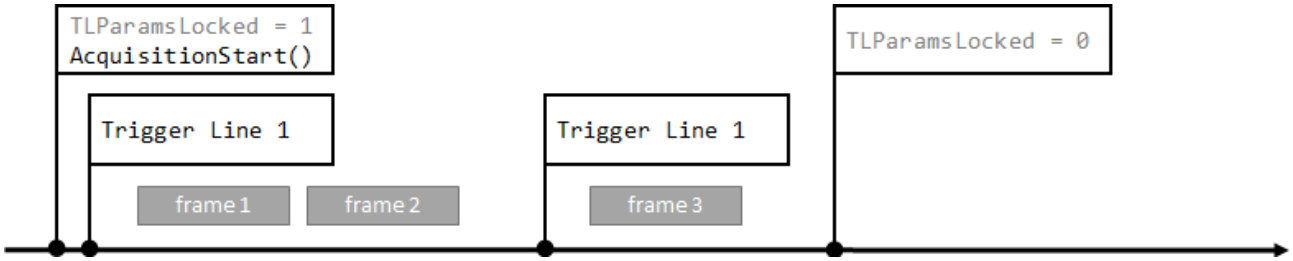


带多图像触发器的多图像采集

配置：

- AcquisitionMode = MultiFrame
- AcquisitionFrameCount = 3 (example)
- AcquisitionFrameBurstCount = 2 (example)

- TriggerSelector = FrameBurstStart
- TriggerMode = On
- TriggerSource = Line1 (example)



8.4.3.3 单一图像采集

默认情况下，图像采集和触发的设置处于互锁状态。下面的解释部分假设AcquisitionMode设置为SingleFrame。

其他情况描述如下：

- 连续的图像采集 [▶ 2832]
- 多图像采集 [▶ 2834]

检查相机的功能

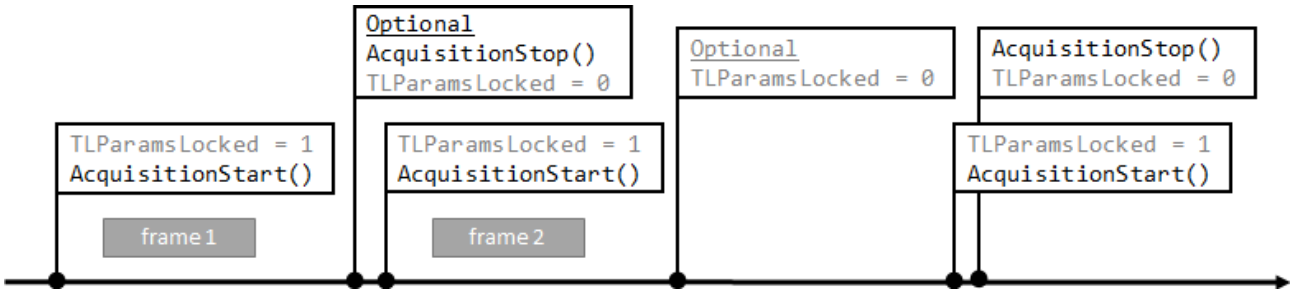
请参考相机手册，了解相机是否支持相应的功能。这同样适用于实际使用的功能名称。根据GenICam 标准功能命名公约（2.4 版），使用功能名称的程序如下。

单一图像采集

如果在每个AcquisitionStart命令后只记录一个图像，则需要以下配置：

- AcquisitionMode = SingleFrame
- TriggerSelector = AcquisitionStart
- TriggerMode = Off

在AcquisitionStart命令之后，仅拍摄一个图像。在两个AcquisitionStart命令之间，可以发出一个AcquisitionStop命令，并且可以取消锁定。解锁有利于在AcquisitionStart命令之间更改相机功能。如果在AcquisitionStart命令之后过早地发出AcquisitionStop命令，在图像采集尚未开始的情况下，将不捕捉图像。



带有图像采集触发器的单一图像采集
或带有单一图像触发器的单一图像采集

如果在需要触发信号的情况下每次AcquisitionStart命令后只记录一幅图像，可以使用以下配置：

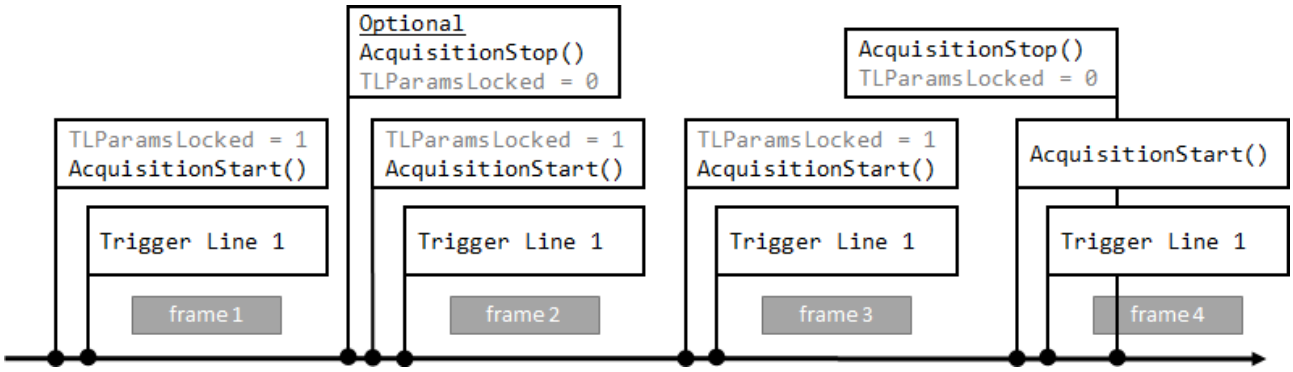
配置 - 带有图像采集触发器的单一图像采集

- AcquisitionMode = SingleFrame
- TriggerSelector = AcquisitionStart
- TriggerMode = On

- TriggerSource = Line1 (example)

配置 - 带单一图像触发器的单一图像采集

- AcquisitionMode = SingleFrame
- TriggerSelector = FrameStart
- TriggerMode = On
- TriggerSource = Line1 (example)



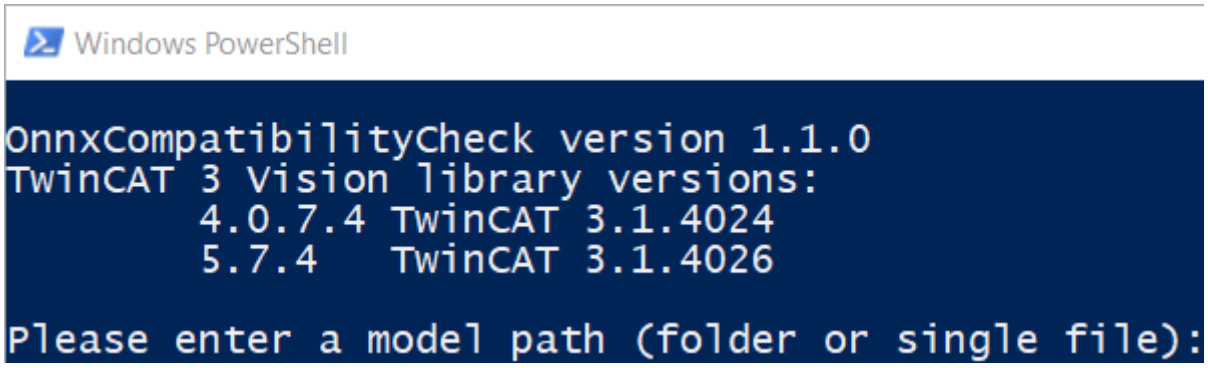
8.5 神经网络兼容性检查

该工具可检查 ONNX 模型与产品“TF7810 | TwinCAT 3 Vision Neural Network”功能组件的兼容性。“TF7810_NeuralNetworkSupplements.zip”可直接从倍福网站下载。

该软件包中包含可执行文件“OnnxCompatibilityCheck.exe”、用于简化使用的 Power Shell 脚本“CheckModelCompatibility.ps1”以及带有例程模型的文件夹“ONNX_Samples”。

执行 Power Shell 脚本

执行脚本后，首先会显示版本信息。首先是应用程序本身，然后是兼容性所涉及的 TwinCAT 3 Vision 库版本。作为用户，系统会要求您指定文件夹或模型路径。如果指定了包含多个 ONNX 模型的文件夹，则这些模型会按字母顺序进行测试。也可以明确指定一个模型，例如“C:\ONNX_Samples\LemonModel.onnx”。



成功的执行过程如下，例如使用 Lemon Sample 模型：

```
Please enter a model path (folder or single file): C:\ONNX_Samples\LemonModel.onnx

Successfully loaded 'LemonModel.onnx'
Model inputs: 1
  'input.1' [1, 3, 224, 224], TCVN_DL_4D_NCHW
Model Outputs: 1
  '367' [1, 3]
The model can be executed with TwinCAT Vision.
```

最后，执行会暂停，以便您查看输出结果。

Power Shell 脚本的选项

用户打开脚本进行编辑时，会在第 2 行和第 3 行发现参数 $modelPath 和 $inputShape。第一个参数 $modelPath 默认为空，因此会提示用户输入路径。或者，也可以在脚本中存储文件夹或模型路径。在这种情况下，不会提示用户输入，而是直接使用存储的路径。

CheckModelCompatibility.ps1

```
1  param(
2      [string]$modelPath = "", # folder or single file, e.g. "C:\temp\model.onnx"
3      [string]$inputShape = "skip" # "skip" if not required or model input shape, e.g. "1,3,224,224;1,3,32,32" for 2 inputs
4  )
```

默认情况下，第二个可选的 $inputShape 参数会存储“跳过”字样，因此不会出现该查询，应用程序也不会在没有该选项的情况下执行。也可以在此处为单个输入指定一个固定值，如“1 3 224 224”。如果有多个输入，必须用分号隔开，例如“1,3,224,224;1,3,32,32”表示两个输入。或者，也可以删除“跳过”一词，以便提示用户输入。也可以在执行脚本时按“Enter（回车键）”跳过“Model Input Shape（模型输入形状）”的提示。

应用“OnnxCompatibilityCheck.exe”

“OnnxCompatibilityCheck.exe”无需 Power Shell 脚本即可直接执行。例如，在 Windows 命令提示符中，成功测试 Lemon Sample 模型的结果如下所示：

Eingabeaufforderung

```
C:\TF7810_NeuralNetworkSupplements\OnnxCompatibilityCheck\x64>OnnxCompatibilityCheck.exe "C:\ONNX_Samples\LemonModel.onnx"

Successfully loaded 'LemonModel.onnx'
Model inputs: 1
  'input.1' [1, 3, 224, 224], TCVN_DL_4D_NCHW
Model Outputs: 1
  '367' [1, 3]
The model can be executed with TwinCAT Vision.
```

如果只指定了应用程序名称或添加了 -?、-h、-help，并按“Enter（回车键）”确认，则会显示以下有关应用程序的信息和注释。

Eingabeaufforderung

```
C:\TF7810_NeuralNetworkSupplements\OnnxCompatibilityCheck\x64>OnnxCompatibilityCheck.exe

OnnxCompatibilityCheck version 1.1.0
TwinCAT 3 Vision library versions:
       4.0.7.4 TwinCAT 3.1.4024
       5.7.4   TwinCAT 3.1.4026

Usage Information:
       -v, -version    show version information
       -?, -h, -help   show help and usage information

Usage:  OnnxCompatibilityCheck.exe <model.onnx> [ModelInputshape]

Arguments:
       <model.onnx> Path to the ONNX model file.
       [ModelInputShape] (Optional) Specify the input shape(s) of the model.
              Enter each shape as a comma-separated list of dimensions between 2 quotation marks.
              For example use the following command to test a model with a single input in NCHW layout:
              OnnxCompatibilityCheck.exe model.onnx "1,3,224,224"
              If the model has multiple inputs, provide each individual shape, separated by semicolon, between 2 quotation marks.
              To test a model with two inputs use:
              OnnxCompatibilityCheck.exe model.onnx "1,3,224,224;1,3,224,224"
              Leave this field empty to automatically use the input shape(s) from the provided ONNX model file.
```

结果

下文介绍了在不同情况下可以输出的结果：

- 开始执行时，系统会检查路径是否有效，文件是否为 ONNX 格式。如果不是这种情况，就会中止执行，并显示相应的消息。
- 如果 ONNX 模型加载成功，则显示为第一条信息，包括文件名。
- 接下来是从 ONNX 文件读取的模型输入层和输出层的信息：
  - 对于输入层，首先输出的是编号，然后是各层的列表，并在方括号中标明各自的名称和张量形式。显示相应维度的元素数量。

  - 如果输入层有两个以上维度，则还会输出用于从二维图像转换 (F_VN_ConvertDataLayout [▶ 184]) 为模型的相应输入格式的 ETcVnDataLayout [▶ 1549]。作为正确识别 `ETcVnElementType` 的前提条件，假定 N（样本数或批量大小）= 1，且通道数小于宽度或高度。
  - 以 Lemon Sample 模型的输入层为例，看起来就像这样：“模型输入：1 'input.1' [1, 3, 224, 224], TCVN_DL_4D_NCHW”。
  - 例如，对于带有多个输入层的模型来说，这看起来就像这样：“模型输入：3 'input.0' [1, 3, 224, 224]，TCVN_DL_4D_NCHW 'input1' [1, 5] 'input2' [1, 10]”
  - 对于输出层，首先输出的是编号，然后是各层名称和张量形状的列表。显示相应维度的元素数量。
  - 以 Lemon Sample 模型的输出层为例，它看起来是这样的：
    “模型输出：1
    层：'367' [1, 3]”
- 神经网络既可以为固定（静态）输入格式而设计，也可以处理多种不同的输入格式（动态）。对于大多数模型来说，输入层是固定的，张量的值存储在 ONNX 文件中。
  然后可以直接读取、显示和使用这些值进行测试。
- 如果模型支持多种输入 / 图像尺寸，在这种情况下就会丢失有关尺寸的信息，如 ONNX 文件中图像的宽度和高度。这被称为输入层的动态形状。如果在输入层中识别到这种形状，则会在层名称后显示“动态”字样，而不是张量的形状：
  - “'input.1' 动态 要测试执行情况，请提供可选的 [ModelInputshape]（对于 1 个输入，请使用 OnnxCompatibilityCheck.exe DynamicShape.onnx“1,3,224,224”）。”
- 为了能够测试带有动态输入层的模型，“Model Input Shape（模型输入形状）”的值也可以在调用应用程序时进行传输。调用和数值传输的示例也会输出，或在帮助中查看。
- 有些模型的输入层在 ONNX 文件中是固定的，但也可以用其他值来执行。如果用户掌握了这些信息或知道其他值，也可以在调用时使用“Model Input Shape（模型输入形状）”附加说明。
- 有关支持输入格式的信息，通常可在模型提供者的文档、模型来源或相关研究文章中找到。如果未明确指定这些信息，也可以通过检查模型结构或使用工具检查 ONNX 文件来获得。
- 如果模型无法执行，那么除基本信息外，还将尝试输出可能的原因。如果原因不明确，则不会显示其他信息。例如，可能的原因包括未知或不支持的运算符 / 数据类型或较新的 ONNX“操作集编号”。如果操作员被列为原因，这也可能是后续问题。因此，还应对上一层的操作员进行分析。

关于导出 PyTorch 模型的说明

对于 PyTorch 2.9 及更高版本，新的标准 ONNX 导出会创建不同的图形结构，从而导致不兼容问题。导出时，在 torch.onnx.export(...) 函数中设置参数 dynamo=False。这会强制使用经过验证的 TorchScript 导出，其结构可由 TwinCAT Vision 处理。然后，照常使用神经网络兼容性检查对模型进行检查。

9 附录

在这一章中，您会发现更多的信息，如常见问题解答、TwinCAT Vision 使用的路径概述以及用于纠正已知问题的故障排除章节。

9.1 常见问题解答

什么是 TwinCAT Vision？

TwinCAT Vision 是一款工业图像处理软件。您可以使用工业相机采集图像，并从中计算出关于所记录场景的信息。

常见的用例包括：

- 产品有无检测
- 目标识别
- 光学测量
- 编码读取

我可以免费测试 TwinCAT Vision 吗？

是的，您可以为运行时组件生成 7 天试用版授权。该Engineering软件免费使用。

我使用 TwinCAT Vision 是否需要 TwinCAT？

是的，TwinCAT Vision 已完全集成在 TwinCAT 中。在安装 TwinCAT Vision 之前，必须先安装兼容的 TwinCAT 版本。

TwinCAT Vision 可以使用哪些相机？

TwinCAT Vision 支持带有 GigE Vision 接口的相机。GigE Vision 相机通过以太网电缆连接到 IPC。

我可以同时使用多台相机吗？

是的，使用 TwinCAT GigE Vision Connector 可以操作多个相机，具体取决于授权。

我可以通过一台交换机连接多台相机吗？

是。请注意最大可用带宽（例如 1 GigE 为 125 MB/s，实际上更接近 110-115 MB/s）。

TwinCAT Vision 有示例吗？

是的，在示例 [▶ 2722]章节中。

TwinCAT Vision 函数是否有基准？

没有，因为视觉算法的执行时间在很大程度上取决于 IPC 的性能、图像大小、函数参数以及图像内容。因此，对算法的工作原理以及影响相应计算时间的因素有基本的了解是很有帮助的。

除了 TwinCAT Vision 之外，在 TwinCAT 中是否还可以使用其他图像处理软件作为替代方案？

使用 TC3 GigE Vision Connector，可以通过 GigE Vision 记录图像。然后可以通过各种方式进行图像处理：

- 通过 PLC 功能库 Tc3_Vision 使用 TwinCAT Vision API
- 使用自写入 TwinCAT PLC 或 C++ 算法
- 在 C++ 模块中使用 TwinCAT Vision API
- 使用 MATLAB® 和 Simulink®

- 通过 ADS 从其他应用程序检索图像数据

TwinCAT Vision 等图像处理软件与智能相机之间有什么区别？

图像处理软件，如 TwinCAT Vision：

- 从相机接收图像数据
- 在计算机上对其进行处理
- 向控制器发送处理后的信息
- TwinCAT Vision 的优势：控制与图像处理是同一个软件，在同一个系统中运行。因此，不需要额外的接口和协议。

智能相机：

- 具有内部处理器
- 直接在相机上执行图像处理操作
- 通过附加接口提供处理后的信息，例如合格/不合格信息或工件位置

TwinCAT Vision 可以在 GPU 上执行吗？

不可以，TwinCAT Vision 仅在 TwinCAT 实时环境中的 CPU 上运行。或者，TwinCAT Vision 提供了通过利用 TwinCAT 的多核功能（与工作任务 [▶ 66]交互）实现自动并行执行的选项。

9.2 见重要路径

Vision 数据路径

注意

数据相关路径

确保以下标准路径不受 Write filters 或类似程序的影响。否则可能导致数据丢失。如果手动调整了相应的路径设置，则设置的路径也会相应调整。

应用	路径
文件源图像的存储位置	C:\Users\Public\TcVision\FileSources\<UniqueId> （由文件源自动定义。每个项目的 <UniqueId> 都不同）。
相机数据流的存储位置	C:\Users\Public\TcVision\CameraStreams （可能根据记录/回放 [▶ 101]中的设置而有所不同）。
PLC 功能读取或写入的图像的存储位置。	C:\Users\Public\TcVision\Images （可能根据服务配置 [▶ 68]中的设置而有所不同）。
PLC 功能读取或写入的容器的存储位置。	C:\Users\Public\TcVision\Containers （可能根据服务配置 [▶ 68]中的设置而有所不同）。
PLC 功能读取或写入的 ML 模型的存储位置。	C:\Users\Public\TcVision\MlModels （可能根据服务配置 [▶ 68]中的设置而有所不同）。
PLC 功能读取或写入的标定结果的存储位置。	C:\Users\Public\TcVision\CalibrationResults （可能根据服务配置 [▶ 68]中的设置而有所不同）。
单个标定模板的存储位置	C:\Users\Public\TcVision\CalibrationPattern

日志文件路径

日志文件	路径
TwinCAT Vision 安装日志	C:\Users\<User>\ TF7xxx-Vision.exe.log Build 4026 及以上版本：

日志文件	路径
(所有其他 TwinCAT 安装日志也位于此文件夹中)	C:\ProgramData\Beckhoff\TcPkg\logs\ 单个产品日志文件位于 MSI 子文件夹中。
TwinCAT Vision Service 日志	C:\ProgramData\Beckhoff\TcVnService\TcVnService.log
一般 TwinCAT Vision 日志	C:\ProgramData\Beckhoff\Vision （可能因 记录 [▶ 70] 中的设置而异）
标定助手 [▶ 106]的结果图像用于分析标定模板上特征点的识别	C: \ProgramData\Beckhoff\Vision_CalibrationAssistantOutput

9.3 故障排除

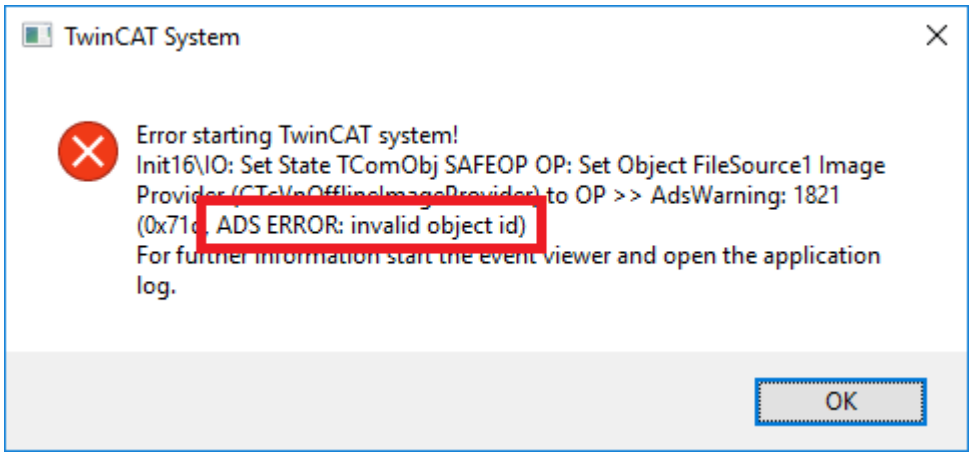
下文介绍了已知的错误情况以及相应的解决方案。如果您能将问题追溯至某个特定的函数或功能块，请首先阅读 API 参考 [▶ 132]中的相应条目！
如果无法独立解决问题，请使用 Support Information Report [▶ 2855]工具收集产品信息，并联系倍福技术支持部门 [▶ 2861]。

9.3.1 TwinCAT 系统启动

下面解释了 TwinCAT 系统无法启动的情况下的解决方案。

9.3.1.1 TwinCAT 系统不能启动

如果 TwinCAT 系统不能启动，经常出现以下信息。根据带有红色边框的错误代码，可以给出纠正问题的指示。在下文中，提供了有关 TwinCAT Vision 应用程序的常见错误代码的信息。



无效对象 ID

错误信息	ADS ERROR：无效对象 id
解决方案 1	**未链接任务** 1. 检查是否已经为所使用的每个视觉设备创建了相应的任务，并添加缺少的任务。 ◦ 检查是否为每个文件源控制创建了图像采集的任务（默认周期为 10 ms）。 ◦ 检查每个 GigE Vision 相机是否与图像采集任务相关联（默认周期为 1 ms）。如果有 Ads Communicator，也与任务相关联（默认周期为 10 ms）。 2. 这些任务必须分配给视觉设备的相应 TcCom 模块。检查作业，必要时予以纠正。 ◦ 检查文件源的图像提供者任务是否与文件源控制的图像提供者相连。 ◦ 检查相机的图像采集任务是否与相机的图像采集和图像采集模拟相联系。 ◦ 将 Ads Communicator 任务与相机的 Ads Communicator 联系起来。 ◦ 请勿将任务分配给相机的图像提供者。

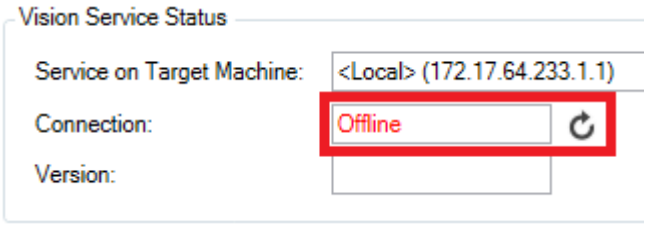
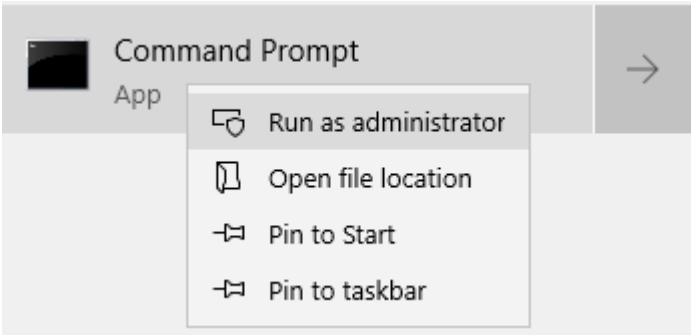
解决方案 2	**网络适配器无效** 网络适配器未调整，例如在系统改变之后。根据前的目标系统，调整网络适配器。 在相关网络适配器中，打开**适配器**选项卡，并通过**搜索**选择新的兼容适配器。
解决方案 3	**网络适配器已删除** 网络适配器已删除，例如在假设相机的模拟不需要的情况下。然而，每个相机对象始终需要网络适配器；如有必要，可以停用但不能删除。 通过 IpStack 创建新的网络适配器，并在相机采集对象的**接口指针**选项卡上链接 IpStack。

关于连接 Vision TcCOM 对象的更多信息，可查看相机 [▶ 120]和文件源 [▶ 126]的相应章节。

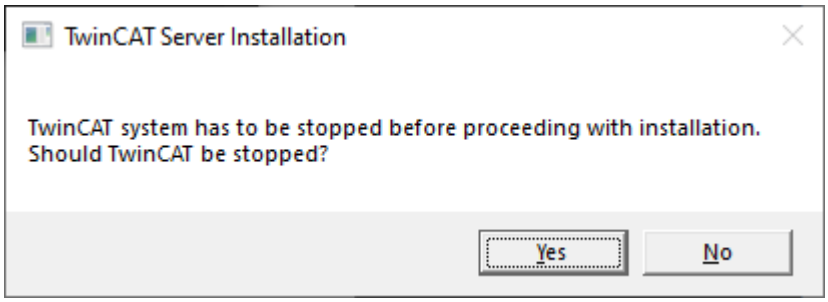
一般性 ADS 错误

错误信息	ADS ERROR：一般 ADS 错误
解决方案 1	**相机比配置的多** 网络适配器的默认设置最多允许 2 台相机同时操作。如果想要在同一适配器上操作更多的相机，请将初始参数`ipMaxReceiver`和`UdpMaxReceiver`增加到所连接相机数量的两倍（见网络适配器的配置 [▶ 56]）。

9.3.1.2 视觉服务不启动

情况	TwinCAT Vision 服务没有启动。这可以通过视觉节点的**服务配置**选项卡上的连接状态“离线”查看。 
解决方案	如果 TwinCAT 版本服务只启动失败一次或失败，可以通过 TwinCAT 重启来重新激活（点击 ）。 然而，如果它通常无法启动（在系统启动后），则必须将该服务重新注册为 TwinCAT 服务。为此，请遵循以下步骤： 1. 将 TwinCAT 切换到配置模式  2. 以管理员身份打开提示。  3. 通过“`cd C:\TwinCAT\3.1\Components\Vision\TcVision`”切换到相应的文件夹。 4. 执行“`TcVnService.exe /tc3serverreg`”命令，以启动视觉服务。

5. 通过点击**是**，确认随后的对话框。

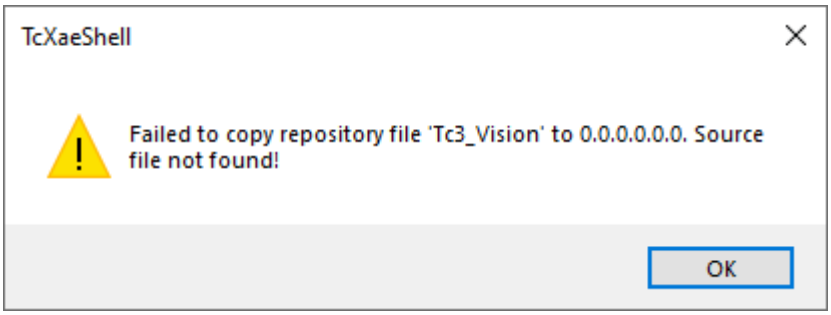
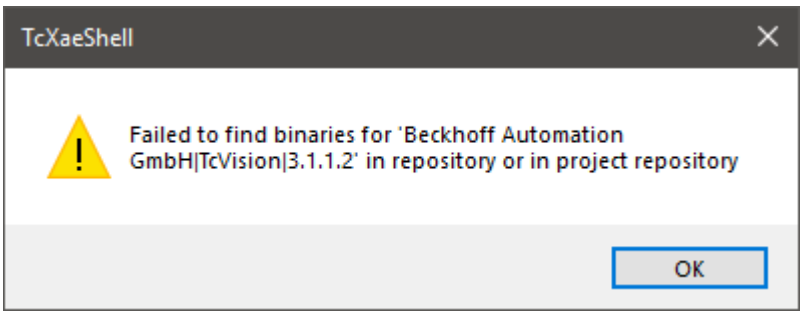


⇨ 视觉服务启动。

TwinCAT 必须处于配置模式，才能启动视觉服务。因此，如果需要重新启动视觉服务，必须重新启动机器。

如果需要手动停止视觉服务，可以使用命令“`TcVnService.exe /unregtc3server`”。

9.3.1.3 无法找到/加载二进制文件或存储库

情况	在配置激活时，会出现两个信息中的一个。根据原因和安装版本，其他文件名或版本号可能出现在信息中。 该信息的原因在于，项目中使用的版本与系统中安装的版本不一致。  
解决方案 1	**检查视觉库** 检查项目中包含的 Tc3_Vision 库的版本。 如果版本与工程系统 C:\TwinCAT\3.1\Repository\Beckhoff Automation GmbH\Tc3_Vision 文件夹中的版本不一致，则更新项目中的库，或安装另一个安装版本。
解决方案 2	**检查相机或 FileSorce 的 TMC 版本** 在**系统 >TcCOM 对象 >项目对象**下检查项目所用列出对象的 TMC 版本。如果版本以黄色突出显示，则说明系统中不存在指定的版本或不匹配。 现在，更新相应的 TMC 版本（通过版本的下拉菜单选择版本，然后右击第一列中的一个对象 ID，并选择**通过更改版本重新加载 TMI/TMC 描述**）到可用版本或安装另一个设置版本。
解决方案 3	**视觉组件不可用**

	当前系统中缺少相应的存储库文件，因为安装时没有选择视觉组件。基本上，工程和目标系统的版本必须匹配。因此，如果系统不同，请检查这两个装置。在纯运行时系统 XAR 上，仅视觉服务可以安装，因为所有其他文件都由工程系统 XAE 传输。 安装缺失的视觉组件或使用修复选项运行设置。

9.3.2 PLC 运行时环境

下面解释了在 PLC 运行时发生错误的情况的解决方案。

9.3.2.1 视觉设备处于 ERROR 状态

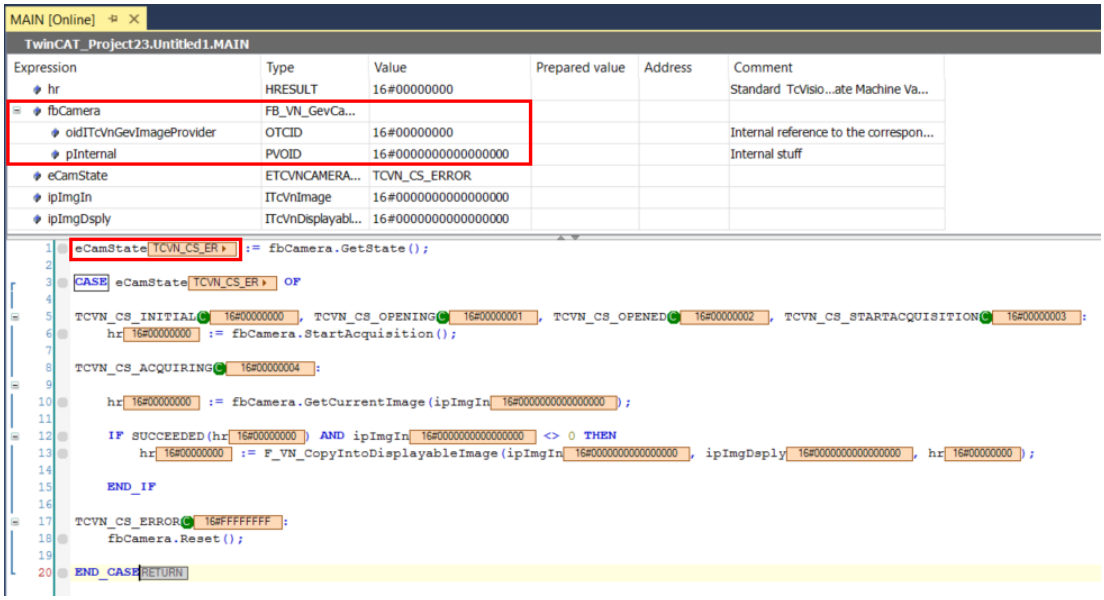
相机功能块在 ERROR 和 INITIAL 状态之间切换

情况	功能块FB_VN_Gev_CameraControl [▶ 1609]或FB_VN_SimpleCameraControl [▶ 1630]的实例在调用方法GetState时交替返回TCVN_CS_ERROR和TCVN_CS_INITIAL。
	在TCVN_CS_ERROR状态下，功能块的 Reset() 方法已成功执行，且相机实例变为TCVN_CS_INITIAL状态。如果相机一直处于TCVN_CS_ERROR状态，请参见设备长期处于 ERROR 状态 [▶ 2845]。
	`eState := fbCamera.GetState()`
	`IF eState = TCVN_CS_ERROR THEN`
	`  fbCamera.Reset();`
	`END_IF`
	在 TCVN_CS_INITIAL 状态下，则调用了 OpenCamera() 或 StartAcquisition() 的功能块方法。调用失败，且实例返回状态 TCVN_CS_ERROR。
	`eState := fbCamera.GetState()`
	`IF eState = TCVN_CS_INITIAL THEN`
	`  fbCamera.OpenCamera();`
	`END_IF`
	或
	`eState := fbCamera.GetState()`
	`IF eState = TCVN_CS_INITIAL THEN`
	`  fbCamera.StartAcquisition();`
	`END_IF`

解决方案 1	检查相机的物理连接和相机的电源。
解决方案 2	检查相机连接的设置（相机的 IP 地址、网络端口）。

功能块长期处于 ERROR 状态

情况	功能块的实例FB_VN_GevCameraControl、FB_VN_SimpleCameraControl或FB_VN_FileSourceControl长期返回状态TCVN_CS_ERROR。即使在调用功能块方法 Reset 时，也没有改变状态为TCVN_CS_INITIAL（如果有，见设备在 ERROR 和 INITIAL 状态之间切换 [▶ 2845]）。 `fbCamera.Reset();`

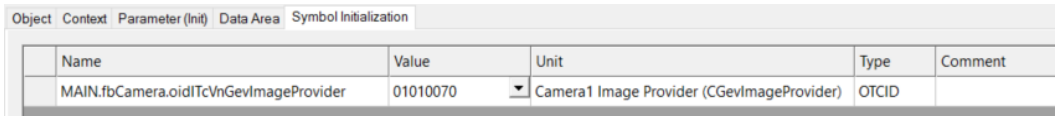


解决方案 1

符号实例必须链接到图像提供者 - FB_VN_GevCameraControl、FB_VN_SimpleCameraControl和FB_VN_FileSourceControl的每个实例必须链接到视觉节点中创建的相机或文件源控制。

请按以下步骤进行：

1. 进行 PLC 项目的成功重建。
2. 然后双击 PLC 项目的实例。
3. 在**符号初始化**选项卡下，将相应相机的图像提供者或相应的文件源控制分配给 FB_VN_GevCameraControl的实例。
4. 重新激活配置。



FB_VN_GevCameraControl 只能链接到相机的图像提供者。

FB_VN_FileSourceControl 只能链接到文件源控制的图像提供者。

FB_VN_SimpleCameraControl 可以链接到相机的图像提供者或文件源控制的图像提供者。

解决方案 2

FB_VN_GevCameraControl或FB_VN_SimpleCameraControl或FB_VN_FileSourceControl的实例被指定为相机或文件源控制的图像提供者，但设备却长期处于错误状态。

检查FB_VN_GevCameraControl或FB_VN_SimpleCameraControl或FB_VN_FileSourceControl的实例的方法调用。许多方法仅可在某种状态下被调用。关于视觉设备状态的详细描述，可查看图像采集的功能块 [▶ 1599]部分。

- 在所有状态下
  - GetState
  - GetCalibPatternRef （来自 FB_VN_GevCameraControl）
  - GetCameraMatrix （来自 FB_VN_GevCameraControl）
  - GetDistortionCoefficients （来自 FB_VN_GevCameraControl）
  - GetRotationMatrix （来自 FB_VN_GevCameraControl）
  - GetTranslationVector （来自 FB_VN_GevCameraControl）
  - SetCameraMatrix （来自 FB_VN_GevCameraControl）
  - SetDistortionCoefficients （来自 FB_VN_GevCameraControl）

	◦ SetRotationMatrix （来自 FB_VN_GevCameraControl） ◦ SetTranslationVector （来自 FB_VN_GevCameraControl） • 在 INITIAL 状态和 INITIALIZING 状态下 ◦ InitializeCamera（来自 FB_VN_GevCameraControl） • 在 INITIAL 状态和 OPENING 状态下 ◦ OpenCamera（来自 FB_VN_GevCameraControl） ◦ StartAcquisition • 在 OPENED 状态和 STARTACQUISITION 状态下 ◦ StartAcquisition • 在 OPENED 状态和 CLOSING 状态下 ◦ CloseCamera （来自 FB_VN_GevCameraControl） • 在 ACQUIRING 状态下 ◦ GetCurrentImage ◦ GetCurrentImageUndistorted（来自 FB_VN_GevCameraControl） ◦ GetCurrentImageWithGvspInfo（来自 FB_VN_GevCameraControl） ◦ GetCurrentImageAndFileName（来自 FB_VN_FileSourceControl） ◦ ClearImageQueue（来自 FB_VN_GevCameraControl） ◦ GetLastImageFromQueue（来自 FB_VN_GevCameraControl） ◦ GetOmittedImagesNum（来自 FB_VN_GevCameraControl） • 在 ACQUIRING 状态和 TRIGGERING 状态下 ◦ TriggerImage ◦ TriggerImageExp（来自 FB_VN_FileSourceControl） ◦ TriggerImageByName（来自 FB_VN_FileSourceControl） • 在 ACQUIRING 状态和 STOPACQUISITION 状态下 ◦ StopAcquisition • 在 ERROR 状态下 ◦ Reset

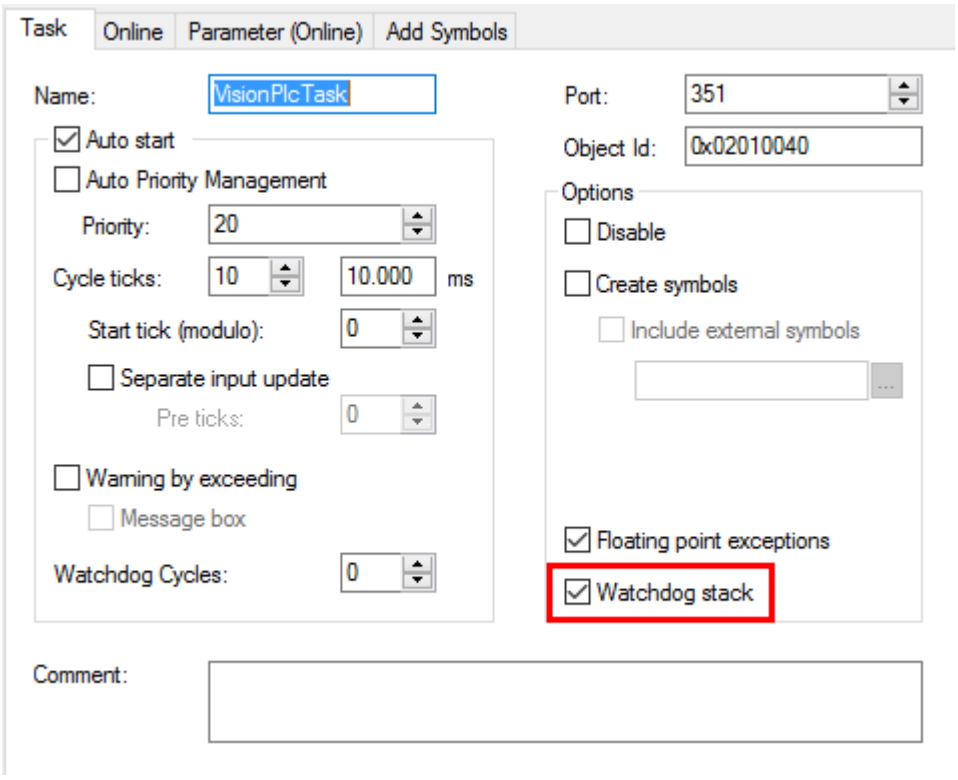
9.3.2.2 重新启动后找不到相机

情况	应用程序完全照常工作。重新启动机器后，无法与相机建立连接。
解决方案 1	IP 地址已经改变。使用相机的配置助手 [▶ 77]中的 Force-IP 选项。

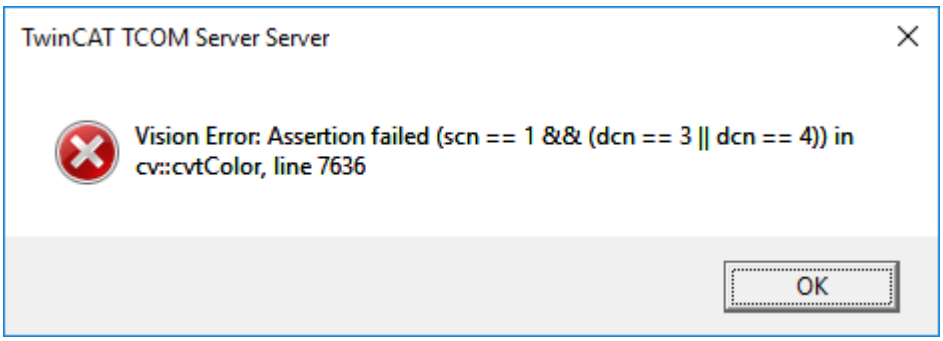
9.3.2.3 启动看门狗时出错

情况	函数F_VN_StartAbsWatchdog [▶ 994]或F_VN_StartRelWatchdog [▶ 996]长期返回错误代码16#71A（NOINTERFACE）。
	
	`hrWD 16#9811071A := F_VN_StartRelWatchdog(5000, S_OK);`
	`hr 16#9811070C := F_VN_DetectBlobs(ipImgIn 16#FFFF900DA079D3`
	`hrWD 16#9811071A := F_VN_StopWatchdog(hrWD 16#9811071A, tRest`

解决方案	在相应的 PLC 任务中激活看门狗堆栈，然后重新激活配置 。



9.3.2.4 视觉错误：断言失败

错误信息	在执行视觉应用程序的过程中，出现了类似以下的错误信息。同时，TwinCAT 进入 EXCEPTION 状态。  这条信息表明，在执行视觉函数的过程中，存在一个无法拦截的冲突。这基本上是指输入数据的正确格式，且应仅在函数没有按照 API 参考使用时才会发生。在“VisionError: Assertion failed”背后的错误规范可能因用例不同而不同。
解决方案与 F_VN_ConvertColorSpace [▶ 1248]	图像可以是单通道，也可以是多通道图像。对于某些函数的调用，预计使用具有定义通道数量的图像。例如，对于函数`F_VN_ConvertColorSpace`，其中枚举`ETcVnColorSpaceTransform`用于指定输入图像的格式。如果工程部登录，相应的函数调用会在 PLC 代码中被标记出来。函数`F_VN_GetPixelFormat`（返回实际通道数量）可以帮助分析为什么实际的通道数量与预期的数量不同。 典型原因： • 文件源控制 在文件源中，设置了与预期不同的图像格式，或者图像格式被设置为“来自文件的原始格式”，而实际图像格式与预期不同。例如，看起来是单色的图像并不总是作为单色图像（单通道图像）存储。 • 相机 对于许多相机，可以在配置中设置像素格式。根据所选择的格式，图像通道的数量也会改变。

	• 以前的处理步骤 以前的函数调用可能已经改变了图像，且因此也改变了通道的数量。

9.3.2.5 错误：SSE 无效操作

错误信息	得到错误信息**SSE 无效操作**。
解决方案	如果在执行的任务上激活了**浮点异常**处理，将以不必要的方式发生这个错误。 按照CPU 内核和任务 [▶ 63]部分的说明，停用选项**浮点异常**。

9.3.2.6 文件来源

错误状态

错误模式	当 PLC 运行时，文件源不断切换到错误状态。
解决方案 1	**内存太少** 路由器内存设置太小。因此，文件源不能将任何图像加载到 PLC 中，并在试图加载时进入错误状态。扩大路由器的内存。

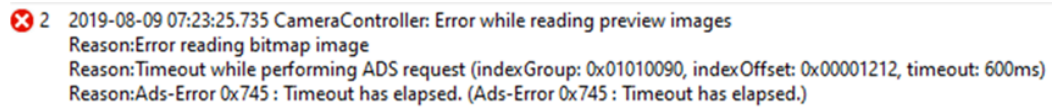
不在 PLC 中的 FS 控制的图像

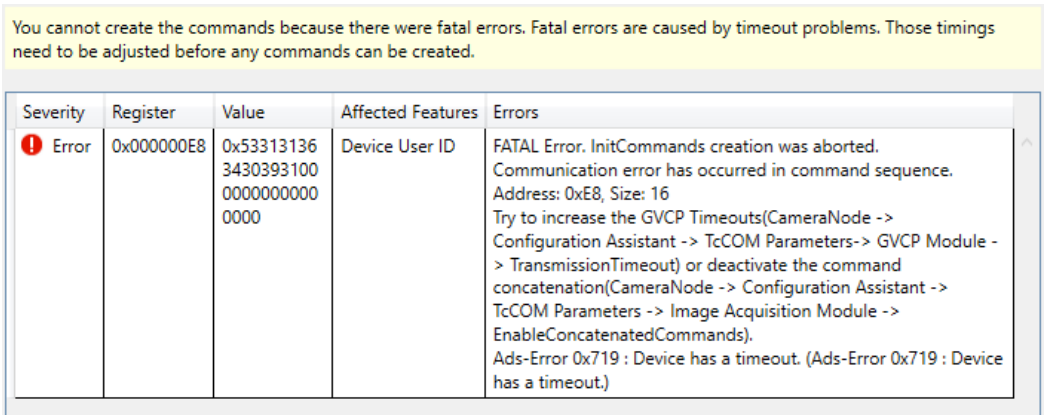
错误模式	在文件源控制中选择了图像，而 PLC 中的文件源功能块处于`ACQUISITION`状态。尽管如此，仍无法通过`GetCurrentImage`检索到任何图像。
解决方案 1	**未设置 BaseDir** 在 TwinCAT 激活后，首次启动了**从目标读取**按钮。在这种情况下，读取文件源图像的路径还没有转移到运行中的 TcCOM 对象。因此，即使在文件源控制中选择了图像，也不能加载任何图像。在按下**从目标读取**后，重新激活 TwinCAT。
解决方案 2	**触发模式** 在文件源控制中选择了触发模式，但在 PLC 中只调用了方法`GetCurrentImage`，没有调用`TriggerImage`。通过调用 PLC 中的方法`TriggerImage`，触发一个图像，或者在文件源控制中停用触发模式。

9.3.3 助手

下面解释了助手操作出现问题的情况下的解决方案。

9.3.3.1 超时

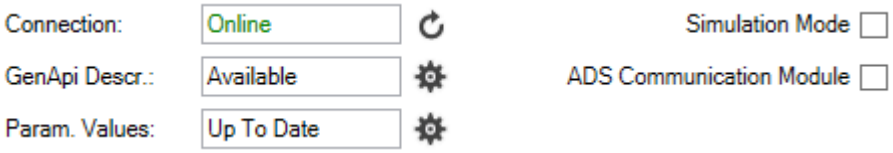
情况	在与相机通信过程中发生超时错误。正如以下案例所述，这可能发生在不同的地方。 在开发环境中的图像显示期间：  2 2019-08-09 07:23:25.735 CameraController: Error while reading preview images Reason:Error reading bitmap image Reason:Timeout while performing ADS request (indexGroup: 0x01010090, indexOffset: 0x00001212, timeout: 600ms) Reason:Ads-Error 0x745 : Timeout has elapsed. (Ads-Error 0x745 : Timeout has elapsed.) 在创建初始化命令时：



当使用 ForceIP 命令时：



在相机对象中建立连接时：



解决方案	在超时错误的情况下，两种不同类型的超时基本上相关：一种是传输超时，它限制了图像采集模块和相机之间的通信。另一种是 ADS 通信超时，它限制了用户界面和图像采集模块之间的通信。 在配置树 [▶ 81]中调整 GVCP 模块的 TcCOM 参数中的传输超时。同时调整 MaxTimeouts 的数量。 如有必要，确保在设置 [▶ 87]中调整 ADS 通信超时，因为这应该大于 MaxTimeouts 和传输超时的乘积。 逐步增加次数，必要时增加尝试次数，直到不再出现错误信息。然而，确保不要把时间和尝试次数设置得过高，否则会出现非常长的等待时间。

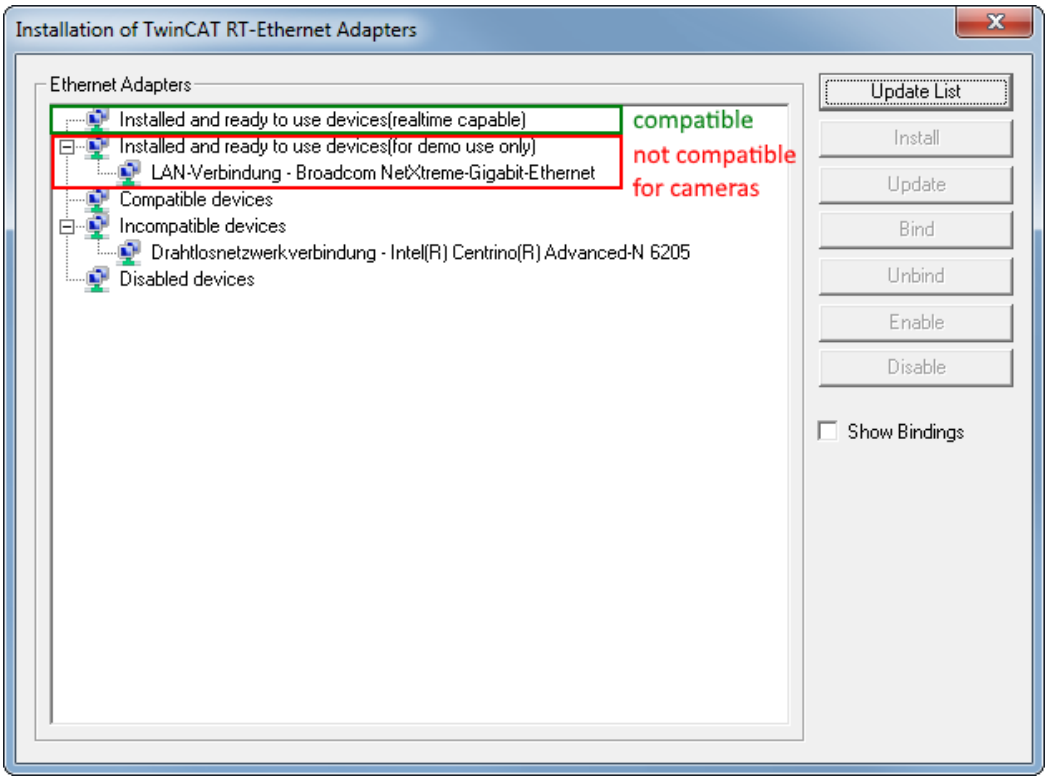
9.3.3.2 图像显示停滞

错误模式	ADS 图像查看和相机助手中的图像显示偶尔会停顿。
解决方案 1	**远程桌面协议** 在个别情况下，远程桌面协议和网卡可能与 TwinCAT Vision 开发环境或图像显示工具的传输不匹配。 请在本地操作并连接 TwinCAT 中的目标设备，或者直接在目标计算机上操作。

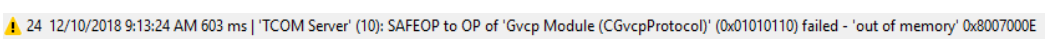
错误模式	部分图像无法显示在 ADS 图像查看和相机助手中。
解决方案 1	**减少数据量**

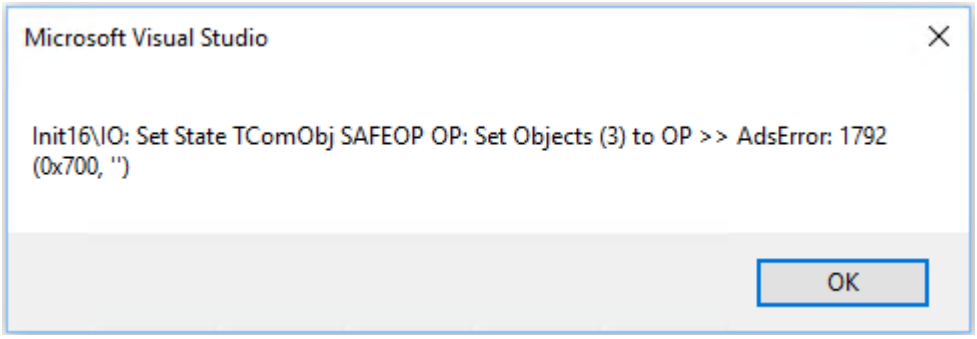
	由于图像是由 ADS 从控制器中检索以便显示，因此可能无法检索到所有的图像数据，特别是在大图像、高帧率或几个 ADS 图像查看窗口的情况下。 因此，使用 ROI 减少同时显示的图像数量，并降低每张图像的帧率或数据量。

9.3.3.3 未通过设备发现找到相机

情况	你正在创建一个 GigE Vision 相机对象。您已经成功设置了一个带有 IP 堆栈的 TwinCAT RT-Ethernet 适配器。 然而，“设备发现”对话框仍然为空，且没有错误信息。
解决方案 1	**选择了错误的网络适配器**
解决方案 2	**没有 GigE Vision 相机** 具有千兆以太网接口的相机不一定要“讲”GigE Vision。请密切注意制造商对传输协议的说明。
解决方案 3	**不兼容的网络适配器** 使用的网络适配器... 

9.3.3.4 加载相机对象时出错

错误信息	 在配置模式下，当重新加载相机模块或需要通过选择目标相机对话框中的发现设备按钮选择相机时，会出现以下 Visual Studio 信息。

	
解决方案	连接的相机比配置的要多。 网络适配器的默认设置最多允许 2 台相机同时操作。如果想要操作更多相机，请将网络适配器的初始参数`ipMaxReceiver`和`UdpMaxReceiver`增加到所连接相机数量的两倍（见网络适配器的配置）。

9.3.3.5 警告：不允许读取/写入

警告	 在连接相机时可能会出现这个警告。
解决方案	该警告表明，TwinCAT Vision 向相机发送的写或读命令不被允许。如果 GenAPI 描述中对寄存器的访问权限的定义偏离了相机的实际行为，就会发生这种情况。通常情况下，这个相机寄存器对于用户来说并不重要。如需确定这与哪个寄存器有关，可以在配置助手 [▶ 77]中搜索显示的寄存器地址。

9.3.4 TwinCAT HMI 软件包

对于 TwinCAT 人机界面软件包不能按预期工作的情况，下面将说明解决方案。

9.3.4.1 服务器扩展

不显示图像

情况	尽管安装了服务器扩展 [▶ 2660]，并且适当地链接了图像符号，但来自 PLC 的图像没有显示在 HMI 中。
解决方案 1	**缺少管理员权限** 您选择了“数据流”作为图像格式，但您还没有以管理员身份启动 HMI 服务器。选择不同的图像格式（BMP、PNG 或 JPG），或以管理员身份启动 HMI 服务器。

已发布的页面中缺少图片

情况	TcHmiVision [▶ 2660]服务器扩展已经成功集成到项目中。图像显示在实时视图或工程服务器提供的 HMI 上。该项目也可以在独立的 TcHmi 服务器上无错误地发布。然而，在公布的 HMI 上没有显示图像。
解决方案 1	**缺少目标许可证** TC3 HMI 服务器许可证中包括一个目标许可证。如果 HMI 服务器需要与多个目标（ADS 连接）进行通信，则需要额外的许可。
解决方案 2	**运行时环境的配置错误** 工程中使用的默认配置可能与公布的配置不同（不同的 AmsNetIds）。因此，请检查运行时环境的配置，并在必要时进行调整。
解决方案 3	**用户权利受限**

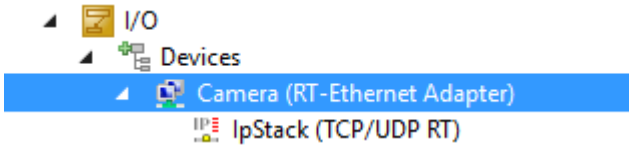
	使用 TwinCAT HMI 用户管理，而当前登录的用户没有权限查看图像或接收扩展数据。检查你的用户组的授权。
解决方案 4	**限制数据量或刷新时间间隔** 由于数据量可能相当大，特别是在几个图像必须同时由 ADS 从控制器检索并由 HMI 服务器转换的情况下，这可能导致瓶颈。有几个选项可以补救：降低控制器中的图像大小，增加单个图像的刷新率，使用不同的刷新率或使用压缩的图像格式。

9.3.5 相机通信

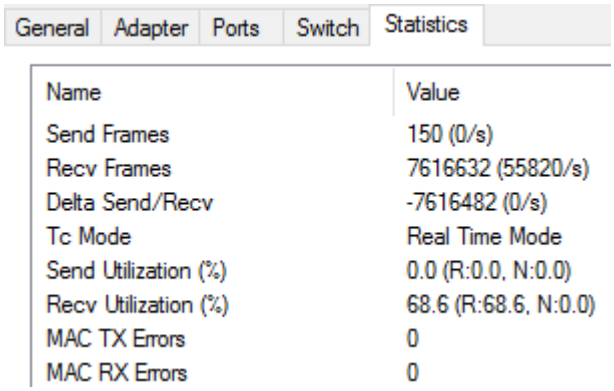
如果出现连接方面的问题（特别是发送或接收帧异常），以及参数设置或图像缺失方面的问题，您可以在网卡中以及相机对象的图像采集模块下找到以下诊断信息。

RT 以太网适配器中的诊断选项

1. 首先选择用于相应相机通信的 RT 以太网适配器。



2. 双击将其打开，并转到 **Statistics（统计信息）**选项卡。



⇨ 在这里，您可以看到帧的发送和接收是否正常，或者是否出现错误。此外，您还可以在 **Recv Utilization（接收利用率）**中查看连接利用率（以百分比为单位）。如果利用率非常高，超过 98%，则表明存在过载情况，尤其是在同一个适配器上运行多台相机时。

更多诊断信息可以在 IpStack 中找到。

1. 为此，首先双击打开底层的 IpStack。
2. 前往 **Parameter (Online)（参数（在线））**选项卡。

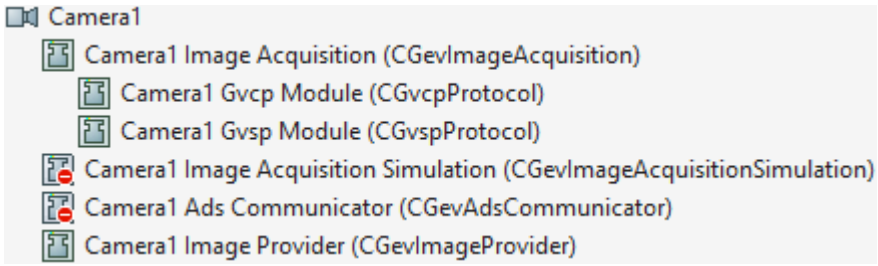
⇨ 在这里，您可以查看各个类别的计数器数值。

Object | Context | Parameter (Init) | Parameter (Online) | Interfaces | Interface Pointer

	Name	Online	CS	Type	PTCID
-	IpStackDiagnosis	...	☐		0x03011108
	.ip.nSendCnt	1372		UDINT	
	.ip.nSendFailCnt	0		UDINT	
	.ip.nRecvCnt	392048		UDINT	
	.ip.nRecvFailCnt	0		UDINT	
	.arpRequest.nSendCnt	1		UDINT	
	.arpRequest.nSendFailCnt	0		UDINT	
	.arpRequest.nRecvCnt	0		UDINT	
	.arpRequest.nRecvFailCnt	0		UDINT	
	.arpReply.nSendCnt	0		UDINT	
	.arpReply.nSendFailCnt	0		UDINT	
	.arpReply.nRecvCnt	1		UDINT	
	.arpReply.nRecvFailCnt	0		UDINT	
	.pingRequest.nSendCnt	0		UDINT	
	.pingRequest.nSendFailCnt	0		UDINT	
	.pingRequest.nRecvCnt	0		UDINT	
	.pingRequest.nRecvFailCnt	0		UDINT	
	.pingReply.nSendCnt	0		UDINT	
	.pingReply.nSendFailCnt	0		UDINT	
	.pingReply.nRecvCnt	0		UDINT	
	.pingReply.nRecvFailCnt	0		UDINT	
	.nLinkStatusChangedCnt	1		UDINT	
	.nAllocFailCnt	0		UDINT	
	.nArpTimeoutFrames	0		UDINT	
	.nDroppedFrames	0		UDINT	

图像采集模块中的诊断选项

1. 首先选择相关的相机对象。



2. 打开 Gvcp 模块下的 **Parameters (online)（参数（在线））**选项卡。

⇨ 在这里，您可以获取有关控制协议通信的信息。这包括heartbeat、寄存器的读取和写入以及**开始采集**等命令。
如果一切正常，应该只有两个 nSuccess 值在递增。如果发生超时，请参见本 超时 [▶ 2849] 章节了解相关解决措施。对于其他问题，则需要使用 Wireshark® 进行更深入的诊断。

Object | Context | Parameter (Init) | Parameter (Online) | Interfaces

	Name	Online	PV	CS	Unit	Type	PTCID	Comment
-	GvcpProtocolDiagnosis	...					0x00010001	Contains statistics for GVCP protocol diagnosis.
	.stSend.nSuccess	1242				ULINT		Number of successfully sent packets
	.stSend.nError	0				ULINT		Number of packets causing errors during transmission
	.stSend.nTimeout	0				ULINT		Number of timeouts that have occurred
	.stRecv.nSuccess	1189				ULINT		Number of successfully received packets
	.stRecv.nError	0				ULINT		Number of received packets signalling or causing errors
	.stRecv.nIgnored	0				ULINT		Number of ignored packets

流协议的诊断信息可以在 **Parameter (online)（参数（在线））** 选项卡下的 Gvsp 模块下找到。这些计数器仅涉及传输的图像数据，分为单个报文/数据包和完整图像/数据块。

如果一切正常，应该只有 nSuccess 和 nCompletedBlocks 值在递增。

nResendRequests 已经表明存在问题，但在此情况下，这些问题可能仍能得到解决，因为所有其他计数器仍然为零。如果发生超时，应在 GVSP Module settings（GVSP 模块设置） [▶ 100]下的配置助手中增加 BlockTimeout。nBlockIdOverflows 计数递增是正常现象，并不表示错误。由于数值范围的原因，该计数器最终会溢出，或者在重启相机时被重置。

Object | Context | Parameter (Init) | Parameter (Online) | Interfaces | Interface Pointer

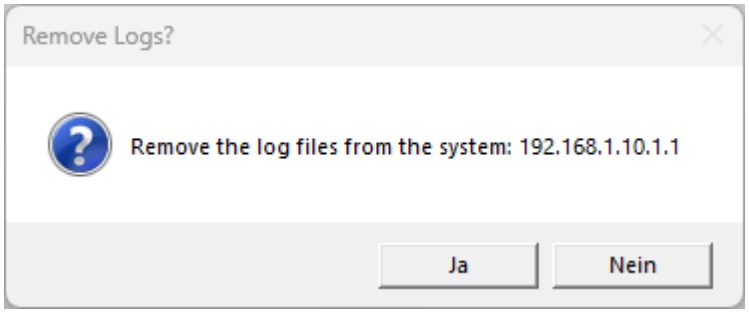
	Name	Online	CS	Type	PTCID	Comment
-	GvspProtocolDiagnosis	...			0x00010001	Contains statistics for GVSP protocol diagnosis.
	.stRecv.nSuccess	390729		ULINT		Number of successfully received packets
	.stRecv.nError	0		ULINT		Number of received packets signalling or causing errors
	.stRecv.nIgnored	0		ULINT		Number of ignored packets
	.nResendRequests	53		ULINT		Number of packet resend requests
	.nCompletedBlocks	139		ULINT		Number of successfully received images
	.nAbortedBlocks	0		ULINT		Number of aborted images
	.nTimedOutBlocks	0		ULINT		Number of timed out images
	.nInvalidBlocks	0		ULINT		Number of invalid images
	.nFurtherErroneousBlocks	0		ULINT		Number of other erroneous images
	.nDuplicateBlocks	0		ULINT		Number of duplicate images
	.nMissingBlocks	0		ULINT		Number of missing images
	.nUnexpectedBlocks	0		ULINT		Number of unexpected images
	.nBlockIdOverflows	0		ULINT		Number of block ID overflows (not an error)

9.3.6 Support Information Report

Support Information Report 是一种收集产品信息并提交给倍福技术支持部门的工具。通过收集系统、配置和日志文件等与产品相关的数据，可以显著减少邮件往来，并加快咨询速度。有关完成此报告的通用分步指南，请参见以下文档。

打开 Support Information Report 的一种可行方法是：右键点击任务栏（系统托盘）中的 TwinCAT 图标，导航至 Tools（工具），并启动 Support Information Report。

打开报告窗口后，首先选择所需的远程系统。如果更改了系统，则会出现提示，询问是否应从先前选择的系统中移除日志文件。如果两个系统的日志文件都相关，例如工程和运行时在不同的计算机上运行，则对此查询回答 **No（否）**。随后，附加路径将被添加到附件字段中。

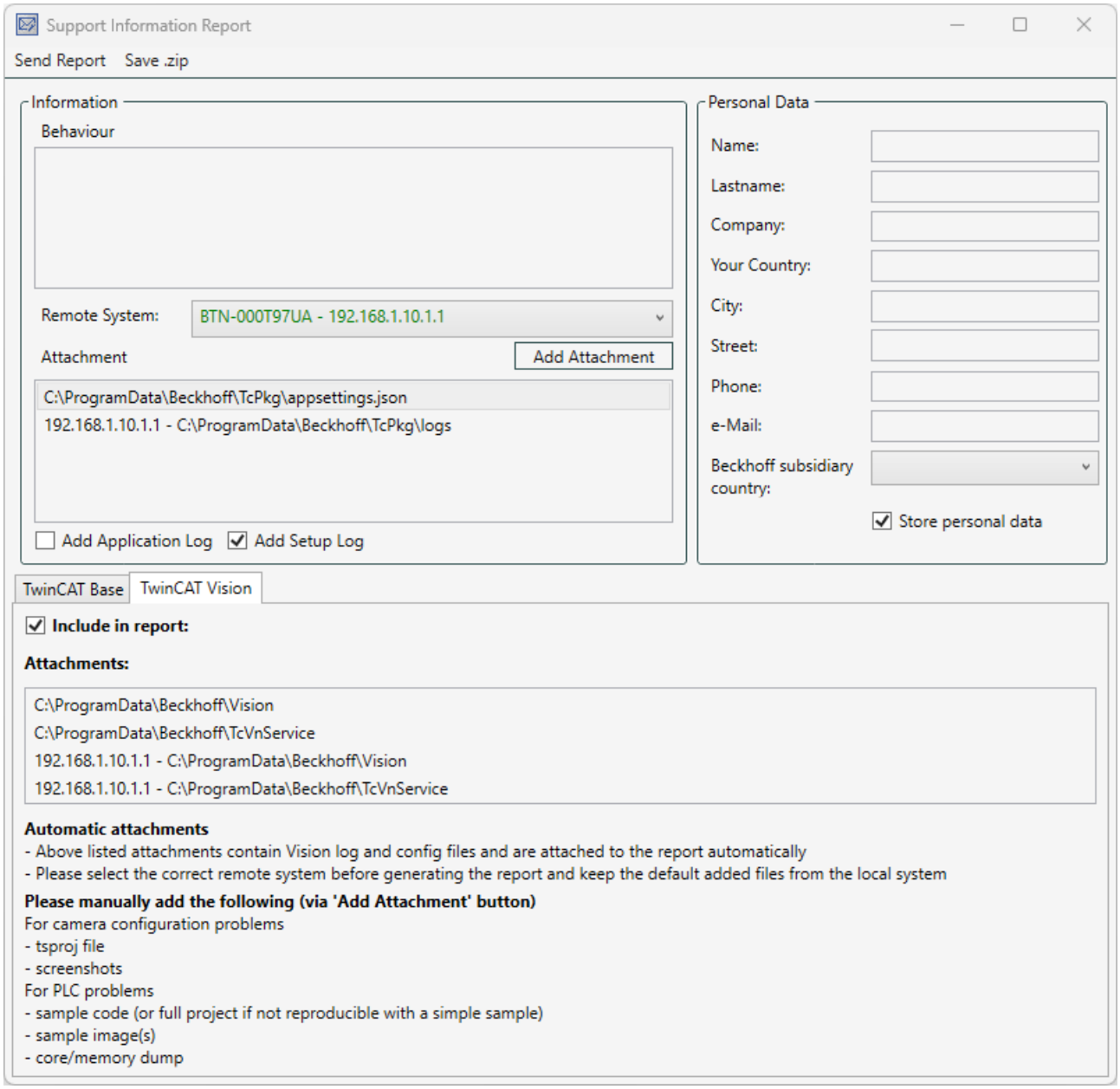


TwinCAT Vision 特定设置

如果在您的 TwinCAT 3.1.4026 系统上安装了 TwinCAT Vision Engineering 工作负载（TF7xxx.Vision.XAE），则在报告窗口的底部会出现 **TwinCAT Vision** 选项卡。

请注意以下产品特定功能：

- **包含在报告中**
  确保在 TwinCAT Vision 选项卡中激活此复选框。这将自动把 "C:\ProgramData\Beckhoff\Vision" 和 "C:\ProgramData\Beckhoff\\TcVnService" 中Vision 特定的日志和配置文件包含到存档中。
- **手动添加附件**
  对于特定问题分析，必须手动添加所需文件。使用 **Add Attachment（添加附件）**按钮来添加这些文件：
  - 对于相机配置问题：
    附加项目文件（.tsproj）以及相关截图和 Wireshark® 记录（如有）。
  - 对于 PLC 问题：
    附加可重现问题的最小化示例程序（如果无法实现，则附上整个项目）、所使用的示例图像，以及任何生成的 TwinCAT 任务/核心/内存转储。



关于 ZIP 压缩包文件大小的说明

所包含的文件夹 "C:\ProgramData\Beckhoff\Vision" 中有一个子文件夹 "_CalibrationAssistantOutput"，其中包含标定助手所使用的图像（如果曾使用过该助手）。为l了最大限度地减小 ZIP 压缩包的文件大小，请在创建报告之前先删除该目录中不必要的图像。

9.4 ADS 返回代码

错误代码分组：
全局错误代码：0x0000 [▶ 2857]... （0x9811_0000 ...）
路由器错误代码：0x0500 [▶ 2858]... （0x9811_0500 ...）
一般 ADS 错误：0x0700 [▶ 2858]... （0x9811_0700 ...）
RTime 错误代码：0x1000 [▶ 2860]... （0x9811_1000 ...）

全局错误代码

Hex	Dec	HRESULT	名称	描述
0x0	0	0x98110000	ERR_NOERROR	无错误。
0x1	1	0x98110001	ERR_INTERNAL	内部错误。
0x2	2	0x98110002	ERR_NORTIME	不具有实时性。

Hex	Dec	HRESULT	名称	描述
0x3	3	0x98110003	ERR_ALLOCLOCKEDMEM	分配已锁定 – 内存错误。
0x4	4	0x98110004	ERR_INSERTMAILBOX	邮箱已满 – 无法发送 ADS 消息。减少每个周期的 ADS 消息数量将有所帮助。
0x5	5	0x98110005	ERR_WRONGRECEIVEHMSG	HMSG 错误。
0x6	6	0x98110006	ERR_TARGETPORTNOTFOUND	未找到目标端口 – ADS 服务器未启动或无法访问。
0x7	7	0x98110007	ERR_TARGETMACHINENOTFOUND	未找到目标计算机 – 未找到 AMS 路由。
0x8	8	0x98110008	ERR_UNKNOWNCMDID	未知命令 ID。
0x9	9	0x98110009	ERR_BADTASKID	任务 ID 无效。
0xA	10	0x9811000A	ERR_NOIO	No IO。
0xB	11	0x9811000B	ERR_UNKNOWNAMSCMD	未知 AMS 命令。
0xC	12	0x9811000C	ERR_WIN32ERROR	Win32 错误。
0xD	13	0x9811000D	ERR_PORTNOTCONNECTED	端口未连接。
0xE	14	0x9811000E	ERR_INVALIDAMSLENGTH	AMS 长度无效。
0xF	15	0x9811000F	ERR_INVALIDAMSNETID	AMS Net ID 无效。
0x10	16	0x98110010	ERR_LOWINSTLEVEL	安装级别 – TwinCAT 2 授权错误。
0x11	17	0x98110011	ERR_NODEBUGINTAVAILABLE	无调试可用。
0x12	18	0x98110012	ERR_PORTDISABLED	端口已禁用 – TwinCAT 系统服务未启动。
0x13	19	0x98110013	ERR_PORTALREADYCONNECTED	端口已连接。
0x14	20	0x98110014	ERR_AMSSYNC_W32ERROR	AMS Sync Win32 错误。
0x15	21	0x98110015	ERR_AMSSYNC_TIMEOUT	AMS Sync 超时。
0x16	22	0x98110016	ERR_AMSSYNC_AMSERROR	AMS Sync 错误。
0x17	23	0x98110017	ERR_AMSSYNC_NOINDEXINMAP	不存在适用于 AMS Sync 的索引映射。
0x18	24	0x98110018	ERR_INVALIDAMSPORT	AMS 端口无效。
0x19	25	0x98110019	ERR_NOMEMORY	无内存。
0x1A	26	0x9811001A	ERR_TCPSEND	TCP 发送错误。
0x1B	27	0x9811001B	ERR_HOSTUNREACHABLE	主机无法访问。
0x1C	28	0x9811001C	ERR_INVALIDAMSFRAGMENT	AMS 片段无效。
0x1D	29	0x9811001D	ERR_TLSSEND	TLS 发送错误 – ADS 安全连接失败。
0x1E	30	0x9811001E	ERR_ACCESSDENIED	拒绝访问 – 拒绝 ADS 安全访问。

路由器错误代码

Hex	Dec	HRESULT	名称	描述
0x500	1280	0x98110500	ROUTERERR_NOLOCKEDMEMORY	无法分配锁定的内存。
0x501	1281	0x98110501	ROUTERERR_RESIZEMEMORY	路由器内存大小无法更改。
0x502	1282	0x98110502	ROUTERERR_MAILBOXFULL	邮箱已达到最大消息数。
0x503	1283	0x98110503	ROUTERERR_DEBUGBOXFULL	调试邮箱已达到最大消息数。
0x504	1284	0x98110504	ROUTERERR_UNKNOWNPORTTYPE	端口类型未知。
0x505	1285	0x98110505	ROUTERERR_NOTINITIALIZED	路由器未初始化。
0x506	1286	0x98110506	ROUTERERR_PORTALREADYINUSE	端口号已分配。
0x507	1287	0x98110507	ROUTERERR_NOTREGISTERED	端口未注册。
0x508	1288	0x98110508	ROUTERERR_NOMOREQUEUES	已达到最大端口数。
0x509	1289	0x98110509	ROUTERERR_INVALIDPORT	端口无效。
0x50A	1290	0x9811050A	ROUTERERR_NOTACTIVATED	路由未激活。
0x50B	1291	0x9811050B	ROUTERERR_FRAGMENTBOXFULL	邮箱已达到最大片段消息数。
0x50C	1292	0x9811050C	ROUTERERR_FRAGMENTTIMEOUT	发生片段超时。
0x50D	1293	0x9811050D	ROUTERERR_TOBEREMOVED	端口已移除。

一般性 ADS 错误代码

Hex	Dec	HRESULT	名称	描述
0x700	1792	0x98110700	ADSERR_DEVICE_ERROR	一般性设备错误。

Hex	Dec	HRESULT	名称	描述
0x701	1793	0x98110701	ADSERR_DEVICE_SRVNOTSUPP	服务器不支持该服务。
0x702	1794	0x98110702	ADSERR_DEVICE_INVALIDGRP	索引组无效。
0x703	1795	0x98110703	ADSERR_DEVICE_INVALIDOFFSET	索引偏移量无效。
0x704	1796	0x98110704	ADSERR_DEVICE_INVALIDACCESS	不允许读取或写入。 可能有几种原因。例如，创建路由时输入了错误的密码。
0x705	1797	0x98110705	ADSERR_DEVICE_INVALIDSIZE	参数大小不正确。
0x706	1798	0x98110706	ADSERR_DEVICE_INVALIDDATA	无效数据值。
0x707	1799	0x98110707	ADSERR_DEVICE_NOTREADY	设备尚未准备好运行。
0x708	1800	0x98110708	ADSERR_DEVICE_BUSY	设备正忙。
0x709	1801	0x98110709	ADSERR_DEVICE_INVALIDCONTEXT	操作系统上下文无效。这可能是由于在不同的任务中使用 ADS 函数块造成的。可以通过在 PLC 中进行多任务同步解决这个问题。
0x70A	1802	0x9811070A	ADSERR_DEVICE_NOMEMORY	内存不足。
0x70B	1803	0x9811070B	ADSERR_DEVICE_INVALIDPARM	参数值无效。
0x70C	1804	0x9811070C	ADSERR_DEVICE_NOTFOUND	未找到（文件…）。
0x70D	1805	0x9811070D	ADSERR_DEVICE_SYNTAX	文件或命令中存在语法错误。
0x70E	1806	0x9811070E	ADSERR_DEVICE_INCOMPATIBLE	对象不匹配。
0x70F	1807	0x9811070F	ADSERR_DEVICE_EXISTS	对象已存在。
0x710	1808	0x98110710	ADSERR_DEVICE_SYMBOLNOTFOUND	符号未找到。
0x711	1809	0x98110711	ADSERR_DEVICE_SYMBOLVERSIONINVALID	符号版本无效。这可能是由于联机更改造成的。创建新句柄。
0x712	1810	0x98110712	ADSERR_DEVICE_INVALIDSTATE	设备（服务器）处于无效状态。
0x713	1811	0x98110713	ADSERR_DEVICE_TRANSMODENOTSUPP	不支持 AdsTransMode。
0x714	1812	0x98110714	ADSERR_DEVICE_NOTIFYHNDINVALID	通知句柄无效。
0x715	1813	0x98110715	ADSERR_DEVICE_CLIENTUNKNOWN	通知客户端未注册。
0x716	1814	0x98110716	ADSERR_DEVICE_NOMOREHDLS	没有更多的句柄可用。
0x717	1815	0x98110717	ADSERR_DEVICE_INVALIDWATCHSIZE	通知大小过大。
0x718	1816	0x98110718	ADSERR_DEVICE_NOTINIT	设备未初始化。
0x719	1817	0x98110719	ADSERR_DEVICE_TIMEOUT	设备超时。
0x71A	1818	0x9811071A	ADSERR_DEVICE_NOINTERFACE	接口查询失败。
0x71B	1819	0x9811071B	ADSERR_DEVICE_INVALIDINTERFACE	请求的接口错误。
0x71C	1820	0x9811071C	ADSERR_DEVICE_INVALIDCLSID	Class ID 无效。
0x71D	1821	0x9811071D	ADSERR_DEVICE_INVALIDOBJID	Object ID 无效。
0x71E	1822	0x9811071E	ADSERR_DEVICE_PENDING	请求待定。
0x71F	1823	0x9811071F	ADSERR_DEVICE_ABORTED	请求已中止。
0x720	1824	0x98110720	ADSERR_DEVICE_WARNING	警告信号。
0x721	1825	0x98110721	ADSERR_DEVICE_INVALIDARRAYIDX	数组索引无效。
0x722	1826	0x98110722	ADSERR_DEVICE_SYMBOLNOTACTIVE	符号未激活。
0x723	1827	0x98110723	ADSERR_DEVICE_ACCESSDENIED	拒绝访问。 有几种可能的原因。例如，单向 ADS 路由用于相反方向。
0x724	1828	0x98110724	ADSERR_DEVICE_LICENSENOTFOUND	缺少授权。
0x725	1829	0x98110725	ADSERR_DEVICE_LICENSEEXPIRED	授权已过期。
0x726	1830	0x98110726	ADSERR_DEVICE_LICENSEEXCEEDED	超出授权。
0x727	1831	0x98110727	ADSERR_DEVICE_LICENSEINVALID	授权无效。
0x728	1832	0x98110728	ADSERR_DEVICE_LICENSESYSTEMID	授权问题：系统 ID 无效。
0x729	1833	0x98110729	ADSERR_DEVICE_LICENSENOTIMELIMIT	授权不受时间限制。
0x72A	1834	0x9811072A	ADSERR_DEVICE_LICENSEFUTUREISSUE	授权问题：未来的时间。
0x72B	1835	0x9811072B	ADSERR_DEVICE_LICENSETIMETOLONG	授权期限太长。
0x72C	1836	0x9811072C	ADSERR_DEVICE_EXCEPTION	系统启动异常。
0x72D	1837	0x9811072D	ADSERR_DEVICE_LICENSEDUPLICATED	授权文件读取了两次。
0x72E	1838	0x9811072E	ADSERR_DEVICE_SIGNATUREINVALID	签名无效。
0x72F	1839	0x9811072F	ADSERR_DEVICE_CERTIFICATEINVALID	证书无效。
0x730	1840	0x98110730	ADSERR_DEVICE_LICENSEOEMNOTFOUND	OEM未知公钥。
0x731	1841	0x98110731	ADSERR_DEVICE_LICENSERESTRICTED	此系统 ID 授权无效。
0x732	1842	0x98110732	ADSERR_DEVICE_LICENSEDEMODENIED	演示授权已禁止。

Hex	Dec	HRESULT	名称	描述
0x733	1843	0x98110733	ADSERR_DEVICE_INVALIDFNCID	无效函数 ID。
0x734	1844	0x98110734	ADSERR_DEVICE_OUTOFRANGE	超出有效范围。
0x735	1845	0x98110735	ADSERR_DEVICE_INVALIDALIGNMENT	无效对齐。
0x736	1846	0x98110736	ADSERR_DEVICE_LICENSEPLATFORM	无效平台级别。
0x737	1847	0x98110737	ADSERR_DEVICE_FORWARD_PL	上下文 – 转到被动级别。
0x738	1848	0x98110738	ADSERR_DEVICE_FORWARD_DL	上下文 – 转到调度级别。
0x739	1849	0x98110739	ADSERR_DEVICE_FORWARD_RT	上下文 – 转到实时。
0x740	1856	0x98110740	ADSERR_CLIENT_ERROR	客户端错误。
0x741	1857	0x98110741	ADSERR_CLIENT_INVALIDPARM	服务包含无效参数。
0x742	1858	0x98110742	ADSERR_CLIENT_LISTEMPTY	轮询列表为空。
0x743	1859	0x98110743	ADSERR_CLIENT_VARUSED	Var 连接已在使用中。
0x744	1860	0x98110744	ADSERR_CLIENT_DUPLINVOKEID	调用的 ID 已在使用中。
0x745	1861	0x98110745	ADSERR_CLIENT_SYNCTIMEOUT	已发生超时 – 远程终端在指定的 ADS 超时中未响应。远程终端的路由设置可能配置不正确。
0x746	1862	0x98110746	ADSERR_CLIENT_W32ERROR	Win32 子系统中发生错误。
0x747	1863	0x98110747	ADSERR_CLIENT_TIMEOUTINVALID	客户端超时值无效。
0x748	1864	0x98110748	ADSERR_CLIENT_PORTNOTOPEN	端口未打开。
0x749	1865	0x98110749	ADSERR_CLIENT_NOAMSADDR	无 AMS 地址。
0x750	1872	0x98110750	ADSERR_CLIENT_SYNCINTERNAL	Ads sync 中发生内部错误。
0x751	1873	0x98110751	ADSERR_CLIENT_ADDHASH	哈希表溢出。
0x752	1874	0x98110752	ADSERR_CLIENT_REMOVEHASH	未在表格中找到密钥。
0x753	1875	0x98110753	ADSERR_CLIENT_NOMORESYM	缓存中没有符号。
0x754	1876	0x98110754	ADSERR_CLIENT_SYNCRESINVALID	收到无效响应。
0x755	1877	0x98110755	ADSERR_CLIENT_SYNCPORTLOCKED	同步端口已锁定。
0x756	1878	0x98110756	ADSERR_CLIENT_REQUESTCANCELLED	请求被取消。

RTime 错误代码

Hex	Dec	HRESULT	名称	描述
0x1000	4096	0x98111000	RTERR_INTERNAL	实时系统中发生内部错误。
0x1001	4097	0x98111001	RTERR_BADTIMERPERIODS	计时器值无效。
0x1002	4098	0x98111002	RTERR_INVALIDTASKPTR	任务指针提供了无效值 0（零）。
0x1003	4099	0x98111003	RTERR_INVALIDSTACKPTR	堆栈指针提供了无效值 0（零）。
0x1004	4100	0x98111004	RTERR_PRIOEXISTS	请求任务优先级已分配。
0x1005	4101	0x98111005	RTERR_NOMORETCB	没有可用的空闲 TCB（任务控制块）。TCB 的最大数量为 64。
0x1006	4102	0x98111006	RTERR_NOMORESEMAS	没有可用的空闲信号。信号的最大数量为 64。
0x1007	4103	0x98111007	RTERR_NOMOREQUEUES	队列中没有可用的空闲空间。队列中的最大位置数为 64。
0x100D	4109	0x9811100D	RTERR_EXTIRQALREADYDEF	已应用外部同步中断。
0x100E	4110	0x9811100E	RTERR_EXTIRQNOTDEF	未应用外部同步中断。
0x100F	4111	0x9811100F	RTERR_EXTIRQINSTALLFAILED	外部同步中断应用失败。
0x1010	4112	0x98111010	RTERR_IRQLNOTLESSOREQUAL	在错误的上下文中调用服务函数
0x1017	4119	0x98111017	RTERR_VMXNOTSUPPORTED	不支持 Intel® VT-x 扩展。
0x1018	4120	0x98111018	RTERR_VMXDISABLED	BIOS 中未启用 Intel® VT-x 扩展。
0x1019	4121	0x98111019	RTERR_VMXCONTROLSMISSING	Intel® VT-x 扩展中缺少函数。
0x101A	4122	0x9811101A	RTERR_VMXENABLEFAILS	Intel® VT-x 激活失败。

具体的正向 HRESULT 返回代码：

HRESULT	名称	描述
0x0000_0000	S_OK	无错误。
0x0000_0001	S_FALSE	无错误。 示例：处理成功，但有一个负面或不完整的结果。
0x0000_0203	S_PENDING	无错误。 示例：处理成功，但还没有结果。

HRESULT	名称	描述
0x0000_0256	S_WATCHDOG_TIMEOUT	无错误。 示例：处理成功，但发生了超时。

TCP Winsock 错误代码

Hex	Dec	名称	描述
0x274C	10060	WSAETIMEDOUT	发生连接超时 - 在创建连接时发生错误，因为远程终端在特定时间后未正确响应，或者所建立连接因所连接主机未响应而无法维持。
0x274D	10061	WSAECONNREFUSED	连接遭到拒绝 - 无法建立连接，因为目标计算机明确拒绝了该连接。此错误通常由尝试连接到外部主机上处于非活动状态的服务（即没有运行服务器应用程序的服务）导致。
0x2751	10065	WSAEHOSTUNREACH	不存在至主机的路由 - 套接字操作引用了不可用的主机。
更多 Winsock 错误代码：Win32 错误代码			

9.5 技术支持和服务

倍福技术支持和服务部门可就倍福产品及系统解决方案的相关问题，以及其规划、调试和运行提供协助。

倍福子公司和代表处

有关倍福产品本地技术支持和服务方面的信息，请联系倍福子公司或代表处。您可以在以下网址找到倍福子公司和代表处的地址：www.beckhoff.com/worldwide。

针对倍福产品和技术，我们提供培训课程、教程和网络研讨会。

倍福技术支持

倍福技术支持部门可协助解答有关倍福产品的使用、规划、编程和调试的技术问题。

☎ 热线电话： +49 5246 963-157
✉ 电子邮件： support@beckhoff.com，技术支持表单
网页： www.beckhoff.com/support

倍福服务部门

我们的服务专员随时为您解答有关售后服务流程的任何问题，例如更换零部件、维修等。

☎ 热线电话： +49 5246 963-460
✉ 电子邮件： service@beckhoff.com，服务表单
网页： www.beckhoff.com/service

下载搜索器

下载搜索器是获取倍福产品相关可下载文档和文件（如应用报告、技术文档、图纸和配置文件）的中央资源入口。

倍福信息系统

倍福信息系统是倍福产品的中央在线文档资源，包含产品文档、自动化主题相关文档以及示例代码。

倍福公司总部

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl, Germany

☎ 电话： +49 5246 963-0
✉ 电子邮件： info@beckhoff.com
网页： www.beckhoff.com

9.6 第三方组件

本软件包含第三方组件。有关详细信息，请参见以下文件夹中提供的许可文件：

TwinCAT 3.1 4024 和 TwinCAT 3.1 4026 中的不同内存路径

Build 4024 及以下版本：

<TwinCatInstallPath>\3.1\System\Legal\TF7xxx Additional license information.txt

Build 4026 及以上版本：

- C:/Program Files (x86)/Beckhoff\Legal\TwinCAT-XAE-GigEVision
- C:/Program Files (x86)/Beckhoff\Legal\TwinCAT-XAE-TMX-Vision
- C:/Program Files (x86)/Beckhoff\Legal\TwinCAT-XAE-Vision
- C:/Program Files (x86)/Beckhoff\Legal\TwinCAT-XAR-VisionService

Trademark statements

Beckhoff®, ATRO®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, MX-System®, Safety over EtherCAT®, TC/BSD®, TwinCAT®, TwinCAT/BSD®, TwinSAFE®, XFC®, XPlanar® and XTS® are registered and licensed trademarks of Beckhoff Automation GmbH.

Third-party trademark statements

Chrome, Chromium and Google are trademarks of Google LLC.

DALI, DALI-2, D4i, DALI+, and DiiA are trademarks in various countries in the exclusive use of the Digital Illumination Interface Alliance.

DSP System Toolbox, Embedded Coder, MATLAB, MATLAB Coder, MATLAB Compiler, MathWorks, Predictive Maintenance Toolbox, Simscape, Simscape™ Multibody™, Simulink, Simulink Coder, Stateflow and ThingSpeak are registered trademarks of The MathWorks, Inc.

Excel, IntelliSense, Microsoft, Microsoft Azure, Microsoft Edge, PowerShell, Visual Studio, Windows and Xbox are trademarks of the Microsoft group of companies.

MAX®, Stratix®, Cyclone®, Altera®, Agilex™, Arria®, Intel, the Intel logo, Intel Core, Xeon, Intel Atom, Celeron and Pentium are trademarks of Intel Corporation or its subsidiaries.

Mozilla and Firefox are trademarks of the Mozilla Foundation in the U.S. and other countries.

PCI Express®, PCIe®, PCI™ and PCI HOT PLUG™ are trademarks or registered trademarks and/or service marks of PCI-SIG.

Wireshark is a registered trademark of Sysdig, Inc.

更多信息:
www.beckhoff.com/twincat-vision/

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
电话号码: +49 5246 9630
info@beckhoff.com
www.beckhoff.com