COGNEX

Ether-Inspect 11 – Overview

Contents

System Requirements 5
Supported Devices 5
Installation 5
File Locations 5
Startup Process 5
Periodic Tasks 6
Logic Flow 6
User Interface 8
Overview 8
SELECT CAMERA AREA 9
Camera Replacement Display 9
SETUP / CONFIGURATION AREA 9
User Settings Display 10
Versions Display 13
Cycle Counts Display 14
INFORMATION LOG AREA 14
Log Viewer Display 14
COMMUNICATIONS AREA 14
Tag States Display 15
Setup Tags Display 15
Tag Help Display 16
Edit PLCs Display 16
IMAGE SAVE OPTIONS AREA 17
Image Playback Options Display 17
FILE OPTIONS AREA 18
INSPECTION SETTINGS AREA 18
Category Visibility Display 19
Changing the Number of Tools 19
Edit ToolBlock Display 20
Calibrate Display 20
Image Filtering Display 21
Image Conversion Display 21
Fixturing Display – Creates a 2D fixtured image 21
Pattern Tool Display 22
Histogram Tool Display 22

Blob Tool Display 23
Measure Tool Display (Calipers) 23
Find Line Tool Display 24
Find Circle Tool Display 25
Codes Display (1D and 2D codes) 25
Color Segmenter Display 27
2D Geometry 28
Cross Section 29
2 Point Fixturing 29
Image Masking 30
Beads 30
VIDI Pre Processing 31
VIDI 31
VIDI Post Processing 32
Custom Toolblock 32
SCRIPTING FEATURES 41
Scripting Tool Display 41
Script Editor Display 41
CAMERA SETTINGS AREA 51
Image Settings Display 52
TRIGGER OPTIONS AREA 53
RESULTS AREA 53
IMAGE AREA 54
Ether-Inspect remote interface 55
Configuration 55
Commands 55
Cameras – This will return list of available cameras from application instance 55
Trigger – Camera can be triggered using this command 55
ShowMultiCam – Camera view can be changed to multi cam using this command 55
ShowSingleCam - Camera view can be changed to single cam using this command 55
AppName – Get the application name 56
AppVersion – Get the application version 56
InstanceNumber – Get the application instance number 56
CameraType – Get camera type by camera name 56
LoadJob – Load job by number 56

JobName – Get current job name as string 57
JobNumber – Get current job number 57
GoLive – Put camera into live mode 57
GoAuto – Put camera into auto mode 57
GoManual – Put camera into manual mode 57
State – Get current state or camera 58
Mode – Get current trigger mode 58
InputTags – This will return list of input tags configured for camera 58
OutputTags – This will return list of output tags configured for camera 58
InputTagType – Get data type of input tag 58
OutputTagType – Get data type of output tag 59
ReadInputTag – Read value of input tag 59
ReadOutputTag – Read value of output tag 59
WriteTag – Write value to output tag 59
Exposure – Set or get camera exposure value 60
Contrast – Set or get camera contrast value 60
Brightness – Set or get camera brightness value 60
MaxExposure – Get maximum exposure count 60
EnableMultiExposure – Enable or disable multi exposure 61
Ether-Inspect web remote interface 62
Configuration 62
URLs 62
Ether-Inspect web server 66
Configuration 66
URLs 66
VisionLine remote interface 67
Configuration 67
Commands from Laser 68
IP Start – Start a new image processing cycle. 68
IP State Req – Inquire state of the image processing. 68
PFO Ready – Programmable focusing optics (PFO) in position. Once this command is received by Ether-Inspect, ... 69

System Requirements

- Windows 10 64 Bit
- VisionPro 9.5 SR2
- ViDi Suite 4.1
- .NET Framework 4.7.2
- Cognex License Dongle

Supported Devices

- Cognex CIC GigE Cameras
- Cognex 3D-A5000 Area Scan Sensors
- Cognex DS1000 Laser Line Sensors
- GigE cameras that are Gen-I-Cam compliant
- Up to 12 sensors are supported per PC (with standard license dongle)
- Up to 32 virtual cameras per Ether-Inspect instance
- PLCs such as Allen-Bradley ControlLogix, Siemens S7, Bechoff TwinCAT, and Modbus compatibles

Installation

- Run EtherInspect.Setup.11.X.Y-Z.exe
- Run Ether-Inspect Configuration tool from desktop to configure network interfaces and cameras

File Locations

The CameraSettings folder is located at C:\ProgramData\Cognex\Ether-Inspect\CameraSettings
The EILogs folder is located at C:\ProgramData\Cognex\Ether-Inspect\EILogs

Startup Process

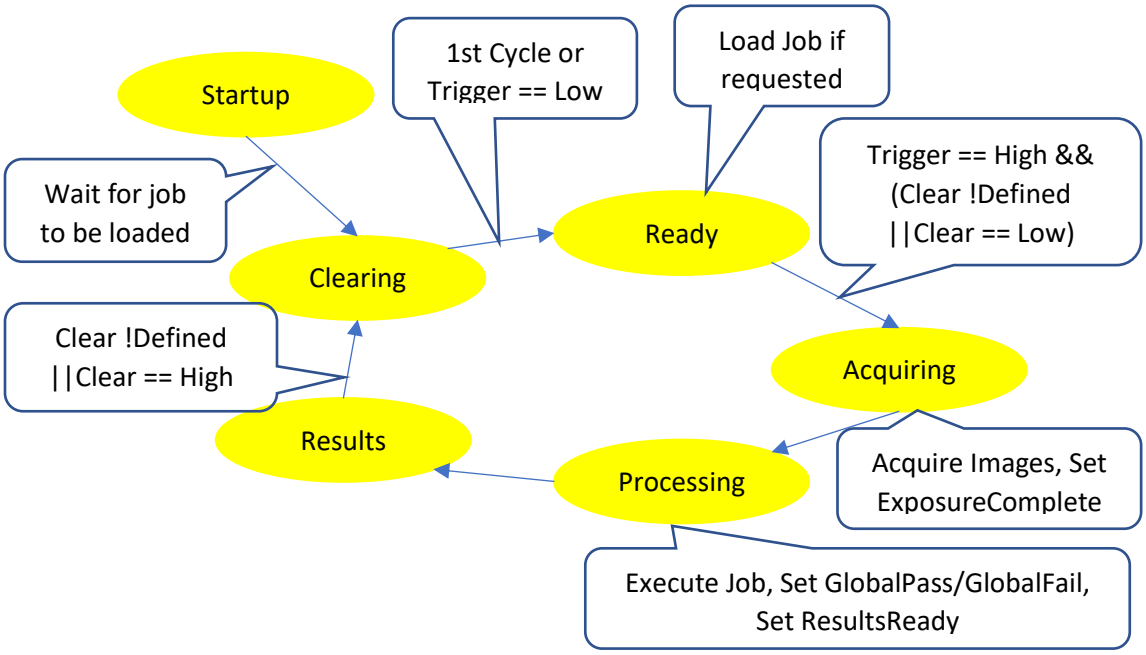
- Check License bits (about 5 seconds to load Wibu libraries)
    - If dongle does not exist, or doesn't have VisionPro, show message box and exit
- Determine instance number
    - If specified on command line, confirm not already used, then claim
    - If not specified, check each instance number (up to 6) and claim first unused
- Load Common.cfg (create if not existing)
- Check IP Addresses (only if Instance == 1)
- If change was required, launch external utility as Administrator to set IPs
- Start WebServer (only if Instance == 1)
- Start additional instances (if started from registry and instance == 1 and desired instances > 1)
- Establish connections to PLCs (defined in SetupX.cfg)
- Search for cameras
- Load virtual cameras if needed
- Load startup jobs for all cameras
- Show Loading Complete

Periodic Tasks

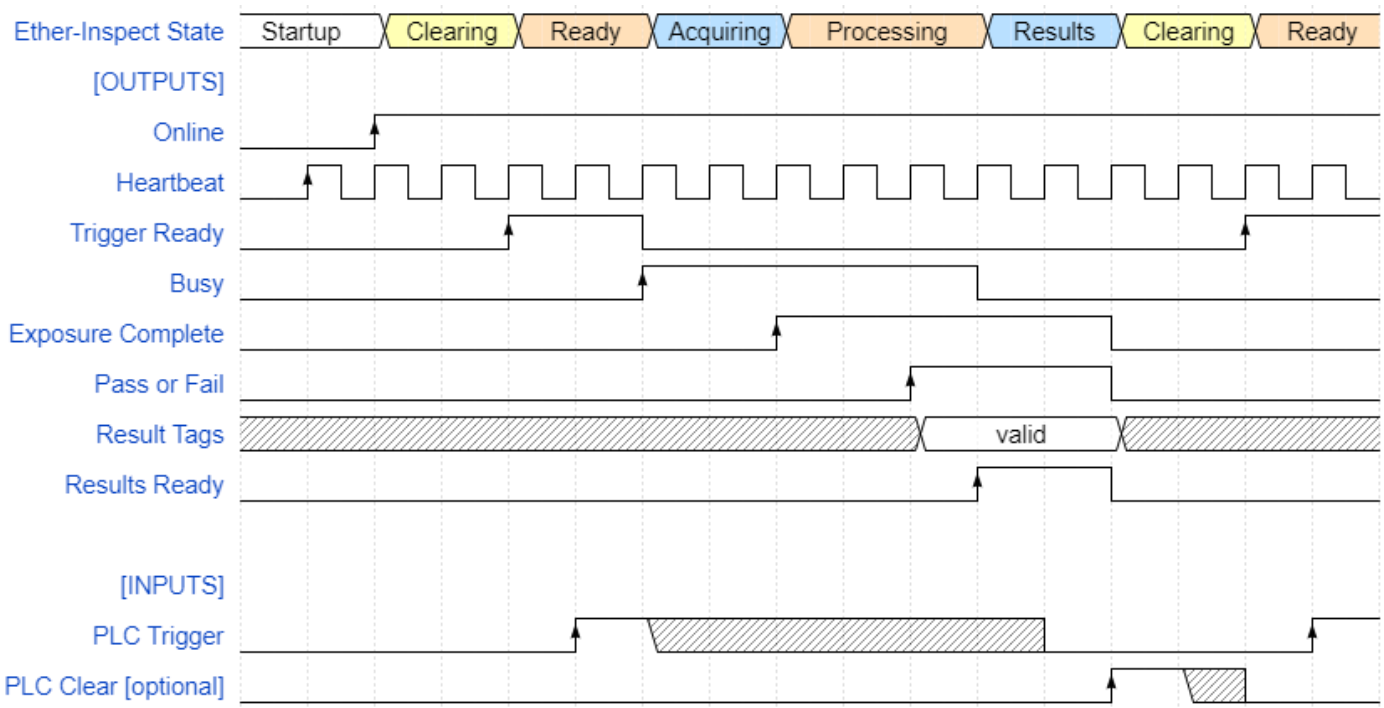
- Every minute
  - If running on development dongle, close application before dongle times out
  - If in engineer mode and idle, update idle time
  - If in engineer mode and idle time is 1 minute less than setting, show warning
  - If in engineer mode and idle time expired, switch to operator mode.
- Every hour
  - Run disk cleanup routine
    - Delete oldest day of results in a job folder
    - Repeat until disk space remaining is above threshold (default 20%, min 10%)

Logic Flow

Each camera in Ether-Inspect 11 is always in one of 6 states (only 5 are used in auto mode) The following diagram shows the transitions between states.

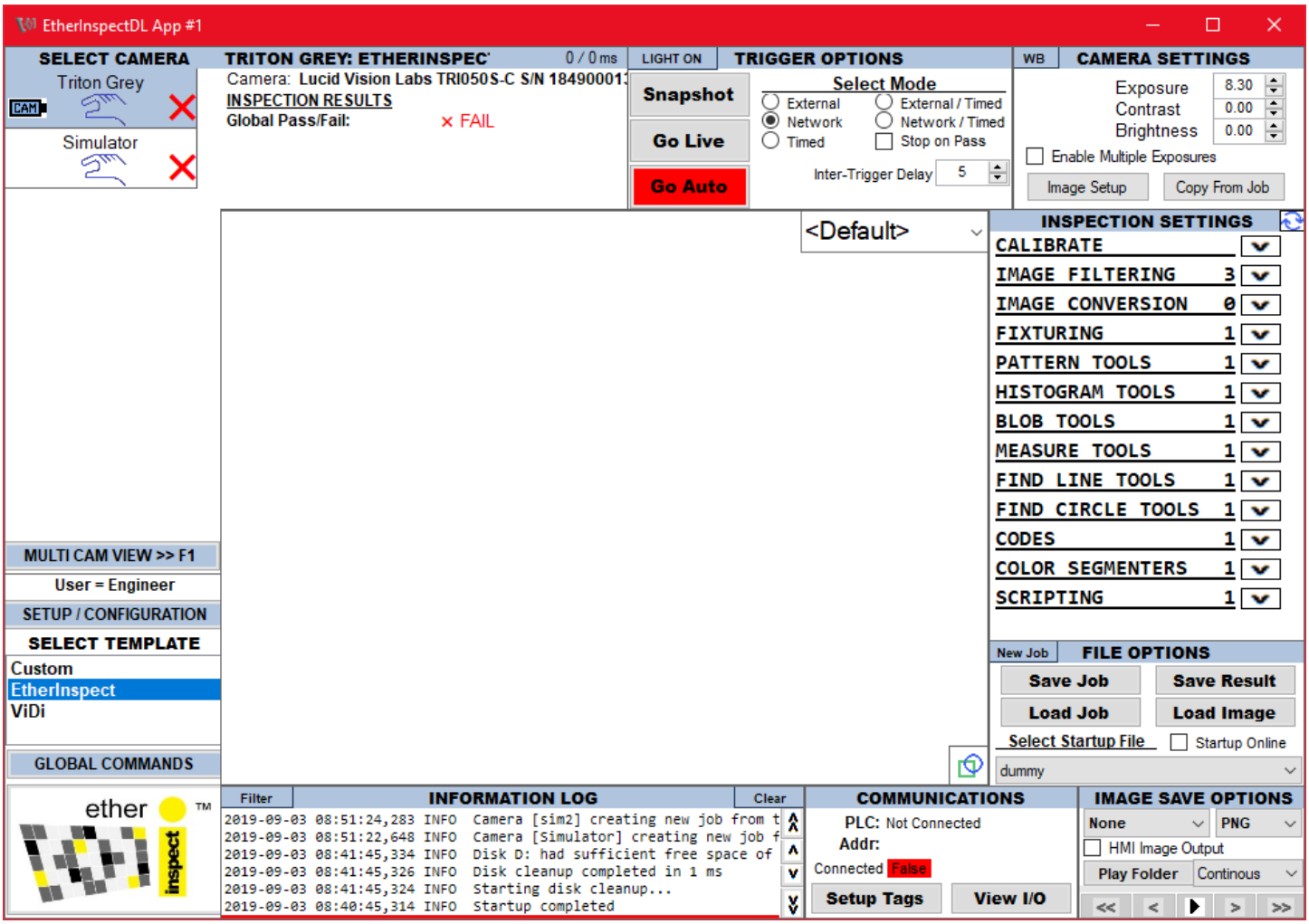


This timing chart shows a typical cycle with Ether-Inspect 11. Note that PLCClear is only valid when ResultsReady is on, and should be dropped when ResultsReady is low. PLCTrigger is only valid when TriggerReady is on and should be dropped when TriggerReady is low.

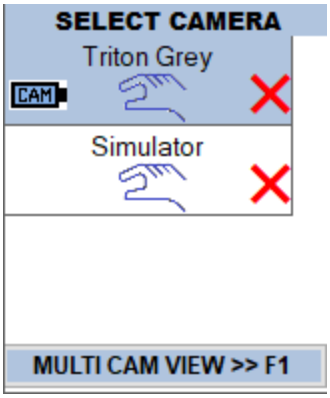


User Interface

Overview

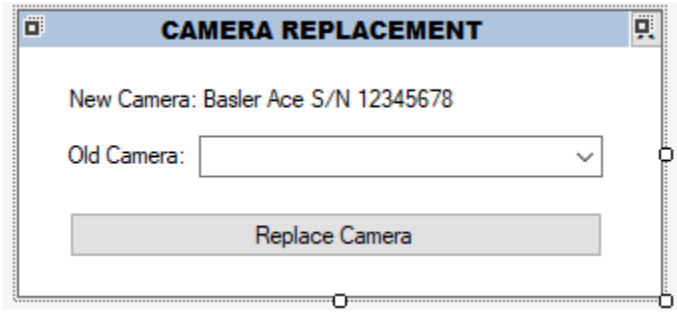


SELECT CAMERA AREA – Scrollable panel to let user choose a camera.

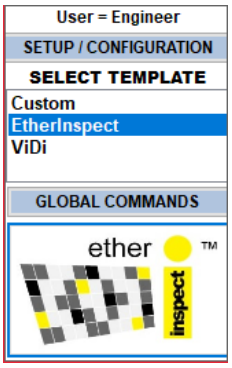


- Clicking a camera will make it display to the right.
- Right-clicking on a camera will allow creating or deleting a virtual camera or simulator as well as renaming a camera.
- Clicking "MULTI CAM VIEW" will display a page of cameras instead of one camera. Click again to change to the next page if configured.

Camera Replacement Display – Allows swapping a camera with one of a different serial number, accessed from the right-click menu on a camera button.



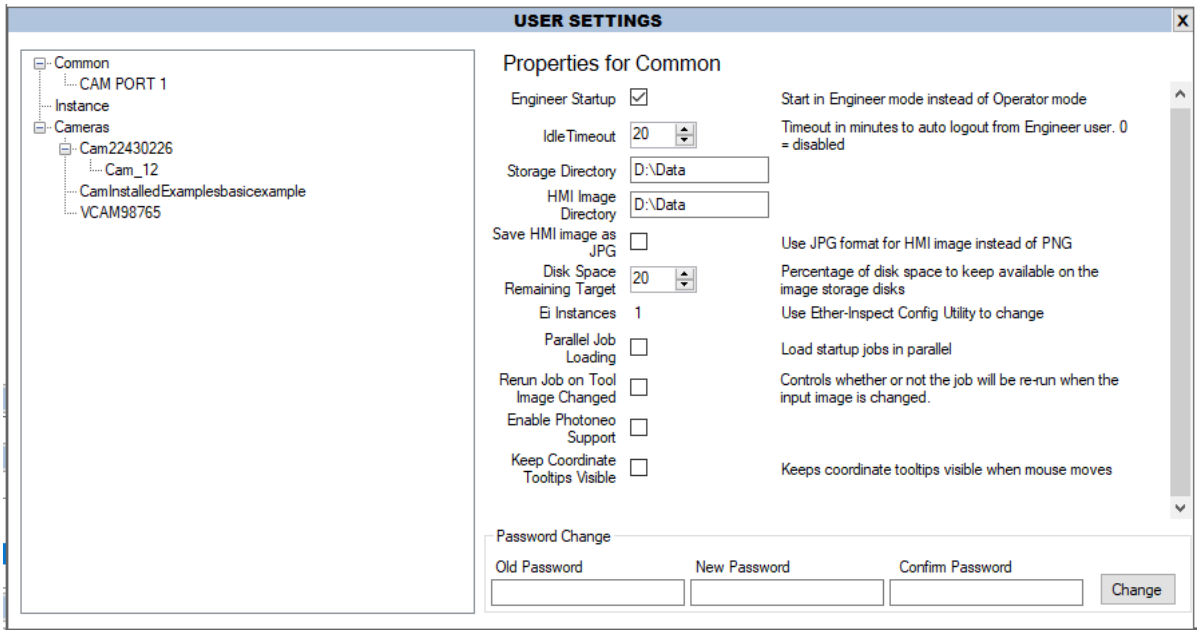
SETUP / CONFIGURATION AREA– Block in lower left shows user level, allows selecting template and accessing settings and global commands.



- Clicking a template name in the list under "SELECT TEMPLATE" will change the current camera to that template
- Clicking "SETUP / CONFIGURATION" will open the User Settings Display
- Clicking the Ether-Inspect logo will display version information
- Clicking "GLOBAL COMMANDS" will display the global command menu

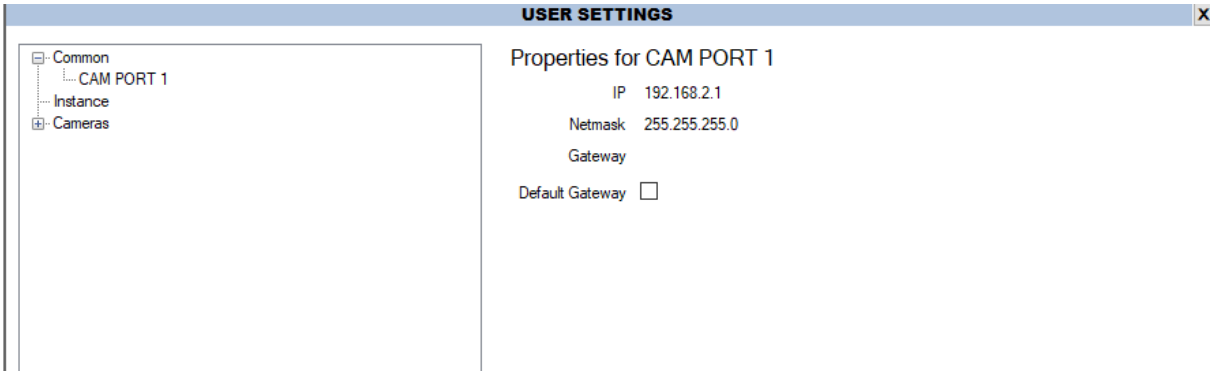
User Settings Display - Allows user to edit application settings, including camera and job specific settings

Common settings



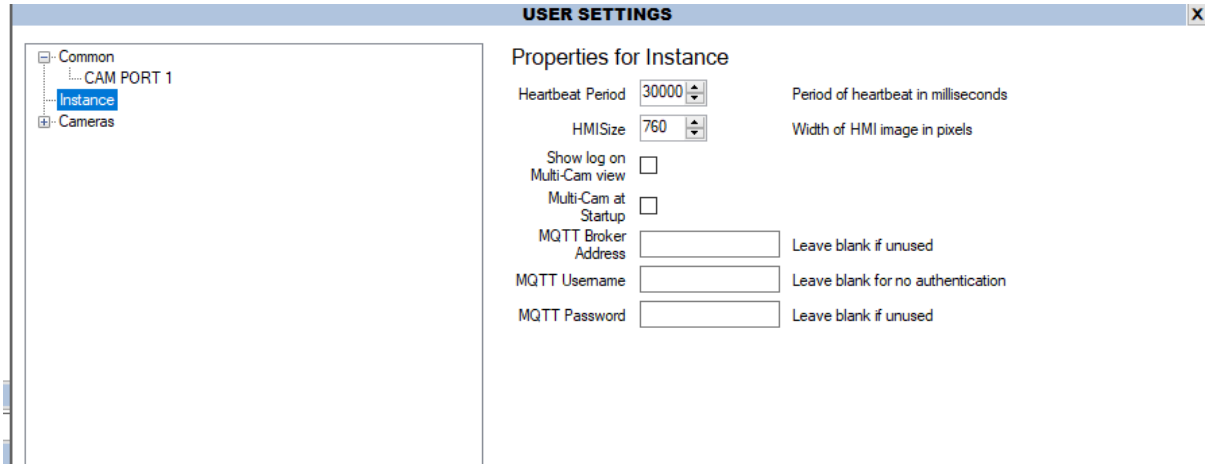
- Engineer Startup - By default EtherInspect will start with Engineer mode but you can change the mode to start with Operator mode.
- Idle timeout – Engineer mode will auto logout after give timeout. If set to 0, the timeout will be disabled.
- Storage Directory – Configure the path where data will be stored. The default path is D:\Data
- HMI Image Directory – Configure the path where HMI images will be stored. The default path is D:\Data
- 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.
- EI Instances – It will be read-only, use Ether-Inspect config utility to change.
- Parallel Job Loading – Use this option to load startup job in parallel.
- Rerun Job on Tool image changed – Controls whether the job will be re-run when the input image is changed.
- Enable Photoneo Support – Photoneo support can be enabled.
- Keep Coordinate Tooltip Visible – On double click on Camera display, the coordinate will be shown on the screen. On mouse move, the coordinate will be hidden. This behavior can be changed with this option, if checked, the coordinate tooltip will stay on screen until click outside of display.
- 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.

Camera Port settings



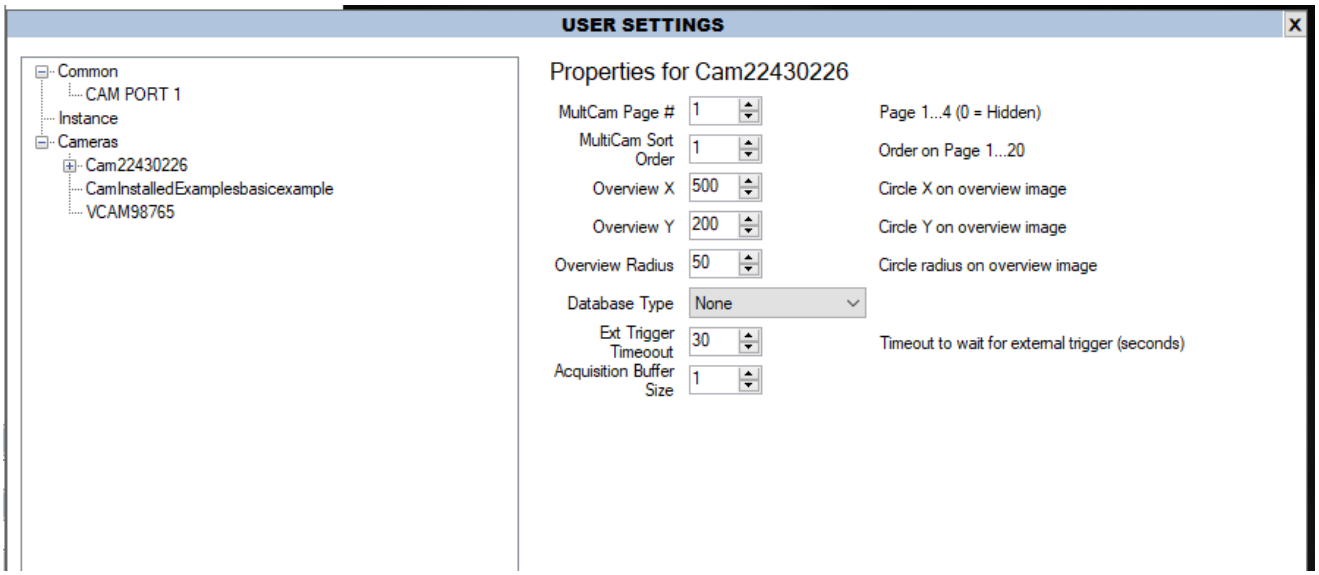
- IP / Netmask / Gateway - Read-only information about current IP, Netmask and Gateway will be shown. It can be configured using Ether-Inspect Config utility.
- Default Gateway – It allows to change if default gateway should be used.

Instance level settings



- Heartbeat Period - Period of heartbeat in milliseconds.
- HMI Size – Configure the width of HMI image in pixels, height will be automatically calculated based on the image's aspect ratio.
- Show log on Multi-Cam view – The information log will be shown when Multi-Cam view is on.
- Multi-cam at Startup – If turn on, multi-cam view will be set on application startup.
- MQTT Broker address – IP address of MQTT broker, leave it 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.

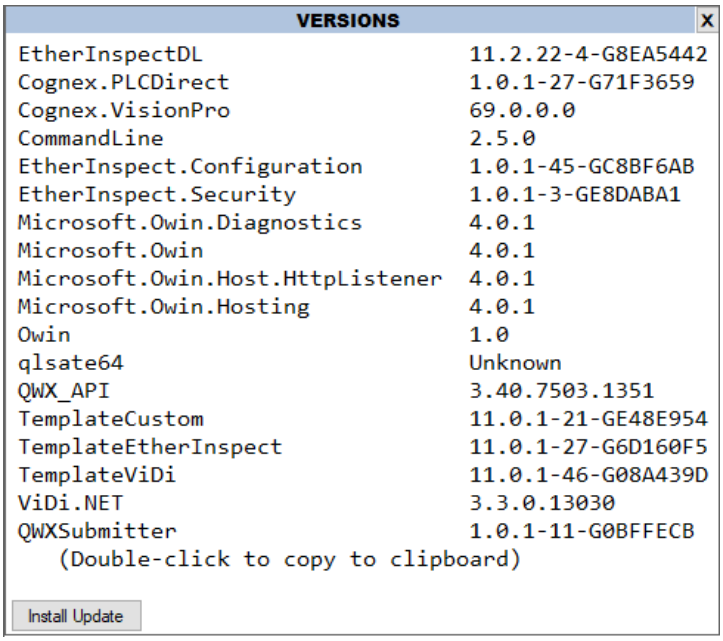
Camera setting



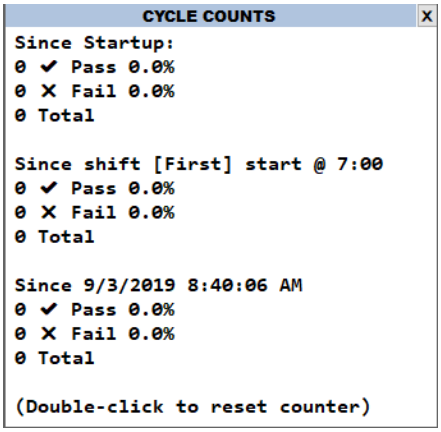
- MultiCam Page # - Page number on which this camera will be shown when MultiCam view is on. The pages can be between 1 to 4, set 0 will hide this camera from MultiCam view.
    - Keep clicking on "MULTICAM VIEW >> F1" will allow to navigate through multiple pages.

- MultiCam Sort order – Configure the order of this camera on MultiCam page, Order number can be between 1 to 20.
- Overview X – This configuration is being used in EtherInspect viewer application, a fault animated circle will be shown at this coordinate on Overview image.
- Overview Y – This configuration is being used in EtherInspect viewer application, a fault animated circle will be shown at this coordinate on Overview image.
- Overview Radius – This configuration is being used in EtherInspect viewer application, a fault animated circle will be animated up to configured radius.
- Database Type – Configure which database to send inspection data. Based on selected type, more configuration will appear.
- Ext Trigger Timeout – Timeout to wait for external trigger in seconds.
- Acquisition Buffer Size – Values greater than 1 allow acquiring a new image while the first is processing.

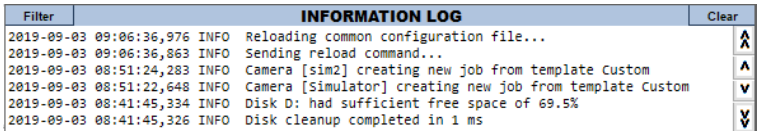
Versions Display – Shows versions of EtherInspect11 and DLLs that are used (including templates)



Cycle Counts Display – Shows counts since startup, since shift start and user-resettable

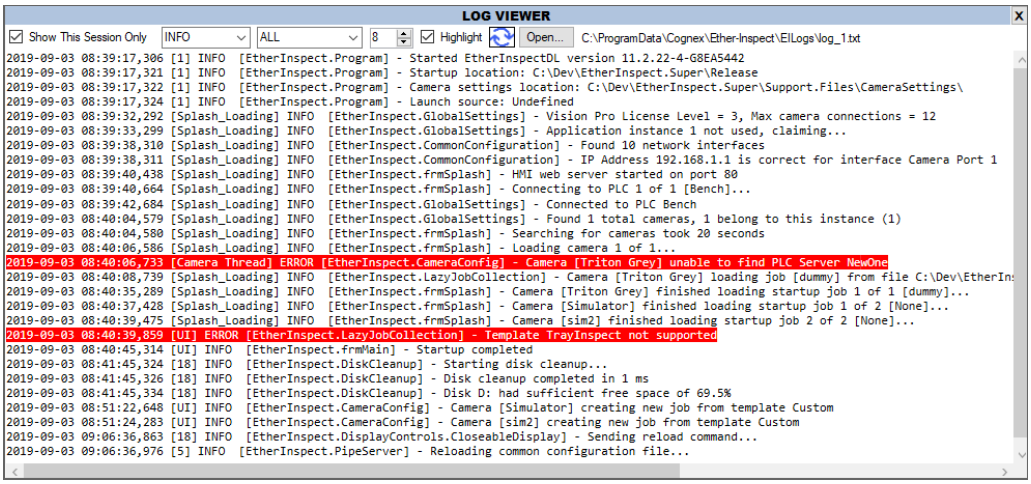


INFORMATION LOG AREA – Shows log entries, holds 1000 entries in memory, all are logged to disk



- Clicking “Filter” will flush the logs to disk and show the log viewer display
- Clicking “Clear” will clear the visible display, log files are unaffected

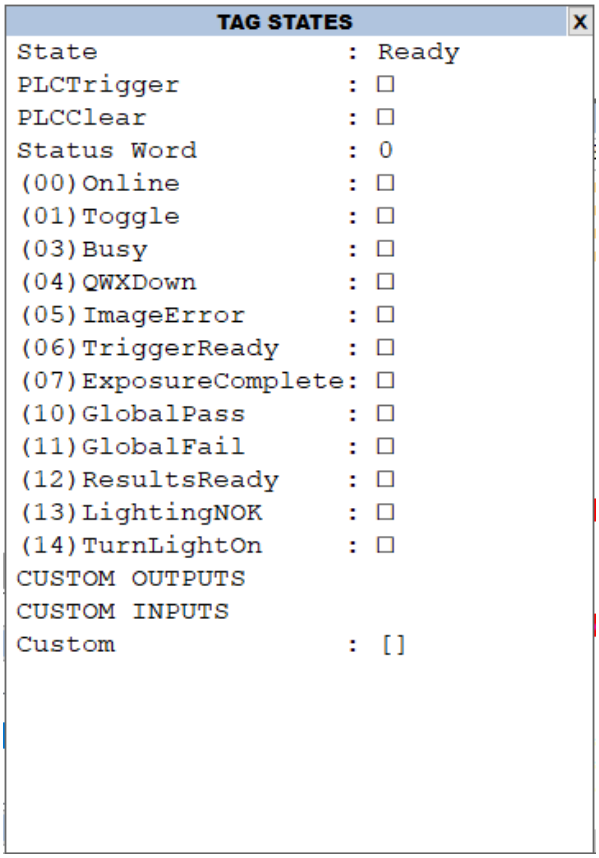
Log Viewer Display – Allows viewing and filtering log files



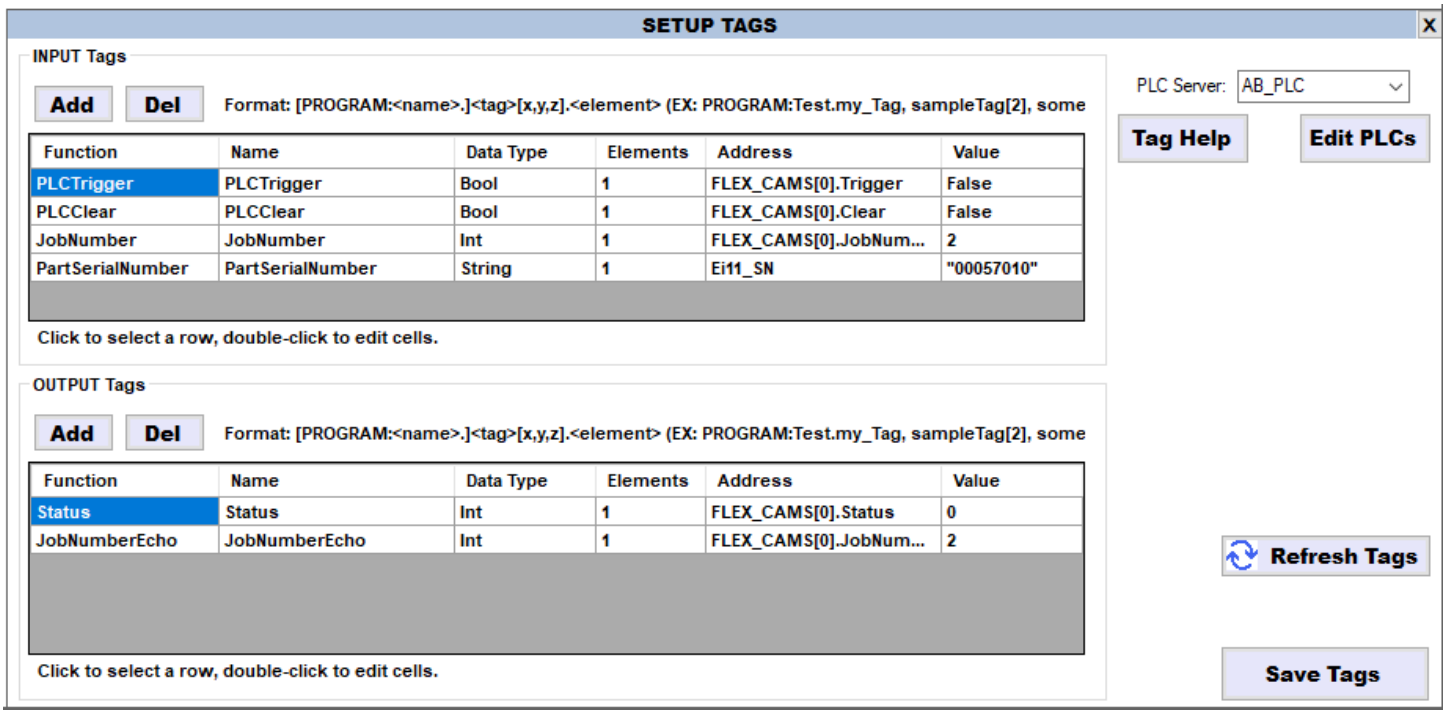
COMMUNICATIONS AREA – Displays PLC connection state, allows viewing IO and editing Tags

- Clicking “View I/O” will show Tag States Display
- Clicking “Setup Tags” will show Setup Tags Display

Tag States Display – Shows current values of PLC Tags and current state of camera thread, 100 ms refresh rate

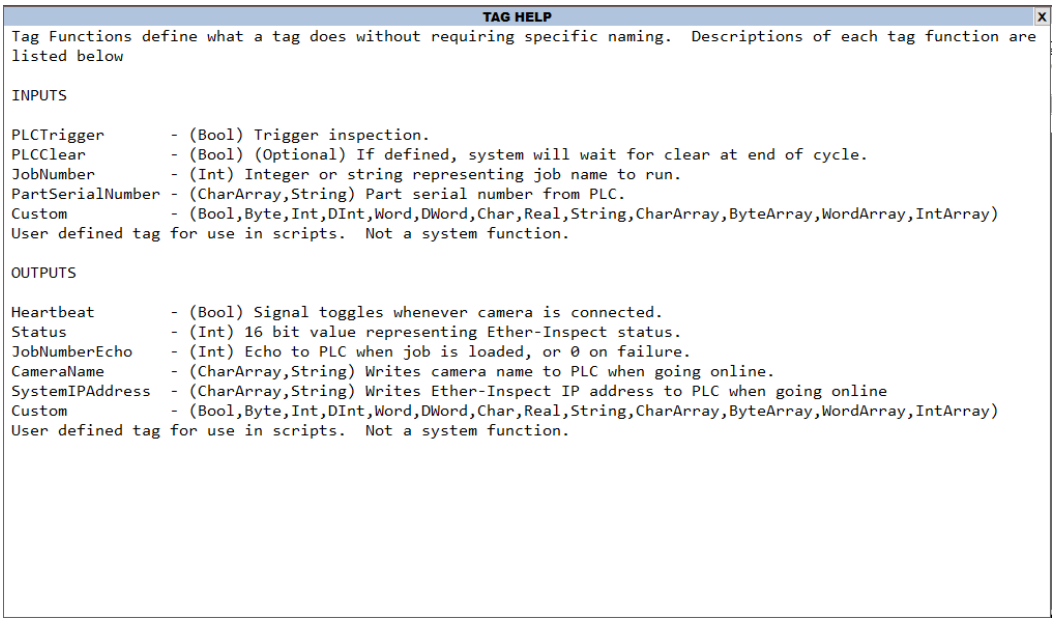


Setup Tags Display – Accessible from CommunicationDisplay, uses 2 TagEditor to edit Tags.



- Clicking “Tag Help” will show Tag Help Display, with information on tag functions and valid data types
- Clicking “Edit PLCs” will show Edit PLCs Display, to add or delete PLC assignments.

Tag Help Display – Shows information on tag functions and valid data types



Edit PLCs Display – Allows editing PLC connections, only one PLC vendor may be used at one time

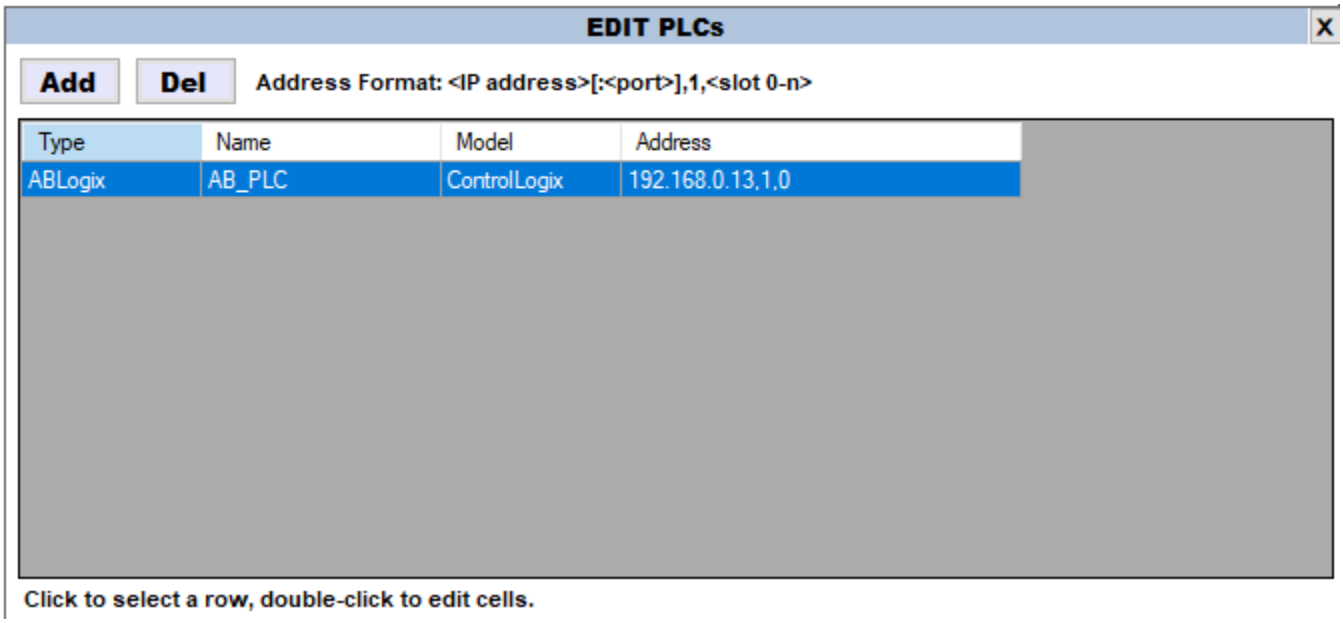
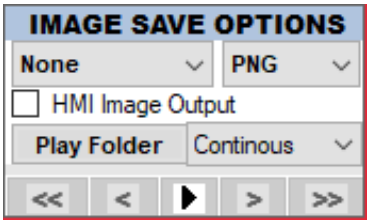
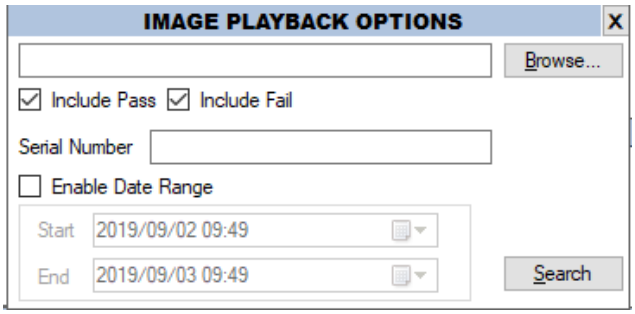


IMAGE SAVE OPTIONS AREA – Has settings for image save types, also allows playing back images

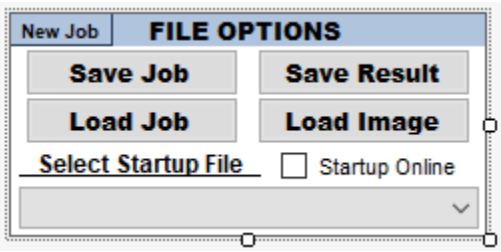


- Clicking the drop-down in the upper left will change what images are stored (None, Pass, Fail or Pass & Fail)
- Clicking the drop-down in the upper right will change the image type that is stored (BMP, PNG, JPG, or CDB). If a 3D sensor is used, CDB format will be used automatically. JPG format is not recommended as stored images cannot be re-introduced into Ether-Inspect.
- Clicking HMI Image Output will enable saving a thumbnail image with graphics
- Clicking Play Folder will open the Image Playback Options Display to create the image list
- Clicking the drop-down in the lower right will change when the image playback stops (Continuous, RunToEnd, StopOnPass, or StopOnFail)
- Clicking "<<" will play the first image in the list
- Clicking "<" will play the previous image in the list
- Clicking Play will play the images in the list, stopping on the condition selected
- Clicking ">" will play the next image in the list
- Clicking ">>" will play the last image in the list

Image Playback Options Display – Sets criteria to search for images to be played



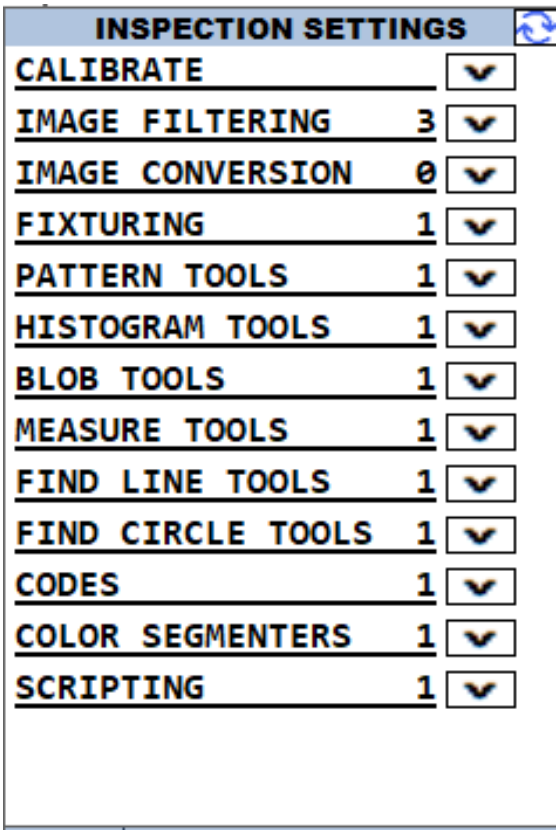
FILE OPTIONS AREA – Allows loading/saving jobs and images



- Clicking “New Job” will create a new job from the current template
- Clicking “Save Job” will prompt for the name of the job to save
- Clcking “Load Job” will show the file open dialog to load a job file
- Clicking “Save Result” will prompt for the folder to save the current image and result XML file
- Clicking “Load Image” will show the file open dialog to play a single image file
- Clicking the drop-down at the bottom will select a startup job
- Clicking “Startup Online” will make the camera go online at startup after the job is loaded

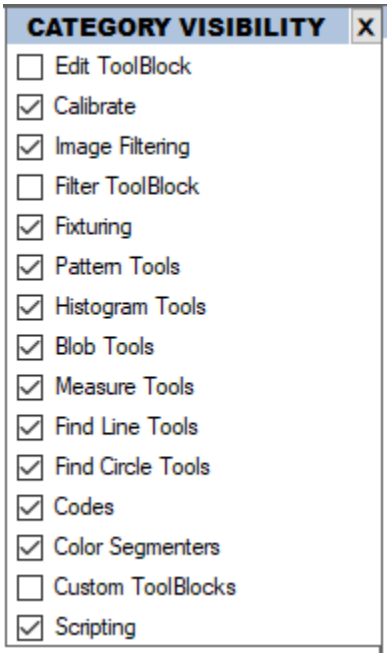
NOTE: Images and Jobs can be dragged onto the Ether-Inspect window to load them

INSPECTION SETTINGS AREA – Allows the user to change settings on various tools in the current template

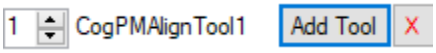


- Double-clicking “INSPECTION SETTINGS” will open the Category Visiblility Display to show or hide categories
- Clicking the arrows in the upper-right will re-run the inspection using the same image
- Clicking the down arrow next to any category will show the tool settings for the first tool in each category

Category Visibility Display – Has checkboxes for each category listed in InspectionSettingsDisplay to hide them

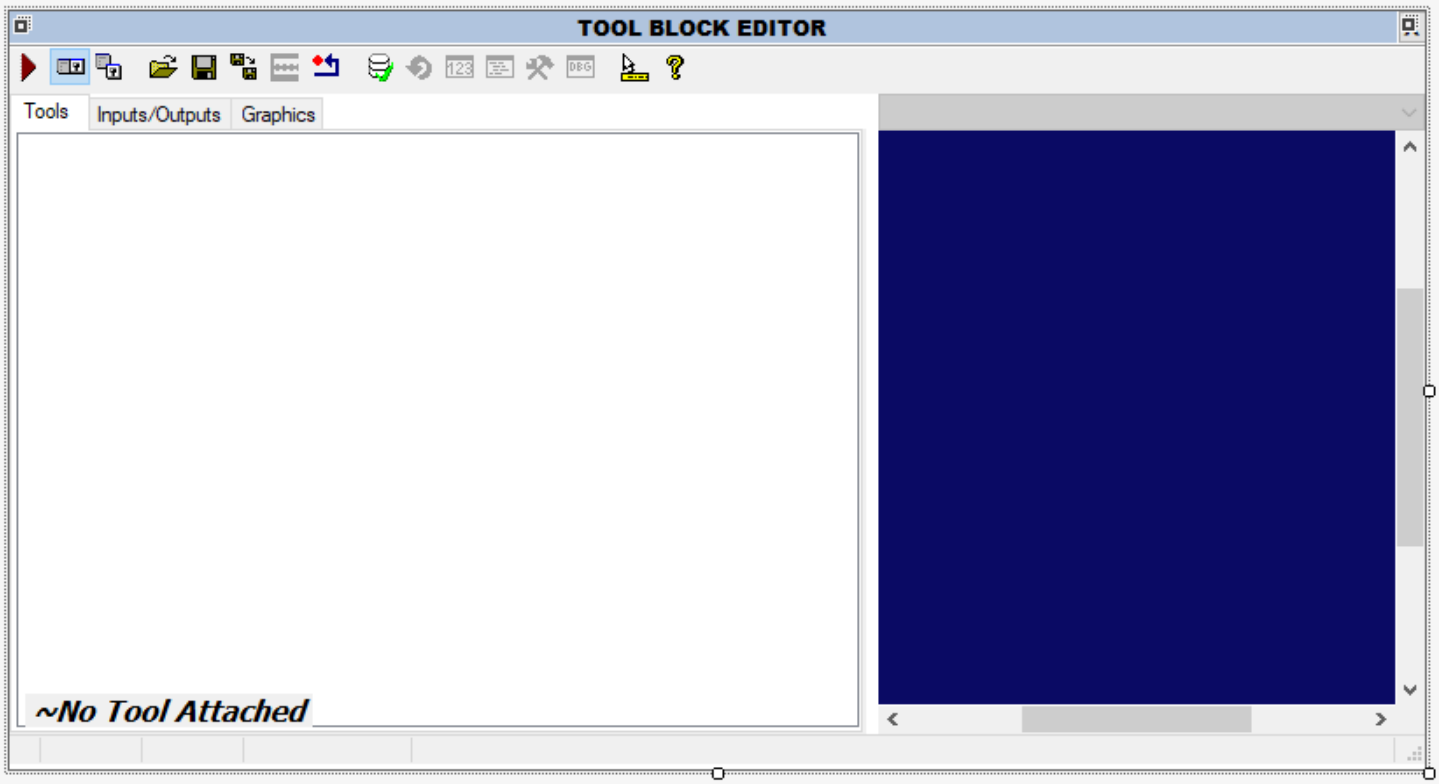


Changing the Number of Tools

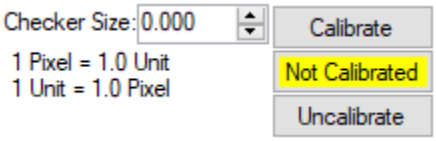


- Clicking “Add Tool” will create a new tool of the same type
- Clicking “X” will delete the current tool
- Double-clicking on the tool name (ex. CogPMAlignTool1) will open the VisionPro tool settings for that tool
- Right-clicking on the tool name will allow renaming the tool
- Changing the number in the up/down will change which tool is displayed

Edit ToolBlock Display – **Allows editing CogToolBlocks**

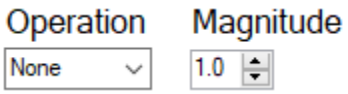


Calibrate Display – **Checkerboard calibration**



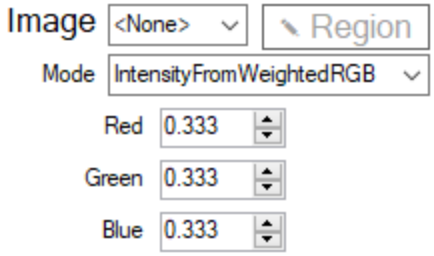
- Set the “Checker Size” to the size of the checkerboard squares (either metric or imperial units)
- Click the “Calibrate” button to calibrate
- Click the “Uncalibrate” button to forget the calibration

Image Filtering Display



- Select the operation to perform (None, Dialate, Erode, Open, Close, Multiply, or Custom)
- Change the “Magnitude” to change the number of times the filter is run (or the constant in Multiply mode)
- If “Custom” is chosen, the Image Filtering Tool can be edited by double-clicking the tool name.

Image Conversion Display

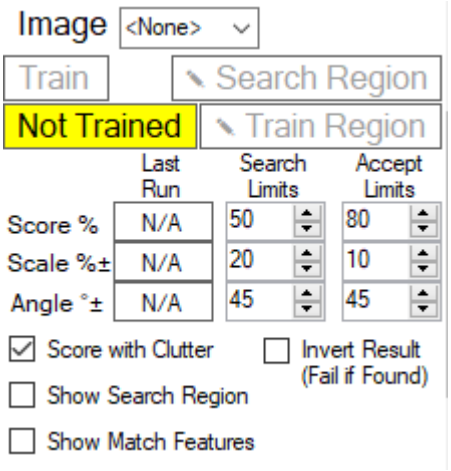


- Click the “Image” drop-down to select the color input image to convert to grayscale
- Click the “Region” button to change the operating region (default is entire image)
- Change the “Red”, “Green” or “Blue” weights to alter how the grayscale image is generated

Fixturing Display – Creates a 2D fixtured image

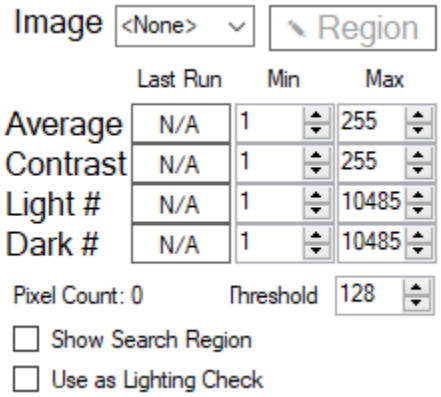
- Quickstart:
  - Click the “Image” drop-down to select the input image to fixture
  - Click the “X” drop-down to select the tool to use for the X coordinate
  - Click the “Y” drop-down to select the tool to use for the Y coordinate
  - Click the “Rot” drop-down to select the tool to use for the Rotation value

Pattern Tool Display



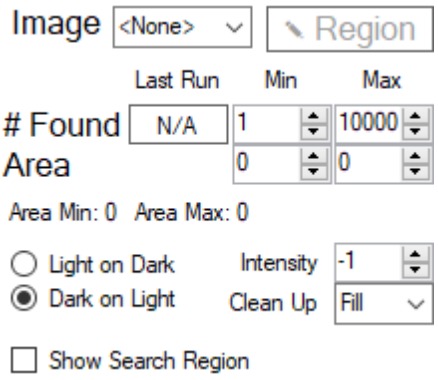
- Quickstart:
  - Click the “Image” drop-down to select the input image
  - Click “Train Region” and adjust the region shape and size
  - Click “Train” to train the pattern
- Score, Scale and Angle are shown for the last run
- “Search Limits” are passed to the VisionPro tool
- “Accept Limits” control the Pass/Fail result in Ether-Inspect
- “Search Region” can be set to smaller than the entire image
- “Score with Clutter” changes scorring setting in VisionPro tool
- “Invert Result” flips Pass/Fail result in Ether-Inspect
- “Show Search Region” will show the bounding box for the search region on the image
- “Show Match Features” will draw the features on the image
- The coordinate axes for the found pattern are always shown on the image

Histogram Tool Display



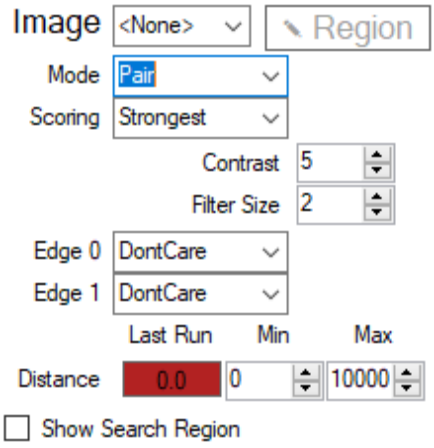
- Quickstart:
  - Click the “Image” drop-down to select the input image
  - Click “Region” and adjust the region shape and size
  - Adjust “Threshold” and “Min” and “Max” values
- “Show Search Region” will show the bounding box for the search region on the image
- “Use as Lighting Check” will turn on LightingNOK bit in Ether-Inspect status word if tool fails

Blob Tool Display



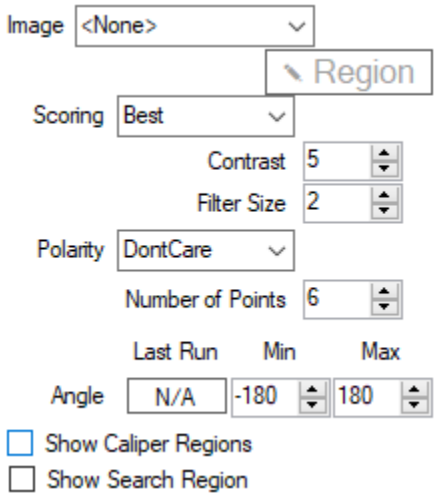
- Quickstart:
  - Click the “Image” drop-down to select the input image
  - Click “Region” and adjust the region shape and size
  - Set “Light on Dark” or “Dark on Light”
  - Set “Clean Up” mode (Fill, Prune, None)
  - Adjust “Intensity” and “Min” and “Max” values
- “Show Search Region” will show the bounding box for the search region on the image

Measure Tool Display (Calipers)



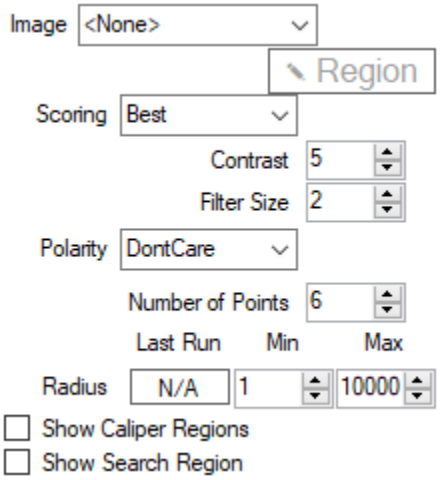
- Quickstart:
  - Click the “Image” drop-down to select the input image
  - Click “Region” and adjust the region shape and size
  - Set “Mode” (Pair or SingleEdge)
  - Set “Scoring”, Pairs (Strongest, Narrowest, Widest, WidestContinous) , Single edges (Best, First, Last, All)
  - Set edge polarity (DontCare, DarkToLight, LightToDark)
  - Adjust “Contrast”, “Filter Size”, and “Min” and “Max” values
- “Show Search Region” will show the bounding box for the search region on the image

Find Line Tool Display



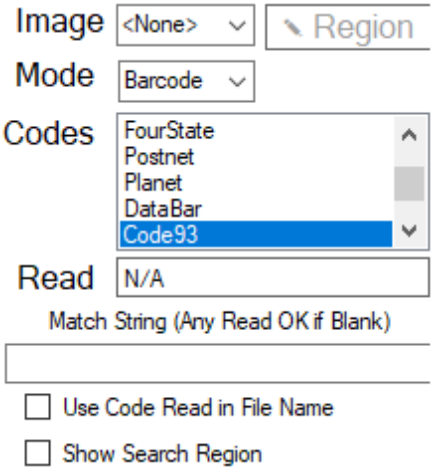
- Quickstart:
  - Click the “Image” drop-down to select the input image
  - Click “Region” and adjust the region shape and size
  - Set “Scoring” (Best, First, Last, All)
  - Set “Polarity” (DontCare, DarkToLight, LightToDark)
  - Set “Number of Points”
  - Adjust “Contrast”, “Filter Size”, and “Min” and “Max” values
- “Show Search Region” will show the bounding box for the search region on the image
- “Show Caliper Regions” will show the bounding box of caliper on the image

Find Circle Tool Display



- Quickstart:
  - Click the “Image” drop-down to select the input image
  - Click “Region” and adjust the region shape and size
  - Set “Scoring” (Best, First, Last, All)
  - Set “Polarity” (DontCare, DarkToLight, LightToDark)
  - Set “Number of Points”
  - Adjust “Contrast”, “Filter Size”, and “Min” and “Max” values
- “Show Search Region” will show the bounding box for the search region on the image
- “Show Caliper Regions” will show the bounding box of caliper on the image

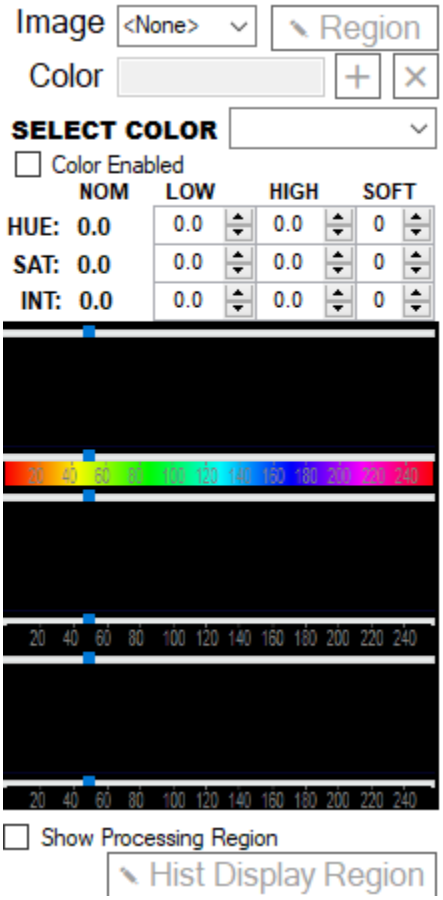
Codes Display (1D and 2D codes)



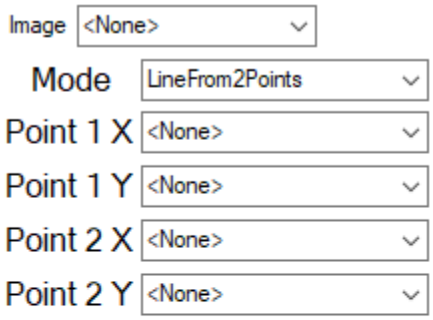
- Quickstart:
  - Click the “Image” drop-down to select the input image
  - Click “Region” and adjust the region shape and size
  - Set “Mode” (Barcode, Matrix)
  - Select “Codes”
  - Set “Match String” if desired
- “Show Search Region” will show the bounding box for the search region on the image

Color Segmenter Display

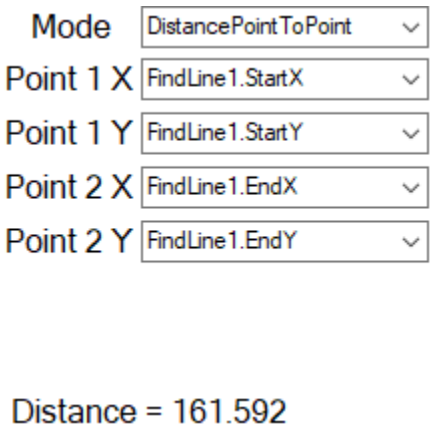
Creates a binary image from a color image, based on selected HSI values



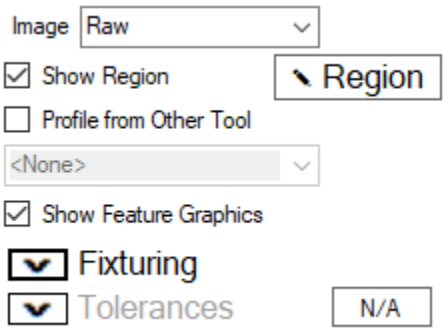
- Quickstart:
  - Click the “Image” drop-down to select the input image
  - Click “Region” and adjust the region shape, if desired
  - Type in a “Color Name”
  - Click “+” to add the color
  - Select the newly created color name
  - Adjust the Hue, Saturation, and Intensity values, using sliders or up/down boxes
- “Show Processing Region” will show the bounding box for the region processed on the image
- “Hist Display Region” allows choosing an area of the image to show the histogram for



- Quickstart:
  - Click the “Image” drop-down to select the input image
  - Click “Mode” and select the mode you want to use for geometry
    - “CircleFromThreePoints”
    - “DistanceLineToPoint”
    - “DistancePointToPoint”
    - “IntersectionOf2Lines”
    - “LineFrom2Points”
  - Based on mode, it will allow to select coordinate X Y or other settings.
- Results will be shown in the bottom section.

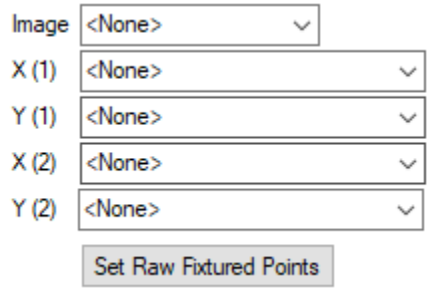


Cross Section



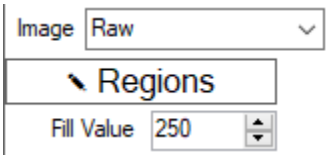
- Quickstart:

2 Point Fixturing



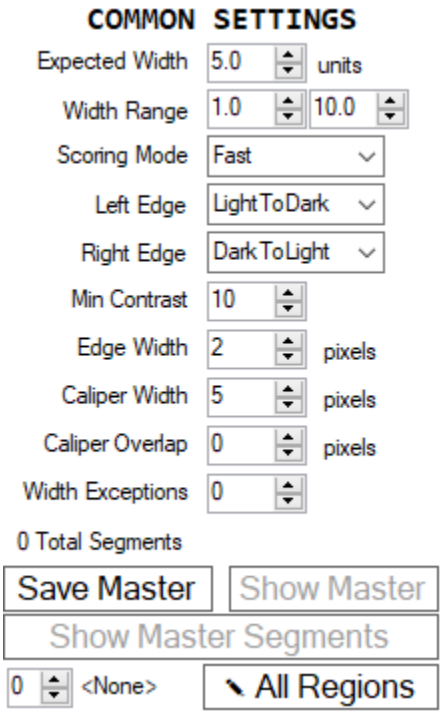
- Quickstart:
  - Click the “Image” drop-down to select the input image
  - Click the “X (1)” drop-down to select the tool to use for the X coordinate of the first point
  - Click the “Y (1)” drop-down to select the tool to use for the Y coordinate of the first point
  - Click the “X (2)” drop-down to select the tool to use for the X coordinate of the second point
  - Click the “Y (3)” drop-down to select the tool to use for the Y coordinate of the second point
- “Set Raw Fixtured Points” – It will set the Raw Fixtured point same as unfixtured point

Image Masking



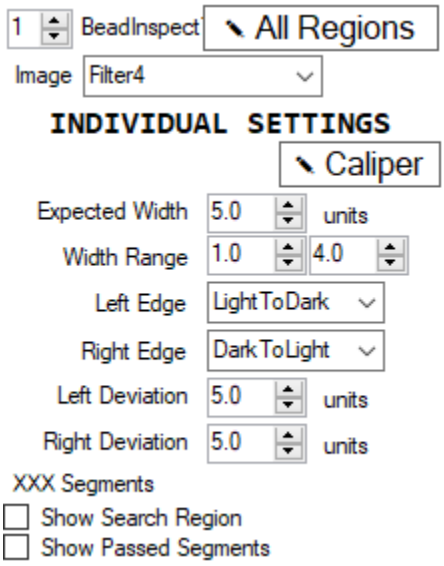
- Quickstart:
  - Click the “Image” drop-down to select the input image
  - Click “Region” and adjust the region shape and size
  - Set the fill value which will be used to fill the masked region

Beads

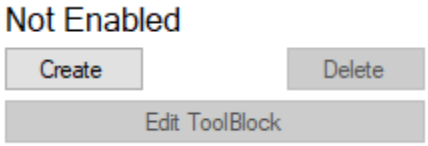


- Quickstart:
  Common setting that will be applied to all the regions. Individual setting instructions are given below
  - Set “Expected Width”
  - Adjust width range “Min” and “Max” values
  - Set “Scoring” (Fast, Enhanced, Advanced)
  - Set “Left Edge” (DarkToLight, LightToDark, DontCare)
  - Set “Right Edge” (DarkToLight, LightToDark, DontCare)
  - Set “Min Contrast”
  - Set “Edge Width” in pixels
  - Set “Caliper Width” in pixels
  - Set “Caliper Overlap” in pixels
  - Set “Width Exceptions”
- “Save Master” – Save master bead
- “Show Master” – Show master bead
- “All Regions” – It will show all the regions on display. There will be an option to remove region within screen.

- Current region number – It will allow to change the index of regions to set the individual region setting

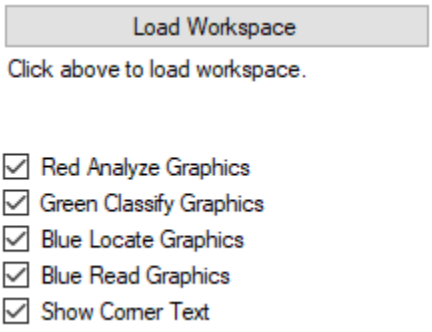


VIDI Pre Processing



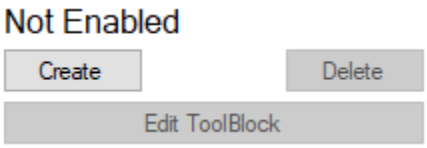
- Quickstart:
  - "Create" will allow to customize tool that will be applied to image before VIDI processing.
  - "Delete" will allow to delete any customization for pre processing.
  - "Edit ToolBlock" will open tool block editor for customization.

VIDI



- Quickstart:
  - "Load Workspace" will allow to select pre configured VIDI workspace file.
  - "Red Analyze Graphics" select whether to show red tool analyze graphics or not.
  - "Blue Locate Graphics" select whether to show green tool classify graphics or not.
  - "Blue Read Graphics" select whether to show blue tool read text graphics on screen.
  - "Show Corner Text" select whether to show short result on upper-left corner of the screen.

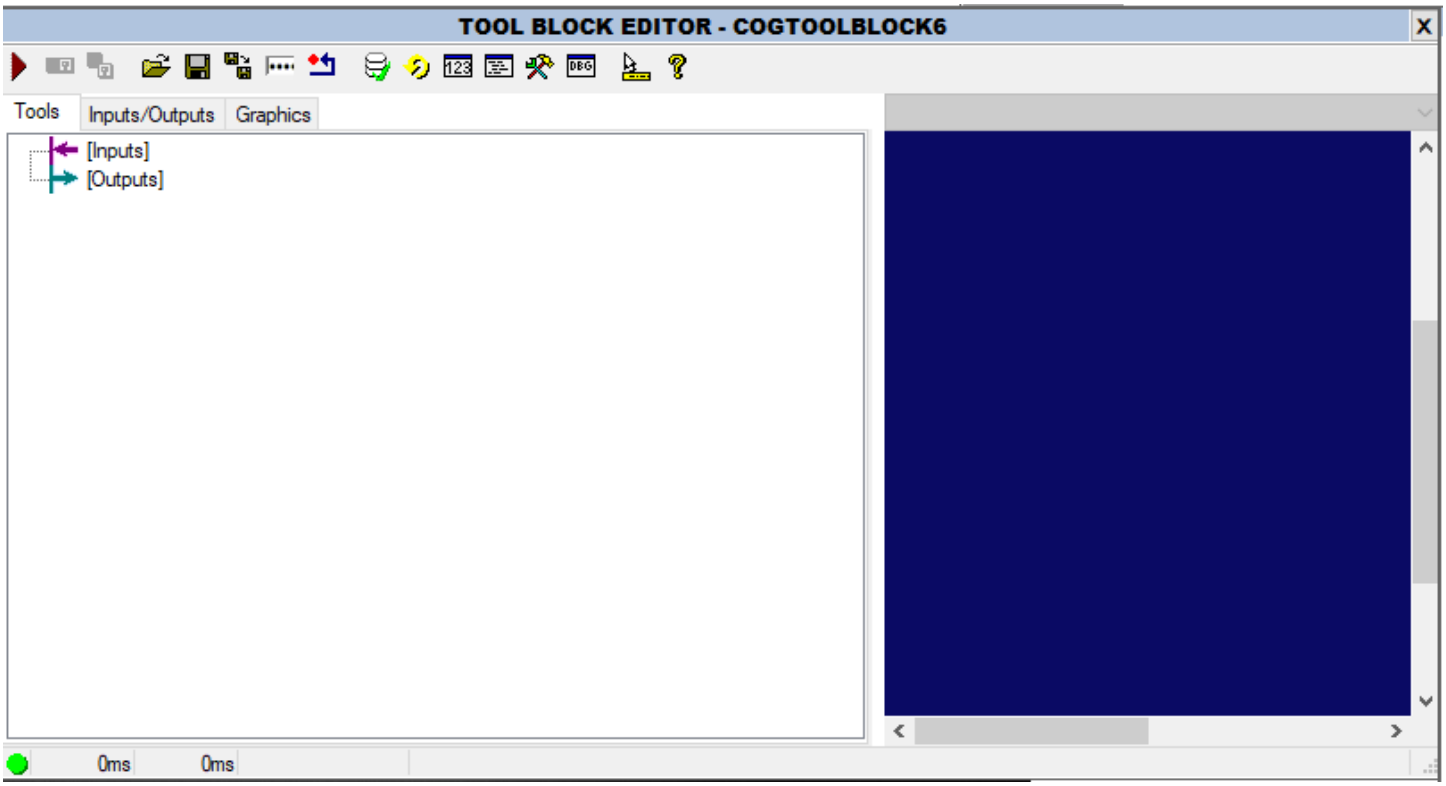
VIDI Post Processing



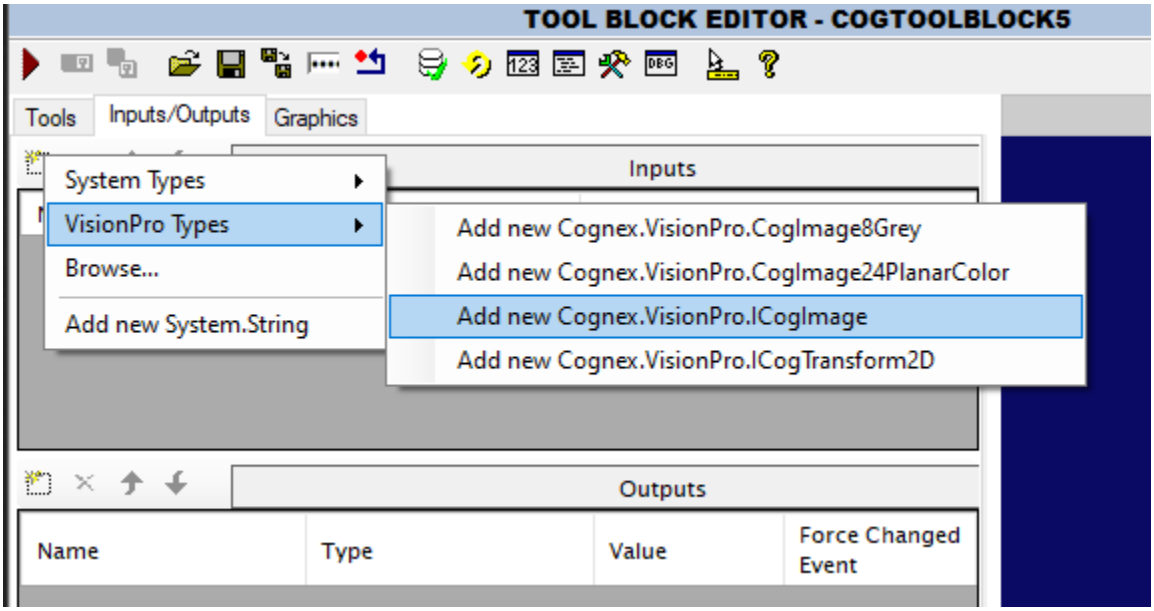
- Quickstart:
  - “Create” will allow to customize tool that will be applied to VIDI processing results.
  - “Delete” will allow to delete any customization for post processing.
  - “Edit ToolBlock” will open tool block editor for customization.

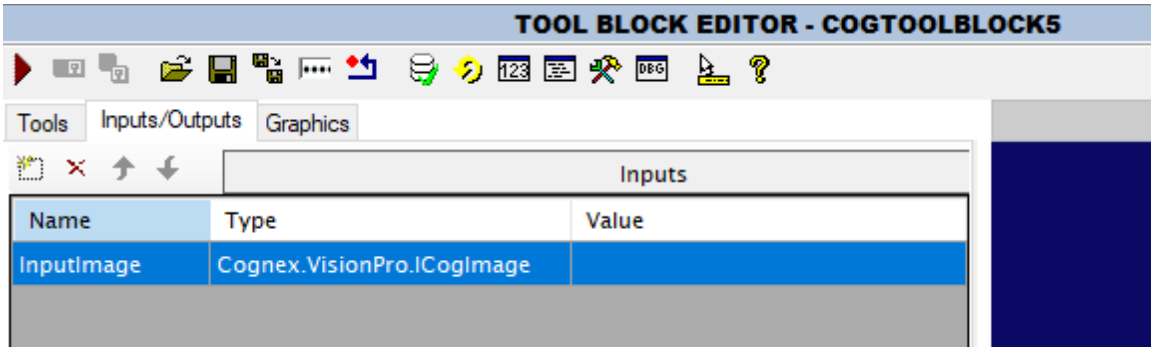
Custom Toolblock

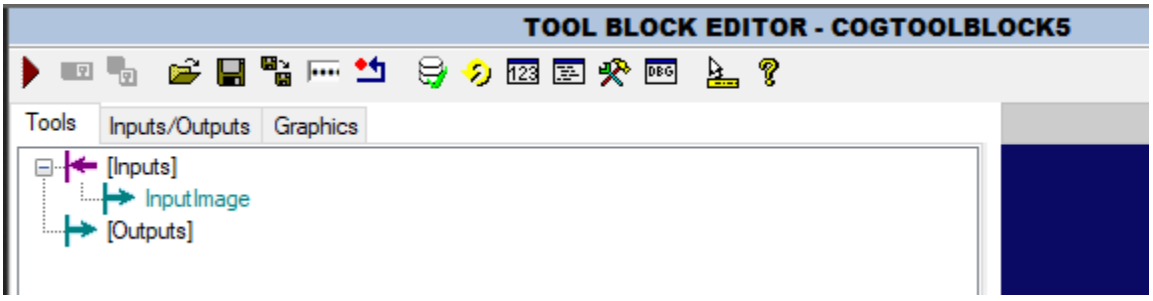
- Quickstart:
  - “Edit ToolBlock” Will allow to customize tool block

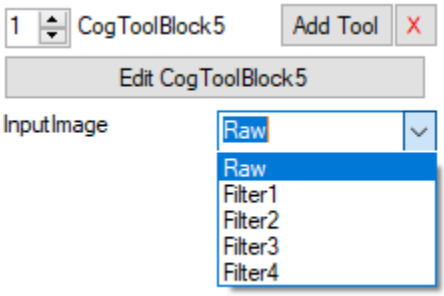


- Add UI Control:
  - “Input Image”



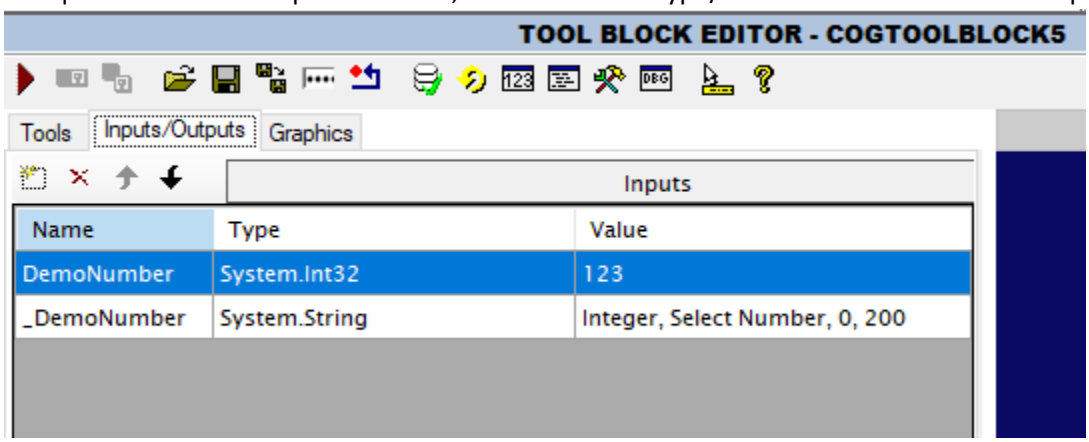






- “Number Picker”

  It requires to add two input terminals, one is to define type/value and other to set the UI properties.



DemoNumber is the main int type while _DemoNumber is to define additional properties of the UI control.

<u>Note - Just DemoNumber will not create any UI control, UI control’s value will be based on main terminal.</u>

_DemoNumber : it should be string type and value should be in specific format –

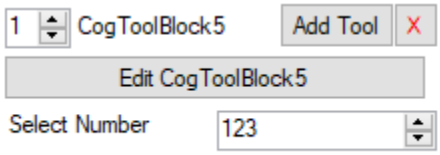
{type}, {display_name}, {min_value}, {max_value}

{type} – it must be “Integer”

{display_name} – It will be visible as text of the control

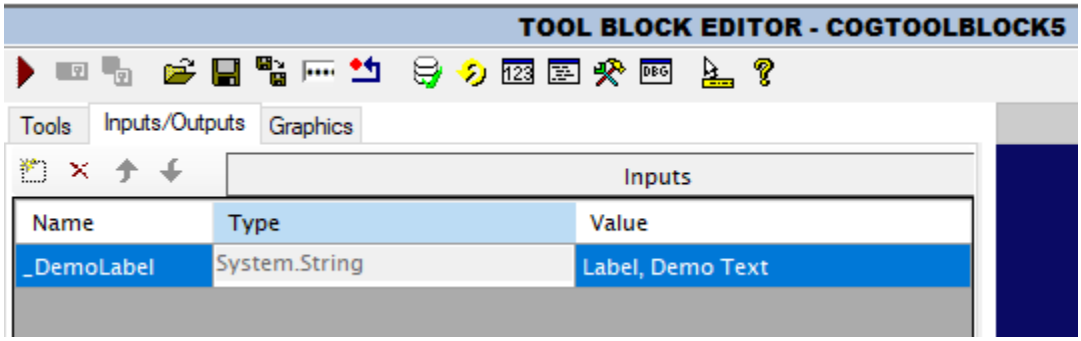
{min_value} – minimum allowed value of number picker

{max_value} – maximum allowed value of number picker



- “Label”

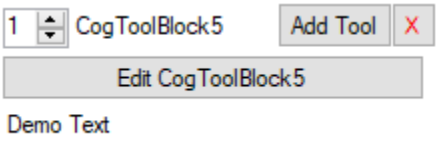
  It only requires one input terminal.



_DemoLabel : it should be string type and value should be in specific format –

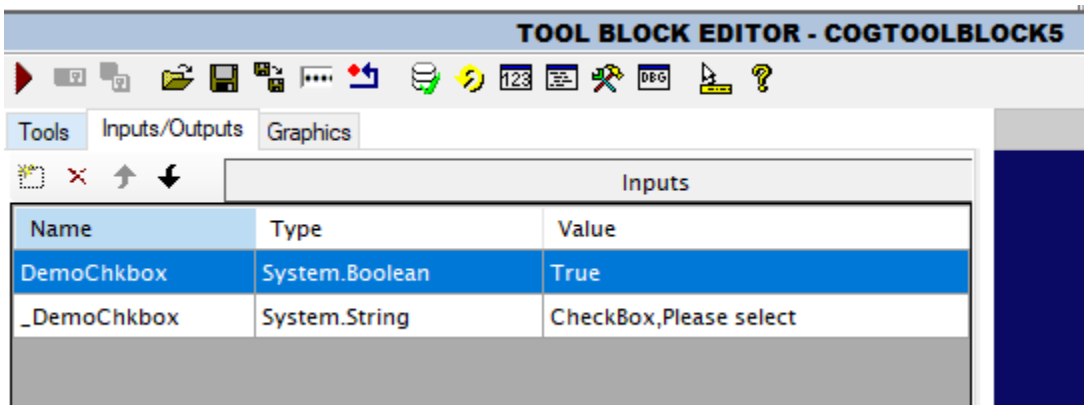
{type}, {display_name}

{type} – it must be “Label”
{display_name} – It will be visible as text of the control



- “Check box”

  It requires to add two input terminals, one is to define type/value and other to set the UI properties.



  DemoChkbox is the main int type while _DemoChkbox is to define additional properties of the UI control.

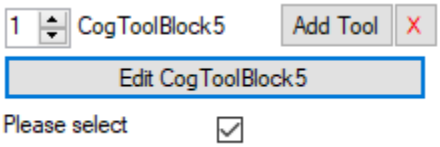
  Note - Just DemoChkbox will not create any UI control, UI control’s value will be based on main terminal.

  _DemoChkbox : it should be string type and value should be in specific format –

  {type}, {display_name}

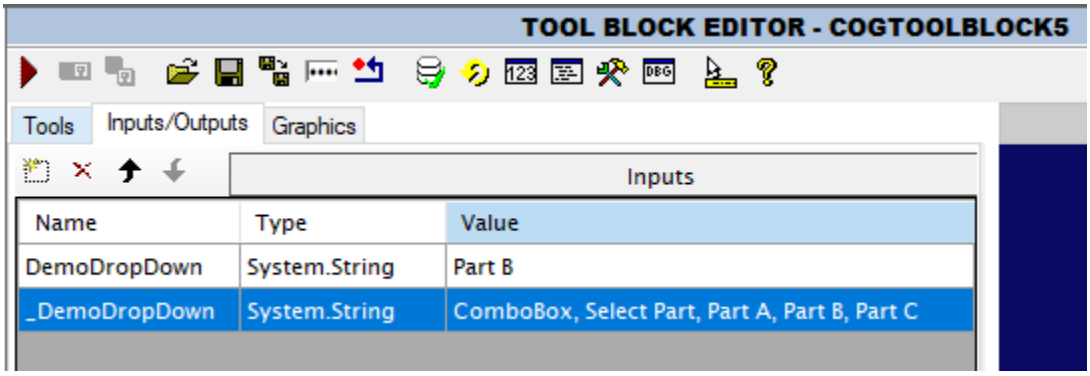
  {type} – it must be “CheckBox”

  {display_name} – It will be visible as text of the control



- “Dropdown”

  It requires to add two input terminals, one is to define type/value and other to set the UI properties.



DemoDropDown is the main int type while _DemoDropDown is to define additional properties of the UI control.

Note - Just DemoDropDown will not create any UI control, UI control’s value will be based on main terminal.

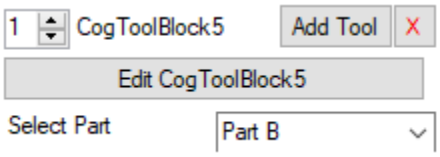
_DemoDropDown : it should be string type and value should be in specific format –

{type}, {display_name}, {comma_separated_values}

{type} – it must be “ComboBox”

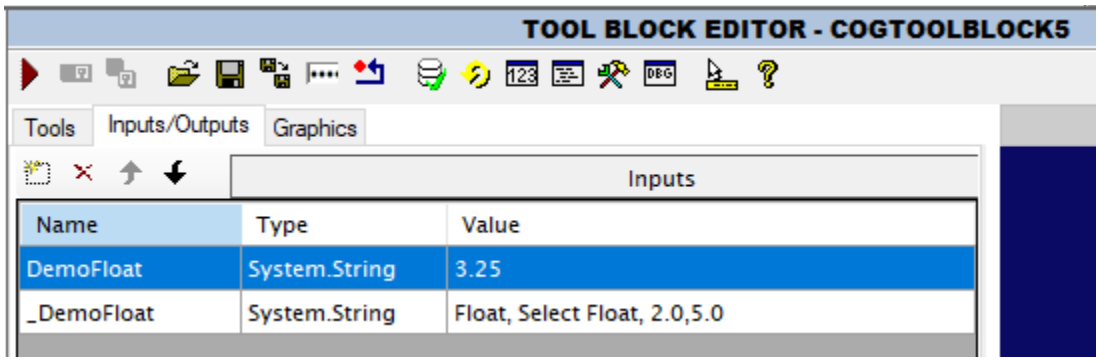
{display_name} – It will be visible as text of the control

{comma_separated_values} – List down all the values you want in dropdown as comma separated



- “Float”

  It requires to add two input terminals, one is to define type/value and other to set the UI properties.



DemoFloat is the main int type while _DemoFloat is to define additional properties of the UI control.

<u>Note - Just DemoFloat will not create any UI control, UI control’s value will be based on main terminal.</u>

_DemoFloat : it should be string type and value should be in specific format –

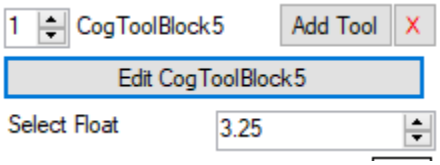
{type}, {display_name}, {min_value}, {max_value}

{type} – it must be “Float”

{display_name} – It will be visible as text of the control

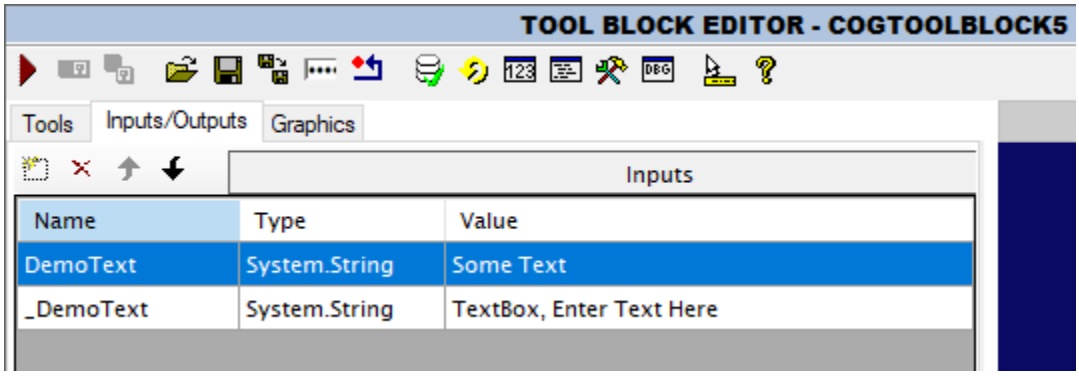
{min_value} – minimum allowed value of number picker

{max_value} – maximum allowed value of number picker



- “Text”

  It requires to add two input terminals, one is to define type/value and other to set the UI properties.



DemoText is the main int type while _DemoText is to define additional properties of the UI control.

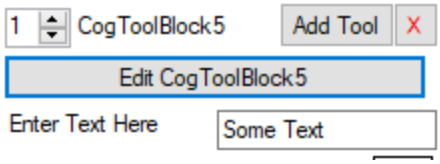
Note - Just DemoText will not create any UI control, UI control’s value will be based on main terminal.

_DemoText : it should be string type and value should be in specific format –

{type}, {display_name}

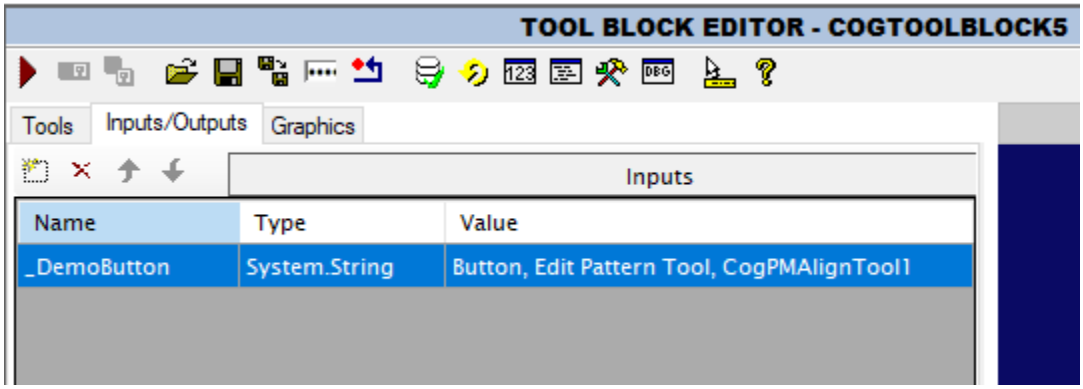
{type} – it must be “TextBox”

{display_name} – It will be visible as text of the control



- “Button”

  It only requires one input terminal.



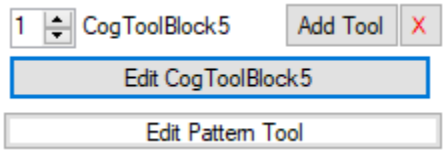
_DemoButton : it should be string type and value should be in specific format –

{type}, {display_name}, {tool_name}

{type} – it must be “Button”

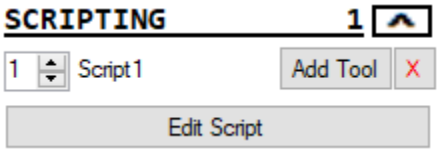
{display_name} – It will be visible as text of the control

{tool_name} – Enter the name of tool you want to edit



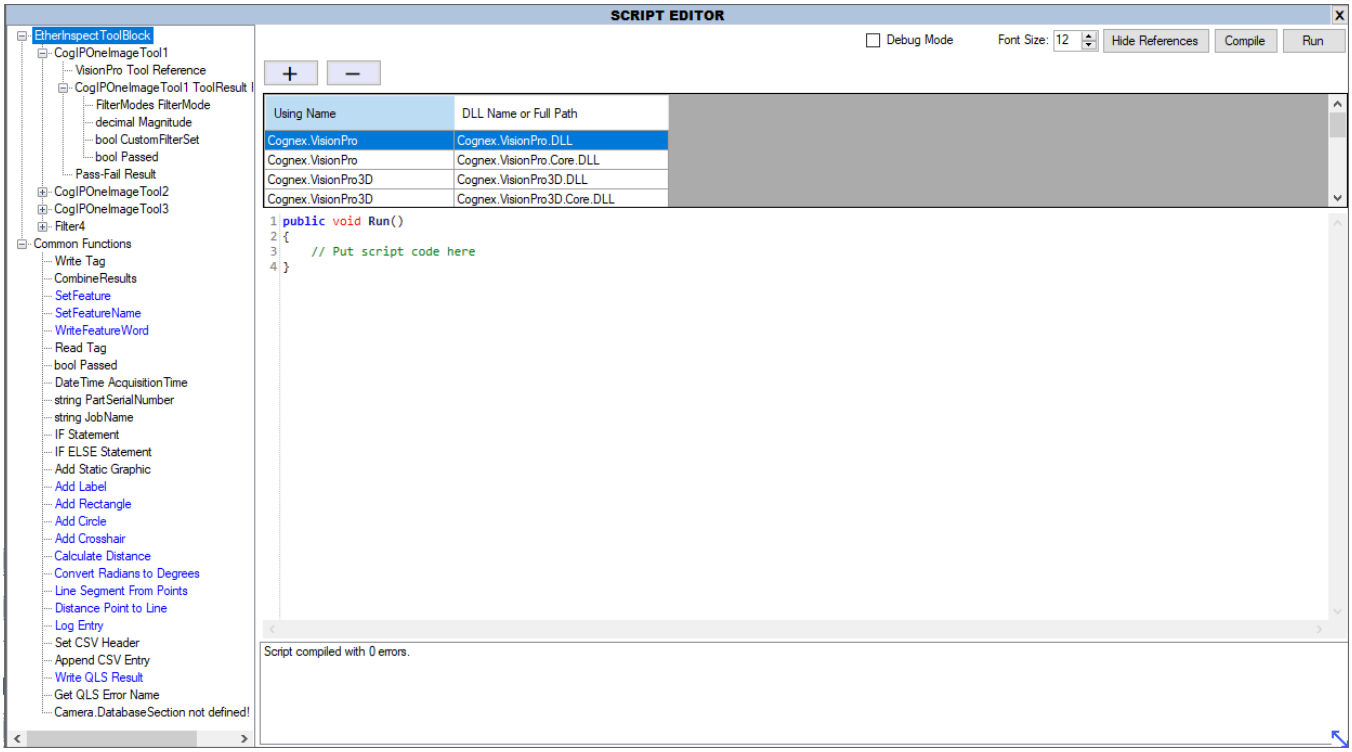
SCRIPTING FEATURES

Scripting Tool Display



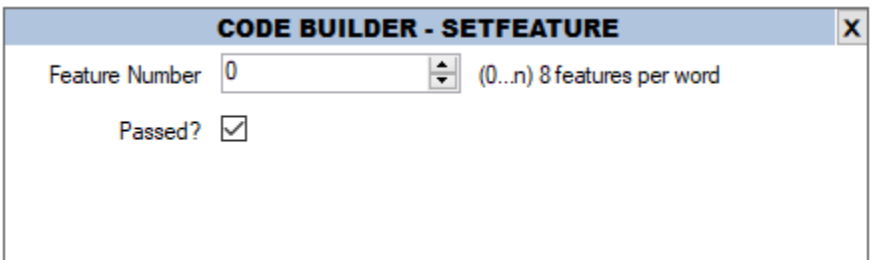
- Clicking “Edit Script” will open the Script Editor Display

Script Editor Display – Allows editing the C# code used in the Ether-Inspect scripting environment



- Changing the number in the up/down box will change the font size
- Clicking “Show References” will show the DLLs that are included when the script is compiled
- Clicking “Compile” will compile the script without running it. Errors show in the box at the bottom.
- Clicking “Run” will compile if needed, and run the script
- Double-clicking on an item in the left tree will paste sample code into the editor window
- Items in blue in the left tree will open the Script Code Builder display to provide user options before pasting in the code editor.

Script function dialog – Set Feature



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

Generated code will be pasted into the script as bellow

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

```

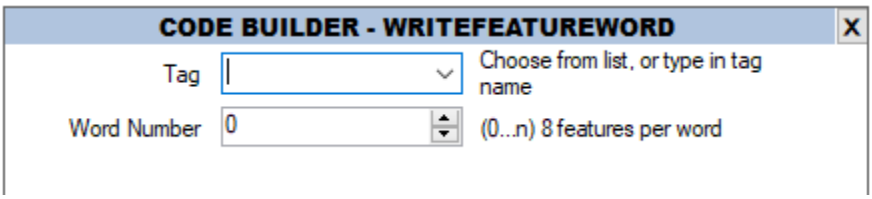
Script function dialog – Set Feature Name

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

Generated code will be pasted into the script as bellow

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

}

```

Script function dialog – Write Feature Word

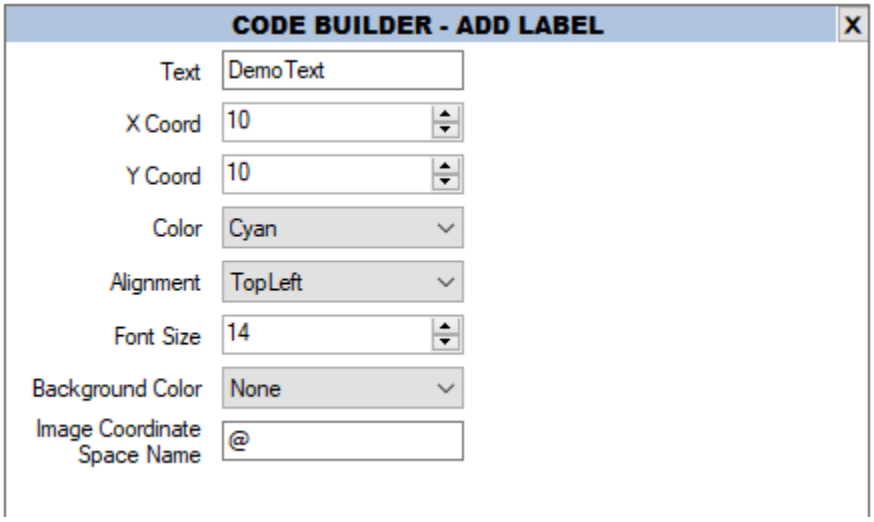


- Quickstart:
  - Tag – Select the tag name from list
  - Word Number – Set the word number you want to set

Generated code will be pasted into the script as bellow

```
1 public void Run()
2 {
3     // Put script code here
4     WriteFeatureWord("",0);
5
6 }
7
```

Script function dialog – Add Label

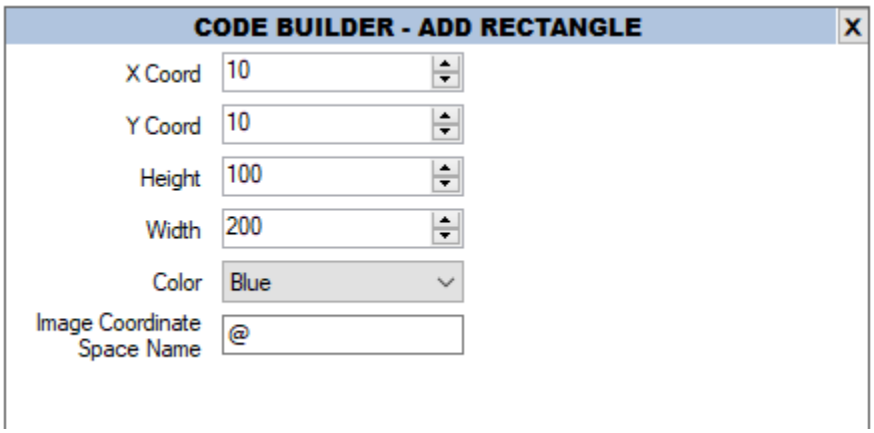


- Quickstart:
  - Text – Enter the text you want to set as Graphic label
  - X Coord – Set the label X coordinate
  - Y Coord – Set the label Y coordinate
  - Color – Select the color from the list
  - Alignment – Select the label text alignment
  - Font size – Adjust the font size
  - Background color – Select the background color
  - Image coordinate space name – Set the coordinate space

Generated code will be pasted into the script as bellow

```
public void Run()
{
    // Put script code here
    AddLabel("DemoText",10,10,CogColorConstants.Cyan,CogGraphicLabelAlignmentConstants.TopLeft,null /* default font */,14,CogColorConstants.None,"
}
```

Script function dialog – Add Rectangle

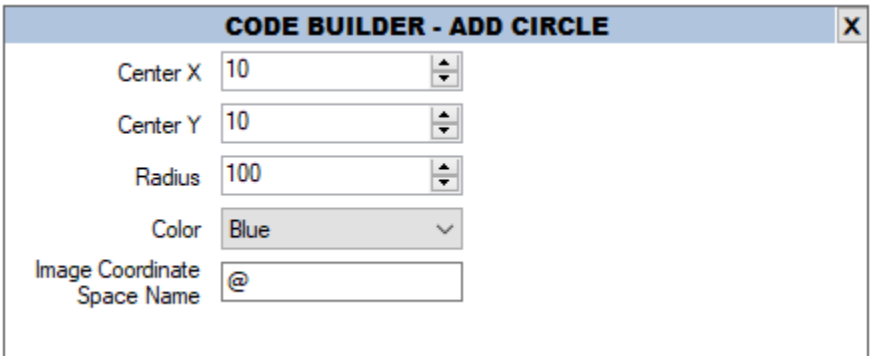


- Quickstart:
  - X Coord – Set the X coordinate of rectangle
  - Y Coord – Set the Y coordinate of rectangle
  - Height – Set the height of rectangle
  - Width – Set the width of rectangle
  - Color – Select the color of rectangle
  - Image coordinate space name – Set the coordinate space

Generated code will be pasted into the script as bellow

```
public void Run()
{
    // Put script code here
    AddRectangle(10,10,100,200,CogColorConstants.Blue,"@");

}
```

Script function dialog – Add Circle

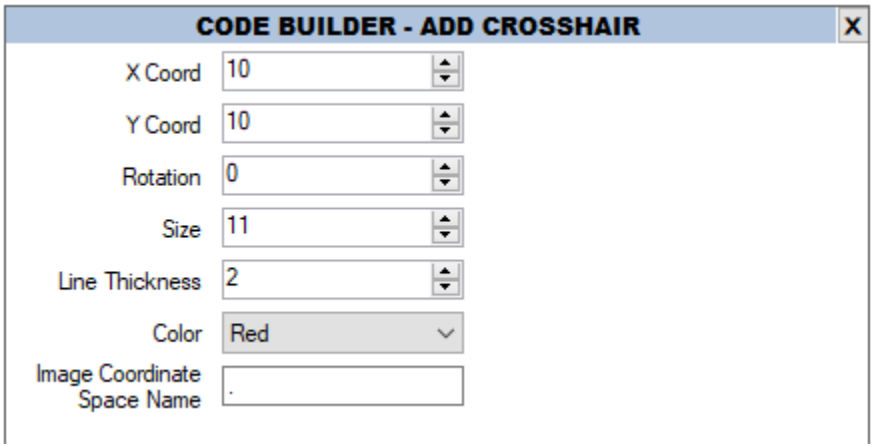


- Quickstart:
  - X Coord – Set the X coordinate of circle
  - Y Coord – Set the Y coordinate of circle
  - Radius – Radius of the circle
  - Color – Select the color of circle
  - Image coordinate space name – Set the coordinate space

Generated code will be pasted into the script as bellow

```
1 public void Run()
2 {
3     // Put script code here
4     AddCircle(10,10,100,CogColorConstants.Blue,"@");
5 }
6
```

Script function dialog – Add Crosshair

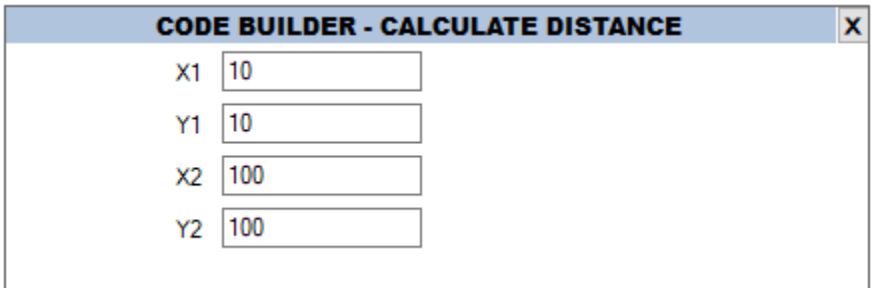


- Quickstart:
  - X Coord – Set the X coordinate of crosshair
  - Y Coord – Set the Y coordinate of crosshair
  - Rotation – Rotation of the crosshair
  - Size – Size of crosshair
  - Line Thickness – Thickness of the crosshair line
  - Color – Select the color of circle
  - Image coordinate space name – Set the coordinate space

Generated code will be pasted into the script as bellow

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

}

```

Script function dialog – Calculate Distance



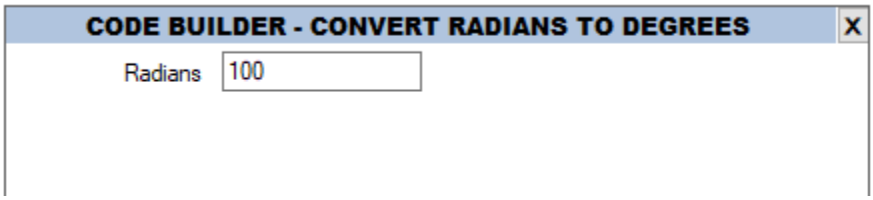
- Quickstart:
  - X1 – X coordinate of first point
  - Y1 – Y coordinate of first point
  - X2 – X coordinate of second point
  - Y2 – Y coordinate of second point

It will return the distance between two points

Generated code will be pasted into the script as bellow

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

}

```

Script function dialog – Covert Radius to Degree

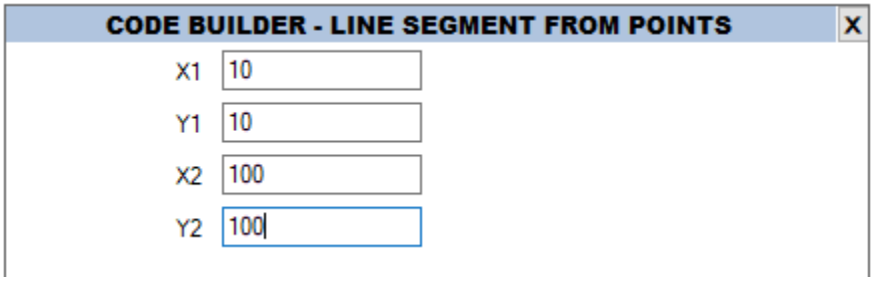


- Quickstart:
  - Radians – Enter radiance value you want to convert it to degree

Generated code will be pasted into the script as bellow

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

}

```

Script function dialog – Line Segment From Points

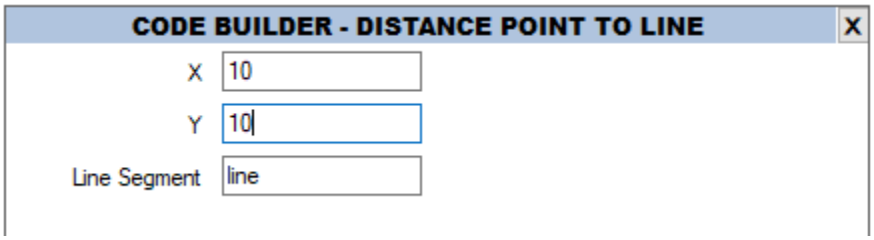


- Quickstart:
  - X1 – X coordinate of first point of line
  - Y1 – Y coordinate of first point of line
  - X2 – X coordinate of second point of line
  - Y2 – Y coordinate of second point of line

Generated code will be pasted into the script as bellow

```
public void Run()
{
    // Put script code here
    CogLineSegment line = LineSegmentFromPoints(10,10,100,100);

}

```

Script function dialog – Distance Point to Line

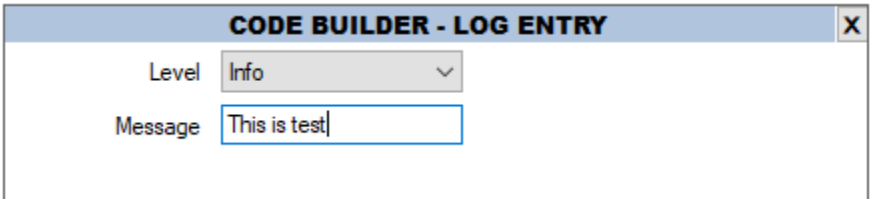


- Quickstart:
  - X1 – X coordinate of point
  - Y1 – Y coordinate of point
  - Line Segment – Line name from which you want to measure the distance between point

Generated code will be pasted into the script as bellow

```
public void Run()
{
    // Put script code here
    double pointToLineDistance = DistancePointToLine(10,10,line);

}

```

Script function dialog – Log Entry

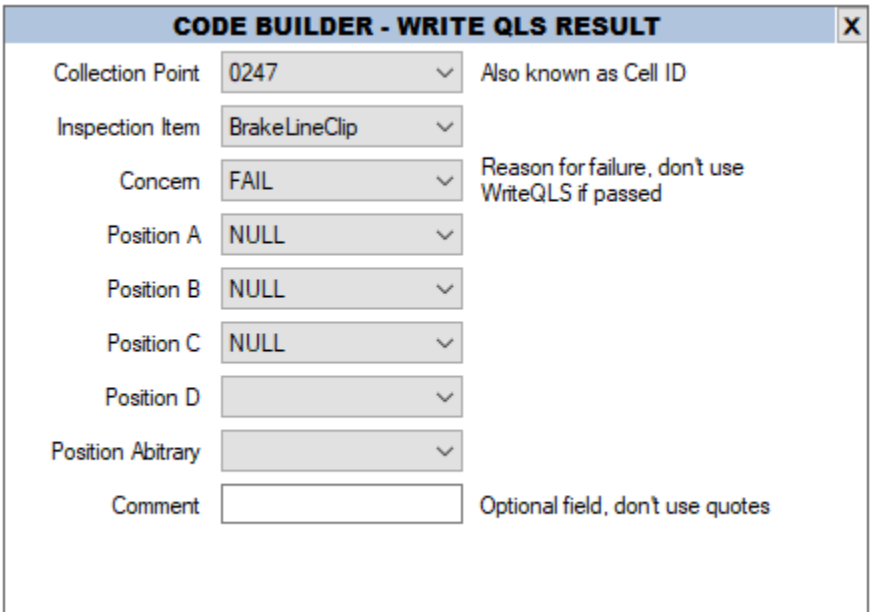


- Quickstart:
  - Level – Select the log level of entry. It could be Debug, Info, Warning or Error
  - Message – Text message you want to enter

Generated code will be pasted into the script as bellow

```
public void Run()
{
    // Put script code here
    Log.InfoFormat("This is test");

}

```

Script function dialog – Write QLS Result

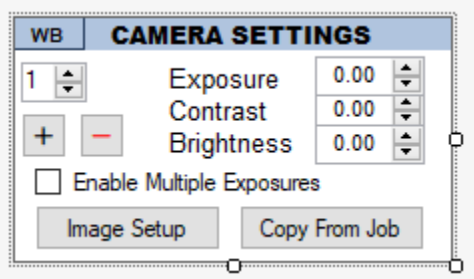


- Quickstart:
  - Collection point – Select the collection point, it is also known as Cell ID
  - Inspection Item – Select the inspection item name from the list
  - Concern – Select the concern from list or update it conditionally from the script
  - Position A – Select the position
  - Position B - Select the position
  - Position C - Select the position
  - Position D - Select the position
  - Position Arbitrary - Select the position
  - Comment – Write comment

Generated code will be pasted into the script as bellow

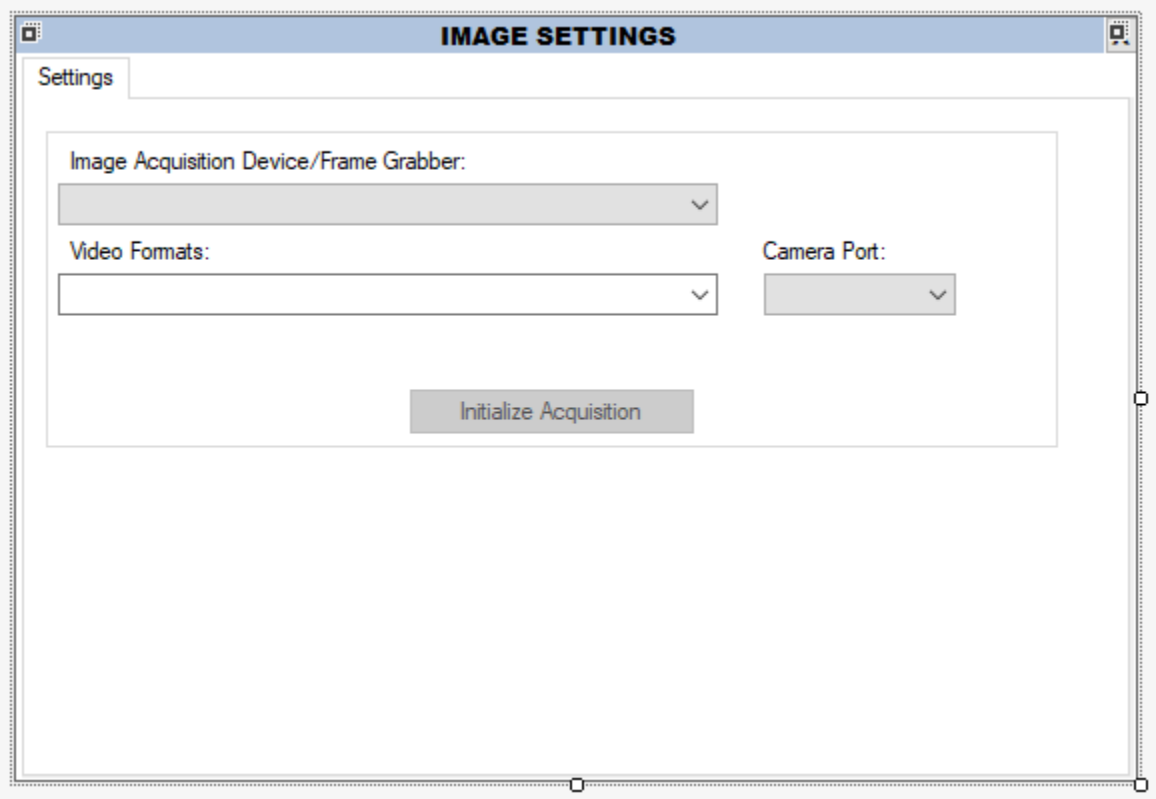
```
public void Run()
{
    // Put script code here
    int error = WriteQLS(JobResult.PartSerialNumber,"0247",JobResult.AcquisitionTime,"BrakeLineClip","FAIL","NULL","NULL","NULL","","","");

}

```

CAMERA SETTINGS AREA – Contains settings for Exposure, Brightness, Contrast, White Balance

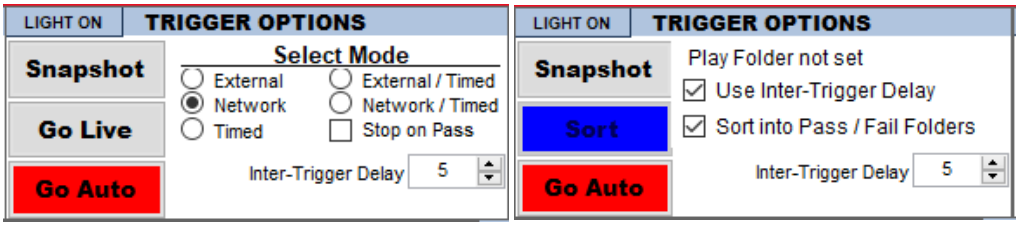


- Clicking “WB” will perform an auto white balance on the camera.
- Clicking “Enable Multiple Exposures” will show tools to add or delete exposure settings (max 5)
- Clicking “Image Setup” will open the Image Settings Display to configure the camera
- Clicking “Copy From Job” will copy the camera settings from another job file

Image Settings Display – **Allows configuring camera properties**



TRIGGER OPTIONS AREA– Has settings for trigger type, inter-trigger delay, and Auto/Manual mode change
Camera version on left, Simulator version on right



- Common Features:
  - Clicking “Snapshot” will capture a single image
  - Clicking “Go Auto” will put the camera in AUTO mode
  - Clicking “Go Manual” will put the camera in MANUAL mode
  - Setting Inter-Trigger Delay will change how long between triggers in Timed mode, or Image Playback mode
- Camera Features:
  - Clicking “Light On” will toggle the “TurnLightON” bit in the Ether-Inspect status word
  - Clicking “Go Live” will toggle a live display of the camera, for optics adjustment, no inspection will be performed
  - Clicking “External” will configure the camera for hardware trigger
  - Clicking “Network” will configure the camera for network trigger
- Simulator Features:
  - Clicking “Sort” will run through the images in the Play Folder.
  - Clicking “Sort into Pass/Fail Folders” determines if “Sort” will copy images into new Pass/Fail subfolders.
  - Clearing “Use Inter-Trigger Delay” will allow running as fast as possible when playing images.

RESULTS AREA – Shows inspection result summary at top

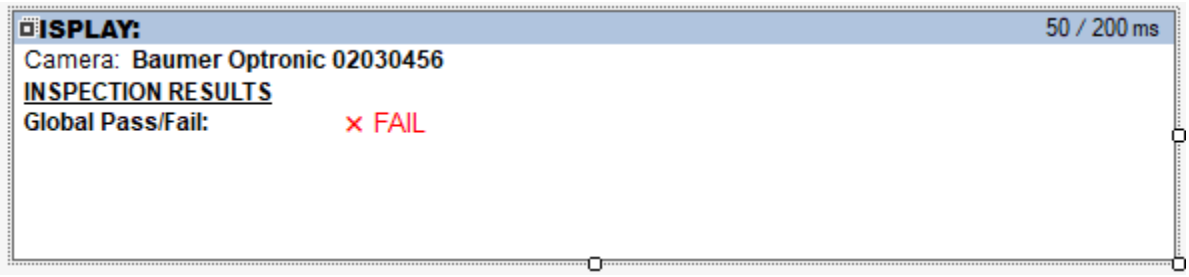
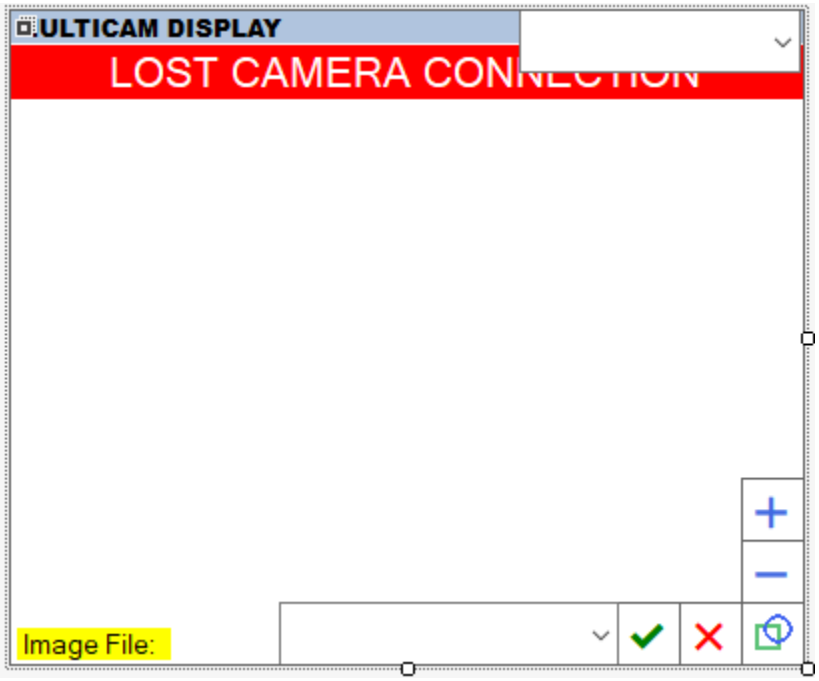


IMAGE AREA – Shows camera image with graphics (when in multicam mode, has a title)



- Clicking the drop-down in the upper right will show images from different tools in the job
- Clicking the shape symbol in the lower right will toggle between (All Graphics, Fail Only Graphics, and No Graphics)
- Additional features when editing a region:
    - Clicking the bottom drop-down will change the shape of the region
    - Clicking the checkmark will accept the region edit
    - Clicking the X will cancel the region edit
    - Clicking the + will add a vertex to a polygon region
    - Clicking the – will delete a vertex from a polygon region

Ether-Inspect remote interface

It is TCP based remote interface from which you can perform some operations in Ether-Inspect. Each application instance will have its own TCP listener and can be controlled separately.

Configuration

TCP Port – Application instance number + 9000. E.g. 9001 for the first Ether-Inspect instance.

Commands

Cameras – This will return list of available cameras from application instance.

Request: "CAMERAS;"

Response: it will return a string in specific format. String format - "{command},{number_of_camera},{comma_separated_cam_name};"

e.g. "CAMERAS,3,CAM1,CAM2,CAM3;"

Trigger – Camera can be triggered using this command

Request: "TRIGGER,{camera_name},{job_number};"

Camera_name (Datatype – string) – a valid camera name, please use the camera name returned from CAMERAS command.

Job_number (Datatype – int) – pass the job number which you want to change before triggering. -1 can be used if you want to trigger with current job.

Response: it will return a string in specific format, "NOK" will be returned in case of any issue. String format - "{command},{camera_name},{inspection_result};"

e.g. "TRIGGER,CAM1,true;"

ShowMultiCam – Camera view can be changed to multi cam using this command

Request: "SHOWMULTICAM;"

Response: it will return a string in specific format. String format - "{command},{result};"

e.g. "SHOWMULTICAM,true;"

ShowSingleCam - Camera view can be changed to single cam using this command

Request: "SHOWSINGLECAM,{camera_name};"

Camera_name (Datatype – string) – a valid camera name, please use the camera name returned from CAMERAS command.

Response: it will return a string in specific format, "NOK" will be returned in case of any issue. String format - "{command},{result};"

e.g. “SHOWSINGLECAM,true;”

AppName – Get the application name

Request: “APPNAME;”

Response: it will return a string in specific format. String format - “{command},{application_name};”

e.g. “APPNAME,Ether-Inspect;”

AppVersion – Get the application version

Request: “APPVERSION;”

Response: it will return a string in specific format. String format - “{command},{application_version};”

e.g. “APPVERSION,11.3.2;”

InstanceNumber – Get the application instance number

Request: “INSTANCENUMBER;”

Response: it will return a string in specific format. String format - “{command},{instance_number};”

e.g. “INSTANCENUMBER,1;”

CameraType – Get camera type by camera name

Request: “CAMERATYPE,{camera_name};”

Camera_name (Datatype – string) – a valid camera name, please use the camera name returned from CAMERAS command.

Response: it will return a string in specific format. String format - “{command},{camera_type};”

e.g. “CAMERATYPE,Virtual;”

LoadJob – Load job by number

Request: “LOADJOB,{camera_name},{job_number};”

Camera_name (Datatype – string) – a valid camera name

Job_number (Datatype – int) – a job number you want to load

Response: it will return a string in specific format. String format - “{command},{result};”

e.g. “LOADJOB,true;”

JobName – Get current job name as string

Request: “JOBNAME,{camera_name};”

Camera_name (Datatype – string) – a valid camera name

Response: it will return a string in specific format. String format - “{command},{job_name};”

e.g. “JOBNAME,DemoJobName;”

JobNumber – Get current job number

Request: “JOBNAME,{camera_name};”

Camera_name (Datatype – string) – a valid camera name

Response: it will return a string in specific format. String format - “{command},{job_number};”

e.g. “JOBNAME,2;”

GoLive – Put camera into live mode

Request: “GOLIVE,{camera_name};”

Camera_name (Datatype – string) – a valid camera name

Response: it will return a string in specific format. String format - “{command},{result};”

e.g. “GOLIVE,true;”

GoAuto – Put camera into auto mode

Request: “GOAUTO,{camera_name};”

Camera_name (Datatype – string) – a valid camera name

Response: it will return a string in specific format. String format - “{command},{result};”

e.g. “GOAUTO,true;”

GoManual – Put camera into manual mode

Request: “GOMANUAL,{camera_name};”

Camera_name (Datatype – string) – a valid camera name

Response: it will return a string in specific format. String format - “{command},{result};”

e.g. “GOMANUAL,true;”

State – Get current state or camera

Request: “STATE,{camera_name};”

Camera_name (Datatype – string) – a valid camera name

Response: it will return a string in specific format. String format - “{command},{state};”

e.g. “STATE,Ready;”

Mode – Get current trigger mode

Request: “MODE,{camera_name};”

Camera_name (Datatype – string) – a valid camera name

Response: it will return a string in specific format. String format - “{command},{mode};”

e.g. “MODE,Manual;”

InputTags – This will return list of input tags configured for camera

Request: “INPUTTAGS,{camera_name};”

Camera_name (Datatype – string) – a valid camera name

Response: it will return a string in specific format. String format - “{command},{number_of_tags},{comma_separated_tag_name};”

e.g. “INPUTTAGS,3,Tag1,Tag2,Tag3;”

OutputTags – This will return list of output tags configured for camera

Request: “OUTPUTTAGS,{camera_name};”

Camera_name (Datatype – string) – a valid camera name

Response: it will return a string in specific format. String format - “{command},{number_of_tags},{comma_separated_tag_name};”

e.g. “OUTPUTTAGS,2,Tag4,Tag5;”

InputTagType – Get data type of input tag

Request: “INPUTTAGTYPE,{camera_name},{tag_name};”

Camera_name (Datatype – string) – a valid camera name

Tag_name (Datatype – string) – a valid tag name, please use INPUTTAGS to get list of input tag names

Response: it will return a string in specific format. String format - “{command},{tag_type};”

e.g. “INPUTTAGTYPE,double;”

OutputTagType – Get data type of output tag

Request: “OUTPUTTAGTYPE,{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

Response: it will return a string in specific format. String format - “{command},{tag_type};”

e.g. “OUTPUTTAGTYPE,int;”

ReadInputTag – Read value of input tag

Request: “READINPUTTAG,{camera_name},{tag_name};”

Camera_name (Datatype – string) – a valid camera name

Tag_name (Datatype – string) – a valid tag name, please use INPUTTAGS to get list of input tag names

Response: it will return a string in specific format. String format - “{command},{tag_value};”

e.g. “READINPUTTAG,12.36;”

ReadOutputTag – Read value of output tag

Request: “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

Response: it will return a string in specific format. String format - “{command},{tag_value};”

e.g. “READOUTPUTTAG,54;”

WriteTag – Write value to output tag

Request: “WRITETAG,{camera_name},{tag_name},{value};”

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

Value (Datatype – object) – value you want to write, make sure datatype is matching

Response: it will return a string in specific format. String format - “{command},{result};”

Result will be true if data was written successfully, false otherwise.

e.g. “WRITETAG,true;”

Exposure – Set or get camera exposure value

Request: "EXPOSURE,{camera_name},{index},{value};"

Camera_name (Datatype – string) – a valid camera name

Index (Datatype – int) – index of multiple exposures

Value (Datatype – double) - the value of exposure you want to set, keep empty (NULL) if you want to get the value of current exposure without set

Response: it will return a string in specific format. String format - "{command},{current_exposure};"

Current_exposure (Datatype – double) - an exposure value will return, if command is called with set value argument then updated value will be returned.

e.g. "EXPOSURE,80;"

Contrast – Set or get camera contrast value

Request: "CONTRAST,{camera_name},{index},{value};"

Camera_name (Datatype – string) – a valid camera name

Index (Datatype – int) – index of multiple contrasts

Value (Datatype – double) - the value of contrast you want to set, keep empty (NULL) if you want to get the value of current contrast without set

Response: it will return a string in specific format. String format - "{command},{current_contrast};"

Current_contrast (Datatype – double) – a contrast value will return, if command is called with set value argument then updated value will be returned.

e.g. "CONTRAST,0.5;"

Brightness – Set or get camera brightness value

Request: "BRIGHTNESS,{camera_name},{index},{value};"

Camera_name (Datatype – string) – a valid camera name

Index (Datatype – int) – index of multiple brightness

Value (Datatype – double) - the value of brightness you want to set, keep empty (NULL) if you want to get the value of current brightness without set

Response: it will return a string in specific format. String format - "{command},{current_brightness};"

Current_brightness (Datatype – double) – a brightness value will return, if command is called with set value argument then updated value will be returned.

e.g. "BRIGHTNESS,0.5;"

MaxExposure – Get maximum exposure count

Request: "MAXEXPOSURE,{camera_name};"

Camera_name (Datatype – string) – a valid camera name

Response: it will return a string in specific format. String format - “{command},{count};”

e.g. “MAXEXPOSURE,5;”

EnableMultiExposure – Enable or disable multi exposure

Request: “ENABLEMULTIEXPO,{camera_name},{value};”

Camera_name (Datatype – string) – a valid camera name

Value (Datatype – boolean) – True or false to enable or disable multi exposure

Response: it will return a string in specific format. String format - “{command},{result};”

Result will be true if mode was updated successfully, false otherwise.

e.g. “ENABLEMULTIEXPO,true;”

Ether-Inspect web remote interface

It is http based web server interface from which you can perform some operations in Ether-Inspect. Each application instance will have its own web server and can be controlled separately.

Configuration

TCP Port – Application instance number + 80. E.g. 81 for the first Ether-Inspect instance.

URLs

http://server-name-or-ip:81/eiremote/v1/version

Returns: String - remote api version

http://server-name-or-ip:81/eiremote/v1/appname

Returns: String - application name

http://server-name-or-ip:81/eiremote/v1/appversion

Returns: String - Ether-Inspect application version

http://server-name-or-ip:81/eiremote/v1/instancenumber

Returns: int - application instance number

http://server-name-or-ip:81/eiremote/v1/cameras

This will return list of available cameras from application instance.

Returns: A list of Camera name List<String>

http://server-name-or-ip:81/eiremote/v1/showmulticam

Camera view can be changed to multi cam using this url request

Returns: boolean (success/fail)

http://server-name-or-ip:81/eiremote/v1/showsinglecam/{cameraname}

Camera view can be changed to single cam using this url request

{cameraname} - a valid camera name, please use the camera name returned from list of cameras url.

Returns: boolean (success/fail)

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/typename

{cameraname} - a valid camera name, please use the camera name returned from list of cameras url.

Returns: String – camera type name

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/trigger

It will trigger the camera and run inspection with current job

Returns: boolean – inspection result

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/trigger/{jobnumber}

{cameraname} - a valid camera name, please use the camera name returned from list of cameras url.

{jobnumber} - pass the job number which you want to change before triggering. -1 can be used if you want to trigger with current job.

Returns: boolean – inspection result

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/triggerwithresult

Returns: json based JobResult

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/triggerwithresult/{jobnumber}

{cameraname} - a valid camera name, please use the camera name returned from list of cameras url.

{jobnumber} - pass the job number which you want to change before triggering. -1 can be used if you want to trigger with current job.

Returns: json based JobResult

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/loadjob/{jobnumber}

Returns: boolean (success/fail)

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/jobname

Returns: string – current job name

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/jobnumber

Returns: int – current job number

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/liveacquire

Change current camera mode to Live

Returns: boolean (success/fail)

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/goauto

Change current camera mode to Auto

Returns: boolean (success/fail)

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/gomanual

Change current camera mode to Manual

Returns: boolean (success/fail)

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/state

Returns: string - current state of camera

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/mode

Returns: string - current trigger mode of camera

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/tags/input

This will return list of available input tag names.

Returns: A list of input tag name as string

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/tags/output

This will return list of available output tag names.

Returns: A list of output tag name as string

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/tags/{tagname}/inputtype

Get the datatype of input tag

Returns: string - datatype of given input tag

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/tags/{tagname}/outputtype

Get the datatype of output tag

Returns: string - datatype of given output tag

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/tags/{tagname}/readinput

Returns: object - current value of input tag

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/tags/{tagname}/readoutput

Returns: object - current value of output tag

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/tags/{tagname}/write/{value}

Write output tag value

Returns: Boolean - true if write was successful, false otherwise

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/exposure/{exposurenumber}/get

Returns: double - current exposure value of given exposure index

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/exposure/{exposurenumber}/set/{value}

Returns: Boolean - true if set operation was successful, false otherwise

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/contrast/{contrastnumber}/get

Returns: double - current contrast value of given contrast index

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/contrast/{contrastnumber}/set/{value}

Returns: Boolean - true if set operation was successful, false otherwise

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/brightness/{brightnessnumber}/get

Returns: double - current brightness value of given contrast index

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/brightness/{brightnessnumber}/set/{value}

Returns: Boolean -true if set operation was successful, false otherwise

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/maxexposurecount

Returns: int – maximum number of exposures

http://server-name-or-ip:81/eiremote/v1/cameras/{cameraname}/enablemultipleexposures/{value}

Returns: Boolean - true if set operation was successful, false otherwise

Ether-Inspect web server

It is http based web server which will provide HMI images and camera results.

Configuration

TCP Port – 80

URLs

http://localhost/

Provides list of available cameras

http://localhost/v*/list

Provides list of available cameras

http://localhost/v*/svg/{CameraName}

Provides latest svg image of given camera

http://localhost/v*/png/{CameraName}

Provides latest png image of given camera

http://localhost/v*/html/{CameraName}

Provides latest html image of given camera

http://localhost/v*/{CameraName}

Provides master html page with latest result of given camera

http://localhost/v*/detail/{CameraName}

Provides detail html page with latest result of given camera

http://localhost/v*/master/{CameraName}

Provides master html page with latest result of given camera

VisionLine remote interface

It is an Ethernet socket based communication between Ether-Inspect and a Trumpf laser controller. The Trumpf laser controller initiates communication.

Configuration

Port – 52000

Setting up VisionLine jobs – XML settings need to be added in Common.cfg file. VisionLine setting can be configured per job number per camera. Add below xml block within <CommonConfiguration> section in 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>

Elements

"CameraName" – this should match with EtherInspect camera name for which you want to configure VisionLine

"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, Will be used in IPSTART.

"PFOYOffset10MicronUnits" – Set Y Offset coordinate you want to set in PFO, Will be used in IPSTART.

"PFOZOffset10MicronUnits" – Set Z Offset coordinate you want to set in PFO, Will be used in IPSTART.

"PFOFocus10MicronUnits" – Set Focus Offset you want to set in PFO, Will be used in IPSTART.

"HighResolutionRotatoin" – set to True if you want to set the high-resolution rotation which will be used in IPRDY response.

Commands from Laser

IP Start – Start a new image processing cycle.

Request: “IPSTART,{ipstart_number};”

{ipstart_number} – set the ip start job number

Response: Ether-Inspect will provide the coordinates to focus the optics (PFO). String format -

“**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 Ether-Inspect is not ready

{code} – 768 (“RUNNING_ERROR = 0x0300”)

IP State Req – Inquire state of the image processing.

Request: “IPSTATEREQ;”

Response: Ether-Inspect will return 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”)

PFO Ready – Programmable focusing optics (PFO) in position. Once this command is received by Ether-Inspect, It will trigger the camera and respond with IPRDY (and vars) or IPBREAK and error code.

Request: “PFORDY;”

Response: Ether-Inspect will return the values of XVar or TVar tags when image processing is complete.

“**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”)