COGNEX

EtherInspect 11
Reference Manual

2024 April 11

Legal Notices

Symbols

The following symbols indicate safety precautions and supplemental information:

WARNING: This symbol indicates a hazard that could cause death, serious personal injury or electrical shock.

CAUTION: This symbol indicates a hazard that could result in property damage.

Note: This symbol indicates additional information about a subject.

Tip: This symbol indicates suggestions and shortcuts that might not otherwise be apparent.

Table of Contents

About EtherInspect

EtherInspect is an easy-to-use, network-ready machine vision inspection software that supports a variety of Cognex machine vision systems including 2D and 3D cameras alongside a simple, customizable user interface. EtherInspect has different user interface templates that allow for a variety of different applications from tray inspection to EV battery inspection. EtherInspect also includes ViDi and VisionPro Deep Learning support for machine learning applications.

Getting Started

About the Release

Software and Hardware Requirements

Software

Before using EtherInspect, install the following software:

- Windows 10/11 64 Bit
- VisionPro
    - VisionPro 9.5 SR2
    - VisionPro 9.6 SR2
    - VisionPro 9.8
    - VisionPro 9.8 SR1
    - VisionPro 9.10
    - VisionPro 9.20
- ViDi/VPDL
    - ViDi Suite 4.1
    - VPDL 1.1
    - VPDL 3.0
    - VPDL 3.1 (Required with VisionPro 9.10)
    - VPDL 3.2 (Required with VisionPro 9.20)
- .NET Framework 4.8 (Installs with VisionPro 9.10 and 9.20)

Hardware

- PC
- Security Dongle
- Camera Options:
    - 2D
        - Cognex CIC (Area/Line)
        - CameraLink Line Scane Cameras (uses BitFlow Framegrabber)
    - 3D
        - DS1000 series (Line)
        - 3D-L4000 Series with VisionPro Firmware (Line)
        - 3D-A5000 Series (Area)
        - PhotoNeo (Area)
- Cat6 Ethernet Cable

- Power Options:
    - Camera Power Supply
    - GigE PoE switch
    - GigE PoE Framegrabber

Supported Cameras and PLCs

For a complete list of supported cameras, see the VisionPro Camera Support website.

Note: Not all acquisition platforms are supported on all operating systems.

- Cognex CIC GigE Cameras
- Cognex 3D-A5000 Area Scan Sensors
- Cognex DS1000 Laser Line Sensors
- GigE cameras that are Gen-I-Cam compliant
- CameraLink linescan cameras using BitFlow framegrabber
- Photoneo PhoXi Sensors
- Up to 12 sensors are supported per PC (with standard license dongle)
- Up to 64 camera connections for each EtherInspect instance.
- DX100

Supported PLC	Version
Allen-Bradley	ControlLogix
	Micro 800
Siemens S7	200
	300
	400
	1200
	1500
Beckhoff	TwinCAT3
Modbus	N/A
Mitsubishi	MC3E
	MC4E
Motoman	N/A

Setting Up Your Vision System for EtherInspect

Read this section to learn how the vision system connects to its standard components and accessories.

Connecting the Ethernet Cable

CAUTION: The Ethernet cable shield must be grounded at the far end.Whatever this cable is plugged into (typically a switch or router) should have a grounded Ethernet connector. A digital voltmeter should be used to validate the grounding. If the far end device is not grounded, a ground wire should be added in compliance with local electrical codes.

Single Camera Setup

1. Connect your Cognex CIC Vision Camera directly to your PC with an Ethernet cable.
2. Use the power adapter to power up your camera.

Multiple Camera Setup

1. Connect the Vision PC to the Ethernet switch with an Ethernet cable.
2. Use a separate POE port on the Ethernet switch to connect the switch to the GigE cameras.

Note: Make sure to use the ports on the switch that support POE.

Connecting the Power and I/O Breakout Cable

CAUTION: To reduce emissions, connect the far end of the Breakout cable shield to frame ground.

Note:

- Perform wiring or adjustments to I/O devices when the vision system is not receiving power.
- You can clip unused wires short or use a tie made of non-conductive material to tie them back. Keep bare wires separated from the +24 V DC wire.

1. Verify that the 24 V DC power supply is unplugged and not receiving power.
2. Attach the +24 V DC connector of the Power and I/O Breakout cable and Ground wires to the corresponding terminals on the power supply. For more information, see Specifications on page 1.

   CAUTION: Never connect voltages other than 24 V DC. Always observe the polarity shown.
3. Attach the M12 connector of the Power and I/O Breakout Cable to the 24 V DC connector of the vision system.
4. Restore power to the 24 V DC power supply and turn it on if necessary.

Software Installation

Refer to the following sections on installing prerequisite Cognex software and EtherInspect.

Note: Attach the security dongle before installing the software.

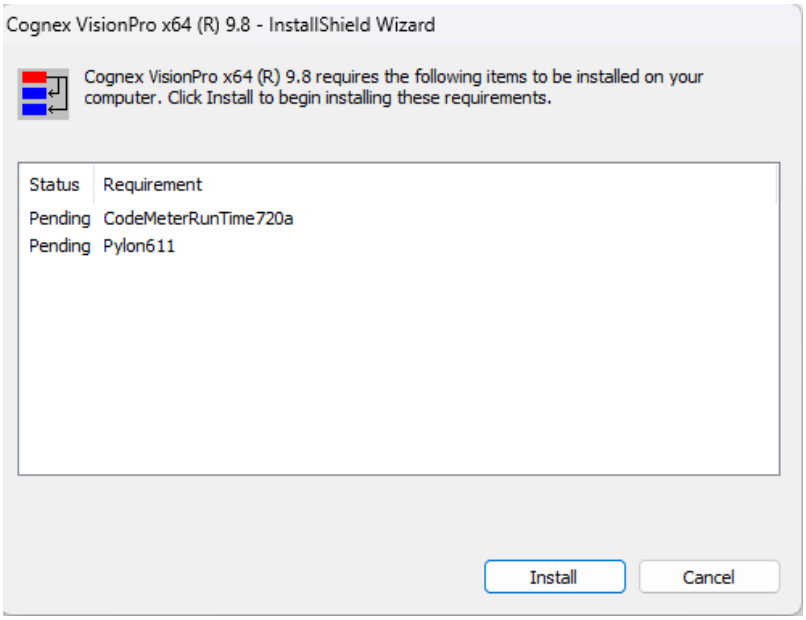
VisionPro Installation

1. Go to the VisionPro Installer on the Cognex support site.
2. Download VisionPro.

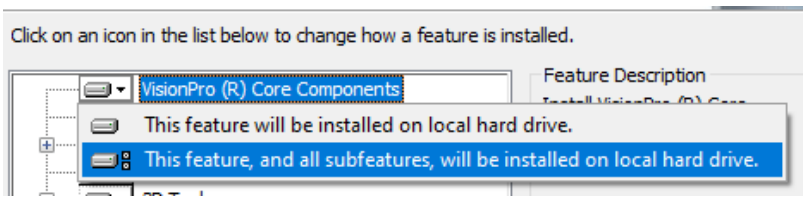
   Note: To download, you need a MyCognex account.

3. Unzip and extract the folder and run *setup.exe*.
4. If prompted, install the PylonRuntime, CodeMeterRuntime, and .NET Framework. Pylon provides the basic GigE drivers to access vision cameras. CodeMeter allows for the renewal, updating, and viewing of Cognex software license info provided on your license dongle.

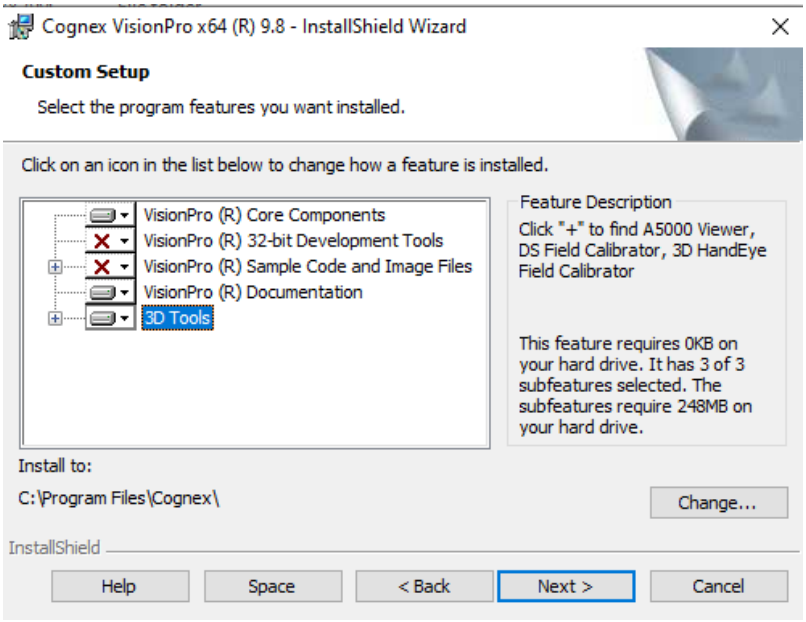
   Note: Updating from 11.3.xx-yy or older requires .NET Framework 4.8.



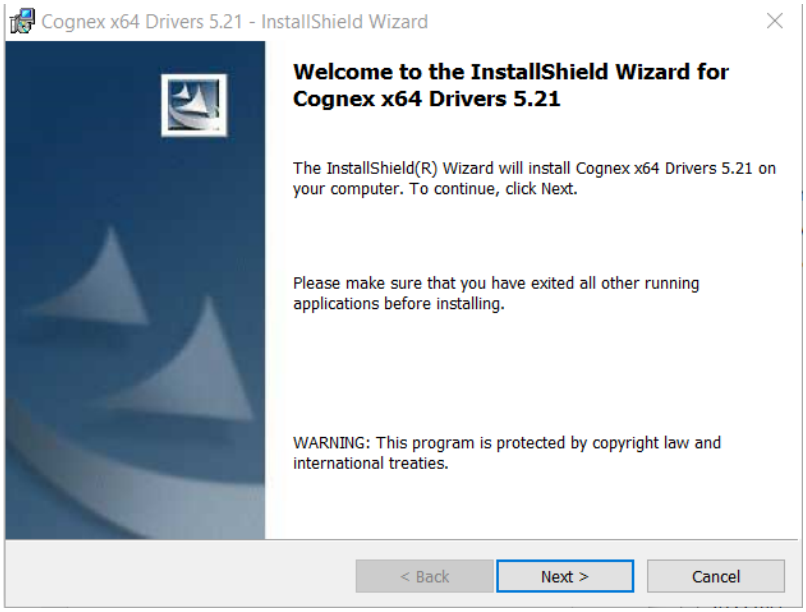
5. The InstallShield Wizard opens. Keep clicking next until you reach the Cognex Software License Agreement page.
6. Review the License Agreement and click **I accept the terms in the license agreement** and continue.
7. Provide a username and organization name.
8. In the Custom Installs page, save both the main feature and the sub-features for VisionPro Core Components, VisionPro Documentation, and 3D tools to your local drive.



9. Remove VisionPro Development Tools and VisionPro Sample Code and Image Files from the list of features to install.



10. Click **Install**.
11. Check the box to launch the **Cognex Driver Installer**. Keep **Launch Asian Language Documents Installer** unchecked. The selection installs drivers to interface with any Cognex vision cameras that you attach to your PC. Restart the application when prompted.
12. A separate install wizard opens that installs the drivers. Click **Next**.

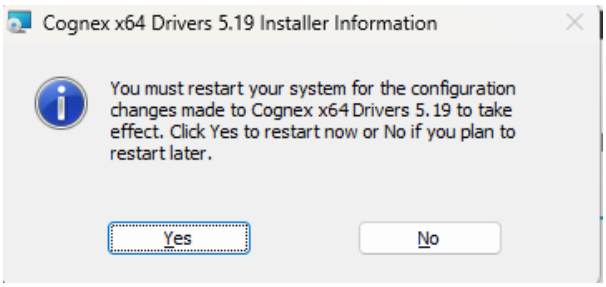


13. Review and accept the Cognex Software License Agreement. Click **Next**.
14. For Setup Type, select **Complete**. Click **Next**.
15. Click **Install**.

16. After installing the Cognex drivers, you are prompted with the option to install the DSMax Frame Grabber.

    Uncheck **Launch DSMAX Frame Grabber Driver Installer** and click **Finish**.

17. The installer prompts you to restart your system to complete the installation. Click **Yes**.



VisionPro GPU Supplement

If you want to use your GPU when running VisionPro Deep Learning workspaces, the VisionPro GPU supplement is required for VisionPro 9.20. After installing VisionPro and before installing VisionPro Deep Learning, you can install the VisionPro GPU supplement.

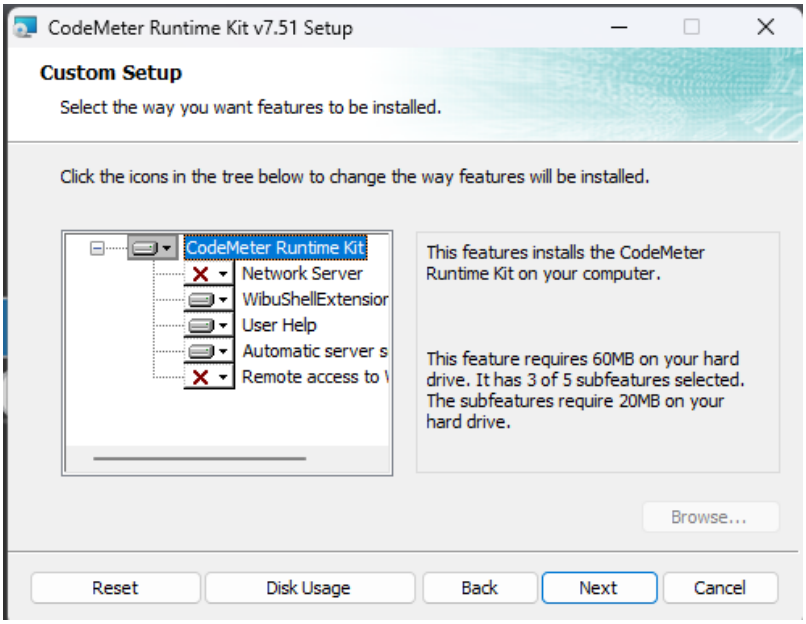
Note: Only VisionPro 9.20 and VisionPro Deep Learning 3.2 use the supplement.

VisionPro Deep Learning/ViDi Installation

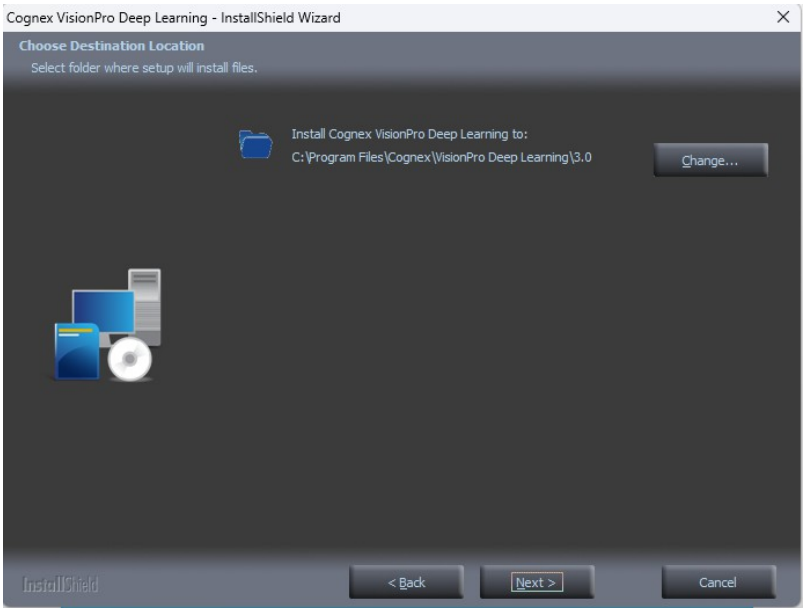
1. Go to the VisionPro Deep Learning Installer on the Cognex support site.
2. Download VisionProDeepLearning.
3. Extract the .zip folder and run the .exe file to install VisionPro Deep Learning.
4. Click **Install**.
5. Once the Setup Wizard appears, click **Next**.

6. Accept the License Agreement for CodeMeter.
7. Choose which users you want to install the application for.

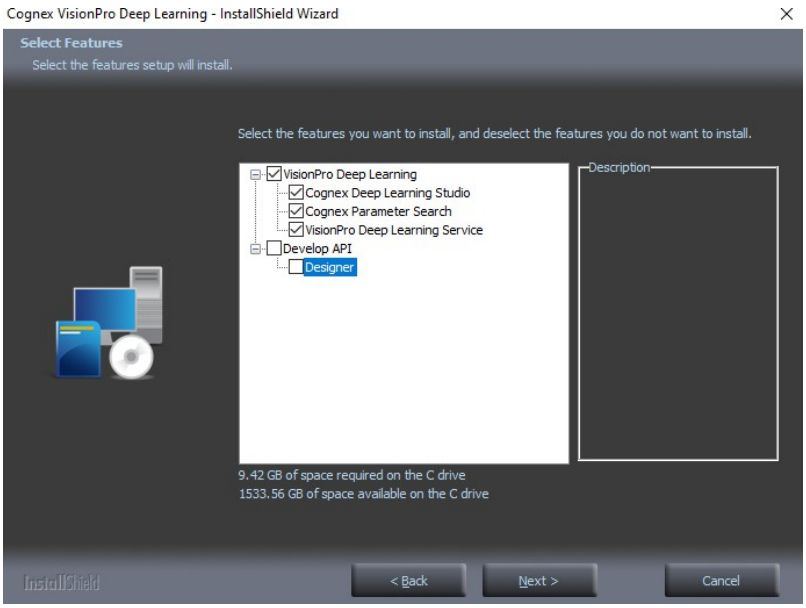
8. Maintain the default installation options set by CodeMeter when dealing with Custom installation. Click **Next**.



9. Click **Install**.
10. Click **Finish**.
11. You are sent back to the main installer for VisionPro Deep Learning. Review and accept the Software License Agreement.
12. Provide a username and organization name.
13. Select **Custom** under Setup Type. Click **Next**.
14. For Destination Location, keep the default configuration and click **Next**.



15. Uncheck the box labeled **DevelopAPI**. Click **Next**.

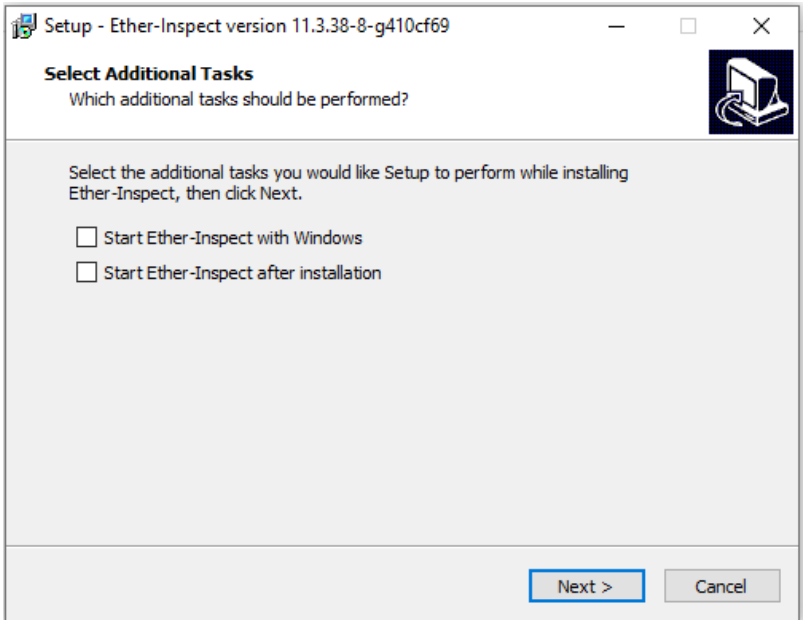


16. Click **Install**.
17. Click **Finish**.

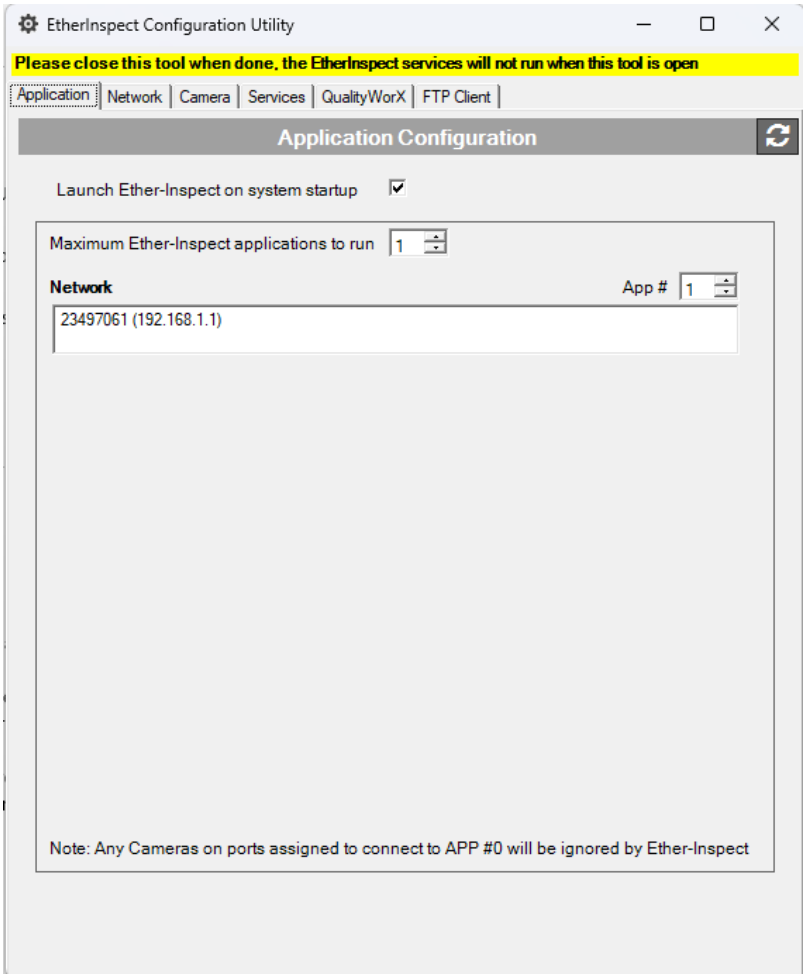
EtherInspect Installation

1. Download and run the latest version of EtherInspectSetup.exe.
2. Run EtherInspect Configuration tool from desktop to configure network interfaces and cameras.
3. Check the box **Start Ether-Inspect with Windows** to start the application immediately after the PC boots up. Check the box **Start Ether-Inspect after installation** to run the application after installation. Choose which boxes to check and continue the installation.

> **Note**: Enabling **Start Ether-Inspect With Windows** enables the Auto-Restart function.



4. Configure the application.



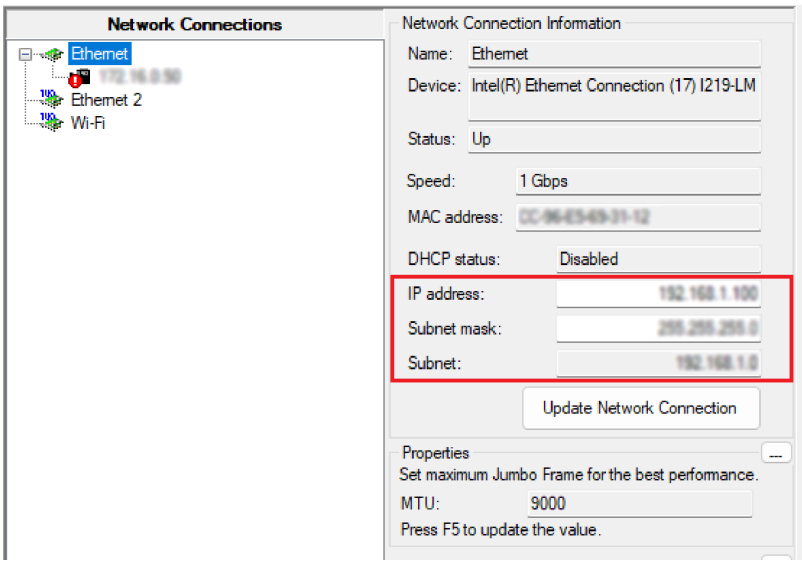
5. Click **Finish**.

Configure a GigE Camera Using Cognex Vision Configurator

The Cognex GigE Vision Configurator allows you to edit the IP address and network properties of GigE cameras.

1. To configure a GigE Camera for image acquisition in EtherInspect, you need to open the Cognex GigE Vision Configurator included in the VisionPro installation.

2. Select the network adapter with a camera attached to it.
3. Change the adapter IP address to a different one from your local network devices.

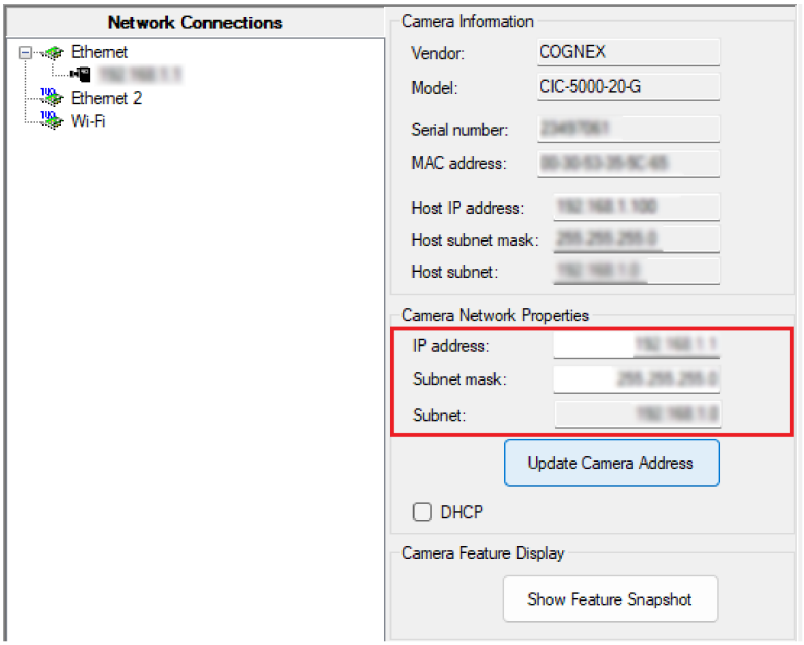


4. Click **Update Network Connection** to save the changes.

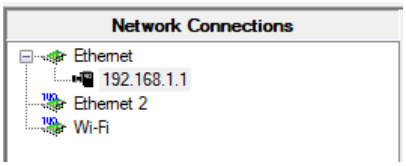
> **Note**: The subnet automatically updates when the network connection is updated.

5. Select the camera connected to your network adapter.
6. Change the IP address to a different one from your local network devices.

> **Note**: Make sure the IP address of your camera is the same subnet as your network adapter for correct image acquisition.

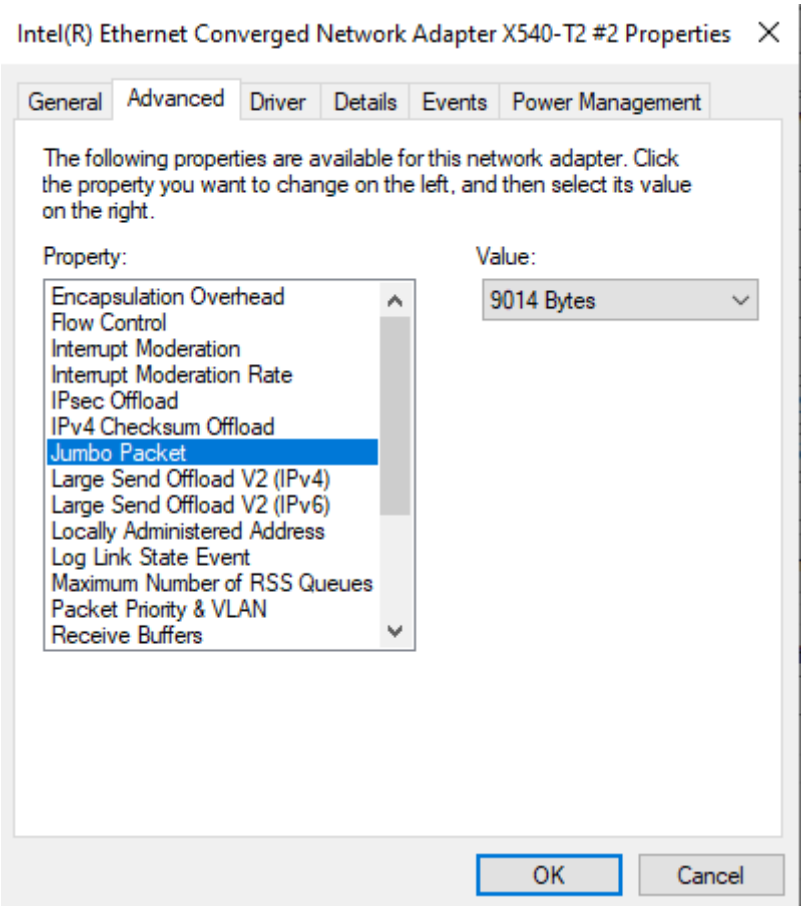


7. Your camera is now ready for image acquisition.

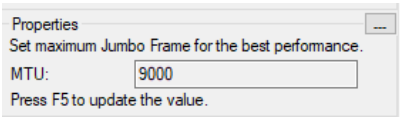


Additional Settings for GigE Network Setup

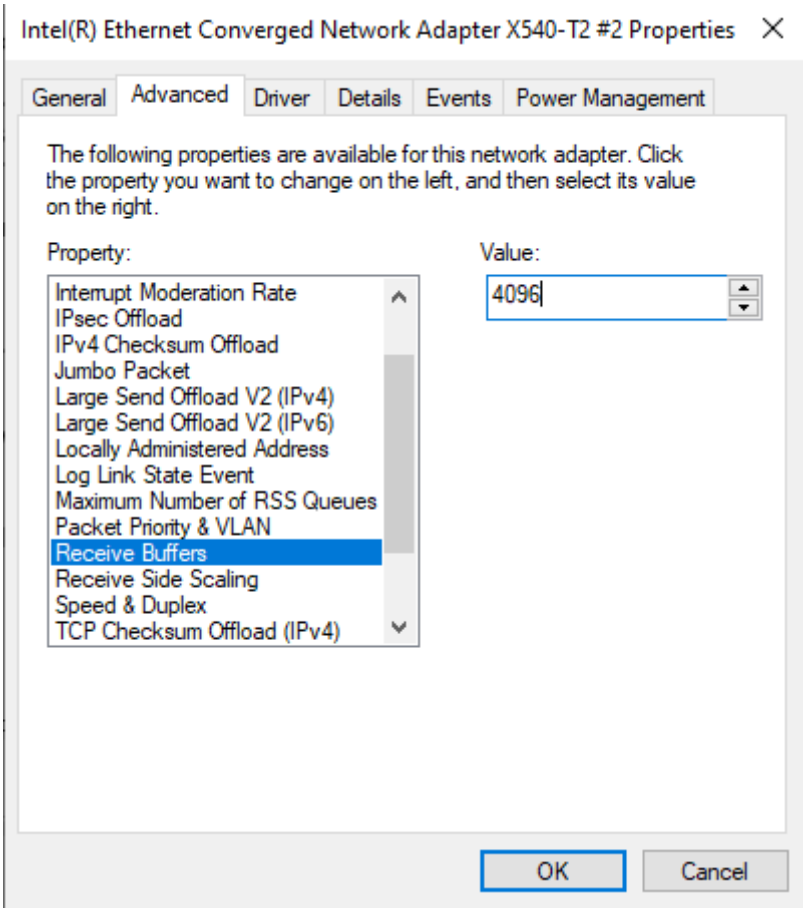
1. To improve the image quality, set the **Jumbo Packet** property of your network adapter to the maximum value.



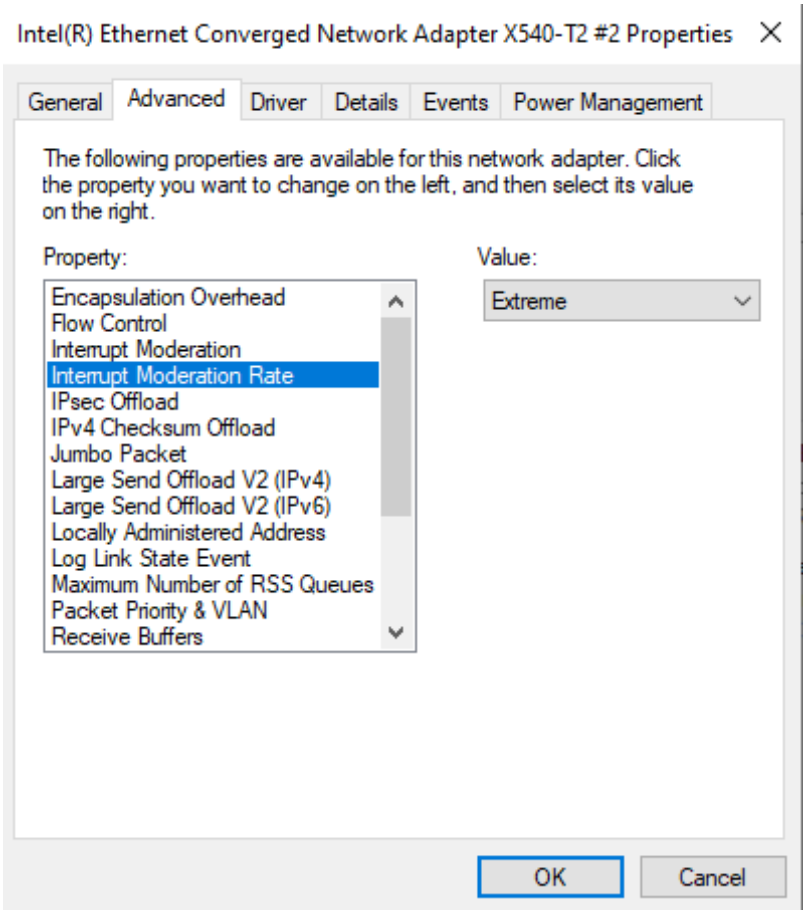
The maximum transition unit (MTU) of your camera increases.



2. To increase the amount of data your network adapter receives, set the **Receive Buffer** property of your camera to the maximum value.

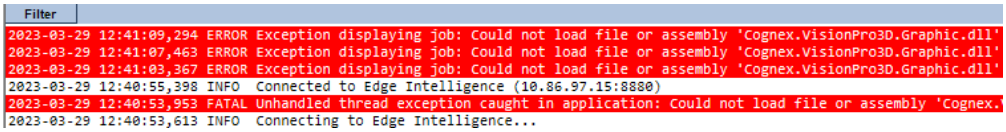


3. To prevent acquisition interruptions, set the **Interrupt Moderation Rate** of your camera to **Disabled**.



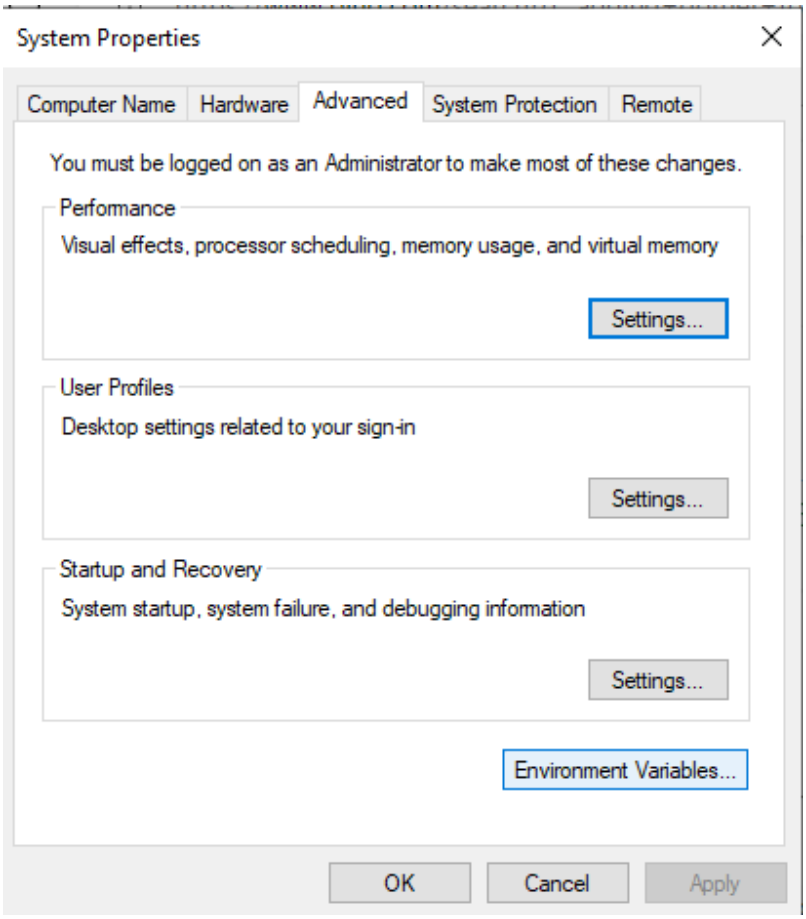
Bitflow User Assistance

A common issue if you have a BitFlow adapter involves selecting a camera after opening up the EtherInspect application. An exception in the debug log prevents you from selecting and running jobs on any cameras:

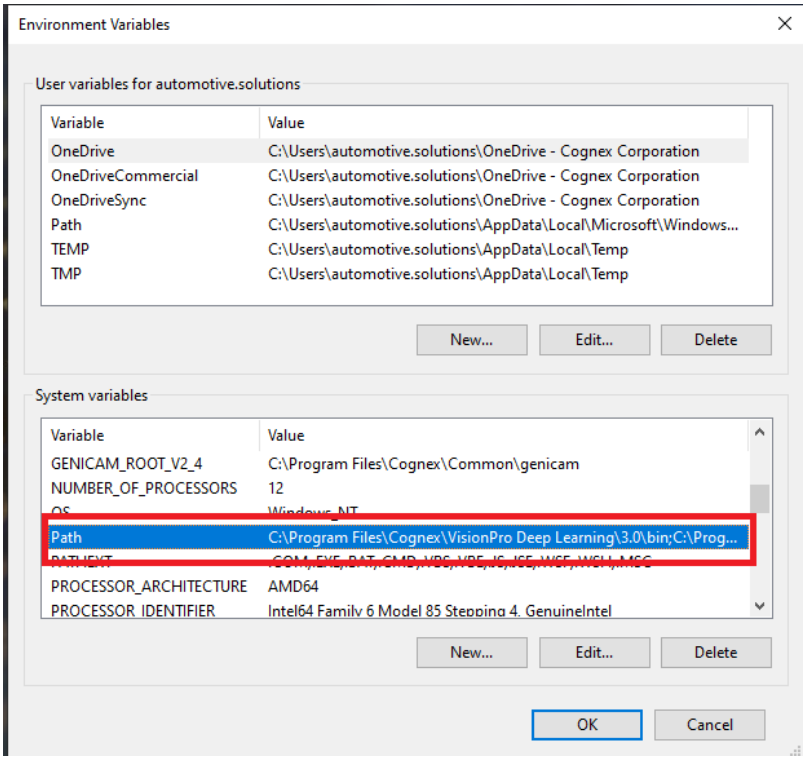


To fix this, you need to modify the environment variables in the **Path** variable of your system.

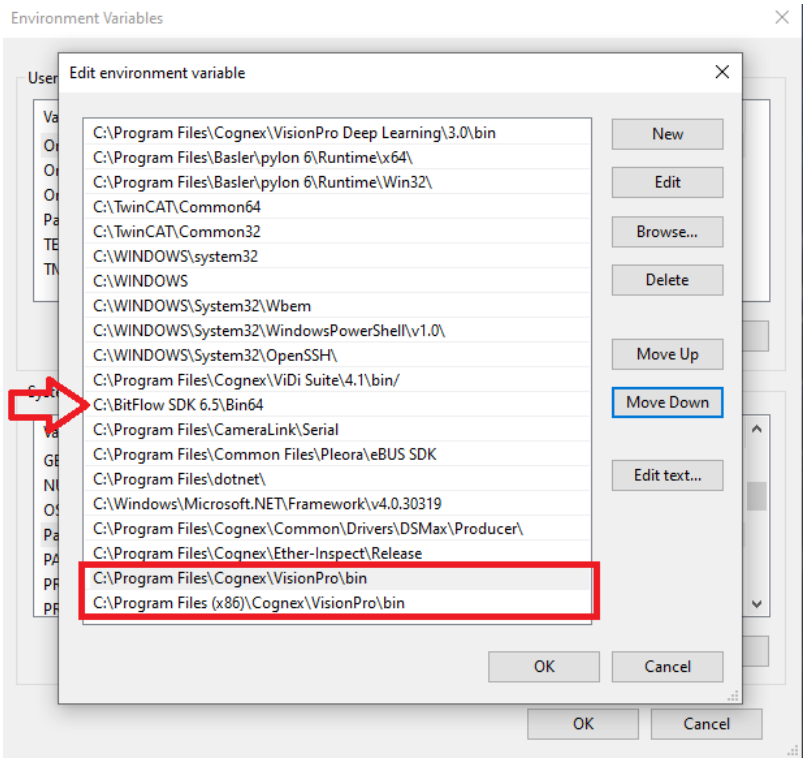
1. Go to your **Environment Variables** list.



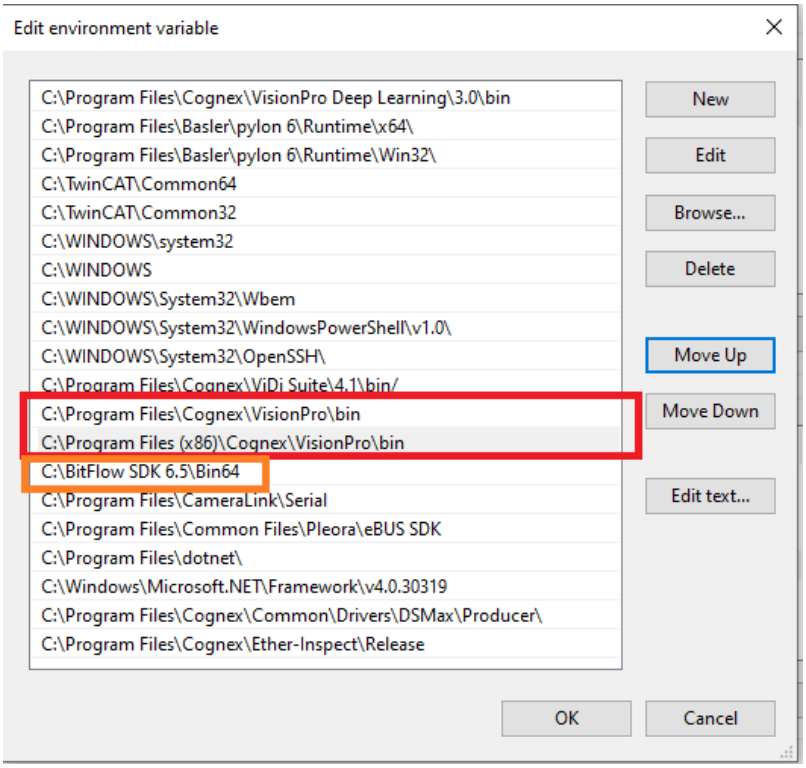
2. Select the **Path** variable under **System Variables** to display a list of variables in order from top to bottom.



3. In the list of environment variables from various applications, select the variables:
    - *C:\Program\Files\Cognex\VisionPro\bin*
    - *C:\Program\Files\(x86)\Cognex\VisionPro\bin*
    - *C:\BitFlow\SDK\X.X\Bin64*



4. Move both Vision Pro Bin Files up above the bin folder for the BitFlow SDK.

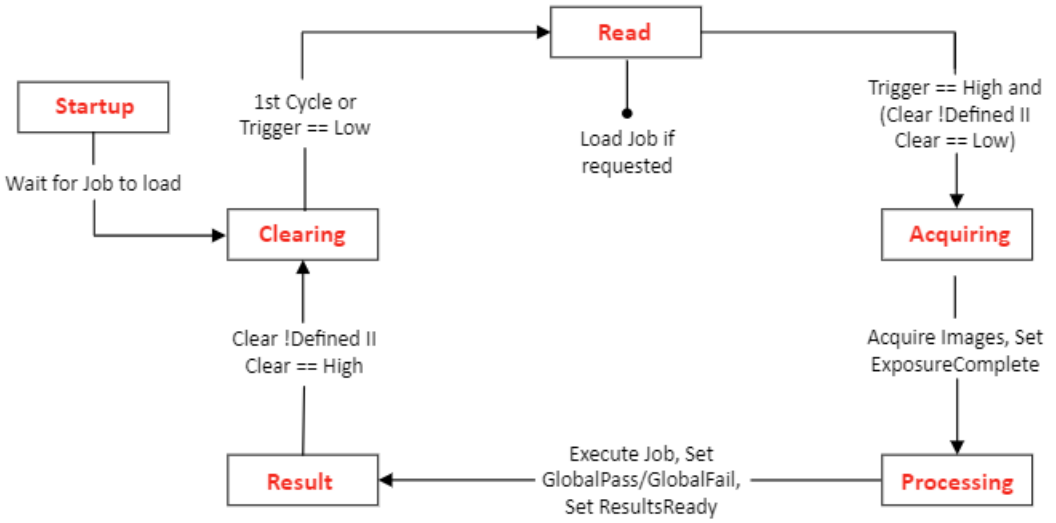


5. You can now access and load jobs onto the GigE cameras on EtherInspect.

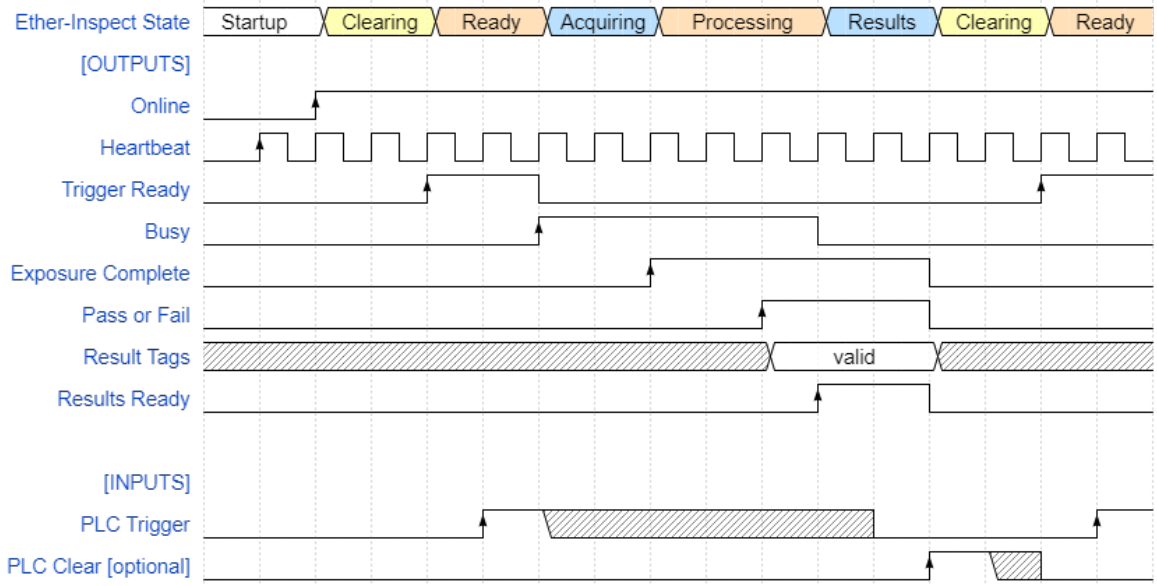
EtherInspect User Guide

Logic Flow

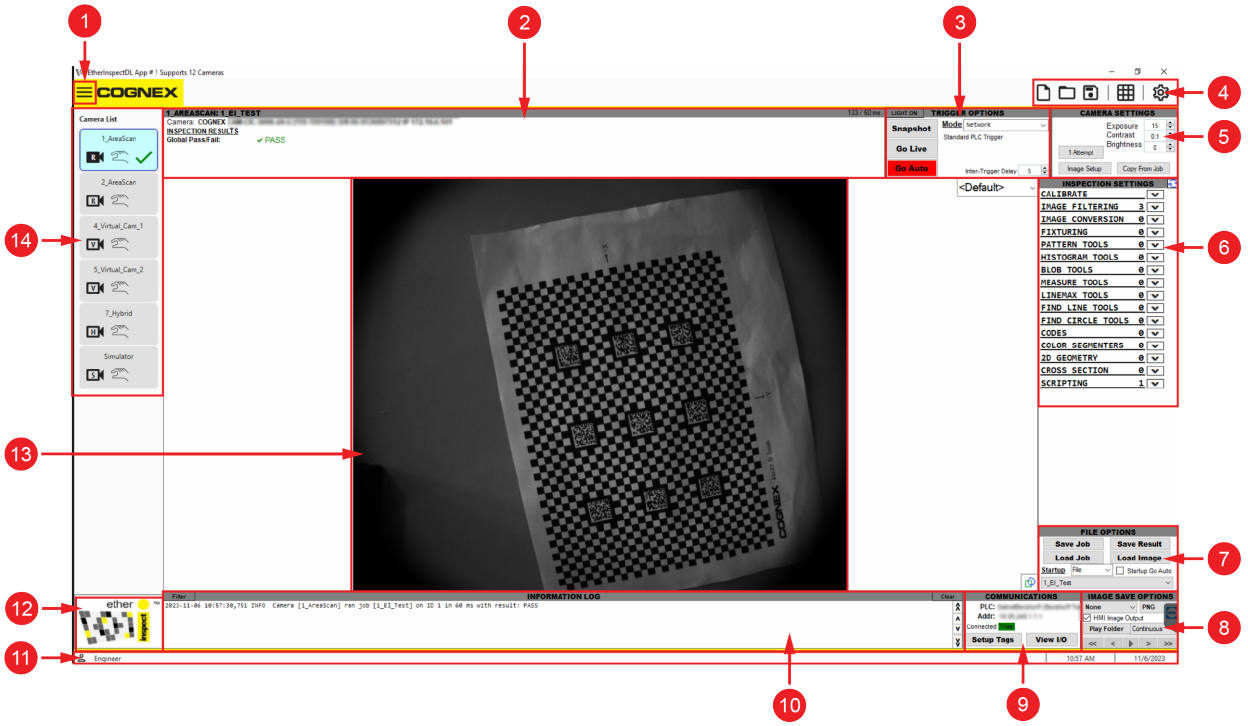
Each camera in EtherInspect is in one of six states, with the exception for Auto Mode. Auto Mode only uses five states. The diagram shows the transitions between states.



The timing chart shows a typical cycle. **PLCClear** is only valid when **ResultsReady** is on, and is dropped when **ResultsReady** is low. **PLCTrigger** is only valid when **TriggerReady** is on and is dropped when **TriggerReady** is low.



User Interface Overview



Number	Name	Description
1	*Main Menu* on page 36	Opens main context menu to run common operations in EtherInspect and retrieve diagnostics and security key info.
2	*Results* on page 39	Shows Pass/Fail results and active camera info.
3	*Trigger Options* on page 40	Capture images and adjust camera triggering mode.
4	*Toolbar* on page 42	The toolbar has the following functions: • Save and load jobs • Create new jobs from template • Enable Multi-Cam view • Go to Settings dialog
5	*Camera Settings* on page 52	Adjust camera configuration and image capture.
6	*Inspection Settings* on page 54	Add VisionPro and EtherInspect tools to use during inspection.
7	*File Options* on page 98	Saves and loads jobs, results, and images. Can also set startup job and set AUTO mode on startup.

8	*Image Save Options* on page 99	Saves images to a folder after inspection and playback.
9	*Communications* on page 100	Configure PLC servers and tags and display I/O information.
10	*Information Log* on page 107	Shows current logs. Expand to filter and display certain logs from session.
11	*Status Bar* on page 108	Contains info on user mode, date, and time.
12	*Versions Display* on page 108	Opens versions dialog for more information on VisionPro Deep Learning/ViDi and VisionPro versions.
13	*Image Display* on page 109	Shows images after acquisition with graphics.
14	*Camera List* on page 112	List of all cameras on the network and the status.

EtherInspect now contains two UI options:

- Modern: Enhanced and rebranded UI with a similar layout to legacy but a simpler style.
- Legacy: Original EtherInspect 11 UI prior to version 11.4.3-0.

The default UI is Modern. You can change this setting in *Setup/Configuration* > *Common* > *Default UI*.

Cameras in EtherInspect

EtherInspect uses a built-in VisionPro library to access all cameras on your local network. Each one is given a different thread and runs separately from other cameras. Each camera can be given a different job for processing.

EtherInspect supports the following camera types:

Icon	Type	Function
	Real	A real camera is the main thread using a physical camera on the network. A real camera can have a set of jobs and configurations. Real cameras can be further extended with copies called virtual cameras.
	Virtual	A virtual camera is a separate thread using a real camera but has an individual set of jobs and configurations. Use a virtual camera to run a different job from the real camera without altering the settings.
	Hybrid	A hybrid camera combines two or more real cameras and merges them into one. Calibration is required if you want to stitch the images together. A hybrid camera allows for a greater field of view than a singular camera.

What is a Job?

A job is an inspection process used to complete tasks. The tasks range from finding the location of a robot wheel to go and pick the robot wheel up to inspecting a tray of items for defects and abnormalities. Each job has an individual set of camera settings, PLC communication settings, trigger options, and vision tools to complete the tasks given.

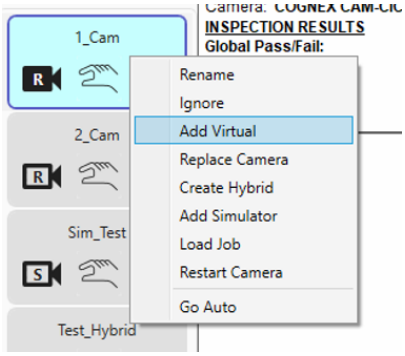
What is a Template?

A template in EtherInspect is a preset list of vision tools used for particular applications. For more information on templates and the type of templates, refer to the EtherInspect Template Documentation.

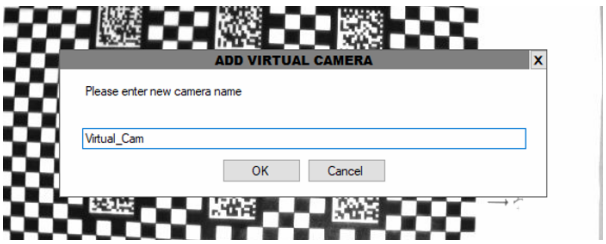
Create a Virtual Camera

Use a virtual cameras to obtain images from the same camera but with different settings and tools.

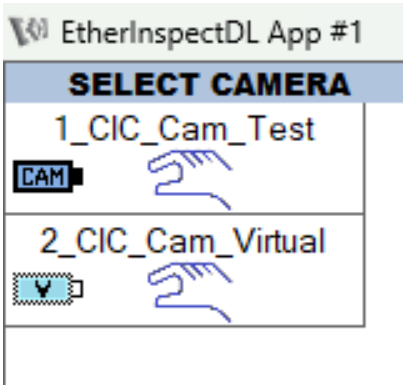
1. Right-click the camera under **Select Cameras**. Click **Add Virtual.**



2. Name your virtual camera. Make sure the name is unique from any existing cameras. Click **OK**.



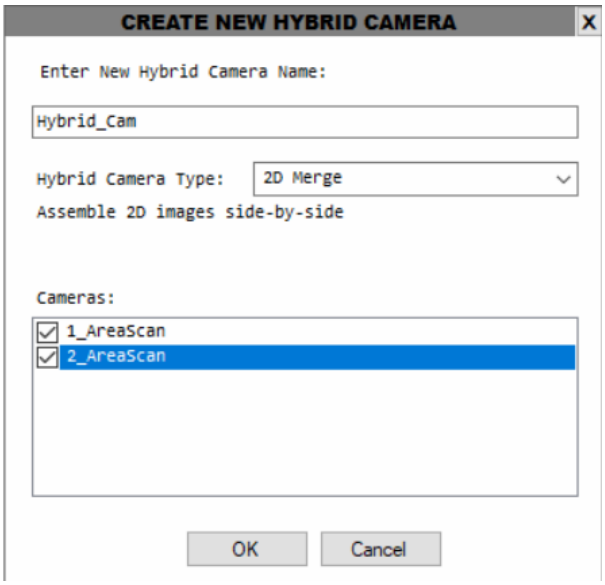
3. After you create the virtual camera, you can set up a template for the virtual camera and edit the settings to be different from your original camera. You can also use different VisionPro tools.



Create a Hybrid Camera

Create a hybrid camera from 2D cameras/sensors (2D merge) or 3D cameras/sensors (3D merge).

1. Right-click on a real camera and select **Create Hybrid**.
2. A dialog box pops up. Select which cameras to merge.
3. Name the hybrid camera, and select the type of hybrid camera to create from the drop down menu.

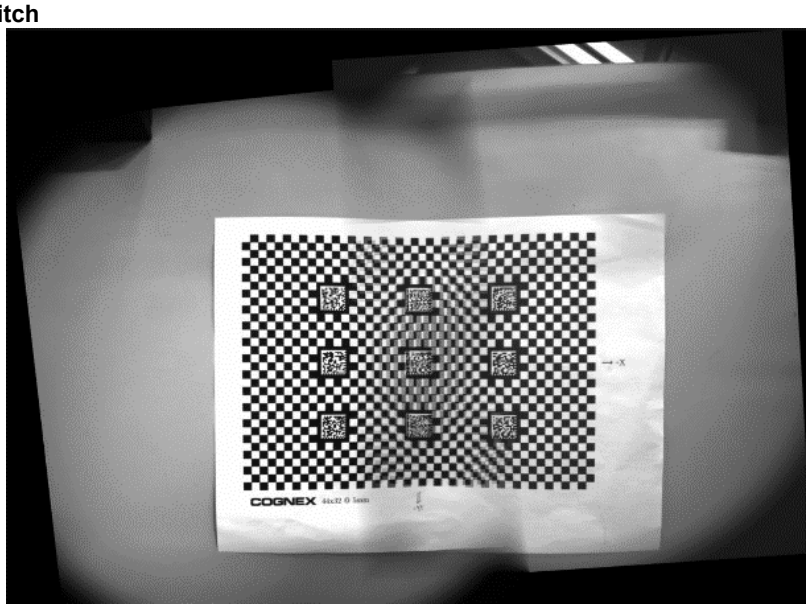


The hybrid camera displays the images from all of the cameras as one image. A hybrid camera has several operation modes which affect how the images are displayed.

The operation modes are:

- **Calibration Stitch**: Stitches images from all of the cameras together through calibration.
- **Tile Horizontal**: Displays images side-by-side.
- **Tile Vertical**: Displays images from top to bottom.
- **Tile Grid**: Displays images in grid.

Set the Calibration Stitch



Calibration Stitch mode runs calibration using fiducial points in the image. The fiducial points serve as reference points to run image stitching. You need a calibration plate or grid to run the calibration stitch.

The calibration modes are:

- Linear
- PerspectiveAndRadialWarp
- LineScanWarp
- LineScan2DWarp
- SinTanLawProjectionWarp
- ThreeParamRadialWarp
- NoDistortionWarp

To change the calibration mode:

1. Open the **User Settings** dialog.
2. Go to *Cameras* > *{HybridCameraName}* and set the operation mode to **CalibrationStitch**.
3. Set the Calibration Mode and enable the setting **Calibrate Next** to run calibration on the next acquisition.

Note: A calibration grid needs to be in view of all the cameras to run calibration successfully.

File Locations

Camera Settings Folder

The CameraSettings folder is located at *C:\ProgramData\Cognex\Ether-Inspect\CameraSettings*. The folder contains:

- CAM configuration files
- Jobs
- Verification images
- Backup files (.BAK, .BAKJSON)
- Common configuration file (Common.cfg)
- Server configuration file (server.cfg)
- Setup configuration file (Setup.cfg)

Logs Folder

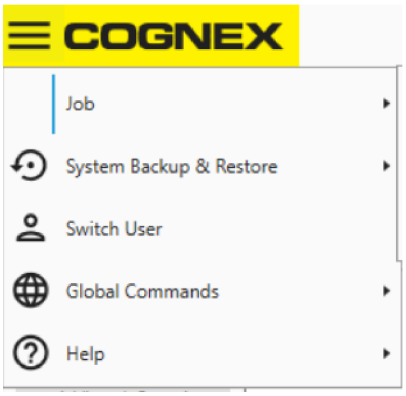
The Logs folder is located at *C:\ProgramData\Cognex\Ether-Inspect\EILogs* and contains logs generated by EtherInspect during runtime.

Server Logs Folder

The ServerLogs folder is located at *C:\ProgramData\Cognex\Ether-Inspect\ServerLogs*. The ServerLogs folder contains logs generated by the EtherInspect Server which monitors EtherInspect jobs that have started and the state during runtime. You can use the logs to troubleshoot Auto Restart events.

Main Menu

The Main Menu contains commonly-used features in EtherInspect.



Name	Description
Job	Load, save, and import jobs.
System Backup and Restore	Create backup and restore from backup.
Switch User	Switch user mode.
Global Commands	Run general commands on multiple cameras.
Help	• About (Opens Versions Dialog) • Documentation (Opens EtherInspect documentation) • Save Security Key Info (Saves license dongle information) • Save Diagnostic Info (Saves diagnostics info from currently loaded job)

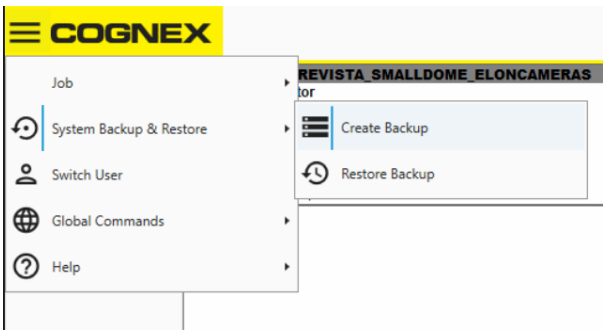
System Backup and Restore

Use system backups to restore EtherInspect to a previous point in the event of application corruption or any application-breaking change.

The two types of system backups are:

- User-made backups: Created manually and named by the user.
- Automatic backups: Created prior to deleting a real or virtual camera.

The **System Backup & Restore** options are found in the Main Menu.



When creating a backup, you are prompted to add a comment describing the backup and chose a location to save the backup. Creating a backup saves the current version of the CameraSettings directory to the target location.

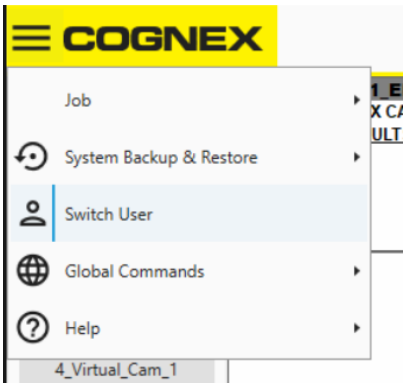
When restoring a backup, EtherInspect requires a restart of the application to load in the previous CameraSettings folder. You need to manually restart the application.

User Modes

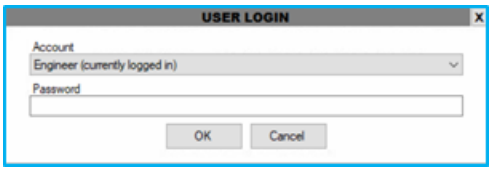
The four user modes in EtherInspect are:

- **Operator**: Can only view the camera feed, monitor communication, and view logs.
- **Supervisor**: Can see only certain categories of tools, serving as the midway point between Operator and Engineer.
- **Engineer**: Has full access to tools and communications and can define what categories are visible.
- **Integrator**: Has full access to tools and communications.

Change the user mode in the main context menu. Select **Switch User**.



The dialog prompts you to choose the user mode followed by a password (if the target user is not the Operator).



Global Commands

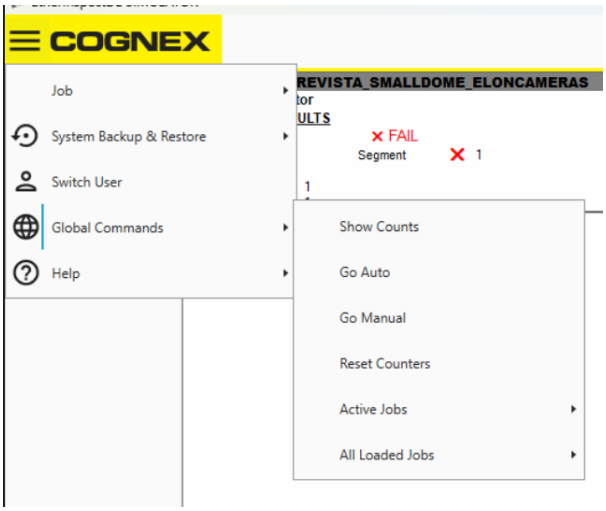
Allows you to run general EtherInspect commands on multiple cameras.

Global commands can run for all Active jobs or all Loaded jobs. **Active jobs** include jobs that are actively running on each of the cameras in the current instance. **Loaded jobs** include all the jobs that are loaded into the cameras in the current instance.

The global commands include:

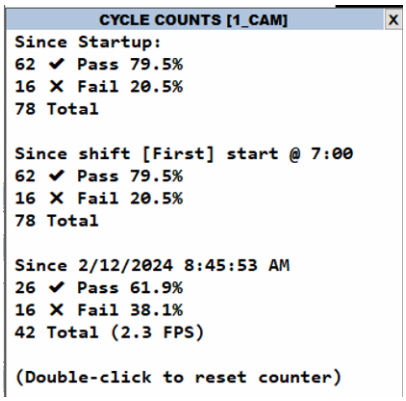
- Showing the cycle counts dialog for pass/fail jobs during a particular shift or since start up.
- Going into AUTO mode.
- Going into MANUAL mode.
- Resetting the counters in the cycle counts dialog.

- For Active and Loaded jobs:
  - Setting image save option.
  - Changing the image save format.
  - Enabling or disabling the HMI image output.
  - Saving jobs.



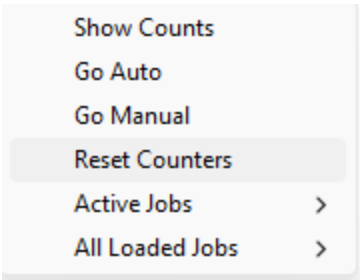
Cycle Counts Display

Shows counts of PASS/FAIL since start up and since the start of a shift.



Reset Counters

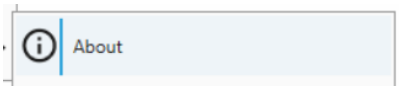
Reset all the PASS/FAIL inspection count back to 0. You can also reset the counters using the **Cycle Counts Display** by double-clicking.



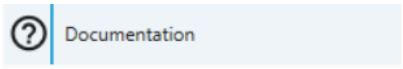
Help Menu

View firmware versions, documentation, save license, and diagnostic information in the **Help Menu**.

The **About** button opens the versions dialog, allowing you to see firmware versions for VisionPro, VisionPro Deep Learning/ViDi, and EtherInspect.



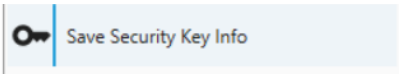
The **Documentation** button opens the EtherInspect documentation in a separate window.



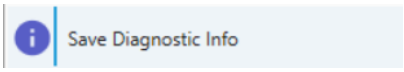
The **Save Security Key Info** button saves information about the currently connected EtherInspect license dongle. The two files saved are:

- .txt file containing license bit information
- .WibuRaC file for updating license dongles

The files are packaged in a .zip file with the file path you specified.

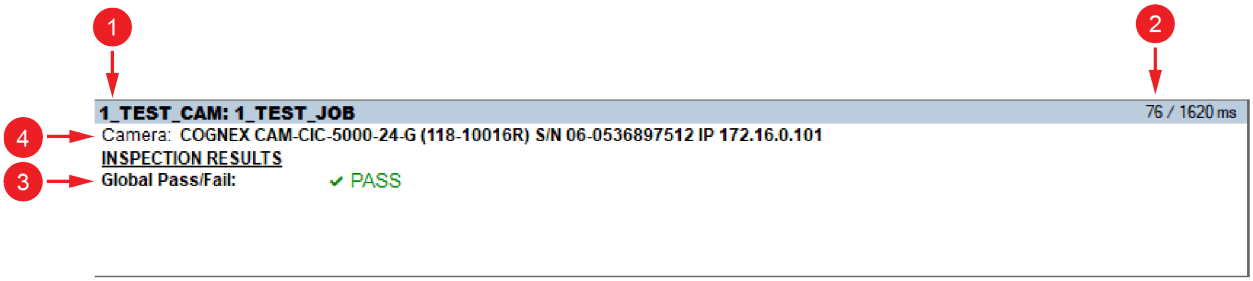


The **Save Diagnostic Info** button saves information you can send to Cognex IT Support for troubleshooting issues. The information includes results for the current job, camera settings or configuration, and log files from the current day.



Results

Shows inspection results, camera model, and acquisition times.



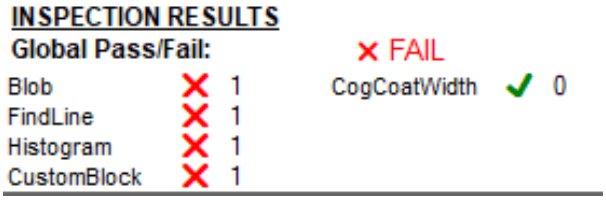
Number	Name	Description
1	Camera / Job Name	Name of camera is on the left, and the name of job is on the right.

2	Acquisition / Processing time	Acquisition time is on the left, and processing time is on the right. Acquisition time is the time spent in the Acquisition state during inspection, which includes: • Waiting for an external trigger (if applicable) • Exposure time • Image transfer time • Time to change state to Processing Processing time is the time spent processing the job including running scripts. **Note**: The time calculations do not include saving images.
3	Camera Info	Camera Information including model, serial number, and IP address.
4	Inspection Results	Includes global inspection results and results for specific tools used in inspection.

Inspection Results

The Inspection results section shows all of the inspection results for each tool and the result for the global inspection. When a tool type fails, it displays a number to the right of the result indicating which tool failed.

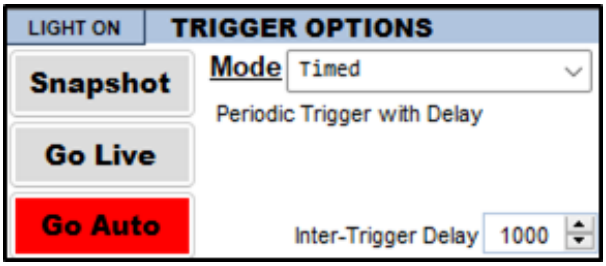
In this example, Blob Tool 1 failed:



Trigger Options

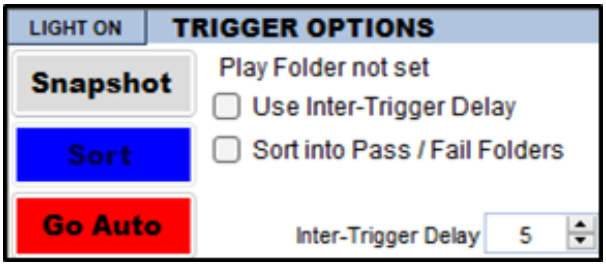
In the Trigger Options settings, you can change the trigger type, inter-trigger delay, and Auto/Manual mode.

Camera Trigger Options



Name	Description
Toggle Light	Triggers TurnLightON bit in status word, which you can use to trigger light in MANUAL mode.
Trigger Mode	Determines when image is captured in AUTO mode.
Snapshot	Captures a single image from the camera.
Go Live	Switches to Live Acquisition mode.
Go Auto	Automatically captures images based on trigger mode.
Inter-Trigger Delay	In Timed or Network / Timed mode, determines time between image captures (in ms).

Simulator Trigger Options



Name	Description
Use Inter-Trigger Delay	Enables or disables the use of an inter-trigger delay between images.
Sort into Pass / Fail Folders	Determines if Sort copies images into new PASS / FAIL subfolders.
Sort	Runs through images in the play folder.

Trigger Modes

Trigger modes determine the rate at which image acquisition occurs. The rate is determined through methods such as inter-trigger delays, triggers within a PLC, and external triggers based off sensor output.

Trigger Mode	Description
Network Trigger Mode	Network trigger mode uses PLC logic to trigger on command. This occurs whenever the PLCTrigger status bit goes high in the logic and runs the image acquisition process. The PLCTrigger bit can only trigger when the system is online and the TriggerReady bit is high.

External Trigger Mode	External trigger mode uses an external sensor output to trigger the cameras. The mode can be set up within PLC where a sensor such as a photo-eye can trigger the camera to start image acquisition.
Timed Trigger Mode	Timed trigger mode uses the inter-trigger delay under Trigger Options to add a time delay between each image capture. The camera runs through the image acquisition process each time the delay elapses.
Network / Timed Trigger Mode	Network / Timed trigger mode is a combination of network and timed trigger mode. The mode uses the inter-trigger delay to space out the timing between each PLC trigger.
Continuous Trigger Mode	Continuous trigger mode runs image acquisition continuously with no delays between each trigger. Use the mode for fast acquisition.
Fast External Trigger Mode	Fast External trigger mode is the same as external trigger mode except it runs with little to no delay between triggers.
Network / External Trigger Mode	Network / External trigger mode is a combination of network mode and external trigger mode where, when the external trigger outputs true, the PLC logic is executed.
Network / Continuous Trigger Mode	Network / Continuous trigger mode is a combination of network and continuous trigger mode where the delay between PLC triggered acquisition is 0.

Toolbar

Icon	Function	Description
	Create a new job (Select Template)	Click the icon to open the New Job dialog. From here, you can select a template to create a new job.
	Load a job (Open)	Click the icon to load a job from disk.
	Save a job	Click the icon to save the currently loaded job on the current camera.
	Going into MultiCam View	Click the icon to open MultiCam View. For more information, see MultiCam view.
	Accessing User Settings	Click the icon to open the *User Settings* on page 42 dialog, which allows you to configure settings related to EtherInspect.

User Settings

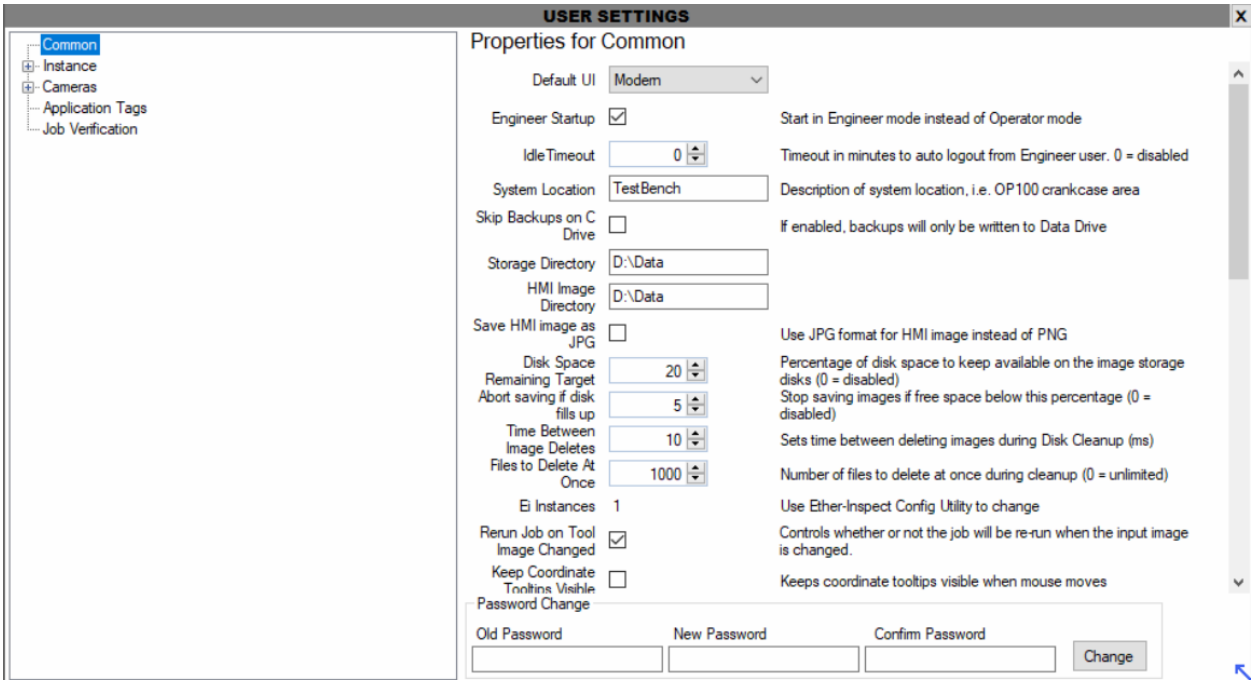
Allows you to edit the following application settings:

- Common Settings
- Instance Settings
- Camera Settings
- Application Tags
- Job Verification

Note: Some settings require an application restart to apply the settings.

Note: Close the settings dialog to apply changes.

Common Settings

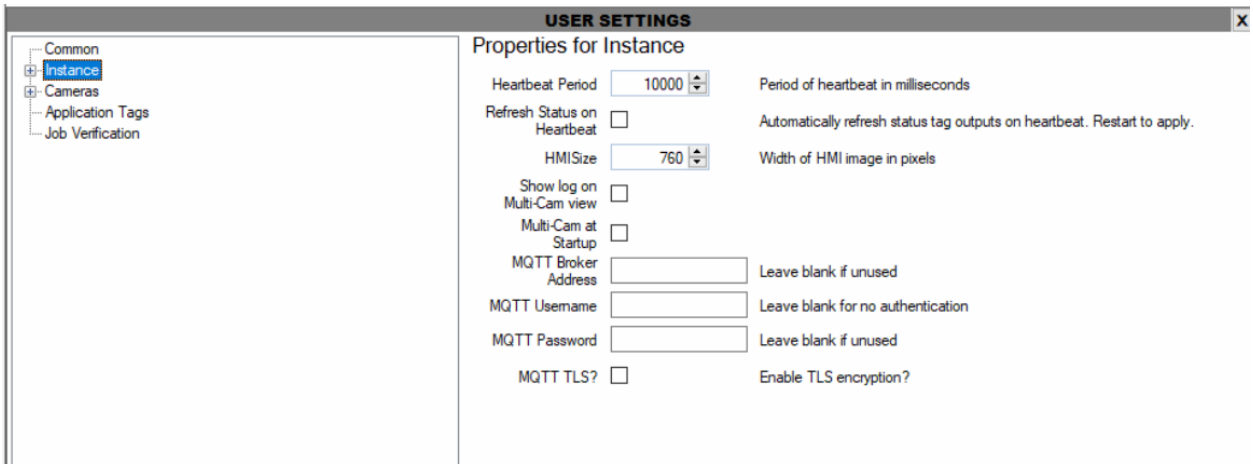


Setting	Description
Default UI	Sets the default UI that EtherInspect loads into when starting the application. For information on the UI options, see the User Interface Overview.
Engineer Startup	By default, EtherInspect starts in Engineer mode but you can change the mode to start in Operator mode.
Idle timeout	Engineer mode automatically logs out after a given timeout. If set to 0, the timeout is disabled.
System Location	The Edge Intelligence / HMI shows this name and description of system.
Skip Backups on C Drive	When enabled, backups are written to the data (D) drive instead of C drive.
Storage Directory	Configure the path where data is stored. The default path is *D:\Data*. **Note**: Data stored within the directory includes PASS / FAIL images during inspection. The images are saved with the job results for that image (in the form of an XML file). This only happens when image saving is done. For more information, see *Image Save Options* on page 99.
HMI Image Directory	Configure the path where HMI images are stored. The default path is *D:\Data*. **Note**: The path configuration works similarly to the storage directory option where PASS / FAIL images from the HMI are stored. The information includes any job results associated with the image.
Save HMI Image as JPG	The default HMI image format is PNG. Set this option to save it as JPG.

Disk Space Remaining Target	Percentage of disk space to keep available on the image storage disks. **Note**: If you exceed the target, image storage disks delete the oldest images and results.
Abort Saving if Disk fills up	Stops saving images if the free space is below the percentage (0 = disabled).
Time Between Image Deletes	Specifies time (in ms) between deleting images during disk cleanup.
Files to Delete at Once	Sets the number of files to delete at once during cleanup (0 = unlimited).
EI Instances	Maximum number of EtherInspect instances on current device. Read-only. **Note**: You can only change the setting in EtherInspect Configuration Utility.
Rerun Job on Tool image changed	Controls whether the job re-runs when the input image has changed.
Keep Coordinate Tooltips Visible	To make sure the coordinate tooltip stays on screen, check the **Keep Coordinate Tooltips Visible** checkbox. By default, the coordinate tooltips only appear when you double-click on the Camera display and disappear when you move the mouse.
Simulators Ignore PLCs	If checked, in simulator mode, EtherInspect does not connect to the PLC.
Enable Common Jobs	To share one copy of a job with multiple cameras, check **Enable Common Jobs**. You can save a job in the CommonJobs directory by right-clicking the **Save Job** button. You can pick a common job for your startup job. The prefix “common” denotes a common job.
Enable 3D Display –	Enables a 3D display for 3D cameras by default.
Run GC every N images	After N amount of images are acquired, run Garbage Collection to free up memory.
Log Memory Usage	Keeps track of memory usage in information log.
Load Job from Disk, if changed	When enabled, the feature runs when requested by the PLC. If the job changes, EtherInspect uses the copy of the job on disk instead of the copy in memory.
Save JSON with Job	Saves JSON file to disk when job is saved, containing information on job configuration.
Web Server Address	The IP address form of a URL to set the web server to retrieve images from inspection.

Use Manual – Prefix	If enabled, EtherInspect adds the prefix *"Manual-"* to saved image files when taking a snapshot in MANUAL mode.
Use Acquisition ID in File Name	Appends AcquisitionID to end of image file name.
Display Update Limit (ms)	Any image that arrives faster than this rate skips updating the display.
ViDi Max Process Count	Number of ViDi processes allowed to use the same GPU. **Note**: Anything greater than one increases tool processing time but allows more tools to run simultaneously.
ViDi Optimized Memory	Amount of memory to reserve on GPU in MB (0 = disabled).
Extra Debug Logging	EtherInspect generates additional log messages if requested by Cognex Tech Support.
Log File Retention (days)	Number of days to keep logs (3 – 90), defaults 90.
Edge Intelligence Address	If set to the address of an Edge Intelligence box, a server is started with a web interface allowing you to monitor EtherInspect systems from a remote PC. **Note**: You must have an Edge Intelligence box set up. **Note**: The IP address of the Edge Intelligence box is the same as the server.
Reboot on Firmware Upload	If enabled, firmware updates received from the web interface are installed automatically and reboots EtherInspect with the updated firmware.
Hide MultiCam Result	Disables pass/fail graphics in MultiCam view and camera select display.
Show Inspection Settings in Auto	Displays inspection settings in AUTO mode.
Allow Empty Device SN	Uses device name if serial number is empty.
Password change	Default old password for Engineer account is "password". Default old password for Integrator account is "integrator". You have an option to set the new password for Engineer/Integrator account.

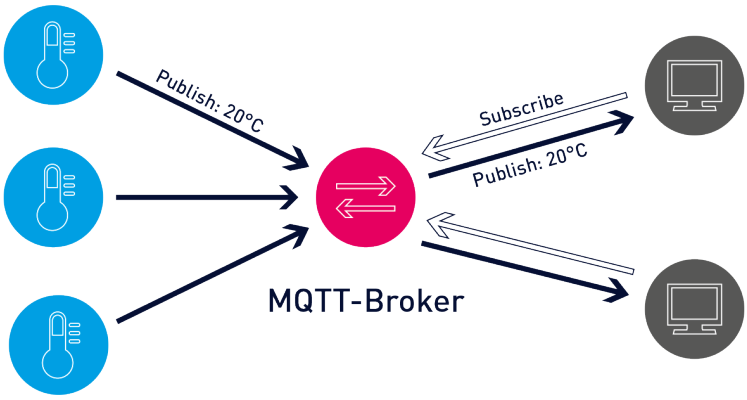
Instance Settings



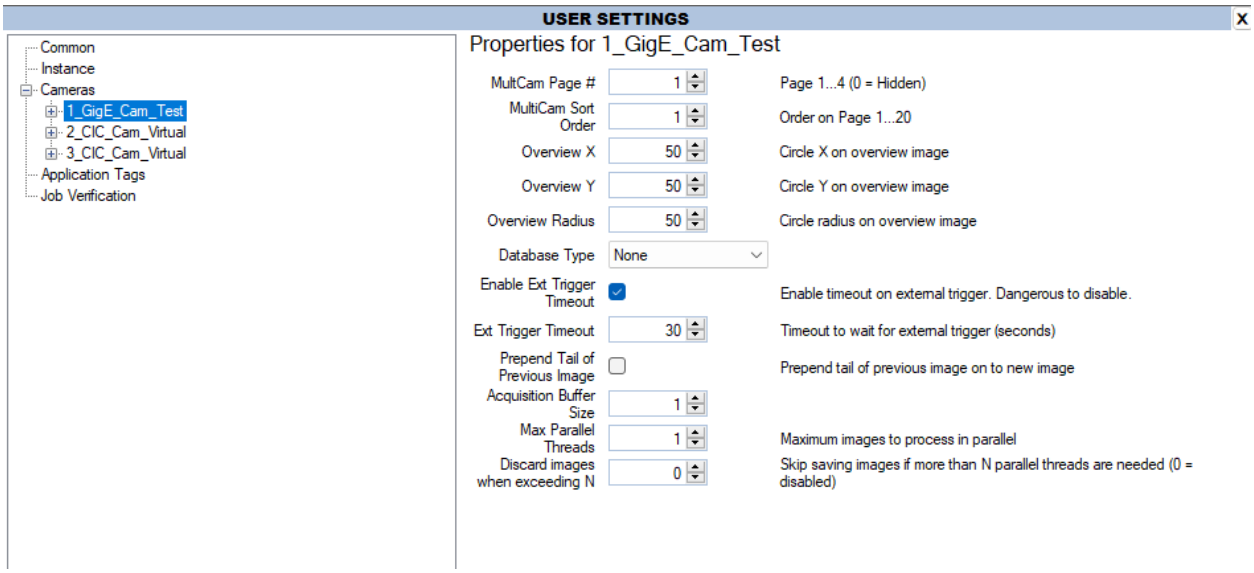
Setting	Description
Heartbeat Period	Period of heartbeat in milliseconds. When the period ends, the network pings the camera to verify it still exists. The PLC can also receive the heartbeat period to monitor camera status. **Note**: Cognex recommends 30000 ms or lower.
Refresh Status on Heartbeat	Automatically refreshes status of PLC tag outputs when heartbeat ends. You must restart the application to apply the setting. **Note**: Cognex does not recommended automatically refreshing because it causes additional PLC traffic.
HMI Size	Configure the width of HMI image in pixels. The height is automatically calculated based on the aspect ratio or the image.
Show log on Multi-Cam view	Shows the information log when MultiCam view is on.
Multi-Cam at Startup	If enabled, the application start up sets the MultiCam view.
MQTT Broker address	IP address of MQTT broker. Leave the address blank if unused.
MQTT Username	Username to connect to MQTT broker. Leave it blank for no authentication.
MQTT Password	Password to connect to MQTT broker. Leave it blank if unused.
MQTT TLS?	Enable MQTT TLS encryption.

MQTT

MQTT is a lightweight, publish / subscribe, machine to machine network protocol used to queue, send, and receive messages. You can use MQTT in remote locations with limited resources and bandwidth for devices.



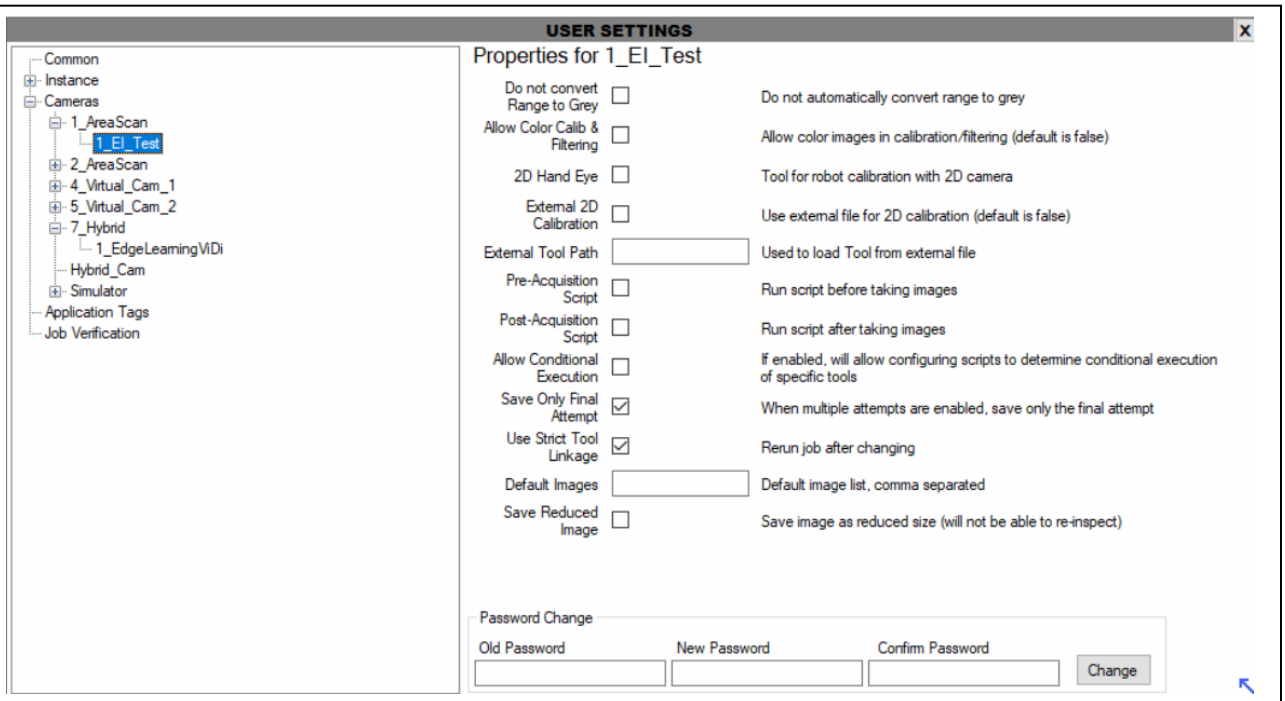
Camera Settings



Setting	Description
MultiCam Page #	Page number on which the camera is shown when MultiCam view is on. The pages can be between 1 to 4. Set 0 hides the camera from MultiCam view. Keep clicking on “MULTICAM VIEW >> F1” to navigate through multiple pages.
MultiCam Sort order	Configure the order of the camera on the MultiCam page assigned to it. Order number can be between 1 to 20. Camera ordering starts from top left to bottom right. More information can be found in *MultiCam View* on page 112.
Overview X	Found in the EtherInspect viewer application. A fault animated circle is shown at this coordinate on Overview image.
Overview Y	Found in EtherInspect viewer application. A fault animated circle is shown at the coordinate on Overview image.
Overview Radius	Found in the EtherInspect viewer application. A fault animated circle is animated up to configured radius.

Database Type	Configure which database to send inspection data. Based on the selected type, more configurations appear.
Enable External Trigger Timeout	Enable timeout on external trigger. **Note**: Disabling can cause unresponsiveness.
Ext Trigger Timeout	Timeout to wait for external trigger in seconds.
Prepend Tail of Previous Image	Prepend tail of previous image onto new image.
Acquisition Buffer Size	Values greater than one allow acquiring a new image while the first is processing.
Max Parallel Threads	Maximum number of images to process in parallel.
Discard Images when Exceeding N	Skip saving images if more than N parallel threads are needed (0 = disabled).

Job Properties



Setting	Description
Do not Convert Range to Grey	Does not convert range to greyscale.
Allow Color Calib & Filtering	Allows color images in calibration or filtering. Default is false.
2D Hand Eye	Enables tool for robot calibration with 2D camera.
External 2D Calibration	Use external file for 2D calibration. Default is false.
External Tool Path	Path used to load tool from external file.
Pre-Acquisition Script	If enabled, you can run a script before acquisition. The Pre-Acquisition Script is not the same as the Pre-Job Scripting tool. If the Post-Acquisition Script is enabled, you can run a script after acquisition. Uses same scripts as pre-acquisition setting. See *Pre-Acquisition* on page 49.

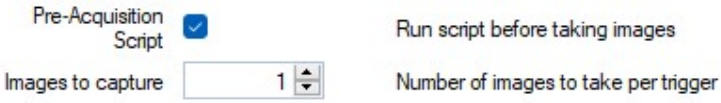
Allow Conditional Execution	If checked, allows configuring scripts to determine conditional execution of specific tools.
Save Only Final Attempt	When multiple acquisition attempts are enabled, save only the final attempt.
Use Strict Tool Linkage	If unchecked, allows you to control the flow of images through the various tools.
Default Images	List of default images, must be comma separated (test_img1, test_img2, test_img3).
Save Reduced Images	Saves images at a reduced size. **Note**: Enable the setting to reduce the scale, with 100% being full-sized images (Reduced Image Scale). You cannot re-inspect reduced images.

Pre-Acquisition and Post Acquisition Scripts

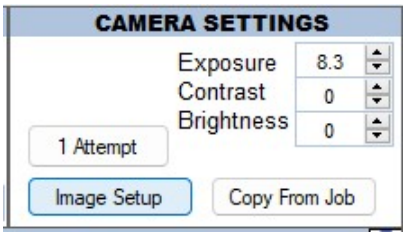
Pre-Acquisition

Pre-Acquisition scripts allow you to run a script before acquisition.

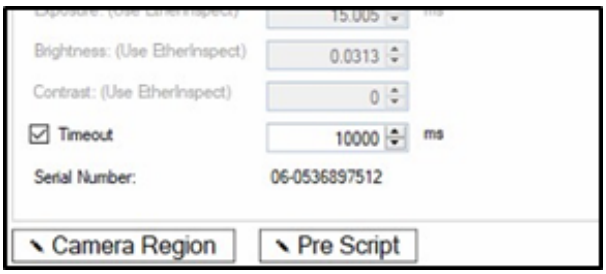
When enabling Pre-Acquisition Script, an additional setting pops up called **Images to Capture** which allows you to specify how many images to capture per camera trigger.



1. Click **Image Setup** in **Camera Settings**.



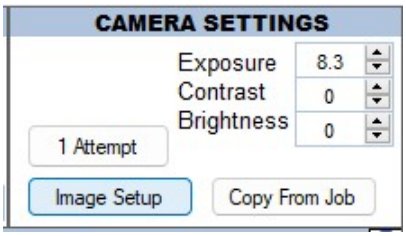
2. Select **Pre Script**. An editor opens for the Pre-Acquisition script.



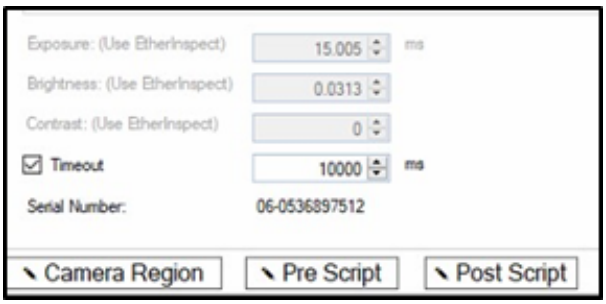
Post-Acquisition

Post-Acquisition scripts allow you to run a script after acquisition.

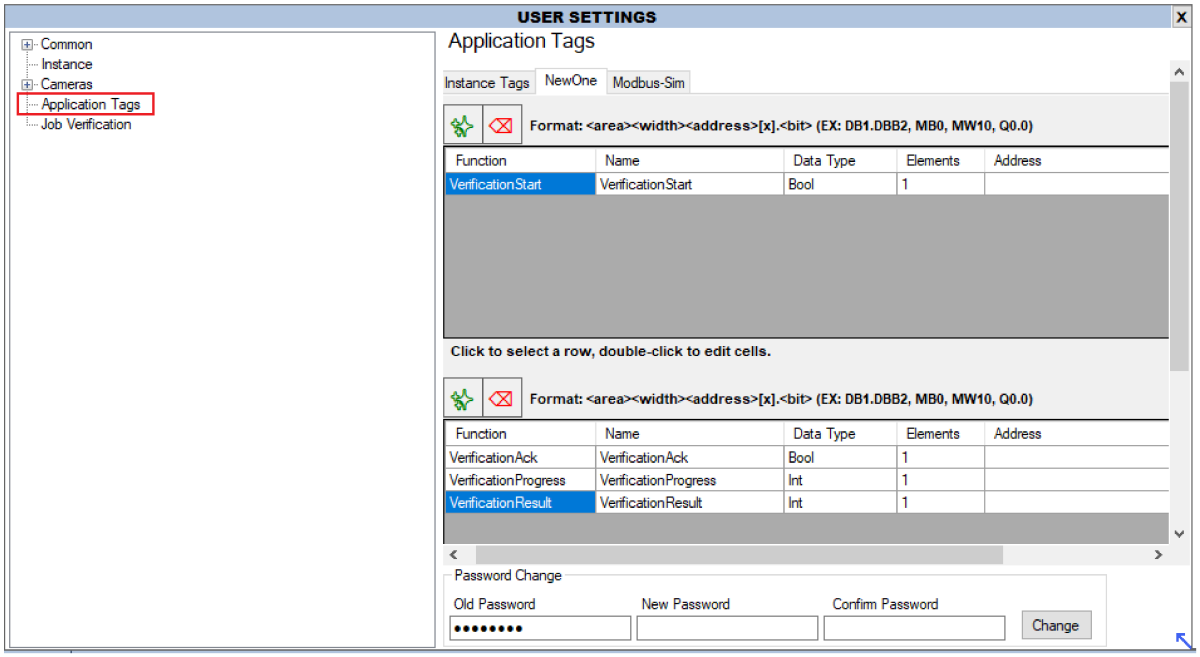
1. Enable the Post-Acquisition script. Click **Image Setup** under **Camera Settings** to edit.



2. Enabling this feature places a **Post Script** button to edit the post-acquisition script.



Application Tags

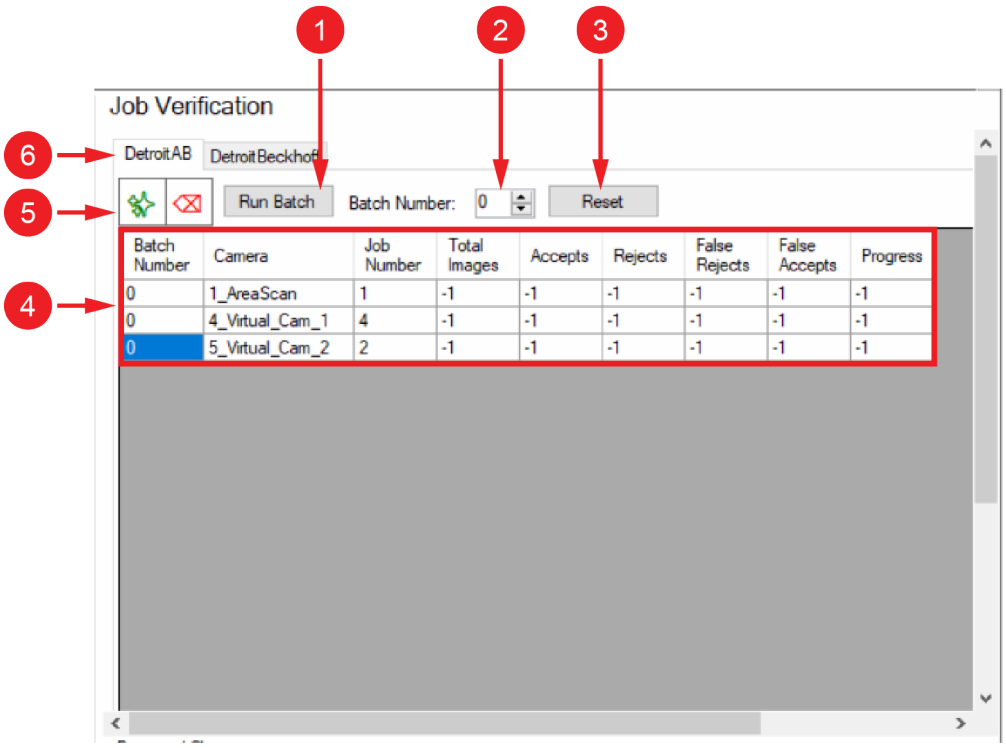


Function	Description
Verification Start	(Input, Boolean) starts the verification when the tag goes high. When EtherInspect receives the Verification start, it outputs VerificationAck (Output, Boolean). This should be used by the PLC to turn off VerificationStart.
VerificationProgress	(Output, Int) gives a 0 – 100 value for progress through the verification batch.

<table>
<tr>
<td>VerificationResult</td>
<td>(Output, Int) provide the overall success / fail for the batch:
<table>
<tr><th>Bit #</th><th>Status</th><th>Description</th></tr>
<tr><td>0</td><td>All Pass</td><td>All jobs from batch passed inspection.</td></tr>
<tr><td>1</td><td>False Accept(s)</td><td>There was a false positive in the batch verification.</td></tr>
<tr><td>2</td><td>False Reject(s)</td><td>There was a false negative in the batch verification.</td></tr>
<tr><td>3</td><td>*Unused*</td><td>Bit is not used.</td></tr>
<tr><td>4</td><td>Empty List</td><td>Empty list was provided.</td></tr>
<tr><td>5</td><td>Error</td><td>An exception/error was experienced during verification.</td></tr>
<tr><td>6</td><td>No images</td><td>No images were provided.</td></tr>
<tr><td>7</td><td>*Unused*</td><td>Bit is not used.</td></tr>
</table>
</td>
</tr>
</table>

Job Verification Testing

EtherInspect has the capability to automatically test production jobs against known Pass/Fail images. Multiple jobs can be tested, and each batch can be tested via PLC trigger and basic results are returned.



Number	Name	Description
1	Run Batch	Runs a batch of jobs based on the job number provided.
2	Batch Number	Select batch number to run job verification on.

3	Reset	Clears the values from the previous job verification and resets the values to -1.
4	Job Verification Table	Shows the number of accepts, rejects, false rejects, and false accepts for each batch.
5	Add/Remove Batch	Add and remove batches from job verification.
6	PLC	Select PLC to add or remove job verification tasks.

Add as many checks as desired. Select the camera name and Job index number. Specify the batch number for each job verification test that you are setting.

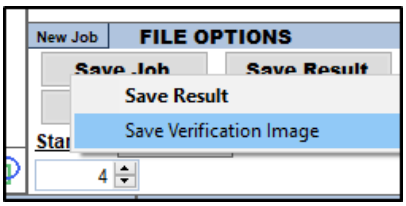
Note: If jobs are revised, the job always uses the most recent for the index number selected. -1 in a field indicates the job did not run a verification check yet.

Setting	Description
Batch Number	Multiple Job verification tests can be set to the same batch number which groups them together as one batch when running it.
Camera	Camera to run job verification off.
Job Number	Specifies the job to use within the camera.
Total Images	Total amount of images that ran during test. This is the sum of Accepts, Rejects, False Rejects, and False Accepts.
Accepts	Number of images in PASS folder that pass inspection.
Rejects	Number of images in FAIL folder that fail inspection.
False Rejects	Number of images in PASS folder that fail inspection.
False Accepts	Number of images in FAIL folder that pass inspection.

When running batches, if the batch number selected is 0, all job verification tests run. If the batch number matches the batch number of a group of job verification tests, the program runs verification on those specific verification tests.

Verification images are stored in: *C:\ProgramData\Cognex\Ether-Inspect\CameraSettings\<CameraName>\<JobNumber>_Verify_<Pass or Fail>*

You can copy images into this directory or save images to the verification batch by right-clicking **Save Result** and choosing **Save Verification image**.

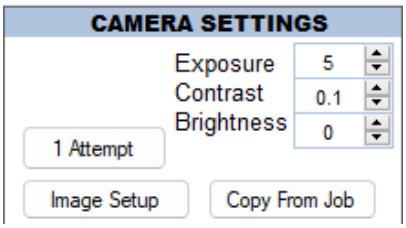


Verification tests can be executed manually by clicking the **Run Batch** button.

The PLC can initiate the execution and receive results of Verification tests.

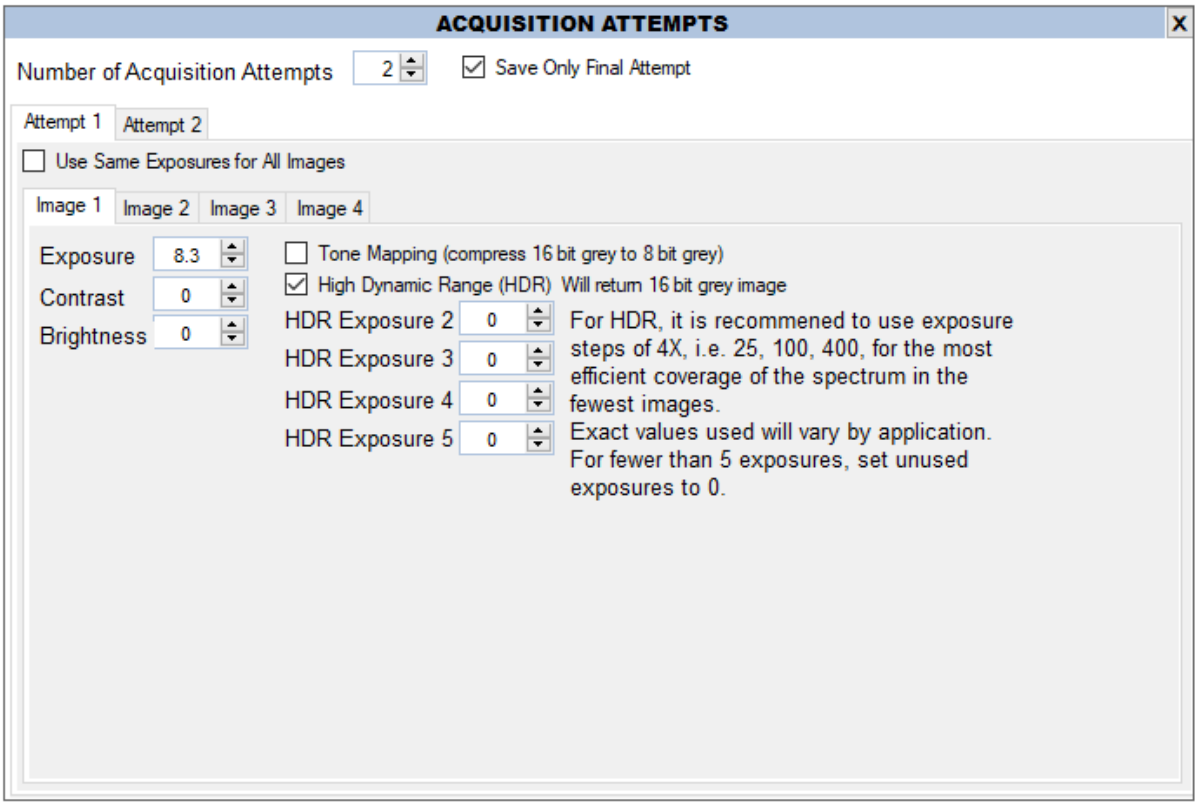
Camera Settings

You can modify the camera settings here. Exposure, contrast, and brightness are not visible if the acquisition settings are too complex to fit. Use the **Acquisition Attempts** button to configure Exposure, Contrast, and Brightness.



Name	Description
Exposure	Changes exposure on camera (Min = 0) (Max = 999).
Contrast	Changes contrast on camera (Min = 0) (Max = 1).
Brightness	Changes brightness on camera (Min = 0) (Max = 1).
Copy From Job	Copies camera settings from a different job.
Image Setup	Allows you to modify VisionPro settings of camera.
Acquisition Attempts	Add acquisition attempts with different camera settings. For each failed attempt, it goes to the next attempt and tries to acquire an image until it passes inspection or all acquisition attempts fail.

Acquisition Attempts



You can use the Acquistion Attempts features simultaneously. The features are:

- Multiple Attempts: Use multiple attempts to attempt the same job multiple times with different image settings. The camera iterates through each attempt until either one attempt results in a PASS or all attempts FAIL.

- Multiple Sensors (Hybrid Cameras): Use multiple sensors (not pictured) to configure image settings for hybrid cameras.
- Multiple Images: Use multiple images if the application requires more than one image with different lighting (such as SurfaceFX). The feature requires enabling Pre-Acquisition Scripting in the Job settings and setting Number of Images greater than one.

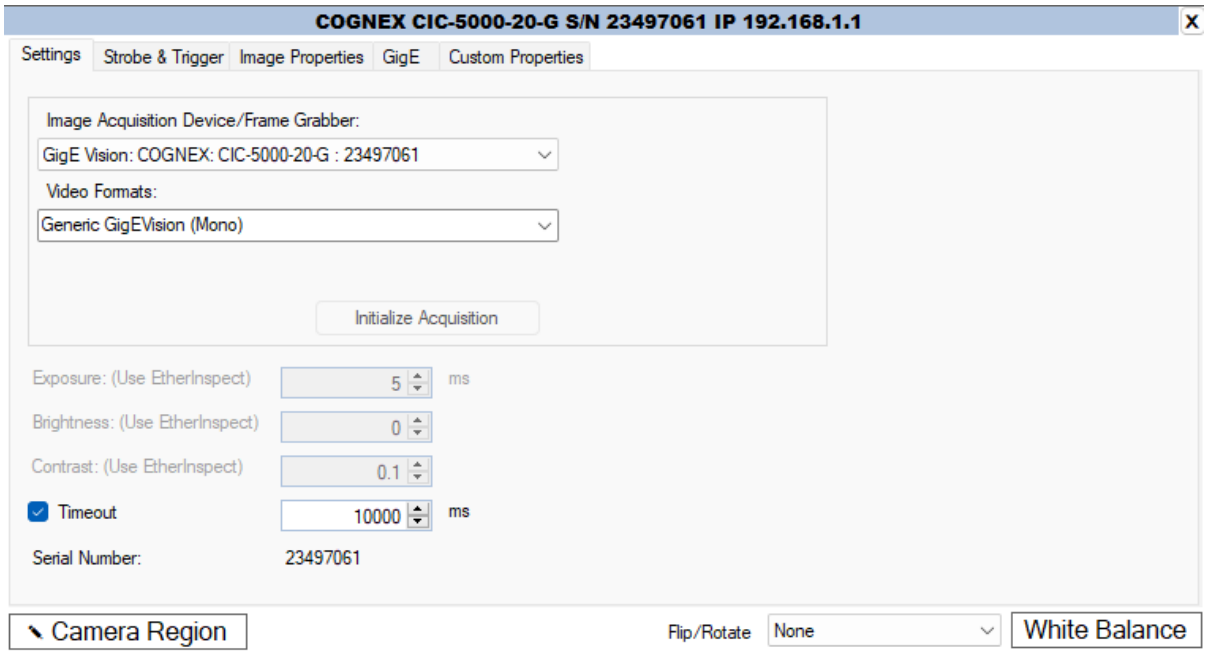
  Note: If **Use Same Exposure for All images** is selected, the tabs for Image 1, Image 2, and so on, are not visible.

- Multiple Exposures (HDR): Use multiple exposures to create a 16 bit High Dynamic Range (HDR) image. HDR images combine pixels from the different exposure levels to create an image the preserves dark and light regions without overexposing. Up to five additional exposures can be configured for HDR, for a total of six images.
- Tone Mapping: The Tone Mapping algorithm compresses a 16 bit image to 8 bits, preserving local contrast.

Image Setup

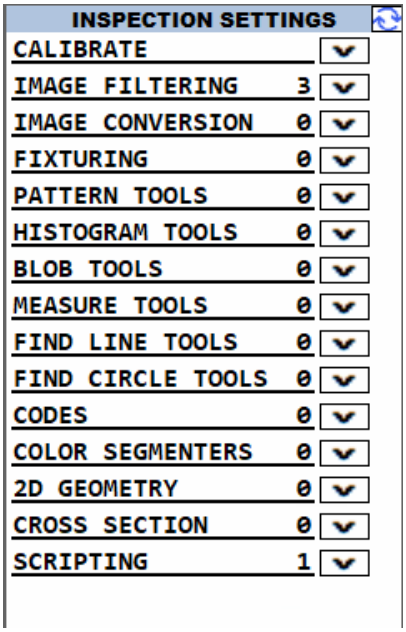
Allows you to configure the camera properties, including changing the white balance, modifying the camera region, setting timeouts, or viewing camera information.

Note: **Exposure**, **Brightness**, and **Contrast** only update when a new image is acquired.



Inspection Settings

Allows you to change settings on various tools in the current template.



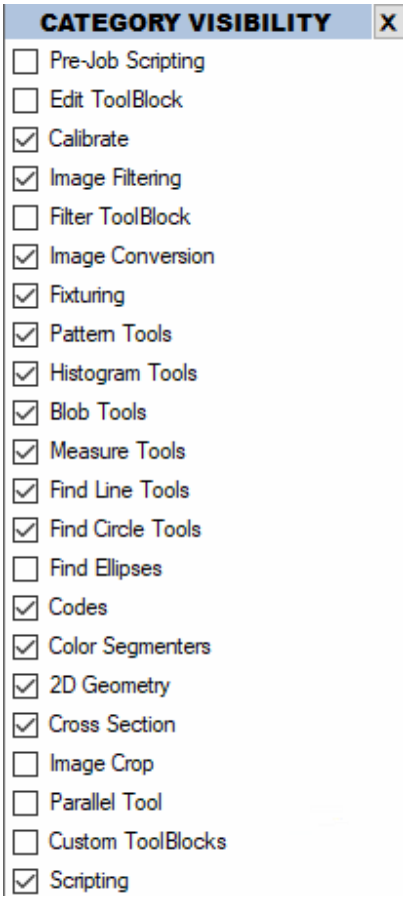
Click **Inspection Settings** at the top to open the Category Visibility Display to show or hide categories.

Click the re-run button in the upper-right to re-run the inspection using the same image.

Click next to any category to show the tool settings for the first tool in each category.

Category Visibility

Use checkboxes to show or hide each category listed in the Inspection Settings Display.

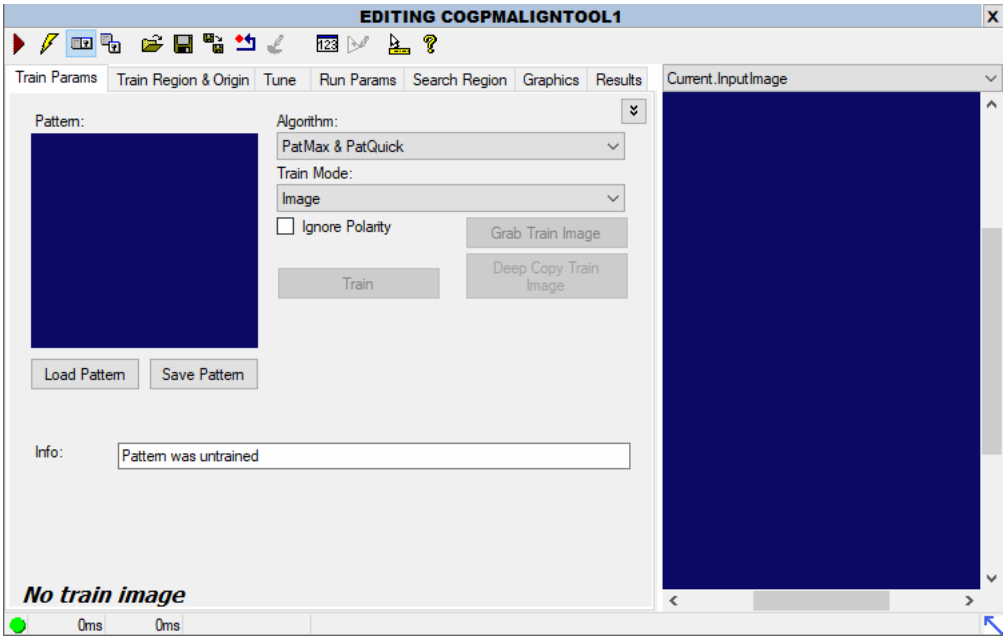


Changing the Number of Tools

1. Click **Add Tool** to create a new tool of the same type.
2. Click **X** to remove the current tool.
3. Double-click the tool name (such as CogPMAlignTool1) to open the VisionPro tool settings for the tool.
4. Right-click the tool name to rename the tool.
5. Change the counter number to change which tool is displayed.

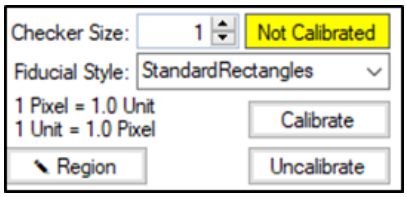
Editing the ToolBlock

Allows editing CogToolBlocks. The example shows the toolblock for a pattern tool. You can access the toolblock by clicking the tool name in the tool list under Inspection Settings.



Calibrate

Checkerboard Calibration



Name	Description
Checker Size	Sets size of checkerboard squares for calibration.
Calibration Status	Shows status of calibration.
Fiducial Style	Choose feature to use as marker for calibration.
Scale	Displays scaling info for calibration.
Calibrate	Run calibration.
Uncalibrate	Forget calibration.
Edit Region	Specify region of image to limit calibration to.

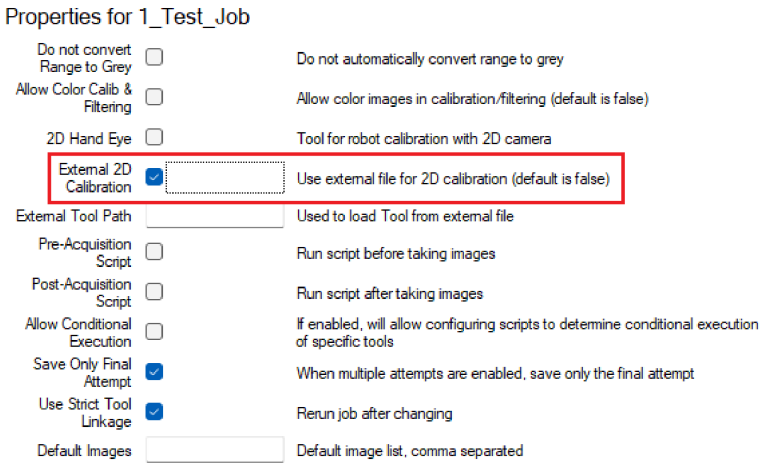
How to Set up Checkerboard Calibration

To run Checkerboard Calibration:

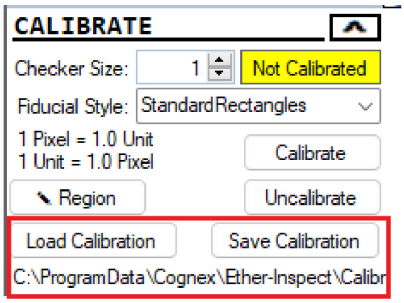
1. Set the **Checker Size** to the size of the checkerboard squares (either metric or imperial units).
2. Choose the **Fiducial Style** to set a visual reference in the image to calibrate off of. It is recommended but not necessary to use the **StandardRectangle** fiducial style.
3. Optional: Click **Edit Region** if you want to designate a portion of the image to be calibrated.
4. Click **Calibrate** to run the calibration process.
5. If the calibration is no longer needed, click **Uncalibrate**.

External Calibration (Sharing between Jobs)

External calibration is disabled by default. To enable the feature, go to *Setup/Configuration* > *Cameras* > *(cameraName)* > *(jobName)* and enable the setting **External 2D Calibration**.



Enabling the setting makes the following controls appear within the Calibrate tool. The Calibrate tool allows you to load calibrations from other jobs and save calibrations to share with other jobs.



Save Calibration saves the current calibration to a file.

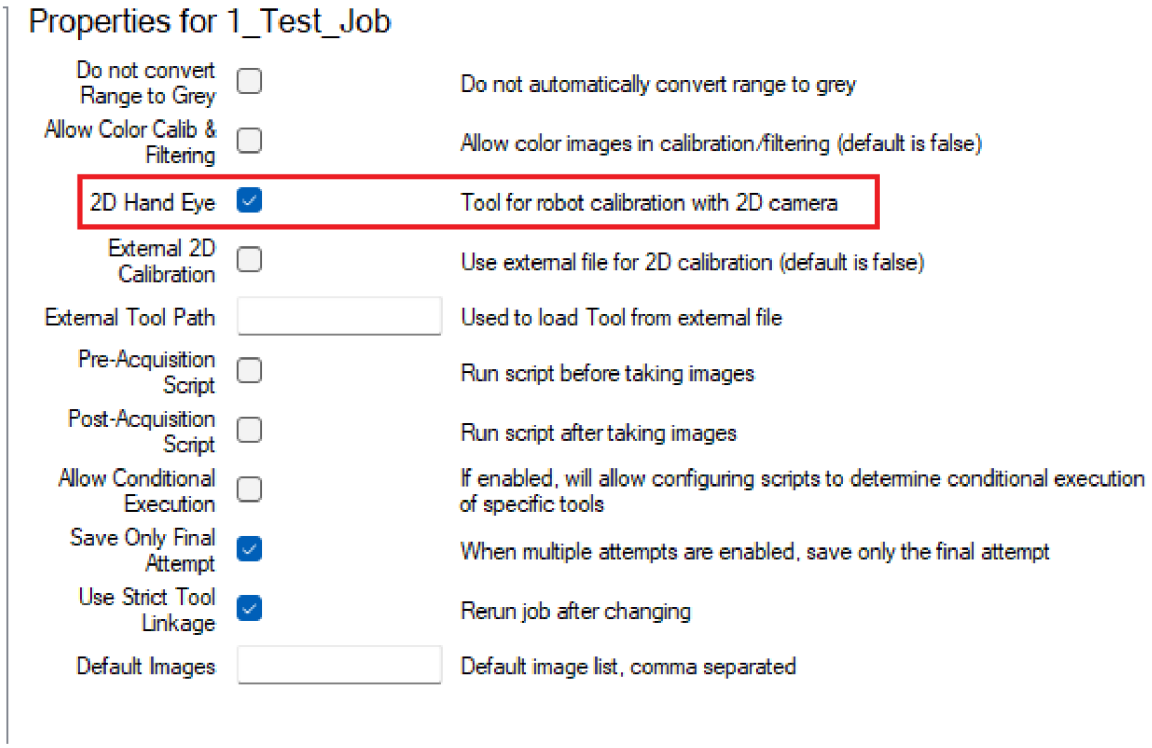
Load Calibration allows you to load a file with a calibration in it.

> **Note**: When you enable the External Calibration option, the calibration is loaded from the file each time the job is loaded. If the calibration changes, it only needs to be saved once from a job using it.

HandEye2D Calibration

Use the HandEye2D tool to calibrate robot coordinates based on image coordinates in EtherInspect. The tool combines checkerboard calibration and N-Point calibration into one tool. The N-Point calibration takes a minimum of 3 and a maximum of 90 points to correlate the robot coordinate system to the vision coordinates.

The HandEye2D calibration is an optional replacement for standard checkerboard calibration. To enable it, go to *Setup/Configuration* > *(cameraName)* > *(jobName)* and enable **2D Hand Eye** within the job properties.



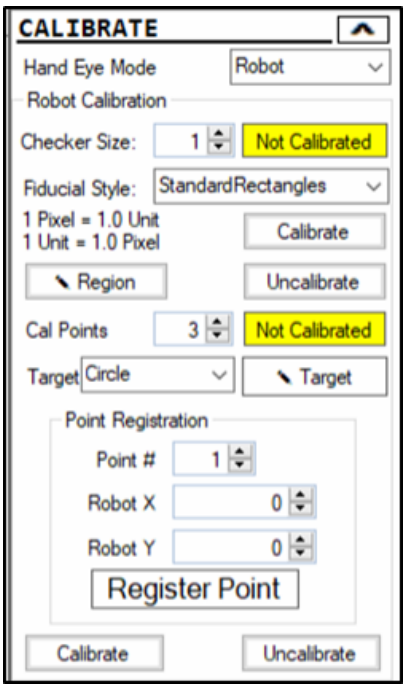
The two HandEye2D Calibration modes are:

- *How to Set up HandEye2D Calibration in Robot Mode* on page 61
- *Checkerboard Calibration for Motion Stage Mode* on page 63

The modes differ based on the setup or configuration and method of calibration used when calibrating the robot.

Robot Mode

Robot mode allows you to register multiple calibration points off of a target.



Name	Description
Hand Eye Mode	Sets the mode for the HandEye2D Calibration tool.
Checker Size	Size of checkerboard tiles.
Checkerboard Calibration Status	Provides the status of the checkerboard calibration (Not Calibrated/Calibrated).
Fiducial Style	Specify the point of reference to base the measurement off of.
Calibrate (Checkerboard)	Runs checkerboard calibration.
Uncalibrate (Checkerboard)	Forgets the current checkerboard calibration.
Calibration Points	Specifies the number of calibration points expected to register.
Calibration Status (Point Registration)	Provides the status of point registration calibration.
Target	Specifies calibration target. Circle indicates calibration uses findCircle tool. Pattern indicates calibration uses PatMax tool.
Target Edit	Opens an edit control for advanced settings on calibration target.
Point Registration	Allows you to specify points of calibration including robot X, robot Y, and point #. Number of points must not exceed Cal Points.
Calibrate (Point Registration)	Runs calibration on points of calibration.
Uncalibrate (Point Registration)	Forgets current calibration from point registration.

The top portion is a standard checkerboard calibration. For more information, see *Checkerboard Calibration* on page 57.

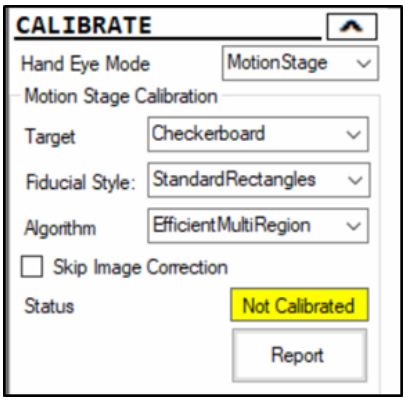
How to Set up HandEye2D Calibration in Robot Mode

1. Set the **Cal Points** for the calibration.
2. Choose the **Target Type** (Circle or Pattern find) and configure the tool using the **Edit Target** button.
3. Run the job to verify that the target is found.
4. Register calibration points (repeat the following steps for each point to add).
    a. Set the **Point #** .
    b. Enter the **Robot X** coordinate.
    c. Enter the **Robot Y** coordinate.
    d. Click **Register Point**.
5. Compute the calibration after all points are registered by clicking **Calibrate**.

Motion Stage Mode

Motion stage works similarly to robot mode except motion stage includes advanced features such as RealPart targets that allow you to use multiple features as points of reference when calibrating rather than using a checkerboard calibration plate. Motion stage mode also uses algorithms to either find the fastest way of calibrating or is exhaustive in the calibration process for more accuracy.

When selecting motion stage as the Hand Eye mode, the following control pops up under the **Calibrate** section in inspection settings.



Name	Description
Target	Calibration target to look out for. Checkerboard uses checkerboard pattern on calibration plates. RealPart bases calibration off features in an image as reference points (Alignment finder tool is necessary to generate these points prior to setting this configuration).
Fiducial Style	Point of reference within target to locate.
Algorithm	Algorithm when running calibration process. This is split into two types: • Exhaustive: Thorough process of calibration (higher accuracy, longer cycle time). • Efficient: Quick and less detailed calibration (shorter cycle time, lower accuracy).

Skip Image Correction	In normal conditions, the application corrects any image distortion. If application is time critical, **Skip Image Correction** skips the process. **Note**: **Skip Image Correction** results in less accuracy.
Calibration Status	Indicates if the system is calibrated.
Report	Generates performance report on alignment calibration.

Algorithm specifies the method by which calibration occurs. The different algorithms are:

- **Standard** - Runs calibration at moderate pace with default parameters.
- **Exhaustive** – When dealing with issues such as poor image quality, this algorithm spends more time searching for the calibration target and processing the fiducial points in the image.
- **ExhaustiveMultiRegion** – Follows same process as exhaustive but focuses on multiple regions in image for calibration.
- **Efficient** – Extracts calibration features in fastest cycle time. Less accurate than exhaustive.
- **EfficientMultiRegion** – Same process as efficient but focuses on multiple regions in image for calibration.

Report generates a performance report based on the accuracy of the calibration process. The report information includes:

- **Camera Pose in Home2D** – X, Y, and theta positions of the camera relative to robot home position.
- **Home2DFromCamera2D** – Offset between camera and robot coordinate systems.
- **Home2D** – Robot Home Position.
- **Raw2D** – Raw uncalibrated position (in pixels).
- **Quality** – Overall accuracy of camera coordinate system.

	Camera Pose in Home2D			Overall Residuals				
Camera Index	PelSize	X	Y	T degs	Home2D RMS	Home2D Max	Raw2D RMS	Raw2D Max	Quality
0	0.134	149.883	110.071	89.997	0.040	0.102	0.30	0.77	Excellent

The report also includes a table of the robot calibration points and their points of accuracy.

Camera 0 Calibration Residuals				
Pose Index	Plate2D RMS	Plate2D Max	Raw2D RMS	Raw2D Max
0	0.049	0.090	0.36	0.68
1	0.045	0.081	0.34	0.61
2	0.046	0.084	0.34	0.63
3	0.046	0.085	0.34	0.63
4	0.045	0.087	0.34	0.65
5	0.048	0.102	0.36	0.76
6	0.047	0.103	0.35	0.77
7	0.043	0.092	0.32	0.69
8	0.044	0.089	0.33	0.66
9	0.014	0.034	0.10	0.26
10	0.016	0.035	0.12	0.26
11	0.015	0.041	0.11	0.31
12	0.013	0.040	0.10	0.30

How to set up HandEye2D Calibration in Motion Stage Mode

Checkerboard Calibration for Motion Stage Mode

1. Set **Target** to **Checkerboard**.
2. Set **Fiducial Style** (Standard, DataMatrix, or DataMatrixWithGridPitch are recommended).
3. Set the algorithm. (Efficient for quick calibration and exhaustive for more accurate calibration).
4. If your application is time critical, enable **Skip Image Distortion** to skip the correction for image distortion.
5. Follow the steps in TCP Interface to run the calibration.

RealPart Calibration

1. Set the **Target** to **Realpart**.
2. Go to **Alignment Finder** tool.
3. Acquire an image of actual part to be inspected.
4. Set the points of reference either through pattern or corner type.
5. Train the points.
6. Go back to **Calibrate**.
7. Set **Part Features** to output of **Alignment Finder** tool.
8. Follow the steps in TCP Interface to run calibration.

TCP Interface

1. Start a TCP listener.
2. Use the following command to start registering points for calibration:

   `StartC,<camera_name>`

   Note: The `camera_name` is the name of the camera running calibration.

3. Use the following command to register points in motion stage mode prior to ending calibration. You can calibrate a maximum of 99 points with a minimum of 3 points.

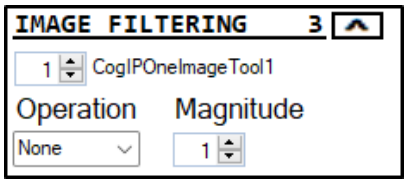
   `HECAL,<camera_name>,<point_number>,<X>,<Y>,<Theta>`

   - The `point_number` is the numbered point (starting from 1).
   - `X` is the motion stage X coordinate for the point.
   - `Y` is the motion stage Y coordinate for the point.
   - `Theta` is the angle/pose of the point.

4. Repeat Step 3 for all points you want to register.
5. Use the following command to end calibration and calculate the coordinate system:

   `EndC,<camera_name>`

Image Filtering



Name	Description
Operation	Options for operations to run on images for filtering
Magnitude	Number of times the filter is run (constant in multiply mode)

Image Filtering Operations

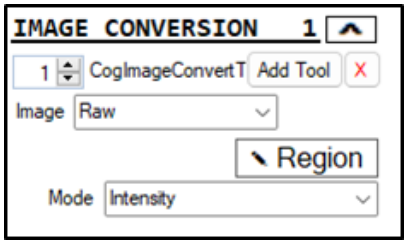
You can apply the following operations to images:

- **Dilate**: Dilates image. Increasing magnitude increases dilation effect.
- **Erode**: Reduces details in the image. Increasing magnitude increases erosion effect.
- **Open**: Reduce outside barrier of image.
- **Close**: Reduce outside barrier of image.
- **Multiply**: Multiplies contrast or exposure rate by a factor specified by user.
- **Custom**: Custom user-defined filter, allows editing of Image Filtering toolblock to create custom rules for tool.

Change the **Magnitude** to change the number of times the filter runs (or the constant in Multiply mode).

Image Conversion

Converts the input image into different format.



Name	Description
Input Image	Choose which image to run the conversion on.
Region Edit	Edit the Region to run the conversion tool on.
Mode	Choose which image format to convert to.

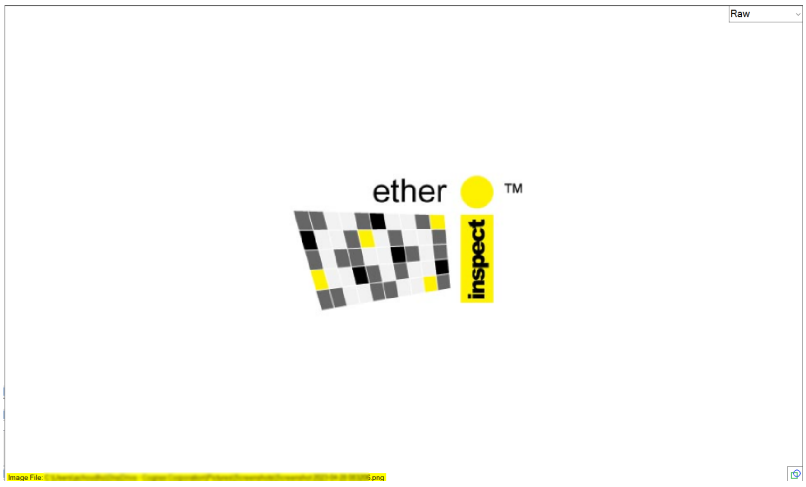
The three modes of conversion are:

- Intensity
- IntensityFromWeightedRGB
- PixelFromRange

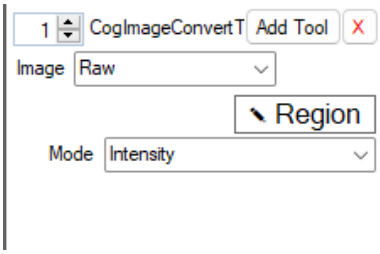
Converting Images to Greyscale

To convert images over to Greyscale, you need to use the **Intensity** setting:

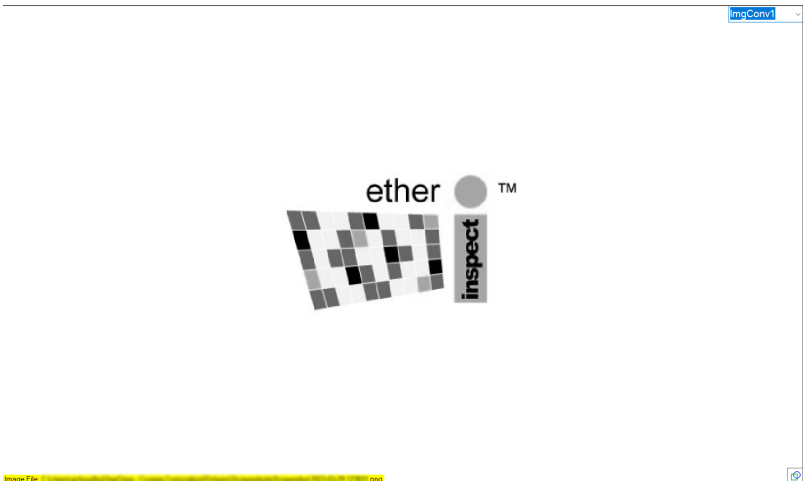
1. Load the image to convert.



2. Add an image conversion tool and set the settings:
    - Image: Raw
    - Region: Entire Image
    - Mode: Intensity

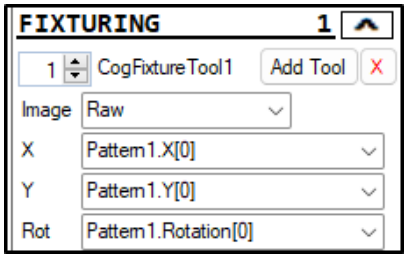


3. Reload the inspection settings, and view the **<ConversionToolName>** filter.



Fixturing

Creates a 2D fixtured image. You can use the fixturing tool with the Pattern tool to center the fixture in reference to the pattern result.



Name	Description
Input Image	Image to be used as reference
Fixture X	X coordinate of Fixture
Fixture Y	Y coordinate of Fixture
Fixture Rotation	Angle of Rotation of Fixture

Fixtures provide a reference point that other tools can use to position themselves. The reference point can also be used to position graphics in an image.

Coordinate Spaces

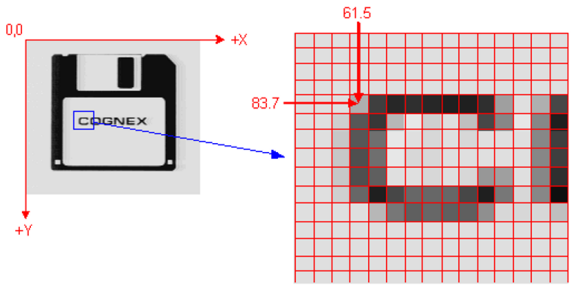
Coordinate Spaces provide references for the locations of points.

The three types of Coordinate Spaces are:

- Root Space
- User Space
- Pixel Space

Root Space

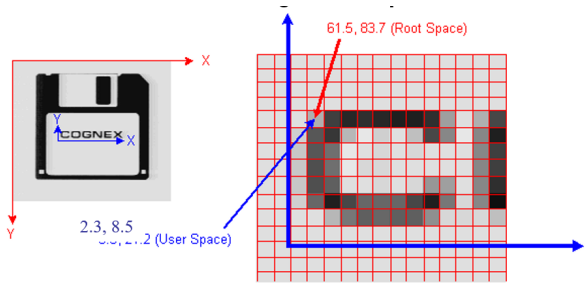
A left-handed coordinate system perfectly aligns with the pixels of an acquired image prior to any image processing.



User Space

Use User-defined Coordinate Spaces to create and manipulate calibrated spaces and fixtures. When defining a user space, you determine:

- Units
- Handedness
- Relation to the root space of the Image



Pixel Space

Pixel space is similar to root space in that:

- Origin is always in the upper-left corner
- Space corresponds to the image pixels

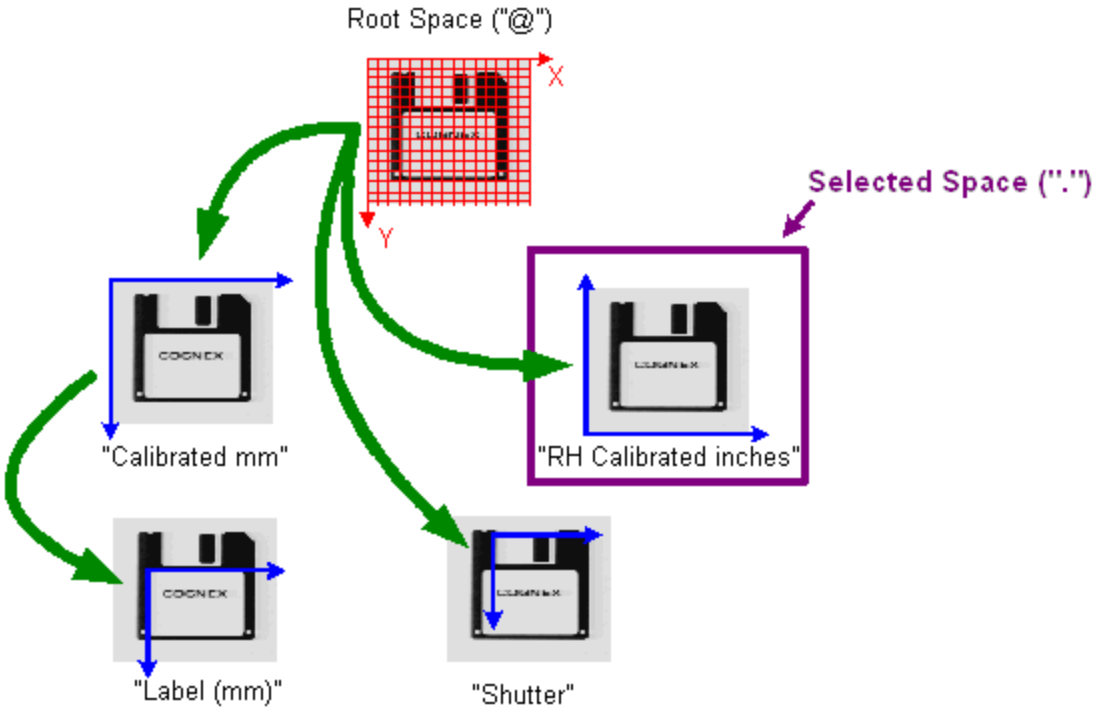
However, pixel space does not readjust itself to reflect the effects of image processing.

Note: Applications rarely use pixel space.

Coordinate Space Trees

Coordinate space trees contain:

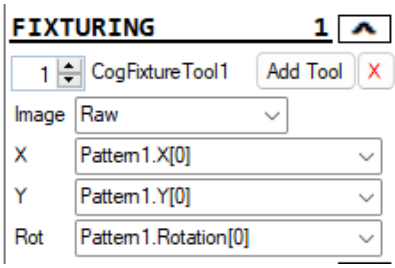
- Root space of the Image
- All user spaces created
- Relation between each space (such as Transformation)



Create a Fixture

To create and set a fixture point, you need an image and a reference point within the image to center the fixture on. The example shows the pattern tool.

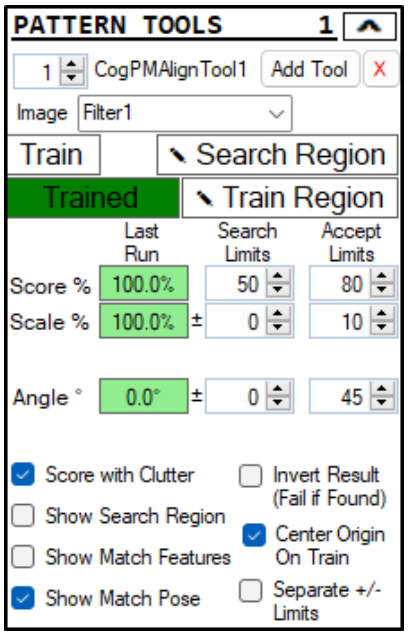
1. Load an image.
2. Create and train a pattern tool off a part within the image that you plan to use as a fixture.
3. Add a fixture tool and set the image to **Raw**.
4. Set the Fixture X, Y, and Rot to the X, Y, and Rot of the trained pattern.



5. The fixture is now set.

Pattern

The pattern tool uses the VisionPro PatMax tool. Use the pattern tool to train models on specific parts in an image. The tool is contour-based and uses contrast in the image to find edges to generate a pattern to train the model off of.



Name	Description
Input Image	Reference image to use for training.
Train	Runs training process.
Search Region Edit	Edit region to limit area of search when looking for trained pattern.
Train Status	Status on whether a model is trained.
Train Region Edit	Edit region to limit area to train on.
Search/Accept Limits	Set limits to restrict search and accept criteria.
Score with Clutter	Includes clutter in the score.

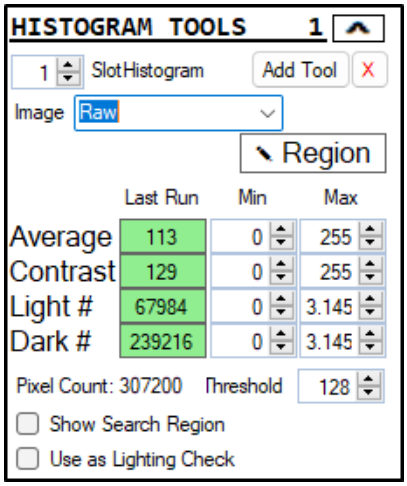
Invert Result	Fails an image if the pattern is found.
Show Search Region	Show or hide the search region.
Center Origin On Train	Centers the origin when training.
Show Match Features	Show or hide the match features.
Show Match Pose	Show or hide the match pose.
Separate +/- Limits	Separates the positive and negative limits.

Train a Pattern in an Image

1. Select the input image.
2. Edit **Train Region** to focus only on the part that you want to train.
3. Optional: Edit **Search Region** to limit area of search.
4. Click **Train** to run the pattern training.
5. Modify the search or accept limits.
6. Optional: Modify the additional parameters.

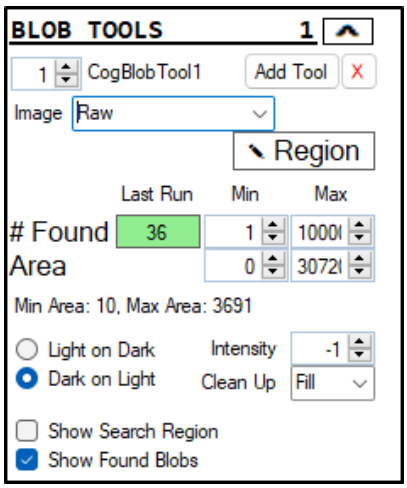
Histogram

Use the histogram tool to run statistical measurements on the pixels within a region of an image. The statistical measurements can find the mean, median, standard deviation, and variance of the pixel values. This generates a histogram of the pixel values and the number of each pixel value that appears within the image.



Name	Description
Input Image	Image to use with histogram tool.
Edit Region	Allows you to limit region to run histogram tool on.
Average	Mean value of pixels within the region.
Contrast	Difference between light and dark pixels.
Light #	The number of light pixels in the image.
Dark #	The number of dark pixels in the image.
Total Pixel Count	Total pixel count of region used in histogram tool.
Threshold	Limit to differentiate between object and background pixels.
Show Search Region	Displays the search region bounding box within an image.
Use as Lighting Check	Turns on LightingNOK bit in EtherInspect status word if tool fails.

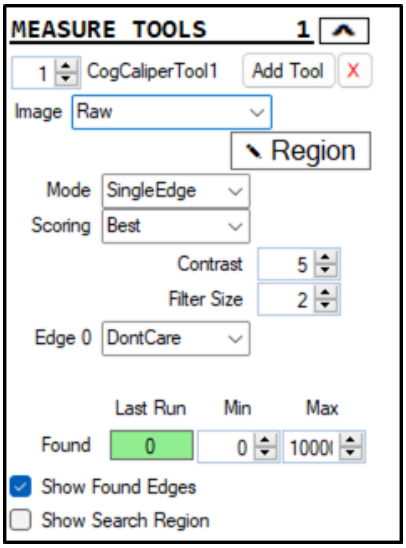
Blob Tools



Name	Description
Input Image	Image to use with blob tool.

Edit Region	Limit region to search for blobs.
# Found	Number of blobs found in region.
Area	Area of individual blobs in region.
Light on Dark	Search for light blobs on dark backgrounds.
Dark on Light	Search for dark blobs on light backgrounds.
Intensity	Threshold for light versus dark. -1 is automatic threshold.
Clean Up	Method to resolve multiple blob detections in the same area of interest. • None: No method used. • Prune: Removes noise from blob detections. • Fill: Combines multiple blob detections in the same region as a singular blob
Show Search Region	Show or hide search region.
Show Found Blobs	Show or hide found blobs.

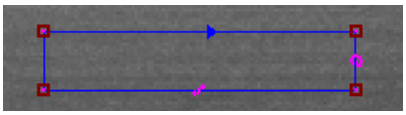
Measure Tools



Name	Description
Input Image	The image the measure (caliper) tool uses.
Edit Region	Limit region to focus on when finding edges, setting a region is required to run the tool.
Edge Mode	Determines method of searching for edges (Single, Pair).
Scoring Mode	Limit which edges are shown in the image based on their score (Best, First, Last, All).
Contrast	Defines the minimum contrast need to find an edge.
Filter Size	Specifies the number of half pixels on both sides of an edge for it to qualify.
Edge Polarity	Determine the polarity when searching for edges.
Found	Lists the number of found edges and allows user to provide minimum or maximum constraints.

Show Found Edges	Show or hide found edges.
Show Search Region	Show or hide search region.

When editing the region, an arrow is on the bounding box:



The arrow determines the scan direction of the caliper tool. In the image above, the tool looks for edges from left to right. This is important when sorting out scored edges.

The two edge modes when dealing with the caliper tool are:

- **Single Edge**: Looks for individual edges (counts each individual edge).
- **Pair Edge**: Looks for a pair of edges (counts two edges as one pair).

Edge Polarity determines whether to find an edge going from dark-to-light pixels, light-to-dark pixels, or both (**DontCare**).

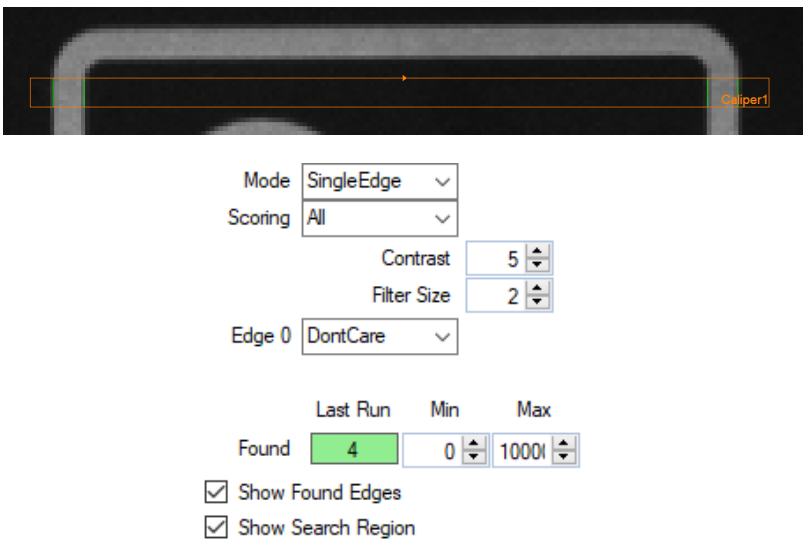
Scoring mode are different when switching between single edge and pair mode.

Single Edge

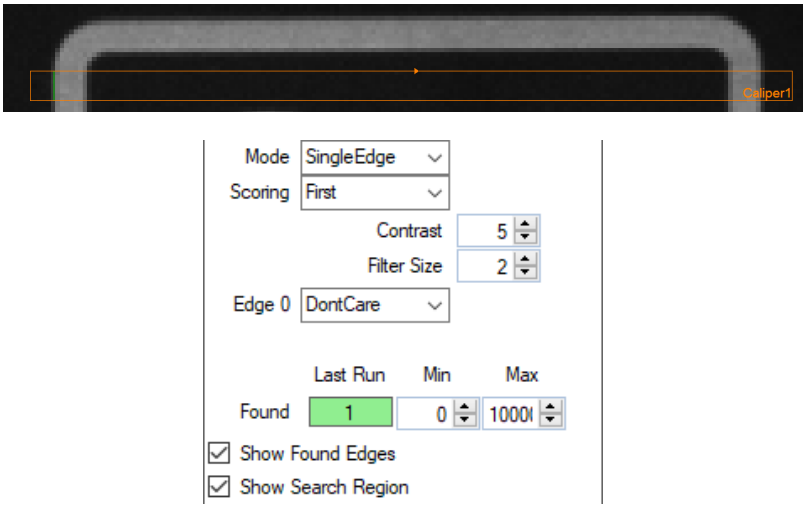
Single Edge mode looks for individual edges within a region of interest. The edges that the tool finds can be filtered based on Scoringmode and Edge Polarity.

The four options for Scoring settings for the edge mode are:

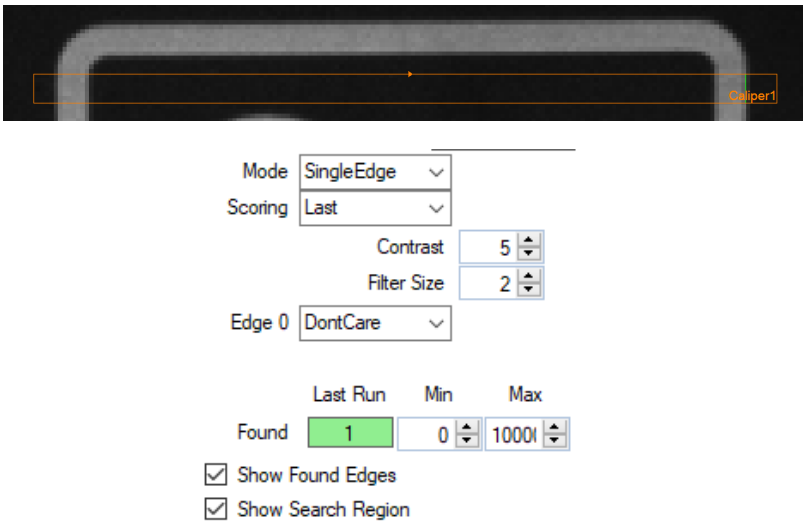
- **All**:Find all edges in the group.



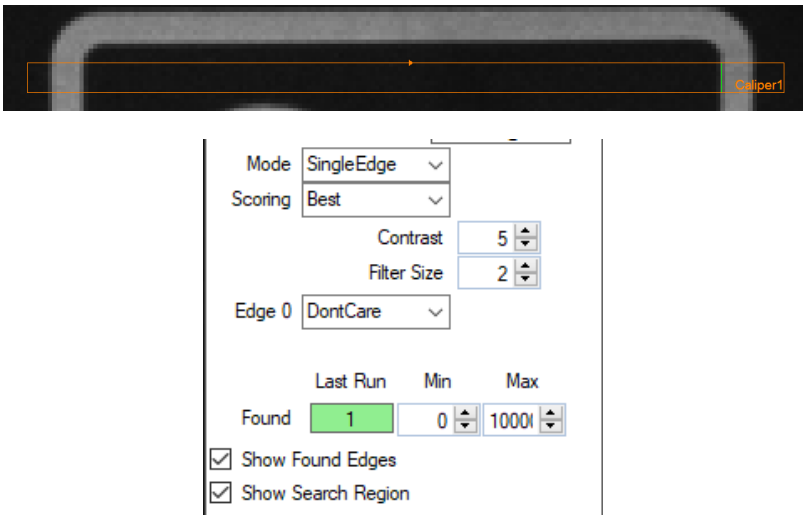
- **First**: Find the first edge in the group.



- **Last**: Find the last edge in the group.



- **Best**: Find the edge with the highest score in the group.

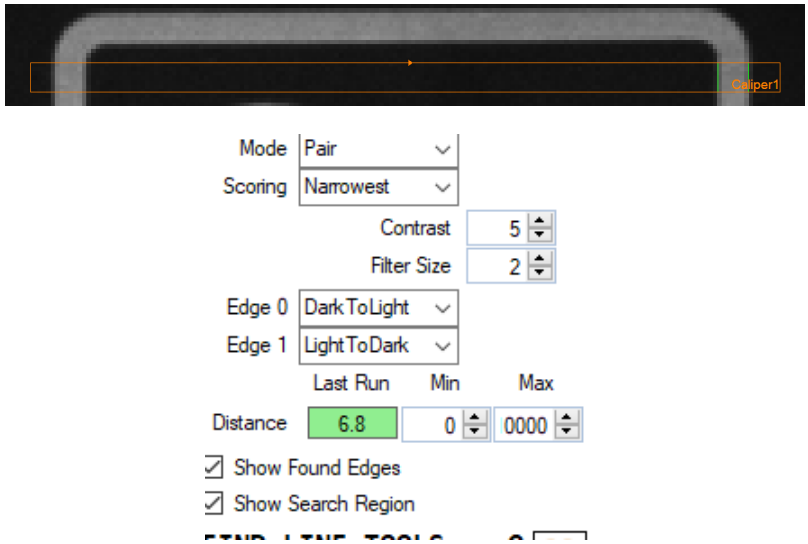


Pair Edge

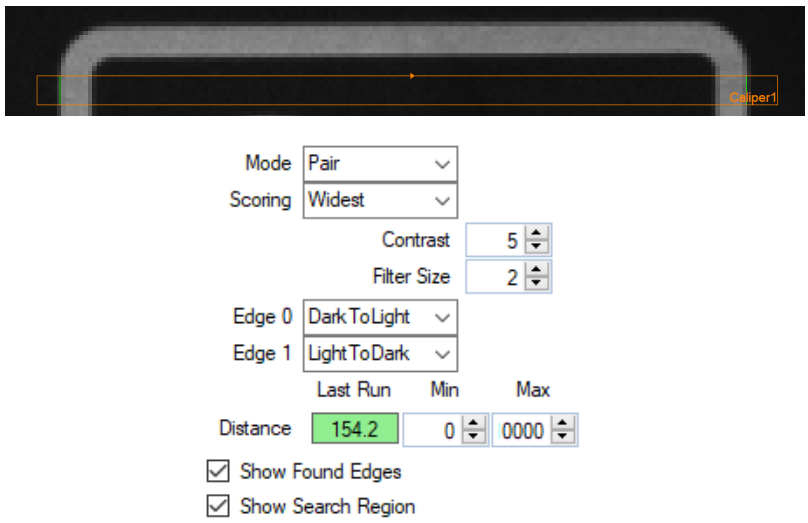
In pair mode, the tool counts two consecutive edges as one pair. This can also be filtered using Scoring mode and Edge Polarity. When setting up an edge pair, the polarity of each edge can be specified. In the examples, edge 0 has a polarity of DarkToLight while edge 1 has a polarity of LightToDark.

The Scoring setting for this edge mode has four options:

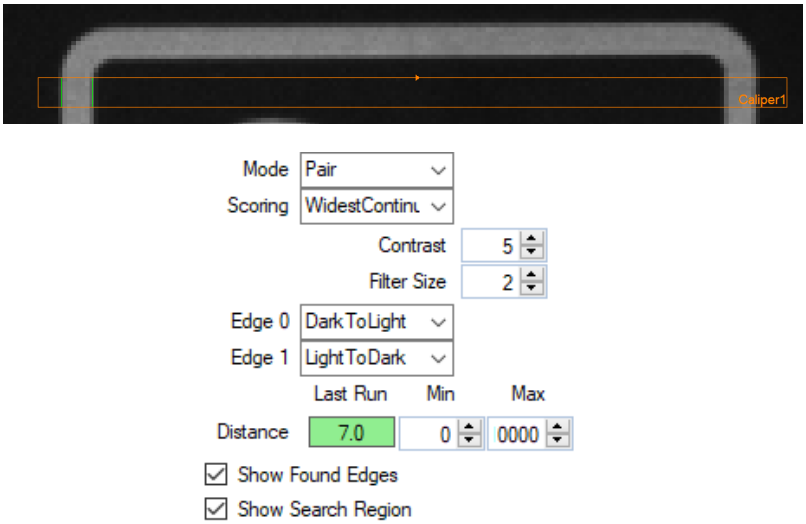
- **Narrowest**: Looks for the edge pair with the shortest distance between them.



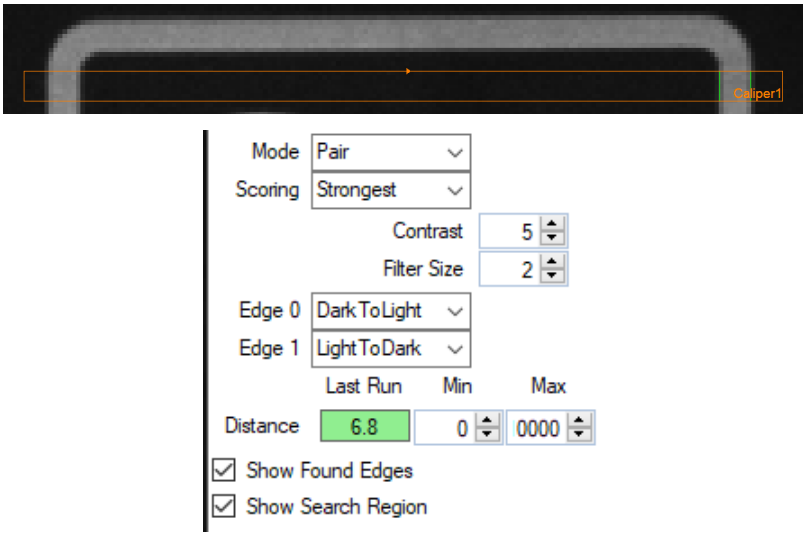
- **Widest**: Looks for the edge pair with the longest distance between them.



- **WidestContinuous**: Looks for the edge pair with the longest distance and contains the same contrast of pixels between the two.

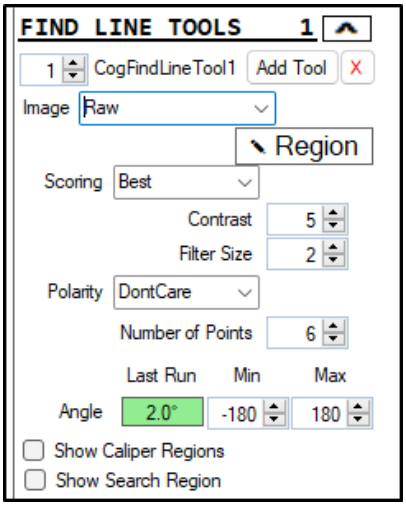


- **Strongest**: Looks for the edge pair with the strongest contrast between the inside and outside of the edge pair.



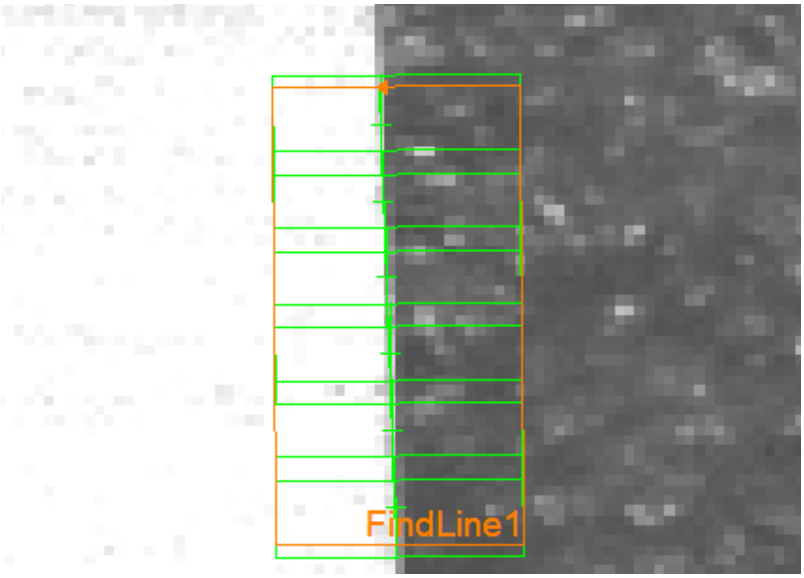
Find Line Tools

Uses a series of caliper tools over a specified region to locate multiple edge points to draw out a line of best fit while minimizing the root mean square (RMS) error.



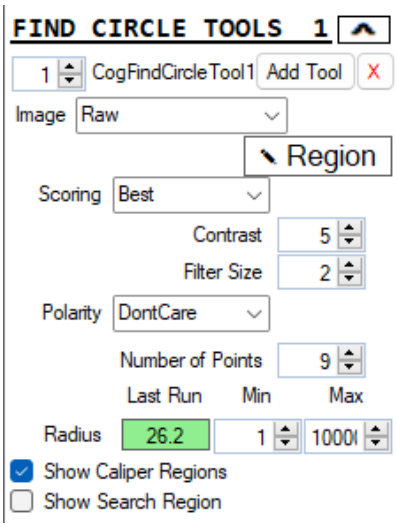
Name	Description
Input Image	Image to use with find line tool.
Edit Region	Define region of interest when finding line, region uses affline rectangle to define where the line is expected.
Scoring	Filter edges based on scoring.
Contrast	Defines the minimum contrast needed to find a line.
Filter Size	Defines half pixel width to filter lines.
Polarity	Method to find lines based on dark and light pixels.
Number of Points	Number of calipers that are created.
Angle	The angle that the found line is at.
Show Caliper Regions	Show or hide caliper regions.
Show Search Region	Show or hide search region.

Example of the Find Line result:



Find Circle Tools

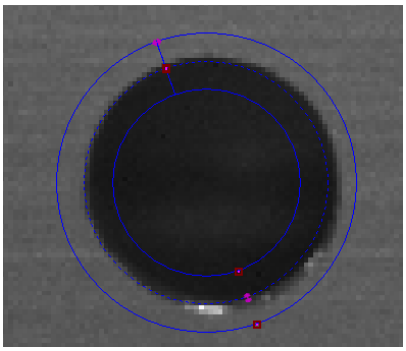
Uses a series of calipers to find a circular region within an image.



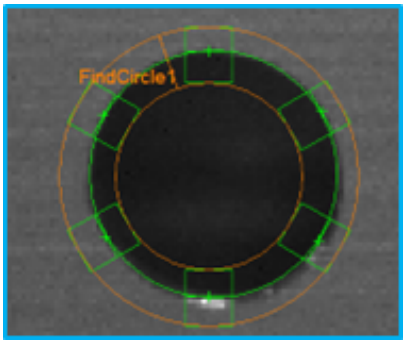
Name	Description
Input Image	Image to be used in find circle tool.
Edit Region	Define region of interest when finding circle, region uses circular annulus to define where the circle is expected.
Scoring Mode	Filter circles based on score.
Contrast	Minimum contrast to find a line.
Filter Size	Define half pixel width to filter circles.
Polarity	Method to find circles based on dark and light pixels.
Number of Points	Number of calipers that are created.

Radius	Radius of found circle.
Show Caliper Regions	Show or hide caliper regions.
Show Search Region	Show or hide search region.

When editing a region, you can rotate, expand, and compress the circle annulus region of interest.



When the tool runs, calipers are placed within the region of interest to find points on the circle.

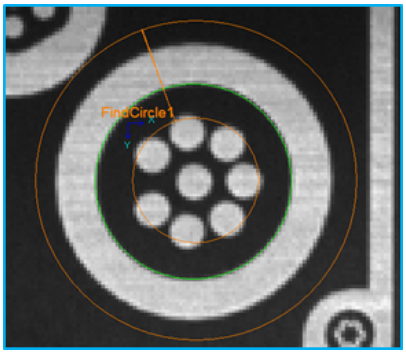


The Scoring mode works similarly to the single edge scoring mode in the **Find Line** tool. The tool uses the Scoring mode when there is more than one circle in its region of interest. If the region of interest only containd a single circle, use the **Best** for the scoring mode.

There are four scoring mode options:

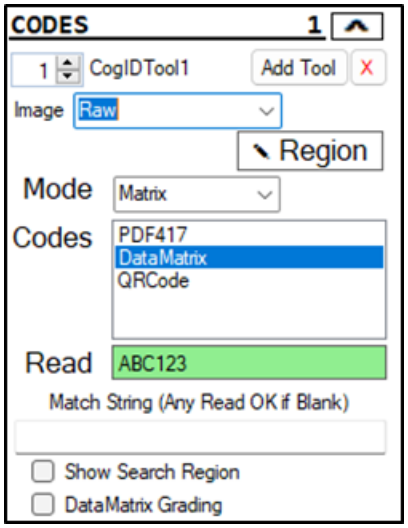
- **First**: Finds the innermost circle in the region.
- **Last**: Finds the outermost circle in the region.
- **Best**: Finds the circle with the highest score.
- **All**: Finds all circles in the region.

Polarity determines whether the tool searches for points on the circle based on dark-to-light pixels, light-to-dark, or both.



Codes (1D and 2D codes)

Finds barcodes, QR, and Data Matrix codes. Decodes string from code.

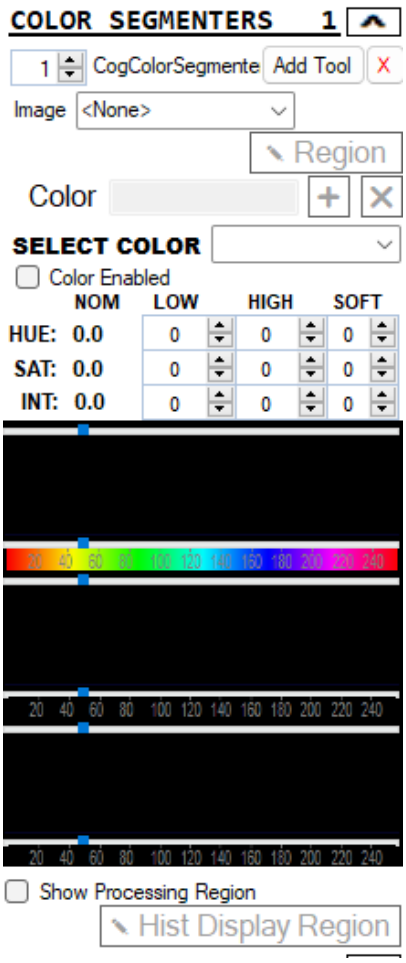


Name	Description
Input Image	Image to run code scan on.
Edit Region	Limit region of interest for code scanning.
Scan Mode	Choose category of code you are reading.
Code Type	Defines type of code to expect.
Read String	String read from code.
Match String	Reference string to expect .
Show Search Region	Show or hide the search region.
Grading	Enables Data Matrix Grading.

Example of a Codes tool result:

Color Segmenter

Creates a binary image from a color image, based on selected HSI values.



Name	Description
Input Image	Image to run color segmentation.
Edit Region	Limit region of interest for color segmenter tool.
Add/Remove Color	Add or remove color of your choice, prompted for sample.
Select Color	Select color from list of created colors.

HSI values	Retrieves HSI values and specifies range of color to look for.
Color Histogram	Allows you to modify HSI values and view histogram results.
Show Processing Region	Show or hide processing region.
Hist Display Region	Specify which region in image to search for color for histogram results.

When adding a color to the list, you are prompted to provide a sample from the image. This goes into **Edit Region** mode where you can place the rectangle in a region that focuses on the target color. From there, you can pick the color form the list within **Select Color** and see the HSI values of it. You can then click **Hist Display Region** to select a region of the image where you want to actively search for that color. This generates a binary image where instances of that color appear in white.

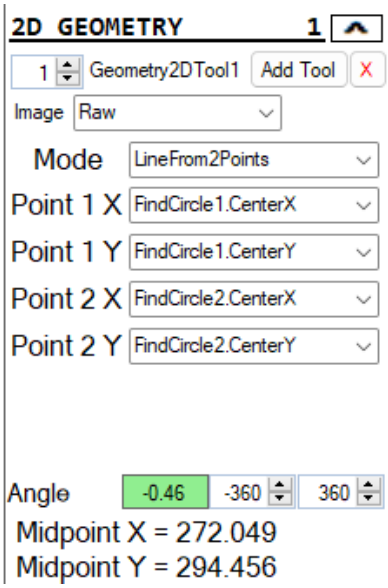
Note: The **Hist Display Region** serves as essentially a color tester.

2D Geometry

Use the 2D Geometry tool to run geometrical computations based on points provided to it in 2D space. 2D Geometry tool can be useful in finding the distance between two points or drawing a line from two points.

LineFrom2Points Tool

Generates and draws line from one point to another. Use fixture points to set up the tool.

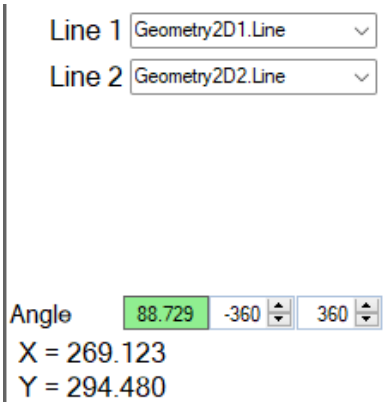


Name	Description
Image	Input image used in 2D Geometry tool.
Mode	Computation Mode.
Point Configuration	Sets points to run computations.
Found Angle	Provides angle of line found from computation.
Midpoint Information	Provides midpoint X and Y of line.

Example of the LineFrom2Points tool results:

IntersectionOf2Lines Tool

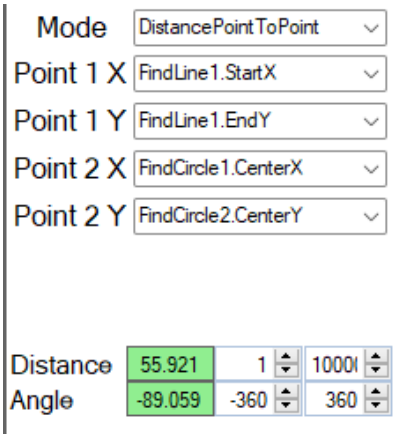
Finds the intersection point of two lines.



Example of the IntersectionOf2Lines tool results:

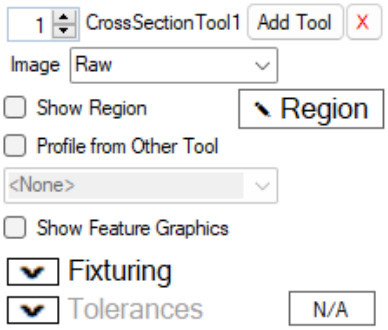
DistancePointToPoint Tool

Finds the distance between two points. Enter the X and Y values for the two points to find the distance. The distance is usually in pixel measurements. If you calibrate EtherInspect for metric or imperial units, the distance is converted to metric or imperial units.



Cross Section

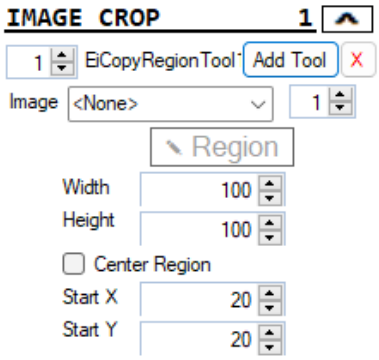
Finds the cross section of a 3D image based on parameters.



Name	Description
Show Region	Show or hide region.
Region	Edit the region.
Profile from Other Tool	If checked, choose another tool from the dropdown menu.
Show Feature Graphics	Show or hide feature graphics.
Fixturing	Expand **Fixturing** to enter an existing or new fixture.
Tolerances	Expand **Tolerances** to enter a tolerance, if available.

Image Crop

Crops an image. Useful for creating an image separate from the original which allows you to run separate tools in a focused region.

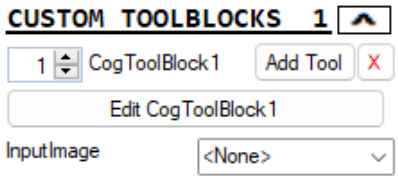


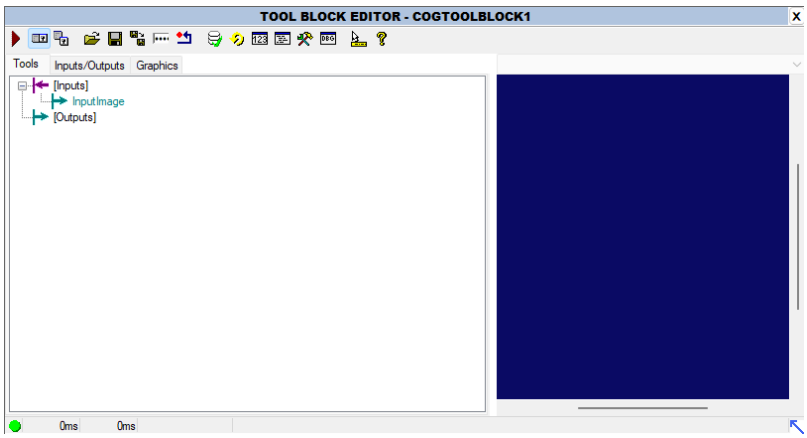
Name	Description
Input Image	Image to be used for cropping.
Edit Region	Selects a region to crop down.
Region Size	Size of region, auto-fills once region is adjusted.
Center Region	Centers region with image space.
Start Position	Start X and Start Y of region, auto-fills once region is adjusted.

Note: After you specify the region and the tool runs once, a filter appears in the image view that you can select to view the cropped image.

Custom ToolBlocks

Allows you to add and edit your own custom VisionPro toolblocks. Any inputs you add to the custom toolblock appear as a control in the **Custom ToolBlocks** section.



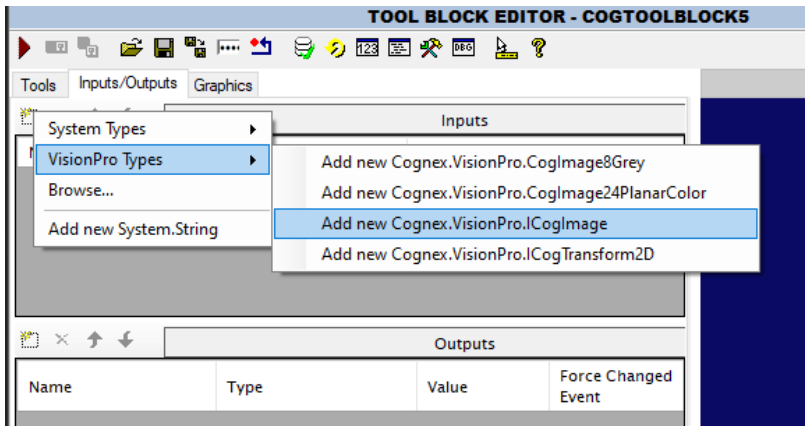


Custom Toolblock UI Controls

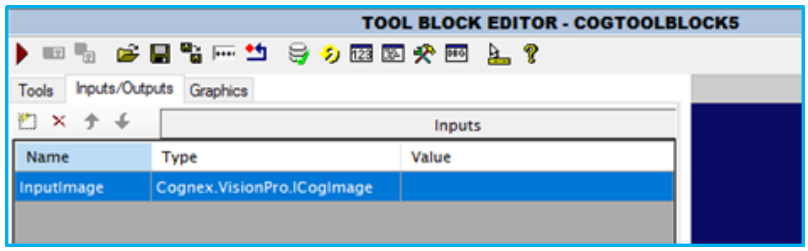
Input Image

Adding an input terminal of type ICogImage allows you to link that image in the UI.

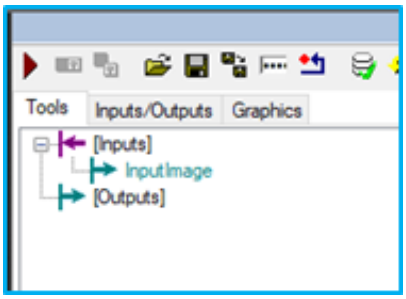
1. Select **Add new Cognex.VisionPro.ICogImage**.



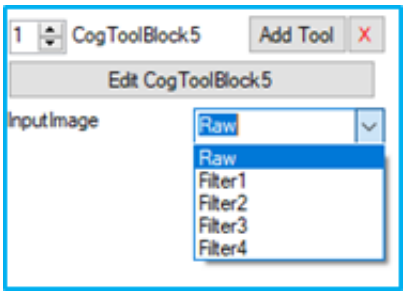
2. Select the InputImage to link.



3. Select InputImage in the **Tools** tab.

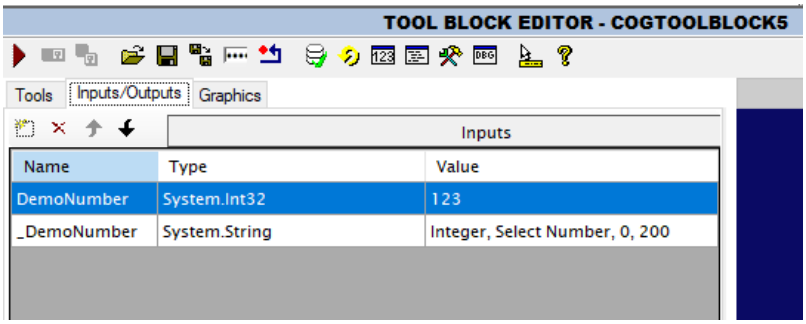


4. Choose the filter for the InputImage.



Number Picker

The Number Picker requires two input terminals. Use one to define type or value, and the other to set the UI properties.



DemoNumber is the default value while **_DemoNumber** is to define additional properties of the UI control.

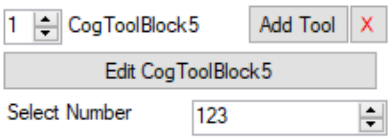
Note: Using only the **DemoNumber** input does not create any UI control. The value of the UI control is based on main terminal.

The string type and value of the **_DemoNumber** is in a specific format:

`{type}, {display_name}, {min_value}, {max_value}`

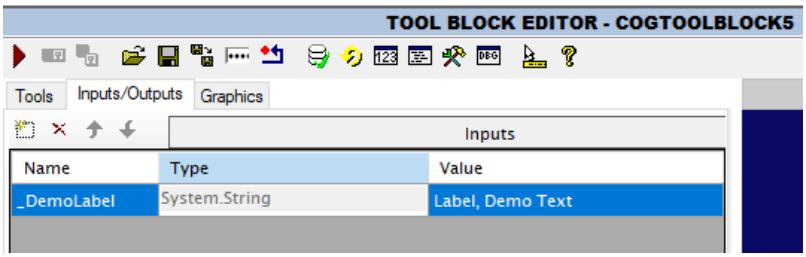
- `{type}`: Must be `Integer`.
- `{display_name}`: Label text for Number Picker.
- `{min_value}`: Minimum allowed value of number picker.
- `{max_value}`: Maximum allowed value of number picker.

Example: `Integer, Select Number, 0 , 200`



Label

The Label control only requires one input terminal. You can label sections of tools.

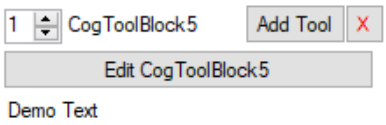


The string type and value of the **_DemoLabel** is in a specific format:

`{type}, {display_name}`

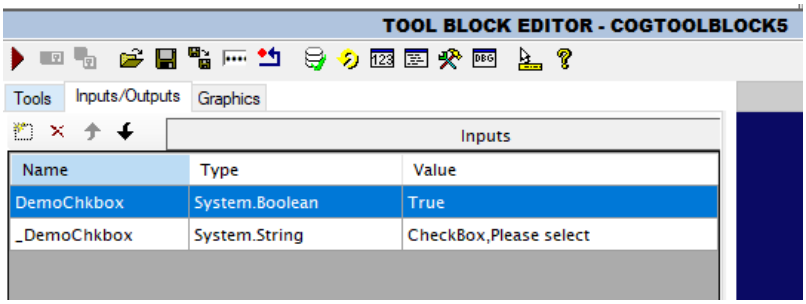
- `{type}`: Must be `Label`
- `{display_name}`: Text for label.

Example: `Label, Demo Text`



Check Box

The Check Box control requires two input terminals. Use one to define type or value and the other to set the UI properties.



DemoChkbox is the default value while **_DemoChkbox** is to define additional properties of the UI control.

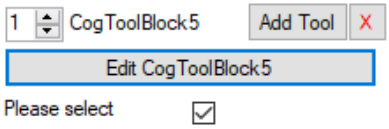
Note: Using only the **DemoChkbox** input does not create any UI control. The value of the UI control is based on main terminal.

The string type and value of the **_DemoChkbox** is in a specific format:

`{type}, {display_name}`

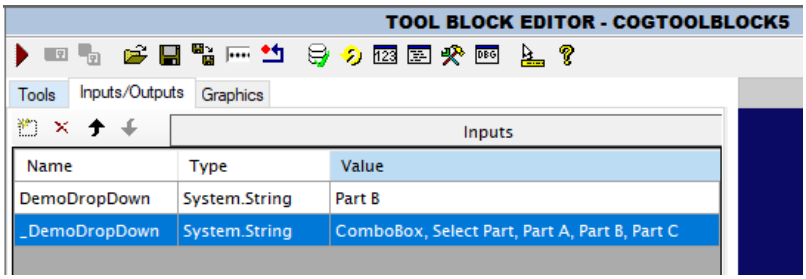
- `{type}`: Must be `CheckBox`.
- `{display_name}`: Label text for Check Box.

Example: `CheckBox, Please select`



Dropdown

The Dropdown control requires two input terminals. Use one to define type or value and the other to set the UI properties.



DemoDropDown is the default value while **_DemoDropDown** is to define additional properties of the UI control.

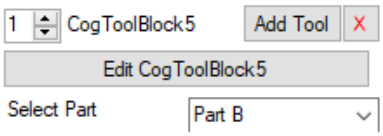
Note: Using only the **DemoDropDown** input does not create any UI control. The value of the UI control is based on main terminal.

The string type and value of the **_DemoDropDown** is in a specific format:

`{type}, {display_name}, {comma_separated_values}`

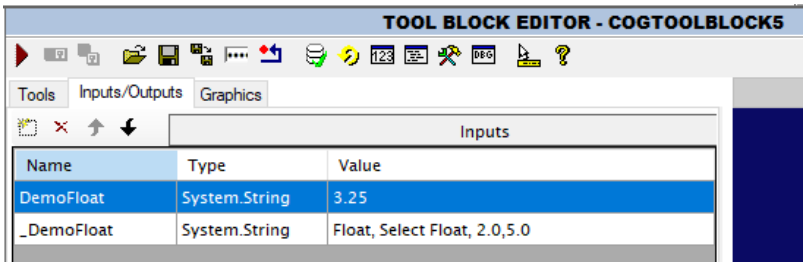
- `{type}`: Must be `ComboBox`.
- `{display_name}`: Label text for Combo Box.
- `{comma_separated_values}`: List down all the values you want in the dropdown menu as comma-separated.

Example: `ComboBox, Select Part, Part A, Part B, Part C`



Float

The Float control requires two input terminals. Use one to define type or value and the other to set the UI properties.



DemoFloat is the default value while **_DemoFloat** is to define additional properties of the UI control.

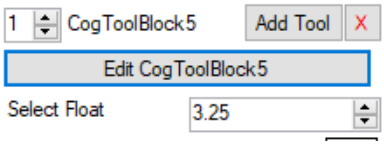
Note: Using only the **DemoFloat** input does not create any UI control. The value of the UI control is based on main terminal.

The string type and value of the **_DemoFloat** is in a specific format:

`{type}, {display_name}, {min_value}, {max_value}`

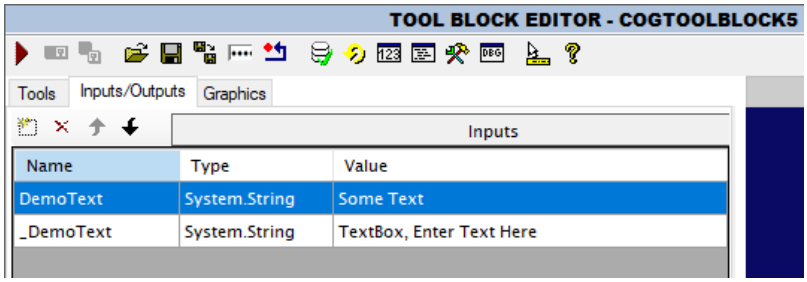
- `{type}`: Must be `Float`.
- `{display_name}`: Label text for Float.
- `{min_value} : Minimum allowed value of number picker.`
- `{max_value}`: Maximum allowed value of number picker.

Example: `Float, Select Float, 2.0, 5.0`



Text Input

The Text Input control requires two input terminals. Use one to define type or value and the other to set the UI properties.



DemoText is the default value while **_DemoText** is to define additional properties of the UI control.

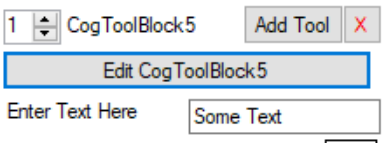
> **Note**: Using only the **DemoText** input does not create any UI control. The value of the UI control is based on main terminal.

The string type and value of the **_DemoText** is in a specific format:

`{type}, {display_name}`

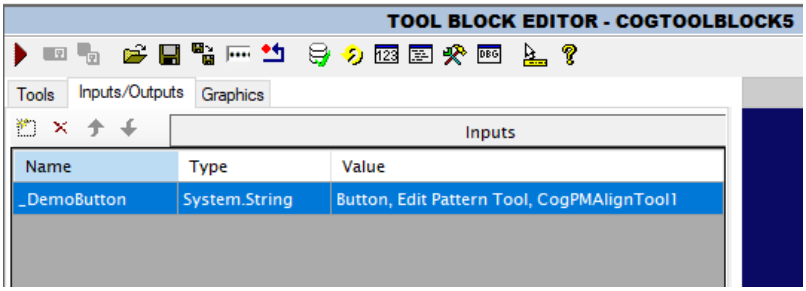
- `{type}`: Must be `TextBox`.
- `{display_name}`: Label text for the Text Box.

Example: `TextBox, Enter Text Here`



Tool Edit Button

The Text Edit Button control requires only one input terminal.

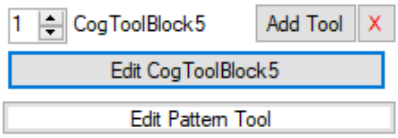


The string type and value of the **_DemoButton** is in a specific format:

`{type}, {display_name}, {tool_name}`

- `{type}`: Must be `Button`.
- `{display_name}`: Label text for the button.
- `{tool_name}`: Name of the tool.

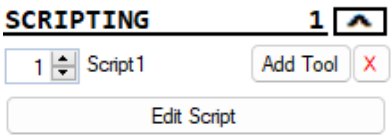
Example: `Button, Edit Pattern Tool, CogPMAlignTool1`



C# Scripting

Scripting Tool

Use the tool to create and edit scripts.

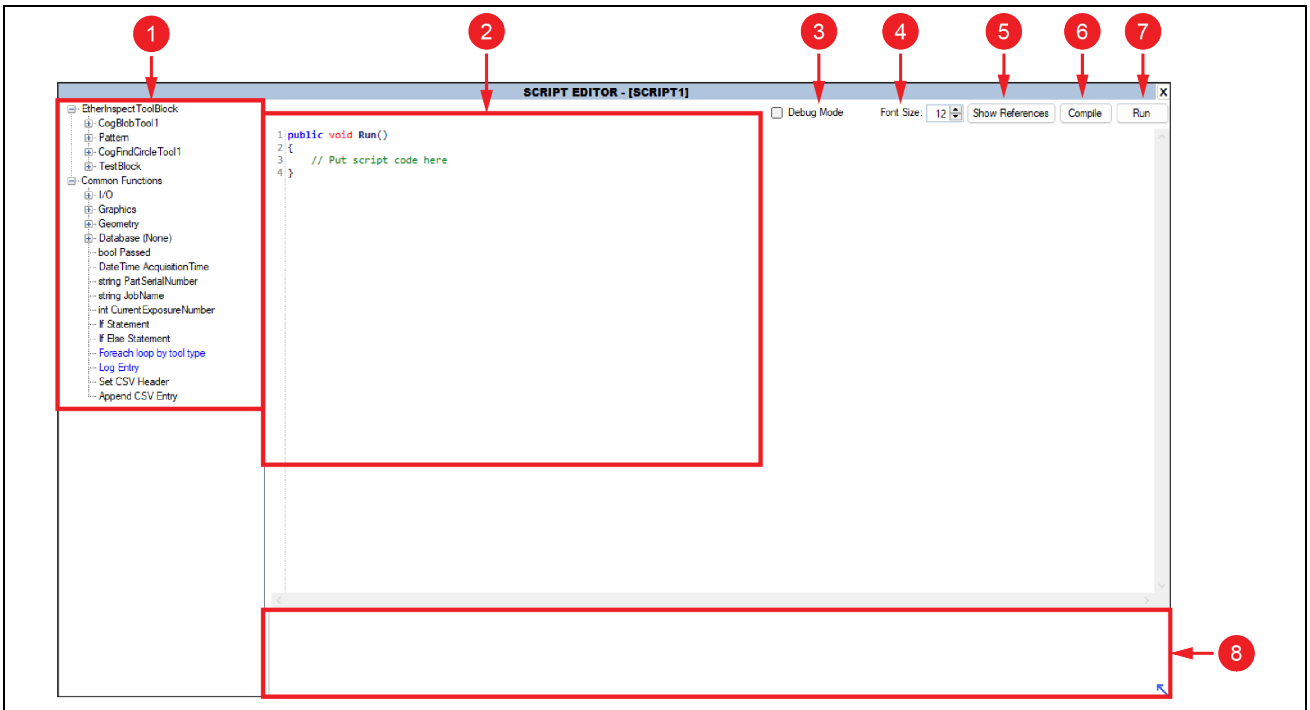


Click **Edit Script** to open the Script Editor

When multiple scripts are created within the scripting tool, each script runs sequentially starting with script 1 onward. The scripts run after image acquisition. If there are no inspections to run and the image view is blank, the script does not run.

Script Editor

Allows editing the C# code used in the EtherInspect scripting environment.



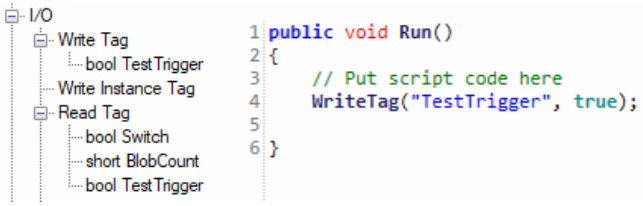
Number	Name	Description
1	Function Library	Contains a list of commonly used functions and EtherInspect toolblock functions including VisionPro references. Allows user to easily add code functions to program.
2	Code Editor	Allows you to edit code. All code must be placed inside Run() function in order for it to execute.
3	Debug Mode	Enabling this feature and recompiling enables a stack trace with line numbers when an exception occurs within the script during execution.
4	Font Size	Changes size of font in code editor.
5	Show References	Show list of all DLLs that are included when script is compiled.

6	Compile	Compiles script without running it. Errors show in log area.
7	Run	Compiles if needed and executes script.
8	Log	Shows any exceptions or errors that occur within the script upon compiling the code. Shows line number and potentially shows stack trace if Debug Mode is enabled.

Note: Use EtherInspect scripts instead of VisionPro scripts. VisionPro scripts are only supported in Custom ToolBlocks.

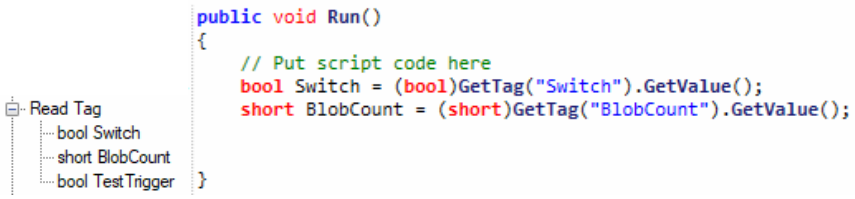
Write Tag Function

The Write Tag function updates the value of an OUTPUT PLC tag linked to EtherInspect. If there is a valid tag added to the tags list under **Setup Tags**, then you can access it in the Function Library to the left by going to **I/O > Write Tag** and expanding the function.



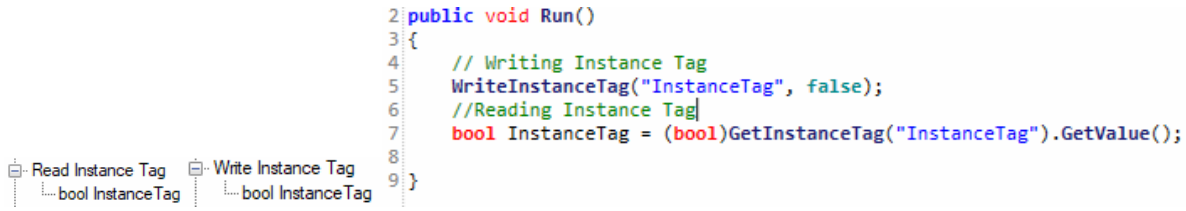
Read Tag Function

The Read Tag function reads the value of an INPUT PLC tag linked to EtherInspect. If there is a valid tag added to the tags list under **Setup Tags**, then you can access it in the Function Library to the left by going to **I/O > Read Tag** and expanding the function.



Write and Read Instance Tags

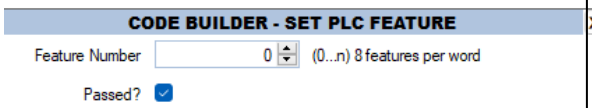
Instance tags are specific to an EtherInspect instance. Instance tags are a type of application tag used to control the flow of an instance. Reading and writing to an instance tag is similar to Write Tag and Read Tag functions.



Script Function Dialog

Use the Script Function dialog to add functions to an image. Generated code is pasted into the script.

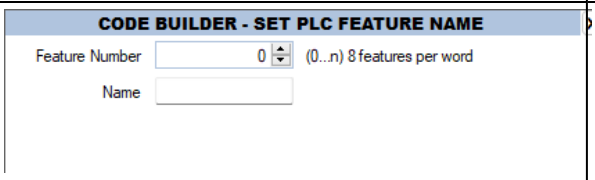
Function	Setting	Generated Code

<table>
<tr>
<td>Set PLC Feature</td>
<td>Sets the result of a feature within PLC.
 

Feature Number: Select feature number to set the result.
Passed: Check if you want to set the features passed.

</td>
<td>

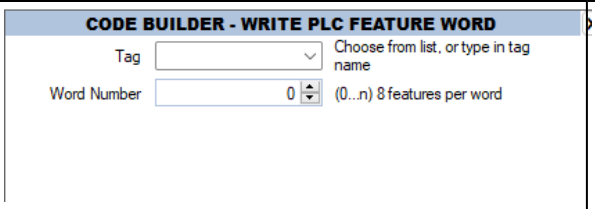
```
public void Run()
{
    // Put script code here
    SetFeature(0,true);
}
```

</td>
</tr>
<tr>
<td>Set Feature Name</td>
<td> 

Feature Number: Select feature number to set the name.
Name: Enter the name you want to set to the feature.

</td>
<td>

```
public void Run()
{
    // Put script code here
    SetFeatureName(5,"FeatureName");

}
```

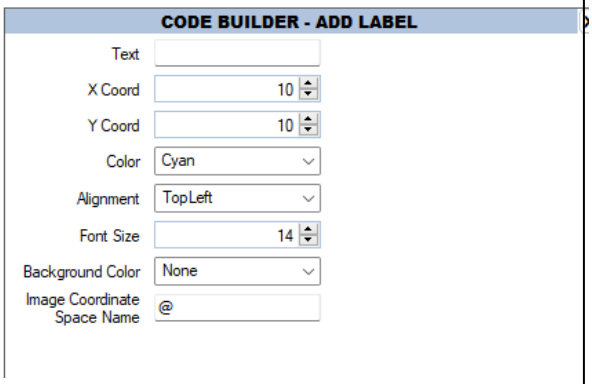
</td>
</tr>
<tr>
<td>Write Feature Word</td>
<td> 

Tag: Select the tag name from list.
Word Number: Set the word number.

</td>
<td>

```
public void Run()
{
    // Put script code here
    WriteFeatureWord("",0);

}
```

</td>
</tr>
</table>

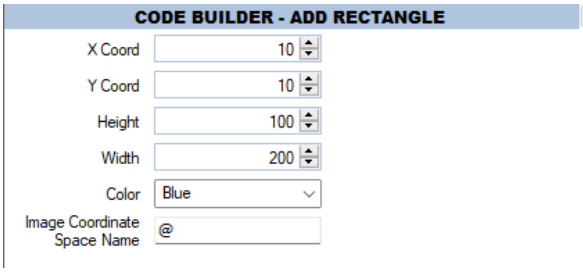
<table>
<tr>
<td>Add Label</td>
<td>Adds a label to the image view based on the parameters you set.
 

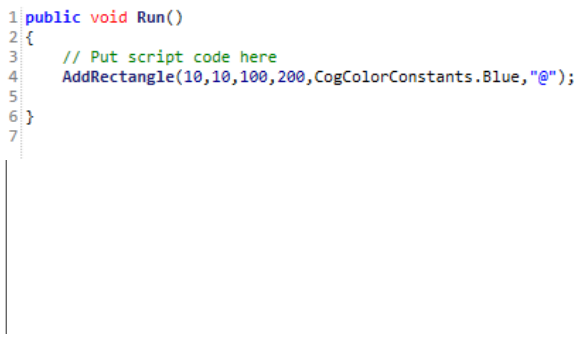
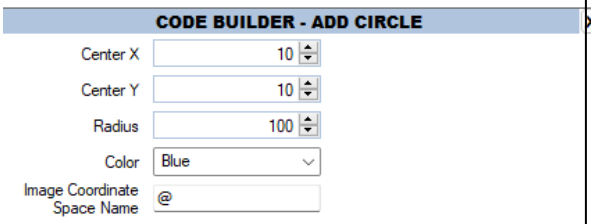
Text: Enter the text you want to set within the label.
X Coord: Set the label X coordinate of the label.
Y Coord: Set the label Y coordinate of the label.
Color: Select the color of the text from the list.
Alignment: Select the label text alignment.
Font size: Adjust the font size.
Background Color: Select the background color.
Image Coordinate Space Name: Set the coordinate space (“@”, “.”, “#”) (Can further expand on this using “/” followed by the name of the fixture point you want to base the coordinate space off).

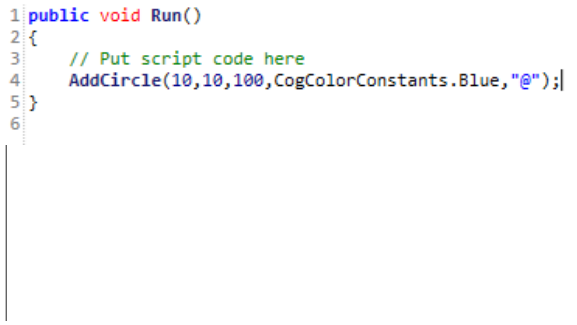
</td>
<td>

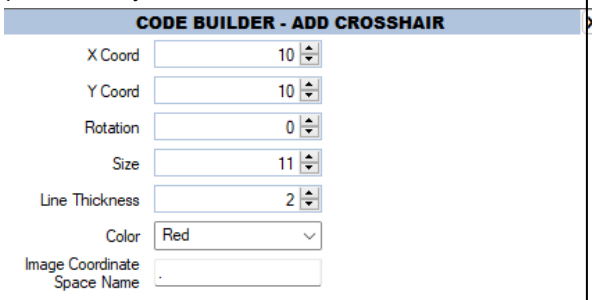
```
//Adding Label
AddLabel("Label Text Here",10,10,CogColorConstants.Cyan,
        CogGraphicLabelAlignmentConstants.TopLeft,null,14,
        CogColorConstants.Red,"@");
```

</td>
</tr>
</table>

<table>
<tr>
<td>Add Rectangle</td>
<td>Adds a rectangle to the image view based on the parameters you set.
 

X Coord: Set the X coordinate of the rectangle.
Y Coord: Set the Y coordinate of the rectangle.
Height: Set the height of the rectangle.
Width: Set the width of the rectangle.
Color: Select the color of the rectangle.
Image Coordinate Space Name: Set the coordinate space.
</td>
<td> 
</td>
</tr>
<tr>
<td>Add Circle</td>
<td>Adds a circle to the image view based on the parameters you set.
 

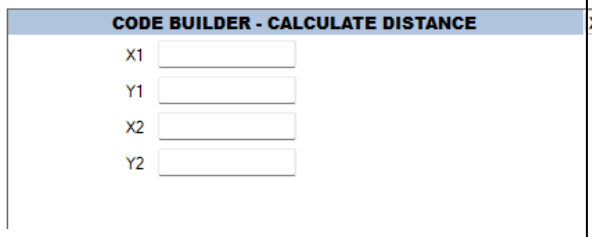
X Coord: Set the X coordinate of the circle.
Y Coord: Set the Y coordinate of the circle.
Radius: Radius of the circle.
Color: Select the color of the circle.
Image Coordinate Space Name: Set the coordinate space.
</td>
<td> 
</td>
</tr>
</table>

<table>
<tr>
<td>Add Crosshair</td>
<td>Adds a crosshair to the image view based on the parameters you set.
 

X Coord: Set the X coordinate of the crosshair.
Y Coord: Set the Y coordinate of the crosshair.
Rotation: Rotation of the crosshair.
Size: Size of the crosshair.
Line Thickness: Thickness of the crosshair line.
Color: Select the color of the crosshair.
Image Coordinate Space Name: Set the coordinate space.

</td>
<td>

```
public void Run()
{
    // Put script code here
    AddCrosshair(10,10,0,11,2,CogColorConstants.Red,".");

}
```

</td>
</tr>
<tr>
<td>Calculate DIstance</td>
<td>Returns the distance between two points. Returns pixel measurements if not calibrated.
 

X1: X coordinate of the first point.
Y1: Y coordinate of the first point.
X2: X coordinate of the second point.
Y2: Y coordinate of the second point.

</td>
<td>

```
public void Run()
{
    // Put script code here
    double distance = CalculateDistance(10,10,100,100);

}
```

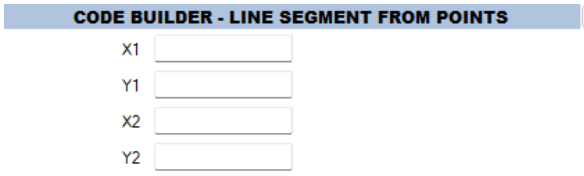
</td>
</tr>
<tr>
<td>Covert Radius to Degree</td>
<td>Converts value in radius to value in degrees.
 

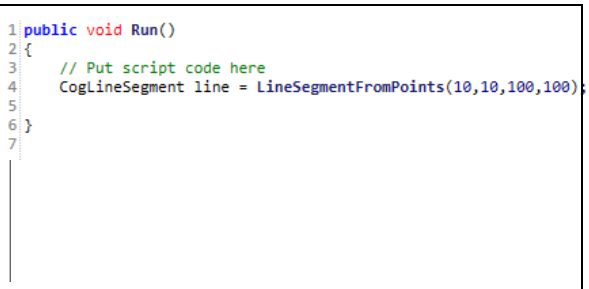
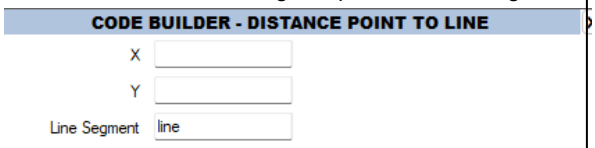
Radians: Enter radius value you want to convert to degree.

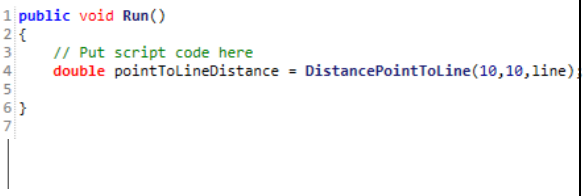
</td>
<td>

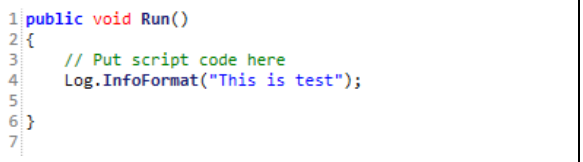
```
public void Run()
{
    // Put script code here
    double degrees = CogMisc.RadToDeg(100);

}
```

</td>
</tr>
</table>

<table>
<tr>
<td>Line Segment From Points</td>
<td>Generates a line segment from the points that are provided.
 

• X1: X coordinate of first point of the line.
• Y1: Y coordinate of first point of the line.
• X2: X coordinate of second point of the line.
• Y2: Y coordinate of second point of the line.</td>
<td> 
</td>
</tr>
<tr>
<td>Distance Point to Line</td>
<td>Finds the distance from a given point to a line segment.
 

• X1: X coordinate of point.
• Y1: Y coordinate of point.
• Line Segment: Name of the line segment that you want to use for the measurement.</td>
<td> 
</td>
</tr>
<tr>
<td>Log Entry</td>
<td>Adds a log entry to the information log.
 

• Level: Select the log level of entry. It could be Debug, Info, Warning, or Error.
• Message: Text message you want to enter.</td>
<td> 
</td>
</tr>
</table>

Script UI Controls

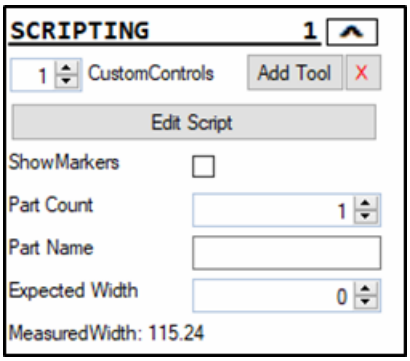
EtherInspect allows you to make your own user-defined controls using scripting.

By creating public static properties at the beginning of a script file, you can modify values at runtime without editing the script.

Placing an attribute called **UserSetting** before the variable definition allows you to define a display name for that value within the controls and reserves an area to place a user-defined control in.

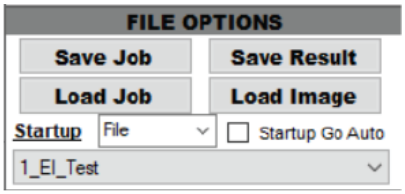
```
[UserSetting(DisplayName = "ShowMarkers")]
public static bool ShowMarkers { get; set; } = false;

[UserSetting(DisplayName = "Part Count", DecimalPlaces = 0, MinRange = 1, MaxRange = 5)]
public static int PartCount { get; set; } = 1;

[UserSetting(DisplayName = "Part Name")]
public static string PartName { get; set; } = string.Empty;

[UserSetting(DisplayName = "Expected Width", DecimalPlaces = 3, MinRange = 1.0, MaxRange = 250.0)]
public static double ExpectedWidth { get; set; } = 0.0;

[UserSetting(DisplayName = "Measured Width", DecimalPlaces = 3)]
public static double MeasuredWidth { get; } = 115.24;

public void Run()
{

}
```

Example of the controls created by the script:



File Options

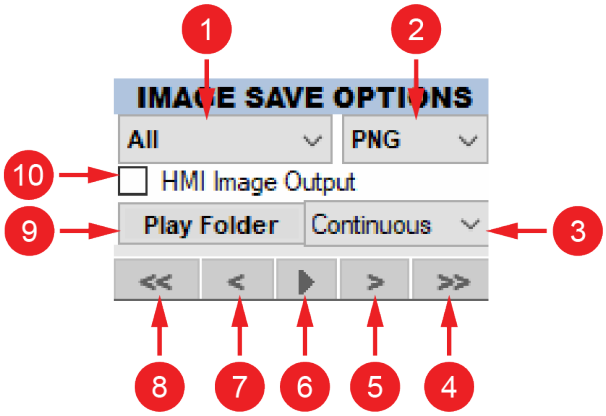
Allows loading or saving jobs and images, and setting startup jobs.



Name	Description
Save Job	Saves Job to disk, need to specify name of job to save as.
Save Result	Saves inspection results including PNG and XML.
Load Job	Loads a job from disk into selected camera.
Load Image	Loads image from disk for inspection.
Job Selection Type	Specify method to select job as startup job.
Startup Online	When application starts up and job loads, go into AUTO mode.
Job	Specify job to use as startup job.

Image Save Options

Image Save Options has settings for image save types, and also allows playing back images.

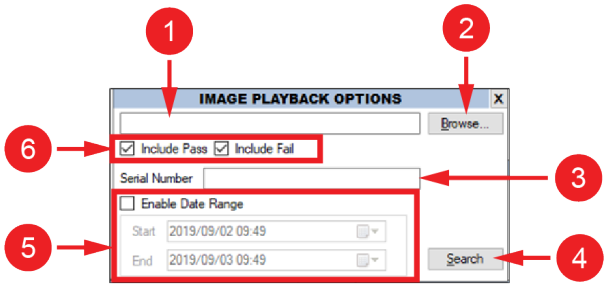


Number	Name	Description
1	Image Save Filter	Allows you to filter which images to save based on inspection result. The four options are: • **NONE**: No images. • **FAIL**: Any images that failed inspection. • **PASS**: Any images that passed inspection. • **ALL**: Save both types of images.
2	Image Type	Allows you to choose what type of format you want the images to be saved in. The five supported image types are: • BMP • PNG • JPG • CDB • WEBP When both of the above settings are set, the images are saved to the directory as specified by the Storage Directory setting. Within specified directory, images are saved to `<CameraName>/Images/<JobName>`.
3	Image Playback Options	Opens the **Image Playback Options** display. A dialog opens to search for images within a directory. You can filter out PASS or FAIL images.
4	Last Image	Opens the last image.
5	Next Image	Opens the next image.
6	Play	Starts image playback.
7	Previous Image	Opens the previous image.
8	First Image	Opens the first image.

9	Playback Rate	Allows you to choose when the image playback stops. The four options are: • **Continuous**: Loop images over and over. • **RunToEnd**: Playback images until end of folder is reached. • **StopOnFail**: Stop on image with FAIL inspection. • **StopOnPass**: Stop on image with PASS inspection.
10	HMI Image Output	Enables saving a thumbnail image with graphics.

Image Playback Options

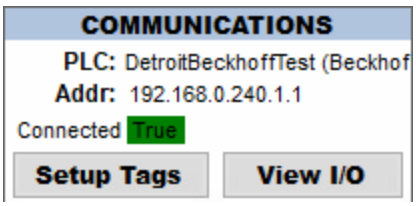
Sets criteria to search for images to be played.



Number	Description
1	Folder Path
2	Browse Folders
3	Serial Number
4	Search Folder
5	Date Time Range Filter
6	Pass/Fail Filter

Communications

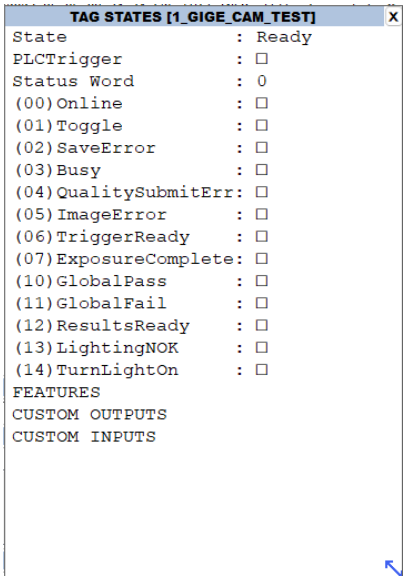
Displays PLC connection state. Allows viewing IO and editing Tags.



- **Setup Tags** - Shows Setup Tags Display
- **View I/O** – Shows Tag States Display

Tag States

Shows current values of PLC Tags and current state of camera thread, 100 ms refresh rate.

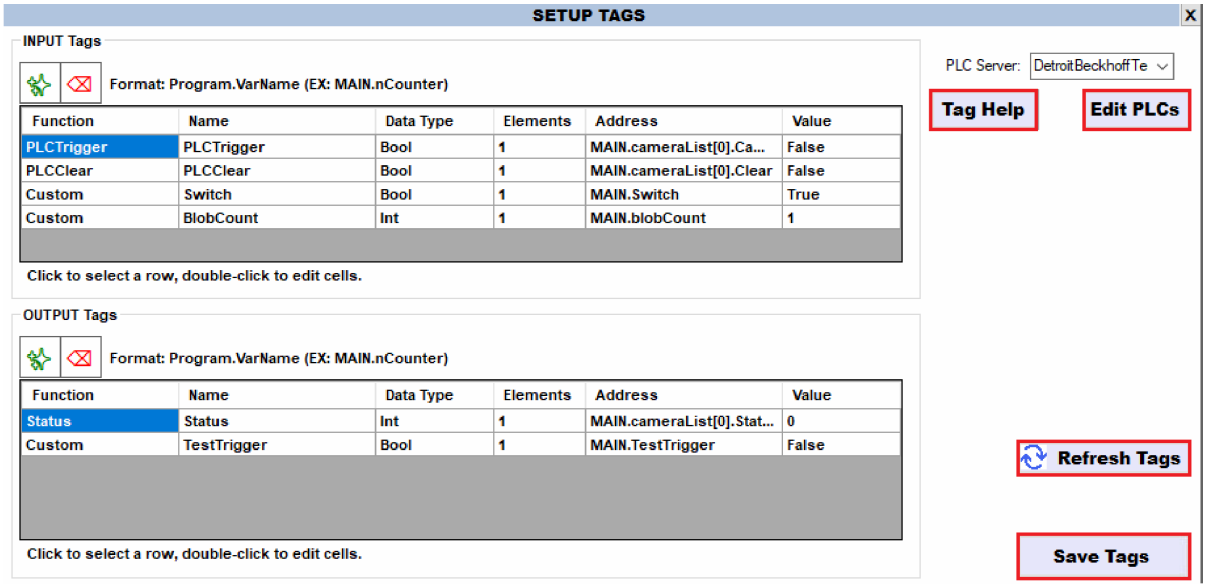


Each tag state is represented by a bit with its corresponding state that it represents being next to the status bit.

Bit	State	Description
0	Online	Camera is online and in AUTO mode.
1	Toggle	Camera trigger confirmed.
2	Save Error	Error saving image.
3	Busy	Busy acquiring image that is already in buffer.
4	Quality Submit Error	Error sending data to QualityWorX.
5	Image Error	Error acquiring image.
6	Trigger Ready	Camera ready to trigger.
7	Exposure Complete	Image exposure acquisition complete.
8	Unused	Unused bit.
9	Unused	Unused bit.
10	Global Pass	Job passed inspection.
11	Global Fail	Job passed inspection.
12	Results Ready	Job results are ready to send.
13	LightingNOK	Light controller faulted.
14	TurnLightOn	Triggers light controller if configured.

Setup Tags

Allows editing tags and displaying current values.



The functions of the Setup Tag settings are:

- **Tag Help**: Shows the Tag Help Display, with information on tag functions and valid data types.
- **Edit PLCs**: Shows the Edit PLCs Display, to add or delete PLC assignments.
- **Refresh Tags**: Reloads the tags and updates the values.
- **Save Tags**: Saves any newly added, removed, or updated tags. Closing the Setup Tags display also triggers this function.

Tag Types

EtherInspect has two types of tags:

- **Input**: Reads incoming data coming from the PLC.
- **Output**: Writes to outgoing data to the PLC.

Both tag sections contain a list of PLC tags that link with the PLC server selected.

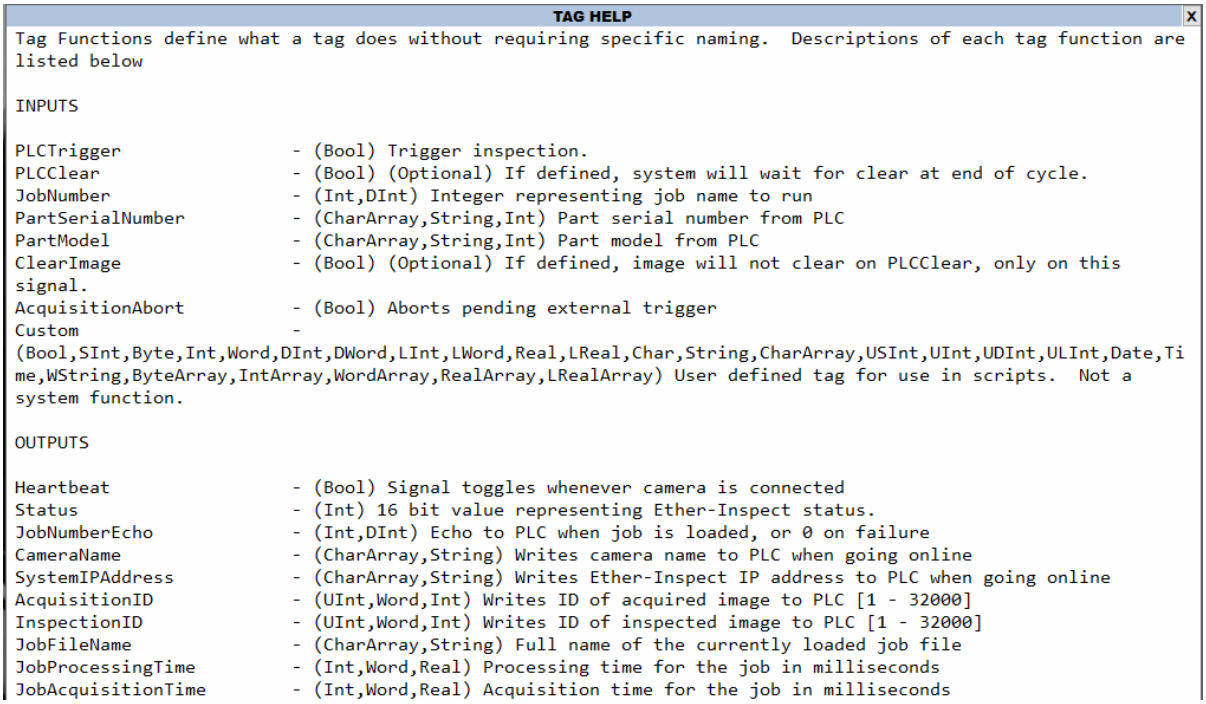
Tag Attributes

The attributes to a tag in EtherInspect are:

- **Function**: Tag function to be used with EtherInspect. If not custom, the tag uses built-in function to grab a certain value from the PLC.
- **Name**: Name of tag. Does not have to be named after variable or tag in PLC. The value is only referenced when there are tags with the same function to differentiate between them.
- **Data Type**: Data type of tag value.
- **Elements**: Number of items stored in the tag.
- **Address**: Memory address of tag that refers to variable or tag in PLC.
- **Value**: Current value of tag.

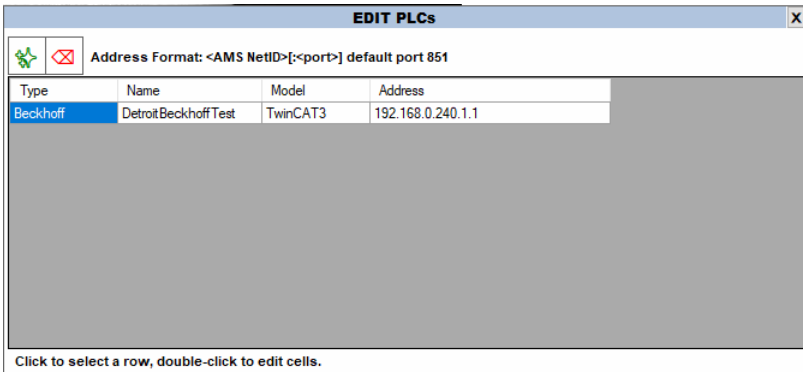
Tag Help

Shows information on tag functions and valid data types.



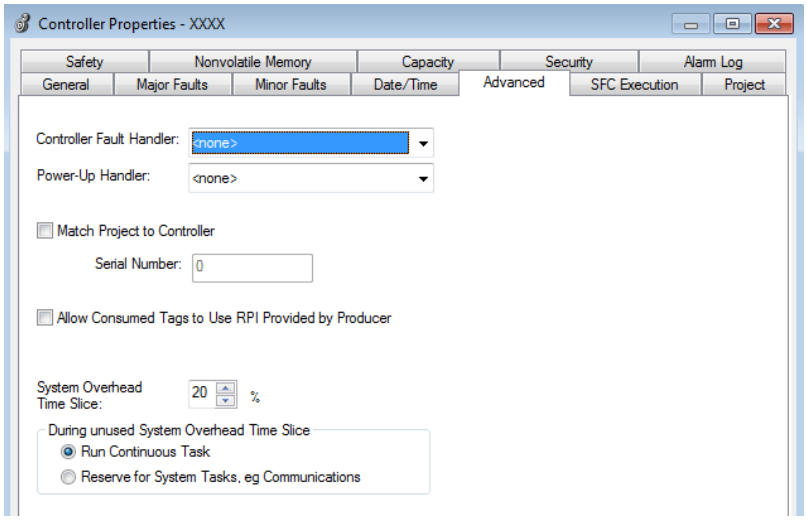
Edit PLCs

Allows editing PLC connections. You can only use one PLC vendor at one time.



Allen-Bradley PLC Information

For Allen-Bradley PLCs, if you experience slow communication, increase the **System Overhead Time Slice** setting in the processor.

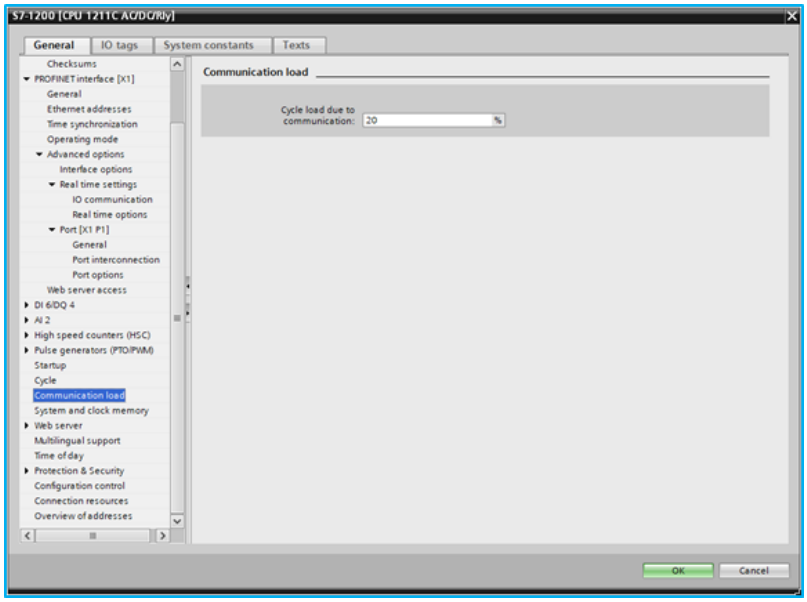


Siemens PLC Information

For Siemens PLCs, if you experience slow communication, increase the **CPU Load due to communication** value as needed, up to 50%.

TIA Portal

The TIA Portal option is under *CPU Properties* > *General* > *Communication Load*.



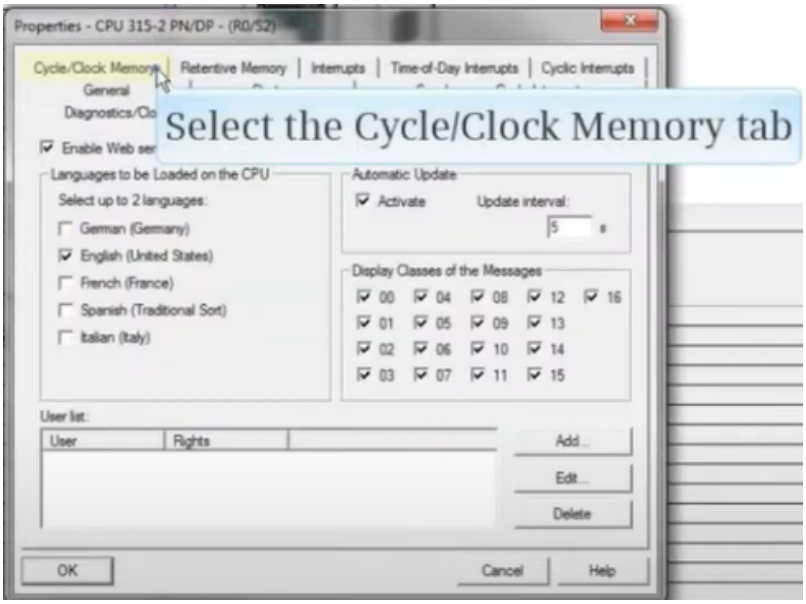
Simatic Manager Step 7

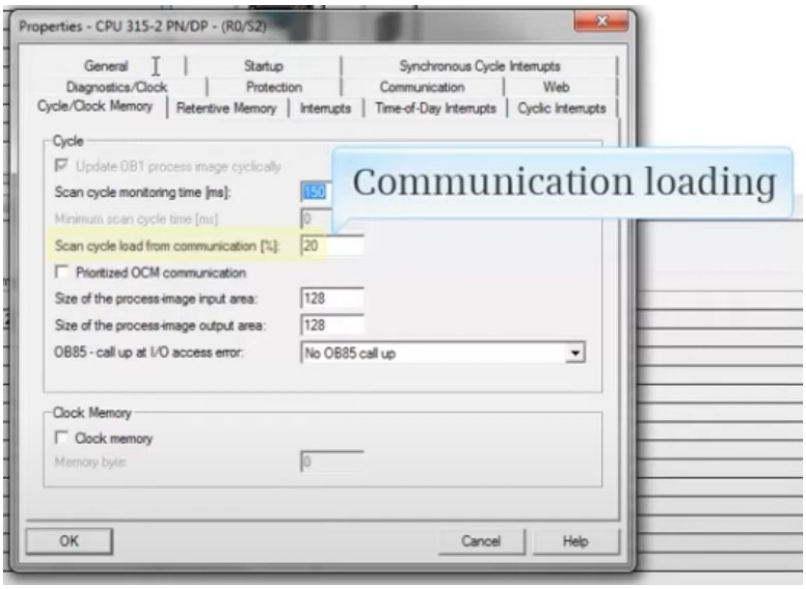
To modify communication load, use the **Scan Cycle Load from Communication** setting.

To access the setting:

1. Double-click the CPU in the list.
2. Go to the Cycle/Clock Memory tab.
3. If the CPU is for communication purposes, increase the **Scan Cycle Load from Communication** value as needed, up to 50%.

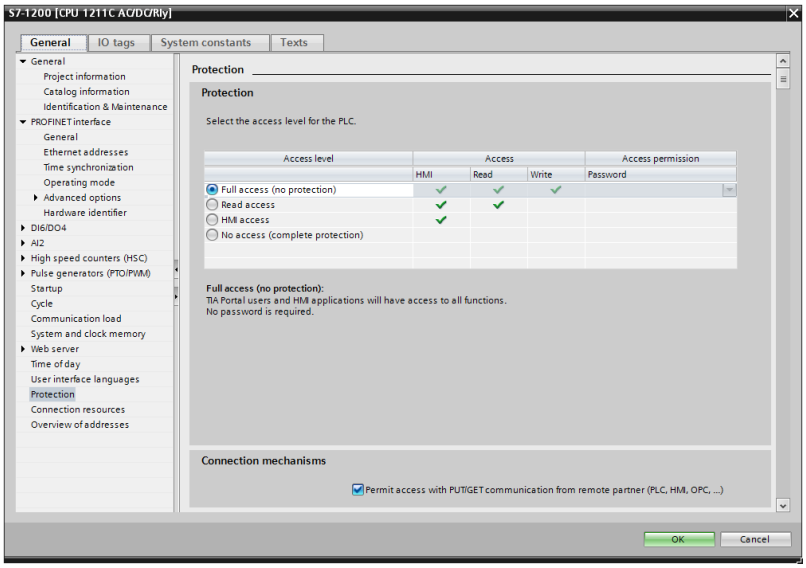
4. Monitor the cycle time from the HW config:
    a. Go to *HW Config* > *Open Station Online*.
    b. Select the PLC.
    c. Right-click your PLC.
    d. Go to **Module Information**.
    e. Click the Scan Cycle Time tab.
5. Adjust scan cycle load until you achieve the desired performance.





Siemens S7-1200 / S7-1500 Settings

If you receive the error code 0xffff8104 or 0x00008104 when reading or writing from a Siemens S7-1200 or S7-1500 with firmware version 4.0 or higher, change the **Protection** setting in TIA Portal.

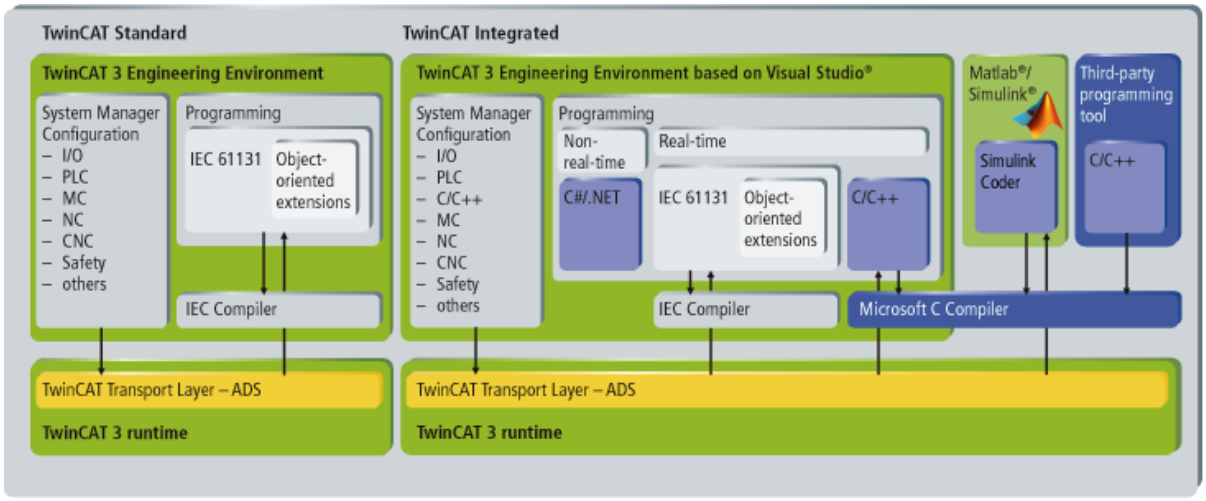


To change the protection settings:

1. Open the PLC project in the S7 TIA Portal software.
2. Go to Project View.
3. In the Project Tree window, select the device causing the error.
4. Click *Alt+Enter* to open the Properties dialog.
5. On the general tab, select **Protection**.
6. Make sure you check one of the following options:
    - Full Access (No Protection)
    - Read Access
    - HMI Access
7. Check Permit Access with Put/Get communication with remote partner.
8. Click **OK** to save the settings.
9. Compile and save project. Download modified program to controller

Beckhoff PLC Information

For Beckhoff PLCs, install a driver package. Either install the Beckhoff TC1000 ADS router or the complete TwinCAT3 suite.

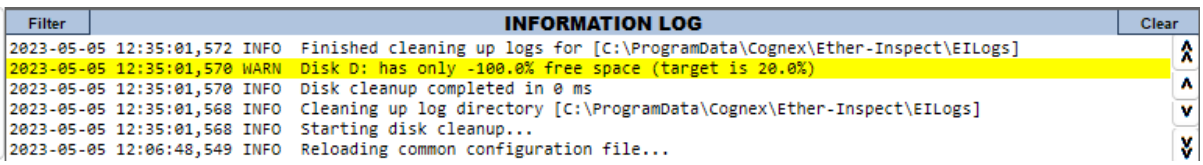


Information Log

Shows log entries, holds 1000 entries in memory, and all are logged to the disk.

The information log includes five levels of log entries:

- `INFO`: High-Level log of the process flow of EtherInspect.
- `WARN`: Warnings (highlighted in yellow in the `INFO` log) notifying the user of any important events to monitor.
- `ERROR`: Errors and exceptions (highlighted in red in the `INFO` log) that occurred during the current instance of EtherInspect.
- `FATAL`: Errors that EtherInspect cannot recover from, causing the application to shut down immediately.
- `DEBUG`: Extensive debug information on processes and events occurring within EtherInspect. Available through the Log Viewer Display.



Note: Within the minimized log, the most recent log entries appear from top to bottom.

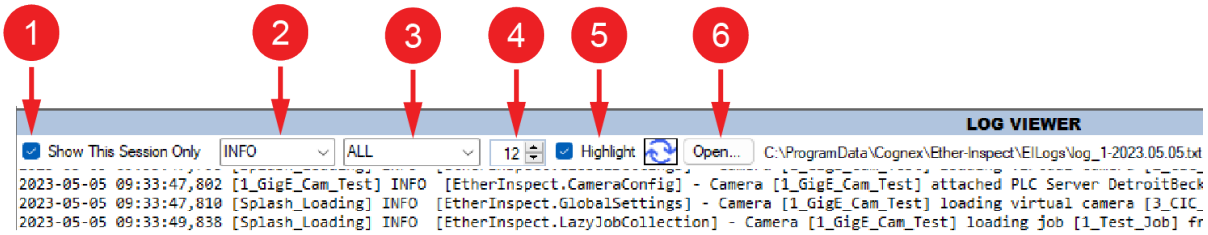
Click **Filter** to flush the logs to disk and show the Log Viewer Display.

Click **Clear** to clear the visible display. Log files are unaffected.

Log Viewer

Allows viewing and filtering log files from different cameras or sessions in EtherInspect.

The log viewer shows an additional level of log:



Number	Setting
1	Toggle Session
2	Log Level
3	Camera Filter
4	Font Size
5	Highlight Warnings & Errors
6	Open Specific Log

Note: Within the expanded log view, the most recent log entries appear from bottom to top.

Exporting Log Files

To export a log file within a certain time frame, specify the dates in the top right after expanding the information log and click **Export**.

This prompts you to specify a location to save the .txt file. To export the log for the current day, keep the end date as today and set the start date to the previous day.

Status Bar

The status bar displays the currently logged-in user, date, and time. The status bar is located at the bottom of the window.

Versions Display

Shows version info of EtherInspect system and DLLs that are used (including templates). Click the EtherInspect logo in the bottom left corner of the screen to open the versions dialog.

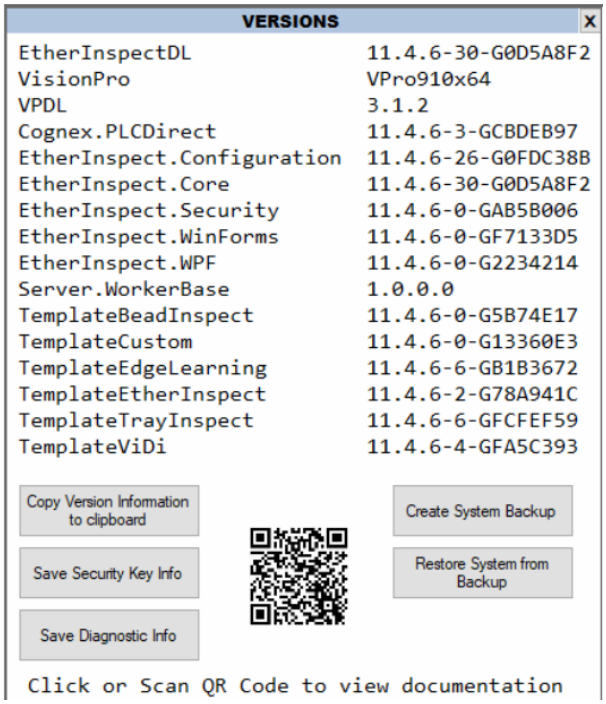
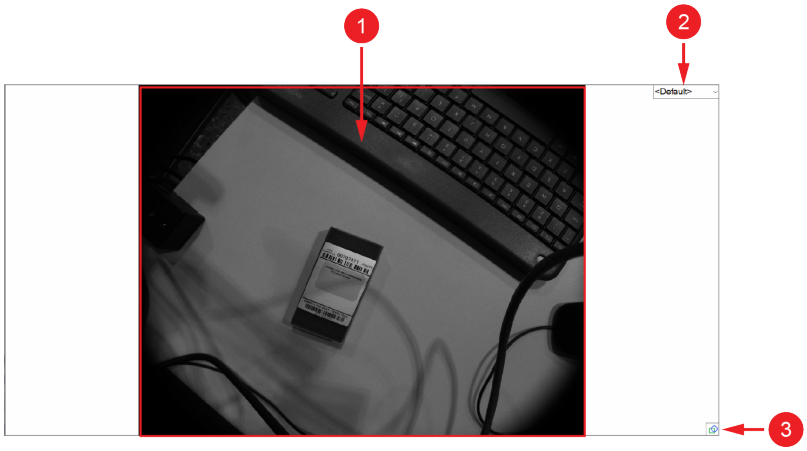


Image Display

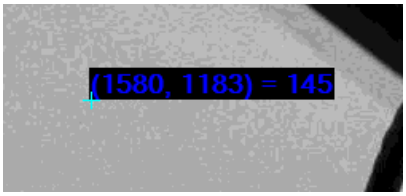
Shows camera image with graphics.

Note: In MultiCam mode, the image has a title.

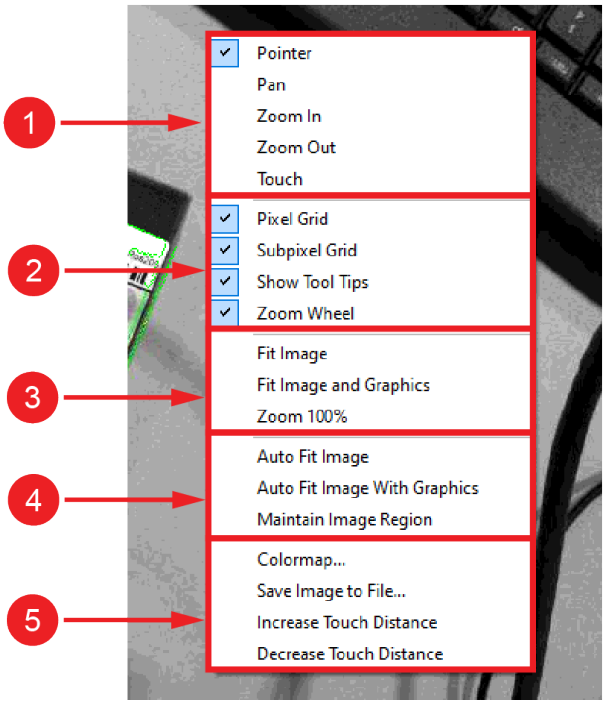


Number	Name	Description
1	Image	Image acquired from camera or loaded from file.
2	Image Filter	Drop-down menu to choose between different image filters generated by tools.
3	Graphics	Changes graphics overlay option (All graphics, Fail Only graphics, No graphics).

Click anywhere in the image shows the pixel coordinates of that point.



Right-click the image to show additional options:



Number	Name
1	Pointer Modes
2	Show and Hide Features
3	Fit and Zoom
4	Auto Fit
5	Miscellaneous Options

Color Map

The Color Map feature allows you to modify the distribution of colors within an image.

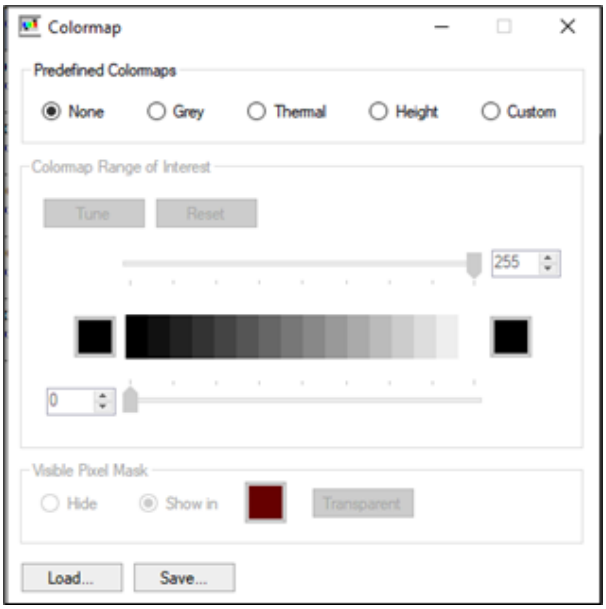
The four types of color maps are:

- Greyscale
- Thermal
- Height
- Custom

You can tune the color map to make parts of the image stand out. This can be useful when you have a 3D image with embossed letters, and you want the letters to stand out in the image.

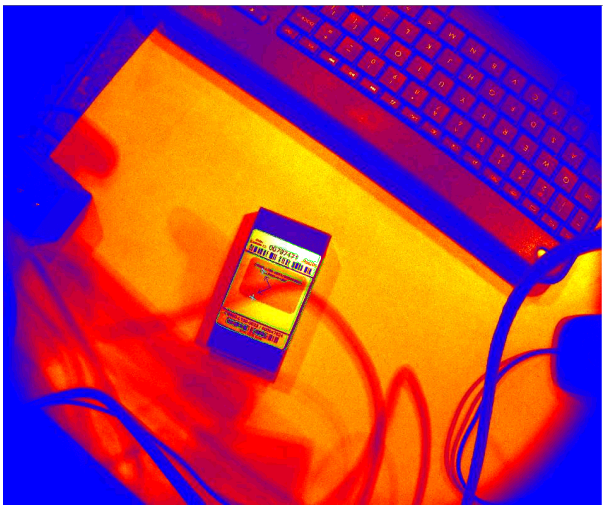
Color maps are for display only and are not for processing or image acquisition. VisionPro tools use the original image, which can detect very small contrast differences. If additional contrast is necessary for vision tools, image filters should be used.

When setting the **Visible Pixel Mask**, you can select the color that missing pixels are displayed as, such as black.



Name	Description
Predefined Colormaps	Choose color map type.
Tune / Reset	Tune the color map after making changes or reset to default.
Max Color Range	Maximum range of hue.
Color Spectrum	Mapping of pixel values with color.
Min Color Range	Minimum range of hue.
Save	Save current color map.
Load	Load a color map from file.

Example (Thermal):



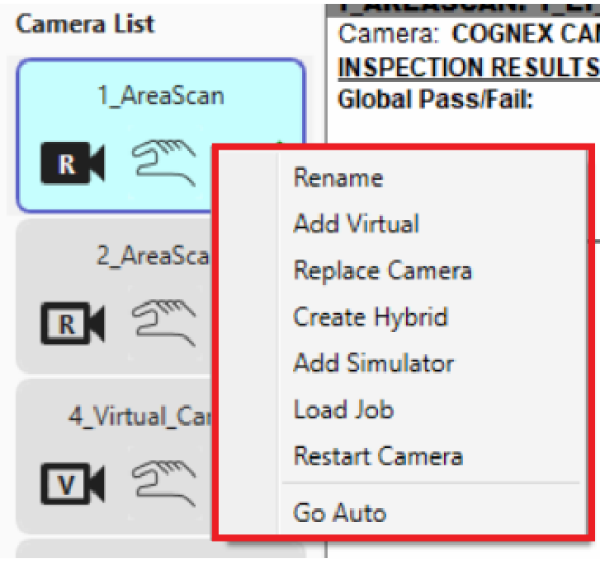
Camera List

The UI shows the settings of the camera. Clicking a camera makes its configurations, tools, and image feed display to the right.

Number	Identifier
1	Camera Name
2	Camera Type (Real, Virtual, Hybrid, or Simulator)
3	Running Mode (Auto or Manual)
4	Inspection Status (Pass or Fail)

Additional Camera Settings

Right-click on a camera to open the settings.

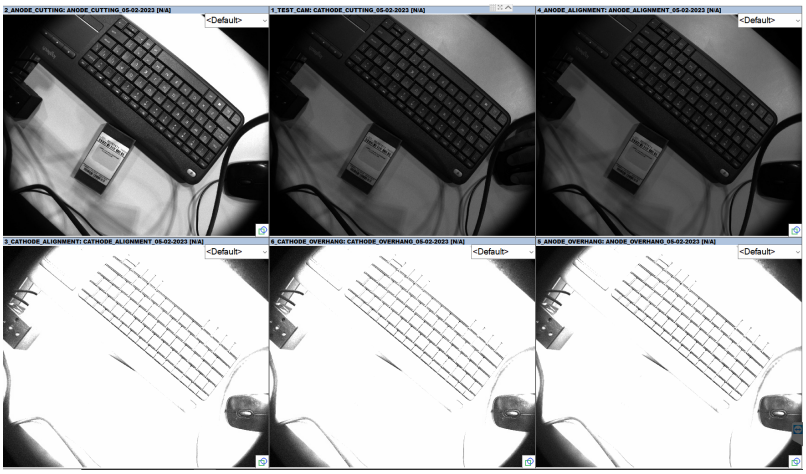


MultiCam View

MultiCam View displays images from all cameras on one or more pages. You can add multiple pages to MultiCam View based on camera settings. Click the button again to change to the next page if configured.

Click the following icon in the top right corner of the EtherInspect window to open MultiCam View.

Click the button again to go to the next page of cameras, if configured.



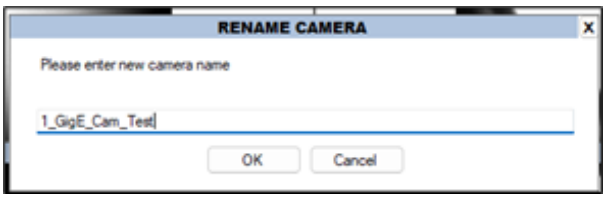
In the settings of each camera, two settings change the look and feel of the MultiCam view: **MultiCam Page #** and **MultiCam Sort Order**.

- **MultiCam Page #** sets the page number that the camera appears on. You can have a maximum of four pages in MultiCam view, each with a set of camera views.
- **MultiCam Sort Order** determines the order that the camera on the page it is assigned to. You can have up to 20 cameras on one page. The order of cameras appear from left to right, starting from top to bottom.

Rename the Camera

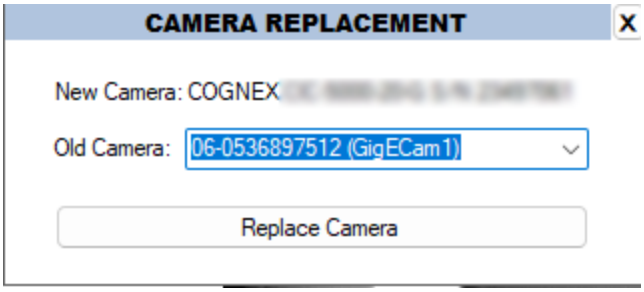
Allows you to rename a camera.

1. Right-click on the camera to open the additional settings.
2. Select **Rename**. The **Rename Camera** dialog opens.
3. Type in the new name for the camera.
4. Press **OK**.



Camera Replacement Display

Allows you to swap a camera with one of a different serial number. Right-click on the camera to open the additional settings. Click **Replace Camera** to open the **Camera Replacement** dialog.


CAMERA REPLACEMENT
New Camera: COGNEX
Old Camera:
06-0536897512 (GigECam1)
Replace Camera

EtherInspect Remote Interfaces

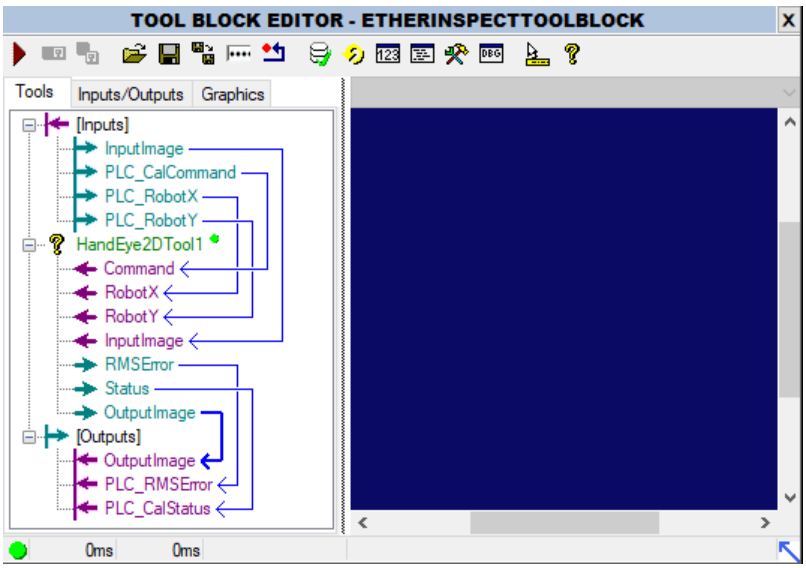
For all these interfaces, EtherInspect acts as the server and listens on the specified port.

Interface Name	Connection Method	Description
HANDEYE2D PLC CONTROL INTERFACE	PLC Tag Interface	Interface for performing robot hand-eye calibration through PLC tags.
TCP REMOTE INTERFACE	TCP Port 9000 + Instance # (9001 – 9006)	General remote-control interface over TCP socket.
RESTFUL WEB SERVER	HTTP Port 80 + Instance # (81 – 86)	RESTful web services interface for remote control.
IMAGE WEB SERVER	HTTP Port 80	Serves camera images for PLC HMIs.
VISIONLINE INTERFACE	TCP Port 52000	Allows remote control from Trumpf laser.

HandEye2D PLC Control Interface

The HandEye2D tool supports a ToolBlock terminal-based interface that can be linked to PLC tags for control.

Some parts of calibration are manual, such as setting the number of points and configuring the calibration target finder (circle or pattern).



Inputs

The inputs are RobotX, RobotY, and Command.

RobotX: Double value representing the X value of the location in the coordinate system of the robot.

RobotY: Double value representing the Y value of the location in the coordinate system of the robot.

Command has several possible values:

- 0 – Do Nothing
- 1 – 90 – Register calibration point 1 – 90

- 100 – Compute N Point calibration
- 150 – Calibrate checkerboard

Outputs

The outputs are RMSError and Status.

RMSError: The error of the N Point calibration fit. This is the only output if Command 100 is requested.

Status: Indicates the result of a Command. The values are:

- 0 – No Action was taken
- 1 – 90 – Point was registered successfully
- 100 – 190 – Point registration failed
- 198 – Argument Error
- 199 – Calculation Error
- 200 – Calculation Ok

Transmission Control Protocol Remote Interface

You can perform some operations on the Transmission Control Protocol (TCP) based remote interface in EtherInspect. Each application instance has an individual TCP listener. You can separately control each application instance.

After the application is connected to the TCP server on port 9001 for instance 1, or 9002 for instance 2, and so on, you can execute the commands:

Command	Description	Request	Response

Cameras	Returns list of available cameras from application instance.	`CAMERAS;`	String format - `{command},{number_of_camera},{comma_separated_cam_name};`
			`Enter a TCP request:`
			`Cameras`
			`Response Message -> Cameras,3,1_Test_Cam,2_Virtual_Cam,CIC_LineScan;`

Trigger	Use the command to trigger the camera.	`SHOWSINGLECAM,{camera_name};`
		camera_name (Datatype – string) – a valid camera name. Use the camera name returned from CAMERAS command.
			String format - `{command},{result};`
			`Enter a TCP request :`
			`SHOWSINGLECAM,1_Test_Cam`
			`Response Message -> SHOWSINGLECAM,False;`
			Note: “NOK” is returned in case of any issue.

AppName	Get the application name.	`APPNAME;`	String format - `{command},{application_name};`
			`Enter a TCP request :`
			`APPNAME`
			`Response Message -> APPNAME,EtherInspectDL;`

AppVersion	Get the application version.	`APPVERSION;`	String format - `{command},{application_version};`
			`Enter a TCP request :`
			`APPVERSION`
			`Response Message -> APPVERSION,11.3.39-49-G7CFC896;`

InstanceNumber	Get the application instance number.	`INSTANCENUMBER;`	String format - `{command},{instance_number};`
			`Enter a TCP request :`
			`INSTANCENUMBER`
			`Response Message -> INSTANCENUMBER,1;`

CameraType	Get camera type by camera name.	`CAMERATYPE,{camera_name};`
		camera_name (Datatype – string) – a valid camera name. Use the camera name returned from CAMERAS command.
			String format - `{command},{camera_type};`
			`Enter a TCP request :`
			`CAMERATYPE,1_Test_Cam`
			`Response Message -> CAMERATYPE,RealCamera;`

LoadJob	Load job by number.	`LOADJOB,{camera_name},{job_number};`
		camera_name (Datatype – string) – a valid camera name
		job_number (Datatype – int) – a job number you want to load
			String format - `{command},{result};`
			`Enter a TCP request :`
			`LOADJOB,1_Test_Cam`
			`Response Message -> LOADJOB,True;`

JobName	Get current job name as string.	JOBNAME,{camera_name}; camera_name (Datatype – string) – a valid camera name	String format - {command},{job_name}; Enter a TCP request: JOBNAME,1_Test_Cam Response Message -> JOBNAME,2_Another_Job;
JobNumber	Get current job number.	JOBNUMBER,{camera_name}; camera_name (Datatype – string) – a valid camera name	String format -{command},{job_number}; Enter a TCP request: JOBNUMBER,1_Test_Cam Response Message -> JOBNUMBER,2;
GoAuto	Put camera into auto mode.	GOAUTO,{camera_name}; camera_name (Datatype – string) – a valid camera name	String format - {command},{result}; Enter a TCP request: GOAUTO,1_Test_Cam Response Message -> GOAUTO,True;
GoManual	Put camera into manual mode.	GOMANUAL,{camera_name}; camera_name (Datatype – string) – a valid camera name	String format -{command},{result}; Enter a TCP request: GOMANUAL,1_Test_Cam Response Message -> GOMANUAL,True;
State	Get current state of camera.	STATE,{camera_name}; camera_name (Datatype – string) – a valid camera name	String format - "{command},{state};" Enter a TCP request: STATE,1_Test_Cam Response Message -> STATE,Ready;
Mode	Get current trigger mode.	MODE,{camera_name}; camera_name (Datatype – string) – a valid camera name	String format -{command},{mode}; Enter a TCP request: MODE,1_Test_Cam Response Message -> MODE,Manual;

InputTags	Returns a list of input tags configured for camera.	`INPUTTAGS,{camera_name};` camera_name (Datatype – string) – a valid camera name	String format - `{command},{number_of_tags},{comma_separated_tag_name};`
			`Enter a TCP request:`
			`INPUTTAGS,1_Test_Cam`
			`Response Message ->`
			`INPUTTAGS,4,PLCClear,PLCTrigger,Switch,First;`

OutputTags	Returns a list of output tags configured for camera.	`OUTPUTTAGS,{camera_name};` camera_name (Datatype – string) – a valid camera name	String format - `{command},{number_of_tags},{comma_separated_tag_name};`
			`Enter a TCP request :`
			`OUTPUTTAGS,1_Test_Cam`
			`Response Message ->`
			`OUTPUTTAGES,4,Heartbeat,Status,CounterEcho,TestSwitch;`

InputTagType	Get data type of input tag.	`INPUTTAGTYPE,{camera_name},{tag_name};` camera_name (Datatype – string) – a valid camera name tag_name (Datatype – string) – a valid tag name. Use INPUTTAGS to get list of input tag names	String format - `{command},{tag_type};`
			`Enter a TCP request:`
			`INPUTTAGTYPE,1_Test_Cam,PLCClear`
			`Response Message -> INPUTTAGTYPE,System,Boolean;`

OutputTagType	Get data type of output tag.	`OUTPUTTAGTYPE,{camera_name},{tag_name};` camera_name (Datatype – string) – a valid camera name tag_name (Datatype – string) – a valid tag name. Use OUTPUTTAGS to get list of output tag names	String format - `{command},{tag_type};`
			`Enter a TCP request:`
			`OUTPUTTAGTYPE,1_Test_Cam,CounterEcho`
			`Response Message -> OUTPUTTAGTYPE,System.Int16;`

ReadInputTag	Read value of input tag.	`READINPUTTAG,{camera_name},{tag_name};`
		camera_name (Datatype – string) – a valid camera name
		tag_name (Datatype – string) – a valid tag name. Use INPUTTAGS to get list of input tag names
			String format - `{command},{tag_value};`
			`Enter a TCP request:`
			`ReadInputTag,1_Test_Cam,PLCTrigger`
			`Response Message -> ReadInputTag,False;`

ReadOutputTag	Read value of output tag.	`READOUTPUTTAG,{camera_name},{tag_name};`
		camera_name (Datatype – string) – a valid camera name
		tag_name (Datatype – string) – a valid tag name, please use OUTPUTTAGS to get list of output tag names
			String format - `{command},{tag_value};`
			`Enter a TCP request:`
			`READOUTPUTTAG,1_Test_Cam,CounterEcho`
			`Response Message -> READOUTPUTTAG,0;`

WriteTag	Write value to output tag.	`WRITETAG,{camera_name},{tag_name},{value};`
		camera_name (Datatype – string) – a valid camera name
		tag_name (Datatype – string) – a valid tag name. Use OUTPUTTAGS to get list of output tag names
		value (Datatype – object) – value you want to write, make sure datatype is matching
			String format - `{command},{result};`
			`Enter a TCP request:`
			`WRITETAG,1_Test_Cam,TestSwitch,True`
			`Response Message -> WRITETAG,True;`
			Note: Result is true if data was written successfully. Otherwise, the result is false.

HandEye2D Calibration TCP Commands

These TCP commands that can initiate HandEye2D Calibration, allowing you to run calibration from a remote system.

Command	Description	Request	Response

StartC	Start HandEye2D calibration.	`STARTC,{camera_name};`
		camera_name (Datatype – string) – a valid camera name
			String format - `{command},{result};`
			`Enter a TCP request:`
			`STARTC,1_Test_Cam`
			`Response Message -> STARTC,0;`
			Note: The result is “1” if successful. Otherwise, the result is “0”.

Cal	Register HandEye2D calibration point in Robot mode.	`CAL,{camera_name},{point_number},{robot_X},{robot_Y};`
		camera_name (Datatype – string) – a valid camera name
		point_number (Datatype – int) – Cal point number 1 -90
		robot_X (Datatype – double) – Robot Pose X Coordinate
		robot_Y (Datatype – double) – Robot Pose Y Coordinate
			String format - `{command},{result};`
			`Enter a TCP request:`
			`CAL,1_Test_Cam,180,100`
			`Response Message -> CAL,0;`
			Note: The result is "1" if successful. Otherwise, the result is "0".

HECal	Registers HandEye2D Calibration point in Motion Stage mode.	`HECAL,{camera_name},{point_number},{X},{Y},{Theta};`
		camera_name (Datatype – string) – a valid camera name
		point_number (Datatype - int) – Calibration point number (1-90)
		X (Datatype – double) – Motion stage Pose X Coordinate
		Y (Datatype - double) – Motion stage Pose Y Coordinate
		Theta (Datatype – double) – Motion stage pose theta
			String Format – `{command},{result};`
			`Enter a TCP request:`
			`HeCal,1_AreaScan,1,10,10,90`
			`Response Message -> HeCal,1;`
			Note: The result is "1" if successful. Otherwise, the result is "0".

EndC	End HandEye2D calibration and compute result.	`ENDC,{camera_name};`
		camera_name (Datatype – string) – a valid camera name
			String format - `{command},{result};`
			`Enter a TCP request:`
			`ENDC,1_Test_Cam`
			`Response Message -> ENDC,0;`
			Note: The result is "1" if successful. Otherwise, the result is "0".

RESTful Web Service

You can perform some operations on the HTTP RESTful web server interface in EtherInspect. Each application instance has an individual web server. You can separately control each application instance.

TCP Port: Application instance number + 80. E.g. 81 for the first EtherInspect instance.

`{cameraname}`: A valid camera name. Use the camera name returned from list of cameras URL.

`{jobnumber}`: Pass the job number which you want to change before triggering. Use -1 if you want to trigger with the current job.

URL	Returns
http://server-name-or-ip:81/eiremote/v1/version	string - remote api version

http://server-name-or-ip:81/eiremote/v1/appname	string - application name
http://server-name-or-ip:81/eiremote/v1/appversion	string - EtherInspect application version
http://server-name-or-ip:81/eiremote/v1/instancenumber	int - application instance number
http://server-name-or-ip:81/eiremote/v1/cameras	string - list of available cameras from application instance
http://server-name-or-ip:81/eiremote/v1/showmulticam	boolean (success/fail) **Note**: Use the URL request to change the Camera view to MultiCam.
http://server-name-or-ip:81/eiremote/v1/showsinglecam/{cameraname}	boolean (success/fail) **Note**: Use the URL request to change the Camera view to single cam.
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/typename	string – camera type name
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/trigger	boolean – inspection result **Note**: The URL request triggers the camera and runs an inspection with current job.
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/trigger/{jobnumber}	boolean – inspection result
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/triggerwithresult	json based JobResult
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/triggerwithresult/{jobnumber}	json based JobResult
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/loadjob/{jobnumber}	boolean (success/fail) **Note**: Camera must be in Auto mode. **Note**: Uses HTTP PUT command instead of GET. Example: `curl -X PUT http://10.1.1.1:81/eiremote/v1/cameras/1_Test/loadjob/3 -d ""`
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/jobname	string – current job name
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/jobnumber	int – current job number
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/goauto	boolean (success/fail) **Note**: Changes current camera mode to Auto. **Note**: Uses HTTP PUT instead of GET. Example: `curl -X PUT http://10.1.1.1:81/eiremote/v1/cameras/1_Test/goauto -d ""`.

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/gomanual	boolean (success/fail) **Note**: Changes current camera mode to Manual. **Note**: Uses HTTP PUT instead of GET. Example: `curl -X PUT http://10.1.1.1:81/eiremote/v1/cameras/1_Test/gomanual -d ""`.
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/state	string - current state of camera
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/mode	string - current trigger mode of camera
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/tags/input	string - list of available input tag names
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/tags/output	string - list of available output tag names
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/tags/{tagname}/inputtype	string - datatype of given input tag
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/tags/{tagname}/outputtype	string - datatype of given output tag
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/tags/{tagname}/readinput	object - current value of input tag
http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/tags/{tagname}/readoutput	object - current value of output tag

Image Web Server

An HTTP web server on port 80 provides HMI images and camera results.

URL	Description
http://localhost/	Provides list of available cameras.
http://localhost/list	Provides list of available cameras.
http://localhost/svg/{CameraName}	Provides latest .svg image of given camera.
http://localhost/png/{CameraName}	Provides latest .png image of given camera.
http://localhost/html/{CameraName}	Provides latest html image of given camera.
http://localhost/{CameraName}	Provides master html page with latest result of given camera.
http://localhost/detail/{CameraName}	Provides detail html page with latest result of given camera.
http://localhost/master/{CameraName}	Provides master html page with latest result of given camera.

VisionLine Interface

An Ethernet socket-based communication connects EtherInspect and a Trumpf laser controller. The Trumpf laser controller initiates communication. EtherInspect listens on port 52000.

Setup VisionLine Jobs

Add the XML settings in the Common.cfg file. You can configure the VisionLine setting per job number per camera. Add XML in the <CommonConfiguration> section in the Common.cfg file.

Example:

```
<VisionLineJobs>
    <VisionLineSettings Enabled="true" IPSTARTNumber=11" CameraName="Cam1"
        JobNumber="1" PFOXOffset10MicronUnits = "24", PFOYOffset10MicronUnits="36",
        PFOZOffset10MicronUnits="48" PFOFocus10MicronUnits="100" HighResolutionRotatoin="false"/>
    <VisionLineSettings Enabled="true" IPSTARTNumber="12" CameraName="Cam1"
        JobNumber="2" PFOXOffset10MicronUnits = "24", PFOYOffset10MicronUnits="36",
        PFOZOffset10MicronUnits="48" PFOFocus10MicronUnits="100" HighResolutionRotatoin="false"/>
    <VisionLineSettings Enabled="true" IPSTARTNumber="13" CameraName="Cam1"
        JobNumber="3" PFOXOffset10MicronUnits = "24", PFOYOffset10MicronUnits="36",
        PFOZOffset10MicronUnits="48" PFOFocus10MicronUnits="100" HighResolutionRotatoin="false"/>
</VisionLineJobs>
```

XML Element	Description
CameraName	The CameraName should match the EtherInspect camera name to configure VisionLine on.
JobNumber	Set the JobNumber for which you want to enable VisionLine.
Enabled	Set `true` to enable VisionLine setting for particular JobNumber and Camera.
PFOXOffset10MicronUnits	Set X Offset coordinate you want to set in PFO. IPSTART uses the coordinate.
PFOYOffset10MicronUnits	Set Y Offset coordinate you want to set in PFO. IPSTART uses the coordinate.
PFOZOffset10MicronUnits	Set Z Offset coordinate you want to set in PFO. IPSTART uses the coordinate.
PFOFocus10MicronUnits	Set Focus Offset you want to set in PFO. IPSTART uses the coordinate.
HighResolutionRotatoin	Set to `true` if you want to set the high-resolution rotation which the IPRDY response uses.

Command	Description	Request	Response

IPSTART	Start a new image processing cycle.	`IPSTART, {ipstart_ number};` {ipstart_number} – set the ip start job number	EtherInspect provides the coordinates to focus the optics (PFO). String formats: `SETPFO {PFOX} {PFOY} {PFOX} {PFOFocus}`- Send PFO offset to Trumpf system • {PFOX} – PFO X Offset 10 Micron Units • {PFOY} – PFO Y Offset 10 Micron Units • {PFOZ} – PFO Z Offset 10 Micron Units • {PFOFocus} – PFO Focus 10 Micron Units `IPBREAK{code}` - When EtherInspect is not ready • {code} – 768 (“RUNNING_ERROR = 0x0300”)
IPSTATEREQ	Inquire state of the image processing.	`IPSTATEREQ;`	EtherInspect returns the current status. String format: `{IPSTATERESP} {code}` • {code} – 0 (“OK = 0x0000”) • {code} – 2 (“IPSTATEREQ_ DURING_ CALCULATION = 0x0002”) • {code} – 768 (“RUNNING_ERROR = 0x0300”) • {code} – 1024 (“FAULTY_ WORKPIECE = 0x0400”)

PFORDY	Programmable focusing optics (PFO) in position. **Note**: After EtherInspect receives the command, EtherInspect triggers the camera and responds with IPRDY (and vars) or IPBREAK and an error code.	`PFORDY;`	EtherInspect returns the values of XVar or TVar tags when image processing is complete. String formats: `IPRDY <XVARS> <XVAR01> 12.6523 </XVAR01> <XVAR02> 2.623 </XVAR02> </XVARS>` – Return XVAR (Dynamic table) data back `IPRDY <TVARS> <TVAR01> 12.6523 </TVAR01> <TVAR02> 2.623 </TVAR02> </TVARS>` – Return TVAR (Static table) data back `IPBREAK {code}` – On error • {code} – 512 (“CALCULATION_ FAULT = 0x0200”) • {code} – 1024 (“FAULTY_ WORKPIECE = 0x0400”) • {code} – 768 (“RUNNING_ERROR = 0x0300”) • {code} – 3 (“PFORDY_ WITHOUT_IPSTART = 0x0003”)