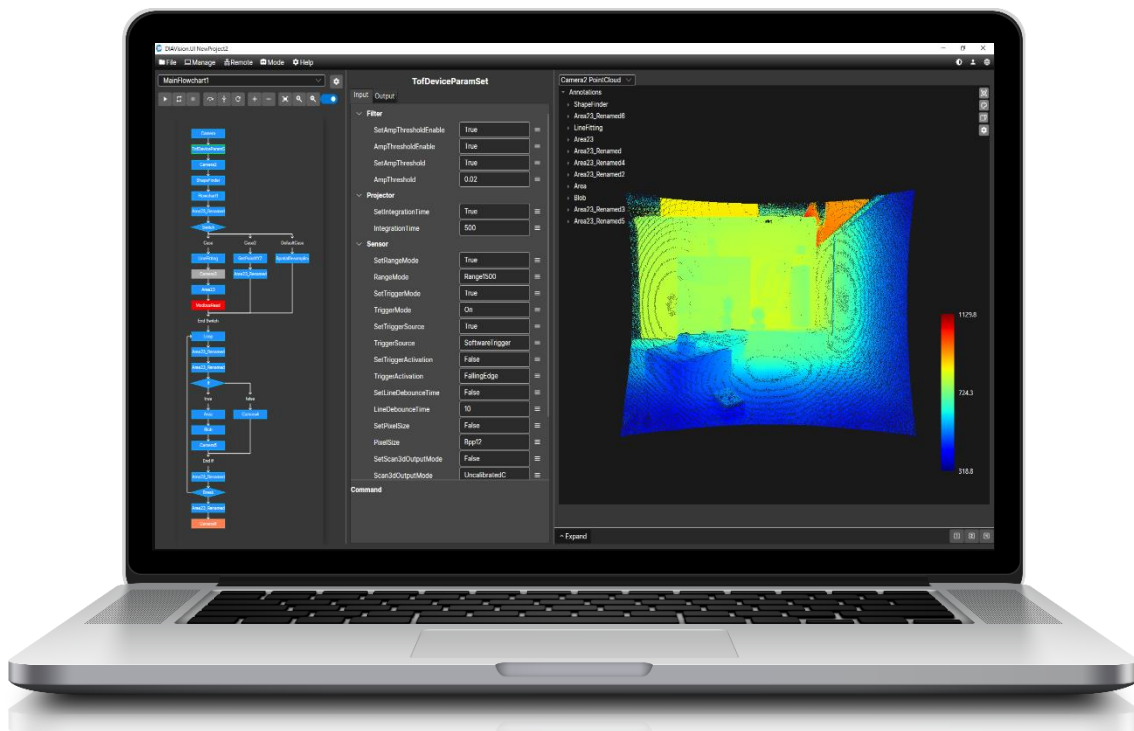


# DIAVision

## Machine Vision Platform

Operation Manual



# Version History

Version	Change History	Release Date	Software Version
V1.0	First Edition	2024/08/08	DMV-IVS V1.0.0

---

# Table of Content

<b>Chapter 1 Product Overview .....</b>	<b>1-1</b>
1.1 Product Introduction .....	1-1
1.2 Hardware Requirements .....	1-1
1.3 Software Installation .....	1-2
1.4 Features .....	1-5
<b>Chapter 2 Input/Output .....</b>	<b>2-1</b>
2.1 Communication Settings.....	2-1
2.2 Input and Output.....	2-2
2.3 Ethernet.....	2-2
2.4 Register .....	2-4
<b>Chapter 3 Basic Operations .....</b>	<b>3-1</b>
3.1 Introduction to The Main Page .....	3-1
3.1.1 Toolbar .....	3-1
3.2 Flowchart .....	3-29
3.2.1 Basic Operation of Flowchart .....	3-30
3.3 Component Input/Output Settings .....	3-32
3.4 Image Display .....	3-35
3.5 Result Statistics Information Setting.....	3-38
<b>Chapter 4 Vision Functions .....</b>	<b>4-1</b>
4.1 Camera.....	4-1
4.1.1 Camera .....	4-1
4.1.2 CameraParamGet .....	4-4
4.1.3 CameraParamSet.....	4-4
4.1.4 TofDeviceParamGet.....	4-5
4.1.5 TofDeviceParamSet .....	4-5
4.2 GPIO.....	4-6
4.3 Robot .....	4-6
4.3.1 RobotMove .....	4-6
4.3.2 RobotReadPose .....	4-8
4.4 Register .....	4-11
4.4.1 RegisterReader .....	4-11
4.4.2 RegisterWriter .....	4-12
4.5 Point Clouds.....	4-13

---

4.5.1 SpatialResampling .....	4-13
4.5.2 ProjectToAnyPlane .....	4-14
4.5.3 CroppingFromImage .....	4-17
4.5.4 PlaneFitting .....	4-19
4.5.5 PassThrough.....	4-20
4.5.6 ProjectToPlane .....	4-21
4.5.7 GetPointXYZ .....	4-23
4.5.8 StatisticalOutlierRemoval .....	4-23
4.6 Z-image Processing.....	4-24
4.6.1 PlaneFitting .....	4-24
4.7 Image Processing .....	4-26
4.7.1 Remap .....	4-26
4.7.2 Area .....	4-27
4.7.3 Blob .....	4-28
4.7.4 CircleFitting.....	4-33
4.7.5 EdegPosition.....	4-36
4.7.6 EdgePosTracing.....	4-38
4.7.7 Intensity.....	4-40
4.7.8 LineFitting .....	4-41
4.7.9 Position.....	4-43
4.7.10 ShapeFinder .....	4-46
4.7.11 PatternMatch .....	4-51
4.7.12 UndistortImage .....	4-52
4.8 File Operation.....	4-53
4.8.1 FileReader .....	4-53
4.8.2 FileWriter.....	4-53
4.8.3 ImageSave .....	4-54
4.8.4 PointCloudSave.....	4-54
4.9 Image Measurement.....	4-55
4.9.1 FindIntersection .....	4-55
4.10 Image Filter .....	4-56
4.10.1 Average .....	4-56
4.10.2 Gaussian.....	4-57
4.10.3 GrayLevelAssign .....	4-58
4.10.4 Invert .....	4-59
4.10.5 Median.....	4-60
4.10.6 ShadingCorrection .....	4-61
4.10.7 ShadingRemoval .....	4-62
4.10.8 Threshold .....	4-63
4.10.9 RelativeThreshold .....	4-64

---

4.11 Image Morphology.....	4-65
4.11.1 BottomHat .....	4-65
4.11.2 Close .....	4-66
4.11.3 Dilation .....	4-67
4.11.4 Erosion .....	4-68
4.11.5 Open.....	4-69
4.11.6 TopHat .....	4-70
4.12 Image Color Filter.....	4-71
4.12.1 ColorExtract.....	4-71
4.12.2 ColorThreshold .....	4-71
4.12.3 ColorDistance .....	4-73
4.13 Process Control.....	4-74
4.13.1 Display.....	4-74
4.13.2 Break .....	4-74
4.13.3 Continue.....	4-75
4.13.4 Delay .....	4-75
4.13.5 ReferenceFrame .....	4-76
4.13.6 Status .....	4-76
4.13.7 Store .....	4-77
4.13.8 Condition.....	4-78
4.13.9 Loop.....	4-79
4.13.10 Switch .....	4-79
4.14 3D Coordinate Conversion .....	4-80
4.14.1 NPointsWithZTransform.....	4-80
4.15 2D Coordinate Conversion .....	4-81
4.15.1 Placement .....	4-81
4.16 Communications .....	4-83
4.16.1 ConnectionRead.....	4-83
4.16.2 ConnectionWrite .....	4-84
4.16.3 ModbusRead.....	4-84
4.16.4 ModbusWrite.....	4-85
4.17 3D Calibration.....	4-86
4.17.1 NPointsWithZ.....	4-86
4.18 2D Calibration.....	4-89
4.18.1 FourPoints .....	4-89
4.18.2 NPointsWithTool .....	4-90
4.19 Mathematics.....	4-94
4.19.1 MatInverse .....	4-94
4.19.2 MatMul .....	4-95

---

<b>Chapter 5 Plug-in Development</b> .....	<b>5-1</b>
5.1 Plug-in Installation, Debugging and File Description .....	5-1
5.1.1 Plug-in Installation Debugging.....	5-1
5.2 Plug-in Attribute Description .....	5-12
5.3 Plug-in UI.....	5-13
5.4 Getting Image and Point Cloud Raw Aata .....	5-14

# Chapter 1

## Product Overview

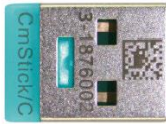
### 1.1 Product Introduction

DIAMVision, Delta's Machine Vision Platform, includes:

1. DMV-IVS software



2. USB Dongle



The DMV-IVS software can be downloaded from Delta website, please refer to Section 1.3 for download and installation details. The USB dongle is required for DMV-IVS activation and execution.

### 1.2 Hardware Requirements

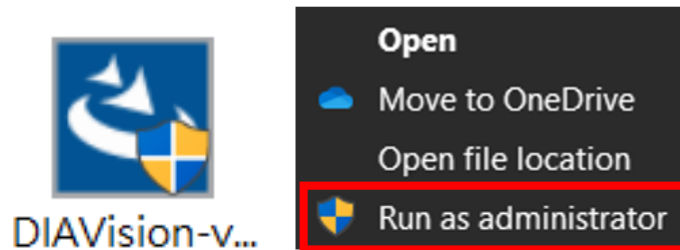
The following are the minimum hardware requirements for DMV-IVS.

Operating System	Win10-x64 20H2
Memory	8G
Storage	32GB
Display Resolution	1920 x 1080
USB Dongle	For DMV-IVS software activation and execution

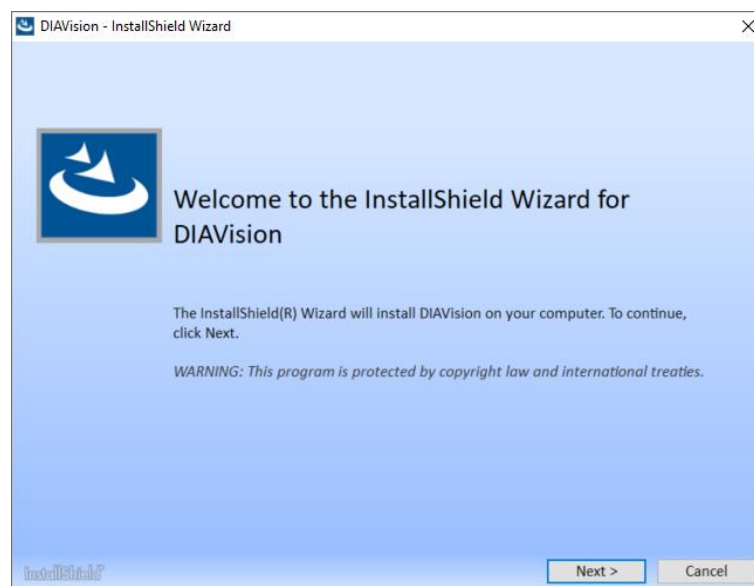
## 1.3 Software Installation

The installation steps are as follows:

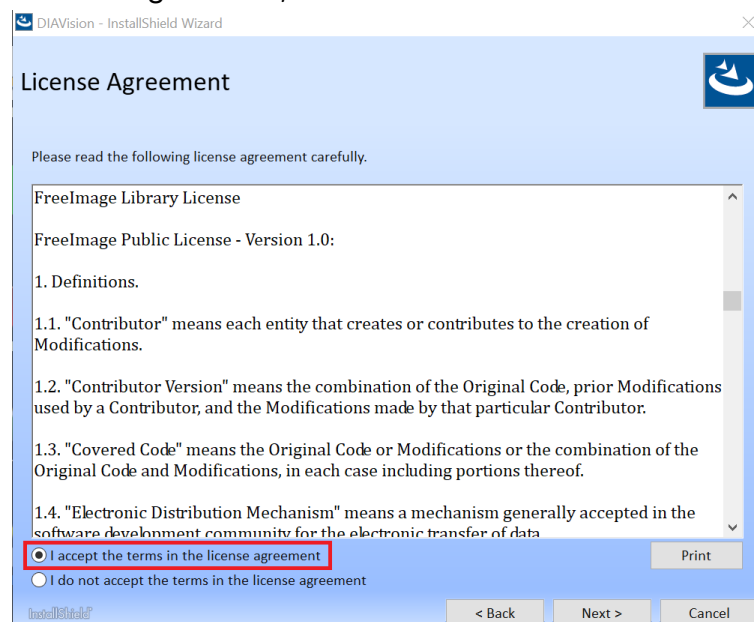
1. Right-click on the installed software and execute as a system administrator.



2. Select "Next".



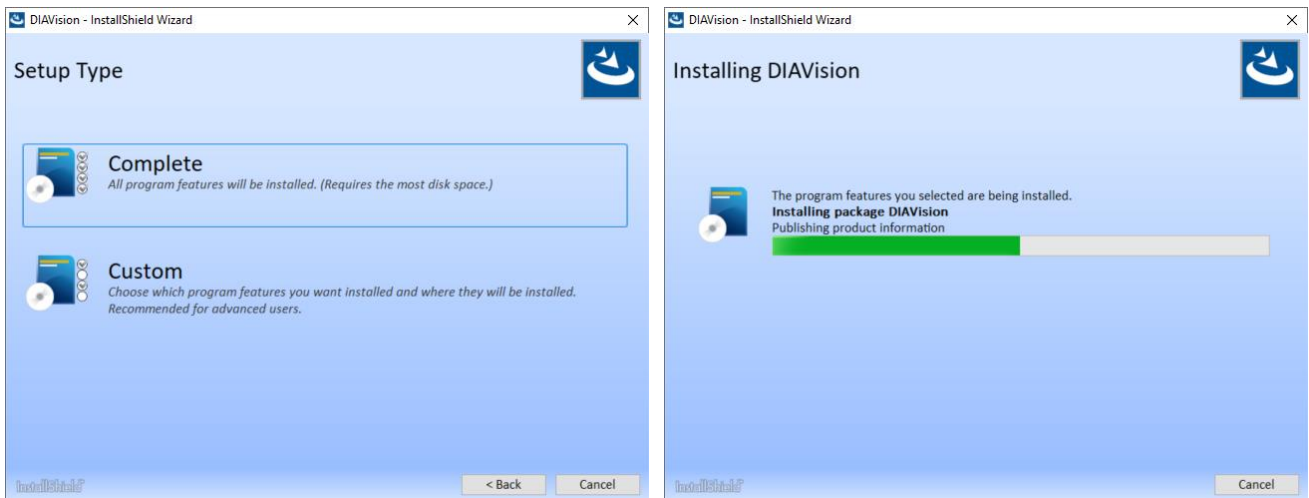
3. Agree and accept the license agreement, and click "Next".



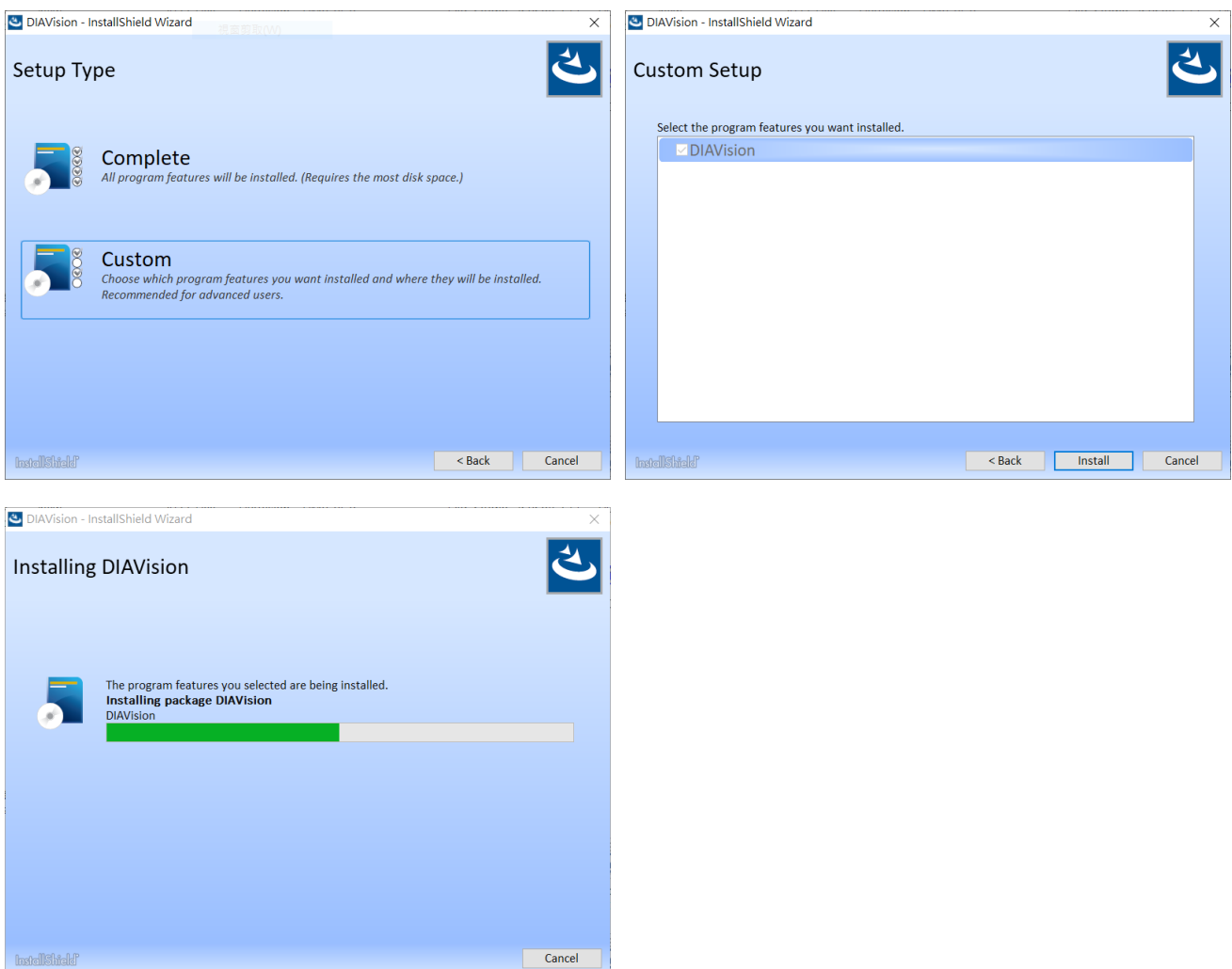


4. Select your preference for Setup Type, then the system will start the installation.

### Complete Installation:



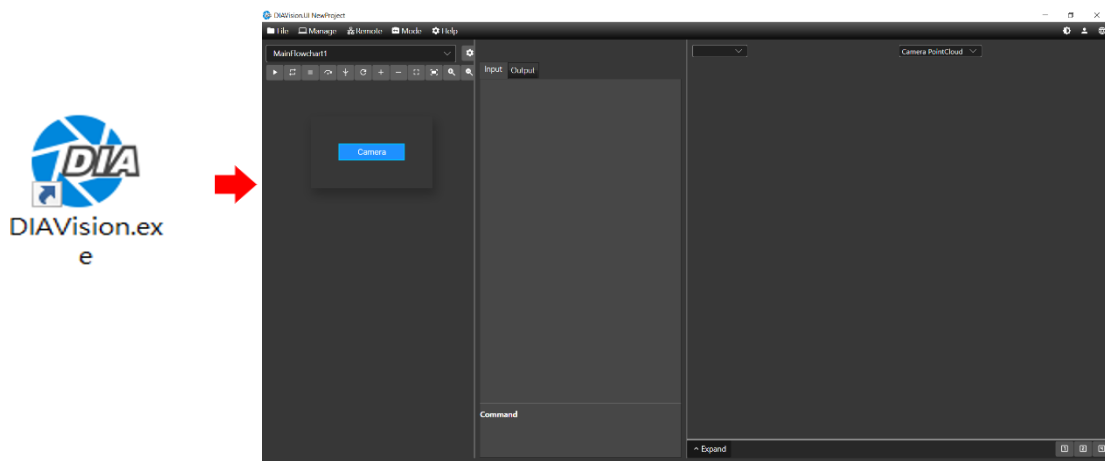
### Custom Installation:



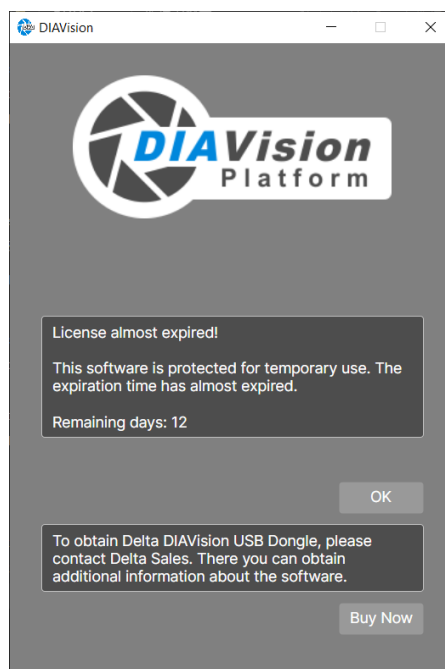
5. After installation completed, please click "Finish".



6. Double-click on the icon on the desktop to launch the software.



7. DIAVision trial permission will have restriction of usage days, the information will be shown on the UI.



## 8. The installation paths:

- Program: C:\Program Files\Delta Industrial Automation\DIAVision\DMV-IVS\
- Data: C:\Users\Public\Documents\DMV-IVS

## 1.4 Features

Maximum number of projects	Unlimited (Depends on storage space)
Function blocks	See Chapter 4 for details
Communication interface	<ul style="list-style-type: none"> <li>• I/O (The number of inputs and outputs depends on the I/O card used by users)</li> <li>• Ethernet</li> </ul>
Communication protocols	Ethernet: Modbus TCP and private TCP
Robot settings	Support ABB, Delta, Epson, Fanuc, Yamaha, Yaskawa
Operation permissions	Guest and User (Password 2222 for sign in)
Supported Languages	English, Simplified Chinese, Traditional Chinese

# Chapter 2

## Input/Output

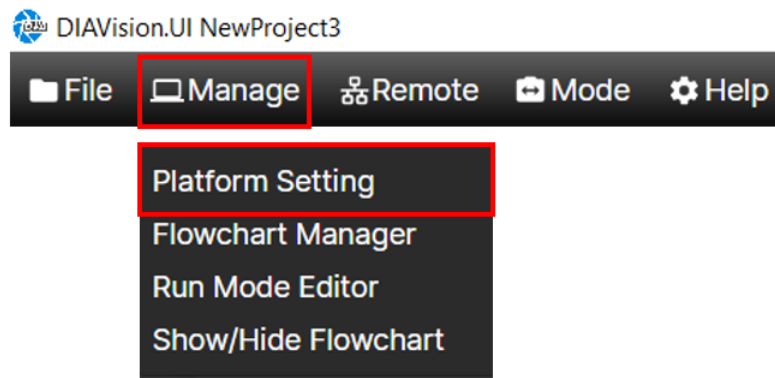
DIASion supports the Input/Output interfaces including:

- I/O (Input/Output)
- Ethernet

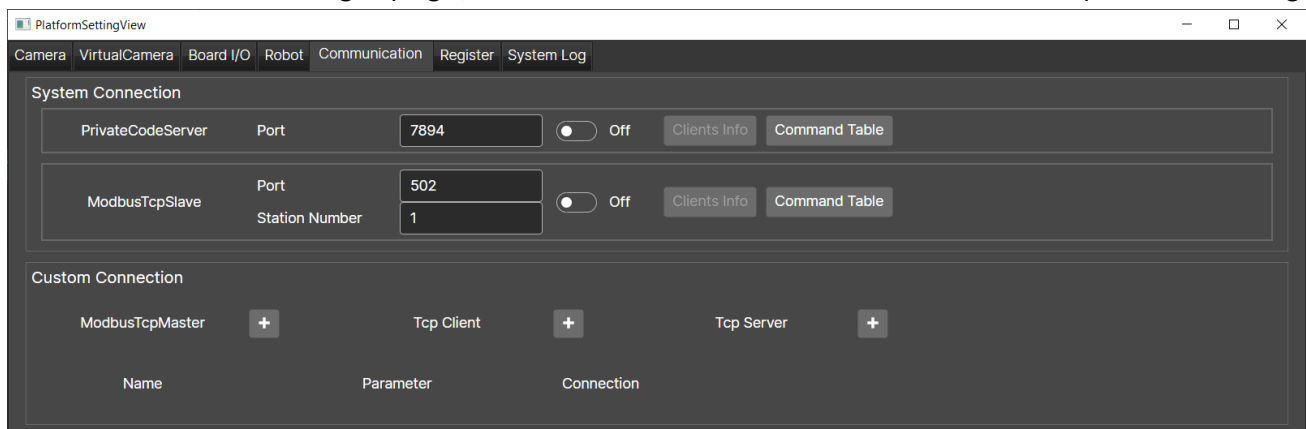
### 2.1 Communication Settings

Please refer to the following steps for the settings.

A. On the Toolbar: select "Manage" → "Platform Settings".



B. On the "Platform Settings" page, select "Communication" for communication protocol setting.

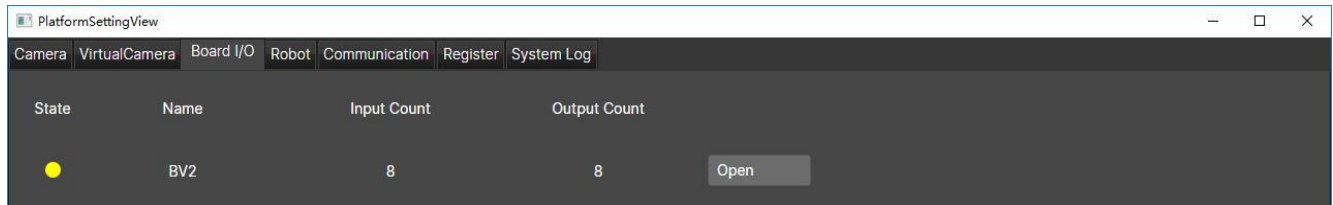


## 2.2 Input and Output

DIARVision supports I/O input and output functions through cameras or computers.

### ➤ Computer I/O

DIARVision supports BV2 computer I/O interface. Users can enable the I/O function via the process: "Manage" → "Platform Settings" → "Board I/O".



Indicator status:

- Green: Enabled
- Yellow: Not Enabled

The logic configuration of the board I/O needs to be set by the GPIO function, as detailed in Chapter 4 Vision Functions.

### ➤ Camera I/O

If connected cameras have an I/O interface, it can be used with GPIO for signal input and output, see Chapter 4 Vision Functions for details.

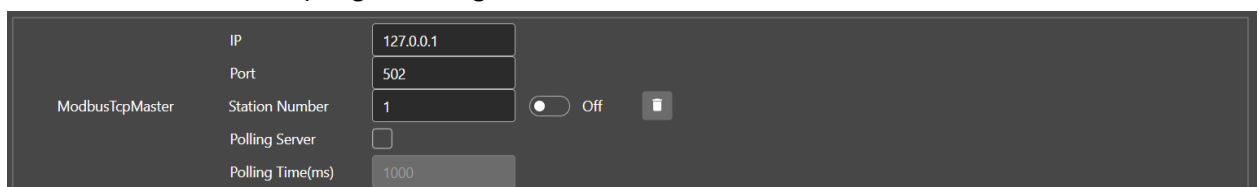
## 2.3 Ethernet

The Ethernet interface supports data transmission via Modbus or TCP communication protocols, and the user can choose either the master or slave application architecture.

### ➤ Modbus

#### ● Modbus TCP Master

Users can add Modbus Read or Modbus Write components to the flow, as detailed in Chapter 4. The specified scratchpad data of the external device can be read by the component or the specified result data can be written to the specified scratchpad of the external device. This can simplify the process of string decoding after string received and reduce the burden for programming.



- IP: The IP address of the device to be connected
- Port: The port to which you want to connect the device
- Station Number: Port station number
- Polling server: Select to enable polling, and the software will confirm whether to connect according to the polling time

- Polling Time: Set the polling time period, in milliseconds
- Modbus TCP Slave
 

When an external controller is allowed to receive a command via the Modbus protocol, the software responds to the received command.

- Port: The software port
- Station number: software station number

## ➤ TCP

- TCP Client
 

The system will send the results in a string communication format.

- IP: The IP address of the device to be connected
- Port: The port to which you want to connect the device
- Station Number: Port station number
- Polling server: Select to enable polling, and the software will confirm whether to connect according to the polling time
- Polling Time: Set the polling time period, in milliseconds
- TCPServer
 

The system receives an external input string command and will respond based on the command received.

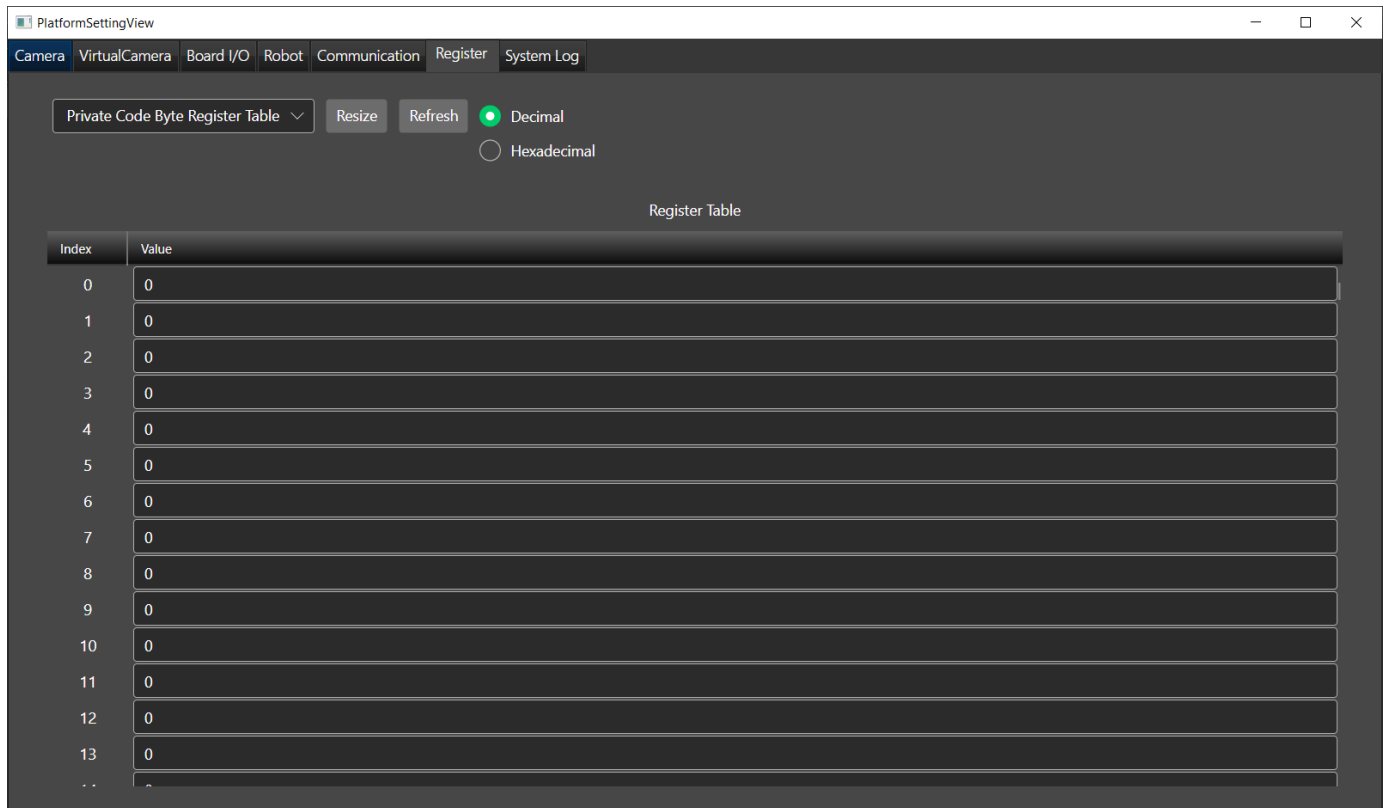
- Port: Listening Port.
- PrivateCodeServer
 

The system receives an external input string command.

- Port: Listening Port.

## 2.4 Register

Users can check the communication result from the Register. The methods to enable the Register: “Manage” → “Platform Settings” → “Register”. The operation instructions are detailed in Chapter 3.



The screenshot shows the PlatformSettingView application window. The title bar reads "PlatformSettingView". The menu bar includes "Camera", "VirtualCamera", "Board I/O", "Robot", "Communication", "Register", and "System Log". The "Register" menu item is selected. Below the menu bar, there is a dropdown menu set to "Private Code Byte Register Table", and buttons for "Resize" and "Refresh". There are two radio buttons for the display format: "Decimal" (selected) and "Hexadecimal".

The main area displays a "Register Table" with the following data:

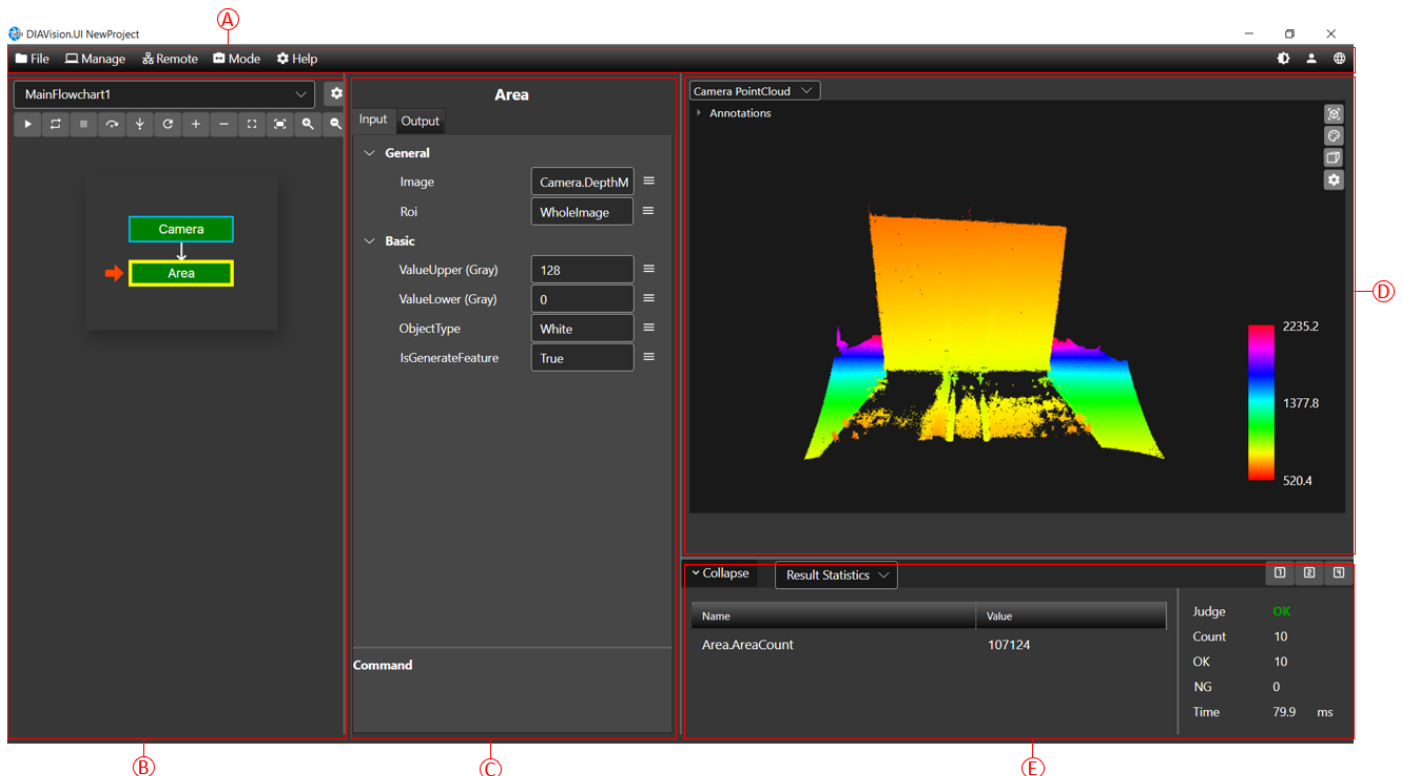
Index	Value
0	0
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
..	0

# Chapter 3

## Basic Operations

### 3.1 Introduction to The Main Page

The following image shows the software edit mode page:



- A. Toolbar: contains settings such as files, software platforms, process management, network interface deployments, and modes.
- B. Project Flowchart: Set up the project component flow.
- C. Component Settings: The property settings of each component in the project.
- D. Image Display Page: Displays the image.
- E. Result Statistics/System Log: Process test result display or switchable system log confirmation software log.

#### 3.1.1 Toolbar

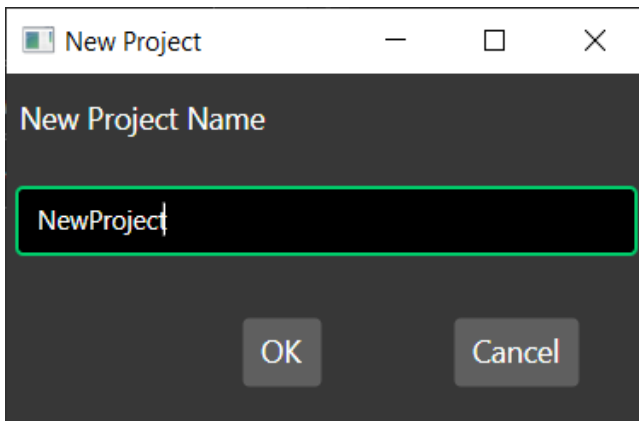
- (1) File: Project-related access function.
- (2) Manage: Manage camera, IO, network, register, robot, and other related functions.
- (3) Remote: Remote-related settings. Includes remote connections, remote project deployments, and remote picture deployments.
- (4) Mode: Toggle between editing mode or running mode.



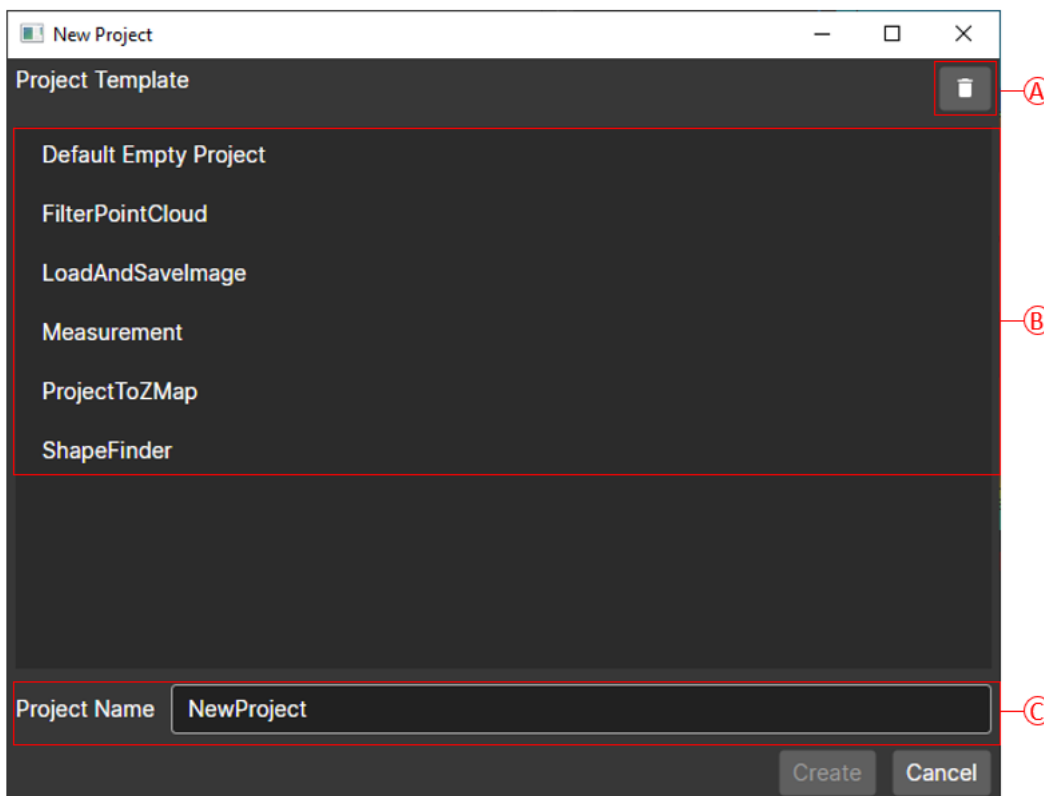
- (5) Help: DIAVision Platform Description.
- (6) Theme: Switch between dark or light theme.
- (7) Permissions: Permission login and logout management.
- (8) Language: Switching language families.

### 3.1.1.1 File

- New: Click "File" → "New" in the function line, enter the project name to create a new project.

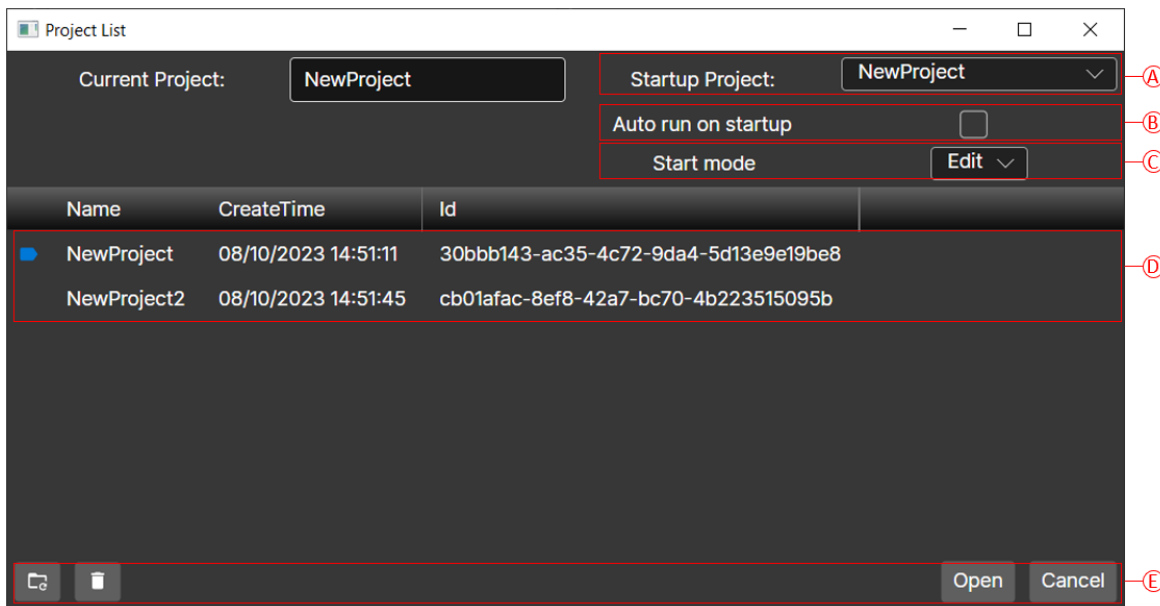


- New from template: Add a project with the original project structure.



- A. Delete the template
- B. Select New Template
- C. Set the project name

- Open: Click "File" → "Open" in the function bar to open the project list window, select the project and press Open to open the selection project.

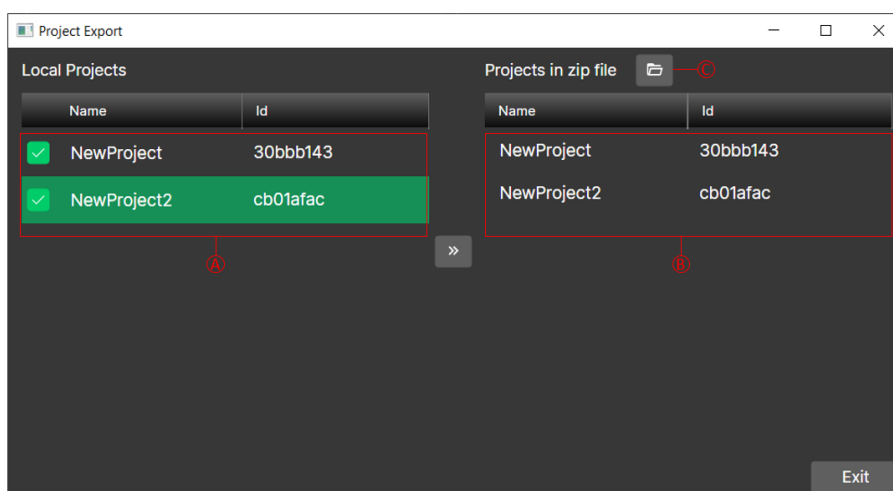


- A. Startup Project: The default project that the user can configure when the software is opened.
- B. Auto run on startup: If you select this option, the software will automatically execute the project and enter the run continuous mode.
- C. Start mode: The user can set it to edit or run mode according to their needs.
- D. List: Displays the items on the current computer. The blue is marked with the current project.
- E. Component Buttons:

 Refresh: Rearrange the list of items.

 Delete: Deletes the selection.

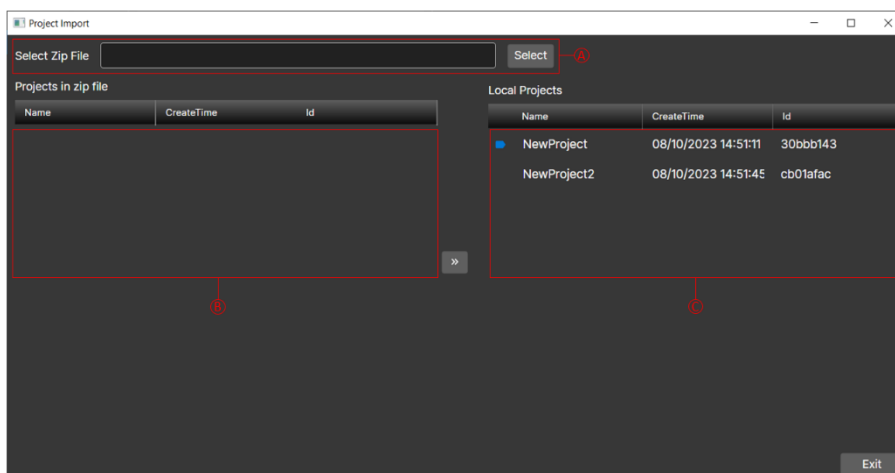
- Save: Saves the content of the current project.
- Save As: Save the project under a different name.
- Export to Template: Saves the project to a template.
- Open project folder: Opens the current project folder.
- Project Export: Users can select single or multiple projects to package and export projects, plugins, and subflows in compressed files.



- A. Existing projects in the local area
- B. Export the finished project
- C. Open the zip folder as shown below. The path is C:\Users\Public\Documents\DMV-IVS.

Name	Date modified	Type
DataList	2023/8/10 下午 04:43	File folder
ExportTemp	2023/8/10 下午 05:14	File folder
FlowchartRepository	2023/6/20 下午 02:50	File folder
Intrusion	2023/3/31 下午 04:30	File folder
Logs	2023/8/10 上午 08:40	File folder
Plugin	2023/7/28 上午 11:36	File folder
Projects	2023/8/10 下午 02:51	File folder
TemplateProjects	2023/8/10 下午 04:43	File folder
DIA_Config.xml	2023/8/10 下午 05:14	XML Document
DIA_Platform.xml	2023/8/10 下午 04:16	XML Document
ExportTemp.zip	2023/8/10 下午 05:14	Compressed (zipp...

- Project Import: Import the DIAVision export zip file into the software.

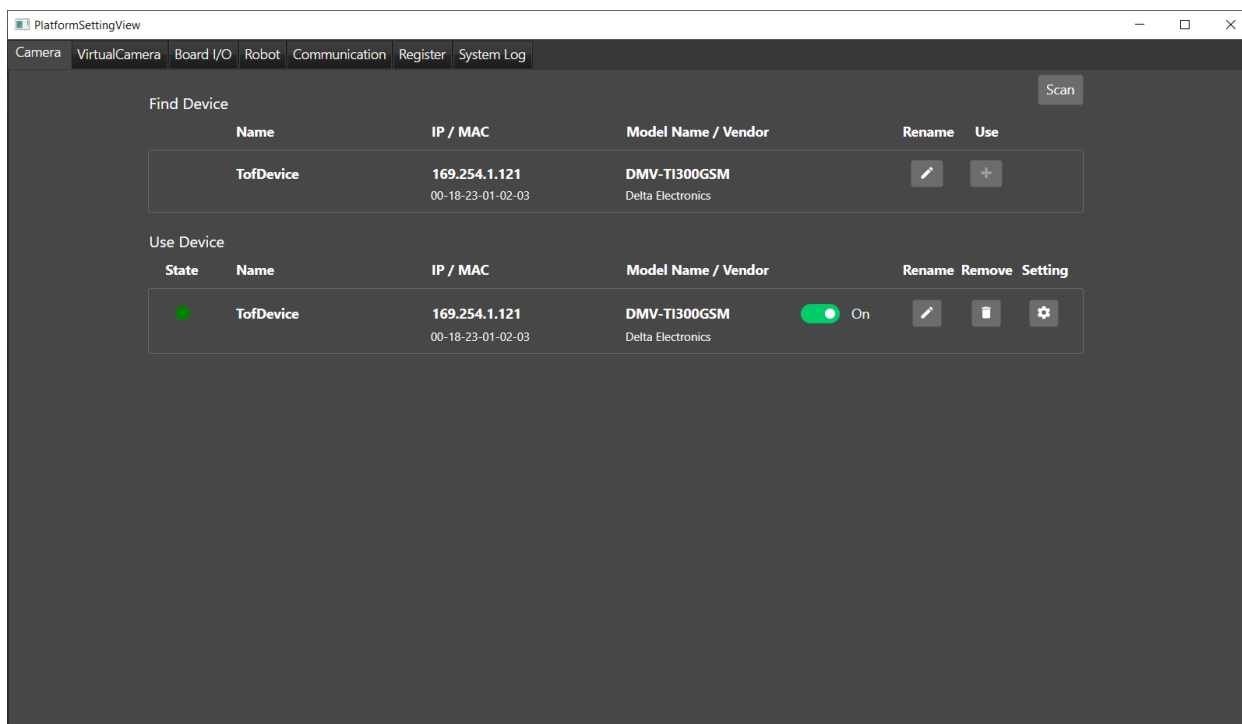


- A. Select the import source
  - B. Export the items in the zip file
  - C. Existing items in the PC
- Close: Close the DIAVision software.

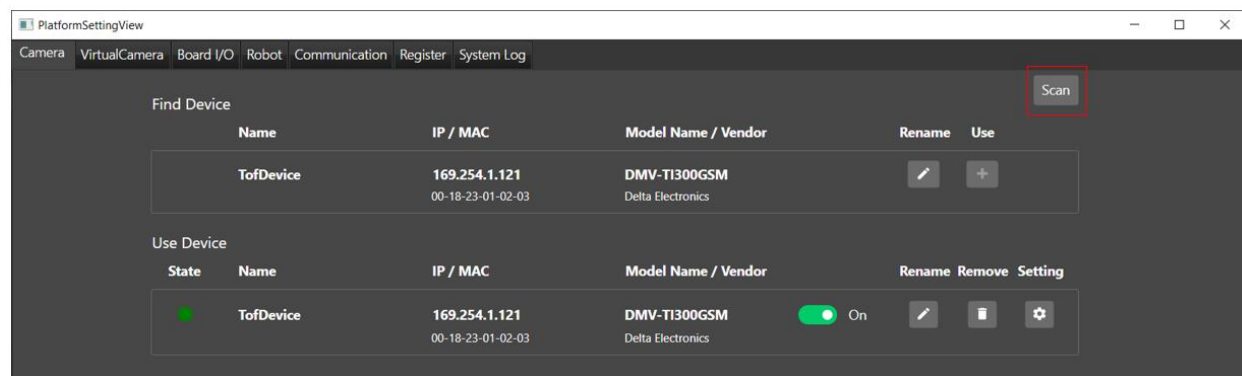
## 3.1.1.2 Manage



### 3.1.1.2.1 Platform Settings

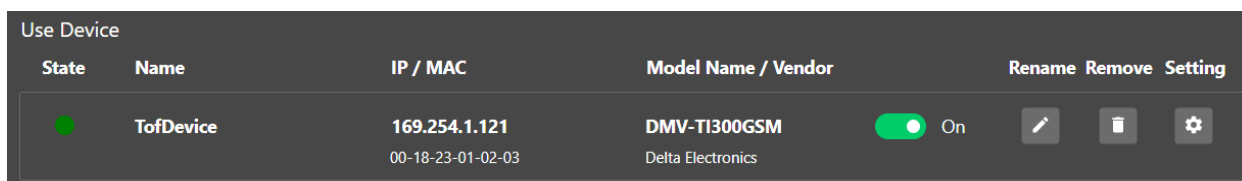
#### 3.1.1.2.1.1 Camera











Users can click on the red area to enter the page and click the "Scan" button in the red area, as shown below.

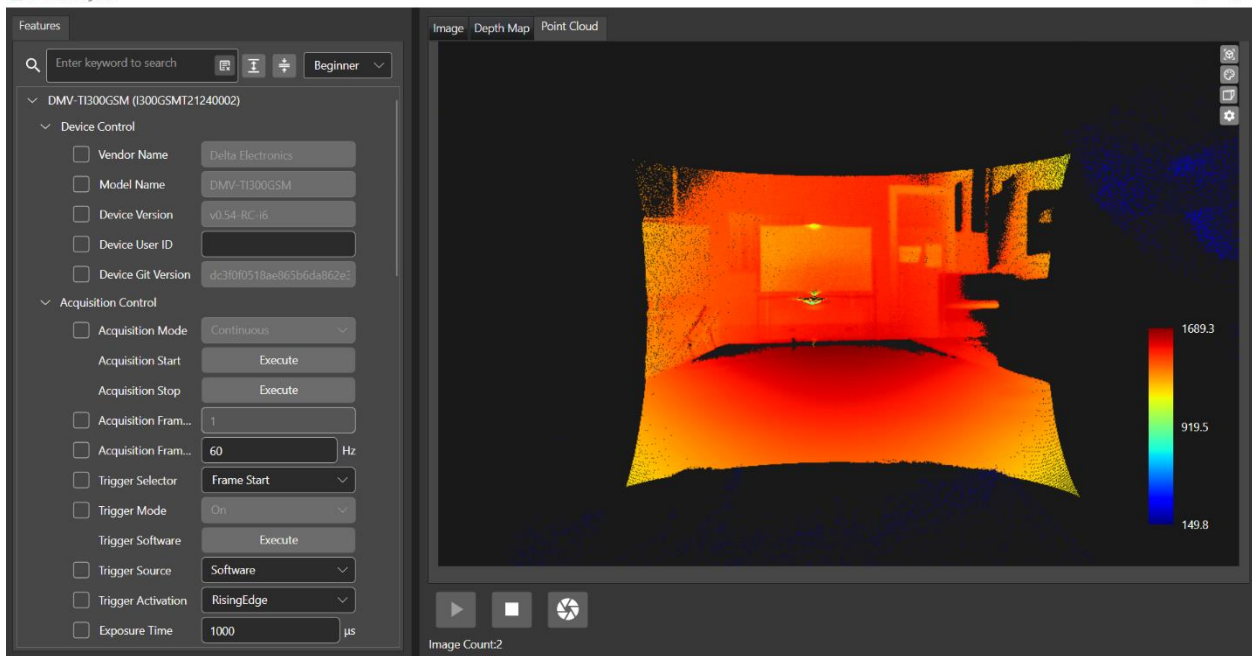


The software lists all connected devices and displays the device Name, MAC, IP, Model Name/ Vendor, and vendor. Users can click  to rename the device. If you want to use the device, click  on it and the software will add the device to the Use Device.

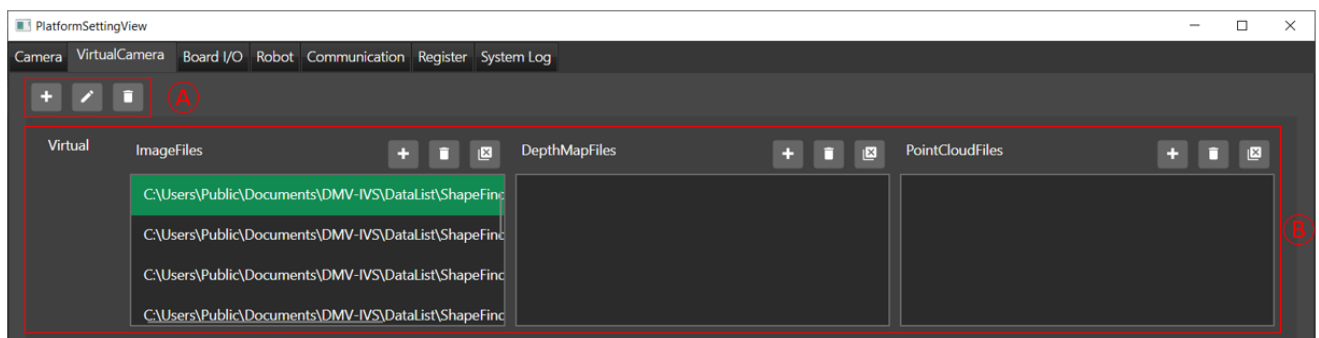


When a device is added to the Use Device, the software automatically connects and displays the device Name, MAC, IP, Model Name/Vendor. If it is operating normally, the status will display , and when the device is abnormal, it will display . The user can decide whether to enable the




device through the switch, and when pressed  , the software will pause the connection, and the status will display  . If the device is enabled again, then tap  to enable the device at this time. When clicked, you  can name the device.  to delete the device. The user can click  to set the parameters of the device, as shown in the following figure. For details on device setup parameters, refer to the user manual for each device.






### 3.1.1.2.1.2 VirtualCamera



#### A. Virtual Camera Configuration

-  New Virtual Camera: A new virtual camera allows users to load images into the virtual camera for testing.
-  Rename: Renames the selected camera.
-  Delete: Deletes the selected virtual camera.

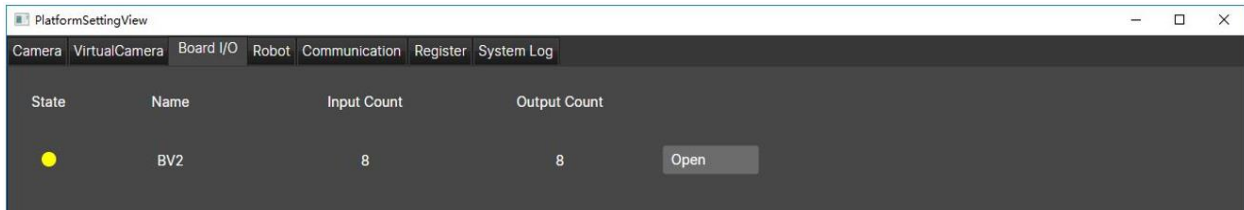
#### B. Virtual Camera Setup

-  New Image: Users can select an input image. However, it should be noted that Image files and DepthMap files only receive png and bmp, while PointCloud only receive pcd format.
-  Delete Selected Images: Deletes the selected images.
-  Delete All: Deletes all input images.
  - ImageFiles: Enter general image (8 bits or 16 bits) images.

- DepthMapFiles: Enter depth images (8 bits or 16 bits) images.
- PointCloudFiles: Enter a point cloud format file. pcd format is currently supported.

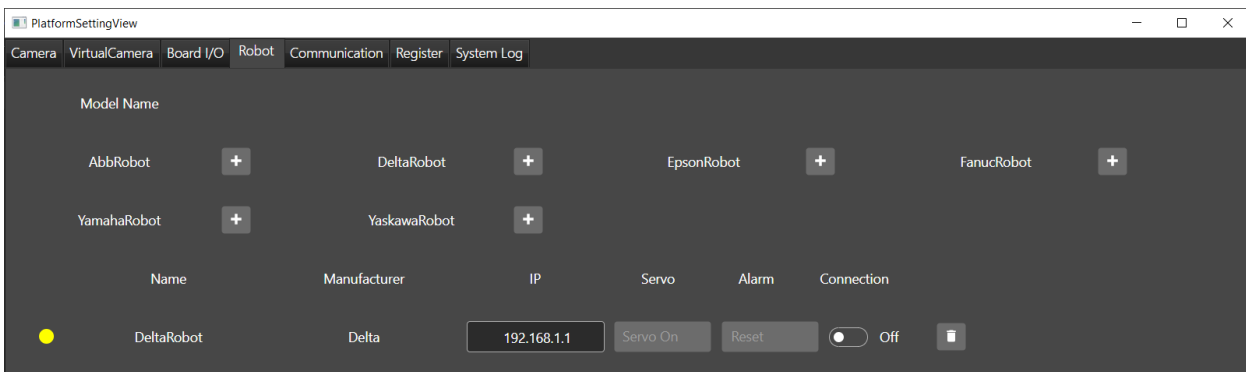
### 3.1.1.2.1.3 Board I/O

Currently the software supports ROSEEK Beaver2 I/O inputs and outputs. ● when State is enabled, and ● for the wait command.



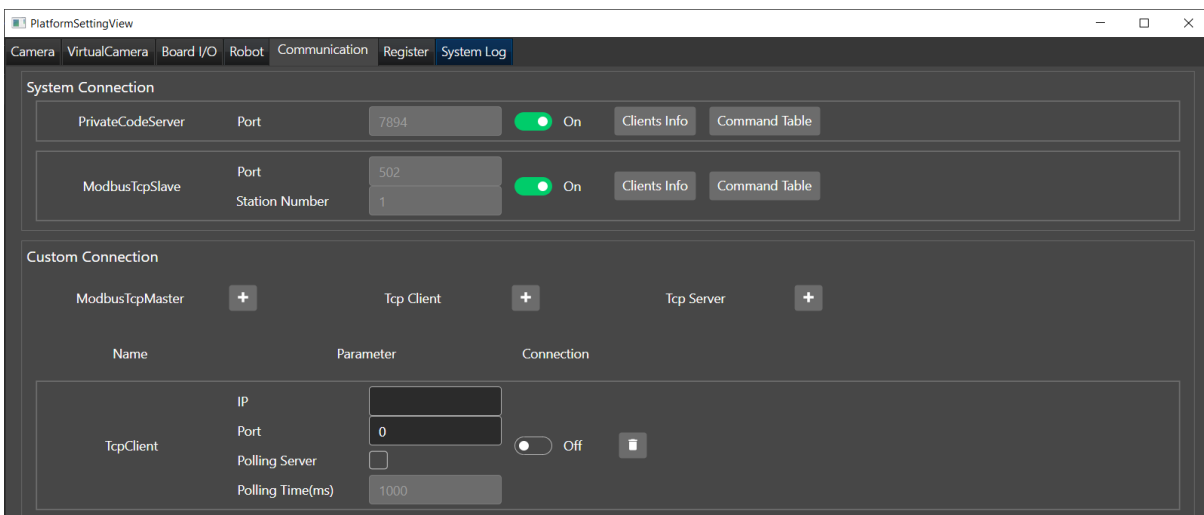
### 3.1.1.2.1.4 Robot

From the Robot page, users can click + of the brand and enter the Robot IP to connect robots. The status ● On is applied when the connection is successful. Users can also start the servo and reset the Robot alarm through this page.



### 3.1.1.2.1.5 Communication

Users can establish communication on the Communication page. It is divided into system connection and custom connection



## System Connection:

- PrivateCodeServer: After the user sets the port  Off, the software will run as the private code server. The host system can read software information as a user. Users can click on the Command Table to query the string, as shown in the following figure.

The screenshot shows a window titled 'PrivateCodeCommandTableView' with two main sections: 'Command Table' and 'Error Table'.

Function	Request	Response	ParameterInfo
Get flowchart status	FlowchartStatus	FlowchartStatus,Busy/Ready	
Flowchart run once	FlowchartRunOnce	FlowchartRunOnce,OK	
Flowchart run continuous	FlowchartRunContinuous	FlowchartRunContinuous,OK	
Flowchart stop	FlowchartStop	FlowchartStop,OK	
Write Register	WriteR,(type),(index),(count),(data1),(data2),(data3)...	WriteR,OK	type: Byte 1, Int 2, Double 3, String 4 index: register start index count: data amount
Read Register	ReadR,(type),(index),(count)	ReadR,(data1),(data2),(data3)...	type: Byte 1, Int 2, Double 3, String 4 index: register start index count: data amount
Get Project Name	PrintName	PrintName (PrintName)	

Error Code	Info
E0001	NoCommand
E0002	SystemBusy
E0003	IllegalParameter
E0004	SystemError
E0005	EmptyResult
E0006	ParameterError
E0007	CommandError
E0008	FlowchartBusy

The user information can be shown on  Clients Info, Client Info as shown in the figure below.

The screenshot shows a window titled 'ClientsInfoView' with a table containing one row of client information.

Client
1 127.0.0.1:58722

- ModbusTcp Slave: After users set the port and station number  Off, the software will run as a ModbusTcp slave. The host system can read or write values to Modbus. Users can click on the Command Table to query the string, as shown in the following figure.

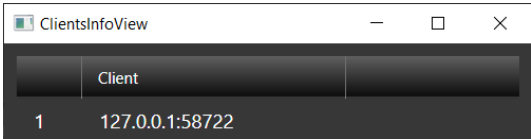
The screenshot shows a window titled 'ModbusTcpFunctionTableView' with two main sections: 'Command Table' and 'Error Table'.

Address	Function	Write Description	Read Description
0x0000	InputRegister	Write register table 0x0000-0x0FFF	Read register table 0x0000-0x0FFF
0x1000	FlowchartRunOnce	Flowchart run once data:1	Get flowchart status data: 0 Ready, 1 Busy
0x1002	FlowchartRunContinuous	Flowchart run continuous data:1	Get flowchart status data: 0 Ready, 1 Busy
0x1004	FlowchartStop	Flowchart stop data:1	Get flowchart status data: 0 Ready, 1 Busy
0x1100	LoadProject	Load project data: The first four digits of the project id as the hex number	Get project ID
0x1108	RefreshProjects	Scan the project folder and refresh the project information list data:1	
0x110A	SystemsReady		Get system IsReady status

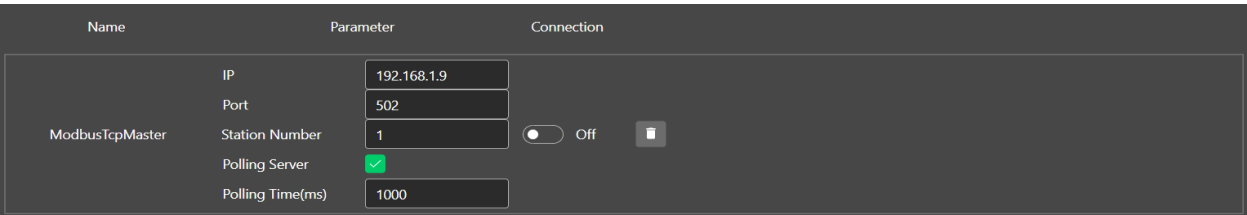
Error Code	Info
0x0000	NoError
0x0001	IllegalFunction
0x0002	IllegalAddress
0x0003	IllegalData
0x0004	IllegalCommand
0x0005	SystemBusy
0x0006	IllegalQuantity
0x0007	IllegalParameter

The user information can be shown on **Clients Info**, Client Info as shown in the figure below.

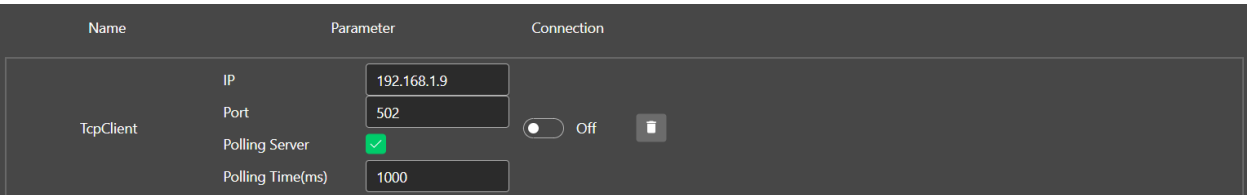


Custom Connection

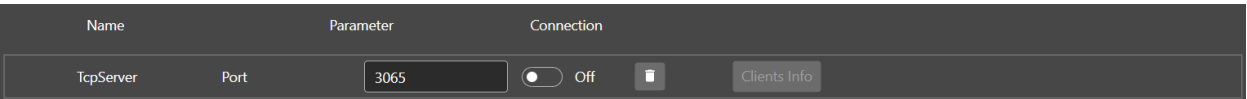
- ModbusTcpMaster:** The software is set as the master, and the software can write and read values to the slaves. Users need to set the slave IP, Port, and Station Number. Connecting when user turns on **Off**. When the Polling Server option is enabled, the software will continue to access the application according to the Polling Time, which is commonly used to continuously confirm the connection to the application with the server. Users can click **Remove** button to remove the connection.



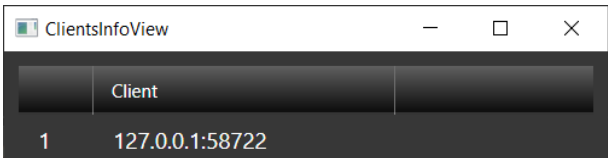
- TcpClient:** The software is set as the party that initiates the TCP connection request, and once the connection is established, users can send data to the server through the connection and wait for the server's response. Users need to set the server IP and Port. Connecting when users turn on **Off**. When the Polling Server option is enabled, the software will continue to access the software according to the Polling Time. Users can click **Remove** button to remove the connection.



- TcpServer:** After the users set the Port, **Off** the software will run as a private code server. The host system can read the software information as a user. Users can click **Remove** button to remove the connection.



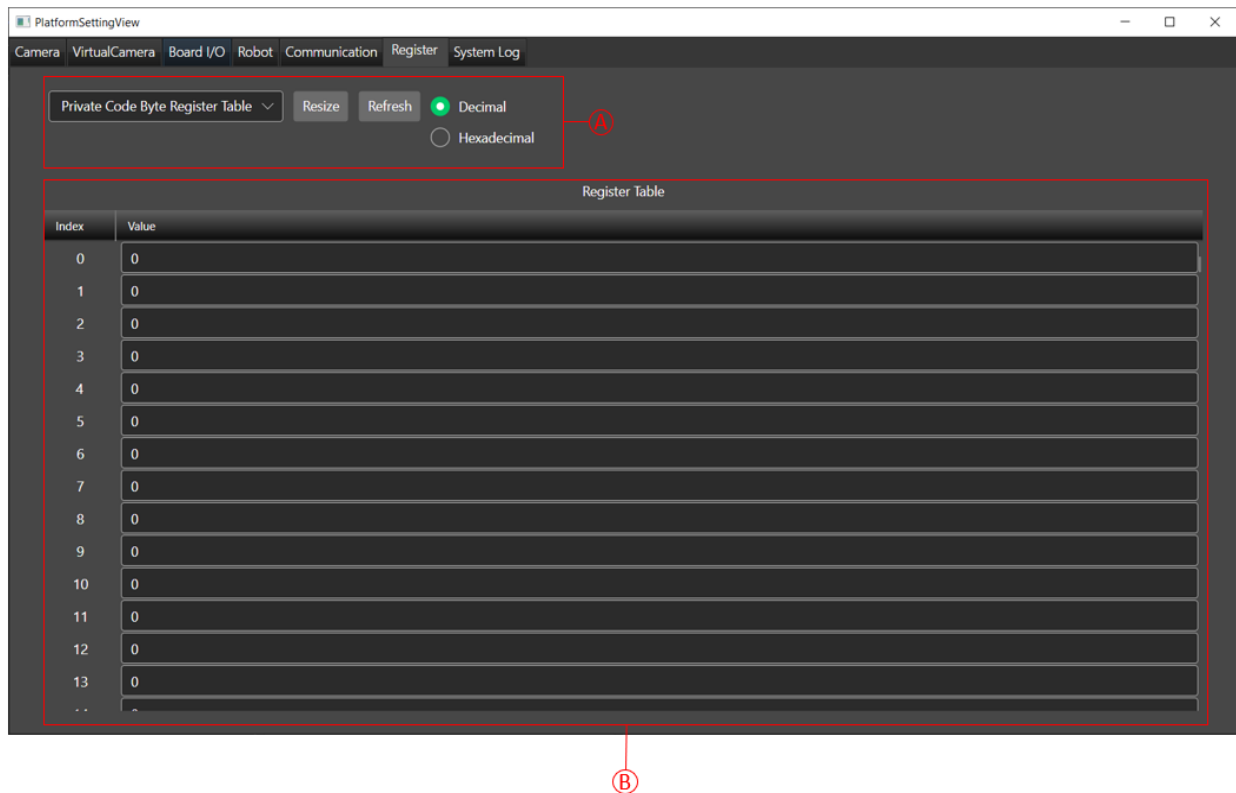
The user information can be shown on **Clients Info**, Client Info as shown in the figure below.





## 3.1.1.2.1.6 Register

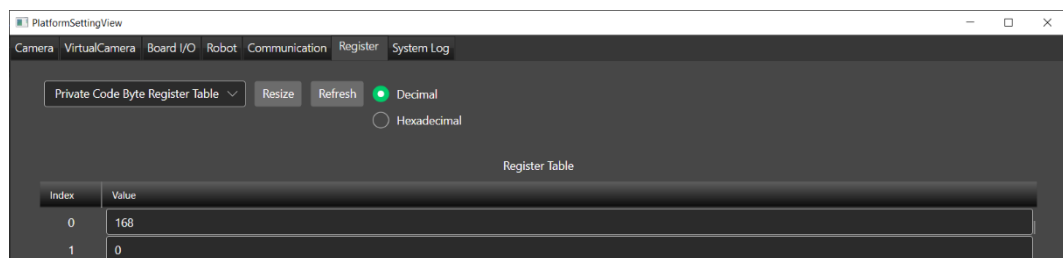
Users can check the process data results with the Register display values.



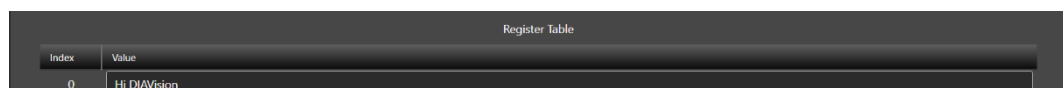
## A. List of Information

## ◆ List:

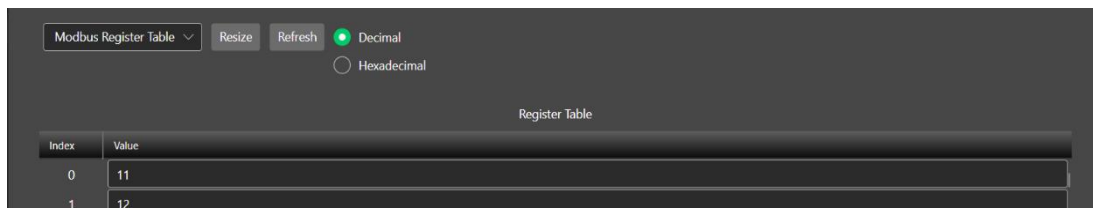
- Private Code Byte Register Table: Used to observe the bit data received at the register index position. Users can switch the results between decimal or hexadecimal.



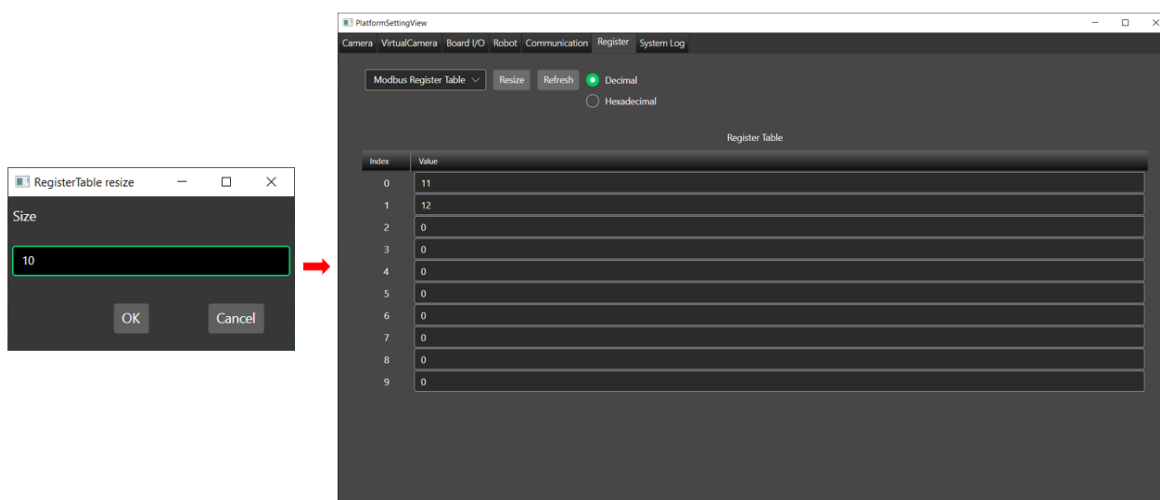
- Private Code Int Register Table: Used to observe the integer data received at the register index position. Users can switch the results between decimal or hexadecimal.
- Private Code Double Register Table: Used to observe the floating-point data received at the register index position. Users can switch the results between decimal or hexadecimal.
- Private Code String Register Table: Used to observe the string type data received at the register index position.



- Modbus Register Table: Used to observe the data received at the register index position. The modbus data format is double word. Users can switch the results between decimal or hexadecimal.



- ◆ Resize: Users can set the number of display numbers according to their needs, as shown in the following figure.



- ◆ Refresh: Re-reads the register data.
- ◆ Decimal: Register data is displayed in decimal format.
- ◆ Hexadecimal: Register data is displayed in hexadecimal format.

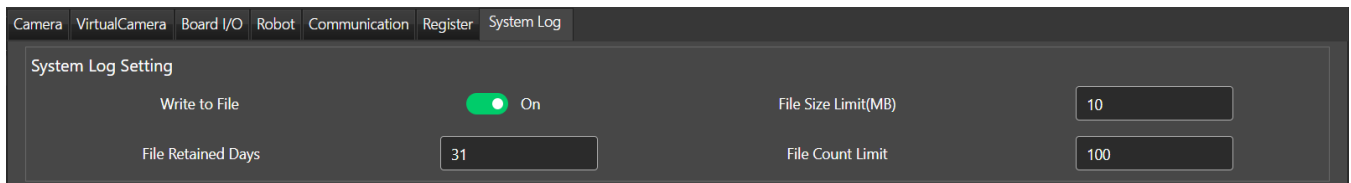
## B. Register Table

- ◆ The register data is displayed in a list, and users can check the value from the list. The index value of register display can be set by "Resize" and up to 10,000 registers can be displayed.



### 3.1.1.2.1.7 System Log

System Log: Records system level information for further diagnostic.



- ◆ Write to file: Save the logs to text files.

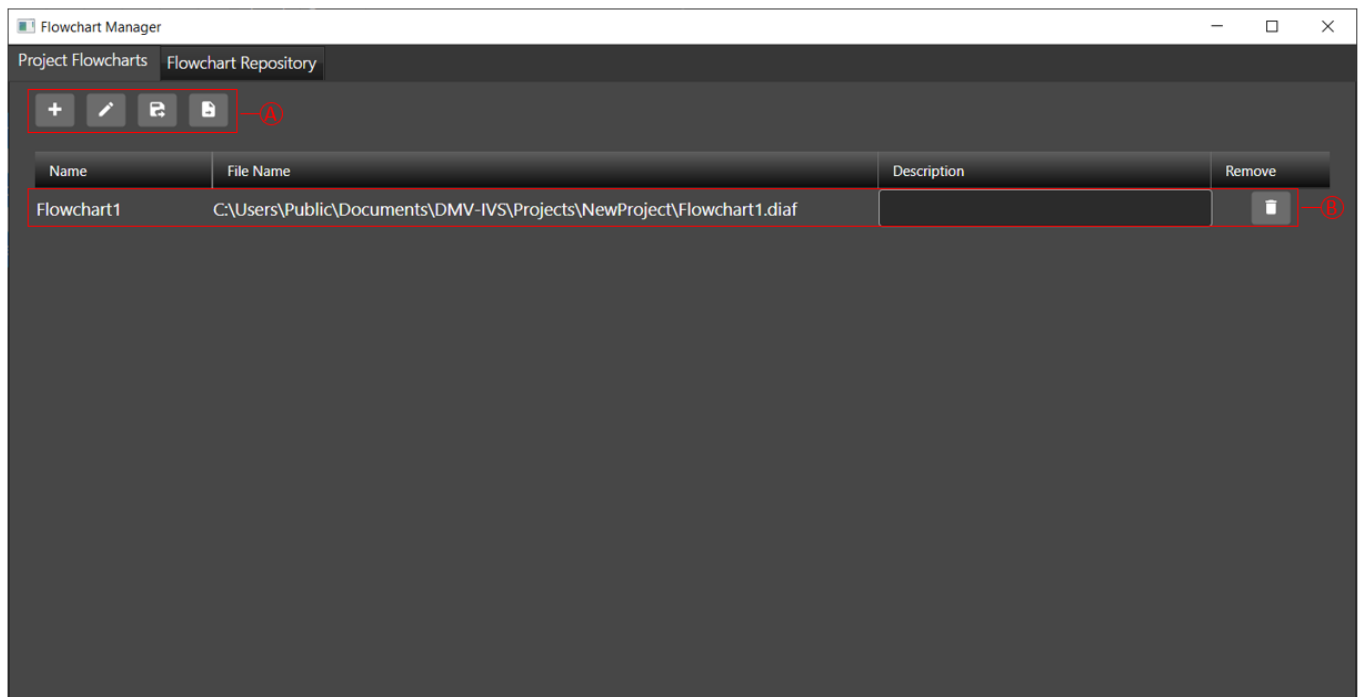
C : \Users\Public\Documents\DMV-IVS\Logs

- ◆ File Retained Days: Sets the number of days to save a text file.
- ◆ File Size Limit (MB): Limits the size of a single text file. When the file size is larger than the size limit, a new text file will be created.
- ◆ File Count Limit: Limits the number of text files. When the number of text files exceeds the upper limit, the system will overwrite the oldest text file with the latest one.

### 3.1.1.2.2 Flowchart Manager

Users can create subflowchart for reuse.

- Project Flowcharts:




#### A. Function icons

- + A new subflow is added to the project, which can only be used in a single project.
- ✎ Renames the subflow.
- 📄 Copies a subflow into the flowchart repository on behalf of other subsequent projects that can reference the subflow.
- 📄 Moves a subflow directly into the flowchart repository.

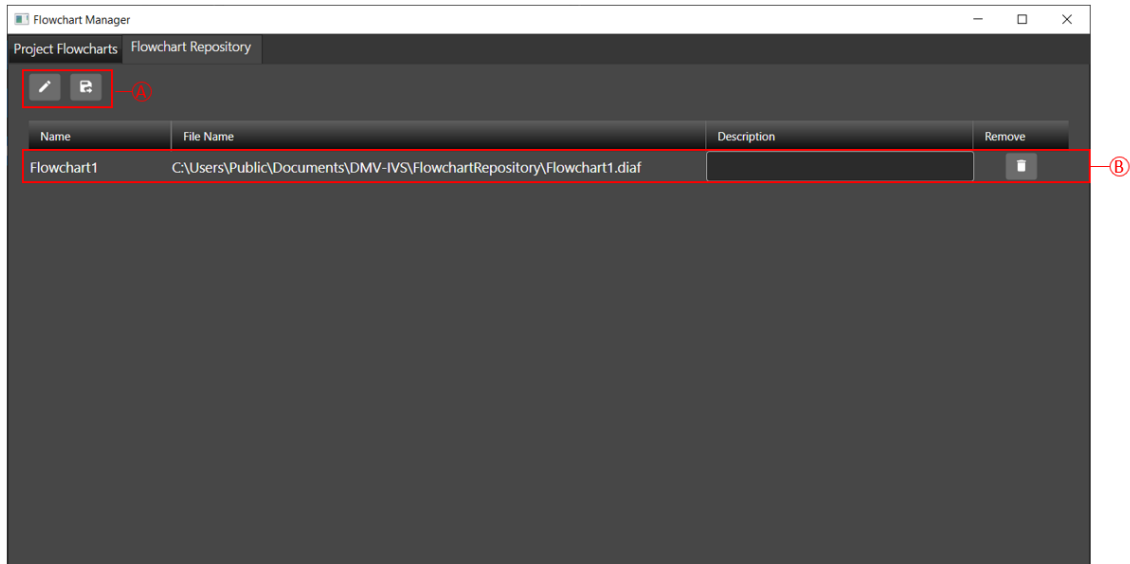
### B. Subflow management

Displays the subflow that are currently available in the project, including the Name, File Name, and Description.


 Users can click the icon to delete the sub-process.

- **Flowchart Repository:**

The created subflowcharts can be reused in different projects.



### A. Function icons

 Renames the subflow.

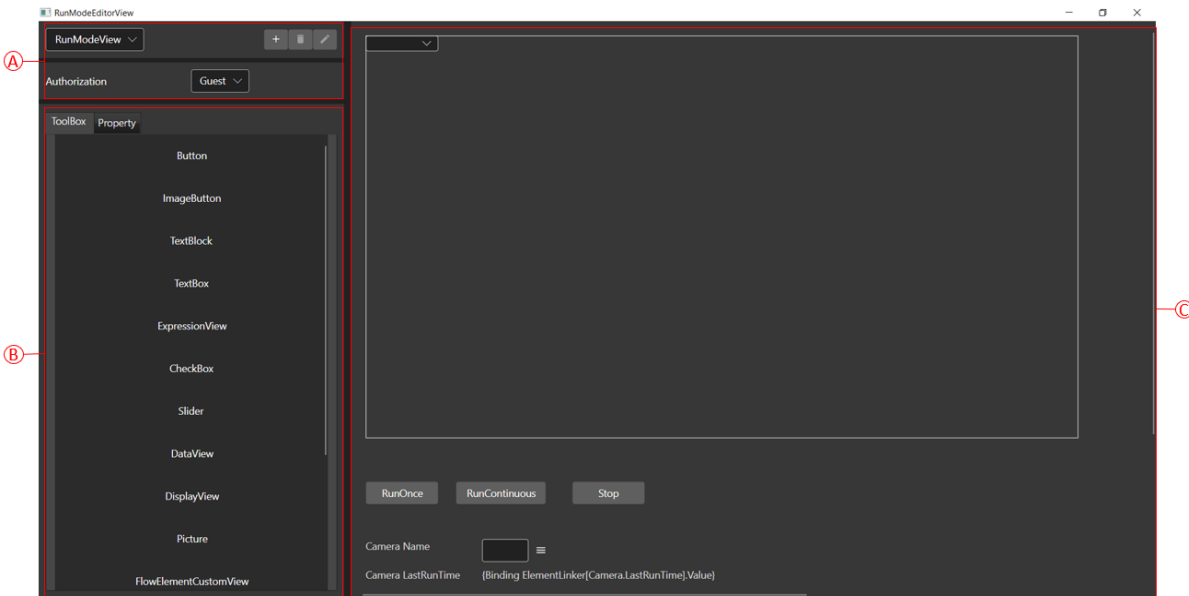
 Copies the subflow of the flowchart repository to the project.

### B. Flowchart Repository Management

Displays the subflow that are currently available in the flowchart repository, including Names, File Name, and Descriptions.

### 3.1.1.2.3 Run mode Editor

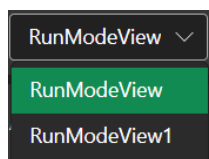
Users can define layout of run mode, including buttons, labels, display views, and icons.



#### A. Page and Permission Settings

RunModeView is the running screen.

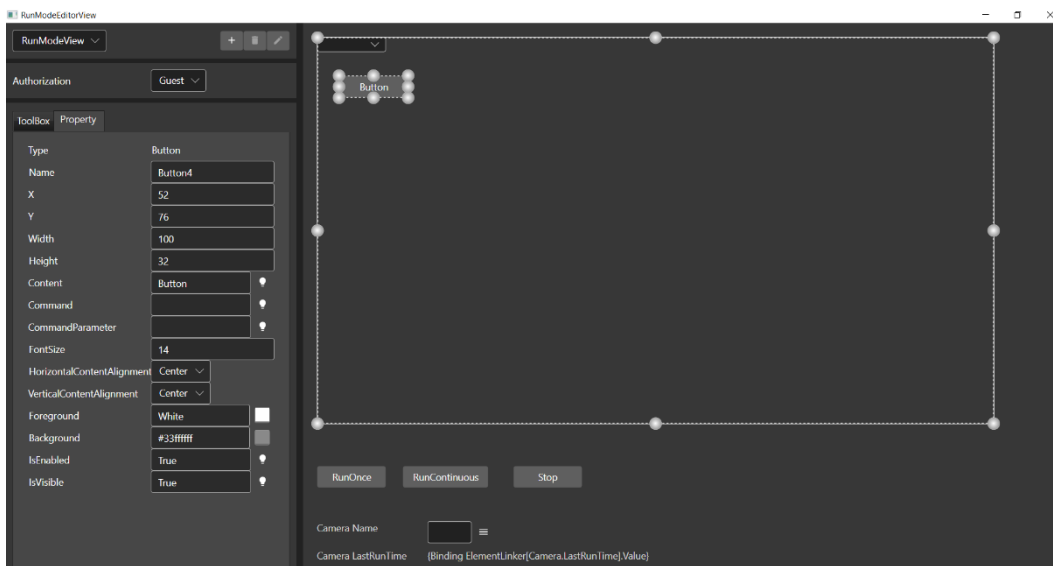
- +** New Page: Adds a new page of run mode. The page will be displayed in the list on the left, as shown below.



- 🗑️** Delete Page: Deletes the currently running page.
- ✏️** Renames RunModeView: Renames the current RunModeView.

#### B. Toolbox and Property

Toolbox: Users can double-click the toolbox component to add objects to the operation screen C, as shown in the following figure.



The following table describes the attributes of each component.

Control item type	attribute	Description of the property
Button	Name	The name of the control
	X	Control item start position X coordinate
	Y	The control item starts at the Y coordinate
	Width	Control the width of the item
	Height	Control item height
	Content	content
	Command	Controls the command executed by the item
	CommandParameter	The parameters required for the command to be executed
	FontSize	Font size
	HorizontalContentAlignment	Content text is horizontally aligned
	VerticalContentAlignment	Content text is vertically aligned
	Foreground	Foreground color
	Background	Background color
	IsEnabled	Whether the control item is enabled
IsVisible	Control whether the item is displayed	
ImageButton	Name	The name of the control
	X	Control item start position X coordinate
	Y	The control item starts at the Y coordinate
	Width	Control the width of the item
	Height	Control item height
	ImagePath	Image path
	Text	Content text
	FontSize	Size
	TextAlignment	Text alignment
	Command	Controls the command executed by the item
	CommandParameter	The parameters required for the command to be executed
	Foreground	Foreground color
	Background	Background color
	IsEnabled	Whether the control item is enabled
IsVisible	Control whether the item is displayed	
TextBlock	Name	The name of the control
	X	Control item start position X coordinate
	Y	The control item starts at the Y coordinate
	Width	Control the width of the item
	Height	Control item height
	Text	Content text
	FontSize	Size
	TextAlignment	Text alignment
	Background	Background color

	IsEnabled	Whether the control item is enabled
	IsVisible	Control whether the item is displayed
Textbox	Name	The name of the control
	X	Control item start position X coordinate
	Y	The control item starts at the Y coordinate
	Width	Control the width of the item
	Height	Control item height
	Text	Content text
	FontSize	Font Size
	TextAlignment	Text alignment
	Foreground	Foreground color
	Background	Background color
	IsEnabled	Whether the control item is enabled
	IsVisible	Control whether the item is displayed
ExpressionView	Name	The name of the control
	X	Control item start position X coordinate
	Y	The control item starts at the Y coordinate
	Width	Control the width of the item
	Height	Control item height
	Expression	Expression content
	IsEnabled	Whether the control item is enabled
	IsVisible	Control whether the item is displayed
CheckBox	Name	The name of the control
	X	Control item start position X coordinate
	Y	The control item starts at the Y coordinate
	Width	Control the width of the item
	Height	Control item height
	Content	content
	IsChecked	Whether this option is selected or not, this parameter is selected by default
	FontSize	Size
	HorizontalContentAlignment	Content text is horizontally aligned
	VerticalContentAlignment	Content text is vertically aligned
	Foreground	Foreground color
	Background	Background color
	IsEnabled	Whether the control item is enabled
IsVisible	Control whether the item is displayed	
Slider	Name	The name of the control
	X	Control item start position X coordinate
	Y	The control item starts at the Y coordinate
	Width	Control the width of the item
	Height	Control item height
	Value	Slider defaults
	Minimum	Minimum value

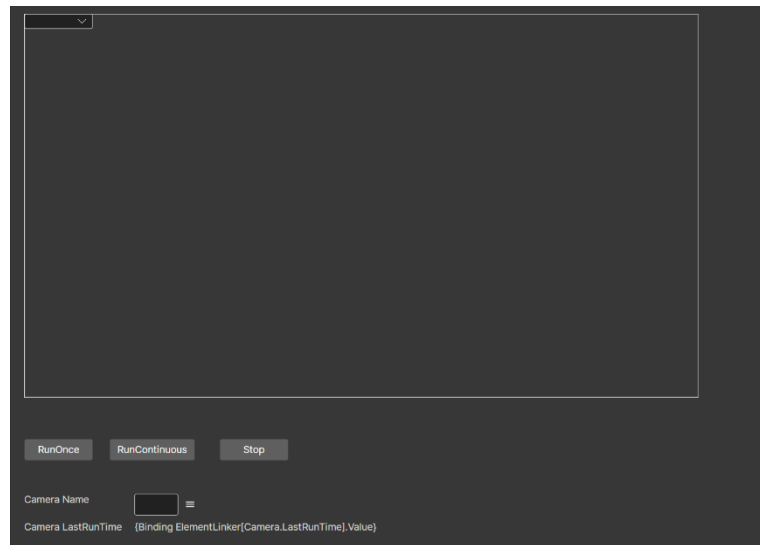
	Maximum	Maximum value
	TickPlacement	Whether or not to display the scale
	TickFrequency	Tick intervals
	FontSize	Size
	Foreground	Foreground color
	Background	Background color
	IsEnabled	Whether the control item is enabled
	IsVisible	Control whether the item is displayed
DataView	Name	The name of the control
	X	Control item start position X coordinate
	Y	The control item starts at the Y coordinate
	Width	Control the width of the item
	Height	Control item height
	DataContext	Screen data binding
	IsEnabled	Whether the control item is enabled
	IsVisible	Control whether the item is displayed
DisplayView	Name	The name of the control
	X	Control item start position X coordinate
	Y	The control item starts at the Y coordinate
	Width	Control the width of the item
	Height	Control item height
	IsEnabled	Whether the control item is enabled
	IsVisible	Control whether the item is displayed
	DefaultImage	Default image
	ImageList	Add a list of selected images
	AnnotationList	Edit the list of picture features
Picture	Name	The name of the control
	X	Control item start position X coordinate
	Y	The control item starts at the Y coordinate
	Width	Control the width of the item
	Height	Control item height
	ImagePath	Image path
	IsEnabled	Whether the control item is enabled
	IsVisible	Control whether the item is displayed
FlowElementCustomView	Name	The name of the control
	X	Control item start position X coordinate
	Y	The control item starts at the Y coordinate
	Width	Control the width of the item
	Height	Control item height
	DataContext	Screen data binding
	IsEnabled	Whether the control item is enabled
	IsVisible	Control whether the item is displayed
Border	Name	The name of the control
	X	Control item start position X coordinate



	Y	The control item starts at the Y coordinate
	Width	Control the width of the item
	Height	Control item height
	BorderThickness	Border width
	CornerRadius	Border fillet radius
	BorderBrush	Border color
	Background	Background color
	IsEnabled	Whether the control item is enabled
	IsVisible	Control whether the item is displayed
Expander	Name	The name of the control
	X	Control item start position X coordinate
	Y	The control item starts at the Y coordinate
	Width	Control the width of the item
	Height	Control item height
	Header	Title text
	IsExpanded	Control whether the item is expanded
	IsEnabled	Whether the control item is enabled
	IsVisible	Control whether the item is displayed
TabControl	Name	The name of the control
	X	Control item start position X coordinate
	Y	The control item starts at the Y coordinate
	Width	Control the width of the item
	Height	Control item height
	SelectedIndex	Preset labels
	IsEnabled	Whether the control item is enabled
	IsVisible	Control whether the item is displayed
	Items	Edit tags, name, add, and delete tags
RobotPanel	Name	The name of the control
	X	Control item start position X coordinate
	Y	The control item starts at the Y coordinate
	Width	Control the width of the item
	Height	Control item height
	RobotName	Arm name
	IsSixAxis	Whether it is a six-axis arm
	IsEnabled	Whether the control item is enabled
	IsVisible	Control whether the item is displayed

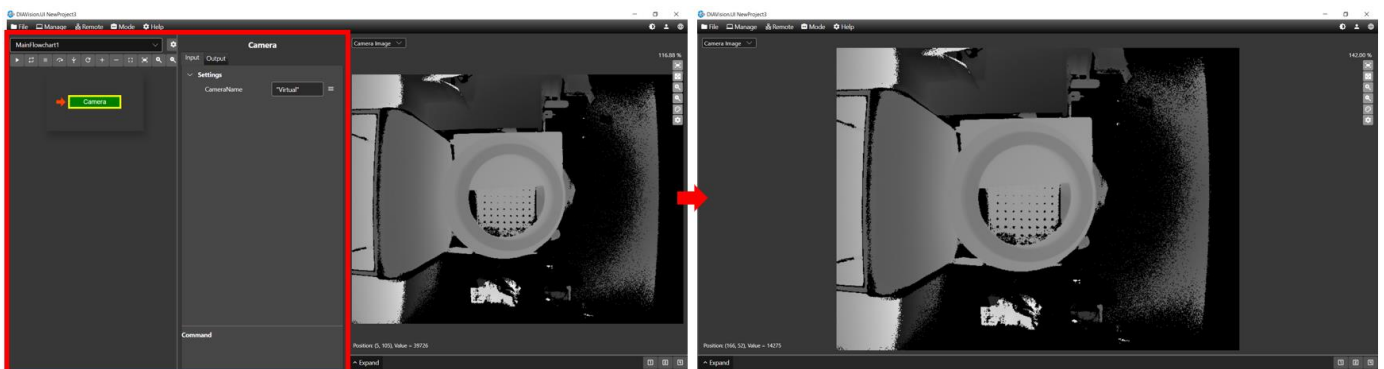
### C. Run mode page

Users can add components to the screen by double-clicking on the tool, as shown in the following figure.



#### 3.1.1.2.4 Show/Hide Flowchart

When users select Show/Hide Flowchart, as shown below, the red area on the left side will be hidden, and only the image will be displayed on the screen.

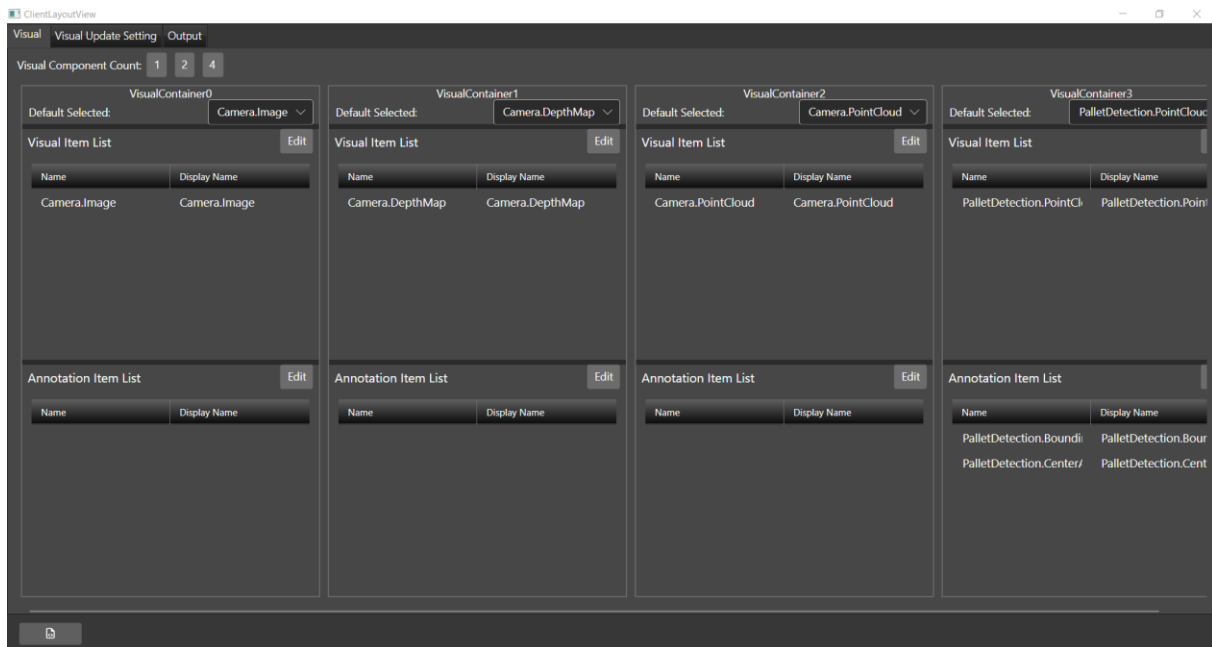






### 3.1.1.3 Remote

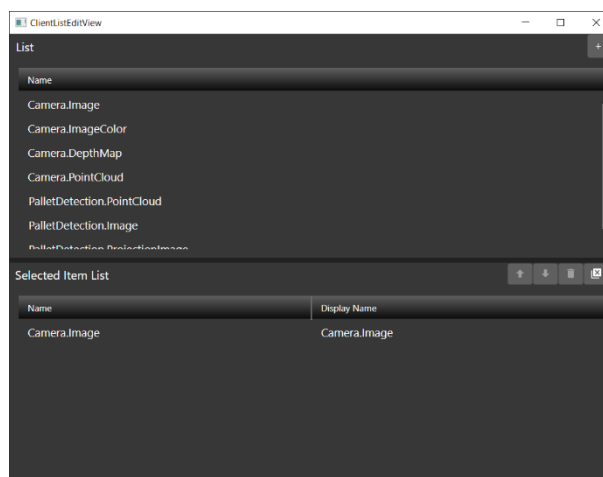
#### 3.1.1.3.1 Client Layout





This feature places the project on a camera with an CPU, such as a Delta ToF camera. Users can monitor and adjust parameters on the browser.

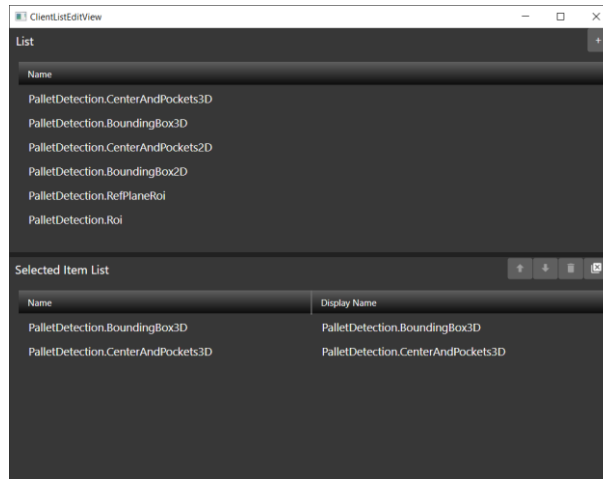
##### ■ Visual



- ◆ Visual Component Count: Users can set how many split images can be displayed on the browser, you can choose 1, 2 or 4 images.
- ◆ Default Selected: This item sets the default screen image for users when entering the browser. However, it should be noted that the user needs to set up the image list first. Once completed, the default selection image will be brought in.
- ◆ Visual Item List: This tab is an image item that is set to be viewed by users in the browser. Users simply clicks "Edit" in the image list to start the setup. If users click the item to be added to the list, the portlet will be added to the Selected Item List. Users can  adjust the order of the list  by deleting a single selected item,  and  to clear the entire list of selected items.



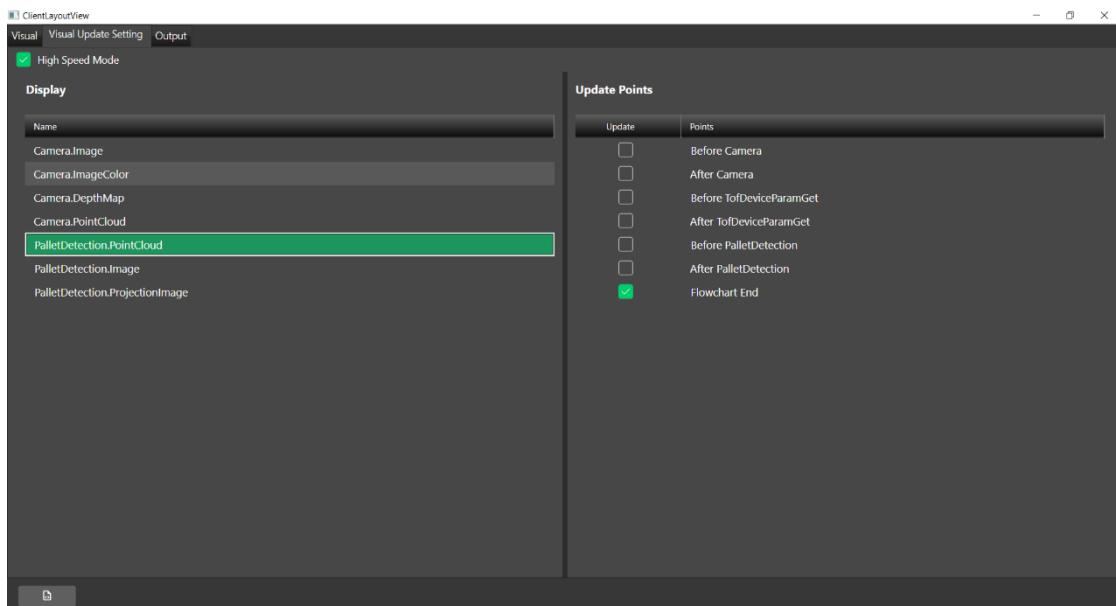
- ◆ Annotation Item List: If the component has feature images, users can add feature screens to this page. The way to add is to click “Edit” to start setting. Users can click the item to add in the list, and then the component will be added to the Selected Item List. Users can  adjust the order of the list  by deleting a single selected item,  and  to clear the entire list of selected items.



### ■ Visual Update Setting

Users can set the time point at which the image is updated. When users click the Display portlet on the left tab, users can set the update point for the right pagination portlet.

- ◆ High Speed Mode: When the update speed is insufficient after selecting an item, the software will discard the old image and update only the latest image.

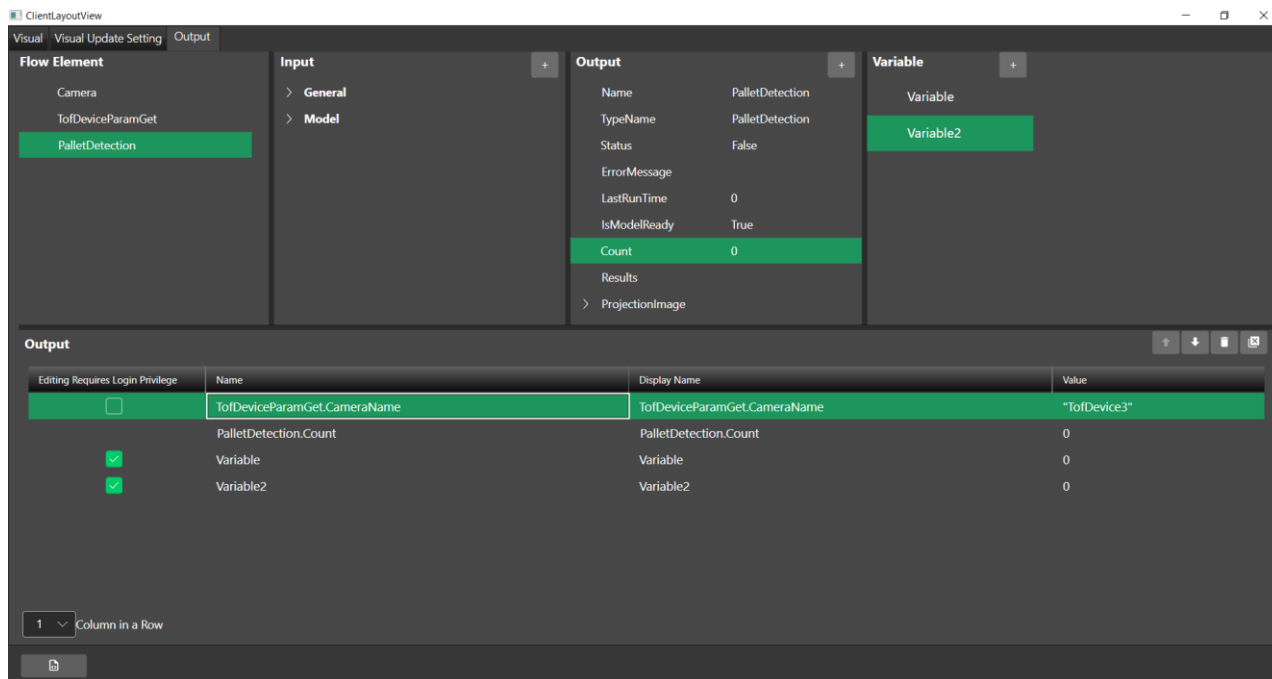


### ■ Output

Users can add parameters to the browser on the output page. Contains parameters and test results as follows.

Users can select the component to add to the process component, add the output or input parameters of the component. When the input parameters are added and the permission to use

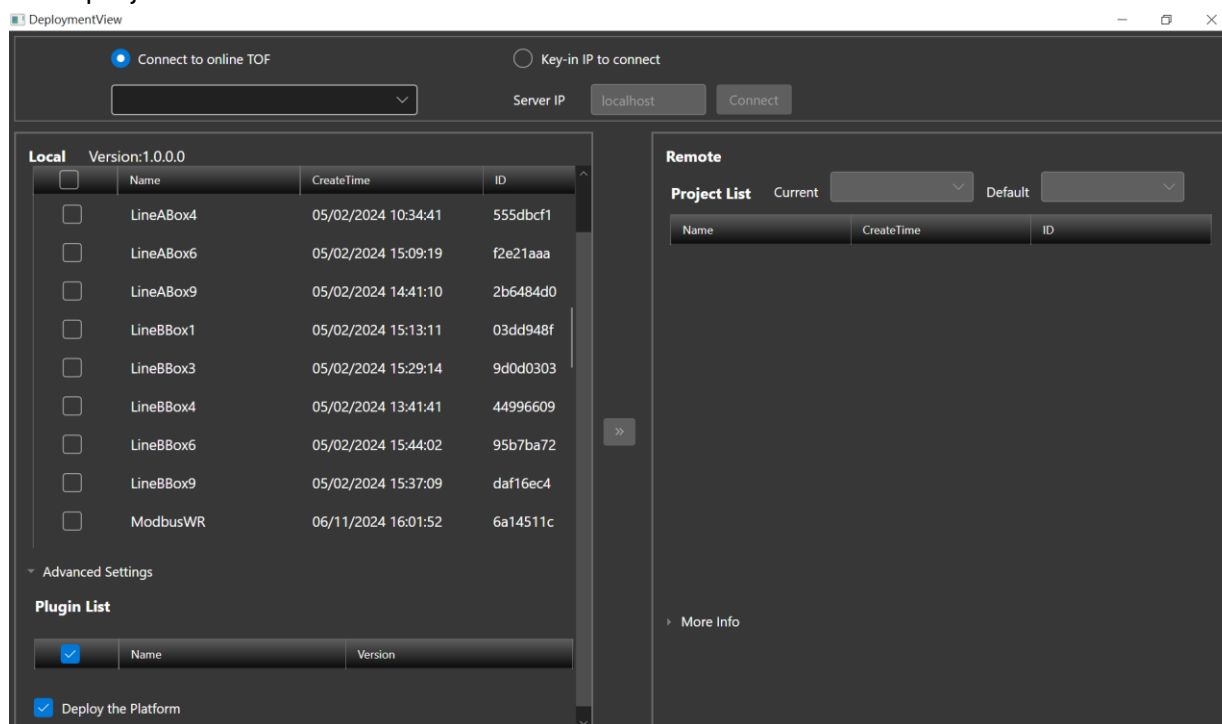
the output table is checked, it means that the user can modify the component conditions in the browser, and the output is to display the execution result of the device in the browser, and the same is true for the variables.

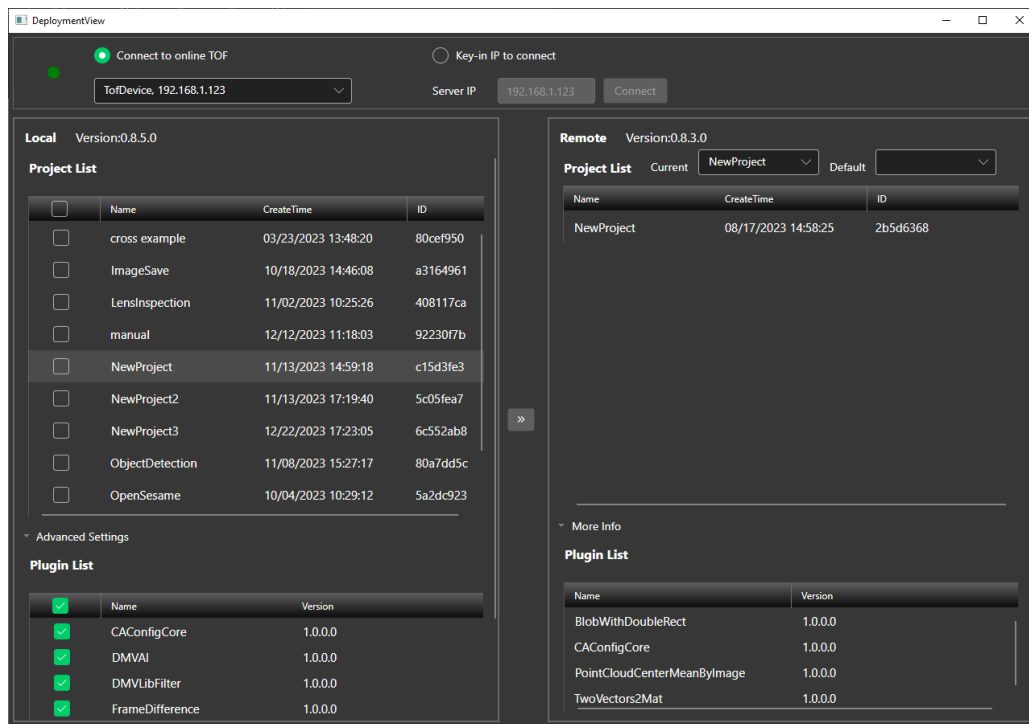


When the setting is completed, you must click  at the bottom left corner of the page to generate a file to complete the setting interface.

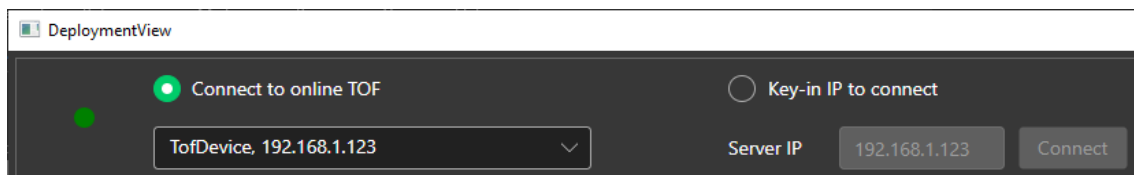
### 3.1.1.3.2 Remote Manager

Uploads projects from the software to the device.

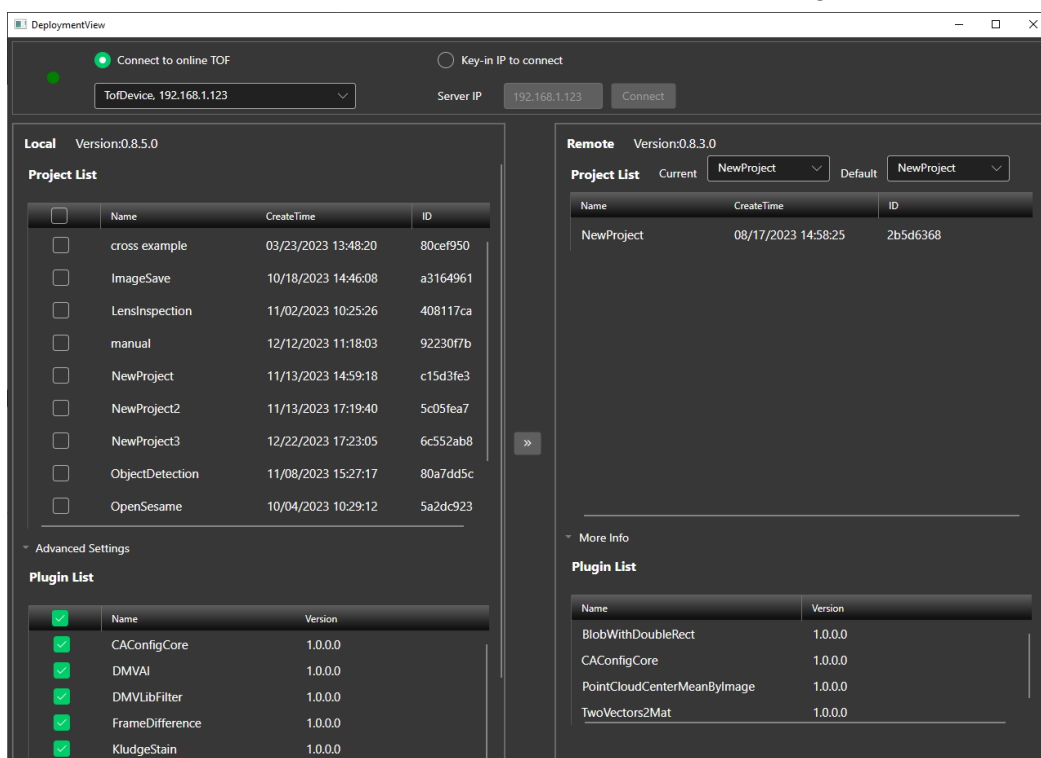




- Connect to online TOF: When users select this item, the software will automatically scan for connected TOF devices. Use this to select from the menu.

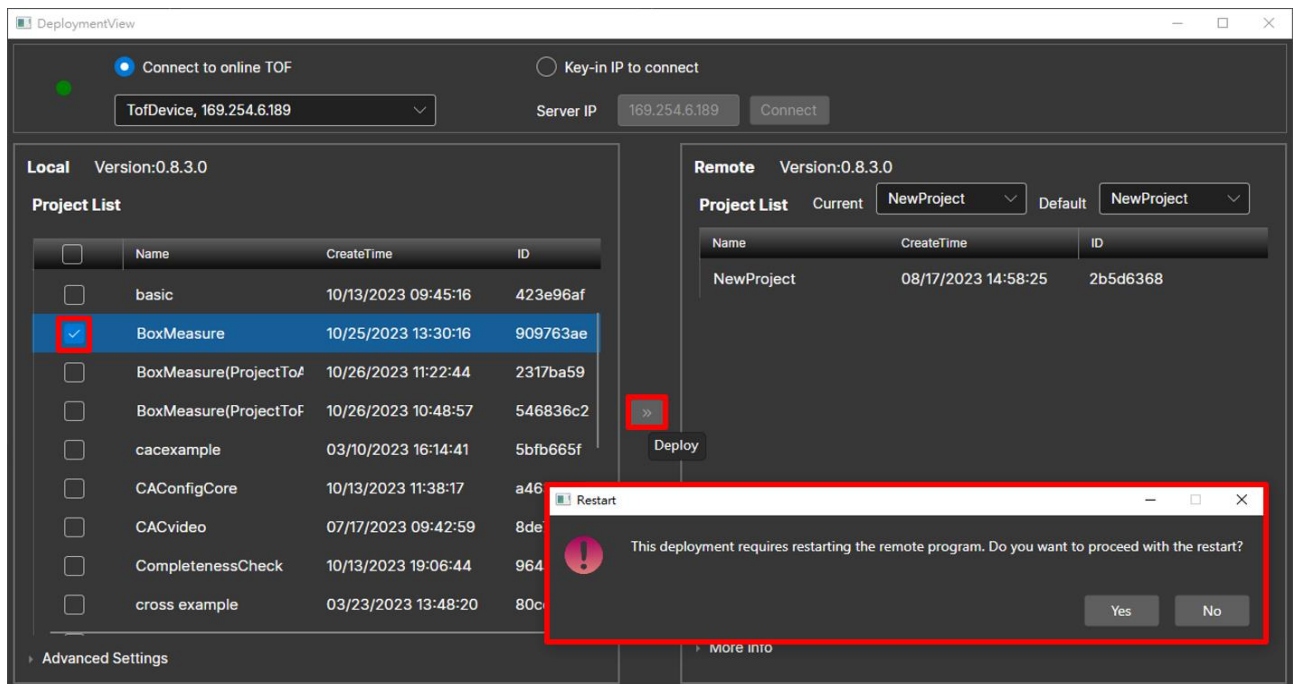


- Key-in IP connect: Users can connect to the device by entering the server IP address.

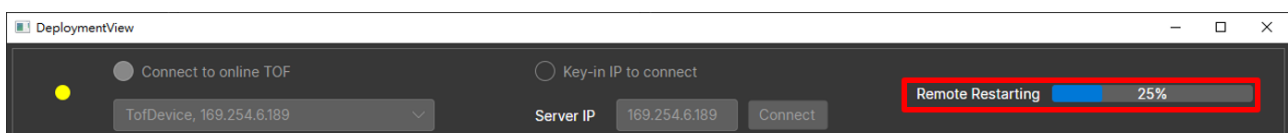


- Local:
  - ◆ Project List: Displays all projects in the PC that also can be selected and uploaded to other devices.
  - ◆ Advanced Setting: Shows the plug-ins available on the PC that also can be checked before uploading to other devices.
  - ◆ Deploy Platform (Plugin List): When users select a project, the representative will upload the platform settings together.
- Remote:
  - ◆ Current: The project currently being executed on the device.
  - ◆ Default: The items that are executed on the device after it is turned on.
  - ◆ Project List: Displays the projects on the remote device.
  - ◆ Plugin List: Displays the plug-in components in the remote device.

After selecting the project, users click Deploy, as shown in the following figure.

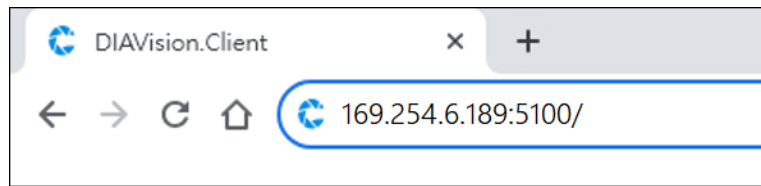


A prompt message will appear in the software, and the light in the upper left corner will flash yellow after clicking OK, and the progress of uploading and restarting will be displayed in the upper right corner.

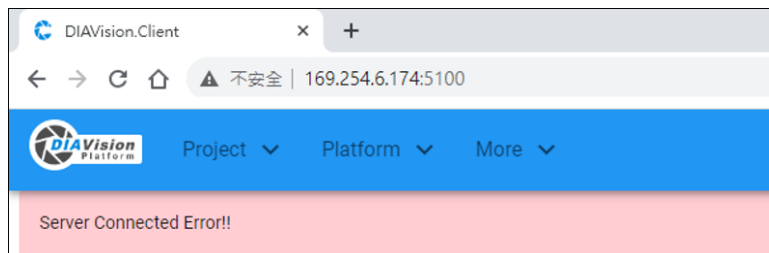


After all the processes are completed, the following instructions will be given on how to check the items in the device.

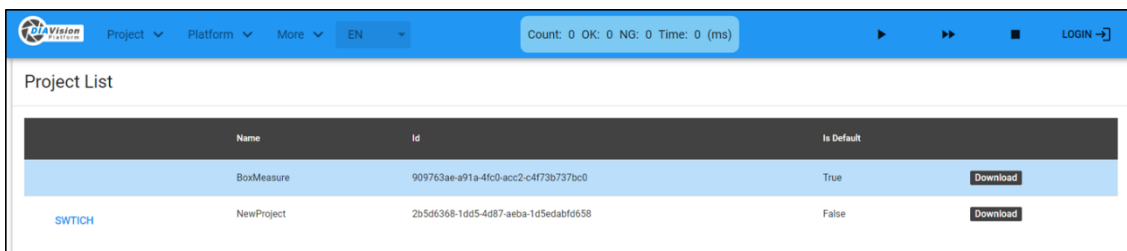
- Opens a web browser and enter the device IP and port number 5100 in the URL field, as follows.



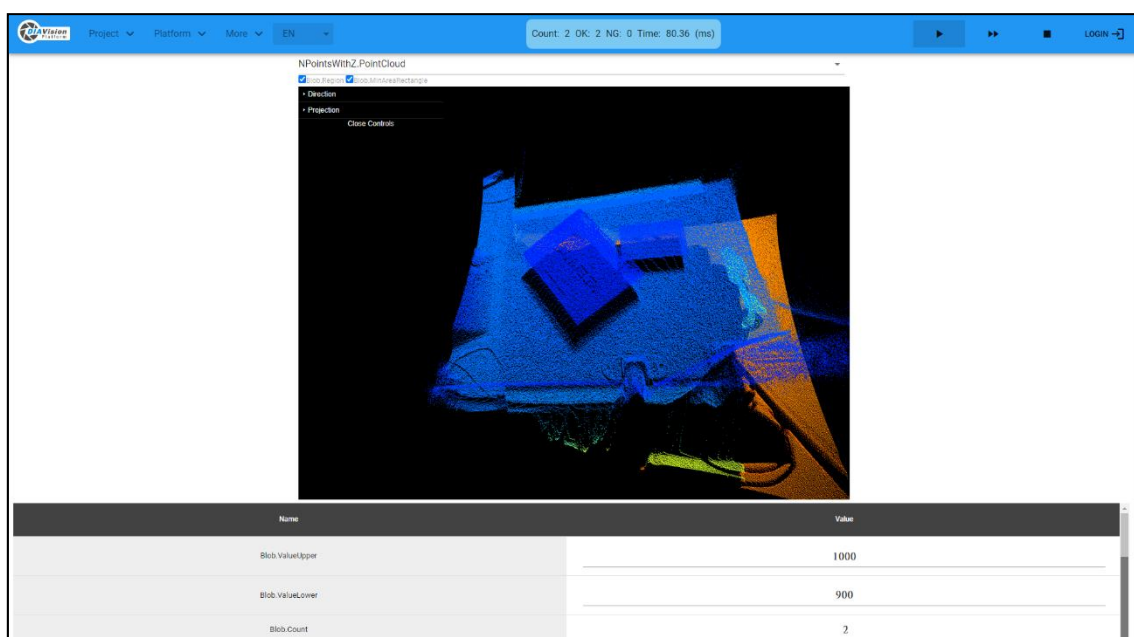
- If the following message is displayed, please try to refresh the page or restart the ToF camera, and wait for 60 seconds before trying to connect again.



- If users enter the browser, users can check the Project List of items currently on the device, as well as the items that are currently running, and the default items.



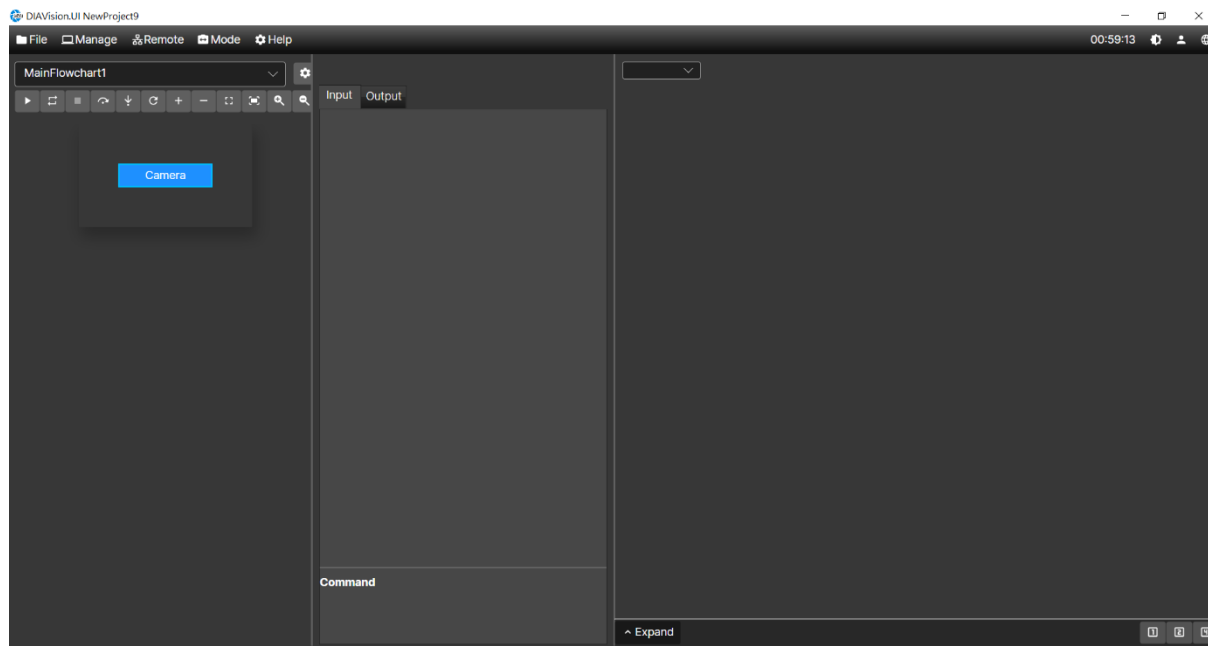
- After the settings are complete, the page is as follows.



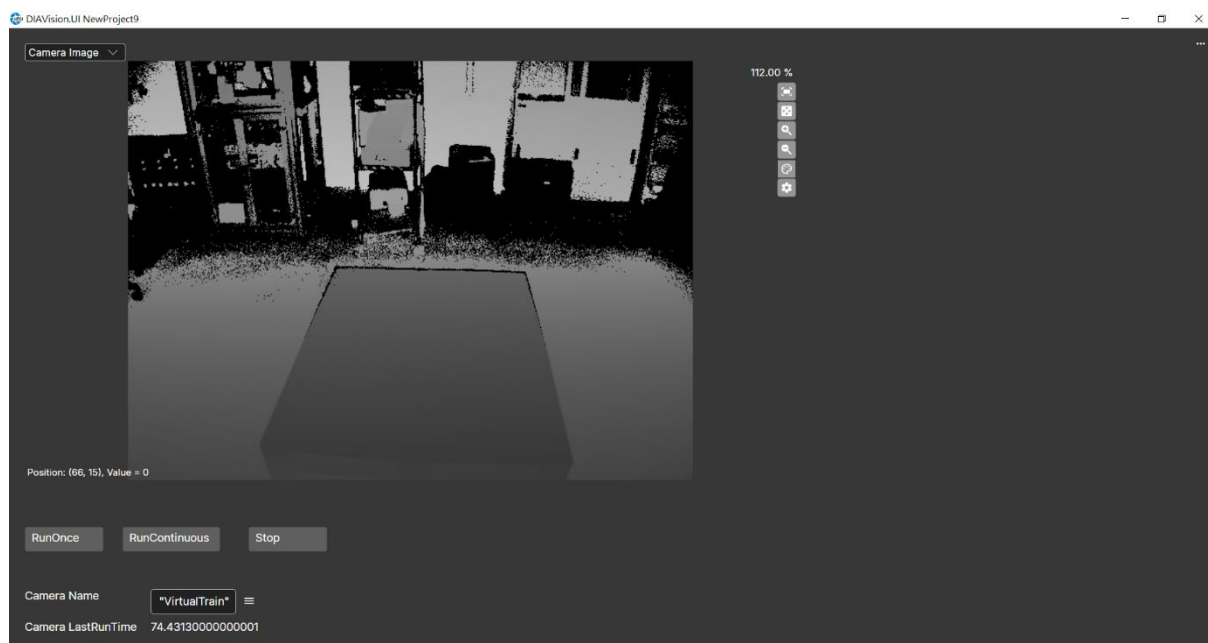



### 3.1.1.4 Mode

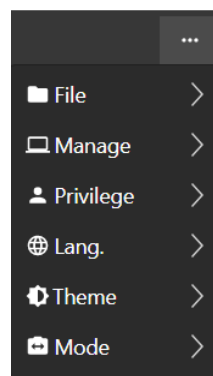
A. Edit Mode: Users can edit flowchart and UI.



B. Run Mode: Displays the edited UI and flowchart execution results.

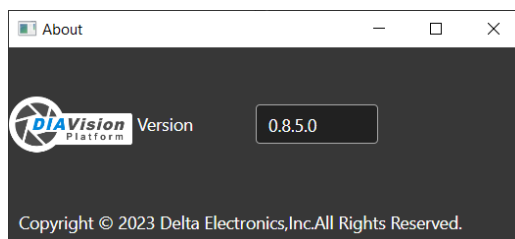


Users can return to Edit Mode by clicking the icon  at the top right of the right picture to enter the mode and return to Edit Mode.



### 3.1.1.5 Description

About: Current software version.



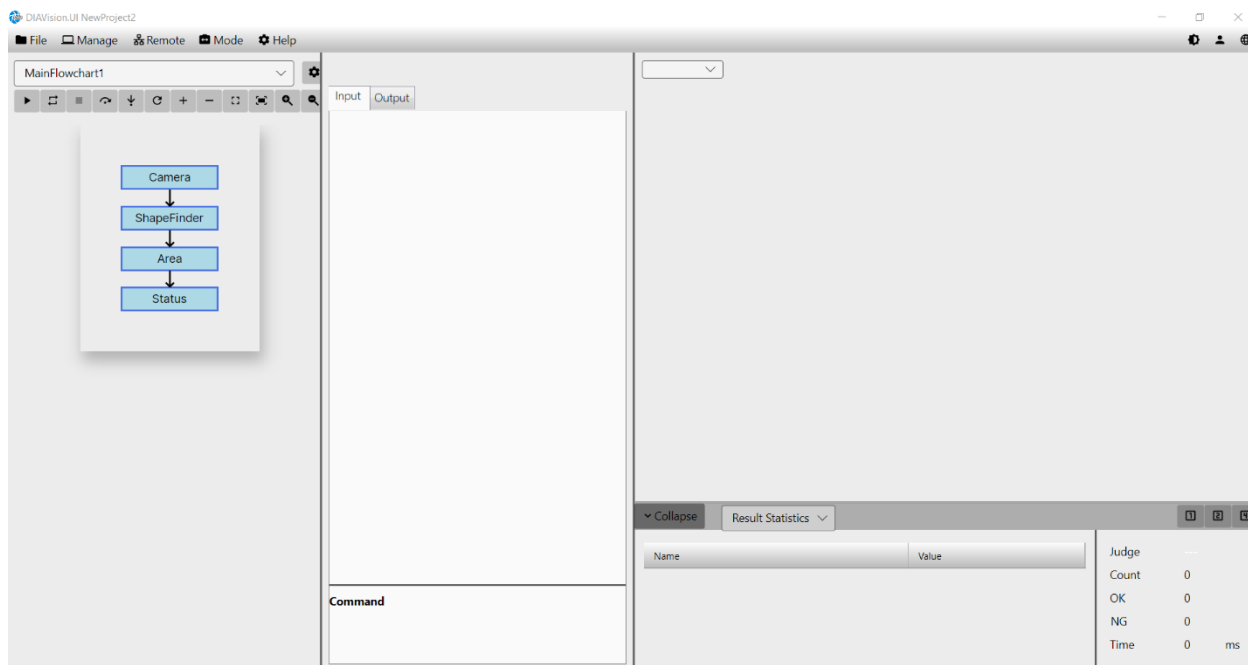
Plugin Information: Displays the plugin components currently included in the software.

Name	Version	Message
CAConfigCore	1.0.0.0	
DIAVision.Core	0.8.5.0	
DIAVision.DIAElements	0.8.5.0	
DIAVision.DIAElements.Amd64	0.8.5.0	
DIAVision.Device3D	0.8.5.0	
DIAVision.Device3DUI	0.8.5.0	
DIAVision.GigeDevice	0.8.5.0	
DIAVision.Lib3D	0.8.5.0	
DIAVision.OpenCVSharp	0.8.5.0	
DIAVision.Robot	0.8.5.0	
DIAVision.UI.Base	0.8.5.0	
DMVAI	1.0.0.0	

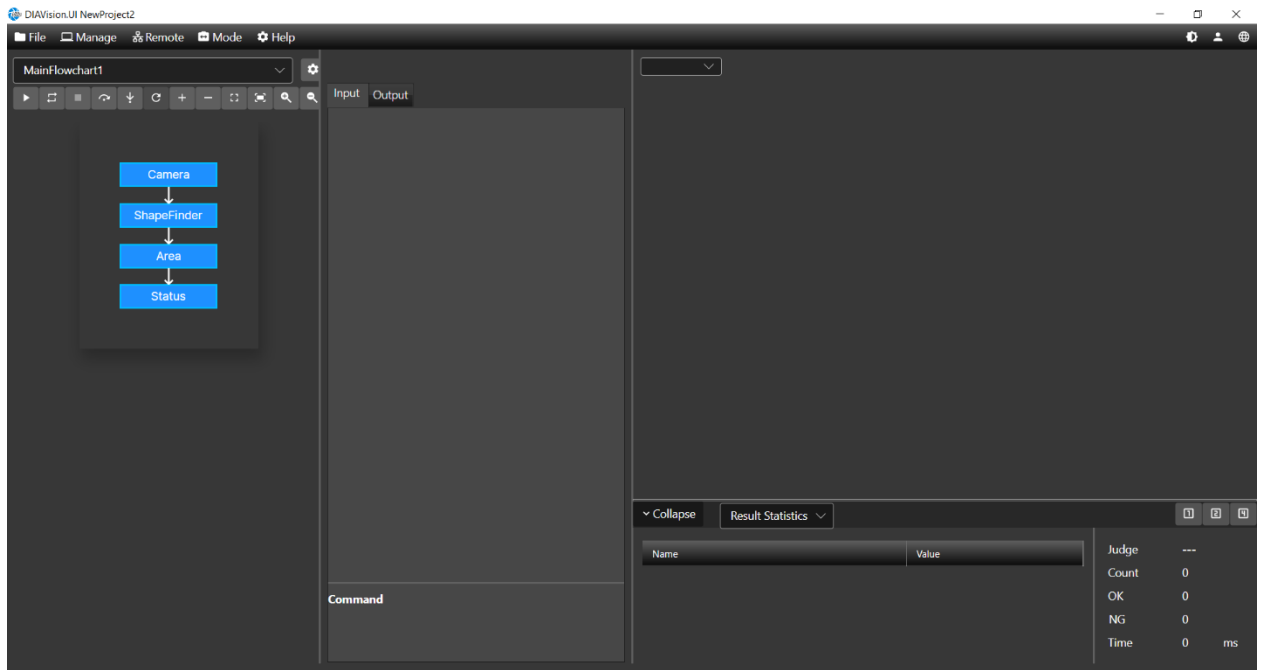
### 3.1.1.6 Theme

Clicks to toggle between a light (white) theme or a dark (dark gray) theme.

- Bright (white) theme:



- Dark (Dark Gray) theme:



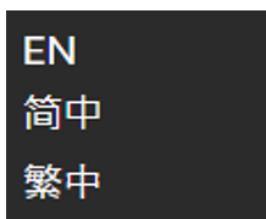
### 3.1.1.7 Permissions

The default permission is guset. User permission is required to build a project.

Account name: **user**. Default password: **2222**.

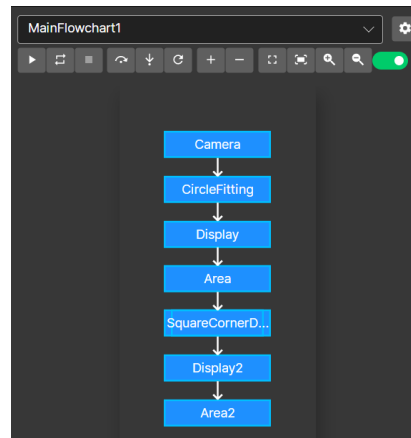
### 3.1.1.8 Languages

The software supports English, Simplified Chinese and Traditional Chinese languages.

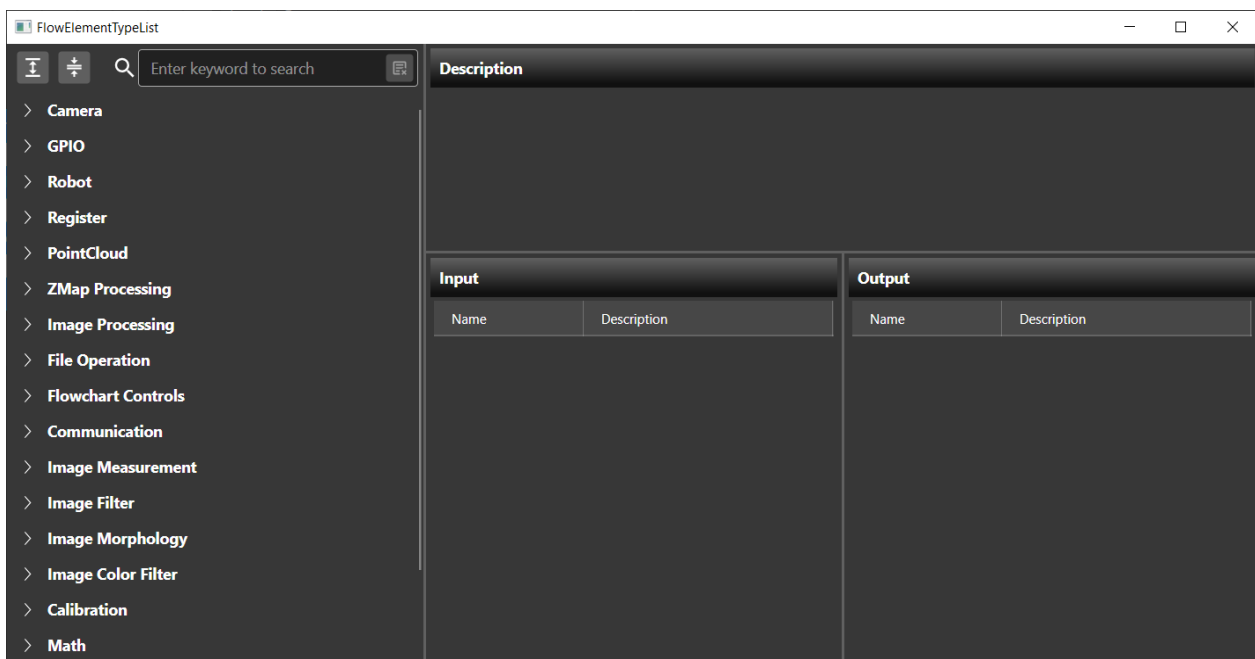


## 3.2 Flowchart

The flowchart is composed by Vision Functions (flowchart components). Users can design execution sequence of Vision Functions in flowchart to perform a specific vision task.



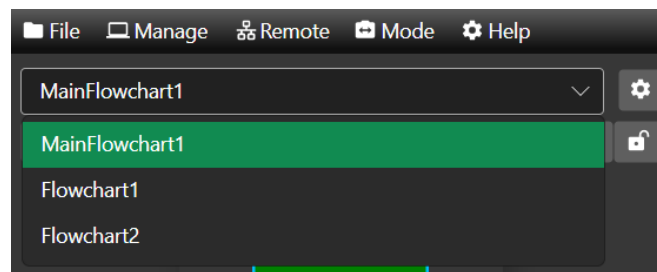
- ▶ Run Once: Executes the entire flow once, from top to bottom, executes to the last component.
- 🔄 Run Continuous: The entire flow is executed continuously until the Stop button is pressed.
- Stop: Immediately stops the flow execution.
- ⏪ Step Over: Executes the component step by step, and if there is a sub-flow, the sub-flow will be executed directly.
- ⏩ Step Into: Executes the component step by step, if there is a sub-flow, it will enter the sub-flow and execute it step by step.
- 🔄 Reset: Forces the current execution phase to end and reinitialize all variables and memory.
- ⊕ Add: After clicking it, the FlowElementTypeList will appear, as shown in the figure below, for users to add components.



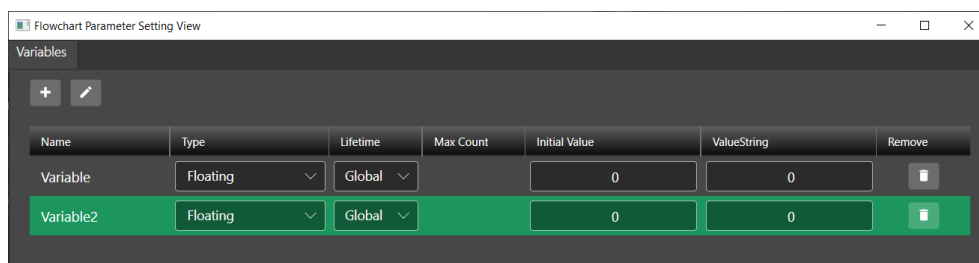
- Remove: Removes the specified component of the flow diagram, and users click on the button to delete the component.
- Reset Flowchart Diagram: Resets the flowchart.
- Auto Fit: Automatically resize the flowchart.
- Zoom In: Zooms in the flowchart.
- Zoom Out: Zooms out the flowchart.
- Unlock Flowchart: When flowchart unlocked, user can re-arrange component.

### 3.2.1 Basic Operation of Flowchart

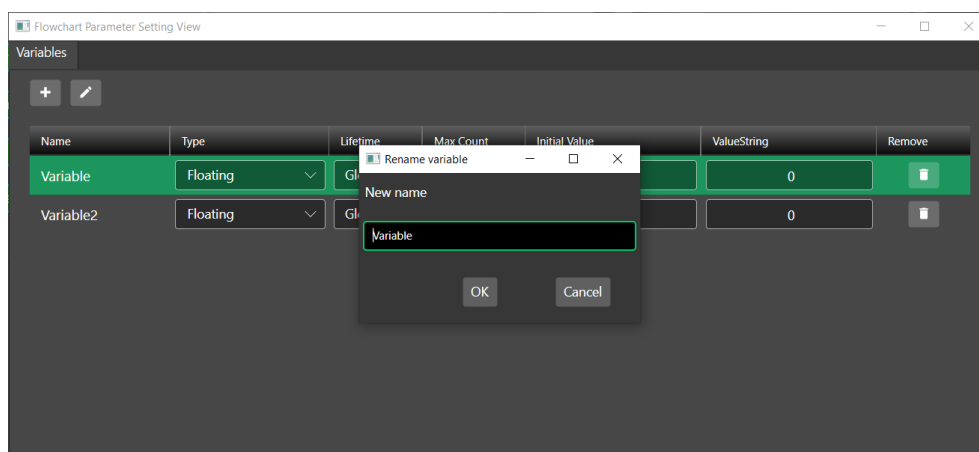
- Flowchart: Users can select the flowchart they want to design through the menu list, shown as below.





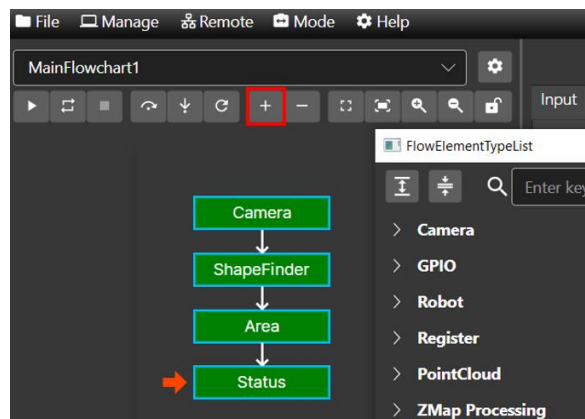
- Variables: Users can control the flow or calculate numerically through variables in the flowchart.



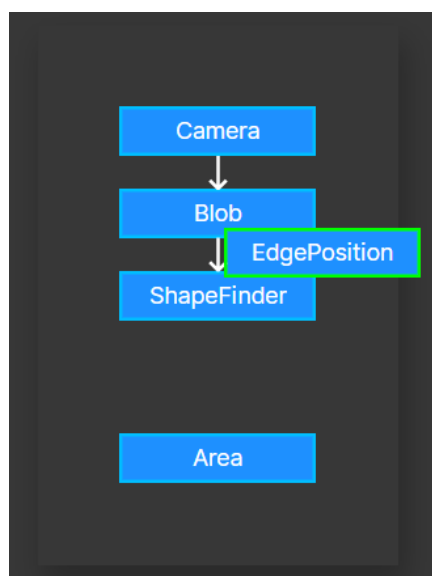
- Add: To add a variable.
- Rename: Modifies the name for the variable.




- Name: The name of the variable.
- Type: The type of variable, also known as data type, including Boolean, Byte, Numeric, Floating, String, Image, PointCloud, Boolean Array, Byte Array, and Numeric Array, Floating Array, String Array, Image Array, PointCloud Array. Users select a specific type and create a variable.
- Lifetime: Divided into Global or Local variables. When resetting the process, the Global variable remains unchanged at its current value, and the Local variable is reset back to its initial value.
- Max Count: If the type is Array, displays the maximum space of the array.
- Initial Value: Sets the variable initialization value or string.
- ValueString: The current variable value or string.
- Remove: Click this button  to remove the variable.
- Click Add Component  in the flow control bar to add a component to the process, as shown below.

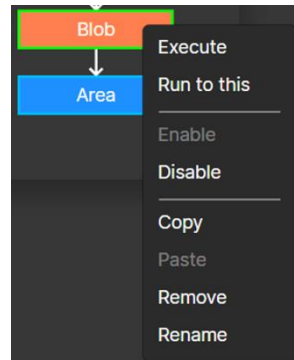


- If users need to change the order of the components, use the mouse to drag the components to the specified position in the flowchart, and the mouse will change the order of the components when it is released.



- If users need to delete it, users can click on the specified component and click  the button, and the flow will delete the component.  
In addition to the flow control bar, clicking on a component and then right-clicking on it will

bring up the menu functions as shown below, with a total of 8 functions:




- Execute: Executes the specified component.
- Run to this element: The current location component of the process execution is executed to the specified component.
- Enable: Enables the executable state of the specified component.
- Disable: Turns off the executable state of the specified component.
- Copy: Copies the specified component.
- Paste: After performing the Copy function, click Paste at the specified position to paste the duplicate component below the specified component.
- Remove: Removes the specified component from the process.
- Rename: Renames the specified component.

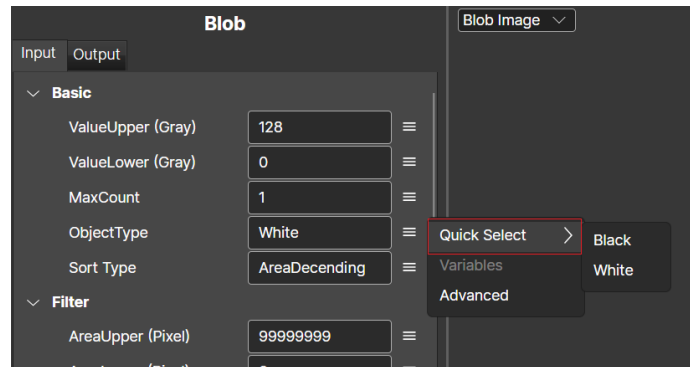
### 3.3 Component Input/Output Settings


Users can set parameters or retrieve execution result of components on Input/Output page.

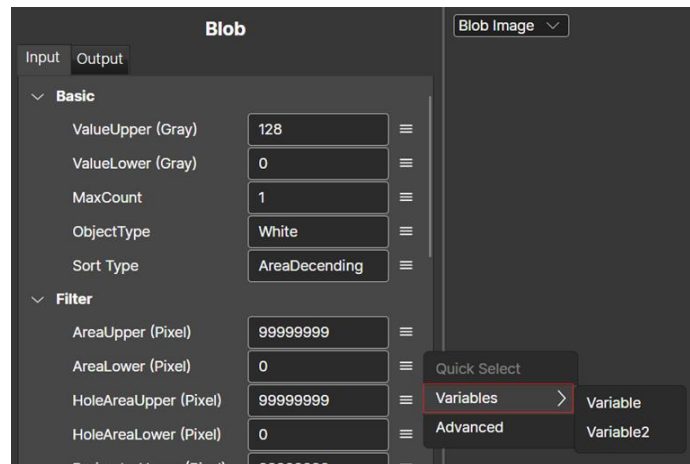
Page	Section	Parameter/Property	Value
Input	General	Image	Camera.Image
		ModelRoi	Rectangle
		Roi	WholeImage
	Basic	TargetCount	3
		Score	75
	Model	Mask	<input checked="" type="checkbox"/>
		Artificial Model	<input type="checkbox"/>
		AngleLower	-180
		AngleUpper	180
		Accuracy	Coarse
		ScaleTolerance	Affine
		IsPyramidLevelCustomized	false
	PyramidLevel	2	
Output	Name	ShapeFinder	
	TypeName	ShapeFinder	
	Status	True	
	ErrorMessage		
	LastRunTime	57.4024	
	AutoPyramidLevel	4	
	Count	0	
	Results		


- Input: Users can set the parameters of components on the Input page.

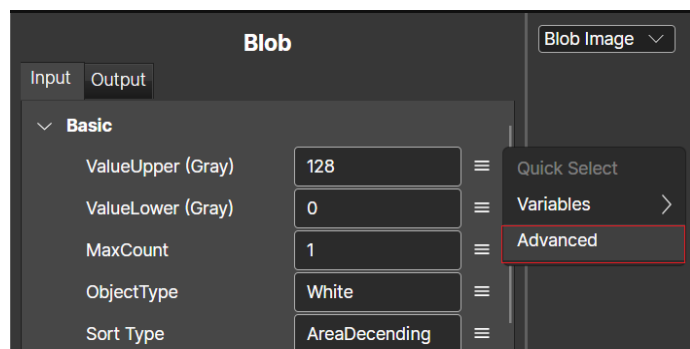
- Quick Select: Clicks  in the component's Input field to open the menu > Quick Select item value.



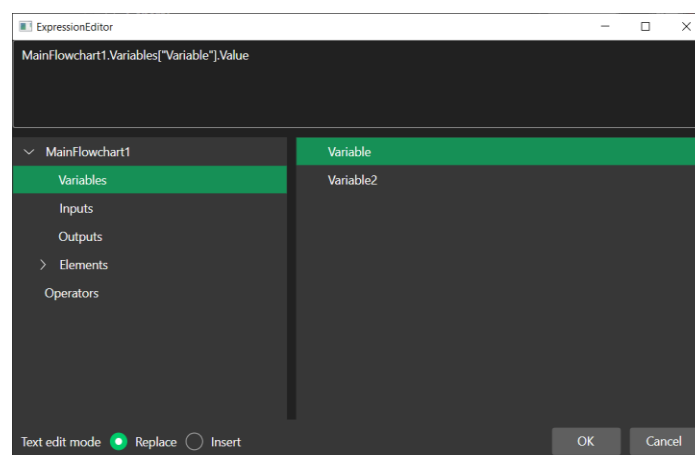
- Variables: Clicks  in the component's Input field to open the menu > Variables is to select the configured flow variables as item values.



- Advanced: Clicks  in the component's Input field to open the menu > Advanced opens the Advanced Settings window.



The Component Input field can be set by the Advanced Settings window, using parameters and operators to form an operation.





- Text edit mode
  - ◆ Replace: When users select an output item, the editor will be replaced entirely, as shown below. The override option is to overwrite all content in the expression editor with new content.



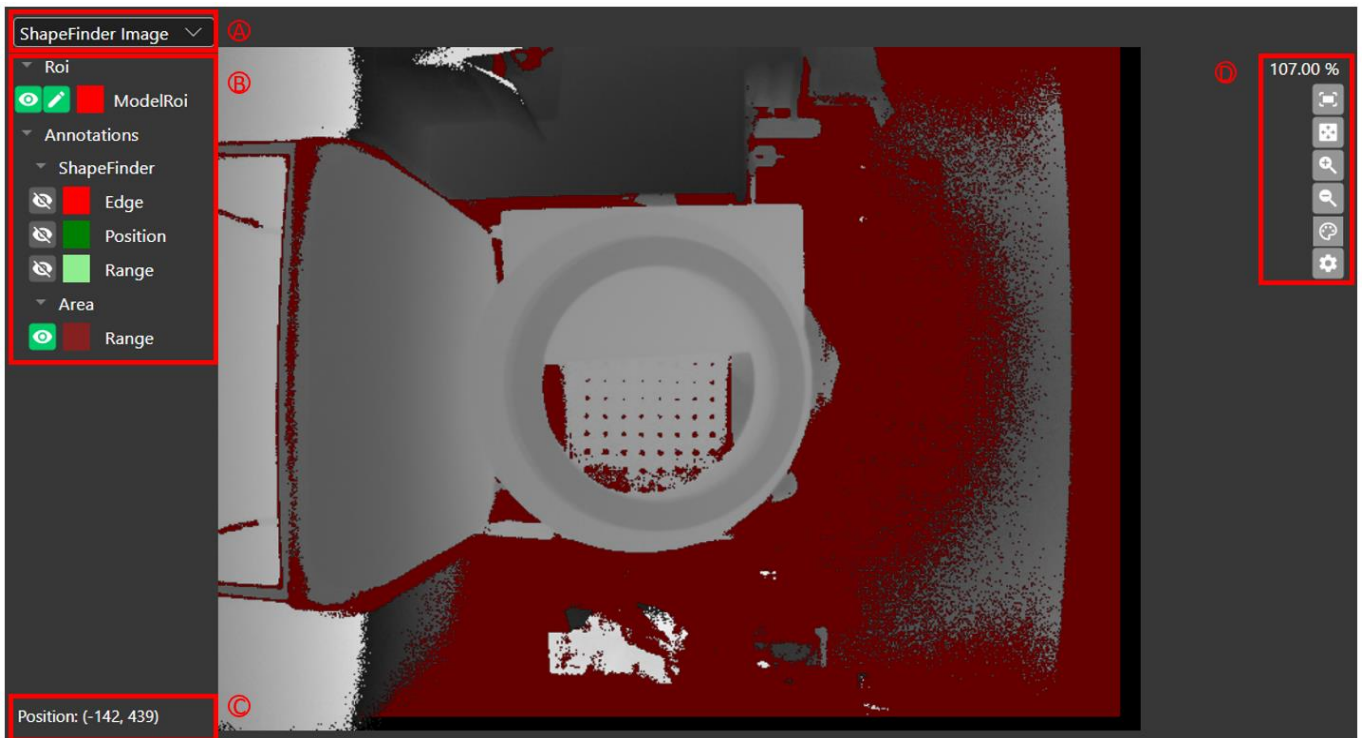
- ◆ Insert: After selecting the output item, the editor will add the output content, as shown below. The insert option is added to Content after the cursor.



- Output: When a component is executed, users can get the result of the component on the output page.

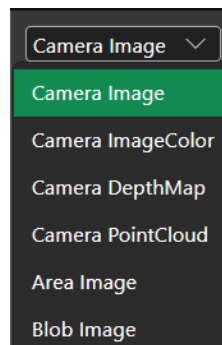
## 3.4 Image Display

The result image or device image will be displayed in this view after flow execution completed as shown in the figure below. Currently, the supported image formats are bmp, png and pcd.





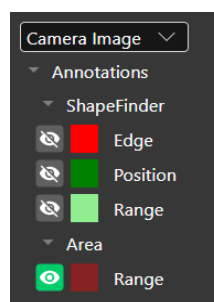
### A. Component images

The current image can be set according to users' needs.



### B. Features

 Enables /  Disables the Feature Maps of components. When component is selected, images and features belong to this component will be displayed.



### C. Image information


- When the image is in 2D, the pixel coordinates (X, Y) coordinates and grayscale values are displayed, as shown in the following figure.

Position: (254, 188), Value = 38048


- When the image is in 3D, the X, Y, and Z coordinate values are displayed.

X: -170.40 Y: 148.74 Z:1750.85


### D. Image settings

 View: Users can click on the image to select the viewing angle.


- Rest: Restore the initial viewing angle.
- Front: The front view.
- Back: The back view.
- Top view.
- Bottom: The bottom view.
- Left: Left view.
- Right: Right view.

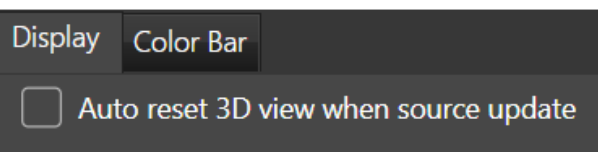
 Color Mode: A variety of colors are currently available, and users can highlight features by selecting the appropriate color mode.

 Stereogram: Perspective and Orthographic display are currently available.

 Setting: Image display settings.

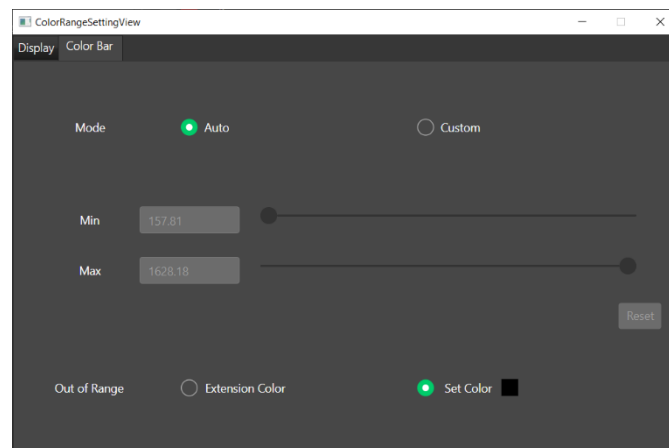
- 3D settings
  - Display: When users check "Auto reset 3D view when source update", the originally adjusted angle of view will be reset when a new image is input.

 ColorRangeSettingView



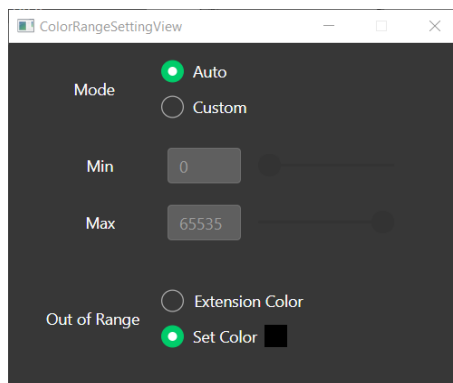
- Color Bar:
  - ◆ Mode:
    - Auto: The software automatically sets the color level.
    - Custom: To set the display range according to users' needs.
      - Min: Sets the minimum display range of Z-value.
      - Max: Sets the maximum display range of Z-value.
      - Out of Range: Sets the display method for out of range.
        - Extension Color: Setting the color outside the Min range makes the depth value smaller; setting the color outside the Max range makes the depth value larger.

- Set Color: Sets the color outside the range of Min and Max values; the default is black.



- 2D Settings

- ColorRangeSettingView



- ◆ Mode:

- Auto: Color level is set automatically.
      - Custom: Selects Custom to set the display range according to your needs.
        - Min: Sets the minimum display range of grayscale values.
        - Max: Sets the maximum display range of grayscale values.
        - Out of Range: Sets the Out of Range Display.
          - Extension Color: Setting a color outside the Min range makes the pixel value smaller; setting a color outside the Max range makes the pixel value larger.
          - Set Color: Sets the color outside the range of Min and Max values; the default is black.

## 3.5 Result Statistics Information Setting

The Result Statistic Information shows the result of flow execution. The result can be displayed upon users' request.

Name	Value
ShapeFinder.Results.[0].Position.X	899.0001220703125
ShapeFinder.Results.[0].Position.Y	449.0060119628906
ShapeFinder.Results.[0].Angle	0.0008828380377963185

Judge	OK
Count	4
OK	4
NG	0
Time	20.1 ms

### A. Result statistics page settings

Collapse/Expand: Expands or closes the window for the Result Statistics column.

Result Statistics/Syslogs:

- Result Statistics: Users can add component output to the Result Statistics page, and it will be updated after flow execution.
- System Log: A file or record that records software system activity, error messages, events, and other important information. They help system administrators and developers monitor the health of the system and provide clues to troubleshoot errors if something goes wrong.

### B. Message page

- Result Statistics Message: If users select Result Statistics on the screen, the object name and value are displayed in the message.

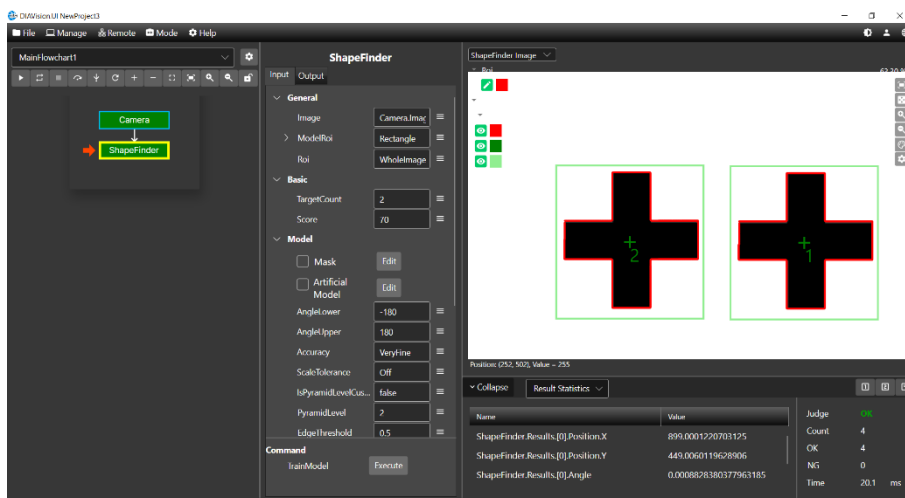
Name	Value
ShapeFinder.Results.[0].Position.X	899.0001220703125
ShapeFinder.Results.[0].Position.Y	449.0060119628906
ShapeFinder.Results.[0].Angle	0.0008828380377963185

- Syslog Messages: If the page is syslog, the event Date, Level, Category, and Message will be displayed.

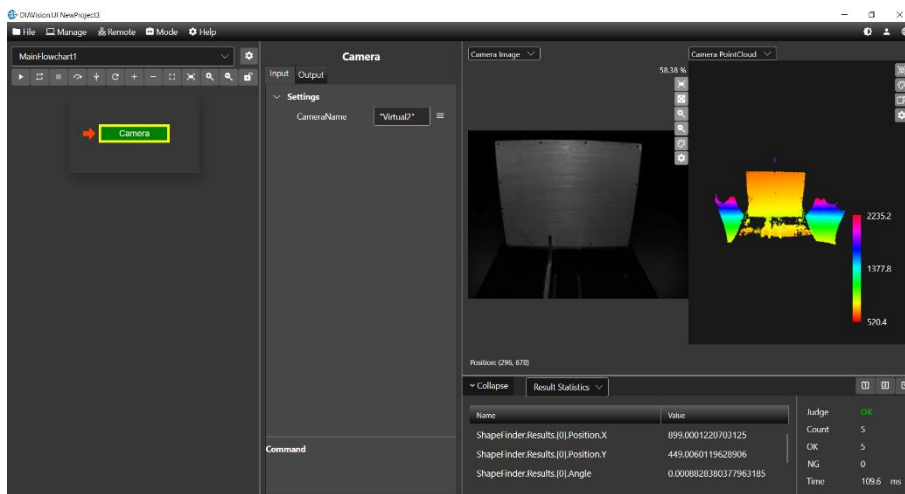
Date	Level	Category	Message
12/26/2023 15:58:23	Error	Others	ShapeFinder Image is null or isn't 8 or 16 bit
12/26/2023 15:58:23	Error	Others	Area Image is null or isn't 8 or 16 bit
12/26/2023 15:46:27	Warning	Camera	Camera2 Connection Failed

C. Image Display Label: Users can select the current page to display 1,2 or 4 paging images.

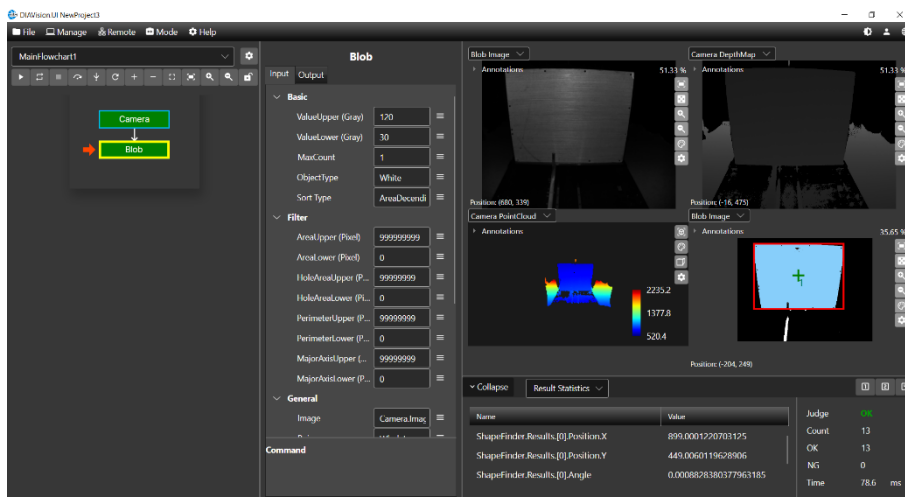
1 paging image display:



2-paging image display:



4-paging image display:



D. Statistical results: The software will record the results of the flow after execution. It contains the current judgment result, the number of runs (the number of triggers), the OK result, the NG result, and the process running time. It should be noted that the statistical results are only valid through the process trigger button (single execution, continuous execution).

Judge	OK
Count	13
OK	13
NG	0
Time	78.6 ms

When users need to reset the statistical results, just right-click on the statistical results area and press Reset.

Judge	OK
Count	13
OK	13
NG	0
Time	78.6 ms

Reset

# Chapter 4

## Vision Functions

### 4.1 Camera

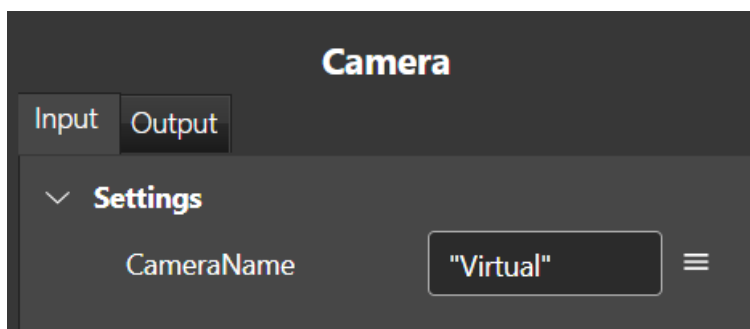
#### 4.1.1 Camera

Function:

1. Device search method can be defined by users, which supports physical / virtual camera connection.
2. Cameras can be added to the camera list and used in the flowchart.
3. Supports image (8-bit/16-bit/24-bit) or point cloud.

Input:

- CameraName: Selects the image source, such as "Camera", "TofDevice", etc.



Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of components.
- Image: 2D image information.
  - Width: The width of the image.
  - Height: The height of the image.
  - Pitch: Refers to the continuous distance that each row of pixel data is stored in memory. If each pixel occupies N bytes, the spacing is equal to the width of the image (in pixels) multiplied by the number of bytes per pixel. For example, if a 16-bit image is 640 wide and 480 high, 16-bit occupies  $16/8 = 2$  bytes for each pixel, so the spacing is  $640 * 2 = 1280$ .
  - Bpp: bit depth, which indicates how accurate the color information stored by the pixel is. The numeric portion represents the number of bytes in each pixel that can be used



to represent color. Higher bit depths generally indicate more color information and higher image quality.

- ScaleX: The ratio of the width of one pixel to one unit in an image. For example, if the image unit is millimeters (mm) and the X ratio is 9, it means that the width of a pixel is equal to 9 mm.
  - OffsetX: Represents the X-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ScaleY: The ratio of one pixel in an image to one unit.
  - OffsetY: Represents the Y-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ScaleZ: The ratio of the depth of one pixel to one unit in an image.
  - OffsetZ: Represents the Z-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - UnitX(mm): The unit of the image pixel X.
  - UnitY(mm): The unit of the image pixel Y.
  - UnitZ(mm): The unit of the image pixel Z.
  - IsAllocated: "Allocate" represents allocating enough computer memory or resources to store image data for further processing, displaying and manipulating images.
- ImageColor: 2D color image information.
    - Width: The width of the image.
    - Height: The height of the image.
    - Pitch: Refers to the continuous distance that each row of pixel data is stored in memory. If each pixel occupies N bytes, the spacing is equal to the width of the image (in pixels) multiplied by the number of bytes per pixel. For example, if a 16-bit image is 640 wide and 480 high, 16-bit occupies  $16/8 = 2$  bytes for each pixel, so the spacing is  $640 * 2 = 1280$ .
    - Bpp: A 24-bit color image stores information in three color channels: red (R), green (G), and blue (B) in the form of 8 bytes (1 byte).
    - ScaleX: The ratio of the width of one pixel to one unit in an image. For example, if the image unit is millimeters (mm) and the X ratio is 9, it means that the width of a pixel is equal to 9 mm.
    - OffsetX: Represents the X-direction displacement of an object in 3D space relative to a reference point or coordinate system.
    - ScaleY: The ratio of one pixel in an image to one unit.
    - OffsetY: Represents the Y-direction displacement of an object in 3D space relative to a reference point or coordinate system.
    - ScaleZ: The ratio of the depth of one pixel to one unit in an image.
    - OffsetZ: Represents the Z-direction displacement of an object in 3D space relative to a reference point or coordinate system.
    - UnitX(mm): The unit of the image pixel X.

- UnitY(mm): The unit of the image pixel Y.
- UnitZ(mm): The unit of the image pixel Z.
- IsAllocated: "Allocate" represents allocating enough computer memory or resources to store image data for further processing, displaying and manipulating images.
- Depth Image: 3D depth image information.
  - Width: The width of the image.
  - Height: The height of the image.
  - Pitch: Refers to the continuous distance that each row of pixel data is stored in memory. If each pixel occupies N bytes, the spacing is equal to the width of the image (in pixels) multiplied by the number of bytes per pixel. For example, if a 16-bit image is 640 wide and 480 high, 16-bit occupies  $16/8 = 2$  bytes for each pixel, so the spacing is  $640 * 2 = 1280$ .
  - Bpp: Indicates the depth value of each pixel using N digits. These depth values are typically presented as grayscale images and are used to represent the distance or depth of each pixel from the camera or observer. For example, a 16-bit depth map provides 256 (65,536) different depth values.
  - ScaleX: The ratio of the width of one pixel to one unit in an image. For example, if the image unit is millimeters (mm) and the X ratio is 9, it means that the width of a pixel is equal to 9 mm.
  - OffsetX: Represents the X-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ScaleY: The ratio of one pixel in an image to one unit.
  - OffsetY: Represents the Y-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ScaleZ: The ratio of the depth of one pixel to one unit in an image.
  - OffsetZ: Represents the Z-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - UnitX(mm): The unit of the image pixel X.
  - UnitY(mm): The unit of the image pixel Y.
  - UnitZ(mm): The unit of the image pixel Z.
  - IsAllocated: "Allocate" represents allocating enough computer memory or resources to store image data for further processing, displaying and manipulating images.
- Point cloud: 3D image coordinate information.
  - Width: The width of the arrangement of point cloud data in memory.
  - Height: The height of the arrangement of point cloud data in memory.
  - Count: The number of point clouds is width \* height.
  - IsAllocated: "Allocate" represents allocating enough computer memory or resources to store point cloud data for further processing, displaying and manipulating point cloud.
  - HasNormal: The normal vector refers to the orientation of a point in 3D space, perpendicular to the surface where the point is located. Indicates that each point is

accompanied by information about the direction of its surface normal.

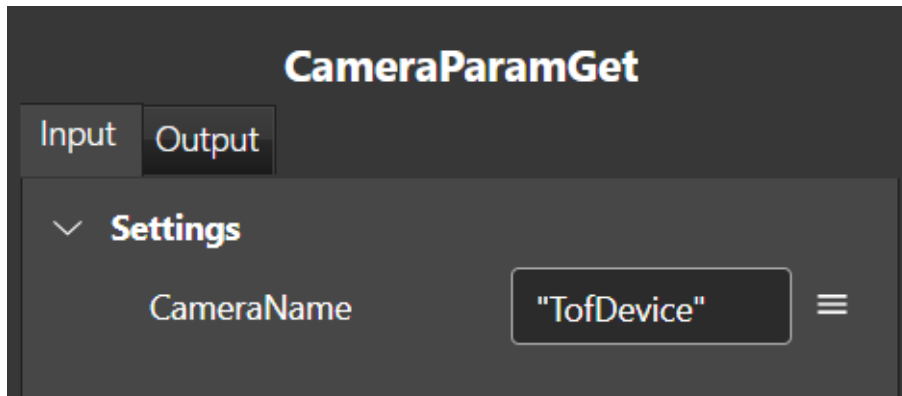
- HasColor: Indicates that each point is accompanied by a color, and the color of the point is described in red, green, and blue.

## 4.1.2 CameraParamGet

Function: Gets the parameters of GIGE cameras.

Input:

- CameraName: Selects the image source, such as "Camera", "TofDevice", etc.



Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of components.
- AnalogGain: The value of the camera gain.
- ExposureTime: The camera sets the exposure time.
- TriggerSource: Camera trigger mode, which is divided into software trigger and IO trigger.

## 4.1.3 CameraParamSet

Function: Sets the parameters of the GIGE camera.

Input:

- CameraName
  - ExposureTime: Sets the exposure time in milliseconds (msec).
  - AnalogGain: Sets the analog gain value of the camera.
  - TriggerSource: Sets the trigger source.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents

abnormal execution.

- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.

#### 4.1.4 ToFDeviceParamGet

Function: Gets the specific ToF camera parameters.

Input:

- Camera Name: The name of the ToF camera which the parameters are obtained.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- CameraDistortionCoeff: The camera distortion factor.
- CameraIntrinsics: Camera internal parameters.
- ConfidenceFilterEnable: Whether to enable confidence filtering.
- ConfidenceFilterValue: The confidence value filtered value.
- ExposureTime: The length of the camera's exposure time, measured in microseconds.
- RangeMode: Detects range distance mode.
- Scan3dCoordinateScale: The scale of the camera scanning coordinates.
- TriggerSource: Trigger the source of the camera capture.

#### 4.1.5 ToFDeviceParamSet

Function: Sets the parameters of the specific ToF camera.

Input:

- CameraName: The name of the ToF camera which the parameters are obtained.
- ExposureTime: Sets the length of the camera's exposure time in microseconds.
- RangeMode: Sets the detection range distance mode. Range 1500 means detection depth up to 1.5 meters, Range 6000 means detection depth up to 6 meters, and so on.
- ConfidenceFilterEnable: Enables confidence filtering.
  - Confidence filtering: The goal is to filter or weightize data to increase confidence in a particular point while reducing uncertainty or noise.
- ConfidenceFilterValue: Sets the confidence threshold, and the software will be regarded as a valid point only if the feature point reaches the confidence value or above.
- TriggerSource: Triggers the source of the camera capture. Users can choose between software triggering or I/O triggering.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.

## 4.2 GPIO

Function: Reads/writes GPIO board or camera I/O.

Input:

- I/O Name: The I/O name of the device.
- Type: Sets the input or output type.
- Port: Input/output port.
- Output Status: The output status.

Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Input Status: Pin Status.

## 4.3 Robot

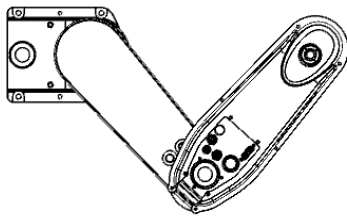
### 4.3.1 RobotMove

Function: Moves robot to the specific pose.

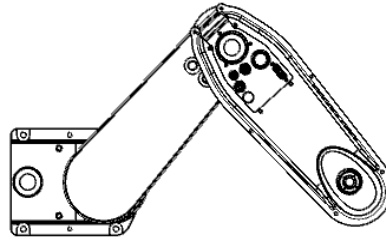
Input:

- Point:
  - ◆ X: Sets the X coordinate of the robot.
  - ◆ Y: Sets the Y coordinate of the robot.
  - ◆ Z: Sets the Z coordinate of the robot.
  - ◆ RX: Sets the RX coordinates of the robot.
  - ◆ RY: Sets the RY coordinates of the robot.
  - ◆ RZ: Sets the RZ coordinates of the robot.
  - ◆ Hands: Sets the current hand system of the robot. The options are divided into left-hand system and right-hand system. This command is applicable only to the four-axis and five-axis robots.

The right-hand system

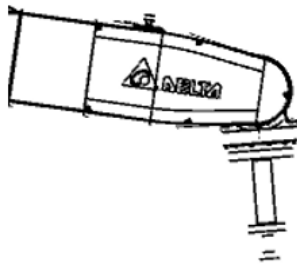


The left-hand system

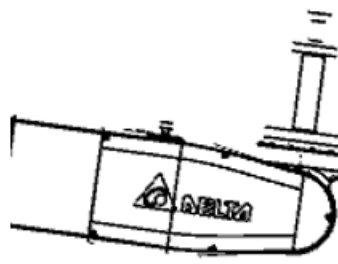


- ◆ Flip: Sets the current wrist rotation of the robot. Options are divided into non-flip and flip. This command is applicable only to six-axis robots.

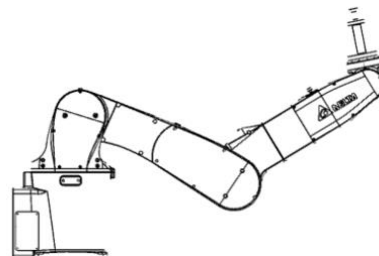
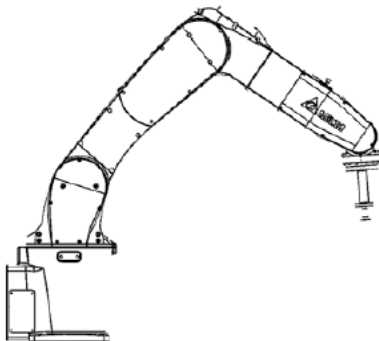
Non-Flip



Flip



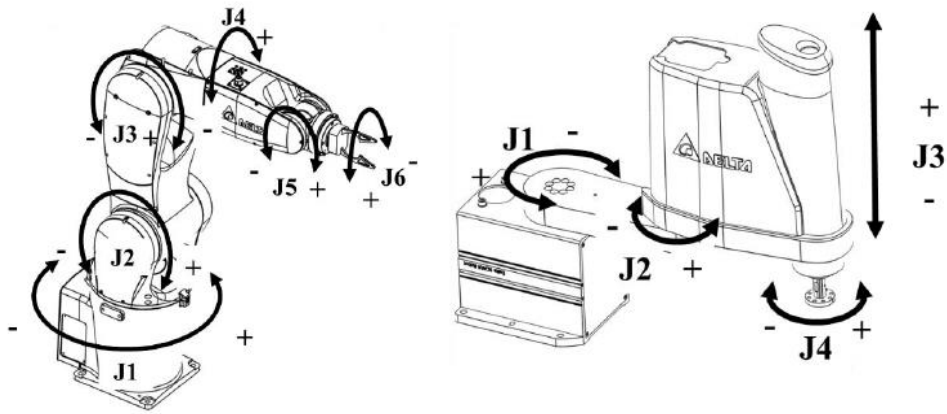
- ◆ Elbow: Sets the current elbow system of the robot. Options are divided into top elbow system and bottom elbow system. This command applies only to six-axis robots.



Top elbow system

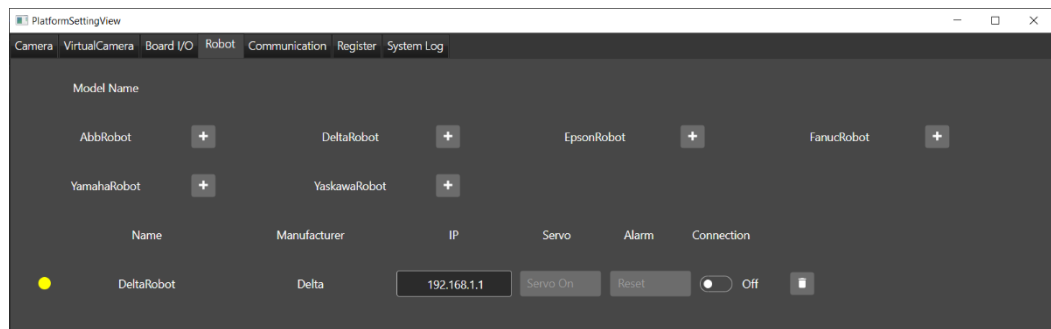
Bottom elbow system

- ◆ Shoulder: Sets the current shoulder system of the robot. Options are divided into left shoulder system and right shoulder system. This command applies only to six-axis robots.
- ◆ J1: Sets the position of the J1 axis of the robot.
- ◆ J2: Sets the position of the J2 axis of the robot.
- ◆ J3: Sets the position of the J3 axis of the robot.
- ◆ J4: Sets the position of the J4 axis of the robot.
- ◆ J5: Sets the position of the J5 axis of the robot.
- ◆ J6: Sets the position of the J6 axis of the robot.



- Set

- ◆ Robot Name: Sets the name of the robot connection. This option is located in <Manage> → <Platform Settings> → <Robots>, as shown in the figure below. Users can create robot connections according to their needs.



After completing the setting, users can quickly select the designated name.

- ◆ Speed: Sets the movement speed of the robot.

Output:

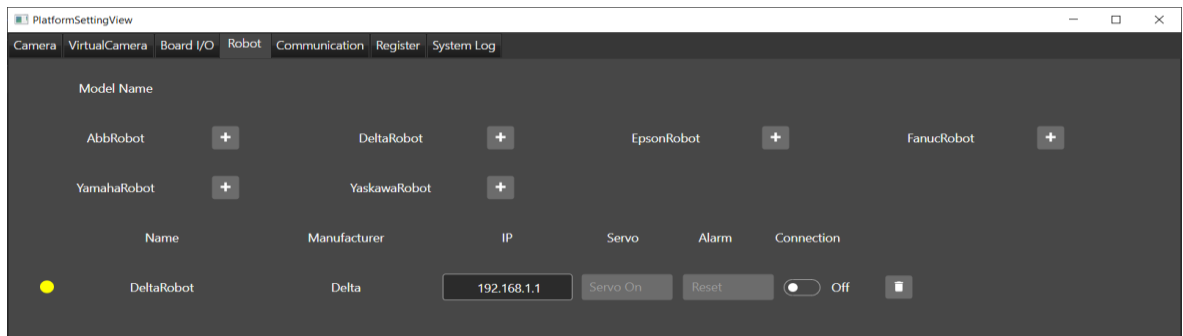
- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.

### 4.3.2 RobotReadPose

Function: Reads robot pose.

Input

- Robot Name: Sets the name of the robot connection. This option is located in <Manage> → <Platform Settings> → <Robots>, as shown in the figure below. Users can create robot connections according to their needs.



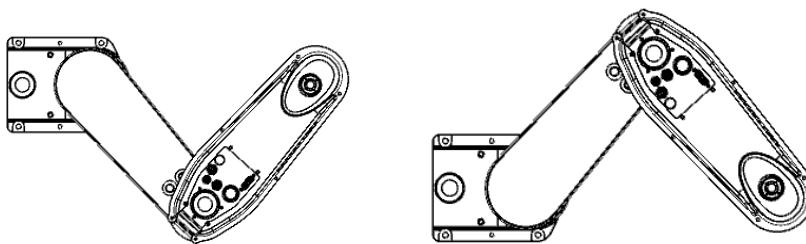
After completing the setting, users can quickly select the designated name.

## Output

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Point:
  - ◆ X: Displays the X coordinate of the robot.
  - ◆ Y: Displays the Y coordinate of the robot.
  - ◆ Z: Displays the Z coordinate of the robot.
  - ◆ RX: Displays the RX coordinate of the robot.
  - ◆ RY: Displays the RY coordinate of the robot.
  - ◆ RZ: Displays the RZ coordinate of the robot.
  - ◆ Hands: Sets the current hand system of the robot. The options are divided into left-hand system and right-hand system. This command is applicable only to the four-axis and five-axis robots.

The right-hand system

The left-hand system

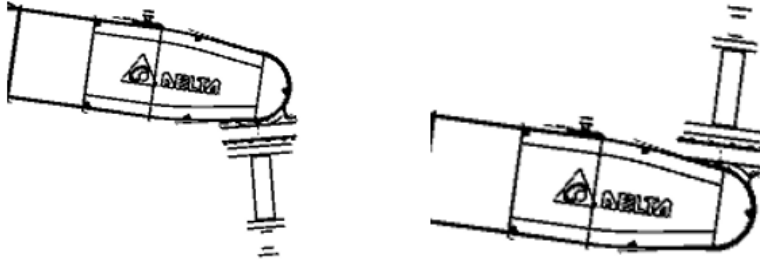


- ◆ Flip: Sets the current wrist rotation of the robot. Options are divided into non-flip and flip. This command is applicable only to six-axis robots.

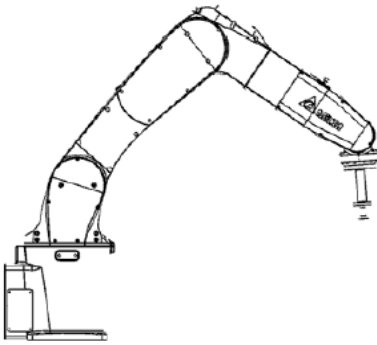
Non-Flip

Flip

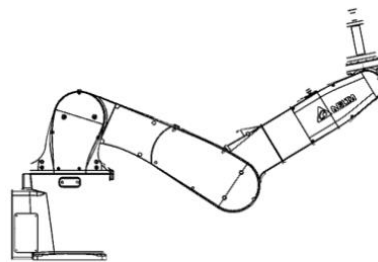




- ◆ Elbow: Sets the current elbow system of the robot. Options are divided into top elbow system and bottom elbow system. This command is applicable only to six-axis robots.

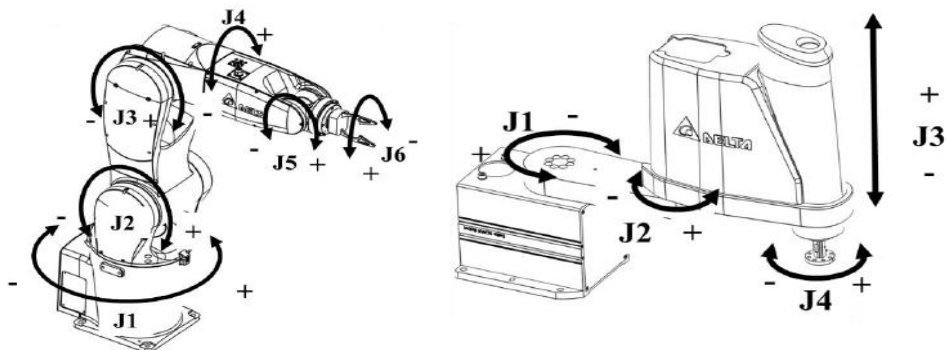


Top elbow system



Bottom elbow system

- ◆ Shoulder: Sets the current shoulder system of the robot. Options are divided into left shoulder system and right shoulder system. This command is applicable only to six-axis robots.
- ◆ J1: Sets the position of the J1 axis of the robot.
- ◆ J2: Sets the position of the J2 axis of the robot.
- ◆ J3: Sets the position of the J3 axis of the robot.
- ◆ J4: Sets the position of the J4 axis of the robot.
- ◆ J5: Sets the position of the J5 axis of the robot.
- ◆ J6: Sets the position of the J6 axis of the robot.



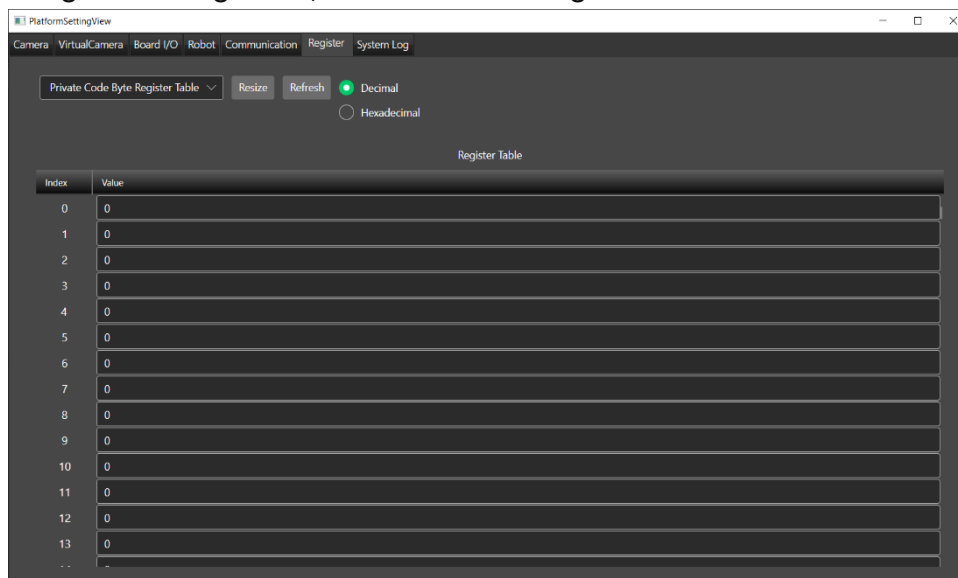
## 4.4 Register

### 4.4.1 RegisterReader

Function: Reads data form Register.

Input:

- Register Name: Sets the name of the read register. Users can select and read the list by selecting a quick selection.
  - Private Code Byte Register Table
  - Private Code Int Register Table
  - Private Code Double Register Table
  - Private Code String Register Table
  - Modbus Register Table
- Format: Sets the reading format. It can be divided into single-value read or array read.
- Index: Sets the reading index number. The index is located in <Manage> → <Platform Settings> → <Register>, as shown in the figure below.



- Count: When setting the read array, users also need to specify the count of reads.

Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Byte Array: If the input is Array, the output reads the Byte array data.
- Byte data: If the input format is single, a single byte of data is output upon reading.
- Double Array: When the input is Array, the output reads the Double Array data.
- Double Data: If the input format is Single, a single read of double-data is output upon reading.

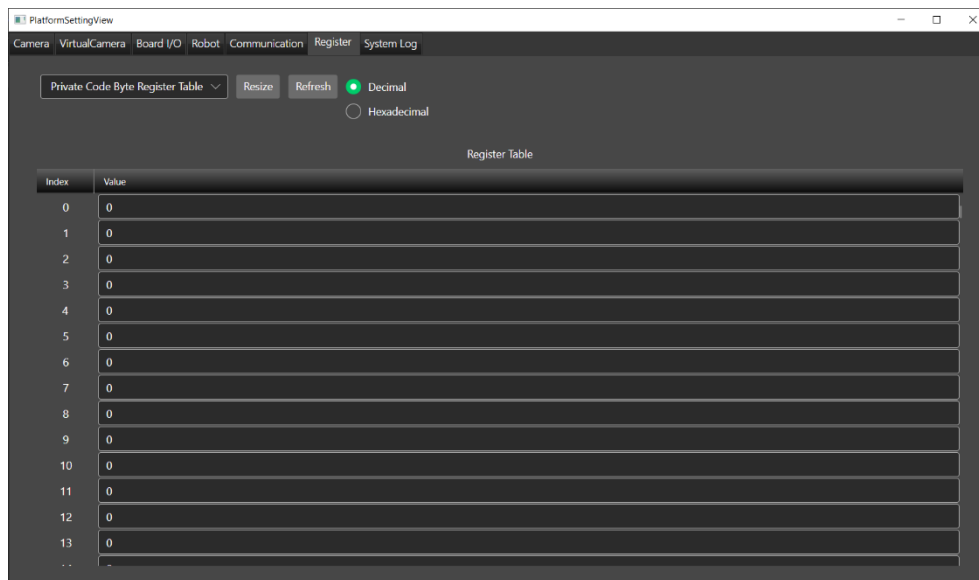
- Int Array: When the input is Array, the output reads the Int array data.
- Int Data: When the input format is Single, a single integer of data is output upon reading.
- String Array: When the input is Array, the output reads the String array data.
- String Data: If the input format is Single, a single read string data is output.
- Ushort Array: If the input is Array, the output reads Ushort array data.
- Ushort Data: If the input format is Single, a single read Ushort data will be output.

## 4.4.2 RegisterWriter

Function: Writes data to Register.

Input:

- Register Name: Sets the name of the write register list.
  - Private Code Byte Register Table
  - Private Code Int Register Table
  - Private Code Double Register Table
  - Private Code String Register Table
  - Modbus Register Table
- Data Format: Sets the data type to be written, as follows.
  - Byte
  - Ushort
  - Int
  - Double
  - String
  - Byte Array
  - Ushort Array
  - Int Array
  - Double Array
  - String Array
- Index: Sets the reading index number. The index is located in <Manage> → <Platform Settings> → <Register>, as shown in the figure below.



- Format Data: Sets the data to be written.

Output:

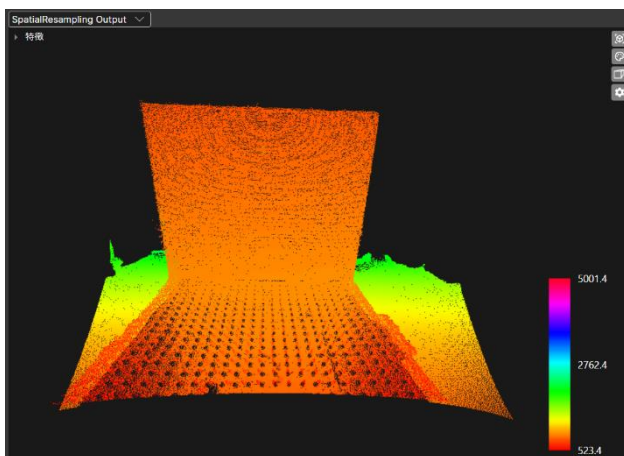
- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.

## 4.5 Point Clouds

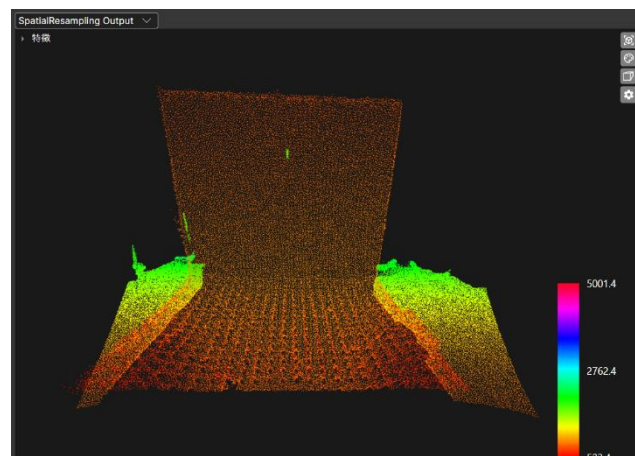
### 4.5.1 SpatialResampling

Function: Resamples the input point cloud and adjust the sampling range.

Before execution



After execution



## Input:

- Mini Dist.: The distance between sampling points (sampling a point cloud within a specified range).
- PointCloud: Point cloud.

## Output:


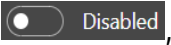

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output:
  - ◆ Width: The width of the image.
  - ◆ Height: The height of the image.
  - ◆ Count: The number of points in the resampled point cloud.
  - ◆ IsAllocated: "Point cloud is allocated" represents allocating enough computer memory or resources to store point cloud data for further processing, displaying and manipulating point cloud.
  - ◆ HasNormal: The normal vector refers to the orientation of a point in three-dimensional space, perpendicular to the surface where the point is located. Indicates that each point is accompanied by information about the direction of its surface normal.
  - ◆ HasColor: Indicates that each point is accompanied by a color, and the color of the point is described in red, green, and blue.

## 4.5.2 ProjectToAnyPlane

Function: Projects the point cloud onto a plane.

## Input:

- PointCloud: Point cloud.
- Plane: Sets the projection plane parameters. The plane equation  $Ax+By+Cz+D=0$ .
- PlaneStepBack(mm): Sets the projection plane retraction distance.
- ImageWidth: Sets the width of the output image.
- ImageHeight: Sets the output image height.
- FillHole: Sets whether the image fills the hole in the area without data.
- OrientationVector: The vector that extends out of the origin of the 3D space after calculation, and this vector is projected onto the plane as the x-axis of the image.
- IsAutoMode:

When IsAutoMode is Enabled , each time the flow element runs, it automatically calculates the projection range parameters. When IsAutoMode is Disabled , the function is disabled, and users can manually acquire automatic parameters once by pressing .

- ◆ PixelSize: The unit of each pixel.
- ◆ ZRangeMax(mm): The maximum distance from the plane.
- ◆ ZRangeMin(mm): The minimum distance from the plane.
- ◆ OriginX(mm): The X origin coordinate of the 3D space after calculation.
- ◆ OriginY(mm): The Y origin coordinate of the 3D space after calculation.
- ◆ OriginZ(mm): The Z origin coordinate of the 3D space after calculation.
- ◆ OriginOffsetX(mm): The X Offset from Origin.
- ◆ OriginOffsetY(mm): The Y Offset from Origin.
- ◆ OriginOffsetZ(mm): The Z Offset from Origin.

Output:

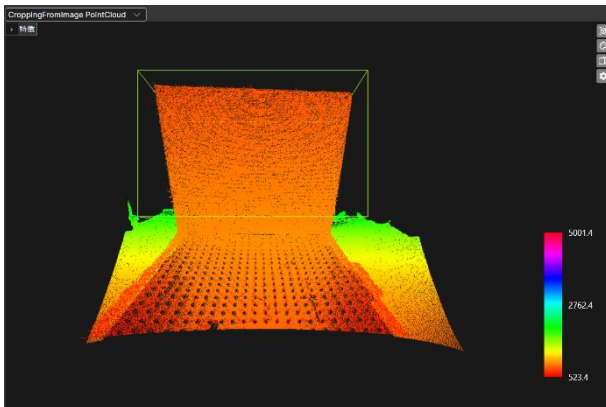
- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- ProjectAnyPanelItem
- ZMap16Bit
  - ◆ Width: The width of the image.
  - ◆ Height: The height of the image.
  - ◆ Pitch: Refers to the continuous distance that each row of pixel data is stored in memory. If each pixel occupies N bytes, the spacing is equal to the width of the image (in pixels) multiplied by the number of bytes per pixel. For example, if a 16-bit image is 640 wide and 480 high, 16-bit occupies  $16/8 = 2$  bytes for each pixel, so the spacing is  $640 * 2 = 1280$ .
  - ◆ Bpp: bit depth, which indicates how accurate the color information stored by the pixel is. The numeric portion represents the number of bytes in each pixel that can be used to represent color. Higher bit depths generally indicate more color information and higher image quality.
  - ◆ ScaleX: The ratio of the width of one pixel to one unit in an image. For example, if the image unit is millimeters (mm) and the X ratio is 9, it means that the width of a pixel is equal to 9 mm.
  - ◆ OffsetX: Represents the X-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ◆ ScaleY: The ratio of one pixel in an image to one unit.
  - ◆ OffsetY: Represents the Y-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ◆ ScaleZ: The ratio of the depth of one pixel to one unit in an image.
  - ◆ OffsetZ: Represents the Z-direction displacement of an object in 3D space relative to a reference point or coordinate system.

- ◆ UnitX(mm): The unit of the image pixel X.
- ◆ UnitY(mm): The unit of the image pixel Y.
- ◆ UnitZ(mm): The unit of the image pixel Z.
- ◆ IsAllocated: " Allocated" represents allocating enough computer memory or resources to store point cloud data for further processing, displaying and manipulating point cloud.
- ZMap8Bit
  - ◆ Width: The width of the image.
  - ◆ Height: The height of the image.
  - ◆ Pitch: Refers to the continuous distance that each row of pixel data is stored in memory. If each pixel occupies N bytes, the spacing is equal to the width of the image (in pixels) multiplied by the number of bytes per pixel. For example, if a 16-bit image is 640 wide and 480 high, 16-bit occupies  $16/8 = 2$  bytes for each pixel, so the spacing is  $640 * 2 = 1280$ .
  - ◆ Bpp: bit depth, which indicates how accurate the color information stored by the pixel is. The numeric portion represents the number of bytes in each pixel that can be used to represent color. Higher bit depths generally indicate more color information and higher image quality.
  - ◆ ScaleX: The ratio of the width of one pixel to one unit in an image. For example, if the image unit is millimeters (mm) and the X ratio is 9, it means that the width of a pixel is equal to 9 mm.
  - ◆ OffsetZ: Represents the X-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ◆ ScaleY: The ratio of one pixel in an image to one unit.
  - ◆ OffsetY: Represents the Y-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ◆ ScaleZ: The ratio of the depth of one pixel to one unit in an image.
  - ◆ OffsetZ: Represents the Z-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ◆ UnitX(mm): The unit of the image pixel X.
  - ◆ UnitY(mm): The unit of the image pixel Y.
  - ◆ UnitZ(mm): The unit of the image pixel Z.
  - ◆ IsAllocated: " Allocated " represents allocating enough computer memory or resources to store point cloud data for further processing, displaying and manipulating point cloud.

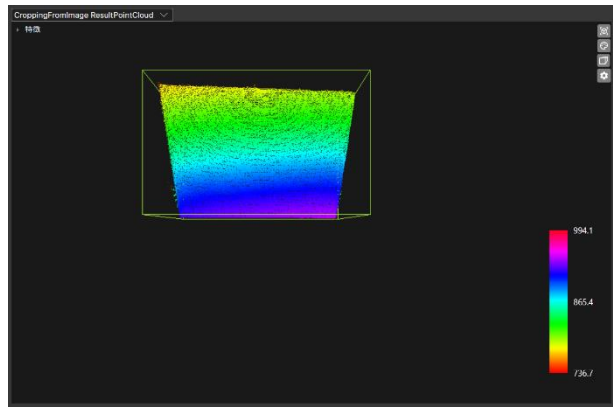
### 4.5.3 CroppingFromImage

Function: Obtains the point cloud within a specific plane region by setting a plane area.

Before execution

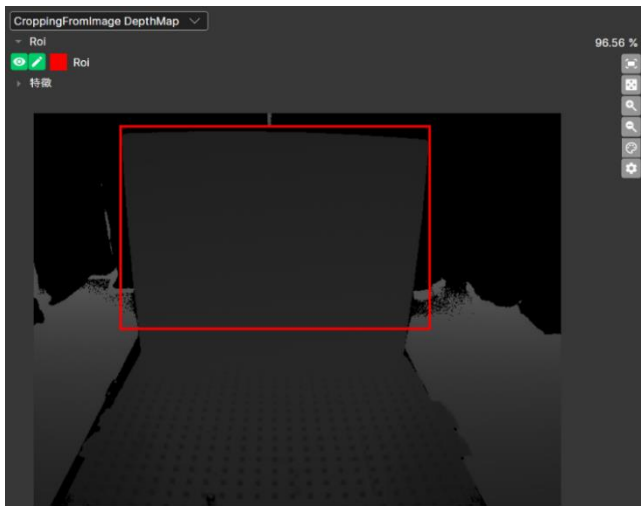


After execution

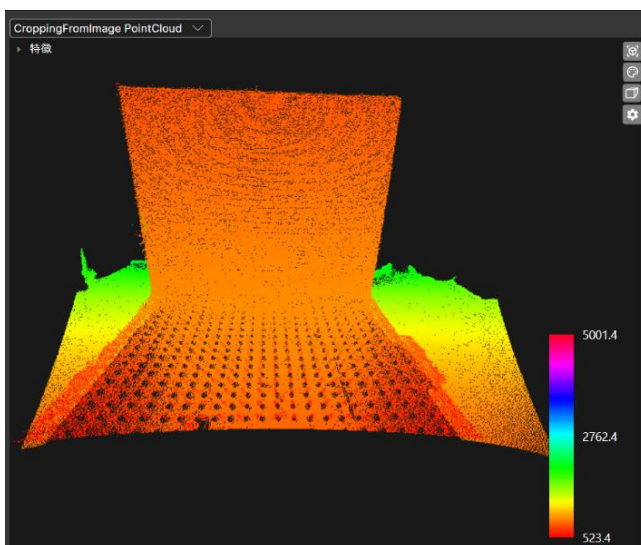


Input:

- Image: The source of the depth image.



- PointCloud: The source of the point cloud.





- ROI: Sets the crop range area, users can enter the range or set it through the depth image. The red area is shown below.



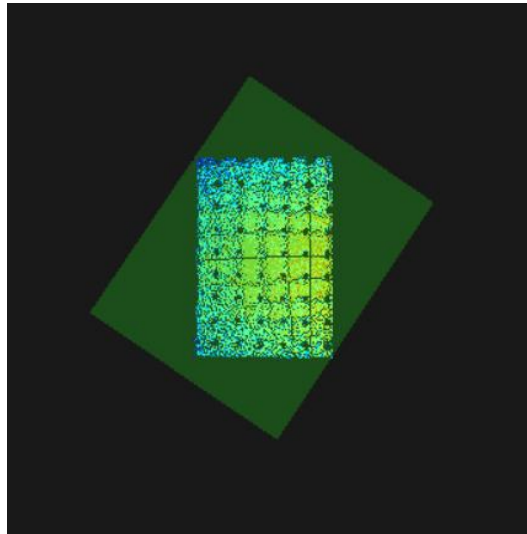
- Enable Value Filter: Enables the Z-range filter. When the user needs to set the Z range, it needs to be checked, and if it is not checked, it will be the Z full range.
- RangeStart: The start range of the Z coordinate, which is valid when Enable Value Filter is true. (Unit: mm)
- RangeEnd: The end range of the Z coordinate, which is valid when Enable Value Filter is true. (Unit: mm)

#### Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- ResultPointCloud: The cropped point cloud information.
  - ◆ Width: The width of the image after execution.
  - ◆ Height: The height of the image after execution.
  - ◆ Quantity: The number of point clouds after cropping.
  - ◆ IsAllocated: "Point cloud is allocated" represents allocating enough computer memory or resources to store point cloud data for further processing, displaying and manipulating point cloud.
  - ◆ HasNormal: The normal vector refers to the orientation of a point in three-dimensional space, perpendicular to the surface where the point is located. Indicates that each point is accompanied by information about the direction of its surface normal.
  - ◆ HasColor: Indicates that each point is accompanied by a color, and the color of the point is described in red, green, and blue.
- ResultPointCloudInfo: outputs the extreme value of the point cloud after cropping. Contains X, Y, Z extremum coordinates.

## 4.5.4 PlaneFitting

Function: Fits a plane from specific point cloud.



Input:

- PointCloud: Point cloud.

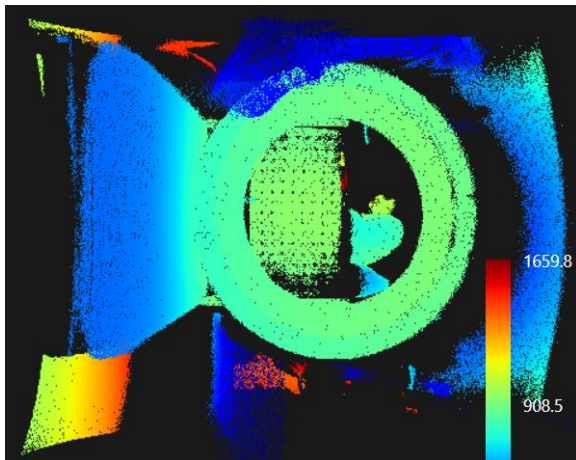
Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- IsFound: Calculates the fitting plane through the point cloud. If the calculation is successful, it returns True, otherwise it returns False.
- PlaneCoefficients: Plane equation  $Ax+By+Cz+D=0$ . The plane coefficients outputs A, B, C, D.
- NormalVector: Outputs the plane normal vector X, Y, and Z.

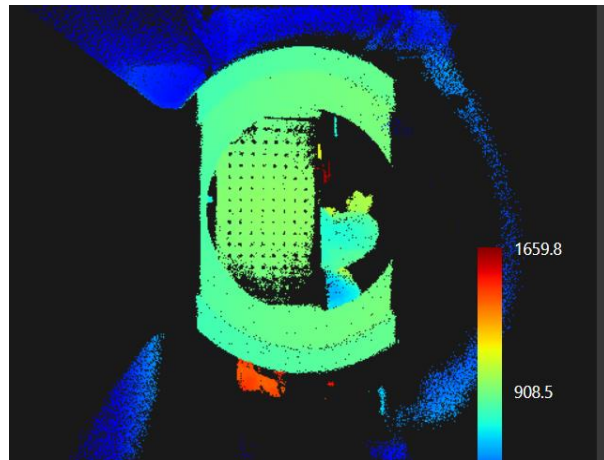
### 4.5.5 PassThrough

Function: Sets a range that will output the point cloud within or outside the range.

Before execution



After execution



#### Input:

- PointCloud: Point cloud.
- Field: Filters the direction of the dimension. Divided into X, Y and Z.
- LimitMin: The minimum value in the field space.
- LimitMax: The maximum value in the field space.
- LimitNegative: Sets to True to leave point clouds beyond the minimum and maximum values, and to False to leave points within the minimum and maximum values.

#### Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output
  - ◆ Width: The width of the point cloud.
  - ◆ Height: The height of the point cloud.
  - ◆ Count: The number of point clouds is width \* height.
  - ◆ IsAllocated: "Point cloud is allocated" represents allocating enough computer memory or resources to store point cloud data for further processing, displaying and manipulating point cloud.
  - ◆ HasNormal: The normal vector refers to the orientation of a point in 3D space, perpendicular to the surface where the point is located. Indicates that each point is accompanied by information about the direction of its surface normal.
  - ◆ HasColor: Indicates that each point is accompanied by a color, and the color of the point is described in red, green, and blue.

## 4.5.6 ProjectToPlane

**Function:** Projects a point cloud to one of the planes in space, such as XY, YZ, and XZ planes.

**Input:**

- PointCloud: Point cloud.
- ProjectionPlane: Projects to the plane direction, such as XY, YZ, XZ plane.
- ImageWidth: The width of the output image.
- ImageHeight: The height of the output image.
- Mode: Sets the projection method
  - ◆ Auto: Automatic projection.
  - ◆ PixelSize: Projection is performed by setting the pixel size and the center of the coordinates.
    - PixelSize(mm): Coordinate size per unit of pixel.
    - ProjectionCenter X(Y): According to the projection plane, enter the X (Y) coordinates of the projection center in the three-dimensional space.
    - ProjectionCenter Y(Z): According to the projection plane, enter the Y (Z) coordinates of the projection center in the three-dimensional space.
  - ◆ BoundingBox
    - XLimitMin: The minimum value of the X range limit in 3D space.
    - XLimitMax: The maximum value of the X range limit in 3D space.
    - YLimitMin: The minimum value of the Y range limit in 3D space.
    - YLimitMax: The maximum value of the Y range limit in 3D space.
    - ZLimitMin: The minimum value of the Z range limit in 3D space.
    - ZLimitMax: The maximum value of the Z range limit in 3D space.

**Output:**

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- OutputImage16Bit
  - ◆ Width: The width of the image.
  - ◆ Height: The height of the image.
  - ◆ Pitch: Refers to the continuous distance that each row of pixel data is stored in memory. If each pixel occupies N bytes, the spacing is equal to the width of the image (in pixels) multiplied by the number of bytes per pixel. For example, if a 16-bit image is 640 wide and 480 high, 16-bit occupies  $16/8 = 2$  bytes for each pixel, so the spacing is  $640 * 2 = 1280$ .
  - ◆ Bpp: bit depth, which indicates how accurate the color information stored by the pixel is. The numeric portion represents the number of bytes in each pixel that can be used

to represent color. Higher bit depths generally indicate more color information and higher image quality.

- ◆ ScaleX: The ratio of the width of one pixel to one unit in an image. For example, if the image unit is millimeters (mm) and the X ratio is 9, it means that the width of a pixel is equal to 9 mm.
  - ◆ OffsetX: Represents the X-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ◆ ScaleY: The ratio of one pixel in an image to one unit.
  - ◆ OffsetY: Represents the Y-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ◆ ScaleZ: The ratio of the depth of one pixel to one unit in an image.
  - ◆ OffsetZ: Represents the Z-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ◆ UnitX(mm): The unit of the image pixel X.
  - ◆ UnitY(mm): The unit of the image pixel Y.
  - ◆ UnitZ(mm): The unit of the image pixel Z.
  - ◆ IsAllocated: "IsAllocated" represents allocating enough computer memory or resources to store image data for further processing, displaying and manipulating images.
- OutputImage8Bit
    - ◆ Width: The width of the image.
    - ◆ Height: The height of the image.
    - ◆ Pitch: Refers to the continuous distance that each row of pixel data is stored in memory. If each pixel occupies N bytes, the spacing is equal to the width of the image (in pixels) multiplied by the number of bytes per pixel. For example, if a 16-bit image is 640 wide and 480 high, 16-bit occupies  $16/8 = 2$  bytes for each pixel, so the spacing is  $640 * 2 = 1280$ .
    - ◆ Bpp: bit depth, which indicates how accurate the color information stored by the pixel is. The numeric portion represents the number of bytes in each pixel that can be used to represent color. Higher bit depths generally indicate more color information and higher image quality.
    - ◆ ScaleX: The ratio of the width of one pixel to one unit in an image. For example, if the image unit is millimeters (mm) and the X ratio is 9, it means that the width of a pixel is equal to 9 mm.
    - ◆ OffsetX: Represents the X-direction displacement of an object in 3D space relative to a reference point or coordinate system.
    - ◆ ScaleY: The ratio of one pixel in an image to one unit.
    - ◆ OffsetY: Represents the Y-direction displacement of an object in 3D space relative to a reference point or coordinate system.
    - ◆ ScaleZ: The ratio of the depth of one pixel to one unit in an image.
    - ◆ OffsetZ: Represents the Z-direction displacement of an object in 3D space relative to a

reference point or coordinate system.

- ◆ UnitX(mm): The unit of the image pixel X.
- ◆ UnitY(mm): The unit of the image pixel Y.
- ◆ UnitZ(mm): The unit of the image pixel Z.
- ◆ IsAllocated: "IsAllocated" represents allocating enough computer memory or resources to store image data for further processing, displaying and manipulating images.

### 4.5.7 GetPointXYZ

Function: Retrieves the 3D coordinates (X, Y, Z) based on the X and Y index of the point cloud.

Input:

- PointCloud: Point cloud.
- PositionX: X index of the point cloud.
- PositionY: Y index of the point cloud.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Point
  - ◆ X (mm): The X coordinate of the selected location.
  - ◆ Y (mm): The Y coordinate of the selected location.
  - ◆ Z (mm): The Z coordinate of the selected location.

### 4.5.8 StatisticalOutlierRemoval

Function: Uses statistical methods to identify and filter outliers that do not conform to the statistical characteristics of the surrounding points in the point cloud. (max distance = average distance + standard deviation multiplier threshold \* std.deviation).

Input:

- MeanK: Averages the estimated number of points.
- PointCloud: The source of the point cloud image.
- StdevMulThresh: Standard deviation multiplier threshold.

Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.

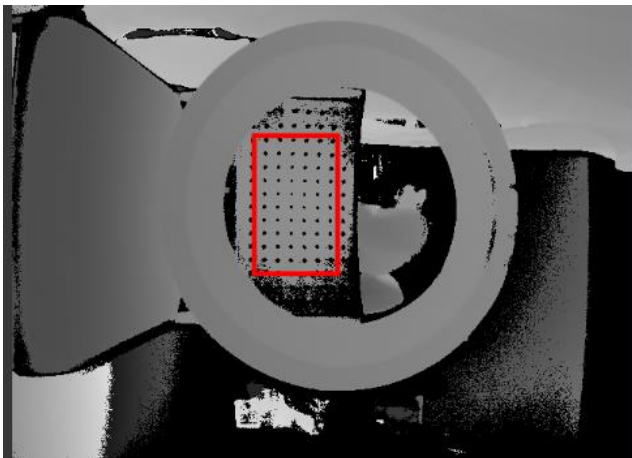
- LastRunTime: Current run time of the function block.
- Output
  - ◆ Width: The width of the image.
  - ◆ Height: The height of the image.
  - ◆ Count: The number of point clouds is width \* height.
  - ◆ IsAllocated: "Point cloud is allocated" represents allocating enough computer memory or resources to store point cloud data for further processing, displaying and manipulating point clouds.
  - ◆ HasNormal: The normal vector refers to the orientation of a point in 3D space, perpendicular to the surface where the point is located. Indicates that each point is accompanied by information about the direction of its surface normal.
  - ◆ HasColor: Indicates that each point is accompanied by a color, and the color of the point is described in red, green, and blue.

## 4.6 Z-image Processing

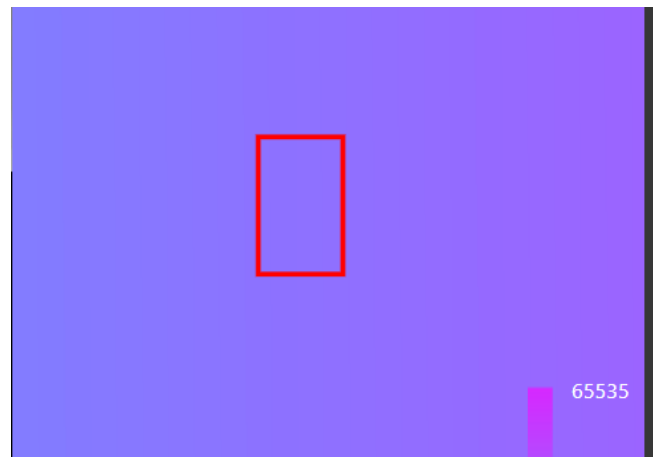
### 4.6.1 PlaneFitting

Function: Fits a plane from the depth map.

Before execution



After execution



Input:

- Image: Enter a depth image for plane fitting.
- ROI: Sets the fitting range.
- Enable Value Filter: Enables the option means that this function will fit within the range.
  - ◆ RangeStart: The starting value in the Z direction.
  - ◆ RangeEnd: The end value in the Z direction.
- IsGenerateImage: Whether to output an image of the planar coefficients.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.

- Status: The current status. “true” represents normal execution, “false” represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- IsFound: Calculates the fitting plane through the depth map. If the calculation is successful, it returns True, otherwise it returns False.
- PlaneCoefficients: Plane equation  $Ax+By+Cz+D=0$ . The plane coefficients outputs A, B, C, D.
- NormalVector: Outputs the plane normal vector X, Y, and Z parameters.
- OutputImage
  - ◆ Width: The width of the image.
  - ◆ Height: The height of the image.
  - ◆ Pitch: Refers to the continuous distance that each row of pixel data is stored in memory. If each pixel occupies N bytes, the spacing is equal to the width of the image (in pixels) multiplied by the number of bytes per pixel. For example, if a 16-bit image is 640 wide and 480 high, 16-bit occupies  $16/8 = 2$  bytes for each pixel, so the spacing is  $640 * 2 = 1280$ .
  - ◆ Bpp: bit depth, which indicates how accurate the color information stored by the pixel is. The numeric portion represents the number of bytes in each pixel that can be used to represent color. Higher bit depths generally indicate more color information and higher image quality.
  - ◆ ScaleX: The ratio of the width of one pixel to one unit in an image. For example, if the image unit is millimeters (mm) and the X ratio is 9, it means that the width of a pixel is equal to 9 mm.
  - ◆ OffsetX: Represents the X-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ◆ ScaleY: The ratio of one pixel in an image to one unit.
  - ◆ OffsetY: Represents the Y-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ◆ ScaleZ: The ratio of the depth of one pixel to one unit in an image.
  - ◆ OffsetZ: Represents the Z-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ◆ UnitX(mm): The unit of the image pixel X.
  - ◆ UnitY(mm): The unit of the image pixel Y.
  - ◆ UnitZ(mm): The unit of the image pixel Z.
  - ◆ IsAllocated: Represents allocating enough computer memory or resources to store image data for further processing, displaying and manipulating images.



## 4.7 Image Processing

### 4.7.1 Remap

Function: Converts images to 8-bit or 16-bit images.

Input:

- Image: Inputs 8-bit or 16-bit image.
- Output Image: Outputs 8-bit or 16-bit image.
- Mode: Auto mode or custom mode.
  - ◆ RangeStart: Pixel value greater than Range Start will be included in the calculation.
  - ◆ RangeEnd: Pixel value less than Range End will be included in the calculation.
  - ◆ IsFillSaturatedValue: If it is checked, pixel value less than the Range Start will be filled in Range Start, and pixel value greater than the Range End will be filled in Range End.

Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output Image
  - ◆ Width: The width of the image.
  - ◆ Height: The height of the image.
  - ◆ Pitch: Refers to the continuous distance that each row of pixel data is stored in memory. If each pixel occupies N bytes, the spacing is equal to the width of the image (in pixels) multiplied by the number of bytes per pixel. For example, if a 16-bit image is 640 wide and 480 high, 16-bit occupies  $16/8 = 2$  bytes for each pixel, so the spacing is  $640 * 2 = 1280$ .
  - ◆ BPP: bit depth, which indicates how accurate the color information stored by the pixel is. The numeric portion represents the number of bytes in each pixel that can be used to represent color. Higher bit depths generally indicate more color information and higher image quality.
  - ◆ X-Scale: The ratio of the width of one pixel to one unit in an image. For example, if the image unit is millimeters (mm) and the X ratio is 9, it means that the width of a pixel is equal to 9 mm.
  - ◆ X-offset: Represents the X-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ◆ Y-scale: The ratio of one pixel in an image to one unit.
  - ◆ Y offset: Represents the Y-direction displacement of an object in 3D space relative to a reference point or coordinate system.
  - ◆ Z-Scale: The ratio of the depth of one pixel to one unit in an image.

- ◆ Z offset: Represents the Z-direction displacement of an object in 3D space relative to a reference point or coordinate system.
- ◆ X unit(mm): The unit of the image pixel X.
- ◆ Y unit(mm): The unit of the image pixel Y.
- ◆ Z Unit(mm): The unit of the image pixel Z.
- ◆ Configuration: Refers to the allocation of enough memory or resources in the computer's memory to store the data of the image. Images must be configured to process, display, or manipulate images.

## 4.7.2 Area

Function: Converts images to black and white through binarization, and counts the number of black or white pixels. The software draws the feature area in red.



Input:

- Image: Inputs 8-bit or 16-bit image.
- ROI: Sets the calculation area range.
- ValueUpper: Pixel value less (inclusive) than Value Upper is valid.
- ValueLower: Pixel value greater (inclusive) than Value Lower is valid.
- ObjectType: Calculates the total area of white object (valid) or black object (invalid)
- IsGenerateFeature: The feature will be drawn on the image when it is set to True.

True



False



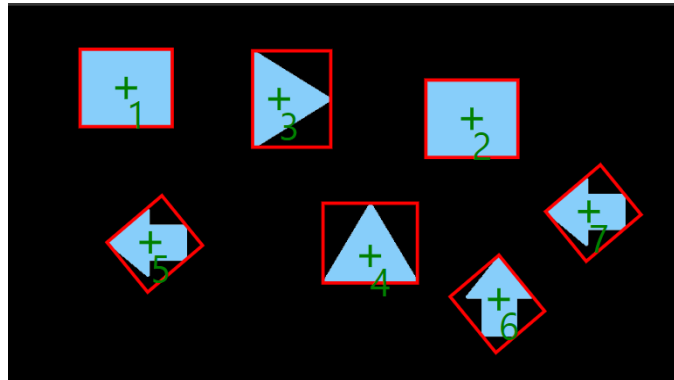
Output

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.

- LastRunTime: Current run time of the function block.
- Area Count: The calculated area.

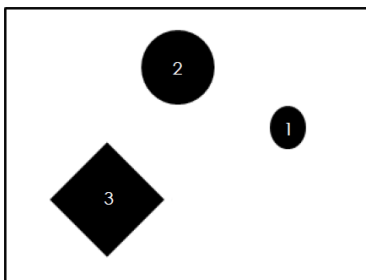
### 4.7.3 Blob

Function: The image is converted to black and white through binarization. Grouping the connected pixels into blobs.

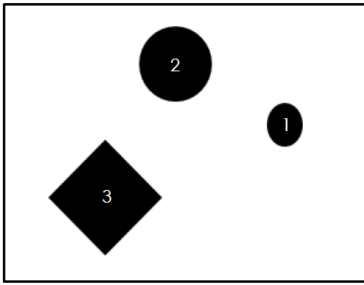


Input:

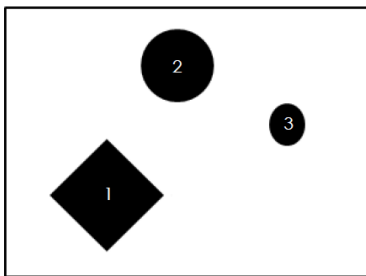
- ValueUpper: Pixel value less (inclusive) than ValueUpper is considered. If it is a regular image, the grayscale value is set. If it is a depth image, the value represents the distance, measured in millimeters.
- Value Lower: Pixel value greater (inclusive) than ValueLower is considered. If it is a regular image, the grayscale value is set. If it is a depth image, the value represents the distance, measured in millimeters.
- MaxCount: The maximum number of blobs to search for.
- ObjectType: Can be set to search for white or black blobs.
- Sorting Type:
  - ◆ AreaAscending: Sorts blobs based on their area in ascending order, from smallest to largest.



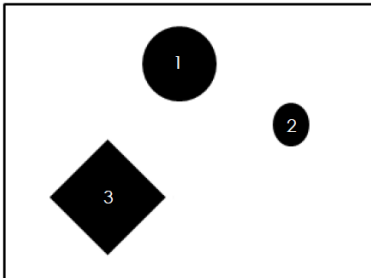
- ◆ AreaDescending: Sorts blobs based on their area in descending order, from largest to smallest.
- ◆ PerimeterAscending: Sorts blobs based on their perimeter in ascending order, from smallest to largest.



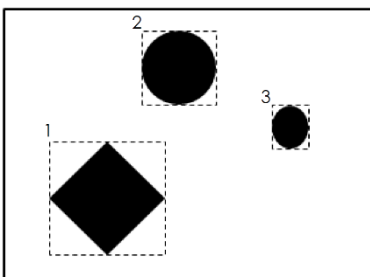
- ◆ **PerimeterDescending:** Sorts blobs based on their area in descending order, from largest to smallest.
- ◆ **CenXAscending:** Sorts blobs based on their center X coordinates in ascending order, from smallest to largest according to X coordinates.



- ◆ **CenXDescending:** Sorts blobs based on their center X coordinates in descending order, from largest to smallest according to X coordinates.
- ◆ **CenYAscending:** Sorts blobs based on their center Y coordinates in ascending order, from smallest to largest according to Y coordinates.

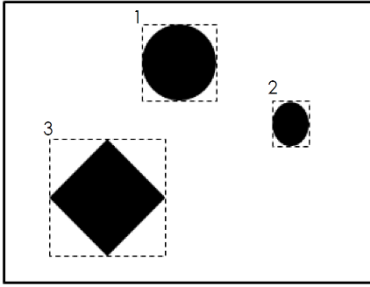


- ◆ **CenYDescending:** Sorts blobs based on their center Y coordinates in descending order, from largest to smallest according to Y coordinates.
- ◆ **BboxXAscending:** Sorts blobs based on their bounding box X coordinates in ascending order, from smallest to largest according to X coordinates.

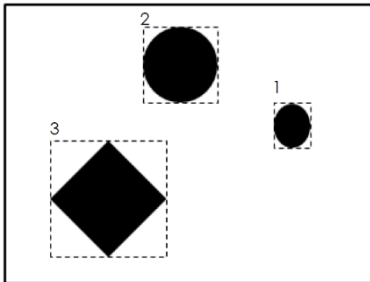


- ◆ **BboxXDescending:** Sorts blobs based on their bounding box X coordinates in descending order, from largest to smallest according to X coordinates.
- ◆ **BboxYAscending:** Sorts blobs based on their bounding box Y coordinates in ascending order.

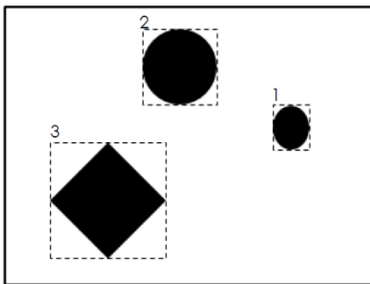
order, from smallest to largest according to Y coordinates.



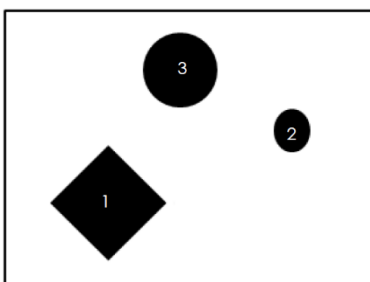
- ◆ BboxYDescending: Sorts blobs based on their bounding box Y coordinates in descending order, from largest to smallest according to Y coordinates.
- ◆ BboxWAscending: Sorts blobs based on their bounding box width in ascending order, from smallest to largest.



- ◆ BboxWDescending: Sorts blobs based on their bounding box width in descending order, from largest to smallest
- ◆ BboxHAscending: Sorts blobs based on their bounding box height in ascending order, from smallest to largest.

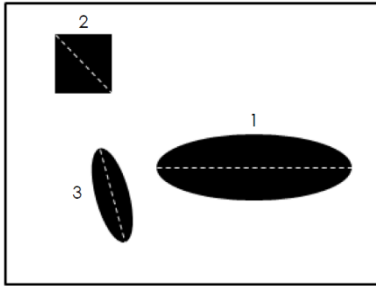


- ◆ BboxHDescending: Sorts blobs based on their bounding box height in descending order, from largest to smallest.
- ◆ RoundnessAscending: Sorts blobs based on their roundness in ascending order, from smallest to largest.



- ◆ RoundnessDescending: Sorts blobs based on their roundness in descending order, from largest to smallest.
- ◆ MajAxisAngAscending: Sorts blobs based on their major axis angle in ascending order,

from smallest to largest.



- ◆ **MajAxisAngDescending:** Sorts blobs based on their major axis angle in descending order, from largest to smallest.
- ◆ **MajAxisLenAscending:** Sorts blobs based on their major axis length in ascending order, from smallest to largest.
- ◆ **MajAxisLenDescending:** Sorts blobs based on their major axis length in descending order, from largest to smallest.
- **AreaUpper:** Sets the maximum area of blob (pixel or mm<sup>2</sup>). Greater than this value will not be considered as a blob.
- **AreaLower:** Sets the minimum area of blob (pixel or mm<sup>2</sup>). Less than this value will not be considered as a blob.
- **HoleAreaUpper:** Sets the maximum area of hole (pixel or mm<sup>2</sup>). If **IsFillHole** is true, the holes which area within **HoleAreaUpper** and **HoleAreaLower** will be filled.
- **HoleAreaLower:** Sets the minimum area of hole (pixel or mm<sup>2</sup>). If **IsFillHole** is true, the holes which area within **HoleAreaUpper** and **HoleAreaLower** will be filled.
- **PerimeterUpper:** Sets the maximum perimeter of blob (pixel or mm). Greater than this value will not be considered as a blob.
- **PerimeterLower:** Sets the minimum perimeter of blob (pixel or mm). Less than this value will not be considered as a blob.
- **MajAxisUpper:** Sets the maximum major axis length of blob. (pixel or mm). Greater than this value will not be considered as a blob.
- **MajAxisLower:** Sets the minimum major axis length of blob. (pixel or mm). Less than this value will not be considered as a blob.
- **Image:** Inputs the image to be inspected, which can be a camera source or a virtual image.
- **ROI:** Selects the range to be inspected by setting the ROI, and the ROI shapes you can choose include rectangle, circle, fan, oval, ring, polygon, rotating rectangle, etc.
- **MinSegmentLength:** Defines how many consecutive pixel values are for blobs.
- **RoundnessUpper:** Sets the maximum roundness of blob. Greater than this value will not be considered as a blob.
- **RoundnessLower:** Sets the minimum roundness of blob. Less than this value will not be considered as a blob.
- **CalculateMinRect:** If set to True, it will calculate the minimum bounding rectangle, and vice versa.
- **IsGenerateFeatureImage:** If set to True, it will calculate and output feature map, and vice versa.

versa.

- **IsFillingHole:** If set to True, it will use HoleAreaUpper and HoleAreaLower to fill the hole, and vice versa.
- **IsFullAngle:** If set to True, the output blob angle range is -180 ~ 180. If set to False, the angle range is -90 ~ 90.
- **CalculateMajorAxis:** If set to True, it will calculate the blob major axis, and vice versa.

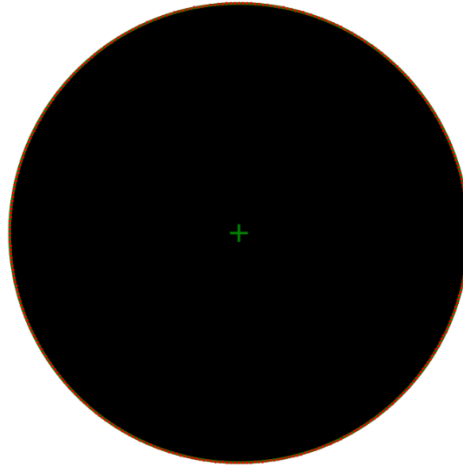
Output:

- **Name:** The name of the function block.
- **Type Name:** The name of the component type.
- **Status:** The current status. "true" represents normal execution, "false" represents abnormal execution.
- **ErrorMessage:** Shows the reasons when the execution is abnormal.
- **LastRunTime:** Current run time of the function block.
- **Count:** The number of output blob.
- **Result:** Outputs information for each blob.
  - ◆ **Position**
    - X: X coordinate of the mass center of the blob (pixel or mm).
    - Y: Y coordinate of the mass center of the blob (pixel or mm).
  - ◆ **BoundingBox**
    - **Center**
      - X: X coordinate of the bounding box (pixel or mm).
      - Y: Y coordinate of the bounding box (pixel or mm).
    - **Width:** The width of the bounding box
    - **Height:** The height of the bounding box.
    - **Angle:** The angle of the bounding box. (0)
  - ◆ **MinimumAreaRectangle**
    - **Center**
      - X: the center X coordinate of the minimum area rectangle.
      - Y: the center Y coordinate of the minimum area rectangle.
    - **Width:** The width of the minimum area rectangle.
    - **Height:** The height of the minimum area rectangle.
    - **Angle:** The angle of the minimum area rectangle.
  - ◆ **MajorAxisAnge:** the angle between the major axis and the y=0 segment.
  - ◆ **MajorAxisLength:** the length of the major axis of the blob (pixel or mm).
  - ◆ **Roundness:** The ratio of the speck profile to the ideal circle, the closer to 100 represents the closer to the ideal circle.
  - ◆ **Perimeter:** The perimeter of the blob (pixel or mm).
  - ◆ **Area:** The area of the blob (pixel or mm<sup>2</sup>).
  - ◆ **Frame**
    - X: The blob X coordinate in the image (pixel).

- Y: The blob Y coordinate in the image (pixel).
- Angle: The blob angle between major axis and the X axis in the image.

#### 4.7.4 CircleFitting

Function: Fits a circle.



Input:

- Image: Selects the image input to be inspected, which can be a camera source or a virtual image.
- ROI: Sets the detection range, which is divided into ring and arc.
  - ◆ CenX: ROI, X center location. The unit is Pixel.
  - ◆ CenY: ROI, Y center location. The unit is Pixel.
  - ◆ OuterRadius: ROI, outer diameter. The unit is Pixel.
  - ◆ InnerRadius: ROI, inner diameter. The unit is Pixel.
  - ◆ ReferenceFrame: Sets the ROI according to the conversion coordinate system of other processes.
- BlockShift: Sets the spacing between the previous detection point and the next detection point.
- BlockWidth: Sets the angle to take a point, the value adjustment range is 1~50; light blue is the block found, the following figure (1) set the value of 1 can see that the number of blocks found is a lot; the following figure (2) set the value of 50 can see that only 8 light blue dots are found.



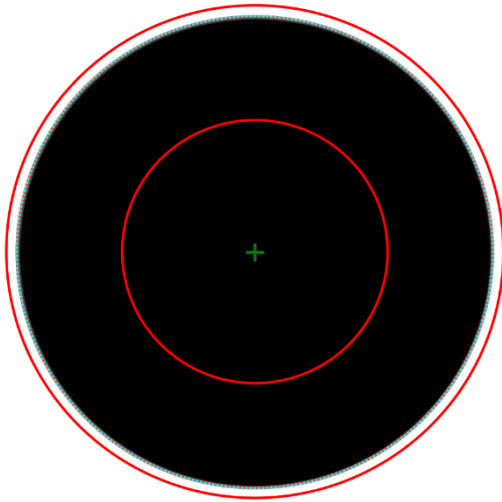


Figure (1) The block width is set to 1

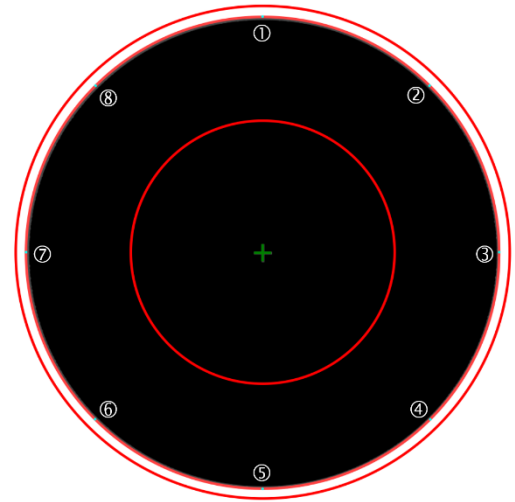


Figure (2) The block width is set to 50

- SmoothLevel: The range of the average edge gray scale value can be set to 1~30, and when the smoothing level is amplified, the slope of the waveform will change, so as to dilute the effect of noise.

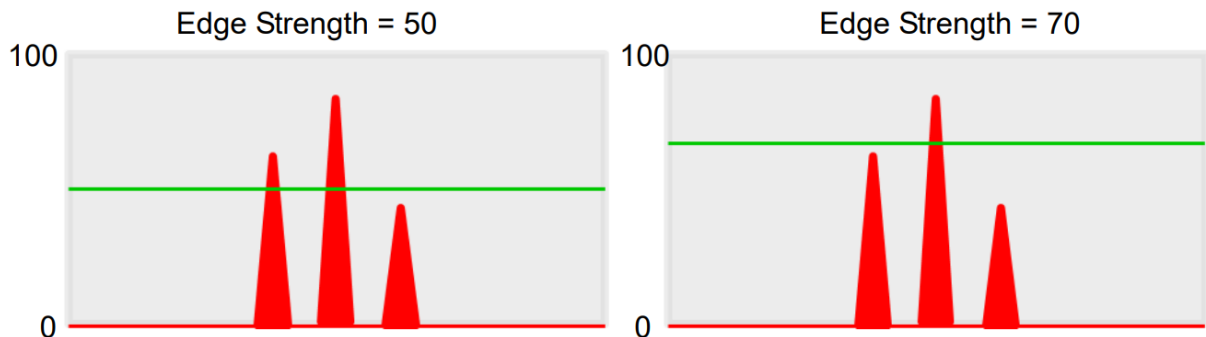


- NoiseLevel: The purpose of noise level is mainly to filter out some small noise on the image, the parameter adjustment range of noise level is 0~255, when the waveform is less than the set value, it will disappear in the waveform diagram, as shown in the figure below when the left noise level is set to 0, you can see that some noise appears, when the noise level is set to 20, the noise will be filtered out.



- EdgeStrength: The edge strength is the threshold of edge finding, the numerical adjustment range of edge strength is 0~100, the basis of adjustment can be fine-tuned by referring to

the waveform shown in the preview figure, when the waveform is greater than the set edge strength, the edge position tool will identify the block as an edge, such as the bottom left figure when the edge strength is set to 50, the system will find two edges, and the bottom right figure when the edge strength is set to 70, there is only one edge that is greater than the threshold.



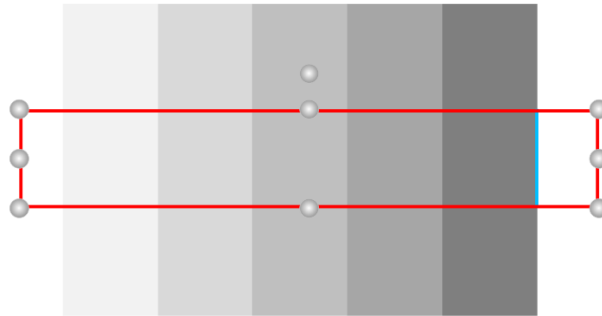
- EdgeType: This parameter will refer to the scan direction and set a boundary from white to black or black to white, or both.
- ScanDirection: From the inside to the outside, the inner diameter is scanned to the outer diameter. Outside-to-inside is the outer diameter scanned to the inner diameter.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- IsFound: Whether to fit a circle. True is a detected fit circle, otherwise it is False.
- Result
  - ◆ Position
    - X: X coordinate of the center of the fitted circle.
    - Y: Y coordinate of the center of the fitted circle.
    - Radius: The radius of the fitted circle.
    - Roundness: Fits the roundness of the fitted circle.
  - ◆ Frame
    - X: Fits the X coordinate of the center of the circle after coordinate conversion.
    - Y: Fits the Y coordinate of the center of the circle after coordinate conversion.
    - Angle: The angle between the center of the fitting circle and the X-axis after coordinate conversion.

## 4.7.5 EdegPosition

Function: Finds the edge of objects according to illumination change.

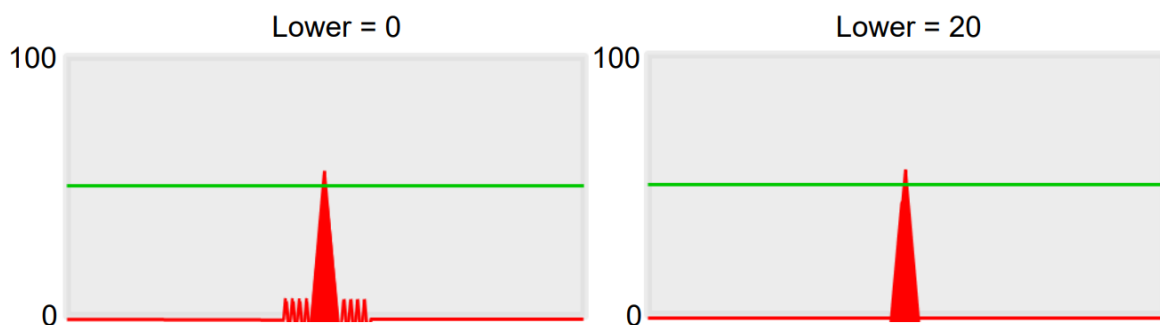


Input:

- Image: Selects the image input to be inspected, which can be a camera source or a virtual image.
- ROI: Selects the range to be inspected by setting the ROI. The ROI type includes rotating rectangles, rings, and arcs.
- SmoothLevel: The range of the average edge gray scale value can be set to 1~30, and when the smoothing level is amplified, the slope of the waveform will change, so as to dilute the effect of noise.



- NoiseLevel: The purpose of noise level is mainly to filter out some small noise on the image, the parameter adjustment range of noise level is 0~255, when the waveform is less than the set value, it will disappear in the waveform diagram, as shown in the figure below when the left noise level is set to 0, you can see that some noise appears, when the noise level is set to 20, the noise will be filtered out.



- EdgeStrength: The edge strength is the threshold of edge finding, the numerical adjustment

range of edge strength is 0~100, the basis of adjustment can be fine-tuned by referring to the waveform shown in the preview figure, when the waveform is greater than the set edge strength, the edge position tool will identify the block as an edge, such as the bottom left figure when the edge strength is set to 50, the system will find two edges, and the bottom right figure when the edge strength is set to 70, there is only one edge left that is greater than the threshold.



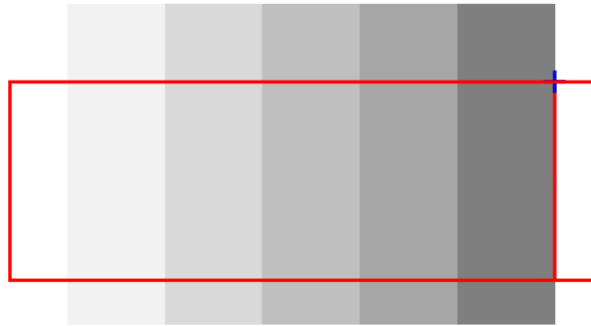
- EdgeType: This parameter will refer to the scan direction and set a boundary from white to black or black to white, or both.
- ScanDirection: Sets the scanning direction according to the ROI setting method. The rotation rectangle supports top-to-bottom, bottom-to-top, left-to-right, and right-to-left scanning. Rings and arcs support clockwise and counterclockwise scanning.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- EdgeCount: Outputs the number of edges detected.
- Results
  - ◆ X: X coordinate of edge point center.
  - ◆ Y: Y coordinate of edge point center.
  - ◆ Angle: The angle between the edge and Y=0. Negative clockwise and positive counterclockwise.

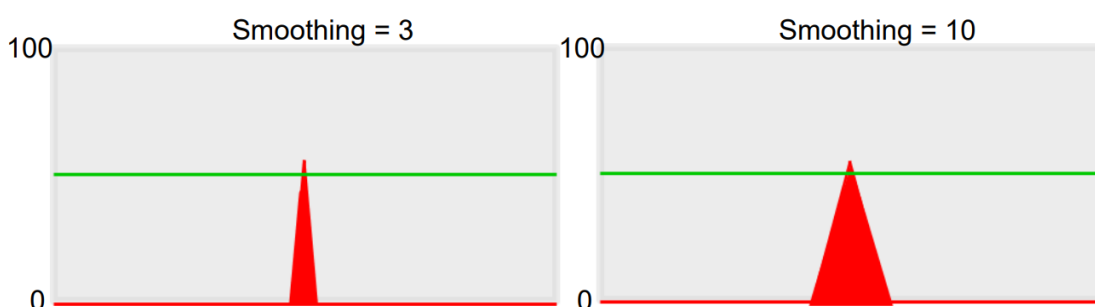
### 4.7.6 EdgePosTracing

Function: The position tracking tool is an advanced function of the edge position tool, the search method is the same as the edge position tool. Taking the rectangle ROI as an example, the search method can be set and the edge can be found by scanning the horizontal/vertical direction, when the block with a large difference in light and dark on the image is detected, this tool identifies the area as one of the edges to be found, the feature of this tool is that it can find several edge points on a plane at the same time and output coordinates.

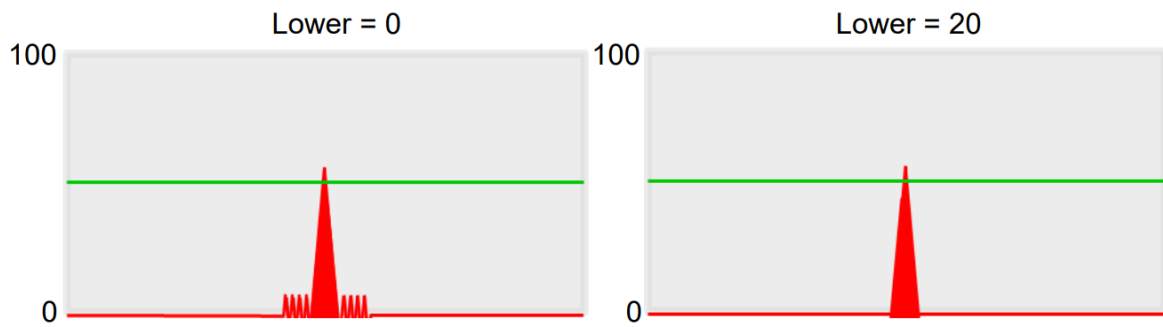


Input:

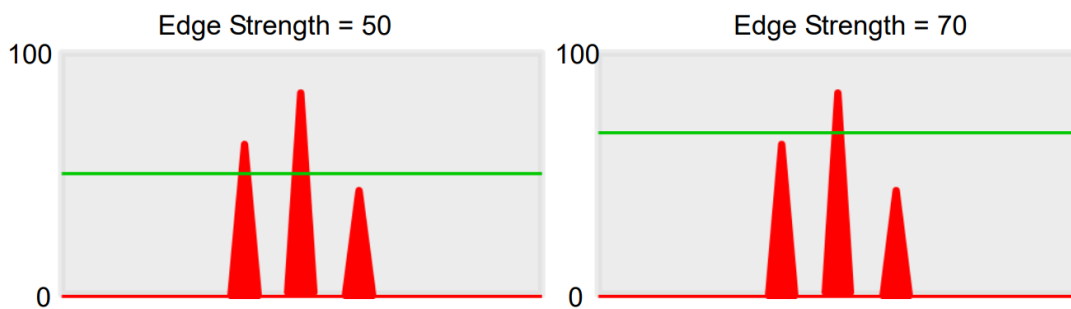
- Image: Selects the image input to be inspected, which can be a camera source or a virtual image.
- ROI: Selects the range to be inspected by setting the ROI.
- BlockShift: When the block width is set to 4 and the block shift is set to 1, after each time the system scans the edge with a width of 4 pixels, the system will perform a lateral displacement of 1 pixel, and then perform a 4-pixel edge scan again.
- BlockWidth: The block width is the horizontal pixel width for each system scan.
- SmoothLevel: The range of the average edge gray scale value can be set to 1~30, and when the smoothing level is amplified, the slope of the waveform will change, so as to dilute the effect of noise.



- NoiseLevel: The purpose of noise level is mainly to filter out some small noise on the image, the parameter adjustment range of noise level is 0~255, when the waveform is less than the set value, it will disappear in the waveform diagram, as shown in the figure below when the left noise level is set to 0, you can see that some noise appears, when the noise level is set to 20, the noise will be filtered out.



- **EdgeStrength:** The edge strength is the threshold of edge finding, the numerical adjustment range of edge strength is 0~100, the basis of adjustment can be fine-tuned by referring to the waveform shown in the preview figure, when the waveform is greater than the set edge strength, the edge position tool will identify the block as an edge, such as the bottom left figure when the edge strength is set to 50, the system will find two edges, and the bottom right figure when the edge strength is set to 70, there is only one edge left that is greater than the threshold.



- **EdgeType:** This parameter will refer to the scan direction and set a boundary from white to black or black to white, or both.
- **ScanDirection:** Sets the scanning direction according to the ROI setting method, including top-to-bottom scanning, bottom-to-top scanning, left-to-right, right-to-left and other directions.

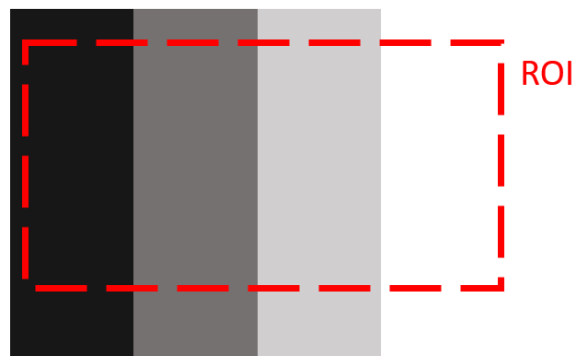
#### Output:

- **Name:** The name of the function block.
- **TypeName:** The name of the component type.
- **Status:** The current status. "true" represents normal execution, "false" represents abnormal execution.
- **ErrorMessage:** Shows the reasons when the execution is abnormal.
- **LastRunTime:** Current run time of the function block.
- **Ave. Dist.:** Calculates the distance from the average point of all edge points to the edge of the ROI.
- **Average Edge X:** The sum of the points X average distance.
- **Average Edge Y:** The sum of the points Y average distance.
- **Edge Count:** The number of counts is the total number of detected points.

- Max Edge X: Outputs the Max X coordinate.
- Max Edge Y: Outputs the Max Y coordinate.
- Min Edge X: Outputs the minimum X coordinate.
- Min Edge Y: Outputs the minimum Y coordinate.
- Max Dist.: Calculates the distance from the maximum edge point to the edge of the ROI.
- Max Dist. Edge Point
  - ◆ X: X coordinate of the maximum distance from the edge.
  - ◆ Y: Y coordinate of the maximum distance from the edge.
- Min Dist.: Calculates the distance from the minimum edge point to the edge of the ROI.
- Min Dist. Edge Point
  - ◆ X: X coordinate of the minimum distance from the edge.
  - ◆ Y: Y coordinate of the minimum distance from the edge.
- Results
  - ◆ [n]: the coordinate information of the nth point.
    - X: X coordinate of point nth.
    - Y: Y coordinate of point nth.

### 4.7.7 Intensity

Function: The Image Intensity tool can measure the maximum brightness, minimum brightness, average brightness, and standard deviation brightness of grayscale images in the detection area.



Input:

- Image: Selects the image input to be inspected, which can be a camera source or a virtual image.
- ROI: Selects the range to be inspected by setting the ROI.
- ValueUpper: Sets the upper limit of the detection range. The unit is a grayscale value, or distance in mm.
- ValueLower: Sets the lower limit of the detection range. The unit is a grayscale value, or distance in mm.
- Mode: Calculates for numeric regions. The inner diameter is within the upper and lower numerical limits, and the outer diameter is outside the upper and lower numerical limits.

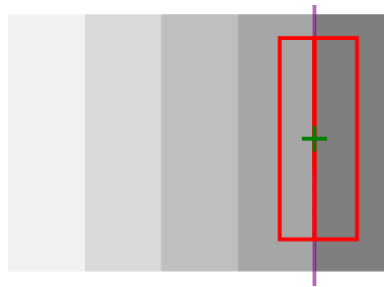
Output:

- Name: The name of the function block.

- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- IsFound: Whether there are values in the ROI that match the range of the values.
- Result
  - ◆ MinimumValue: The minimum value of the test result in the ROI.
  - ◆ MaximumValue: The maximum value of the test result in the ROI.
  - ◆ AverageValue: The average of the test results within the ROI.
  - ◆ StandardDeviation: The numerical standard deviation of the test result within the ROI.

### 4.7.8 LineFitting

Function: Fits a line.



Input:

- Image: Selects the image input to be inspected, which can be a camera source or a virtual image.
- ROI: Selects the range to be inspected by setting the ROI.
- BlockShift: When the block width is set to 4 and the block shift is set to 1, after each time the system scans the edge with a width of 4 pixels, the system will perform a lateral displacement of 1 pixel, and then perform a 4-pixel edge scan again.
- BlockWidth: The block width is the horizontal pixel width for each system scan.
- SmoothLevel: The range of the average edge gray scale value can be set to 1~30, and when the smoothing level parameter is amplified, the slope of the waveform will change, so as to reduce the effect of noise.



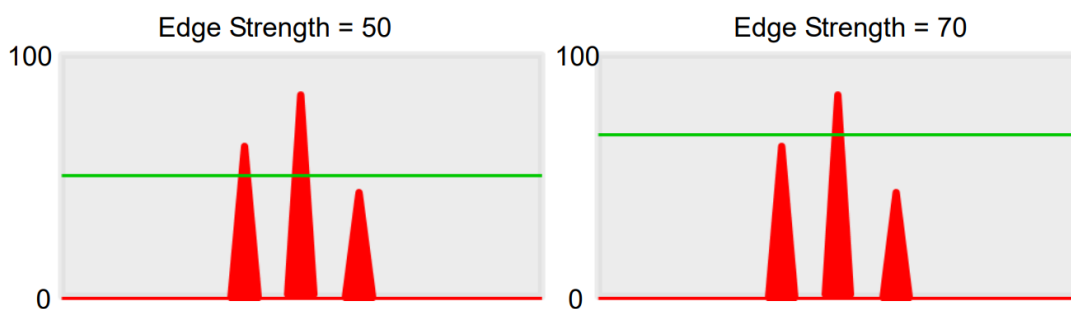
- NoiseLevel: The purpose of noise level is mainly to filter out some small noise on the image,



the parameter adjustment range of noise intensity is 0~255, when the waveform is less than the set value, it will disappear in the waveform diagram, as shown in the figure below when the left noise intensity is set to 0, you can see that some noise appears, when the noise intensity is set to 20, the noise will be filtered out.



- EdgeStrength:** The edge strength is the threshold value of edge finding, the numerical adjustment range of edge strength is 0~100, the basis of adjustment can be fine-tuned by referring to the waveform shown in the preview figure, when the waveform is greater than the set edge strength, the edge position tool will identify the block as an edge, such as the bottom left figure when the edge strength is set to 50, the system will find two edges, and the bottom right figure when the edge strength is set to 70, there is only one edge left that is greater than the threshold value.



- EdgeType:** This parameter will refer to the sweep direction and set a boundary from white to black or black to white, or both.
- ScanDirection:** Sets the scanning direction according to the ROI setting method, including top-to-bottom scanning, bottom-to-top scanning, left-to-right, right-to-left and other directions.

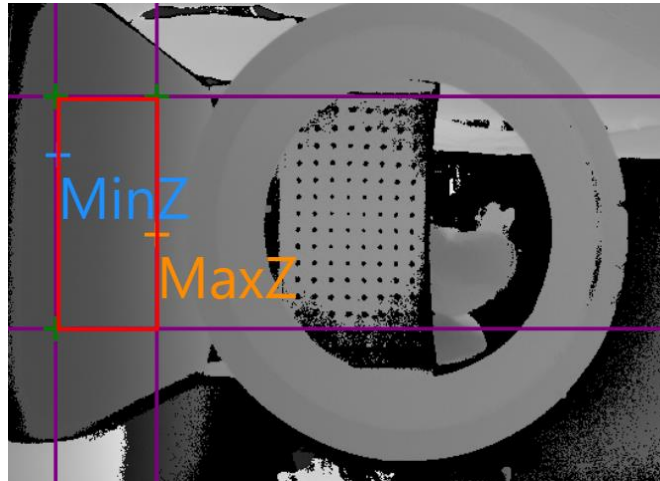
Output:

- Name:** The name of the function block.
- TypeName:** The name of the component type.
- Status:** The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage:** Shows the reasons when the execution is abnormal.
- LastRunTime:** Current run time of the function block.
- IsFound:** Whether the line segment is detected. True is detected, otherwise it is false.

- Line: Outputs A, B, and C coefficients of the  $AX+BY+C$  equation.
- MiddlePoint
  - ◆ X: Outputs the X coordinates of the midpoint of the fitted line segment.
  - ◆ Y: Outputs the Y coordinates of the midpoint of the fitted line segment.

### 4.7.9 Position

Function: The maximum and minimum coordinates of X, Y, and Z are calculated based on the values in ROI.



Input:

- Image: Selects the image input to be inspected, which can be a camera source or a virtual image.
- ROI: Selects the range to be inspected by setting the ROI.
- ValueUpper: Sets the upper limit of the detection value range. The unit is a grayscale value, or distance in mm.
- ValueLower: Sets the lower limit of the detection range. The unit is a grayscale value, or distance in mm.
- Mode: Calculates for numeric regions. The inner diameter is within the upper and lower numerical limits, and the outer diameter is outside the upper and lower numerical limits.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- AvgPosInfo
  - ◆ Point
    - X: The average value of X in the output ROI, which is the same as the image unit.
    - Y: The average value of Y in the output ROI, which is the same as the image unit.
    - Z: The average value of Z in the output ROI, which is the same as the image unit.

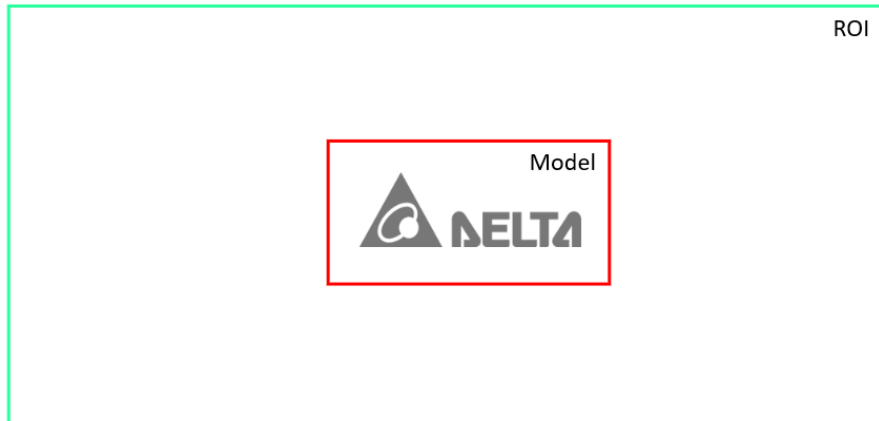
- ◆ Frame
  - X: The coordinates X of the average position in the image coordinate system, in Pixel.
  - Y: The coordinate Y of the average position in the image coordinate system, in Pixel.
  - Angle: The rotation angle between the average position and the X axis in the image coordinate system.
- ◆ Found: Whether there are values in the ROI that match the range of the values.
- MaxXInfo
  - ◆ Point
    - X: X coordinate of the maximum X coordinate point in the output ROI.
    - Y: Y coordinate of the maximum X coordinate point in the output ROI.
    - Z: Z coordinate of the maximum X coordinate point in the output ROI.
  - ◆ Frame
    - X: X coordinate of the largest X coordinate point in the output image coordinate system.
    - Y: Y coordinate of the largest X coordinate point in the output image coordinate system.
    - Angle: The rotation angle of the maximum X-coordinate point and the X-axis in the output image coordinate system.
- MaxYInfo
  - ◆ Point
    - X: Outputs the X coordinate of the maximum Y coordinate point.
    - Y: Outputs the Y coordinate of the maximum Y coordinate point.
    - Z: Outputs the Z coordinate of the maximum Y coordinate point.
  - ◆ Frame
    - X: X coordinate of the largest Y coordinate point in the output image coordinate system.
    - Y: Y coordinate of the largest Y coordinate point in the output image coordinate system.
    - Angle: The rotation angle between the maximum Y coordinate point and the X axis in the output image coordinate system.
- MaxZInfo
  - ◆ Point
    - X: Outputs the X coordinate of the maximum Z coordinate point.
    - Y: Outputs the Y coordinate of the maximum Z coordinate point.
    - Z: Outputs the Z coordinates of the largest Z coordinate point.
  - ◆ Frame
    - X: X coordinate of the largest Z coordinate point in the output image coordinate system.

- Y: Y coordinate of the largest Z coordinate point in the output image coordinate system.
- Angle: The rotation angle between the maximum Z coordinate point and the X axis in the output image coordinate system.
- **MinXInfo**
  - ◆ **Point**
    - X: Outputs the X coordinate of the smallest X coordinate point.
    - Y: Outputs the Y coordinate of the maximum Z coordinate point.
    - Z: Outputs the Z coordinate of the smallest X coordinate point.
  - ◆ **Frame**
    - X: X coordinate of the smallest X coordinate point in the output image coordinate system.
    - Y: Y coordinate of the smallest X coordinate point in the output image coordinate system.
    - Angle: The rotation angle between the smallest Z coordinate point and the X axis in the output image coordinate system.
- **MinYInfo**
  - ◆ **Point**
    - X: Outputs the X coordinate of the minimum Y coordinate point.
    - Y: Outputs the Y coordinate of the maximum Z coordinate point.
    - Z: Outputs the Z coordinate of the minimum Y coordinate point.
  - ◆ **Frame**
    - X: X coordinate of the smallest Y coordinate point in the output image coordinate system.
    - Y: Y coordinate of the smallest Y coordinate point in the output image coordinate system.
    - Angle: The rotation angle between the smallest Z coordinate point and the X axis in the output image coordinate system.
- **MinZInfo**
  - ◆ **Point**
    - X: Outputs the X coordinate of the minimum Z coordinate point.
    - Y: Outputs the Y coordinate of the maximum Z coordinate point.
    - Z: Outputs the Z coordinate of the minimum Z coordinate point.
  - ◆ **Frame**
    - X: X coordinate of the smallest Z coordinate point in the output image coordinate system.
    - Y: Y coordinate of the smallest Z coordinate point in the output image coordinate system.
    - Angle: The rotation angle between the smallest Z coordinate point and the X axis in the output image coordinate system.

- PixelCount: The total number of pixels output.

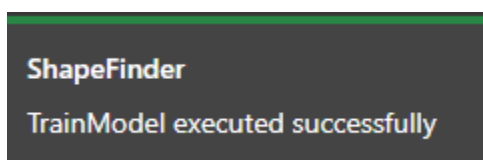
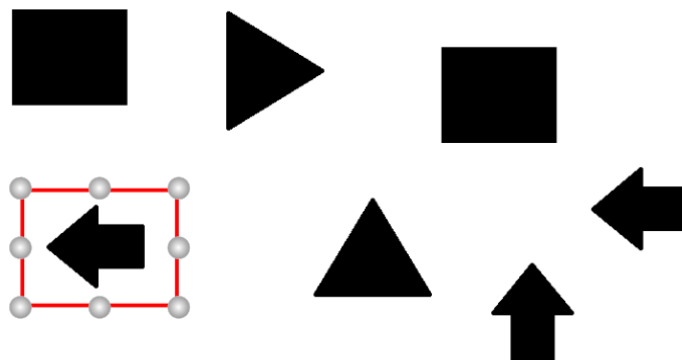
### 4.7.10 ShapeFinder

Function: Finds objects in image according to shape information.



Input:

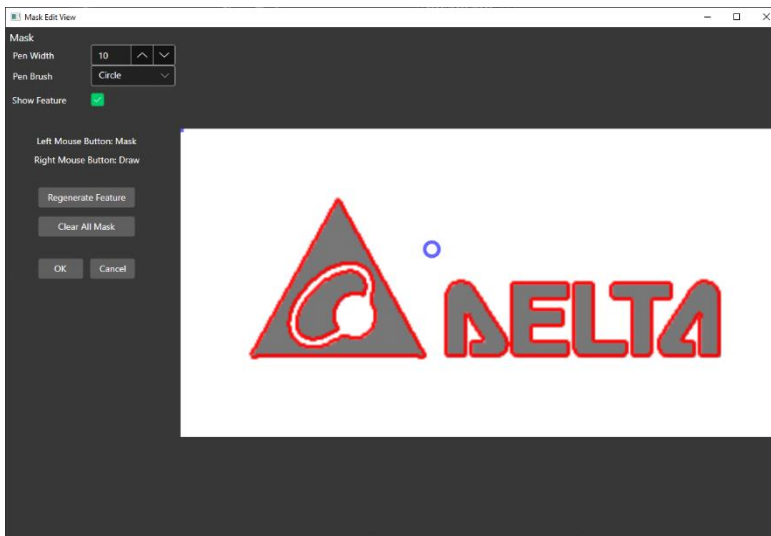
- General
  - ◆ Image: Selects the image input to be inspected, which can be a camera source or a virtual image.
  - ◆ ModelROI: Sets the scope of the model, as shown in the following figure. After confirming the model, click Train Model in the command, and a message will appear in the software when it is completed.



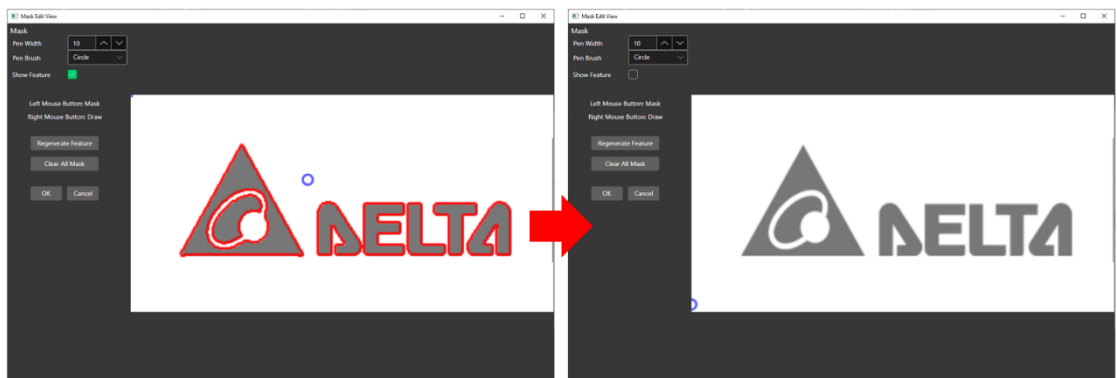
- ◆ ROI Sets the search model area, and the software will search for the set area.
- Basic
  - ◆ TargetCount: Sets the maximum number of searches. If 1 is set, the software will find up to one object within the ROI range.
  - ◆ Score: The degree to which the objects are similar. The software sets the score range

from 1-100. The higher the score, the higher the similarity is required for the object to be output.

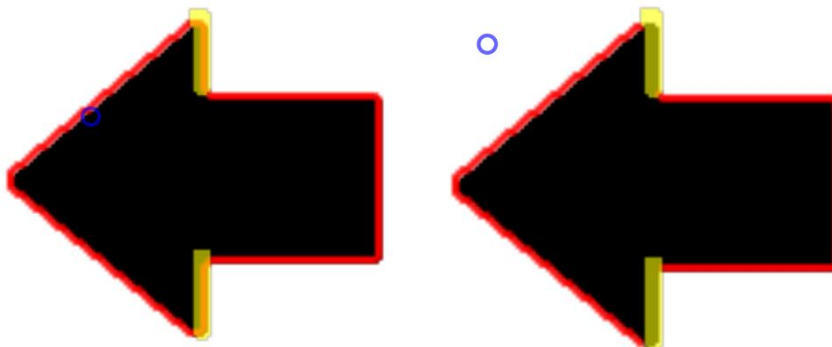
- Model
  - ◆ Mask: Clear non-characteristic areas.



- Pen Width: The width of the masking clearance pen.
- Pen Brush: The shape of the masking clearance pen.
- Show Feature: Checks the option to display the current feature, otherwise the feature will not be displayed, as shown in the following figure.

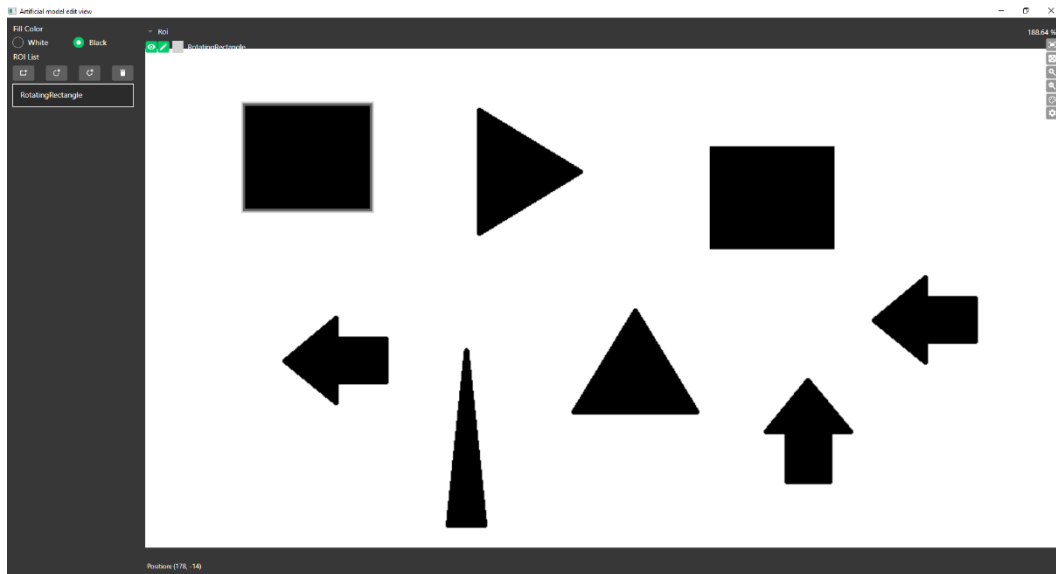


- Regenerate Feature: When users press Regenerate Feature, the red feature points will be cleared.

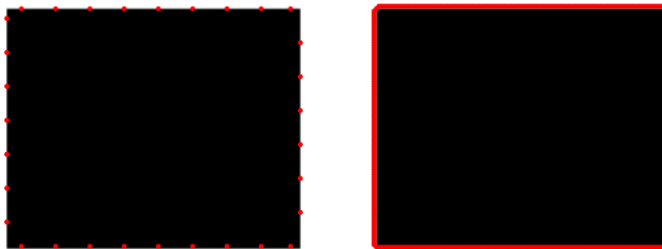


- Clear All Mask: Clears all the set mask areas.

- ◆ Artificial Model: Uses geometry to generate a black-and-white image for subsequent model training.



- ◆ AngleLower: Sets the lower limit of the rotation angle of the model.
- ◆ AngleUpper: Sets the upper limit of the rotation angle of the model.
- ◆ Accuracy: Sets the number of boundary sampling points. The image on the left shows low accuracy, and the image on the right shows high accuracy.



- ◆ ScaleTolerance: This parameter is the amount of allowable sample size variation. A zoom value of 10% can be tolerated.
  - UniScale: Allows proportional scaling based on sample size.
  - Affine: Allows one-sided scaling based on sample size.
- ◆ IsPyramidLevelCustomized: Sets the pyramid Hierarchy option.
  - Pyramid: Compares images at different scales to prevent the content you're looking for from having different sizes on the image.
- ◆ PyramidLevel: Customizes the pyramid level, you can manually adjust the pyramid level to improve the ability to find objects. The higher the pyramid level, the easier it is to find the sample, but the longer time to detect.
- ◆ EdgeThreshold: The edge threshold is the threshold value of edge search, and the value range is 0~1. When the set value is smaller, the point with lower intensity will also be regarded as a feature point, and the relative noise will be more.
- ◆ Detect Adjacency: In addition to comparing the features, the background of the model ROI will also be compared, so if there are adjacent objects in this ROI, their scores will be relatively low, and they can be identified by setting a score threshold.

- Advanced

- ◆ IgnorePolarity: Sets whether to ignore the polarity of the edges, as shown in the figure below, black marks on white and white marks on black background, set either as a sample, both can be found, ignoring the polarity of the edges.



- ◆ ObjectDistance: The distance between objects and objects. When the distance is less than the target minimum spacing, one of the objects is filtered.
- ◆ OverlapRatio: Sets the overlap ratio between objects and objects, with a value range of 0~1. When the two objects overlap more area, the lower the set value, the more the two objects can be found.
- ◆ ContrastSensitive: This parameter represents the contrast intensity distinction between the background and the sample. When the contrast between the two is low, increasing the sensitivity can improve the detection success rate, but it will take a longer time to detect.
- ◆ Sort Type: Multiple sorting methods are provided.
  - ScoreDescending: Sorts by the similarity of each shape, and the higher the similarity is the first.
  - HorizontalAscending: Sorts by the horizontal position of each shape, the smaller the X coordinate, the earlier the order.
  - HorizontalDescending: Sorts by the horizontal position of each shape, the larger the X coordinate, the higher the order.
  - VerticalAscending: Sorts by the vertical position of each shape, the smaller the Y coordinate, the earlier the order.
  - VerticalDescending: Sorts by the vertical position of each shape, the larger the Y coordinate, the higher the order.
  - TopLeftToBottomRight: First sort from small to large in Y direction and then sort from small to large in the X direction.
  - BottomRightToTopLeft: First sort from large to small in Y direction and then sort from large to small in the X direction.
  - TopRightToBottomLeft: First sort from small to large in Y direction and then sort from large to small in the X direction.
  - BottomLeftToTopRight: First sort from large to small in Y direction and then sort from small to large in the X direction.
- ◆ IsRotationCenterCustomized: The default X and Y of the inspection results are output at the center point coordinates of the template ROI. By customizing the center of rotation, the user can set the X and Y center coordinates. The X and Y of the detection



results will be output at the user-defined location. Select the display mark to see the following figure, you can directly drag the left mouse button to the position you want to set or enter the coordinate value in the field.



- ◆ RotateCx Coordinate: When the custom rotation center is enabled, users can set the X coordinate of the rotation center.
- ◆ RotateCy coordinate: When the custom rotation center is enabled, users can set the Y coordinate of the rotation center.

Command:

TrainModel: Trains the ROI features of the model.

Output:

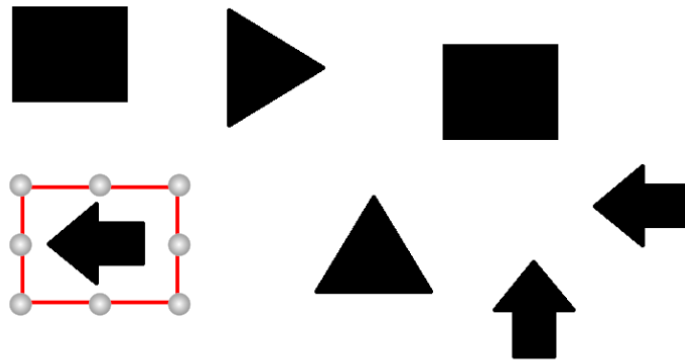
- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- AutoPyramidLevel: Displays the current pyramid level value.
- Count: Outputs the number of objects detected by the current process.
- Results
  - ◆ [n]: The Nth object information.
    - Position
      - ◇ X: The Nth object X coordinate.
      - ◇ Y: The Nth object Y coordinate.
    - Angle: The Nth object angle.
    - Score: The Nth object score.
    - Scale: The Nth object shape scale.
    - Frame
      - ◇ X: The converted X coordinate of the nth object.
      - ◇ Y: The converted Y coordinate of the nth object.
      - ◇ Angle: The converted angle of the nth object.

### 4.7.11 PatternMatch

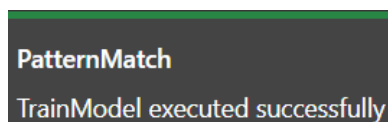
Function: Finds objects in image according to gray level information.

Input:

- General
  - ◆ Image: Selects an input image, which can be a camera source or a virtual image.
  - ◆ ModelROI: Sets the scope of the model, as shown in the following figure.



After confirming the model, click on the training model in the command, and a message will appear in the software when it is completed.



- X: The X coordinate of the center of the ROI, in pixels.
  - Y: The Y coordinate of the center of the ROI, in pixels.
  - W: ROI width dimension, in pixels.
  - H: ROI height dimension in pixels.
  - ReferenceFrame: A relative coordinate system established with reference to other components.
- ◆ ROI: Sets the search model area, and the software will search for the set area.
- Basic
    - ◆ TargetCount: The number of search targets specified for the ROI.
    - ◆ Score: Sets the model score threshold, the higher the score, the higher the similarity.
  - Model
    - ◆ AngleLower: The lower bound of the angle to be searched.
    - ◆ AngleUpper: The upper bound of the angle to be searched.
    - ◆ Pre-calculate model angle: Selects whether to perform model angle calculations. If you select true, a model will be created first, which shortens the running time.
    - ◆ PyramidLevel: Customizes the pyramid level. You can manually adjust the pyramid level to improve the ability to find objects. The higher the pyramid level, the more pyramid layers used, the faster it will be, but the easier it will be to locate the failure.

Command:

TrainModel: Trains the ROI features of the model.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- ModelPyramidLevel: Displays the current pyramid level value.
- Results
  - ◆ [n]: The Nth object information.
    - Positon
      - ◇ X: The Nth object X coordinate.
      - ◇ Y: The Nth object Y coordinate.
    - Angle: The Nth object angle.
    - Score: The Nth object score.
    - Frame
      - ◇ X: The converted X coordinate of the nth object.
      - ◇ Y: The converted Y coordinate of the nth object.
      - ◇ Angle: The converted angle of the nth object.

### 4.7.12 UndistortImage

Function: Distorted images are corrected using the camera's internal parameters and distortion coefficients.

Input:

- DistortionCoefficients: Enters the distortion coefficient parameter, which can be input into the array or software variables.
- IntrinsicMatrix: Enters the internal parameter matrix, which can be substituted into the array or software variables.
- Image: Input image format, 8-bit or 16-bit image is acceptable.

Output:



- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: The corrected image.

## 4.8 File Operation

### 4.8.1 FileReader

Function: Reads the contents of the txt file.

Input:

- Folder: Sets the path location of the txt file, and users can also select the file through the  button.
- FileName: Sets the file name in the path folder, and users can also select files through the  button.
- Operation:
  - ◆ Open: Opens the txt file.
  - ◆ Close: Closes the txt file.
  - ◆ ReadToEnd: Reads from the first line to the last line of the file.
  - ◆ ReadLines: Reads specifies number of lines from the file.
  - ◆ ReadCharacters: Reads specified number of characters from the file.
  - ◆ ReadUpToExpectedText: Reads from the beginning of the file until the expected text is met.
  - ◆ MoveSeekToLine: Moves the read location to the specified row.



Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Lines: Lines read from the file.
- Text: Text read from the file.

### 4.8.2 FileWriter

Function: Writes user-defined content to the txt file.

Input:

- Folder: Sets the path location of the txt file, and users can also select the file through the  button.
- FileName: Sets the file name in the path folder, and users can also select files through the  button.
- Operate
  - ◆ Create: Creates a txt file in the folder directory.
  - ◆ OpenOrCreate: Creates or opens a txt file in the folder directory.
  - ◆ WriteText: Writes specified text to the file.

- ◆ WriteTextCollection: Writes specified text collection to the file.
- ◆ Close: Closes the file in the folder directory.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- WriteText: The written string.

### 4.8.3 ImageSave

Function: Saves a result image of the flow, either as a 2D image or as a depth map.

Input:

- Image: The expression of the source image to be saved.
- Folder: The path to save the image. This could also be selected by clicking "Select" button.
- FileName: The file name of the saved image.
- FileExtension: The target image format, which could be BMP, PNG, JPG, GIF, or TIFF.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.

### 4.8.4 PointCloudSave

Function: Stores a point cloud of the flow.

Input:

- PointCloud: The point cloud to be saved.
- Folder: Sets the path for storing point clouds. This could also be selected by clicking "Select" button.
- FileName: Sets the file name of the saved point cloud, and it must be in string format.
- FileExtension: The target image format, which could be PCD only for now.

Output:

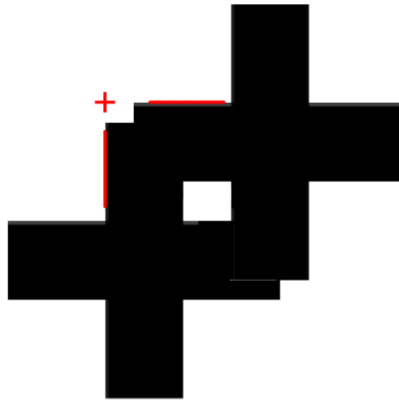
- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.

- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.

## 4.9 Image Measurement

### 4.9.1 FindIntersection

Function: Finds the coordinates of the intersection of the two lines in the 2D image.



Input:

- Type: Sets the intersection type.
- Line1: The first line used to calculate the intersection.
- Line2: The second line used to calculate the intersection.

Output:

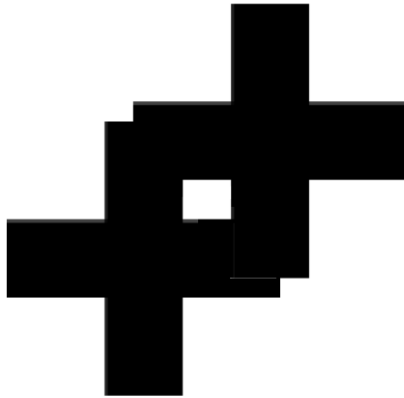
- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Found: Whether the intersection was found or not.
- Point: The coordinates of the intersection.

## 4.10 Image Filter

### 4.10.1 Average

Function: Averages the grayscale values within the region.

Before running



After running



Input:

- Input: Enters the source of the image.
- ROI: Sets the execution range of the execution filter.
- KernelSize: The kernel size used to calculate the image convolution.
- Direction: The scanning direction of the pixels.

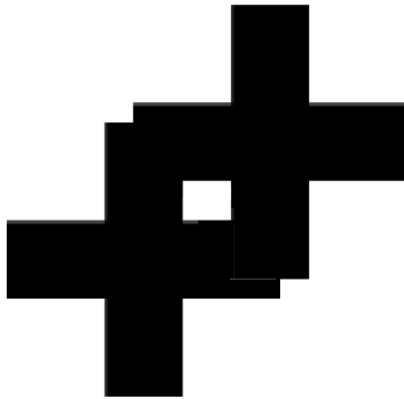
Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: Outputs the image.

## 4.10.2 Gaussian

Function: Gaussian averages the grayscale values within the region.

Before running



After running



Input:

- Input: Image source.
- ROI: Sets the execution range of the execution filter.
- KernelSize: The kernel size used to calculate the image convolution.

Output:

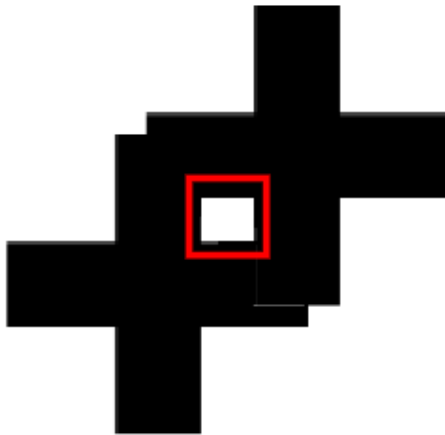
- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: Outputs the image.



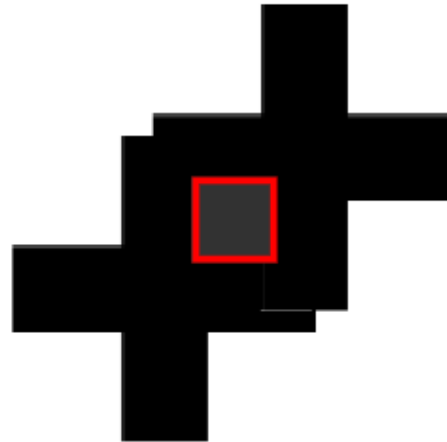
### 4.10.3 GrayLevelAssign

Function: Populates the specified grayscale value into a specific range.

Before running



After running



#### Input:

- Input: Image source.
- ROI: The execution range of the filter.
- Check Area: Sets whether to check the area.
- AreaLower: The lower threshold of area.
- AreaUpper: The upper threshold of area.
- GrayLower: The lower threshold of the grayscale value.
- GrayUpper: The upper threshold of the grayscale value.
- ObjectType: Whether the target pixel falls within or outside the threshold range.
- Fill: The grayscale value to be filled in the ROI.

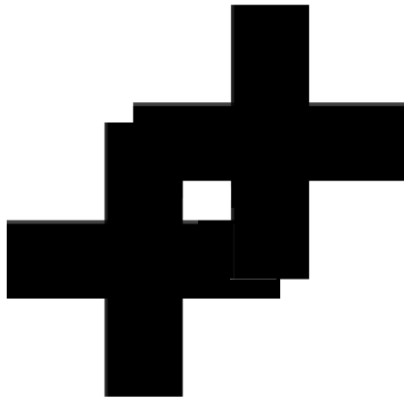
#### Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: Output image.

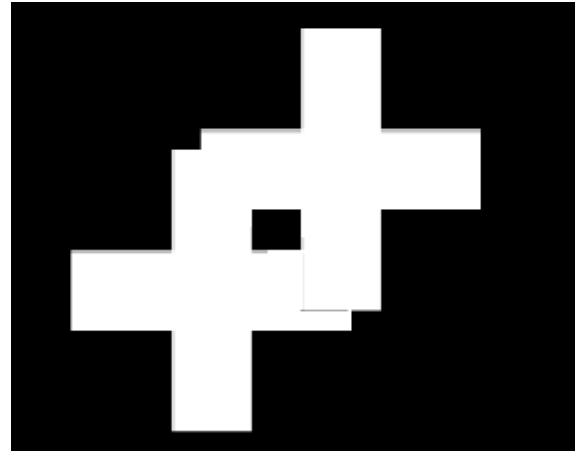
## 4.10.4 Invert

Function: Sets the input image to invert.

Before running



After running



Input:

- Input: Image source.
- ROI: The execution range of the filter.

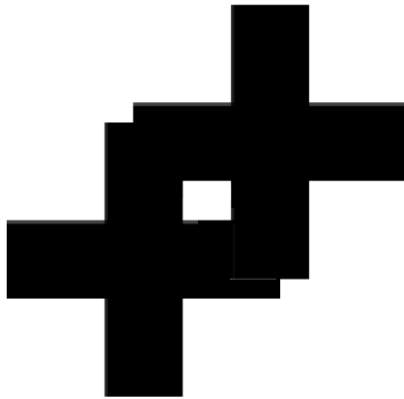
Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: Output image.

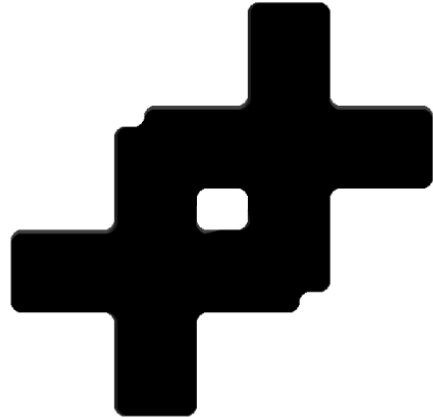
### 4.10.5 Median

Function: Replaces the pixel value with the median of all pixel gray scale values within the kernel.

Before running



After running



Input:

- Input: Image source.
- ROI: The execution range of the filter.
- KernelSize: The kernel size used to calculate the image convolution.
- Direction: The direction in which the pixel value is scanned.

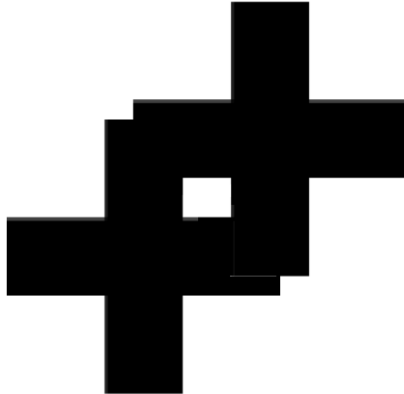
Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: Output image.

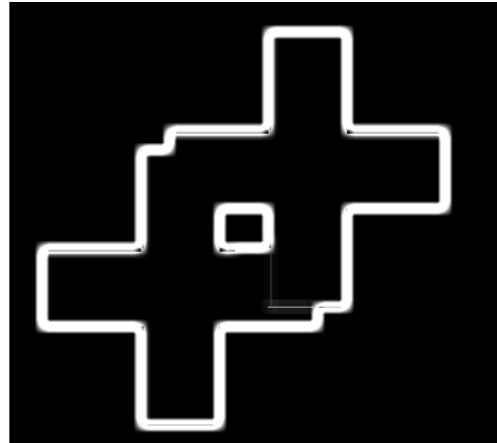
## 4.10.6 ShadingCorrection

Function: Emphasize areas of imperfection in the image.

Before running



After running



Input:

- Input: Image source.
- ROI: The execution range of the filter.
- DefectType: Sets whether the target defect is light or dark.
- KernelSize: The kernel size used to calculate the image convolution.
- Level: Amplification factor of pixel grayscale value differences.
- Noise: The threshold of grayscale values used to remove noise
- Method: Selects the filter type.

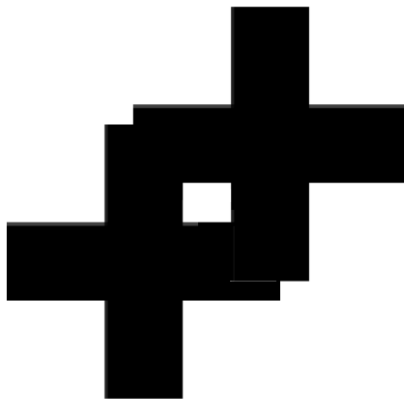
Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: Output image.

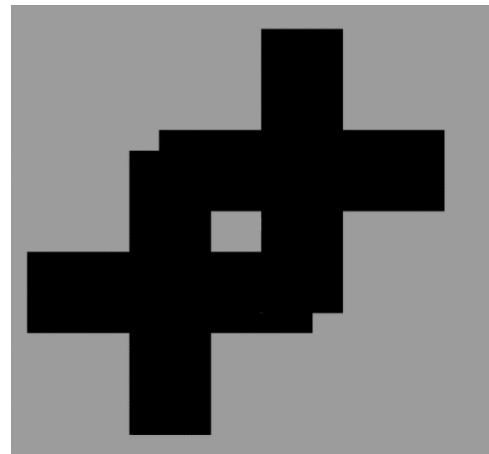
### 4.10.7 ShadingRemoval

Function: Accentuates high-contrast features and fade background areas.

Before running



After running



#### Input:

- Input: Image source.
- ROI: The execution range of the filter.
- DefectType: Sets the target defects as light or dark.
- Gain: The magnification of the difference in the grayscale values.
- Noise: The threshold of grayscale values used to remove noise
- Method: Selects the filter type.

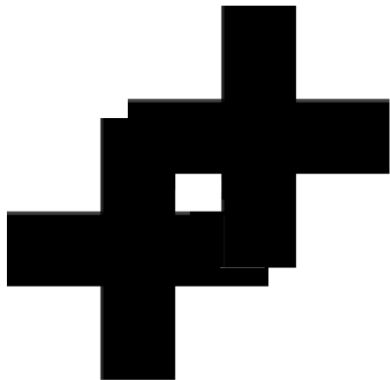
#### Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: The output image.

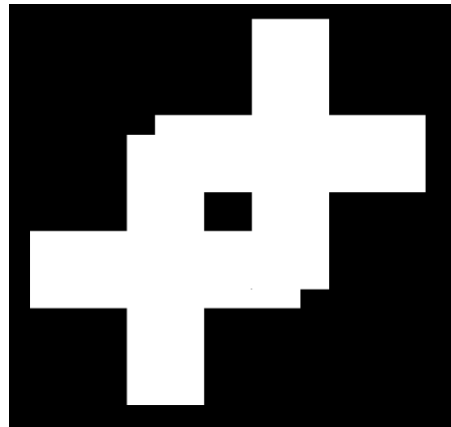
## 4.10.8 Threshold

Function: Binarizes the image.

Before running



After running



Input:

- Input: Image source.
- ROI: The execution range of the filter.
- GrayLower: The lower limit of the grayscale values.
- GrayUpper: The upper limit of the grayscale values.
- ObjectType: Sets the target to be black or white.
- Auto: Sets whether to automatically calculate the upper and lower limits of grayscale values or not.

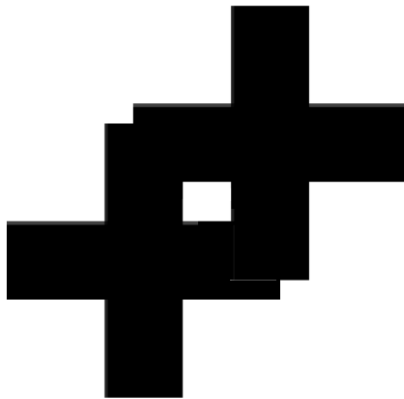
Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: The output image.

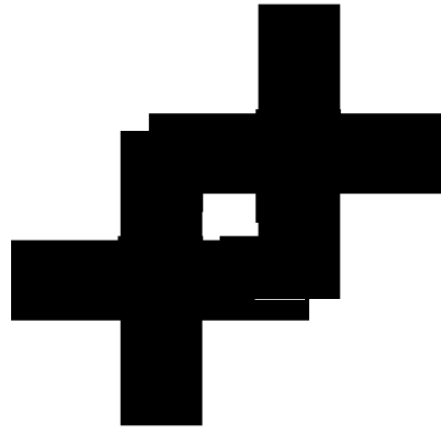
### 4.10.9 RelativeThreshold

Function: Binarizes the image using the ratio calculated according to the distribution of the pixels within the ROI range.

Before running



After running



Input:

- Input: Image source.
- ROI: The execution range of the filter.
- Ratio: Adjusts the ratio threshold.
- GrayUpper: The upper limit of the grayscale values.
- LowTail: Ignores the pixels which are the lowest n% in the distribution.
- HighTail: Ignores the pixels which are the highest n% in the distribution.
- ObjectType: Sets the foreground to black or white.

Output:

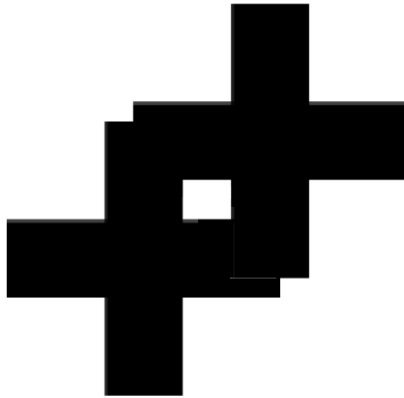
- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: The output image.

## 4.11 Image Morphology

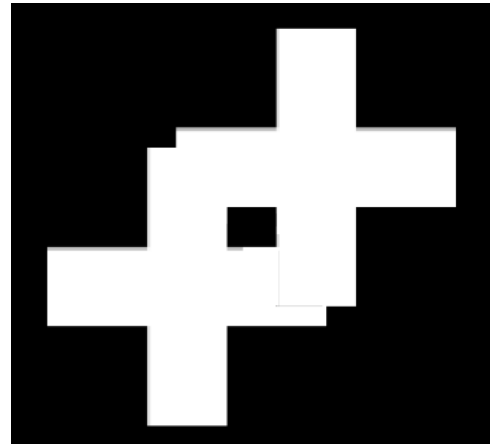
### 4.11.1 BottomHat

Function: Highlight areas that are darker than around the outline of the original image.

Before running



After running



Input:

- Input: Image source.
- ROI: The execution range of the filter.
- KernelSize: The kernel size is used to calculate the image convolution.
- Direction: The direction for scanning pixels.
- Kernel shape: The kernel shape is used to calculate the image convolution.

Output:

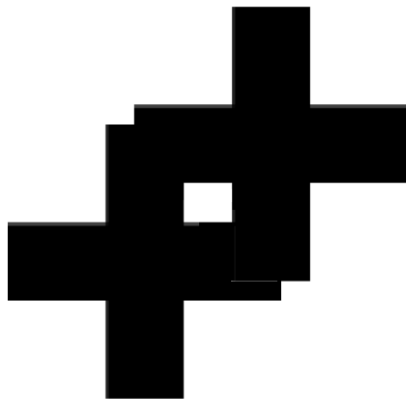
- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: The output image.



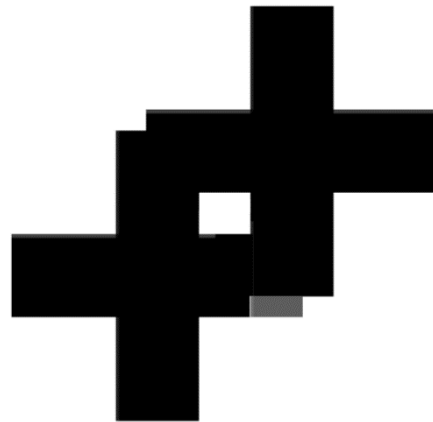
## 4.11.2 Close

Function: Reduces or completely removes dark areas of the image.

Before running



After running



Input:

- Input: Image source.
- ROI: The execution range of the filter.
- KernelSize: The kernel size is used to calculate the image convolution.
- Direction: The direction for scanning pixels.
- Kernel shape: The kernel shape is used to calculate the image convolution.

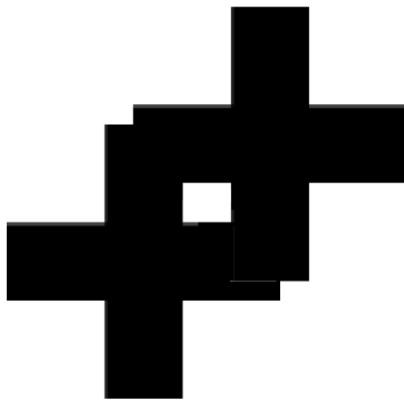
Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: The output image.

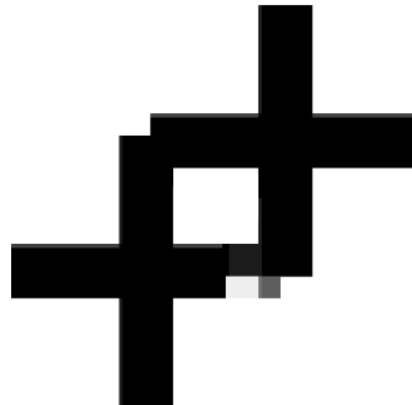
### 4.11.3 Dilation

Function: Strengthens features in bright areas and suppresses features in dark areas.

Before running



After running



Input:

- General
  - Input: Image source.
  - ROI: Sets the execution range of the execution filter.
- Basic
  - KernelSize: The kernel size used to calculate the image convolution.
  - Direction: The direction in which the pixel value is scanned.
  - Kernel shape: The kernel shape used to calculate the image convolution.

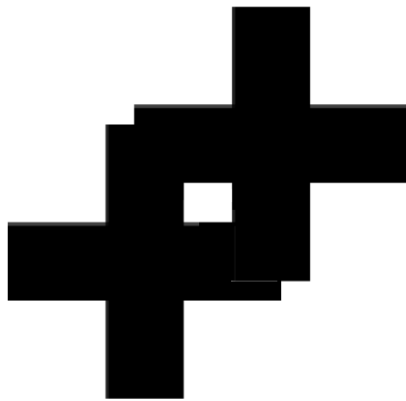
Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: The output image.

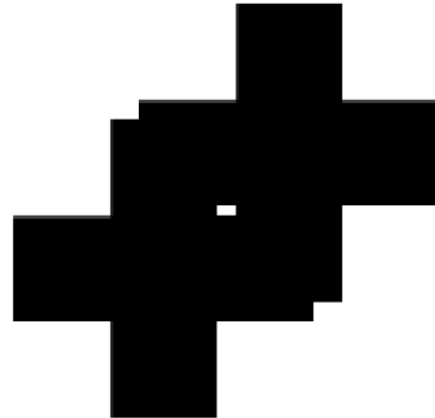
## 4.11.4 Erosion

Function: Intensifies features in dark areas and suppress features in light areas.

Before running



After running



Input:

- General
  - Input: Image source.
  - ROI: Sets the execution range of the execution filter.
- Basic
  - KernelSize: The kernel size used to calculate the image convolution.
  - Direction: The direction in which the pixel value is scanned.
  - Kernel shape: The kernel shape used to calculate the image convolution.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: The output image.

### 4.11.5 Open

Function: Some of the highlights are filtered out and the underlying bright color features are retained.

Before running



After running



Input:

- General
  - Input: Image source.
  - ROI: Sets the execution range of the execution filter.
- Essential
  - KernelSize: The kernel size used to calculate the image convolution.
  - Direction: The direction in which the pixel value is scanned.
  - Kernel shape: The kernel shape used to calculate the image convolution.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: The output image.

## 4.11.6 TopHat

Function: Highlight areas that are brighter than around the outline of the original image.

Before running



After running



Input:

- General
  - Input: Image source.
  - ROI: Sets the execution range of the execution filter.
- Basic
  - KernelSize: The kernel size used to calculate the image convolution.
  - Direction: The direction in which the pixel value is scanned.
  - Kernel shape: The kernel shape used to calculate the image convolution.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: The output image.

## 4.12 Image Color Filter

### 4.12.1 ColorExtract

Function: Proceed color image to grayscale for the user-specified color channels.

Before running



After running



Input:

- Input: Image source.
- Mode: Sets the image color conversion channel. Grayscale channel, red channel, blue channel, and green channel can be set.

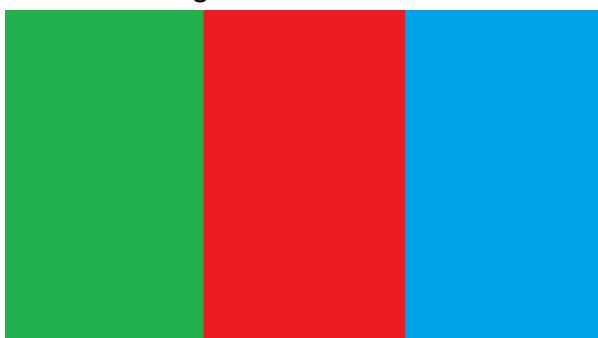
Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: The output image.

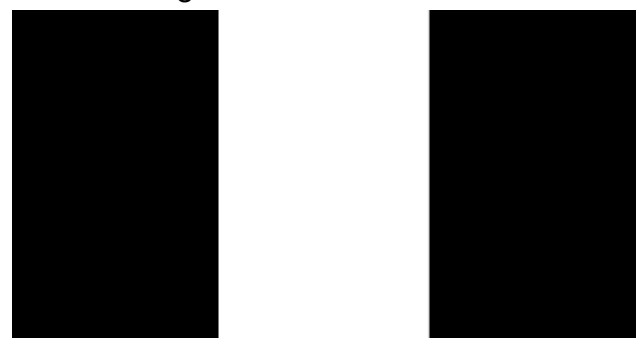
### 4.12.2 ColorThreshold

Function: Binarizes color images according to a specified color range.

Before running



After running



Input:

- General
  - Input: Sets the source image.

- Basic
  - Mode: Sets the processing color space, including RGB and HSV.
  - ObjectType: Sets the pixels in the color range to be black or white after binarization.
- HSV
  - InverseHueRange: Sets whether to reverse the hue range. true is the range outside the upper and lower limits, and false is the range within the upper and lower limits.
  - HueLower: The lower hue range value.
  - HueUpper: The maximum Hue Range value.
  - InverseSatRange: Sets whether to reverse the saturation range. true is the range outside the upper and lower limits, and false is the range within the upper and lower limits.
  - SatLower: The lower limit of the saturation range.
  - SatUpper: The upper limit of the saturation range.
  - InverseValueRange: Sets whether to reverse the luminance range. true is the range outside the upper and lower limits, and false is the range within the upper and lower limits.
  - ValueLower: The lower limit of the luminosity range.
  - ValueUpper: The upper limit of the luminosity range.
- RGB
  - InverseRedRange: Whether to reverse the red threshold range.
  - RedLower: Sets the lower red color threshold.
  - RedUpper: Sets the upper limit of the red color threshold.
  - InverseGreenRange: Whether to reverse the green threshold range.
  - GreenLower: Sets the lower green color threshold.
  - GreenUpper: Sets the upper limit of the green color threshold.
  - InverseBlueRange: Whether to reverse the blue threshold range.
  - BlueLower: Sets the lower blue color threshold.
  - BlueUpper: Sets the upper limit of the blue color threshold.

**Output:**

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: The output image.

### 4.12.3 ColorDistance

Function: Uses the proximity of the pixel color to the reference color value to convert the image to grayscale.

Before running



After running



Input:

- General
  - Input: Sets the source image.
- Basic
  - Mode: Sets the processing color space, including RGB and HSV.
- HSV
  - Hue: The reference hue value.
  - Sat: The reference saturation value.
  - Value: Refers to the luminosity value.
- RGB
  - Red: The red value of the reference.
  - Green: The green value of the reference.
  - Blue: The blue value of the reference.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: The output image



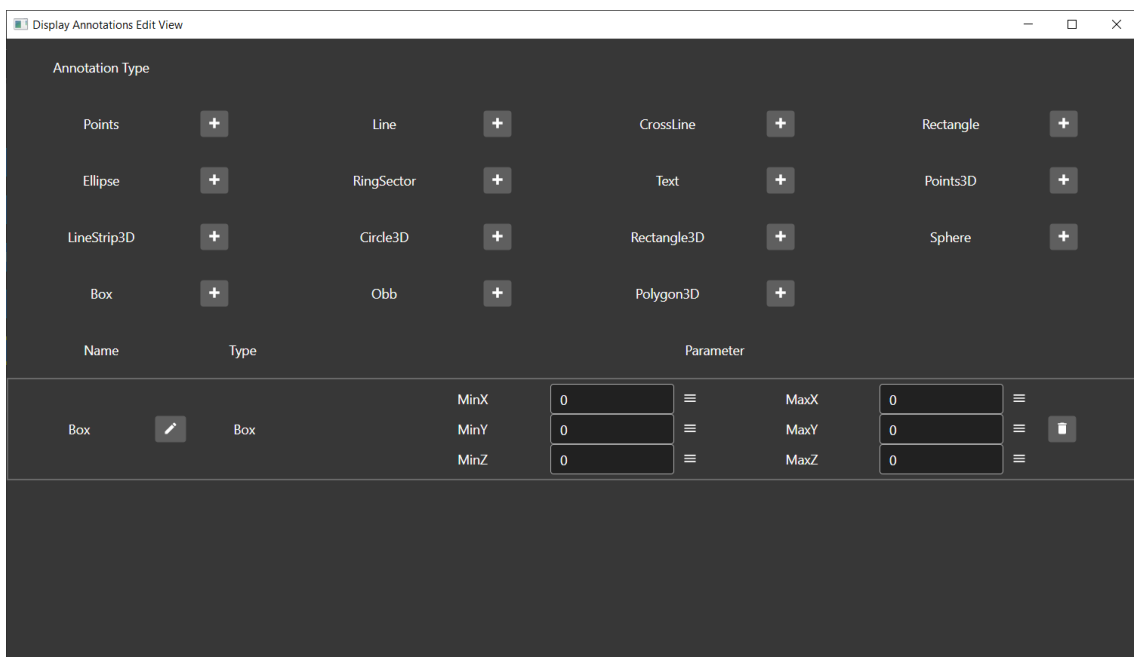
## 4.13 Process Control

### 4.13.1 Display

Function: Creates display tools such as points, line segments or geometric shapes to attach to the image.

Input:

- Feature: Adds the type of feature you want to display in the field below and enter the parameters of the feature.
  - 2D features: Points, Line, CrossLine, Rectangle, Ellipse, RingSector, Text.
  - 3D features: Points3D, LineStrip3D, Circle3D, Rectangle3D, Sphere, Box, Obb, Polygon3D.



Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.

### 4.13.2 Break

Function: The "break" component is used to prematurely terminate the execution of a loop. During the execution of the loop, if certain conditions are met, the "break" component immediately terminates the loop and exits it. This helps control the execution of the loop, allowing it to stop under certain conditions.

Input:

- Condition: Sets terminate conditions, terminate when the condition is true, otherwise not.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Result: Displays the result of the conditional judgment as True or False.

### 4.13.3 Continue

Function: The "continue" component is used to skip the current iteration and proceed with the next iteration. This helps control the execution of the loop, allowing specific iterations to be skipped under certain conditions.

Input:

- Condition: Sets skip condition, skip the current iteration when condition is true.

Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Result: Displays the result of the conditional judgment as True or False.

### 4.13.4 Delay

Function: Delay the process for a period of time. When the process reaches this component, it will pause for the specified duration before continuing execution.

Input:

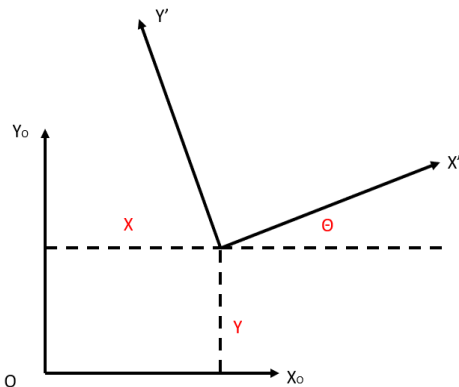
- Delay Time: The time interval set in milliseconds.

Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.

### 4.13.5 ReferenceFrame

Function: Creates a custom coordinate system in a 2D image.



Input:

- Angle: Sets the angle of the coordinate system. Positive angles are counterclockwise, while negative angles are clockwise.
- X: X offset value.
- Y: Y offset value.

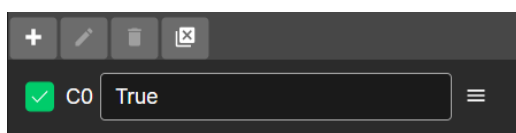
Output:





- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Frame
  - ◆ X: X value of the origin of the coordinate system.
  - ◆ Y: Y value of the origin of the coordinate system.
  - ◆ Angle: Angle of the coordinate system.


### 4.13.6 Status

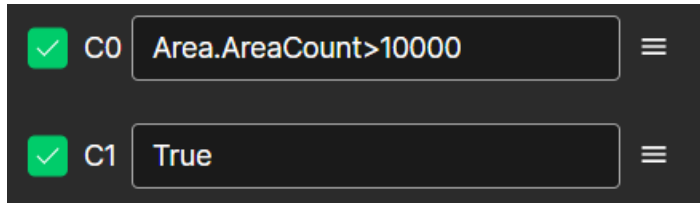
Function: Status is used to judge whether the current status of flow matches the specified conditions.

Input:



- : Adds a judgment condition.
- : Renames the judgment condition.
- : Deletes the selected judgment condition.
- : Deletes all judgment conditions.

- : Enables or disables this judgment condition.
- Conditional statement: Users inputs a conditional statement into the input box. The software outputs True or False based on the evaluation result, as shown in the image below.



Output:

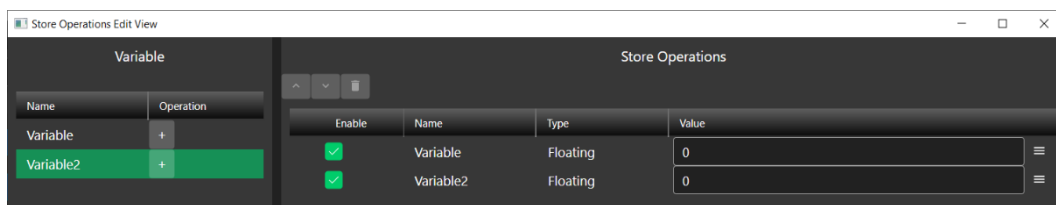
- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Count: The total number of executions.
- NG Count: The total number of NG counts.
- OK Count: The total number of OK counts.
- Result: The current execution result, True is OK and False is NG.
- Status conditions
  - ◆ Count: The number of judgment conditions.
  - ◆ Name:
    - The Nth judgement condition.
    - Result: Evaluation result of the judgement condition.
    - Name: The name of the judgment condition.

### 4.13.7 Store


Function: Performs numerical calculations or other processing on variables.

Input:

- Edit:





Variable:

Sees Chapter III for details. If users add variable operation, click on the operation page , and the operation will be added to the editing area.

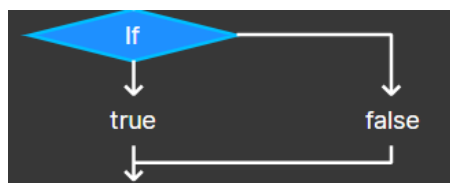
Store Operations:

- ◆ Move up : Moves the order of variable operation forward.

- ◆ Move down : Moves the order of variable operation backward.
  - ◆ Delete : Removes the variable operation.
  - ◆ Enable: Enables the variable operation when checked.
  - ◆ Name: The name of the variable.
  - ◆ Type: The type of variable.
  - ◆ Value: The expression of variable operation.
- Name: The name of the function block.
  - Type Name: The name of the component type.
  - Status: The current status. “true” represents normal execution, “false” represents abnormal execution.
  - ErrorMessage: Shows the reasons when the execution is abnormal.
  - LastRunTime: Current run time of the function block.

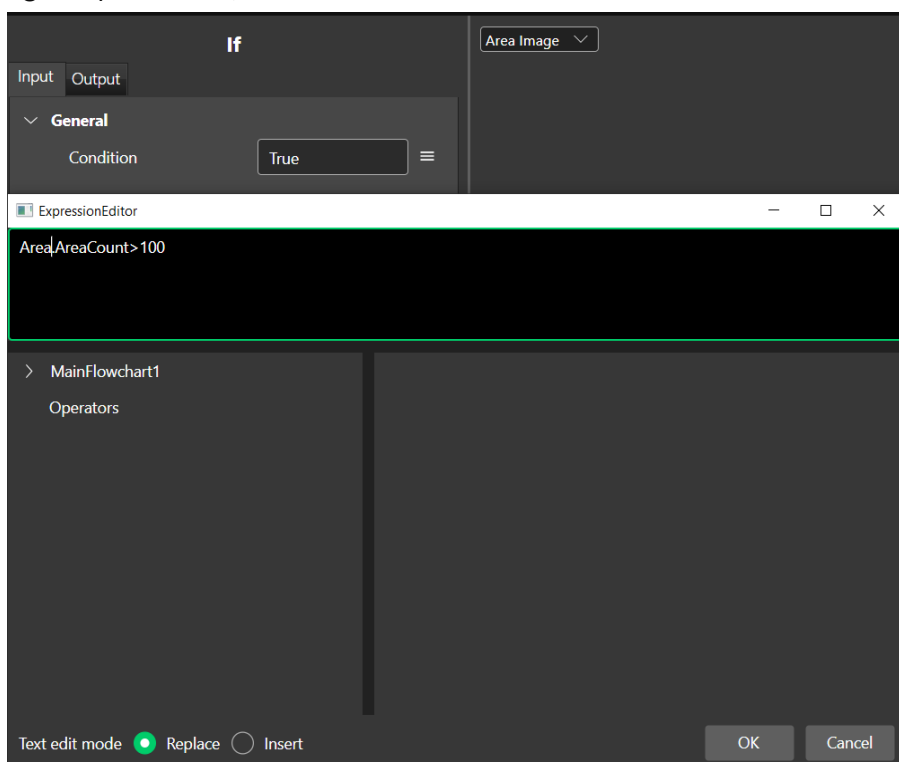
### 4.13.8 Condition

Function: Determines the execution of conditional processes (where the process evaluation condition is True) based on a specified expression.



Input:

- Condition: Determines process conditions and execute corresponding processes based on the conditions. Users can force the condition to be true or false. It can also be achieved through expressions, such as `Area.AreaCount>100`.

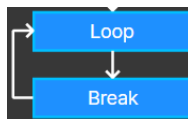


Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Result: The execution result.

### 4.13.9 Loop

Function: The first initial variable statement of the loop is executed once, and thereafter, each time the loop restarts, it evaluates the stop loop condition to determine whether to execute the next loop. After each loop iteration, the increment statement is executed once.



Input:

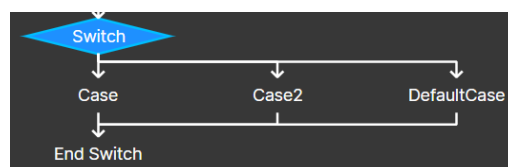
- Start Number: The initial value of the cycle execution.

Output:


- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Count: The total number of loop executions.

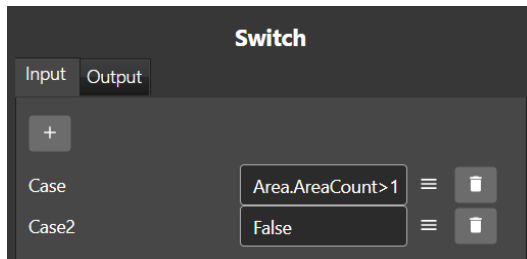
### 4.13.10 Switch


Function: When the given conditional expression of the Case is true, the flow block inside the Case will be executed.



Input:

- Add Case  : Clicks this button to add a new case.
- Case[n]: The conditional expression of the case. The corresponding flow block will be executed if the expression evaluates to true.



- Delete Process  : Deletes the selected case.

Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.

## 4.14 3D Coordinate Conversion

### 4.14.1 NPointsWithZTransform

Function: Coordinate conversion is carried out with the 2D transformation matrix and Z-axis distance which was calculated from NPointsWithZ, and the image point coordinates are converted into arm coordinates.

Input:

- PointCloud: The source of the 3D point cloud.
- ImagePoints: 2D coordinate points. Image pixel coordinate collection, ex: new Collection<Point2D>() {new Point2D(100, 100)}
- Homography: The 2D transformation matrix between robot and 3D camera
- ZDistance: The z-axis distance between robot and 3D camera

Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Points: The robot coordinates of the converted points are output.
- Conversion Result: Outputs the conversion result. True indicates that the conversion is successful, and False indicates that the conversion fails.

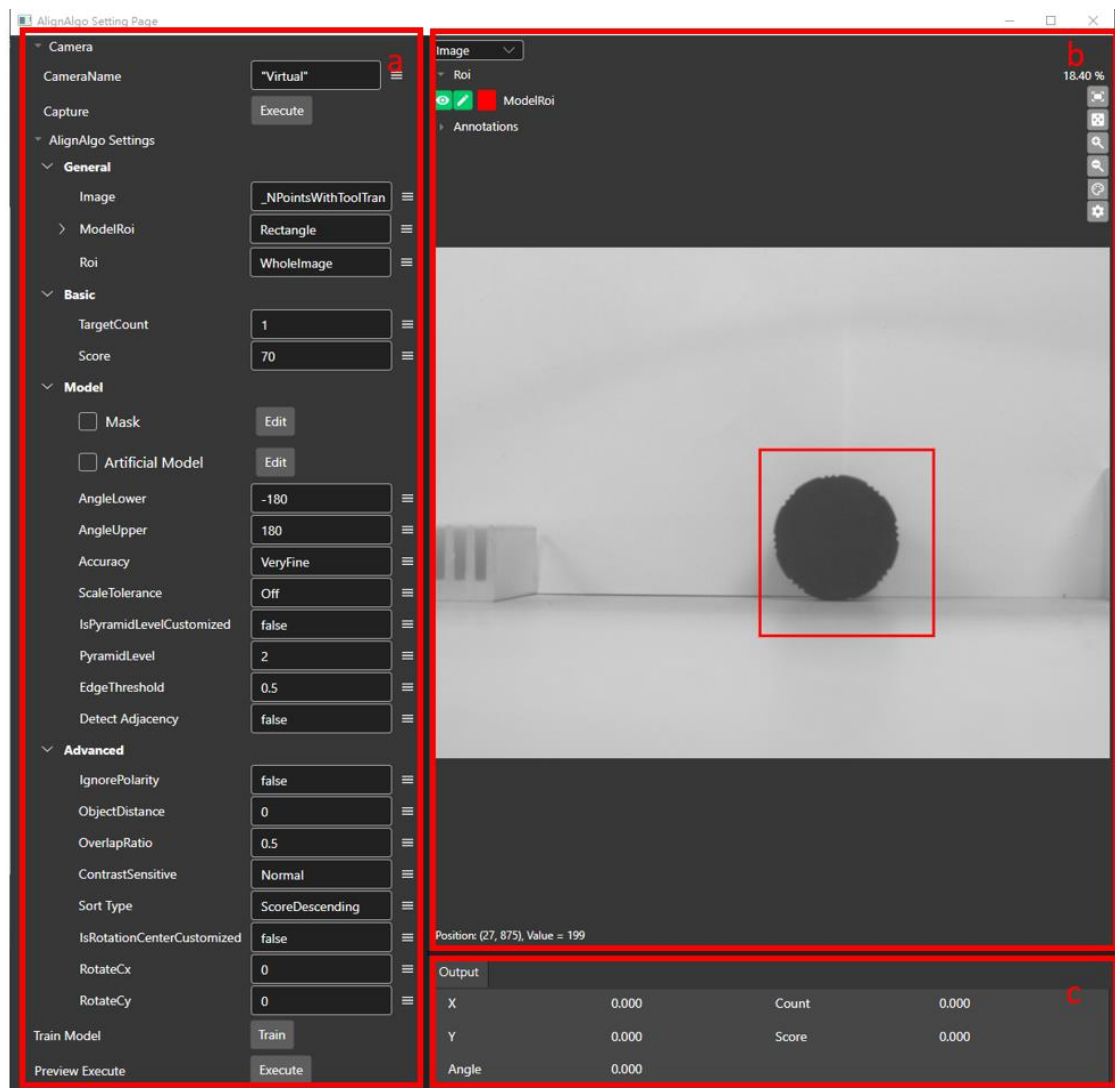
## 4.15 2D Coordinate Conversion

### 4.15.1 Placement

Function: According to the results corrected by the NPointWithTool or by providing a 3x3 homography matrix, the image coordinates can be converted to arm coordinates.

Input:

- Imager: Performs a conversion of image sources.
- homography x: Sets the source of the transformation matrix, such as homography or a 3x3 matrix from NPointWithTool.
- RotationRelation: The rotation relationship between the image and the arm coordinates, which can be corrected by NPointWithTool.
- AlignAlgo: The function of the algorithm used in the operation process.
- Calibration algorithm setting



#### a. Parameter Settings

- Camera
  - ◆ CameraName: Sets the name of the camera.
  - ◆ Capture: Taps the button to capture the image.



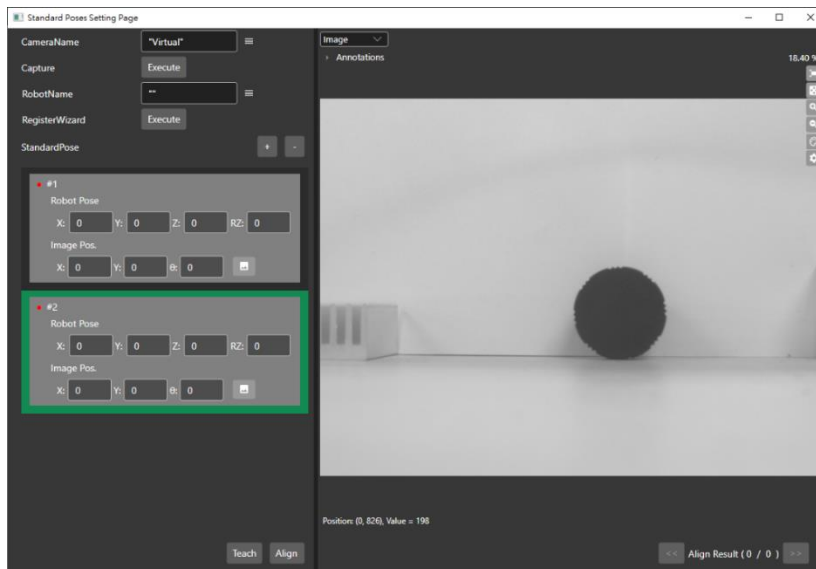
- **AlignAlgo Setting**

Refers to the AlignAlgo function. The setting of this align algorithm is used for subsequent registration of standard bits and positioning in operating mode.

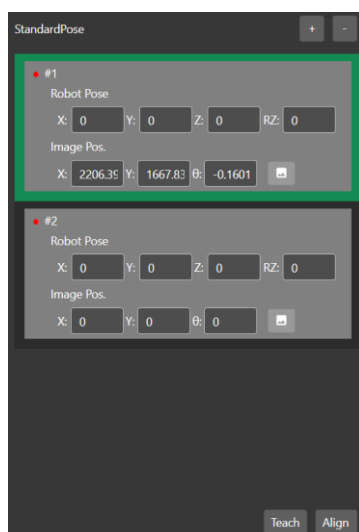
b. Current imagery: Displays the current image and post-execution features.

c. Output: Outputs the results of the calibration algorithm, such as coordinates, quantities, and fractions.

- **Standard pose settings**



- **CameraName:** Sets the name of the camera.
- **Capture:** Taps the button to capture the image.
- **RobotName:** Sets the name of the connecting arm.
- **RegisterWizard:** Clicks the RegisterWizard button, and the software will guide the user step by step to complete the registration of the standard poses.
- **StandardPose:** Users can set the number of standard bits according to their needs. Clicks to add a standard pose **+**, and click to delete the item you want to delete to delete a standard pose **-**. When the standard poses are all set, the component setting is completed



- ◆ Robot Pose: Enters the coordinates when the arm grabs the object. Tap Teach to automatically read the current position of the arm.
- ◆ Image Pos.: Obtains the image coordinates of the object after the align algorithm. Users can click Align to automatically obtain the image coordinates.
- Output Rule: Output conversion result relationship. When the target is located in the one-to-one operation mode, the algorithm directly assigns which standard bit (nozzle) to use for grabbing. If two targets are located, and the user registers 3 standard bits in the registration stage,  $2 \times 3 = 6$  sets of results will be output. On behalf of a certain positioning target, when grabbing with nozzle 1, nozzle 2 and nozzle 3, the grasping attitude is respectively. As for which nozzle the specific user wants to use to grab, it is up to the user to decide through the host computer.

Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Point: The converted attitude.

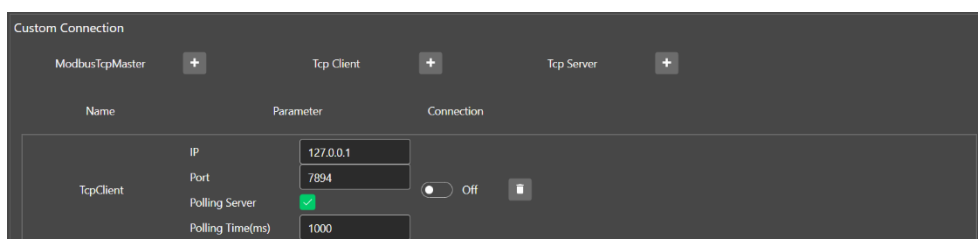
## 4.16 Communications

### 4.16.1 ConnectionRead

Function: If the software is set to the master, the slave message can be read through this function block.

Input:

- Connection Name: Sets the name of the device connected to the slave. The connection method is shown by the communication page in the platform page, as shown in the figure below.



- Wait Timeout: A TCP timeout occurs when a TCP connection is transmitting data if the software does not receive the data from the sender within a predetermined amount of time. The unit is set to millisecond. Enter a value of -1 to wait until a response and 0 to not wait.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.

- Status: The current status. “true” represents normal execution, “false” represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Received Bytes: Received data, expressed in bytes.
- Receive String: Receives data, represented as a string.

## 4.16.2 ConnectionWrite

Function: This function block allows users to write messages to other devices.

Input:

- Connection Name: Sets the name of the content in the custom connection, as shown in the following figure.



- Data Format: Sets the format of the data to be transferred.
- Data String: The data to send, represented as a string.
- Data Bytes: The data to send, represented as a byte.
- Is Wait: Sets whether to wait. True is waiting, and False is not waiting.
- Wait Timeout: A TCP timeout occurs if the software does not receive the data from the sender within a predetermined period of time. The unit is set to millisecond. Enter a value of -1 to wait until a response and 0 to not wait.

Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. “true” represents normal execution, “false” represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Received Bytes: Received data, expressed in bytes.
- Receive String: Receives data, represented as a string.

## 4.16.3 ModbusRead

Function: Uses Modbus read commands to read data from other devices.

Input:

- Connection Name: Sets the name of the content in the custom connection.
- Address: Sets the reading location. The address value is hexadecimal.
- Quantity: Sets the number of read addresses.

- Timeout: Sets the timeout period. The unit is millisecond.
- Decoding type: Decodes with single word (16 bit) or double word (32 bit).

Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Received Data: Saves the integer set of received data.

#### 4.16.4 ModbusWrite

Function: Uses Modbus Write commands to write data to other devices.

Input:

- Connection Name: Sets the name of the content in the custom connection.
- Write function: Chooses a single register or multiple registers to write.
- Address: Sets the written location. The address value is hexadecimal.
- Data: Sets the data set to be written to the register.
- Timeout: Sets the timeout period. The unit is ms. Enters a value of -1 to wait until a response and 0 to not wait.

Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.

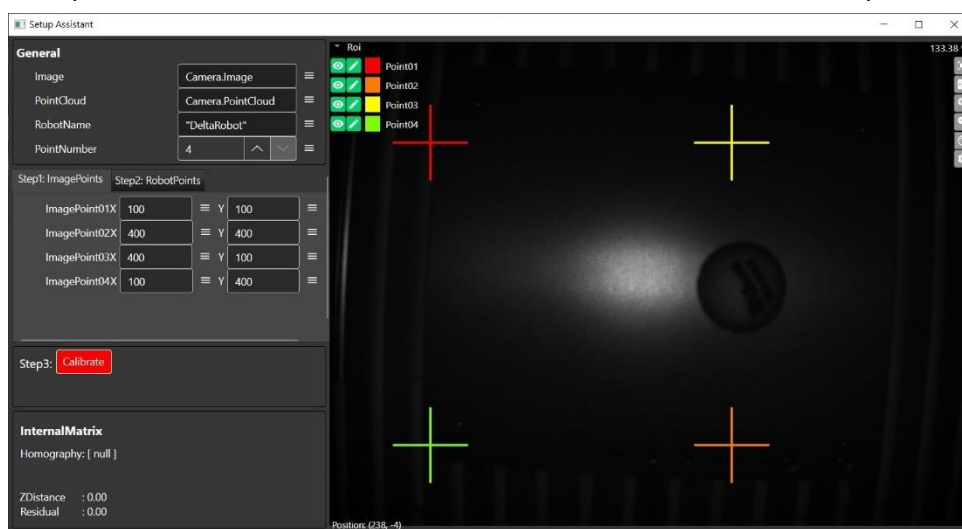
## 4.17 3D Calibration

### 4.17.1 NPointsWithZ

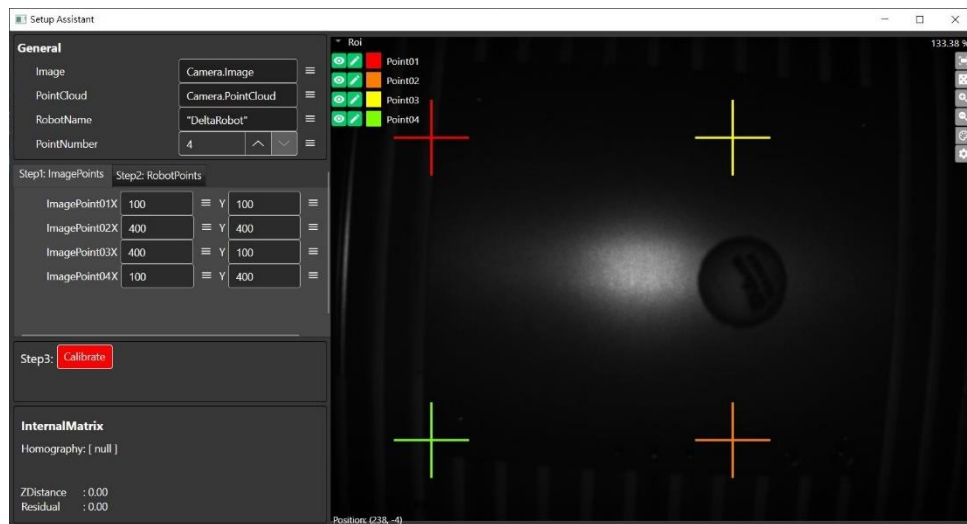
Function: Through N points, calculates the transformation matrix between the 3D camera and the robotic arm, as well as the Z-axis distance between them.

Input:

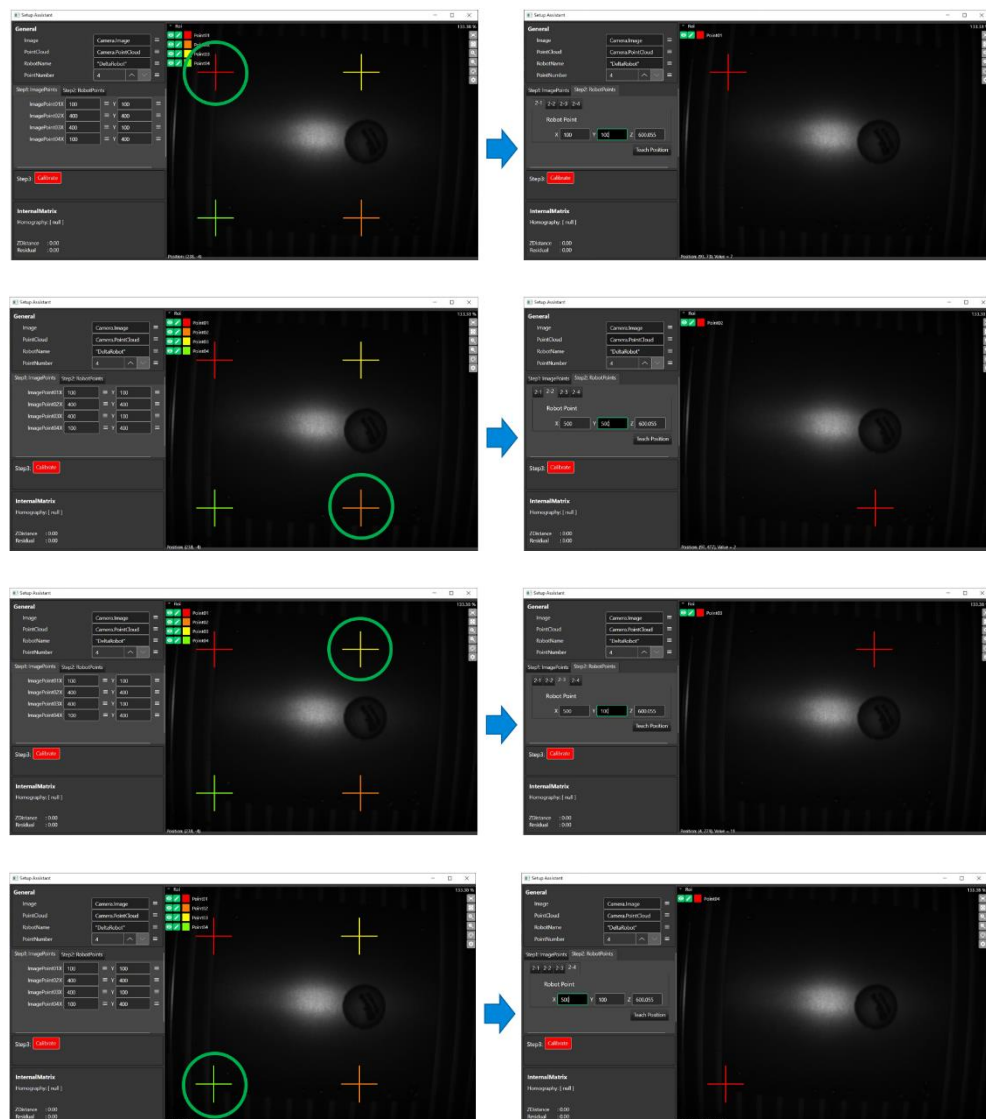
- General
  - Imagery: A source of 2D imagery.
  - Point Cloud: The source of the 3D point cloud.
  - Arm Name: Sets the name of the connected arm.
  - Setup Assistant: Calibration assistant. Presses the button to open setting window.



- ◆ General
  - Image: The source of the 2D image.
  - PointCloud: The source of the 3D point cloud.
  - RobotName: Sets the name of the connected arm.
  - PointNumber: Sets the number of calibration points. Users can enter the quantity directly or click "<" and ">" to modify the quantity.
  - Image Points: Sets the image coordinates of each feature point sequentially, users can directly enter the value, link other parameters, or drag the cross on the right screen.



- **RobotPoints:** Sets the arm coordinates of each feature point in order. When clicking on different points, the right screen will only display the corresponding cross symbol. Please move the arm to the corresponding feature point then record the pose of the arm. Users can directly enter the value or click "Teach Position" button fill the data into the field.



- Calibrate: Establishes a conversion relationship between the image coordinates and the robot coordinates. If the point information is normal, the calibration will be displayed as successful. The calibration failure may be due to the invalid point cloud corresponding to the image point.
- InternalMaxtrix: Displays the results of the calibration, including the conversion matrix, Z-axis distance, and calibration residuals.

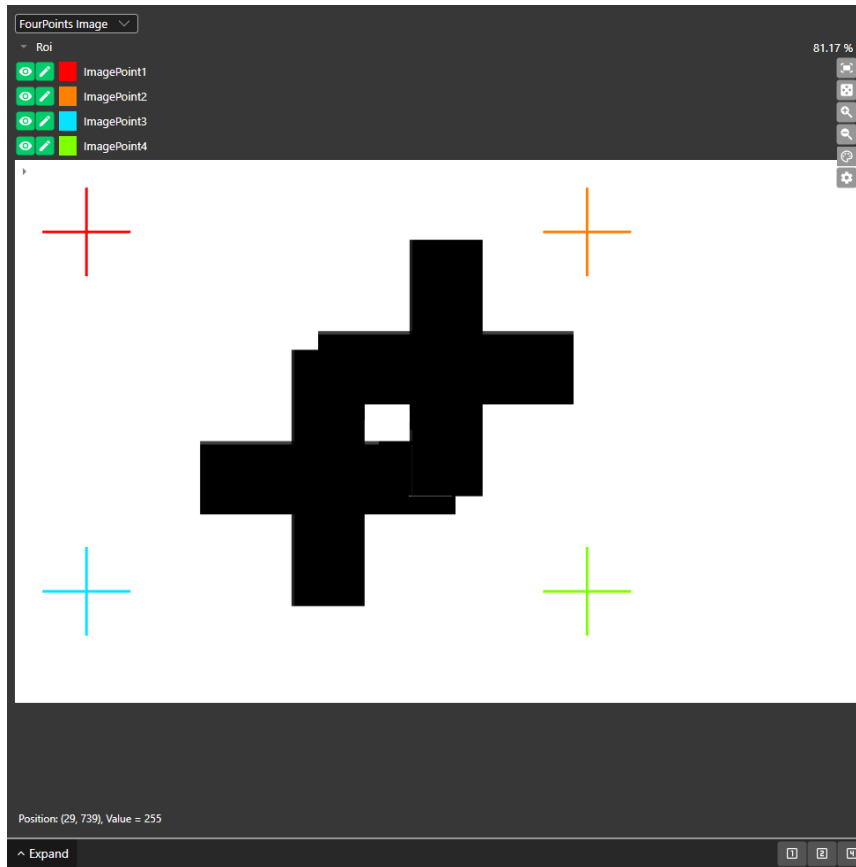
**Output:**

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Homography: 3x3 transformation matrix from 2D image to 2D robot.
- IsDataModified: Whether the data has been modified after calibration.
- Pixel Unit: Output pixel size per unit, in mm.
- Residual: Error after transformation.
- Z-axis distance: The Z-axis distance between robot and 3D camera.

## 4.18 2D Calibration

### 4.18.1 FourPoints

Function: Calculates the homography matrix between the camera and the robot through four correspondence points.



Input:

- Image: The image to do calibration.
- ImagePoint1: First image point for calibration.
- ImagePoint 2: Second image point for calibration.
- ImagePoint 3: Third image point for calibration.
- ImagePoint 4: Fourth image point for calibration.
- RobotP1X: X of robot point 1.
- RobotP1Y: Y of robot point 1.
- RobotP2X: X of robot point 2.
- RobotP2Y: Y of robot point 2.
- RobotP3X: X of robot point 3.
- RobotP3Y: Y of robot point 3.
- RobotP4X: X of robot point 4.
- RobotP4Y: Y of robot point 4.

Output:



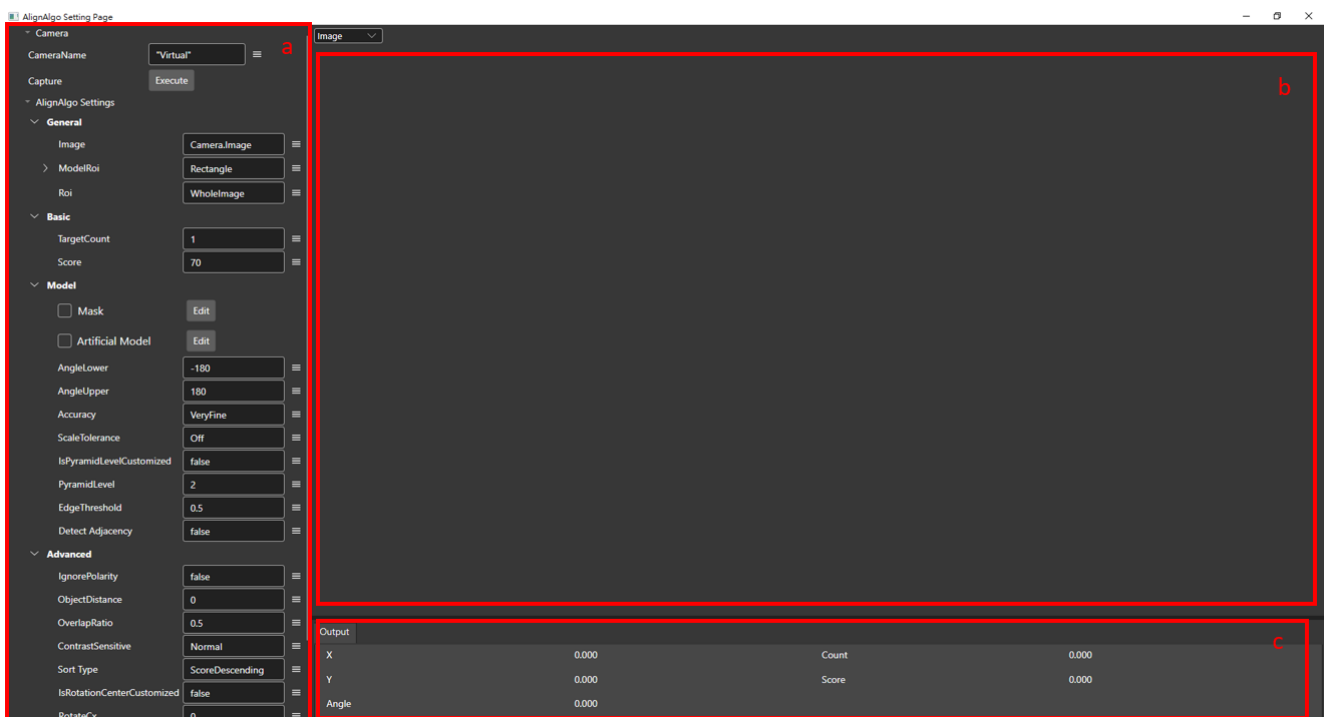
- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Homography: Outputs a matrix that converts 2D images to 2D robot coordinates.
- PixelUnit: The ratio of mm to pixel.

## 4.18.2 NPointsWithTool

Function: To train the transformation relationship between the image coordinates and the robot coordinates, incorporating the tool frame.

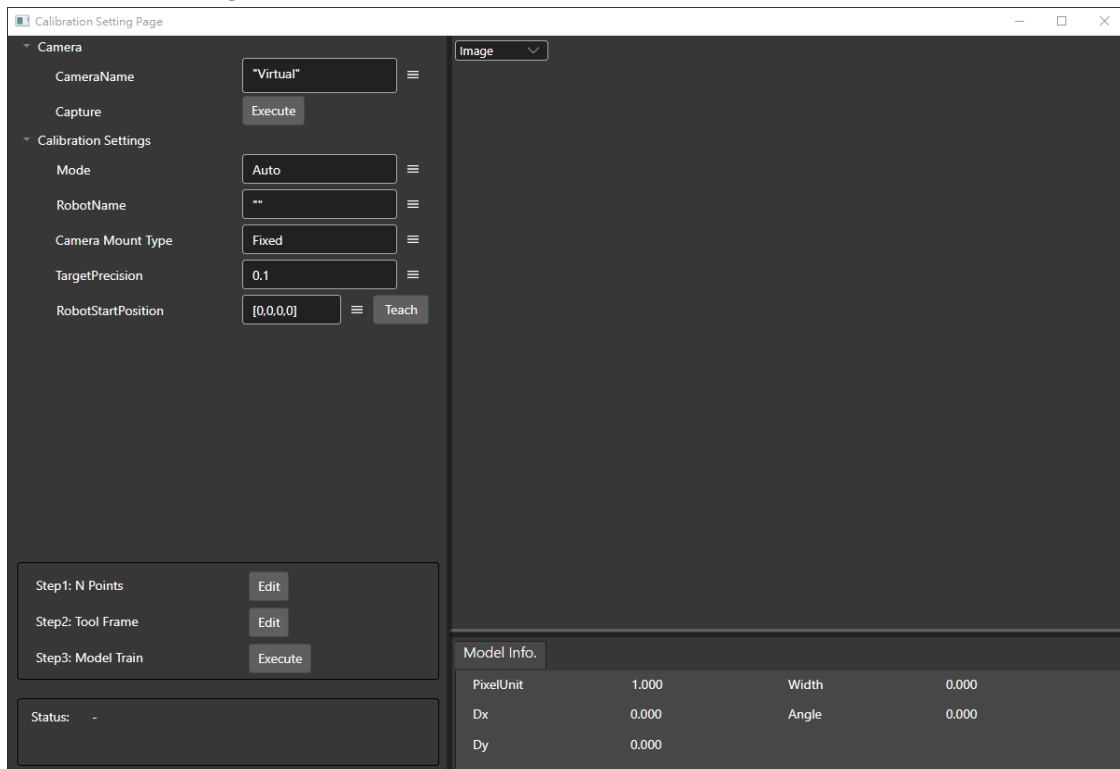
Input:

- AlignAlgo: Algorithm type used for alignment.
- AlignAlgoSetting: Sets the parameters of the alignment algorithm.

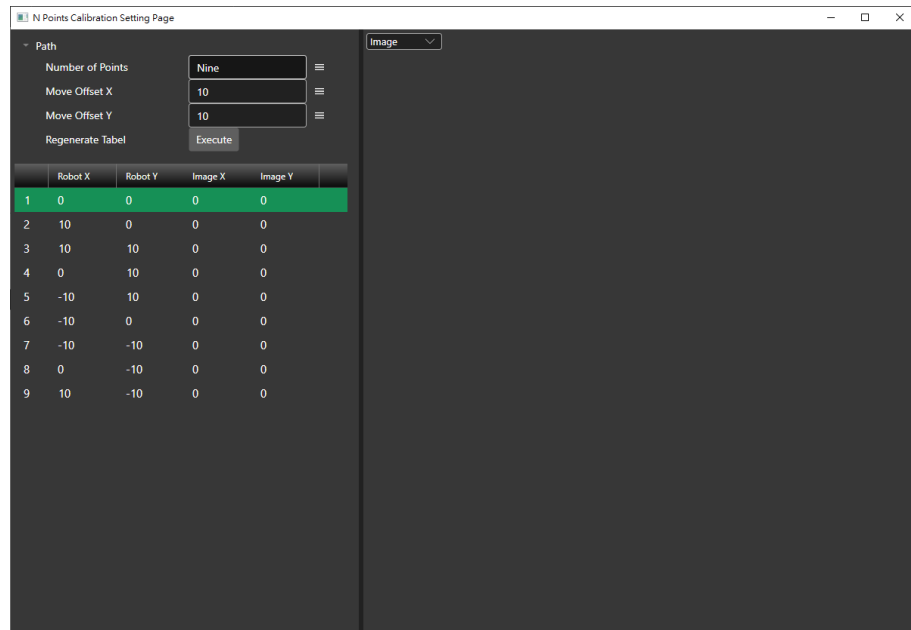


- a. Align algo settings: Settings of align algo element. Please refer to chapter 4 for different types of align algo.
- b. Image Display Area: Displays image and align results.
- c. Calibration Result Coordinates: Displays the align results.

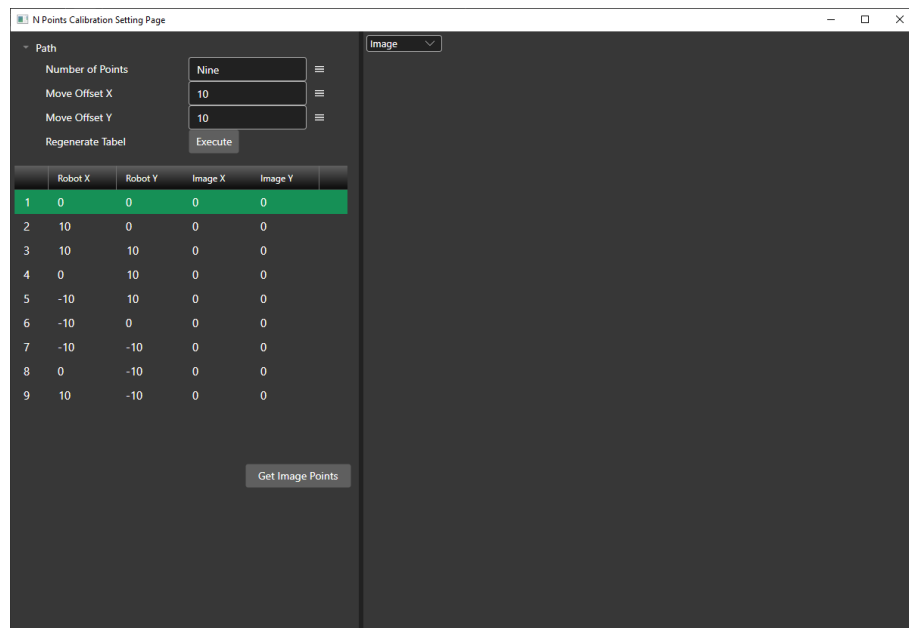
## ■ CalibrationSetting:



- Camera
  - ◆ CameraName: Sets the name of the camera.
  - ◆ Capture: Clicks “Execute” button to capture image.
- Calibration settings
  - ◆ Mode:
    - Automatic: The software automatically controls the robot and runs automatic calibration.
    - Manual: Users need to actively fill values to perform calibration
  - ◆ RobotName: Sets the name of the connected arm.
  - ◆ Camera Mount Type: Sets the of camera mount type. Such as if the camera is on a fixed position, or the camera is mounted on the robot joint.
  - ◆ TargetPrecision: Sets the target accuracy in mm.
  - ◆ RobotStartPosition: Sets the robot start position. If users click on Teach, the software will automatically get the current robot position.
- Step 1: N Points Calibration
  - ◆ Auto mode: When users set the Number of Points, Move Offset X / Move Offset Y, the “Regenerate Table” button is necessary to click.

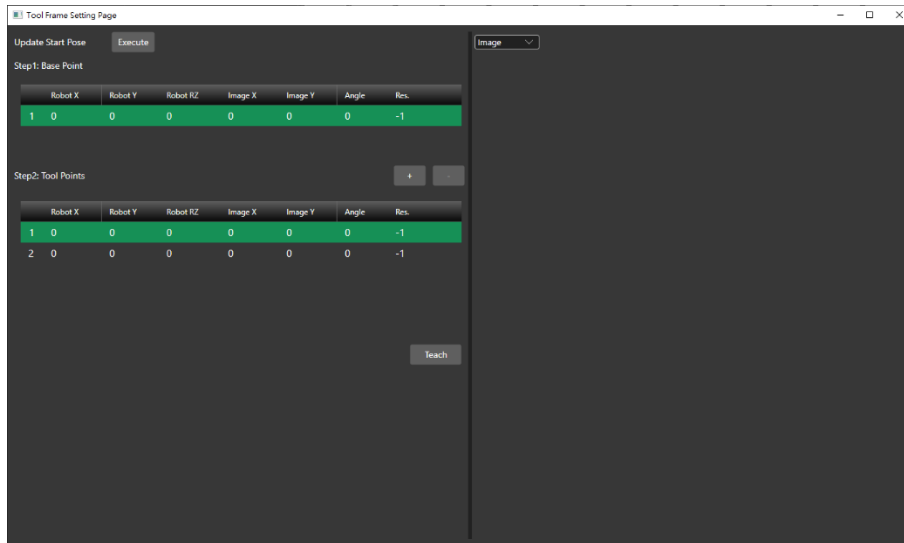


- Manual mode: When users set the Number of Points, Move Offset X / Move Offset Y, the “Regenerate Table” button is necessary to click. The image point is obtained by clicking “Get Image Points” and auto fill the result into the table.



- ◆ Path
  - Number of Points: Sets the number of points to do calibration.
  - Move offset X: Sets the X offset distance between points and points.
  - Move offset Y: Sets the Y offset distance between the point and the point
  - Regenerate Table: Clicks the button to regenerate the point according to the set value. When the starting position of the robot is modified, the table needs to be regenerated.

- Step 2: Tool Frame

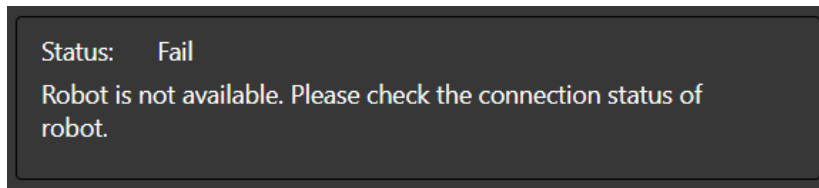


- ◆ Update Start Pose: After clicking the button, the software will update the arm start pose coordinates. If the starting posture of the outer arm is changed, users need to click the update button.
- ◆ Base Point: The base coordinate point. The robot will move attempting to bring the results of the 2D alignment close to the Base point's result until the error is less than the tolerance value.
- ◆ Tool Points: The robot will start from each tool point, moving the robot to try to close the 2D align result to base point.
- ◆ Sets the position of the moving coordinates, including X, Y, and angle. The software will correct the coordinates and datum points after the rotation angle. Users can add the point coordinates by clicking **+**, and delete selected coordinates by clicking **-**. Users can also move the robot to the position and click Teach to write the point.
  - Auto Mode: Users need to provide the initial robot pose for each tool point and ensure that the target object is successfully aligned.

Step2: Tool Points								+	-
	Robot X	Robot Y	Robot RZ	Image X	Image Y	Angle	Res.		
1	100	100	15	0	0	0	-1		
2	0	0	0	0	0	0	-1		
3	100	100	40	0	0	0	-1		

- Manual mode: Continuously moving robot position and verify target is well alignment via image manually. Then record the current robot pose in the tool point table.

- Step 3: Model Train: Clicks the button to start calibration. The calibration status and error messages will be displayed as below.



- ◆ Automatic mode: Clicks the training button, the software will automatically move the arm to perform a series of data collection, and finally calculate a 3x3 transform matrix.
- ◆ Manual mode: Clicks the training button, the transform matrix will be calculated through the point data manually collected by step1 and step2.

Output:

- Name: The name of the function block.
- Type Name: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- IsTransformValid: Is the transformation matrix valid or not.
- RotationRelation: The rotation relationship between the image coordinates and the robot coordinates (in the same direction or in the opposite direction).
- Transform: The matrix of the conversion between the image coordinates and the robot coordinates.

## 4.19 Mathematics

### 4.19.1 MatInverse

Function: Calculates the inverse matrix of the matrix.

Input:

- Input: Enters the matrix for the calculation of the inverse matrix.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: Outputs the inverse matrix result.

## 4.19.2 MatMul

Function: Matrix multiplication.

Input:

- Input 1: Enters the first matrix information.
  - A: Matrix for multiplying matrices.
  - ColumnA: The number of columns in matrix A.
  - RowA: The number of rows in matrix A.
- Input 2: Enters the second matrix information.
  - B: Matrix for calculating matrix multiplication
  - ColumnB: The number of columns in matrix B.
  - RowB: The number of rows in matrix B.

Output:

- Name: The name of the function block.
- TypeName: The name of the component type.
- Status: The current status. "true" represents normal execution, "false" represents abnormal execution.
- ErrorMessage: Shows the reasons when the execution is abnormal.
- LastRunTime: Current run time of the function block.
- Output: Outputs the result of matrix multiplication.
- ColumnOutput: The number of columns in the output result matrix.
- RowOutput: The number of rows in the output result matrix.

# Chapter 5

## Plug-in Development

Objectives:

1. Customize flow elements.
2. Customize inputs/outputs and link them with parameters in the flow.
3. Reuse the customized input parameter setting page in the Run mode view.

### 5.1 Plug-in Installation, Debugging and File Description

#### 5.1.1 Plug-in Installation Debugging

##### 5.1.1.1 Development Tools

Development Tools:

- .NET 6
- Visual Studio

UI Suite:

- Avalonia

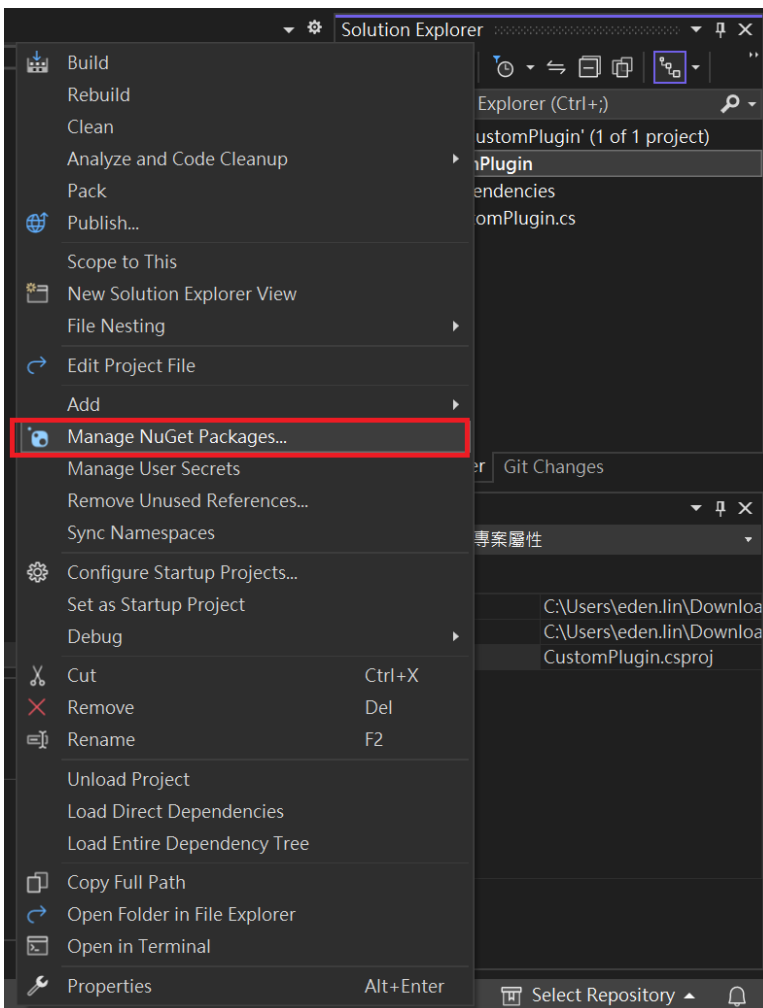
Note: DIAVision has several built-in plug-in samples, which can be stored in C:\Program Files\Delta Industrial Automation\DIASystem\DMV-IVS\PluginSamples

Project References:

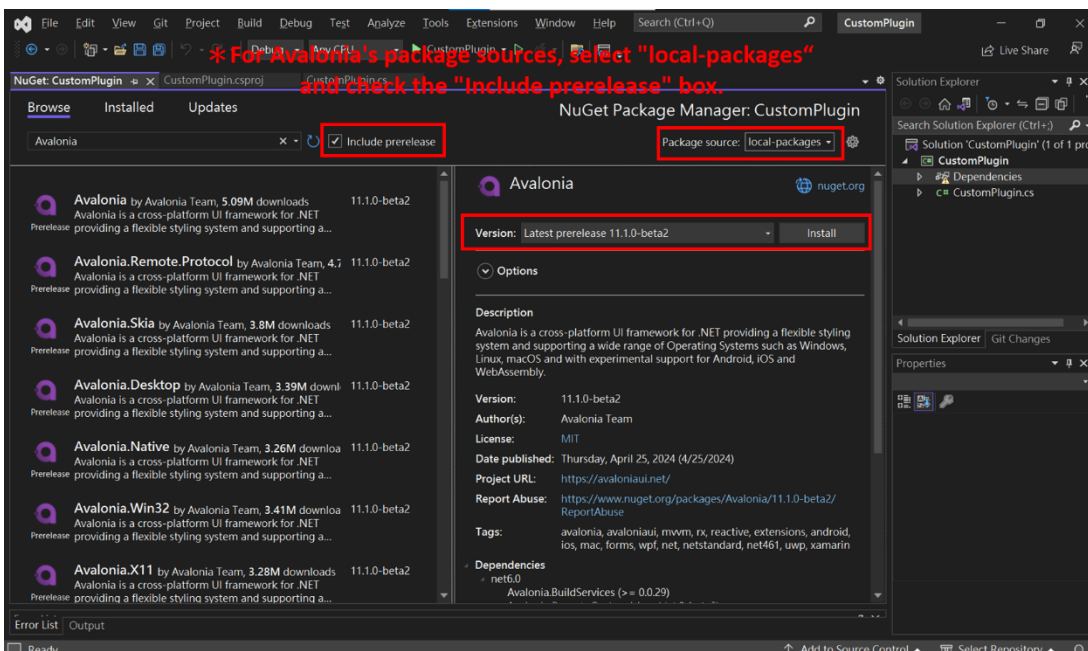
- DIAVision.Core
- DIAVision.UI.Base (UI required)

##### 5.1.1.2 Installation Kits

Right-click on the project and click "Manage NuGet Packages" to open the operation interface to install the package.



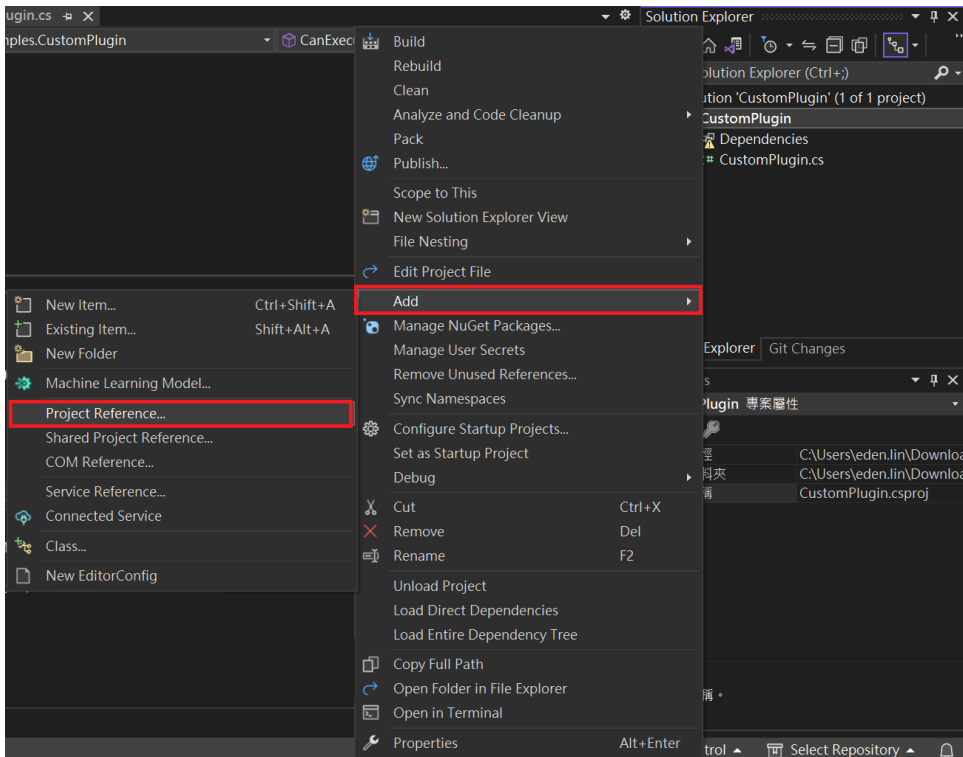
For Avalonia's package sources, select "local-packages" and check the "Include prerelease" box.



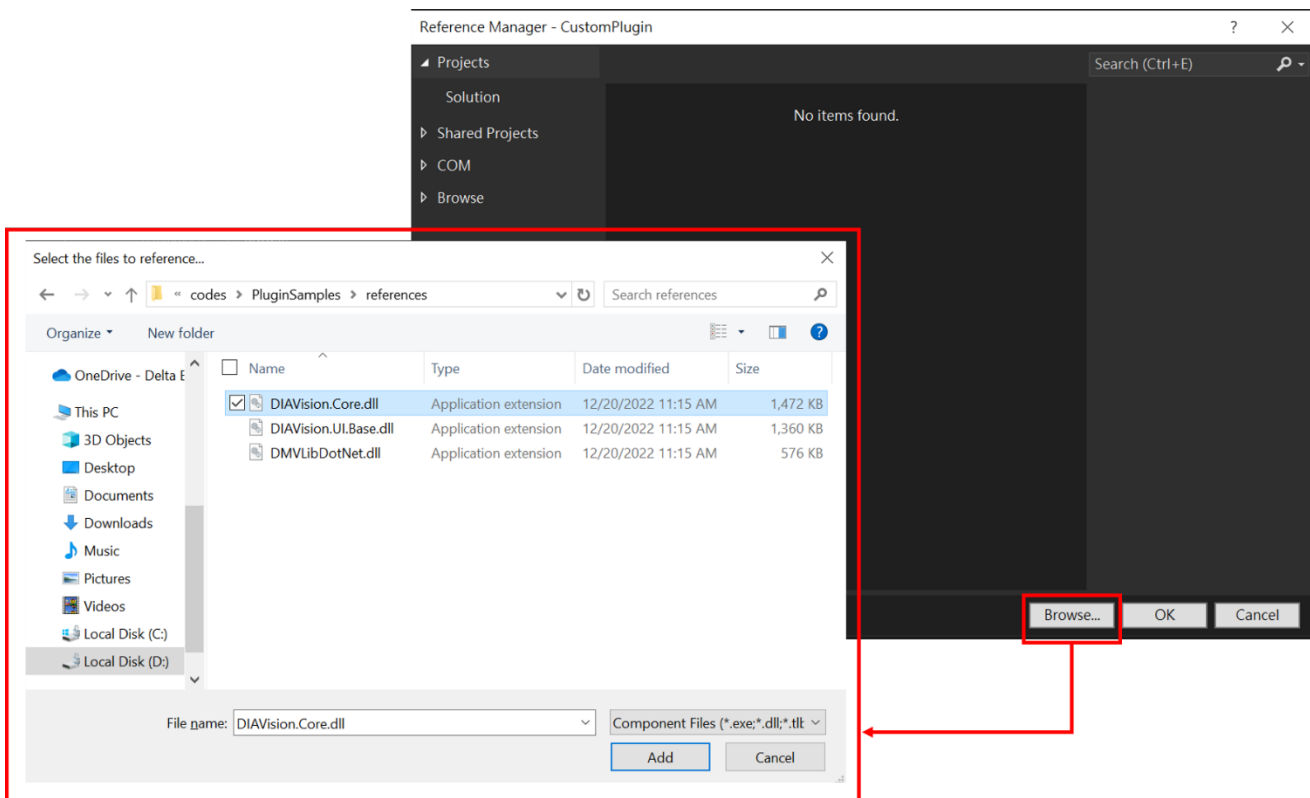


### 5.1.1.3 Add Project Reference

Right-click the project, "Add" > "Project Reference", and open "Reference Manager".



Clicks "Browse" to select the dll file users want to add, select DIAVision.Core and DIAVision.UI.Base (UI requirements apply)



### 5.1.1.4 Plug-in Compilation

The following is an example of the plugin ImageMinMax (the path is <DIAVision directory>\DMV-IVS\PluginSamples\ImageMinMax)

- Creates a project with Visual Studio Code.
- Opens the ImageMinMax.cs file.
- Uses the terminal command "dotnet build" to create {plug-in name}.dll.

### 5.1.1.5 Plug-in Installation

The following is an example of the plug-in ImageMinMax to explain how to install the plug-in:

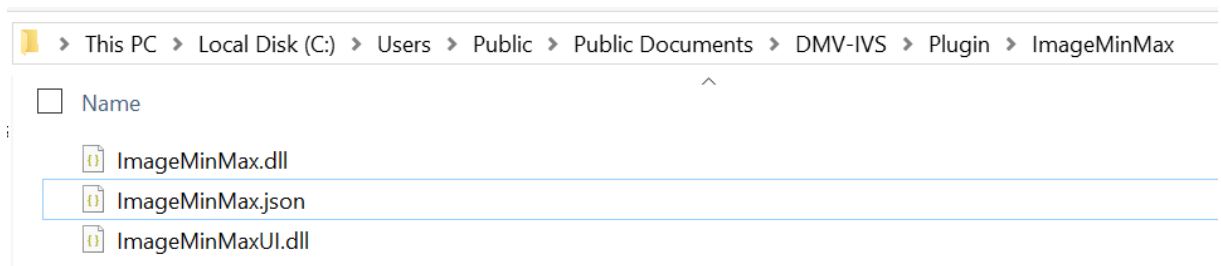
- Writes the plug-in description in json format, as shown in the following figure.

```

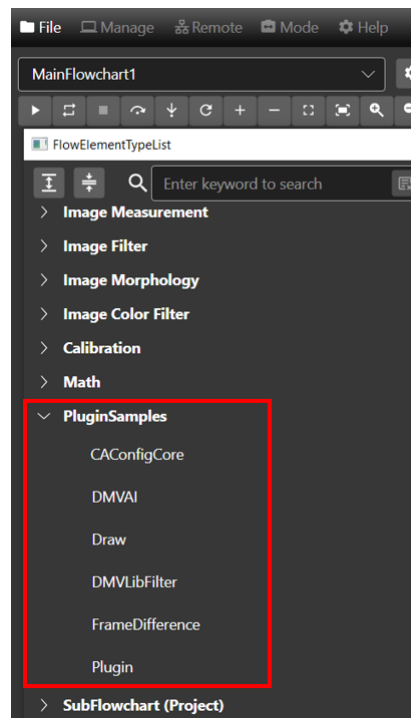
{} ImageMinMax.json
C: > Users > Public > Documents > DMV-IVS > Plugin > ImageMinMax > {} ImageMinMax.json > ...
1  {
2    "typeNames": [
3      "PluginSamples.ImageMinMax"
4    ],
5    "assemblyFileNames": [
6      "ImageMinMax.dll"
7    ],
8    "assemblyFileNames": [
9      "ImageMinMaxUI.dll"
10   ],
11   "IsSupportedWinX64": true,
12   "IsSupportedLinuxX64": true,
13   "IsSupportedLinuxArm64": true
14 }

```

- Copies the json file and all the required dll files to C:\Users\Public\Documents\DMV-IVS\Plugin\{PluginName}/\*.dll & .json, as shown below.



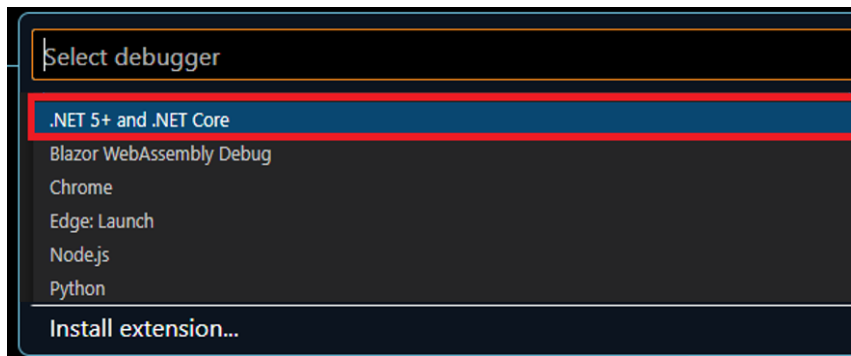
- Starts DIAVision.
- The newly added plugins are displayed in the <FlowElementTypeList>, as shown in the following figure.



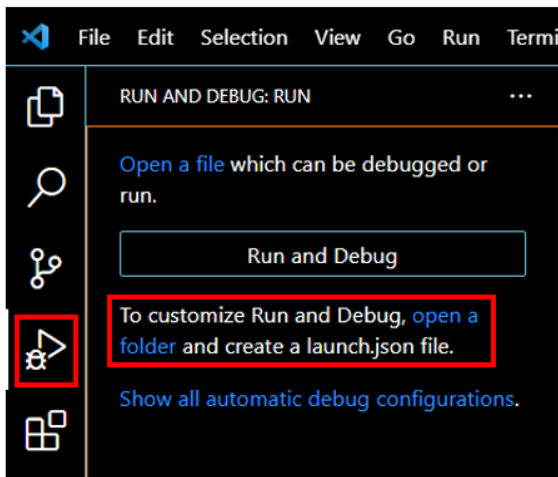
### 5.1.1.6 Plug-in Debugging

The following is an example of a plugin, ImageMinMax, that explains how to debug the plugin:

- Clicks Run > Debug and Visual Studio Code will create launch.json files (select .net5.0 and .net core).



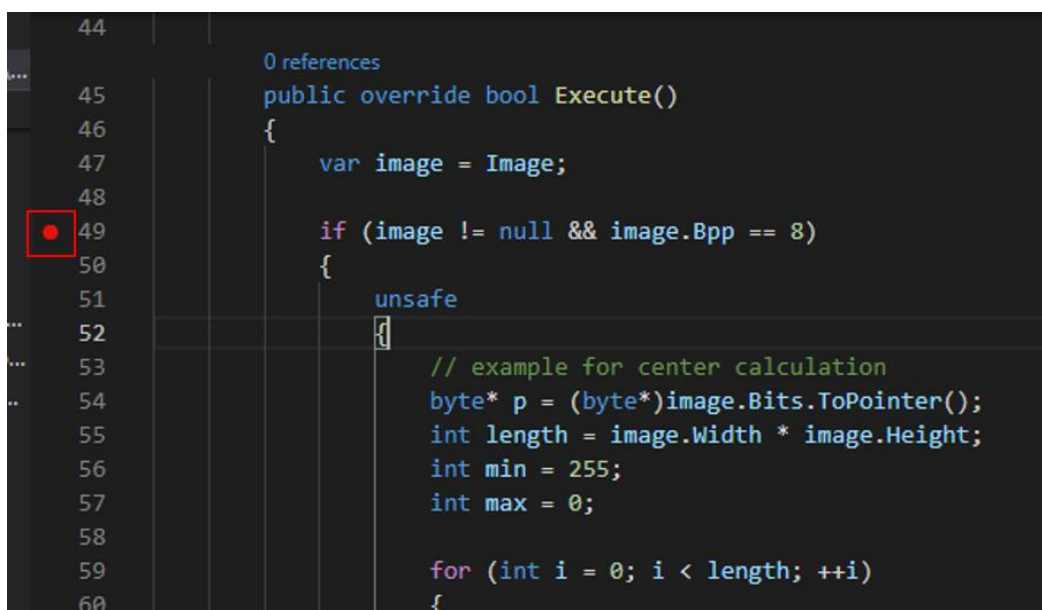
- If users can't create a launch.json automatically, please manually click "Create a launch.json File" in the Run & Debug tab.



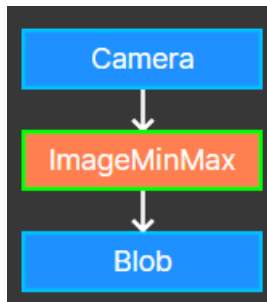
- Edits launch.json file to the following code and save it.

```
{
  Use IntelliSense to learn about possible attributes.
  Hover to view descriptions of existing attributes.
  For more information, visit: https://go.microsoft.com/fwlink/?linkid=830387
  "version": "0.2.0",
  "configurations": [
    {
      "name": ".NET Core Attach",
      "type": "coreclr",
      "request": "attach",
      "processName": DIAVision.UI.exe
    }
  ]
}
```

- Sets breakpoints in the plugin code.



- Run Debugging: Starts debugging in the Debug menu.
- The ImageMinMax plug-in is added to the software platform to operations in the flow. Sets the breakpoint to the ImageMinMax.cs -> Execute function, that is, the breakpoint is triggered when the ImageMinMax component is executed in DIAVision.UI.



- Debugging is performed by operating the DIAVision plug-in with breakpoint settings and function bar.

```

46     {
47         var image = Image;
48
49         if (image != null && image.Bpp == 0)
50         {
51             unsafe
52             {
53                 // example for center calculation
54                 byte* p = (byte*)image.Bits.ToPointer();
55                 int length = image.Width * image.Height;
56                 int min = 255;
57                 int max = 0;
58
59                 for (int i = 0; i < length; ++i)
60                 {
61                     if (p[i] >= RangeStart && p[i] <= RangeEnd)
62                     {
63                         min = Math.Min(min, p[i]);
64                         max = Math.Max(max, p[i]);
65                     }
66                 }
67             }
68         }
69     }
  
```

## 5.1.1.7 Developing Plug-in

### 5.1.1.7.1 Developing Plug-in Functions

This section uses ImageMinMax to explain how to write a plugin.

- Visual Studio Code opens the ImageMinMax plug-in project.
- Opens the ImageMinMax.cs file.

```

1 using System;
2 using DIAVision.Core.CommonTypes;
3 using DIAVision.Core.FlowElements;
4 using DIAVision.Core.FlowElements.Attributes;
5
6 namespace PluginSamples
7 {
8     [UIEditor("PluginSamples.ImageMinMaxUI")]
9     [Category("PluginSamples")]
10    public class ImageMinMax : FlowElement
11    {
12        [Input]
13        [Linkable]
14        [InitialValue("Camera.Image")]
15        public Image Image
16        {
17            get
18            {
19                return (Image)GetInputValue(nameof(Image));
20            }
21        }
22
23        [Input]
24        [Linkable]
25        [InitialValue("10")]
26        public int RangeStart => (int)GetInputValue(nameof(RangeStart));
27
28        [Input]
29        [Linkable]
30        [InitialValue("200")]
31        public int RangeEnd => (int)GetInputValue(nameof(RangeEnd));
32
33
34        [Output]
35        public double Max { get; set; }
36
37
38        [Output]
39        public double Min { get; set; }
40
41        public override bool CanExecute()
42        {
43            return true;
44        }
45
46        public override bool Execute()
47        {
48            var image = Image;
49
50            if (image != null && image.Bpp == 8)
51            {
52                unsafe
53                {
54                    // example for center calculation
55                    byte* p = (byte*)image.Bits.ToPointer();
56                    int length = image.Width * image.Height;
57                    int min = 255;
58                    int max = 0;
59
60                    for (int i = 0; i < length; ++i)
61                    {
62                        if (p[i] >= RangeStart && p[i] <= RangeEnd)
63                        {
64                            min = Math.Min(min, p[i]);
65                            max = Math.Max(max, p[i]);
66                        }
67                    }
68
69                    Max = max;
70                    Min = min;
71                }
72
73                return true;
74            }
75        }
76    }

```

- ImageMinMax needs to reference the following namespaces:
  - `using System;`
  - `using DIAVision.Core.CommonTypes;`
  - `using DIAVision.Core.FlowElements;`
  - `using DIAVision.Core.FlowElements.Attributes;`
- ImageMinMax inherits the FlowElement class, so DIAVision can add ImageMinMax to the flowchart.

```

public class ImageMinMax : FlowElement
{
}

```
- ImageMinMax will use customized UI and to display ImageMinMax's plug-in class in DIAVision's Add Element List View screen, so add class attribute description (see [4.2 plug-in attribute](#)).
  - `[UIEditor("PluginSamples.ImageMinMaxUI")].`
  - `[Category("PluginSamples")].`
- ImageMinMax uses 3 inputs: Image class, 2 integers RangeStart and RangeEnd, and all 3 inputs are set and initialized with input attribute (see [4.2 plug-in attribute](#)).
  - `[Input]`

```

[Linkable]
[InitialValue("Camera.Image")].
public Image Image
{
    get
    {
        return (Image)GetInputValue(nameof(Image));
    }
}

```

```

[Input]
[Linkable]
[InitialValue("10")]
public int RangeStart => (int)GetInputValue(nameof(RangeStart));

```

```

[Input]
[Linkable]
[InitialValue("200")]
public int RangeEnd => (int)GetInputValue(nameof(RangeEnd));

```

- ImageMinMax generates 2 double outputs: Max and Min, and is set using the output attribute (see 4.2 plug-in attribute).

```

[Output]
public double Max { get; set; }

```

```

[Output]
public double Min { get; set; }

```

- When the project is archived, all input values will be saved to the project file (path: C:\Users\Public\Documents\DMV-IVS\Projects\[ProjectName]\[ProjectName].xml). If you need to save other data, override the "SaveCustomData" and "LoadCustomData" functions defined in the InputObject.
- ImageMinMax will find the maximum (Max) and minimum (Min) values of pixels from the input Image, so you need to override the "Execute" and "CanExecute" of the FlowElement class (both are inherited from the FlowElement class).

```

public override bool CanExecute()
{
    return true;
}

```

```

public override bool Execute()
{
    var image = Image;

    if (image != null && image.Bpp == 8)
    {
        unsafe
        {
            example for center calculation
            byte* p = (byte*)image.Bits.ToPointer();
            int length = image.Width * image.Height;
            int min = 255;
            int max = 0;

            for (int i = 0; i < length; ++i).
            {
                if (p[i] >= RangeStart && p[i] <= RangeEnd)
                {
                    min = Math.Min(min, p[i]);
                    max = Math.Max(max, p[i]);
                }
            }

            Max = max;
            Min = min;
        }
    }

    return true;
}

```

#### 5.1.1.7.2 Write the Plug-in User Interface

If users need to set up the UI, ImageMinMax uses the custom UI as an example. Opens the ImageMinMax.xaml.cs file, class inherits the PageEditor class, and the content is as follows.



```

ImageMinMax.xaml.cs
C: > Program Files > Delta Industrial Automation > DIAVision > DMV-IVS > PluginSamples > ImageMinMax > UI > ImageMinMax.xaml.cs
1  using Avalonia.Markup.Xaml;
2  using DIAVision.UI.Base;
3
4  namespace PluginSamples
5  {
6      // example for ui plugin. it's test only now
7      public class ImageMinMaxUI : PageEditor
8      {
9          public ImageMinMaxUI()
10         {
11             InitializeComponent();
12         }
13
14         private void InitializeComponent()
15         {
16             AvaloniaXamlLoader.Load(this);
17         }
18     }
19 }

```

- Displays the UI content of ImageMinMax is written in xaml files. This is shown in the following code.

```

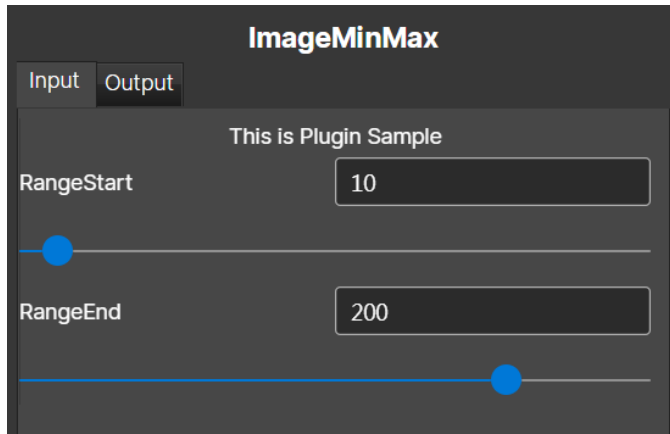
1  <UserControl xmlns="https://github.com/avaloniaui"
2             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
3             xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
4             xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
5             mc:Ignorable="d" d:DesignWidth="800" d:DesignHeight="450"
6             x:Class="PluginSamples.ImageMinMaxUI">
7      <Grid>
8          <StackPanel>
9              <TextBlock Text="This is Plugin Sample" HorizontalAlignment="Center" Margin="5"/>
10             <UniformGrid Columns="2">
11                 <TextBlock Text="RangeStart" VerticalAlignment="Center" />
12                 <TextBox x:Name="TextBoxRangeStart"
13                     Text="{Binding FlowElement.InputExpressions[RangeStart].ExpressingString}"/>
14             </UniformGrid>
15             <Slider Value="{Binding #TextBoxRangeStart.Text}"
16                 Minimum="0" Maximum="255" TickFrequency="1" IsSnapToTickEnabled="true" />
17             <UniformGrid Columns="2">
18                 <TextBlock Text="RangeEnd" VerticalAlignment="Center" />
19                 <TextBox x:Name="TextBoxRangeEnd"
20                     Text="{Binding FlowElement.InputExpressions[RangeEnd].ExpressingString}"/>
21             </UniformGrid>
22             <Slider Value="{Binding #TextBoxRangeEnd.Text}"
23                 Minimum="0" Maximum="255" TickFrequency="1" IsSnapToTickEnabled="true" />
24         </StackPanel>
25     </Grid>
26 </UserControl>

```

#### Help:

- Line 1~6: Use custom controls and set the UI to the `x:Class="PluginSamples.ImageMinMaxUI"` class.
- Line 9: The UI screen displays a line of center text "This is Plugin Sample".
- Line 11~12: The RangeStart label text and RangeStart input controls are displayed.
- Line 15: Displays a slider and binding the input RangeStart.
- Line 18~19: The RangeEnd label text and the RangeEnd input control items are displayed.
- Line 22: Displays a slider and binding the input RangeEnd.

- The plug-in is compiled and installed in DIAVision, and ImageMinMax is added to the flowchart, as shown in the UI shown below.



## 5.2 Plug-in Attribute Description

- Input and output attributes

The name of the property	description
Input	input parameter
InitialValue	input parameter initial value
Linkable	The [Input][Linkable] parameter allows you to connect to other input/output elements
DynamicName	Set the parameter menu displayed in the UI by overriding the "GetDynamicValueNames" function. Reference Example: RobotMove.cs
DynamicInitialValue	Set the initial value by overriding the "GetDynamicInitialValue" function. Reference Example: RobotMove.cs
Category	Displays the parameter class
Description	Description displayed when the mouse hovers over the parameter name on the UI
Output	output parameter
Command	Display the command button on the UI, click the button and slider the function
SupportedROIs	The ROI supported by the ROI parameter type

- Plugin class attribute

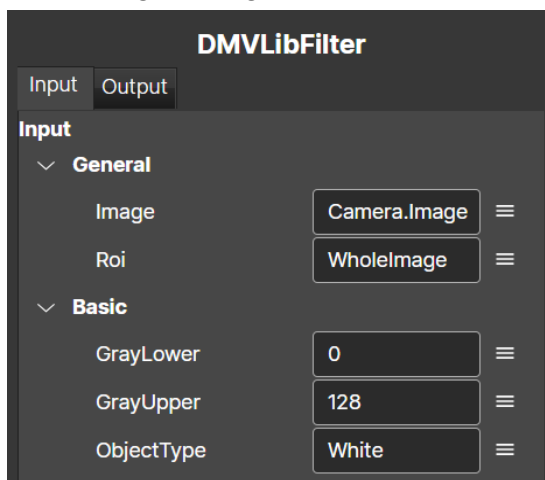
The name of the	description
-----------------	-------------

property	
Category	Plug-in category displayed on the Add Element List View
Description	Displays a description
UIEditor	UI view class, plugin class setting
IEditorViewModel	UI view model class, plugin class setting

## 5.3 Plug-in UI

### (1) Default UI

If no customized UI, DIAVision will use the default UI, as shown below, with the Input block expanding/closing parameters and input fields based on classification.



### (2) Customize the UI

- UI Framework

The custom UI uses the AvaloniaUI Framework, and the custom UI will be displayed in the original Input block.

AvaloniaUI uses the MVVM (Model-View-Viewmodel) architecture, which is written in the same markup language as Microsoft's .net WPF syntax, for details, please refer to the AvaloniaUI documentation: <https://docs.avaloniaui.net/>.

- C# (View Model)

Refers to Sample: ImageMinMax.xaml.cs.

- XAML (View)

Adds xaml file to write View content.

Example: ImageMinMaxUI.xaml.

## 5.4 Getting Image and Point Cloud Raw Aata

### (1) Getting image data

Lin 4: The grayscale value of the image.

Lin 5: If the unit of the image is mm, the value users get is also in mm.

Line 9, 11: Uses unsafe syntax to get image raw data by calling (byte\*)<image parameter>.Bits.ToPointer() function.

example: unsafe function block in ImageMinMax.cs program code.

Lin 11: Uses byte to get the 8-bit image value.

Lin 15: When the image UnitZ is None, the ScaleZ of the image is 1, and the OffsetZ is 0, the resulting imageDataWithUnit is the same as the imageRawData value.

Lin 24: Use "ushort" to get the 16-bit image value.

```

1 public void GetImageData()
2 {
3     var image = Image;
4     var imageRawData = new double[image.Width * image.Height];
5     var imageDataWithUnit = new double[image.Width * image.Height];
6
7     if (image.Bpp == 8)
8     {
9         unsafe
10        {
11            byte* imgInPtr = (byte*)image.Bits.ToPointer();
12            for (int i = 0; i < image.Width * image.Height; i++)
13            {
14                imageRawData[i] = imgInPtr[i];
15                imageDataWithUnit[i] = (imgInPtr[i] * image.ScaleZ) + image.OffsetZ;
16            }
17        }
18    }
19
20    if (image.Bpp == 16)
21    {
22        unsafe
23        {
24            ushort* imgInPtr = (ushort*)image.Bits.ToPointer();
25            for (int i = 0; i < image.Width * image.Height; i++)
26            {
27                imageRawData[i] = imgInPtr[i];
28                imageDataWithUnit[i] = (imgInPtr[i] * image.ScaleZ) + image.OffsetZ;

```

- (2) Gets the Point Cloud raw data
- (3) Uses unsafe syntax to get point cloud raw data by calling (float\*)<pointcloud parameter>.Bits.ToPointer() function.

Reference example: unsafe function block in PointCloudCenter.cs code.

```
unsafe
{
    float* p = (float*)points.PointsPtr.ToPointer();
    long length = points.Count;
    double sumX = 0;
    double sumY = 0;
    double sumZ = 0;
    for (long i = 0; i < length; ++i).
    {
        sumX += p[i * 3];
        sumY += p[(i * 3) + 1];
        sumZ += p[(i * 3) + 2];
    }

    X = sumX / length;
    Y = sumY / length;
    Z = sumZ / length;
}
```



**Delta Electronics Industry Co., Ltd**  
**Mechanical and electrical business group**  
33068 No. 18, Xinglong Road, Taoyuan District, Taoyuan City  
TEL: 886-3-3626301  
FAX: 886-3-3716301

\*The contents of this manual are subject to change without notice