IMAGO TECHNOLOGIES

Manual

Vision Cam AI.go

Version 1.7.0.0 – December 2023

Contents

1 Handling and Safety Instructions

Depending on the operating conditions, the housing temperature can exceed 60 °C. There is a risk of injury!

Caution! LED risk group 2. Do not look directly into the LED flash!

EMC conformity according to EN/IEC 61000-6-2:2005 is qualified for cable lengths ≤ 30 m.

Electrical installation should be executed without power applied to the device and all connected devices.

Please take special note of the voltage range which may be applied to the device. Otherwise, permanent damage to the device may result!

Due to the characteristics and physical principles inside flash memory, **memory cards have a finite lifetime** dictated by the number of write operations. Therefore, take care of the regular write operations to prevent an early flash damage.

2 Introduction

If tasks in industrial machine vision can easily be formalized, traditional systems are usually the right choice. However, they often reach their limits when the defects are hard to formalize. On top of that, the implementation of such systems requires an image-processing specialist. Machine vision systems based on deep learning technologies often offer a better solution.

Vision Cam AI.go from IMAGO Technologies is a very flexible complete embedded deep learning based machine vision system. Vision Cam AI.go was mainly developed for end-users with little or no experience in the fields of programming or machine vision. The high flexibility of the deep learning camera Vision Cam AI.go allows the use of the system in numerous industrial areas of application in order to ensure a reliable check of quality features.

2.1 Main features

- 21 - 28 V DC power supply
- Passive cooling without heat sink
- Image sensors:
  - 2560 × 1936 pixels
  - Global shutter
  - Monochrome or Color
- Lens:
  - C-Mount 8mm or 16mm
- Digital inputs / outputs:
  - 4× digital output
  - 2× digital input
- Ethernet interface 1000 Mbit/s
- Housing:
  - 21 - 28 V_{DC} power supply
  - Passive cooling without heat sink
- ViewIT
  - Web-GUI available via the IP-address of the device
  - Image acquisition
  - Image processing operators
  - OPC UA and REST API
  - Easy handling of parameters
  - Storing of images

3 Operating Conditions

Power Supply:

Parameter	Min.	Typ.	Max.	Unit
Supply voltage	21	24	28	V
Supply current (@24V)			0.5	A

Digital Input:

Parameter	Min.	Typ.	Max.	Unit
Input voltage range	0		25	V
Rising edge threshold voltage	7.4		9.4	V
Falling edge threshold voltage	4.7		6.3	V
Input resistance		15.5		kΩ

Digital Output:

Parameter	Min.	Typ.	Max.	Unit
Output current			50	mA
Output high voltage		V_{Supply} - 0.2		V

Environment:

Parameter	Value	Unit
Weight, including cable, lens, and tube	370	g
Operating temperature	5 … 45	°C
Operating humidity, relative, non-condensing	5 … 95	%
Storage temperature	-30 … +70	°C
Storage humidity, relative, non-condensing	5 … 95	%

4 Interfaces

4.1 Power and I/O

A non-shielded, 0.5 meter cable with an M12 8-pin connector is used for power supply and I/O signals.

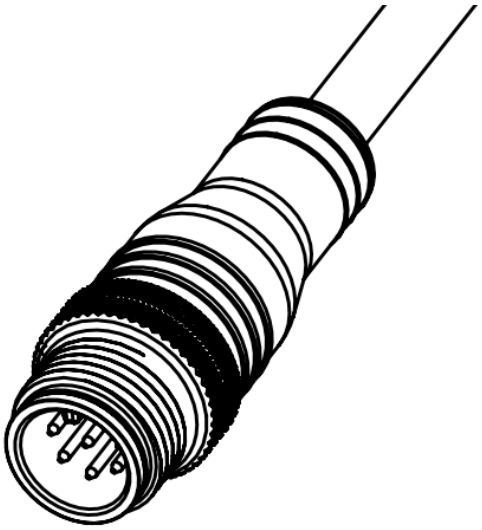


Figure 1: Power supply and I/O cable

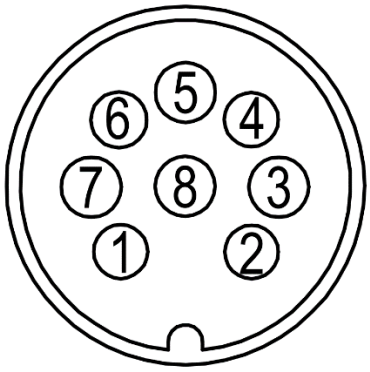


Figure 2: Pin assignment M12 connector

Pin	Function
1	Power (21V-28V)
2	GND
3	Digital IN0
4	Digital IN1
5	Digital OUT3
6	Digital OUT0
7	Digital OUT1
8	Digital OUT2

Table 1: Pin assignment M12 connector

4.1.1 Digital I/Os

The following illustration shows the electrical equivalent circuit for the digital input and output signals:

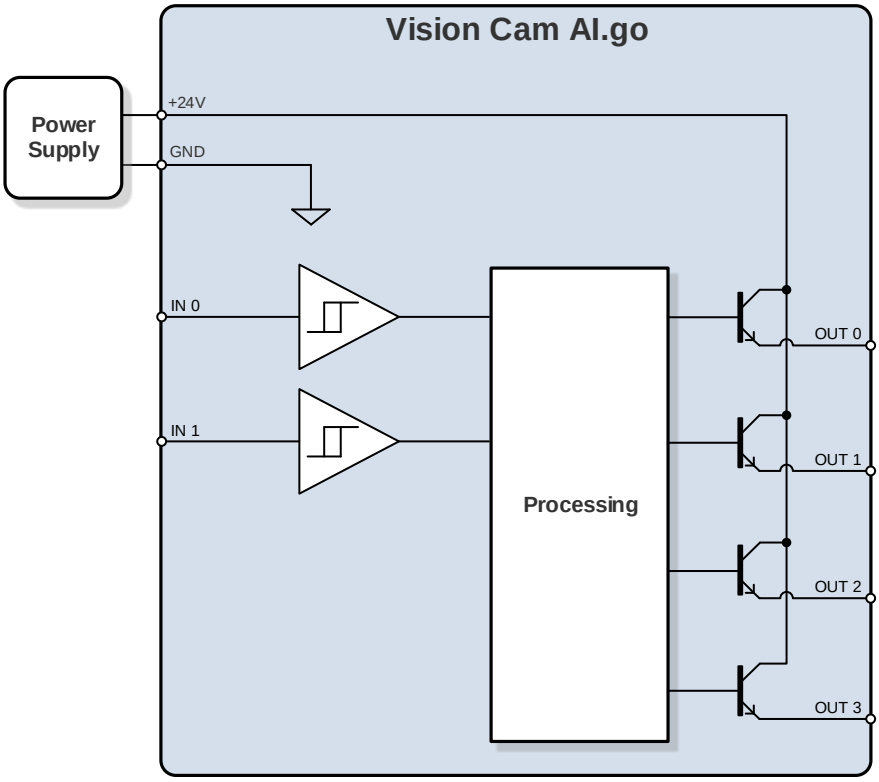


Figure 3: Simplified digital I/O circuit

The input signals use a Schmitt trigger circuit with the power supply GND as voltage reference.

The digital output circuit uses open-emitter configuration. All outputs are internally supplied by the 24V power input.

4.2 1000 Mbit/s Ethernet

A shielded 0.5 m Ethernet cable with a M12 8-pin X-coded connector is used for network communication.

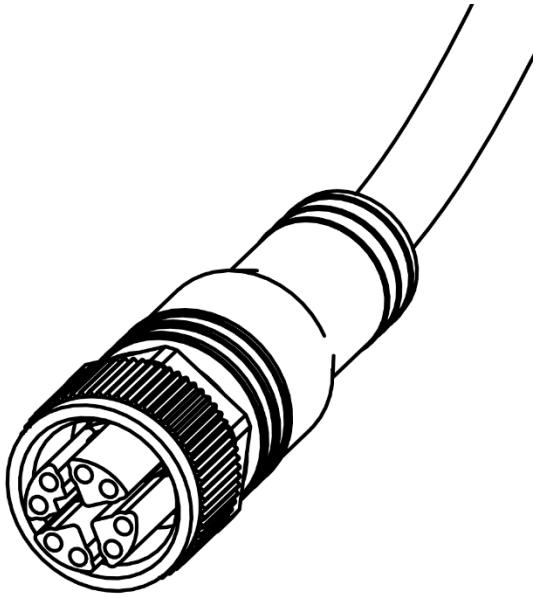


Figure 4: Ethernet cable

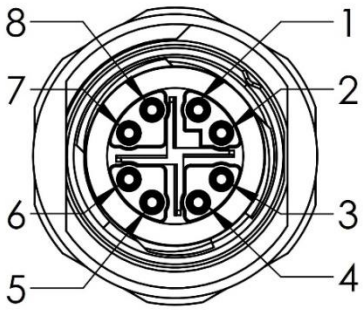


Figure 5: Pin assignment M12 connector

PIN	Function
1	D1+
2	D1-
3	D2+
4	D2-
5	D4+
6	D4-
7	D3-
8	D3+

Table 2: Pin assignment M12 connector

We recommend using shielded cables, for example:

Length	Product	IMAGO order code
1 m	Phoenix contact 1407471 “NBC-MSX/ 1,0-94F/R4AC SCO”	10007049
2 m	Phoenix contact 1407472 “NBC-MSX/ 2,0-94F/R4AC SCO”	10007050
5 m	Phoenix contact 1407473 “NBC-MSX/ 5,0-94F/R4AC SCO”	10008076

4.3 Status LEDs

The device is equipped with two status LEDs, namely “ETH” and “PWR”. The meaning of each status LED and its color is listed in the tables below.

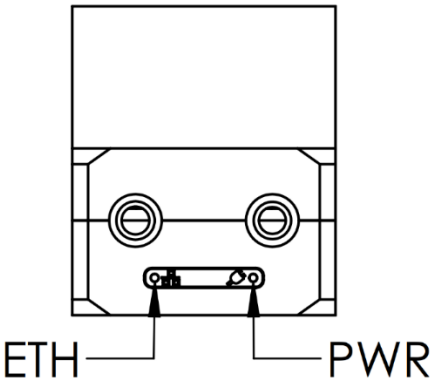


Figure 6: Vision Device status LEDs

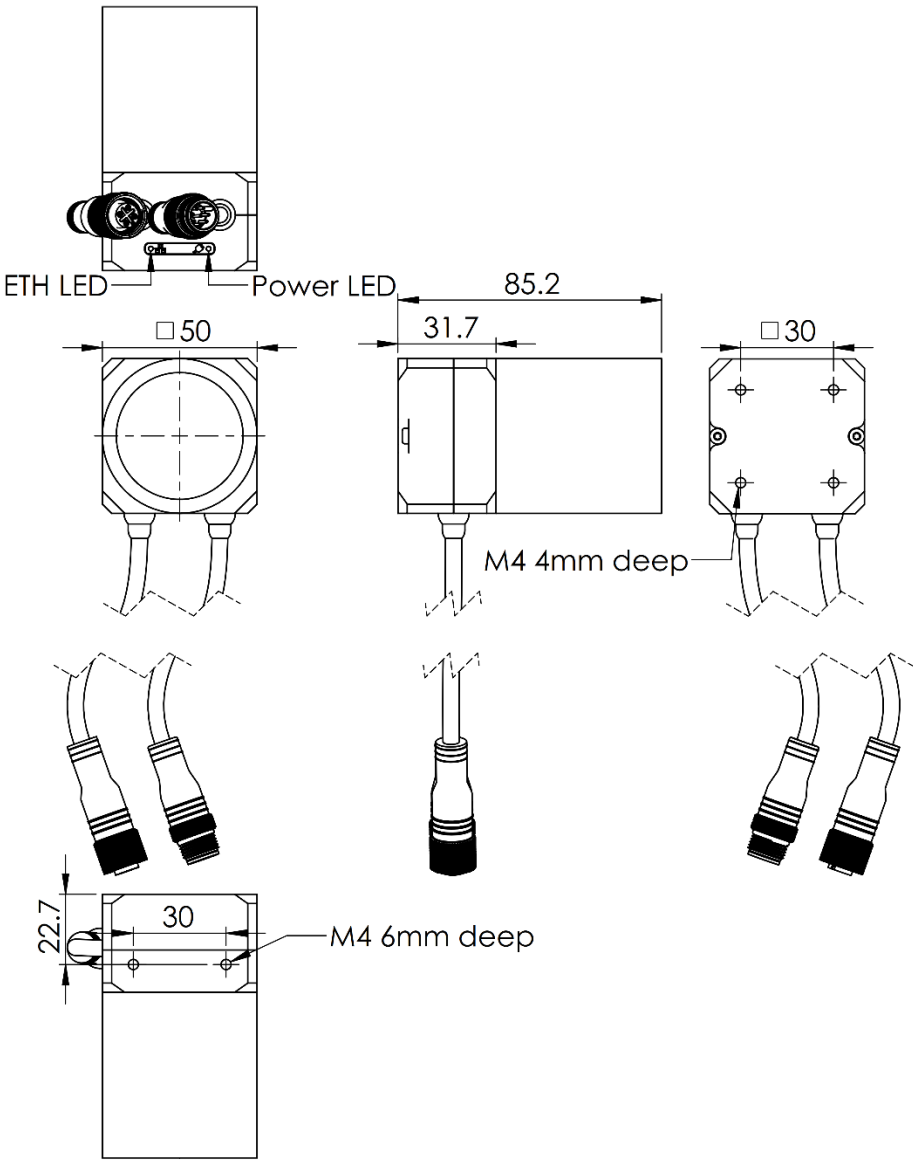
PWR LED	State
Green	Power on
Red	Digital IN0 signal active

Table 3: PWR LED states

ETH LED	State
Green	Ethernet link is up
Red	Ethernet activity

Table 4: ETH LED states

4.4 Mechanical Drawing



5 Framework (ViewIT)

ViewIT is a framework for easily handling image processing functions on IMAGO Technologies devices like Vision Cam XM or Vision Sensor PV. The framework gives access to the live view from the camera, acquisition parameters, general purpose digital inputs and outputs and allows using self-designed image processing algorithms.

5.1 Features

- Intuitive web interface for interaction with human user (HMI)
- OPC UA and REST API for interaction with other devices (M2M)
- Flexible using and debugging of image processing operators (HALCON, general C++ or Python) (Optional OpenCV or other 3rd party libraries)
- Easy handling of parameters and settings
- Simple use of digital inputs and outputs
- Storing and re-analysis of “good” and “bad” images

5.2 Process Cycle

Data processing is done in a cycle. Separate plug-ins for image processing and communication can be selected. Before the cycle starts for the first time, the initialization function of the selected plug-in is called. This can be used to initialize data or read model information for pattern matching.

The processing cycle starts with grabbing an image in triggered or free-run mode. At next a function named `Pre()` from the optional communication (I/O) plug-in is called. The purpose of this function is getting parameters and data from external interfaces like Ethernet sockets or databases. In the next step the values of input parameters are determined by user settings, OPC UA or REST API. It is also possible to modify these input parameters from inside of the communication plug-in function. For example, depending on a value from a database the algorithm checks for red or blue objects.

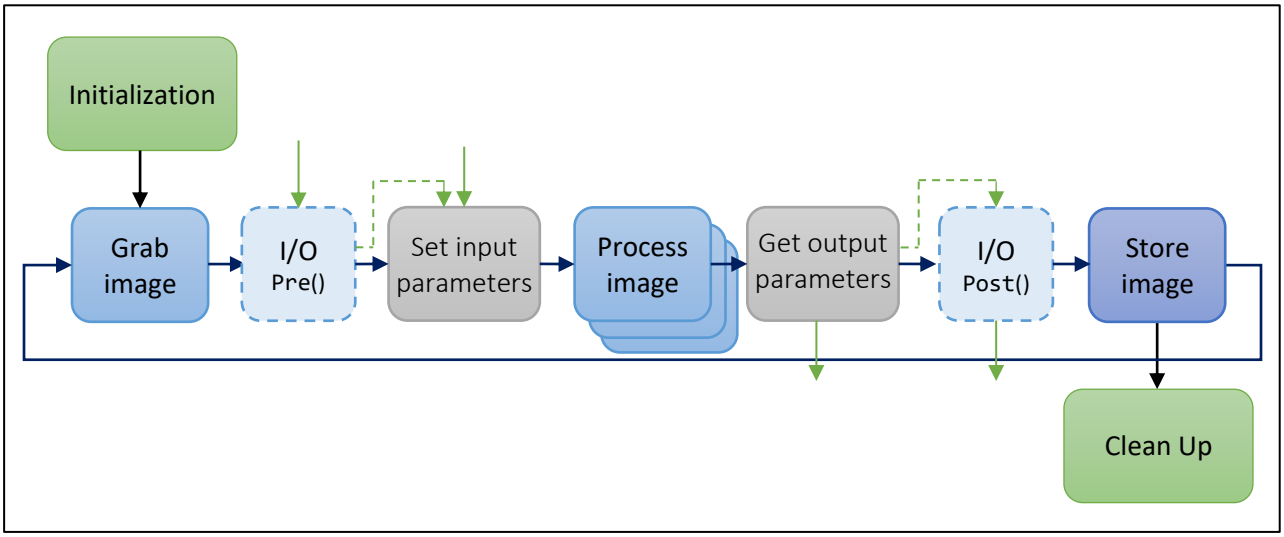


Figure 7: Process Cycle

The image is processed in accordance with these settings. The user can select up to three different image processing procedures which are called one after another. Each procedure can forward its output parameters to the next one.

After this, the resulting values are written, and optionally digital outputs are set. The output parameters from image processing are passed to the second function of the communication plug-in called `Post`(). This can handle the data, for example write a detected code over a network socket. Depending on the result of image processing the grabbed image is stored into the ring buffer for “good” or “bad” images.

Subsequently the process starts again with the acquisition of a new image.

When a different procedure is selected or the application ends, a clean-up function is called. This is useful to free previously allocated resources.

Thus, there are two different types of plug-in procedures, **Image Processing**, and **Communication (I/O)**.

For further information how to write plug-ins see the additional document `ViewIT_Plug-In_API.pdf`. This describes the interfaces of the procedures for HALCON, General C++, Python, and even how to create documents with JupyterLab.

Further information how to use the resources of ViewIT from other applications can be found in the **interactive OpenAPI** documentation, which is accessible from the About menu inside the application.

6 User Interface

Interaction with the human user is done by a web interface which can opened by every modern browser. Chrome, Edge (Chromium) and Firefox have been tested. The OPC UA interface (see section 0) and REST API (see section 6.3) are provided for communication with other devices.

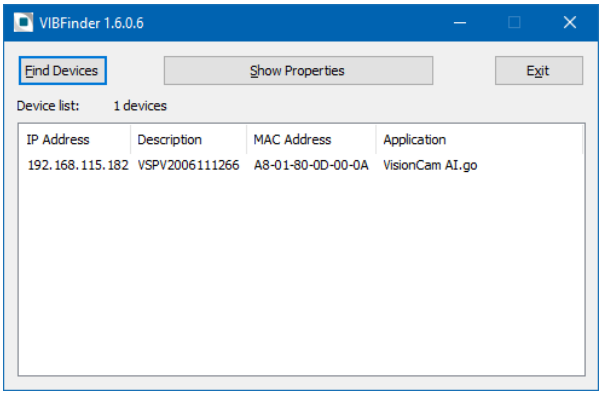
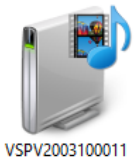


Figure 8: VIBFinder

The IP address of the Vision Device can be static or dynamically assigned by a DHCP server in the local network.

If you do not know the IP address, you can use the **VIBFinder** tool to scan the network for IMAGO devices. The Description column contains the serial number.

Select your device, press right mouse button, and choose Open Web Browser to open the main screen.



Depending on the configuration a device running ViewIT is also visible as **Media Device** in the Windows network environment. Select Properties to get information about the associated web site.

6.1 Web Interface

After entering the IP address of the Vision Cam or Vision Sensor into the address field of a web browser the **Main Screen** appears. Instead of using the IP address you can enter the serial number followed by `.local` into the address field, e. g. `vspv2008111858.local`.

It is also possible to open a secure connection by **HTTPS**. ViewIT is using a self-signed certificate because it is probably running in local network with possibly changing IP addresses. Therefore, a warning is shown in the web browser that opening this web site is a potential security risk. To trust the certificate, select Advanced..., accept the risk and continue with the unsafe web page. This will create a security exception for this web site and the certificate. See section 7.4 how create an own certificate.

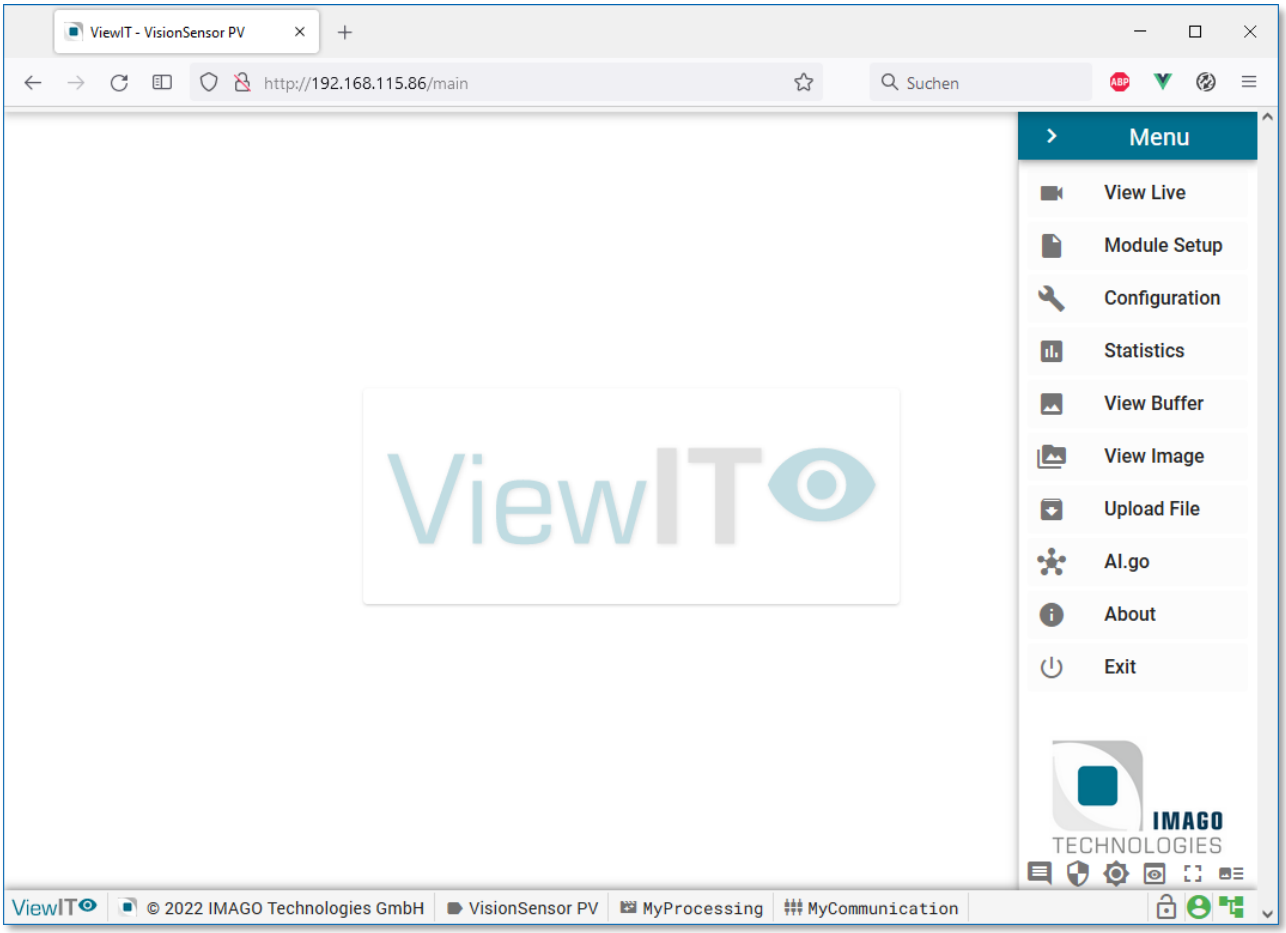


Figure 9: Main Screen

Located on the right side of the browser window is the **Navigation Panel**. It contains links to the pages of the application. The size of the drawer can be reduced by pressing the › button at the top. There are some small buttons at the bottom of the navigation drawer:

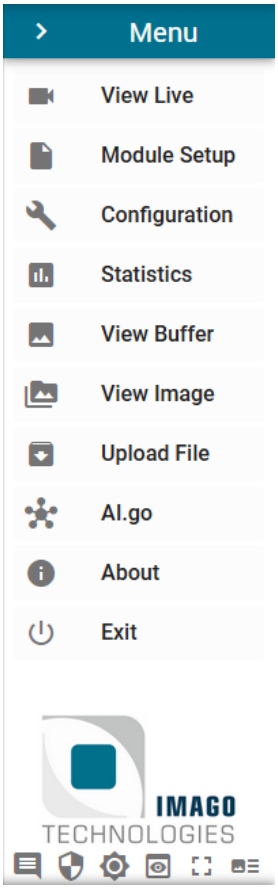
- shows the log window. It contains warnings and error messages.
- will move the navigation drawer to the other side of the browser window.
- opens the “Administrator Login Dialog” (see section 6.1.1.2)
- switches between light and dark user interface theme.
- opens the web interface of selected image processing plug-in

switches to full screen mode of the browser
will move the navigation drawer to the other side of the browser window.

Selecting an item in the navigation drawer will open the corresponding page.
The **Indicator Bar** is located at the bottom of the window. It notifies among other things the current state of authentication and user level.

6.1.1 Navigation Panel

The **Navigation Panel** consists of the buttons for accessing the functions of the Vision Sensor, setting the appearance of the GUI, and displaying the Event Log.



Press the arrow to collapse or expand the panel.
Live View Page (see 6.1.3)
Module Setup Dialog (see 6.1.4)
Configuration Dialog (see 6.1.5)
Statistics Dialog (see 6.1.6)
View Buffer Dialog (see 6.1.7)
View Image Dialog (see 6.1.7)
Upload File Dialog (see 6.1.9)
AI.go Dialog (see 6.1.11)
About Dialog (see 0)
Exit Dialog (see 6.1.12)

Event Log | Login | Toggle theme (dark or bright) | Web Interface | Full screen | Switch side

Figure 10: Navigation Panel in the full state with the function of the buttons

6.1.1.1 Event Log

The **Event Log** contains the messages of the Vision Device, such as error messages etc. It is hidden by default, see Figure 10. In case of any error messages the button will show the number of unread messages in a red circle. Once the button is clicked all the messages are marked as read and remain in the Event Log.

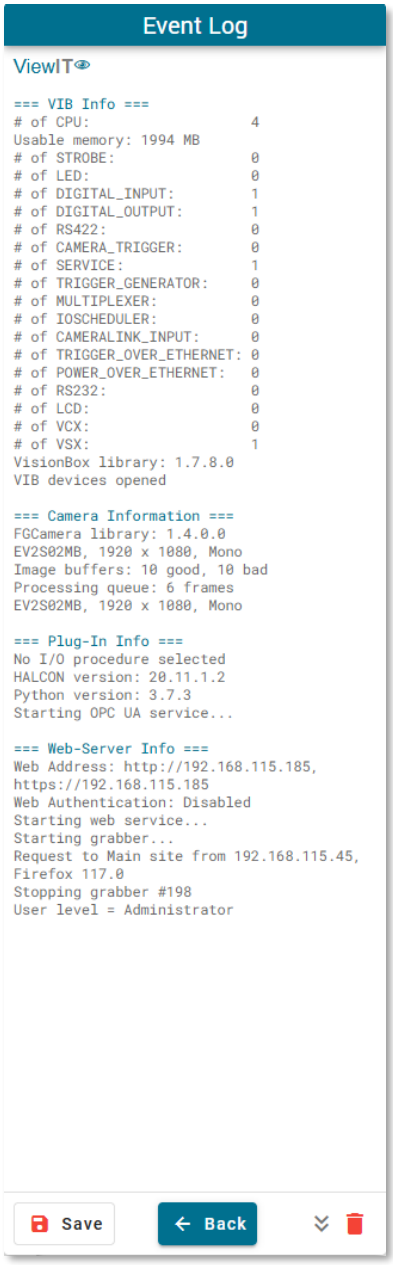


Figure 11: Example of event log

Save writes the event messages to a HTML file in flash memory.

Back closes the window

scrolls to the end of the list

clears the log (only available if user level is Administrator)

6.1.1.2 Administrator Login Dialog

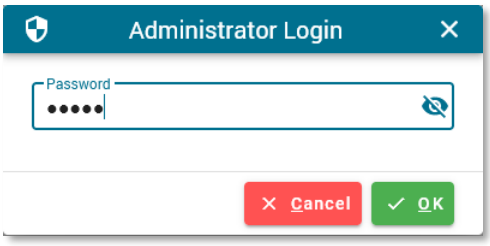


Figure 12: Administrator Login

The application starts in normal user mode. This is indicated by a symbol in the status bar. In this mode the change of settings and writing of data is disabled.

After pressing in the navigation drawer the **Administrator Login** dialog appears. Press to switch between hidden and shown password.

The default administrator password is `admin`.

Cancel closes the dialog.

OK tries to log-in with the entered password.

If the correct password is used, the symbol in status bar changes from to . If a wrong password is entered, the application switches back to normal user mode.

6.1.2 Indicator Bar

The **Indicator Bar** contains several indicators.

There are indicators for the following information:

1. IMAGO Technologies copyright note
2. Name of the system (can be changed on the Configuration page)
3. Name of image processing procedure
4. Name of communication procedure
5. HTTP authentication indicator
6. User level indicator
7. OPC UA authentication indicator

The indicators are numbered in Figure 13, the numbers correspond to the list above. When placing a mouse courser above an indicator a description of its function will appear in a pop-up blob.

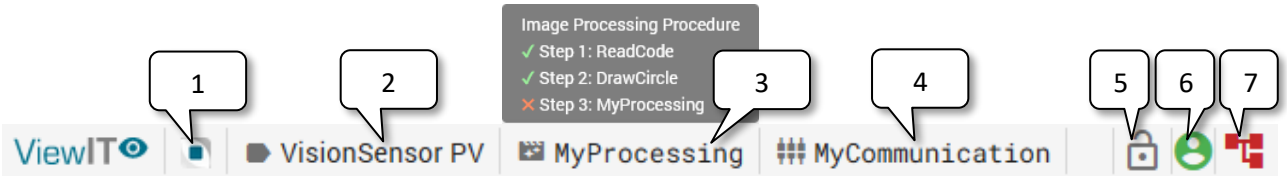


Figure 13: Indicator bar. The name of the Vision Device is “VisionSensor PV”, as indicated by 2.

6.1.3 Live View

In this page the live image data from the camera is visualized.

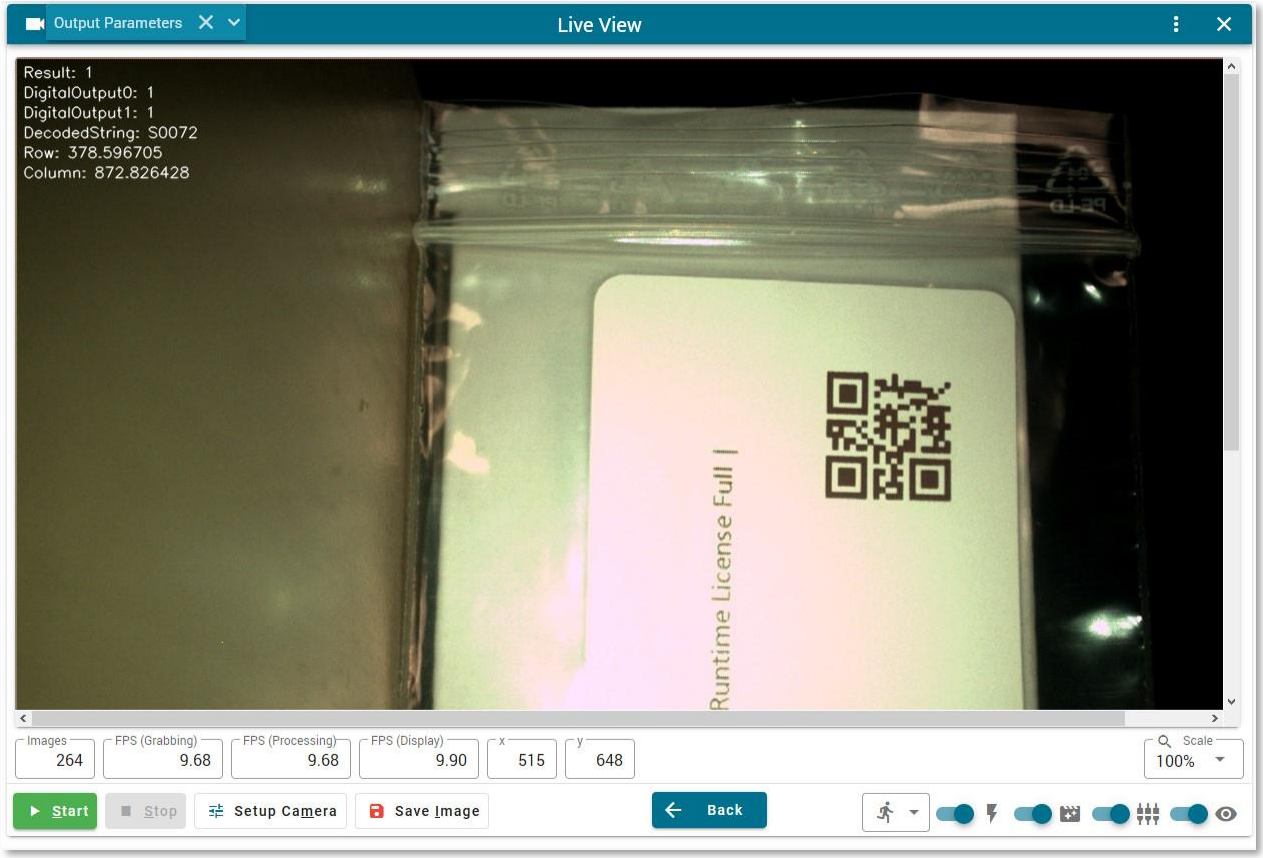


Figure 14: Live View

The status line below the camera image shows the number of grabbed images, frames per second grabbed by the camera, processed and shown in browser window.

If image processing is too slow to process all grabbed images the FPS (Processing) counter is marked.

Scale changes the magnification factor of the displayed camera image.

The action line contains the following controls:

Start enables live view.

Stop disables live view.

Setup Camera opens the Camera Setup dialog (see section 6.1.5.3.1).

Save Image opens the Save Image dialog (see section 6.1.3.1).

Back closes the page and activates the navigation drawer again. You can also press ☒ in the upper right corner of the window.

Some **switches** are placed at the right side of the action line:

changes between Free-Run ( ), Software ( ) or hardware Triggered ( ) camera modes.
In software mode an additional button Trigger appears. Pressing this button takes a single image.

enables / disables internal strobe of camera (if available).

enables / disables image processing.
When image processing is enabled, an additional pop-up window called Output Parameters is visible that shows the output values. You can fold (^) and unfold (˅) this window. Select the parameters from a processing step via the ⋮ symbol.

enables / disables communication.

switches visualization on or off (might influence the performance of the image processing).

6.1.3.1 Save Image Dialog

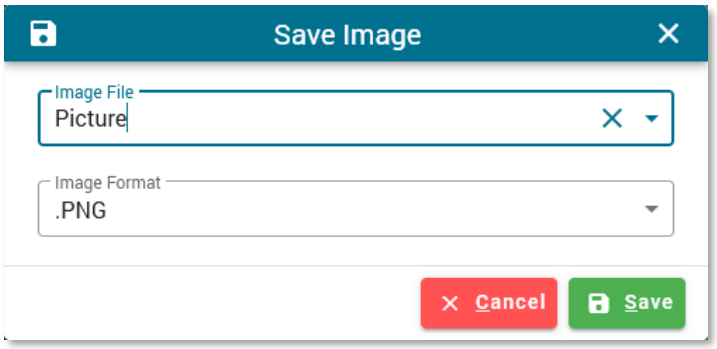


Figure 15: Save Image Dialog

In this dialog the user can enter a name for an image to save to the flash memory.

When entering a file name, existing files with a similar name are shown in the text field.

Select .PNG, .JPG, .BMP or .WEBP as image file format.

Cancel closes the dialog.

Save saves the current image.

If the file exists, the user is asked whether to replace the file.

6.1.4 Module Setup

The configuration is split into six separate settings: Camera, Processing Step 1 to 3, I/O and OPC UA. Each setting is managed by a single configuration file. Therefore, each parameter group can be changed without modifying other settings. The current selected file is shown in each group.

The six settings are managed together in a **Recipe**. This allows switching and selecting of all separate settings in a single step.

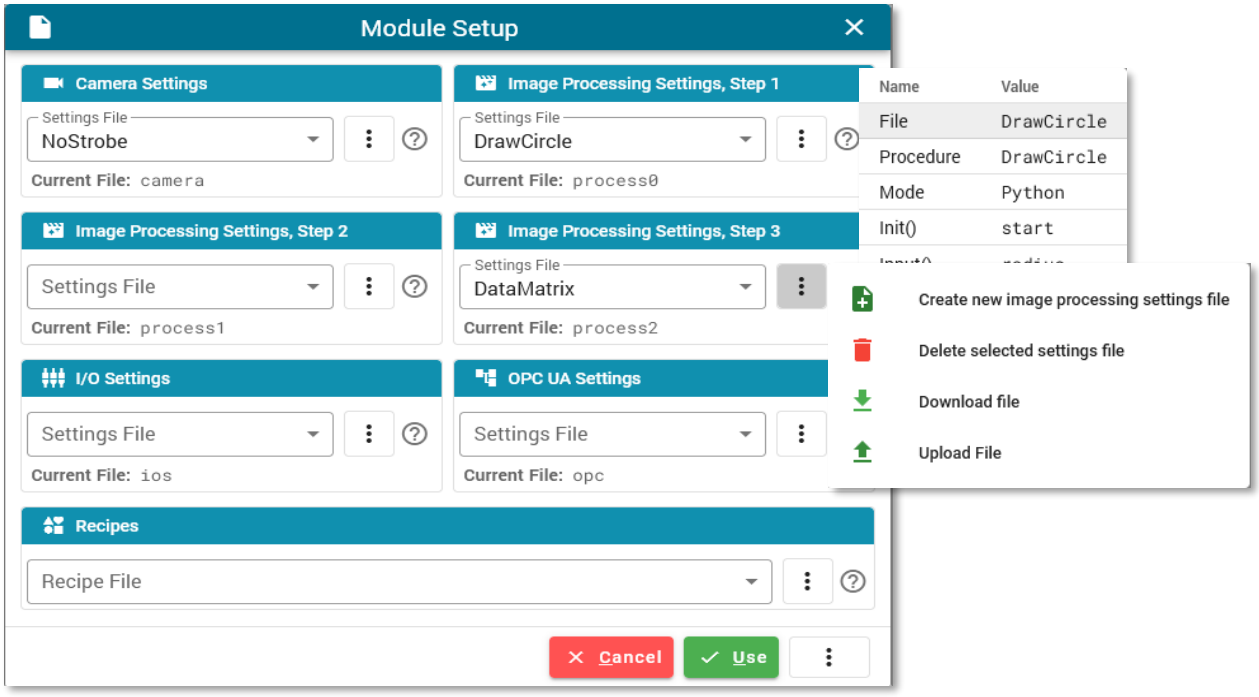


Figure 16: Module Setup Dialog

Click the Settings File input box inside a group to select a different stored configuration.

⋮ unfolds the following options for each group:

creates a new setting file based on the current values of the group.
If the file already exists, the user is asked whether to replace the file.

deletes the selected file after a confirmation dialog.

loads the group configuration as file to the PC.

selects a configuration file from the PC and sends it to the configuration.
It is also possible to drag and drop a file to a group box.

gets the six settings files from selected recipe.

clears selected settings files.

opens a pop-up window showing information about the selected settings file or the current settings if no file is selected.

Cancel closes the dialog and rejects the changes.

Use accepts the changes and closes the dialog.

⋮ unfolds the following options:

Load reloads the settings from flash memory reverting all changes.

Save writes the current settings to flash memory.

6.1.5 Configuration

This dialog manages the program settings. It has numerous tabs – one for each setting module and an additional General setting. A further Jupyter tab is available if JupyterLab is installed.

Cancel closes the dialog and rejects the changes.

Use accepts the changes and closes the dialog.

⋮ unfolds the following options:

Load reloads the settings from flash memory reverting all changes.

Save writes the current settings to flash memory.

6.1.5.1 General Tab

This tab contains common program settings that are independent from any module.

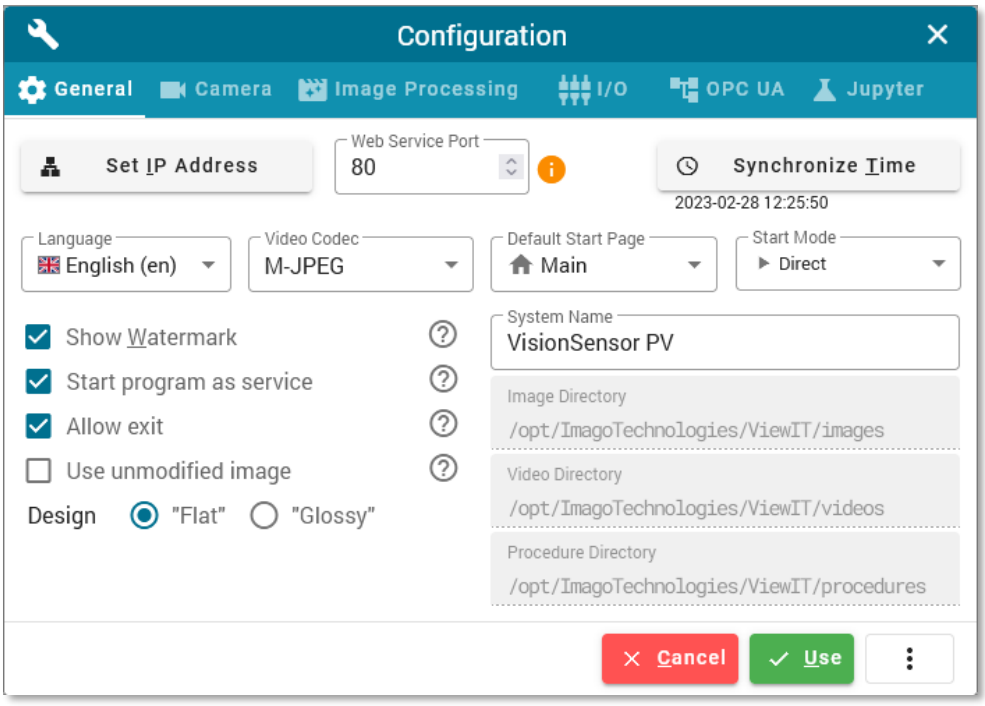


Figure 17: Configuration General Tab

Set IP Address opens a new dialog for changing the current IP address of the device (see 6.1.5.2).

Synchronize Time sets the internal clock of the Vision Device to the time provided by the browser.

Web Service Port	Port for the web browser, default is 80
Language	Application language; English or German
Default Start Page	Sets the web page which is opened when the IP address in the web browser is
Start Mode	Defines the behavior when opening the web page ▶ Direct: Open the web page without any question ✓ Acknowledge: Before entering the page, a dialog must be confirmed. *** Password User: Enter user password to open page. *** Password Admin: Enter administrator password to open page.
Show Watermark	Show ViewIT logo in background of main window
Start program as Service	If enabled, ViewIT is automatically started when powering on or rebooting the Vision Cam.
Allow exit	If disabled, only administrator can quit the application.
Use unmodified image	If enabled, acquired images are stored without overlay and processing output into the ring buffer.
Flat / Glossy design	Change appearance of title bars and buttons
System Name	Name of system, useful when multiple devices using ViewIT are available
Image Directory	Folder for storing images
Video Directory	Folder for storing videos
Procedure Directory	Folder for storing image processing procedures and data Place your HDevelop files (extension `.hdpl`), C++ libraries (`lib???.so`) or Python scripts (extension `.py`) in this folder on the flash card.
I/O Procedure Directory	Folder for storing I/O processing procedures

6.1.5.2 IP Address Setup

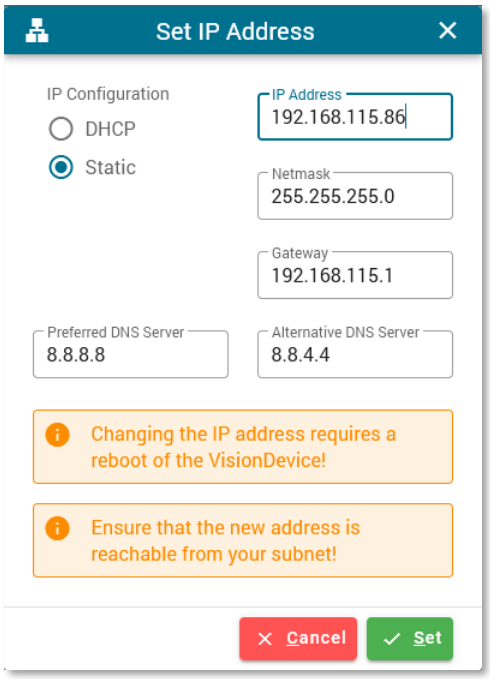


Figure 18: IP Setup Dialog

The user can switch between a static IP address and a dynamic address obtained by a DHCP server.

If Static IP Configuration is selected a Netmask and the addresses of Gateway and DNS Servers are available.

Changing the IP address requires a reboot of the camera. Ensure that the new address is reachable from the subnet of the computer where the web browser is running.

6.1.5.3 Camera Tab

This tab contains settings for image acquisition.

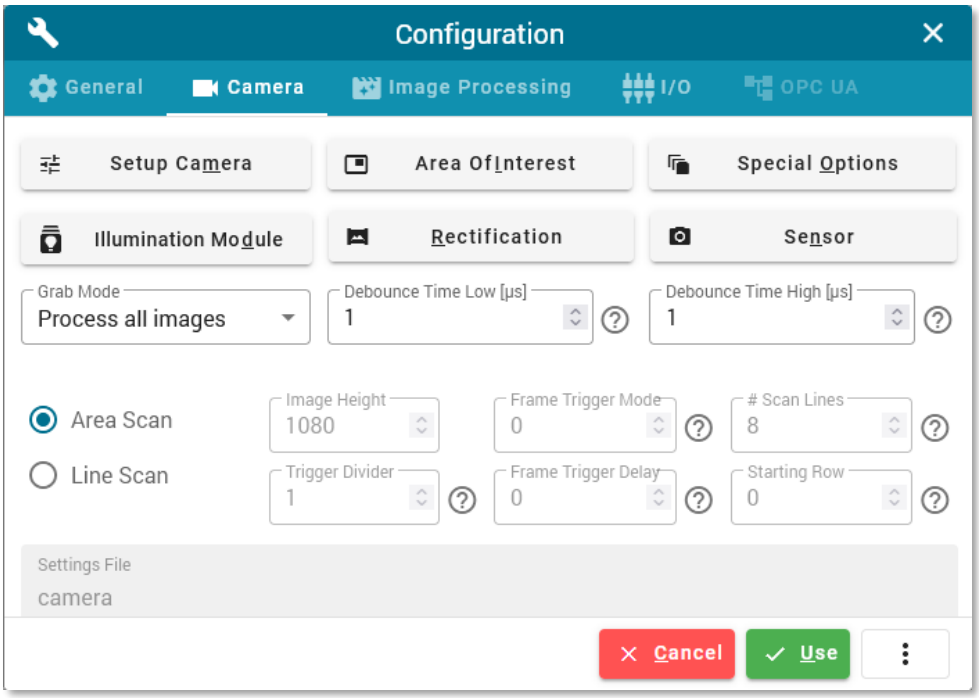


Figure 19: Configuration, Camera Tab

Setup Camera opens the **Camera Setup** dialog (see section 6.1.5.3.1).

Area Of Interest opens the **Area Of Interest** dialog (see section 6.1.5.2.2).

Special Options opens a dialog that contains sensor dependent settings like using a test pattern or flipping the acquired image.

Illumination Module opens a dialog that allows configuration of a connected illumination unit (see section 6.1.5.3.4).

Rectification opens a dialog for image rectification settings and calibration (see section 6.1.5.3.5).

Sensor opens a dialog that allows a selection between physical and virtual sensor (see section 6.1.5.3.6).

Grab Mode selects between grabbing all images (recommended for line scan sensor) and processing all images.
Debounce Time Low / High sets debounce time for digital input signals in µs.

On some devices a change between Line Scan and Area Scan mode is possible.

In Line Scan mode some changes are possible:

- Image Height: complete height of the image
- Frame Trigger Mode: Sets the frame trigger mode, used for starting frames in line scan mode.
  - 0: Do not user frame trigger, frames are captured continuously. (Default)
  - 1: Frames are software triggered by using the special option “Frame Trigger”.
  - 2: Use rising edge on input1.
  - 3: Use falling edge on input1.
- #Scan Lines: Number of simultaneously grabbed lines
- Trigger Divider: In hardware triggered mode, the trigger signal can be divided to reduce the frequency of the sensor. The divider counts trigger events based on the “TriggerLine” setting and generates a sensor trigger for every n-th event.
  - Minimum value: 1 (Default)
  - Maximum value: 65536
- Frame Trigger Delay: Frame start delay after receiving the frame trigger counted in multiple of the sensor scan height.
  - Minimum value: 0 (Default)
  - Maximum value: 65535

6.1.5.3.1 Setup Camera

This dialog manages camera settings like exposure time, gain, white balancing, and strobe parameters. When this dialog is opened in **Live View** with active image acquisition all parameter changes are immediately applied to the live image.

Cancel closes the dialog and rejects the changes.

Use accepts the changes and closes the dialog.

⋮ unfolds the following options:

Load reloads the settings from flash memory reverting all changes.

Save writes the current settings to flash memory.

Focus performs an automatic search of the focus. The area for detection of image sharpness can be Full (whole image) (⛶), Center (20 % of image width) (⊙) or Spot (10 %) (⊙).
(Only available if an adjustable lens is mounted.)

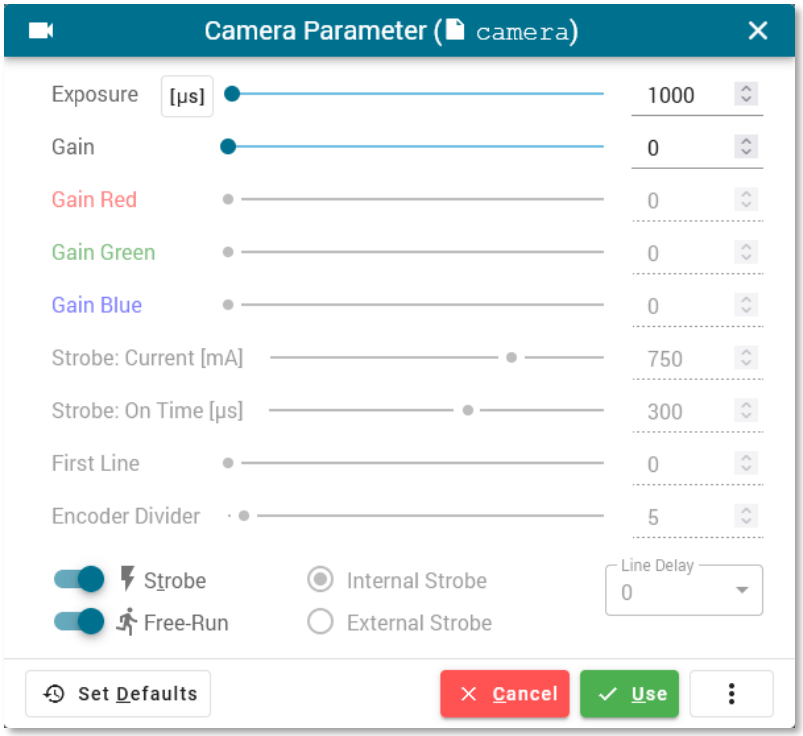


Figure 20: Camera Parameter Dialog

Some parameters are only available for special camera types, e. g. Gain Red exists only for a color camera.

For a better handling of the large range of exposure values the user can switch between milliseconds [ms] and microseconds [µs].

Set Defaults Sets the parameters to the sensor default values.

6.1.5.3.2 Area Of Interest Dialog

Figure 21: Area of Interest Dialog

This dialog enables setting of an area of interest. The camera will grab images of reduced size.

The region is defined by the white rectangle with green corners.

You can grab each corner with the mouse and drag it. Press the mouse button inside the rectangle to move the area without changing its size. The mouse pointer will change its shape to indicate the mode.

Use the green range sliders to adjust the size. The blue sliders will move the position.

Maximize resets the area to the default maximum size.

The checkbox switches between showing the selected area maximized in View Live ( ) or showing the area inside the full-size image surround by gray color ( ).

Cancel closes the dialog and rejects the changes.

Use accepts the changes and closes the dialog.

Size ( ) and coordinates of top left ( ) and bottom right ( ) corners are indicated.

Due to restrictions of the sensor the coordinate values are clipped to a multiple of 8 after accepting the settings.

6.1.5.3.3 Special Options Dialog

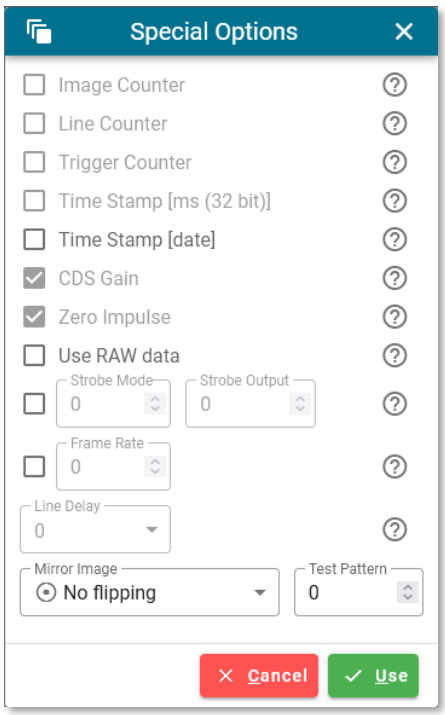


Figure 22: Special Options Dialog

This dialog contains sensor specific features which in some cases are only available for a single type.

Image Counter inserts a frame counter into the first pixels of a grabbed image.

Line Counter inserts trigger and sensor readout counters at the beginning of each line.

Trigger Counter enables or disables insertion of a trigger counter into grabbed image.

Time Stamp inserts the grabbed time as 32-bit value into the image.

CDS Gain enables analog gain value for the CDS pixel stage for a “Dragster” line scan sensor.

Zero Impulse uses zero impulse from encoder for image start of a line sensor.

Use RAW data ignores Bayer filter of the sensor and returns only monochrome data.

Time Stamp [date] enables or disables insertion of a time stamp into grabbed image. The first pixels of the image are set to values given in the table at the right side.

For the year 2023 the first pixel will have a gray value of 7, the second pixel 231 (7 * 256 + 231).

Pixels	Meaning	Range
YY	Year	1900 … 2100
M	Month	1 … 12
D	Day	1 … 31
H	Hour	0 … 23
M	Minute	0 … 59
S	Second	0 … 59
mm	Millisecond	0 … 999
uu	Microsecond	0 … 999
tttttttt	Timestamp	Microseconds

Strobe Mode sets operating mode of the integrated LED unit and the exposure output signal.

Mode	Integrated LED	Digital Output OUT
0	Enabled	Controlled by digital output
1	Off	Sensor exposure signal
2	Off	Controlled by digital output
3	Enabled	Sensor exposure signal

Frame Rate limits the sensor frame rate in free run mode to the desired value in Hertz. The actual frame rate may be lower if the exposure time setting is too high, or if the sensor’s limit is exceeded. The lowest achievable frame rate is 16 Hz for line scan mode, else 2 Hz.

Line Delay selects readout delay for one of the sensor lines of a dual line sensor. This setting can be used to compensate the physical displacement of both lines depending on the transport direction used in the application.

Mirror Image can flip the image horizontally, vertically or in both directions.

Test Pattern activates a sensor internal pattern. The count and style of patterns depends on the type of sensor.

6.1.5.3.4 Universal Illumination Module

If a **Universal Illumination Module** is connected the sensor can grab a sequence of up to four subsequent frames, each with different illumination settings.

Cancel closes the dialog and rejects the changes.

Use accepts the changes and closes the dialog.

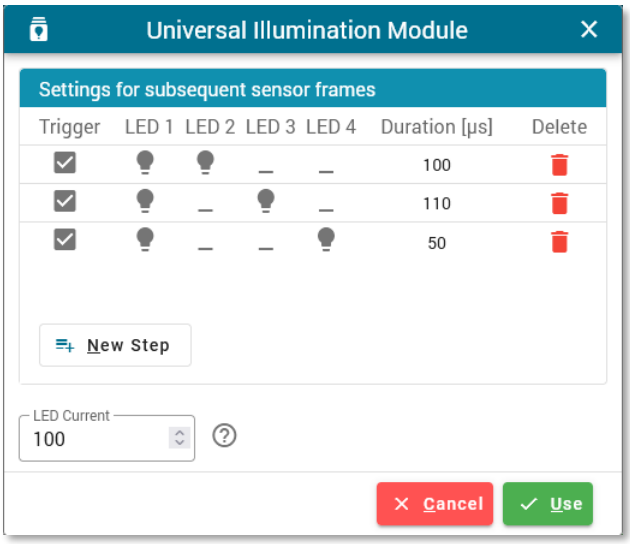


Figure 23: Universal Illumination Module Dialog

- Trigger
  Trigger mode for the selected sequencer step. This setting is ignored in free run mode.
  - Off ☐
    Disable sensor trigger for this step and acquire image when the sensor is ready.
  - On ☑
    Enable sensor trigger for this step (default).
- LED 1 … LED 4
  Active illumination elements
- Duration [µs]
  Length of time in which the LED units are switched on.
- 🗑
  Deletes the current step.

New Step creates a new line of settings. After the last step the sequence starts again.

The value LED Current sets the current in percent for adjustable LED units.

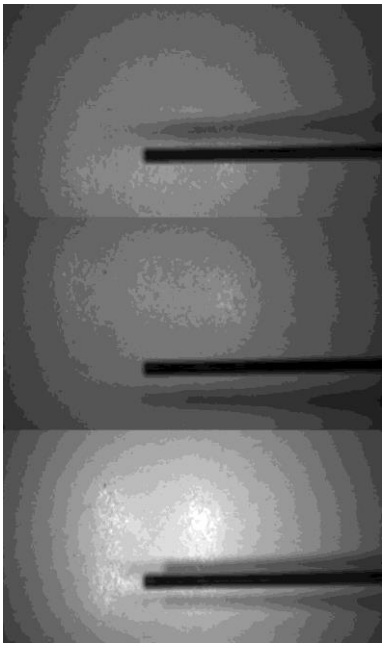


Figure 24: Combined image

You can use the Line Scan mode to create a single image consisting of a combination of different illumination modules.

If you have created a configuration with three steps like in the figure above, go to the camera configuration tab, select an Image Height of 3000 and set number of Scan Lines to 1000. The adjacent image gives a good impression of the different positions of the shadows caused by varying LED groups.

6.1.5.3.5 Rectification Dialog

Rectification is used to correct errors of the optical system like lens distortion and correction of perspective. Two types of rectification algorithms are available: OpenCV and HALCON. A valid license is necessary for using the HALCON algorithm. If Use Camera Rectification is enabled, the options are available.

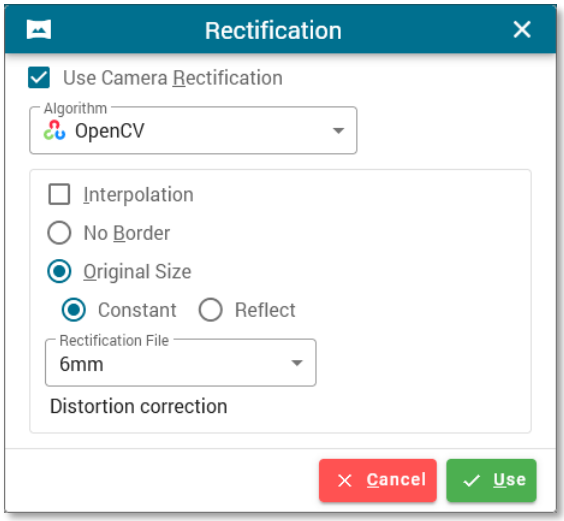


Figure 25: Rectification Dialog (OpenCV)

In **OpenCV mode** following options exists:

Interpolation creates better image quality at the expense of speed.

Due to the rectification the resulting image is no longer a perfect rectangle with right angles. Around the valid image data are undefined areas to form the rectangular shape.

No Border crops the image to a rectangle without undefined areas.

Original Size shows the full image with undefined areas painted in black (Constant) or by mirroring the valid image data (Reflect).

Finally select a previously saved Reflection File which can contain a describing comment that is show below.

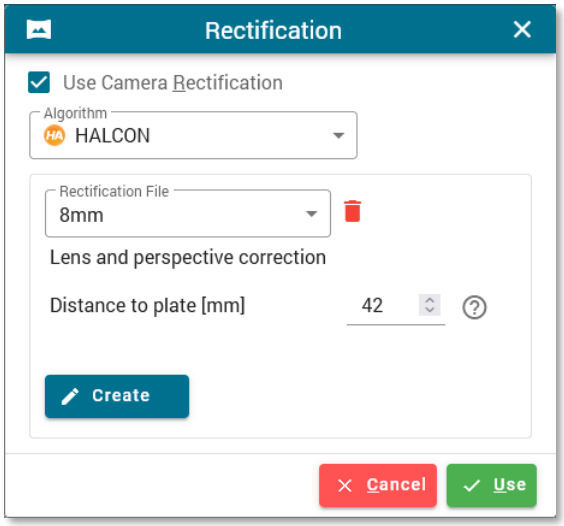


Figure 26: Rectification Dialog (HALCON)

In **HALCON mode** the Reflection File can be selected. Below the comment created at calibration is shown.

Deletes the currently selected file.

Distance to plate sets the distance from working point to calibration plate in mm. The thickness of the plate is already considered when creating the rectification file.

Create opens the **Create new rectification** dialog.

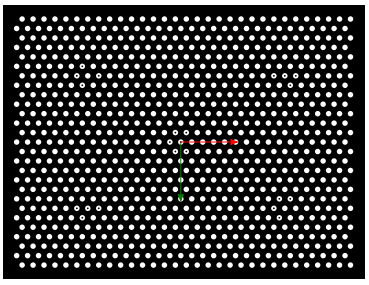


Figure 27: Calibration Plate

A calibration plate with hexagonally arranged marks is necessary for the calibration process. A PostScript file can be created via the HALCON operator `create_caltab()`.

If you have created your own calibration plate, copy the `.cpd` file to the folder `/opt/ImagoTechnologies/ViewIT/rectifications/calplates/` on the file system.

The **Create new rectification** dialog guides the user in four steps to generate a calibration setting. Each step has the following buttons:

← Back moves to the previous step.

→ Next goes to the next step.

× Cancel closes the dialog.

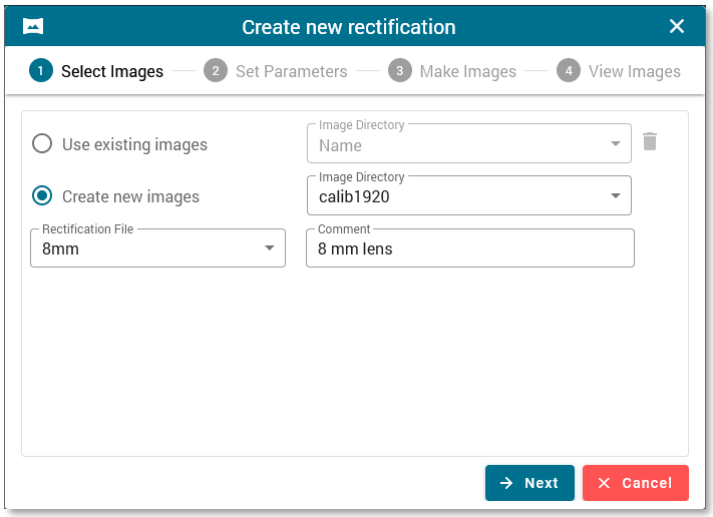


Figure 28: Create Rectification (Step 1)

The first step decides which images are used for the calibration.

Use existing images will read previously saved images from an Image Directory.

Deletes the currently selected directory.

Create new images will use live images that are later saved into a new directory.

A name for the Rectification File must be specified, while the Comment is optional.

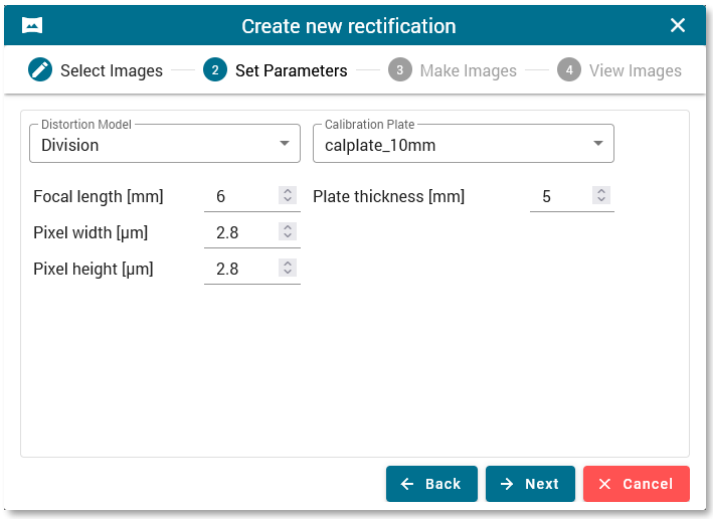


Figure 29: Create Rectification (Step 2)

In step 2 the parameters are set.

Distortion Model selects between Division and Polynomial. Using the division model is generally the best choice, while a polynomial more accurate.

Calibration Plate picks a .cpd file with the description of a HALCON calibration plate (see Figure 27).

Set Focal length of the lens in mm and Width and Height of a single pixel from the sensor in µm. Get the pixel size of your sensor from this web site: api.imago.tech/FGCamera/_s_e_n_s_o_r_s.html The e2v sensor used in a Vision Sensor PV3, for example, has a pixel size of 2.8 µm x 2.8 µm.

Plate thickness is the width of the used calibration plate in mm.

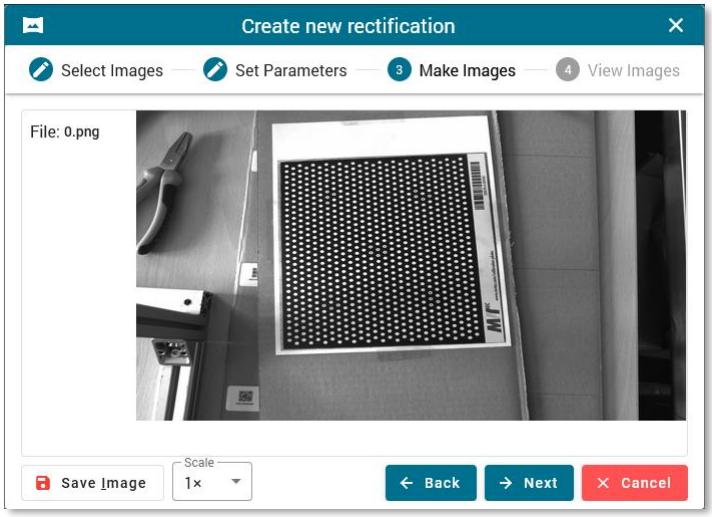


Figure 30: Create Rectification (Step 3)

This step is skipped if Use existing images was selected in step 1.

The live image from the camera is shown. Use Scale to zoom in and out.

Save Image writes the image with ascending number to the folder.

In order to achieve an accurate calibration result, you should pay special attention to the acquisition of your calibration images. See Figure 31 for an exemplary setup.

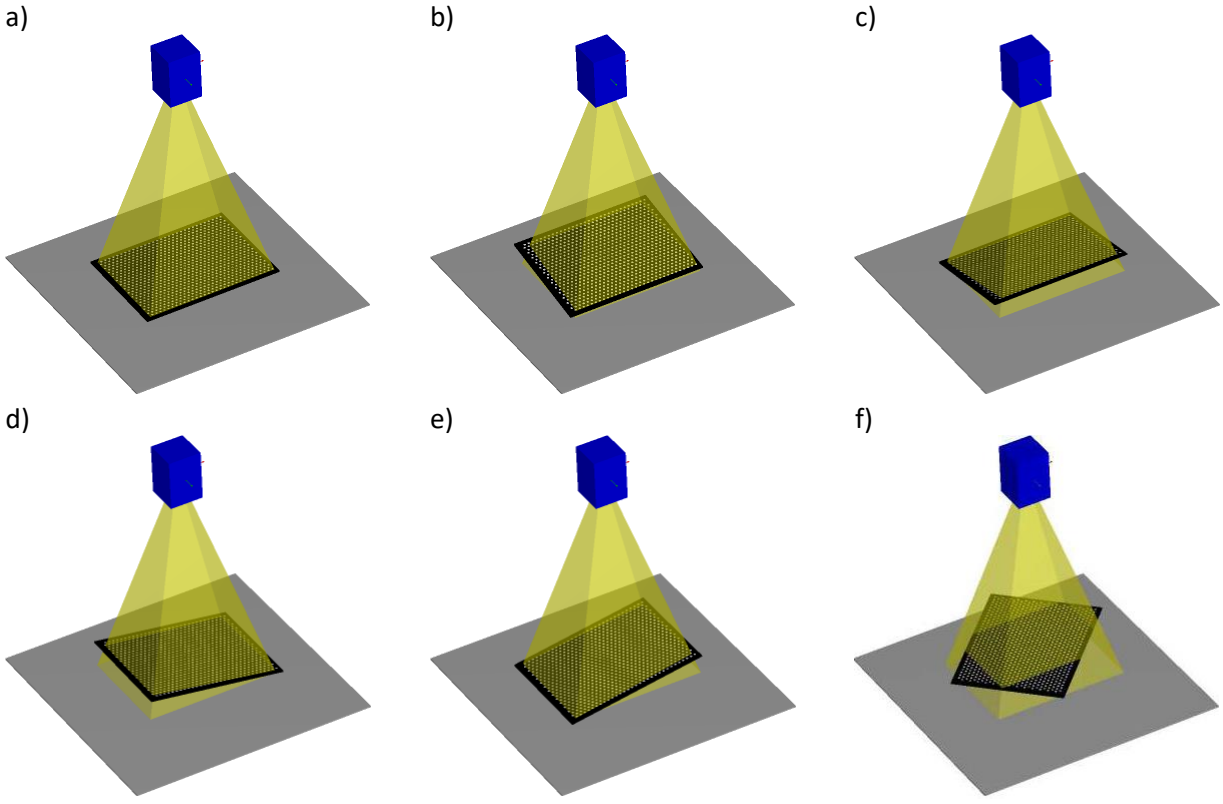


Figure 31: Acquisition of calibration images

a) For the first image the plate is placed in the measurement plane. This will be the reference image. Place the calibration plate such that it lies on top of the measurement plane. If this is not possible, place the calibration plate in a position parallel to the measurement plane and "move" the coordinate system by adapting the parameter Plate Thickness.

b – e) Plate is tilted in different directions.

f) Plate is placed parallel to the measurement plane (with a rotation around the z axis).

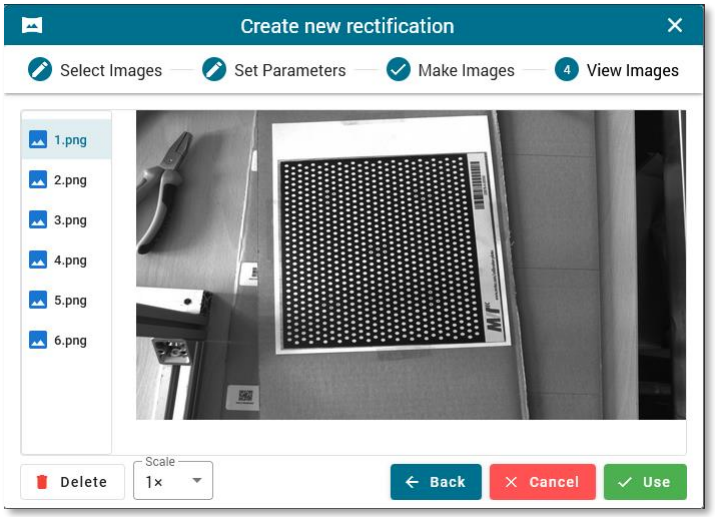


Figure 32: Create Rectification (Step 4)

In the last step you can inspect the saved images.

Delete deletes the selected image.

✓ Use creates a new rectification set and closes the dialog.
Now you can select the new rectification file in the **Rectification Dialog**.

6.1.5.3.6 Sensor Dialog

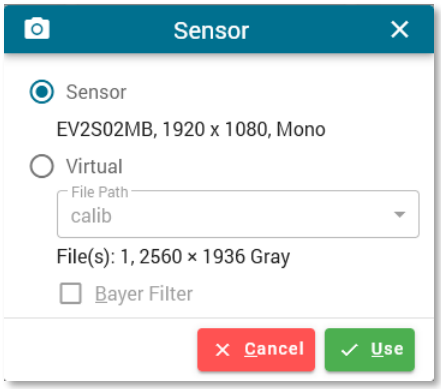


Figure 33: Sensor Dialog

Sensor selects the physically present sensor. Name and properties are shown below.

Virtual uses `.BMP` files in a directory as the data source. This allows the algorithms to be used repeatedly on the same image data.

File Path sets this directory. The number of files and their image format is also displayed. The virtual camera may have a different resolution and color format than the physical sensor.

All folders in `/opt/ImagoTechnologies/ViewIT/images` containing an image sequence are displayed in this list box.

If a color sensor is using a Bayer pattern, images are nevertheless created in monochrome format. The algorithm cannot distinguish these images from true gray images. After enabling Bayer Filter these images are converted to color.

6.1.5.4 Image Processing Tab

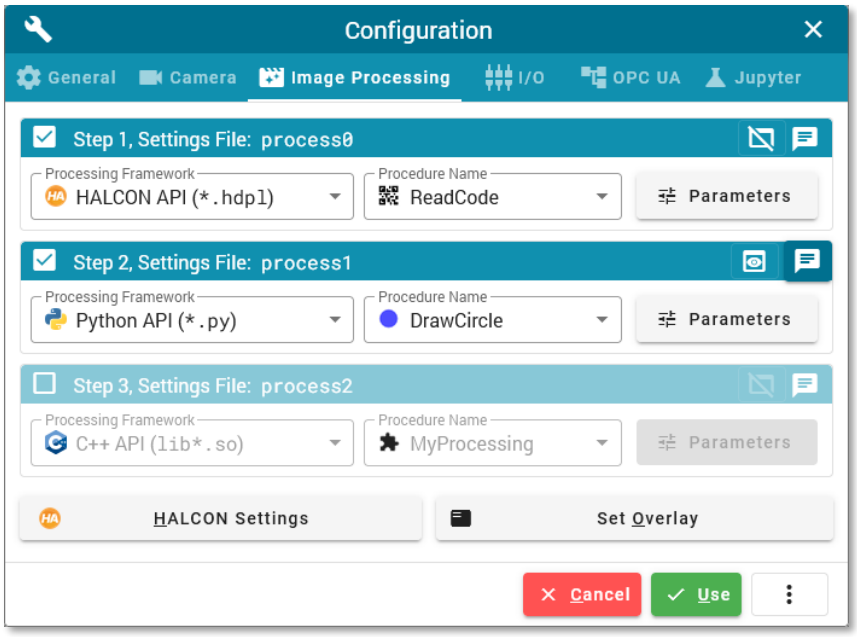


Figure 34: Configuration, Image Processing Tab

This tab configures the image processing features.

When using the HALCON API algorithms are written as a HALCON procedure and executed by **HDevEngine**. This means that the user can easily update the machine vision part of the application by replacing or adding individual HDevelop files.

It is also possible to use procedures created by the General C++ API or Python API. Development of these procedures is described in a separate document.

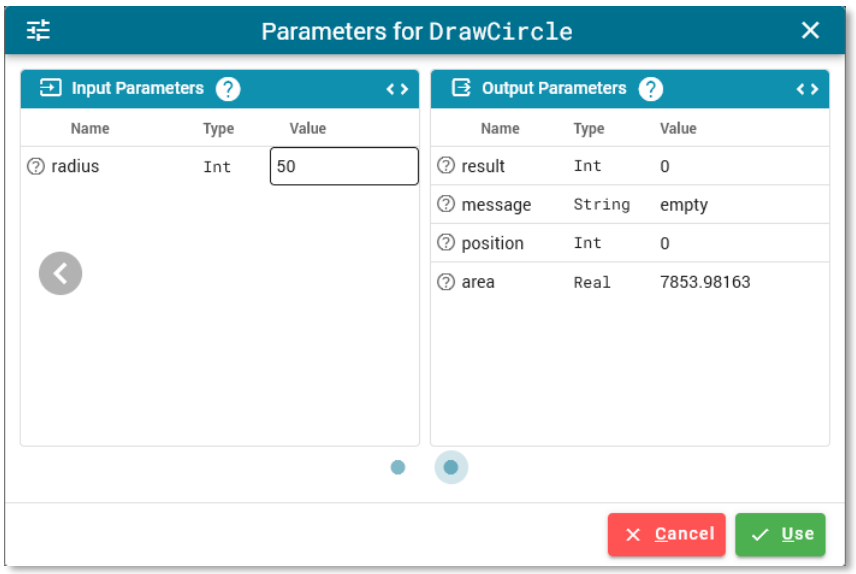
Up to three consecutive image processing steps are selectable. Every step has the following features:

- Switch on (☑) or off (☐) the processing step.
- Processing Framework
  Select between HALCON API, General C++ API and Python API image processing engine.
- Procedure Name
  Select a procedure from the list of automatically detected available procedures.
  Hover the mouse over the symbol to get information about the selected procedure. If Python Framework is selected, the source code of the procedure is shown by pressing the symbol.
- Parameters
  Open **Parameters Dialog** (6.1.5.4.1).
- Use web interface from this step (☑) or not (☐).

HALCON Settings opens a dialog that handles parameters for HALCON API (0).

Set Overlay opens a dialog that handles drawing of processing data into the grabbed image (6.1.5.4.3).

6.1.5.4.1 Parameters Dialog



Up to three different types of parameters are available, but only two types are shown at the same time.

You can use the ● markers or the < and > symbols to flip through the parameter pages.

Use <> to maximize a parameter group.

Figure 35: Parameters Dialog

- Init Parameters
  The optional initialization parameters are automatically determined from the selected procedure.
  Hovering the ⓘ symbol gives more information about the parameter.
  If the procedure has no initialization parameters, the group is not shown.
  Click into the frame around a Value to edit the parameter.
- Input Parameters
  Also, the input parameters are automatically determined from the selected procedure. The procedure is called with the given parameter values.
- Output Parameters
  These parameters are also automatically determined. The values from the last image processing are shown.

An input parameter with name `DigitalInput`*N* will automatically use the current value of the digital input *N* when the image processing procedure is called.

An output parameter with name `DigitalOutput`*N* sets the digital output *N* after executing the procedure.

Input parameters with names `ViewIT.Camera`, `ViewIT.Communication`, `ViewIT.Grabbing`, `ViewIT.Images`, `ViewIT.Processing`, `ViewIT.Status` or `ViewIT.Data` get current settings as JSON string.

6.1.5.4.2 Halcon Settings Dialog

This dialog is only available if a valid HALCON license is present.

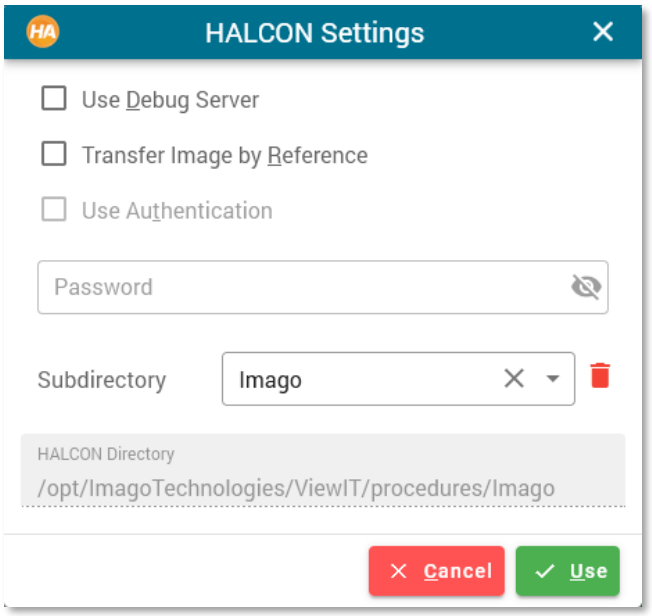


Figure 36: HALCON Settings Dialog

- Use Debug Server
  If the debug server is enabled, you can use HDevelop for attaching to the imaging process of ViewIT to step through the code and watch data. When using default settings, the server waits on port 57786 and image processing runs normally until HDevelop is connected.
- Transfer Image by Reference
  Transferring images by reference is faster but can cause problems if a HALCON procedure uses images from different process cycles. (Only for monochrome cameras.)
- Use Authentication
  When connecting to HDevelop a password is highly recommended.
  (Only available if Use Debug Server is enabled.)
- Password
  Enter the same password into HDevelop to allow the connection.
  (Only available if Use Authentication is enabled.)
- Subdirectory
  Creates a new or selects an existing subdirectory for HALCON functions. Therefore, it is possible to use different libraries with individual `Init`() and `CleanUp`() functions.

6.1.5.4.3 Overlay Dialog

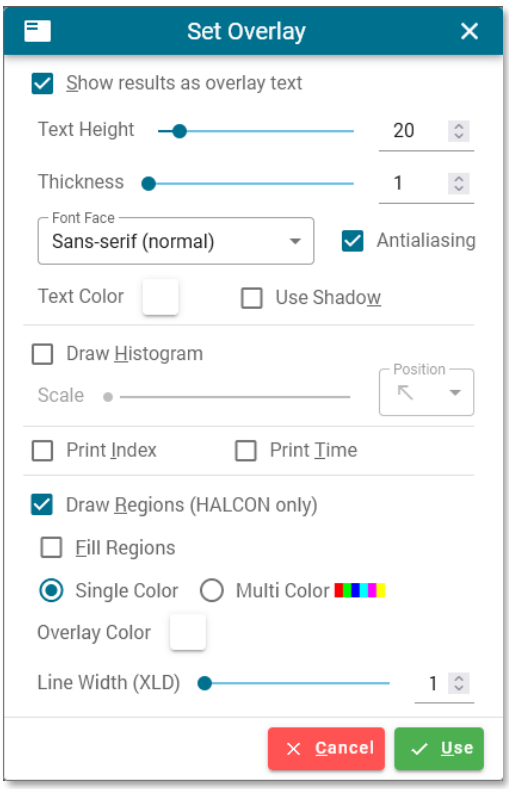


Figure 37: Set Overlay Dialog

- Show results as overlay
  Prints the results of the output parameters into grabbed image.
  - Text Height
    Size of text overlay
  - Thickness
    Line width of font
  - Font Face
    Font type
  - Antialiasing
    Smooths edges of text.
  - Text Color
    Press on the colored rectangle to open a color picker window.
  - Use Shadow
    Text is printed with a black shadow for better readability.
- Draw Histogram
  Draws distribution of colors or gray values into image.
  - Scale
    Sets size of histogram graph.
  - Position
    Position of graph in image, e. g. left top.

- Print Index prints the current image number in human readable form into the image.
- Print Time prints time stamp containing date and time like 2023-10-20 08:15:10,012 into the image.

The following options are only available for HALCON API:

- Draw Regions
  Draw regions returned from image processing into grabbed image.
- Fill Regions
  Switches between margin and filled style for region drawing.
- Single / Multi Color
  Sets the line color.
- Overlay Color
  Press on the colored rectangle to open a color picker window.
- Line Width (XLD)
  Sets line width when drawing region contours.

✕ Cancel closes the dialog and rejects the changes.

✓ Use accepts the changes and closes the dialog.

6.1.5.5 I/O Tab

This tab configures the communication features.

In the processing cycle previously to image processing a function from the optional communication (I/O) plug-in named `Pre()` is called. The purpose of this function is getting parameters and data from external interfaces like Ethernet sockets or databases. After image processing the output parameters are passed to the second function of the communication plug-in named `Post()`. This can handle the data, for example write a detected code over a network socket.

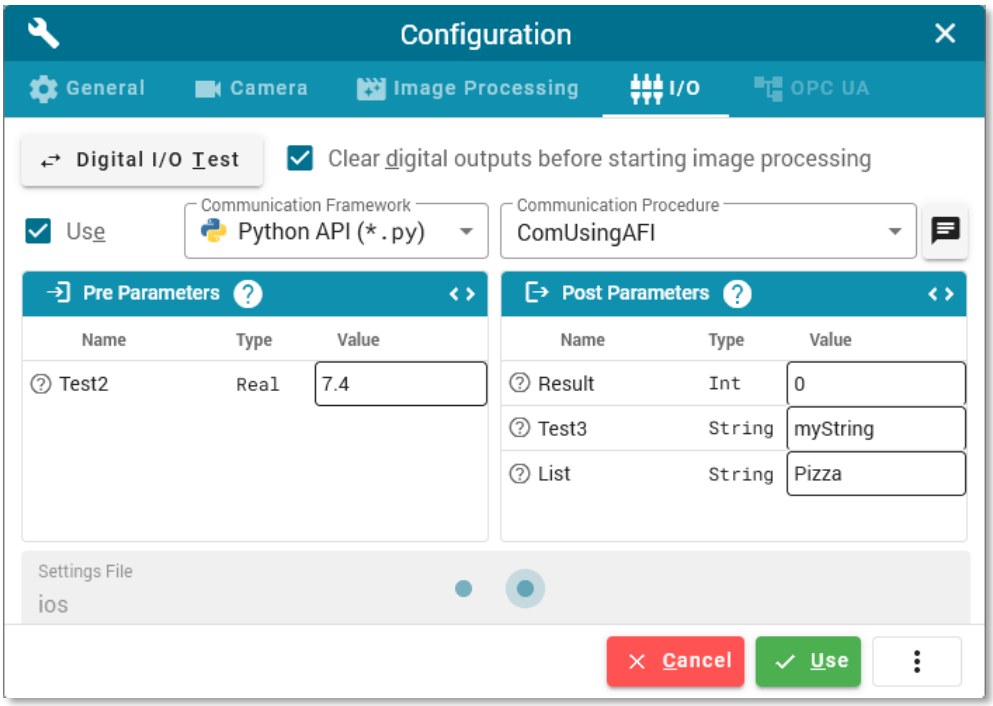


Figure 38: Configuration, I/O Tab

Digital I/O Test opens the **Digital I/O Test** dialog (see section 6.1.5.5.1).

The image processing procedure can set digital outputs. The values are retained until the processing procedure will change them. If the switch Clear digital outputs before starting image processing is enabled, all outputs are set to 0 before image processing starts. This is helpful when one output is used to indicate when the image processing has finished.

Using of the I/O functions is optional. Therefore, the plug-in can be disabled by the Use ☑ checkbox.

Communication Framework	Select between General C++ API and Python API communication engine.
Communication Procedure Name	Select a procedure from the list of automatically detected available procedures. Hover the mouse over the symbol to get information about the selected procedure. If Python Framework is selected, the source code of the procedure is shown by pressing the symbol.
Init Parameters	The optional initialization parameters are automatically determined from the selected procedure. Hovering the symbol gives more information about the parameter. If the procedure has no initialization parameters, the group is not shown. You can use the markers or the and symbols to flip through the parameter pages.
Pre Parameters	Also, the parameters for the Pre() function are automatically determined from the selected procedure. The procedure is called with the given parame-
Post Parameters	These parameters are also automatically determined.

Change a parameter value by clicking into the frame around the Value box.

Parameters with names `ViewIT.Camera`, `ViewIT.Communication`, `ViewIT.Grabbing`, `ViewIT.Images`, `ViewIT.Processing`, `ViewIT.Status` or `ViewIT.Data` (for all information) get current settings as JSON string.

Like image processing an input parameter with name `DigitalInput`*N* will automatically use the current value of the digital input *N* and an output parameter with name `DigitalOutput`*N* sets the digital output *N* after executing the procedure.

6.1.5.5.1 Digital I/O Test Dialog

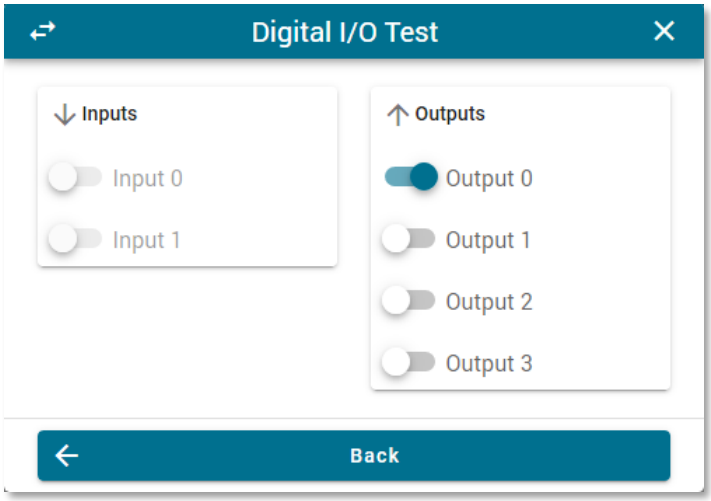


Figure 39: Digital I/O Test Dialog

The current state of all available inputs is indicated on the left side. Use the switches to enable / disable any output.

Back closes the dialog and disables all outputs.

6.1.5.6 OPC UA Tab

OPC Unified Architecture (OPC UA) is a machine-to-machine communication protocol. The inputs and outputs of the image processing and some other values are placed into the address space tree (see section 0).

The user can enable or disable the service by means of the Use OPC UA checkbox.

For authentication, an anonymous user or a combination of Username and Password is possible.

If the Use Authentication check box is disabled, the anonymous login is used.

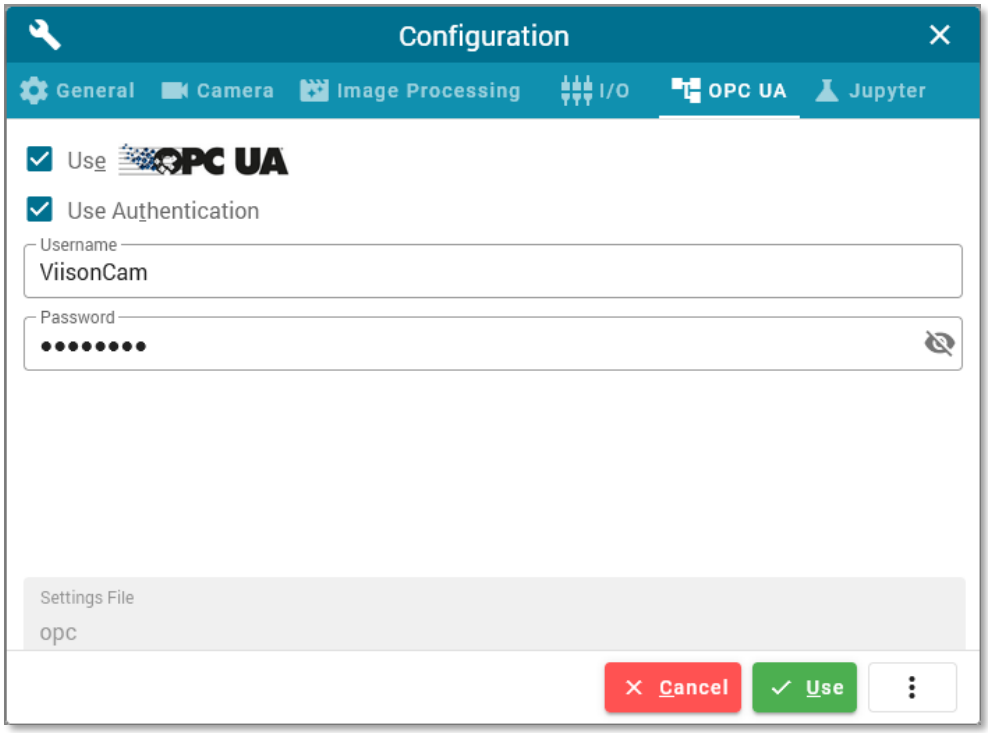


Figure 40: Configuration, OPC UA Tab

6.1.5.7 JupyterLab Tab

For more information about JupyterLab see section 0. This tab is only available if a JupyterLab server is installed.

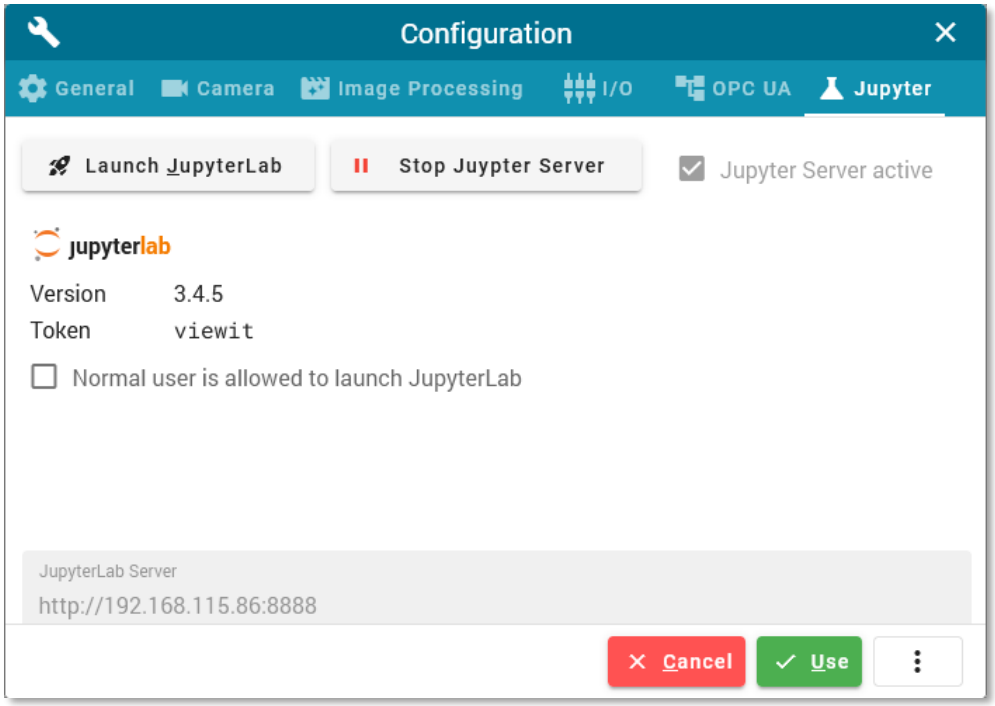


Figure 41: Configuration, JupyterLab Tab

Launch JupyterLab starts the JupyterLab server. This will take some seconds. Then a new browser tab is opened. Probably you have to allow opening pop-up windows in your web browser.
If the server is already running only the browser tab is switched.

Stop Jupyter Server shuts down a running JuypterLab server. All unsaved changes of open notebooks are lost!

By default, only an administrator is allowed to launch the JupyterLab server, but you can allow a normal user to launch the server.

You can open the web page of the JupyterLab server manually by entering the given URL. For authentication you need the shown Token.

6.1.6 Statistics

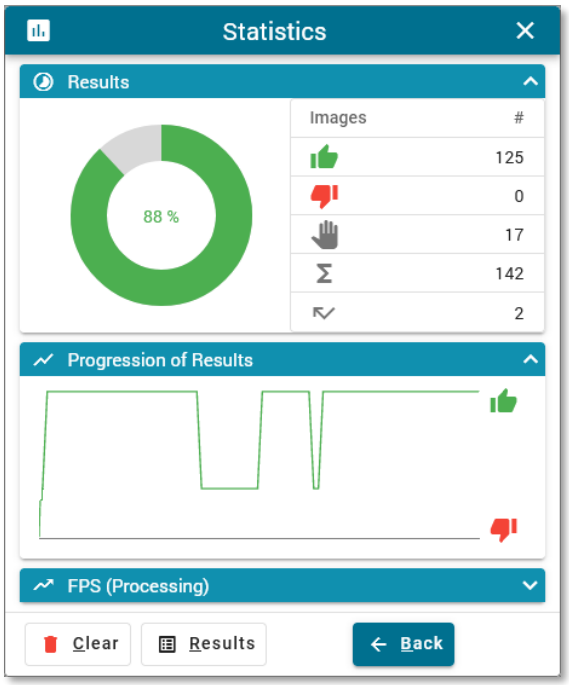


Figure 42: Statistics Dialog

This dialog shows some statistics of the image processing. You can collapse (⌃) and expand (⌄) the panels.

The table indicates the number of good (👍), bad (👎), not processed (✋), total (Σ) and missed (↙) of grabbed images. The pie chart ◔ visualizes the percentage of good images.

The line diagram 📈 shows the Progression of Results from the last frames.

A third graph 📈 shows the number of frames per second (FPS) achieved while image processing.

← Back closes the dialog.

Clear sets all values to 0.

Results opens the **Results** dialog which shows a list of the previous calculated output values from the image processing procedure.

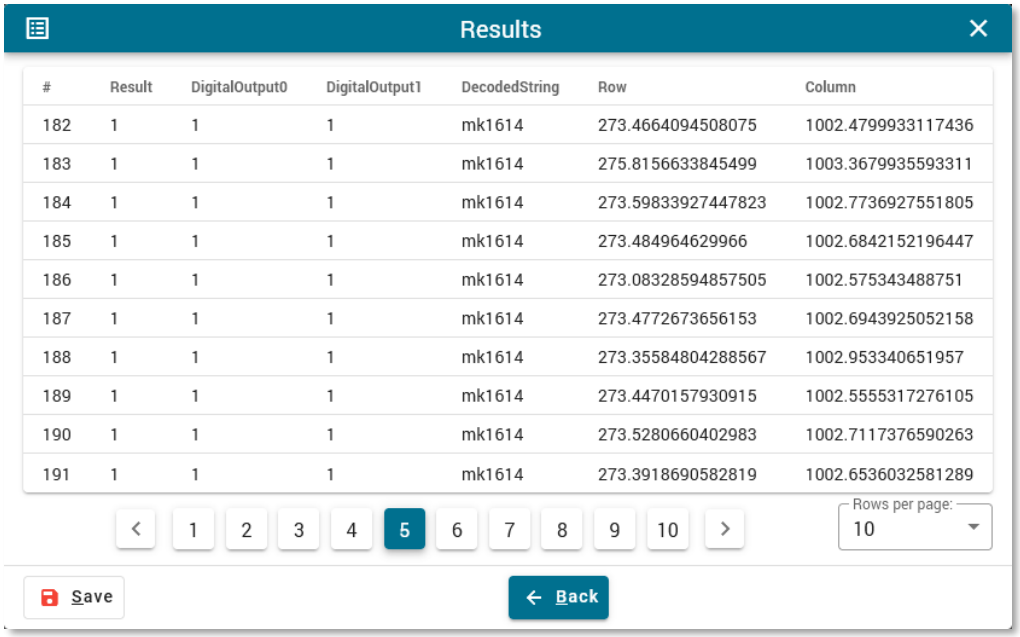


Figure 43: Results Dialog

Save opens a dialog which allows saving the data as `.htm`, `.csv` or `.tsv` file.

Refresh updates the values when the dialog has been opened from active Live View.

← Back closes the dialog.

6.1.7 View Buffer

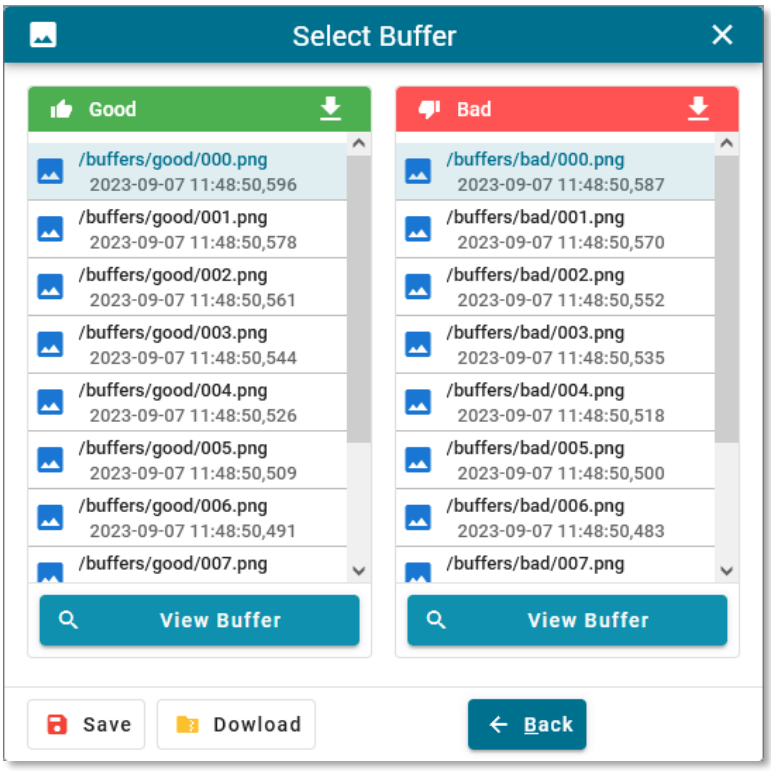


Figure 44: Select Buffer Dialog

Depending on the result of image processing a grabbed camera image is stored into the good or bad ring buffer.

Select a buffer from the good or bad list and press View Buffer to view the image (see section 6.1.7.1).

Save writes all buffers to the image folder. Existing images are overwritten.

Download puts all images of both buffers into a single `.ZIP` archive and downloads it.

downloads all good or bad images as single `.PNG` files.

Back closes the dialog.

6.1.7.1 Image Viewer Dialog

The **Image Viewer** dialog is similar to the Live View (6.1.3) and has the following options:

Processing calls the image processing procedure with the current image and shows the results in a small window.

If the checkbox is enabled the image is displayed again after calling the image processing procedure. This is useful if the processing has drawn some results into the image.

Save Image opens the Save Image dialog (see section 6.1.3.1) and permits saving of the image.

Back closes the dialog.

Scale changes the magnification factor of the displayed image.

shows the previous image from the buffer.

shows the next image from the buffer.

6.1.8 View Image

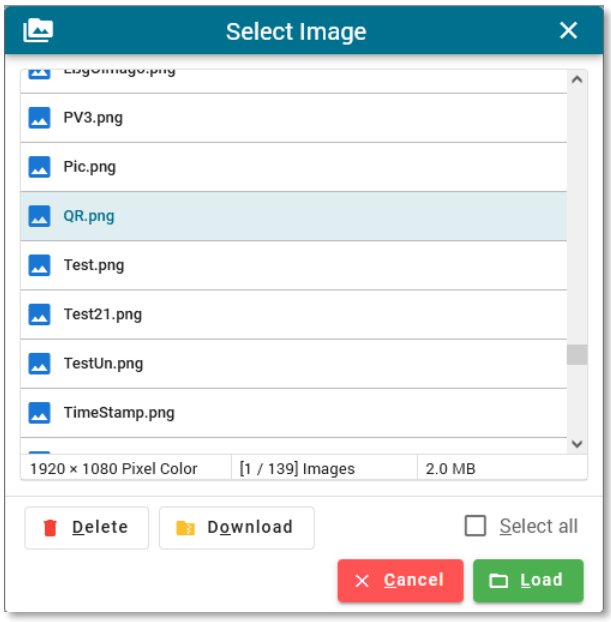


Figure 45: Load Image Dialog

Select an existing file from the list.

Cancel closes the dialog.

Load loads the image and opens the Image Viewer Dialog (see section 6.1.7.1).

Delete deletes the selected files after acknowledging.

Download downloads all selected images as a single `.ZIP` file.

When selecting a file, date of creation and size of the file are shown.

Use Ctrl key to select multiple files or Shift key to select a group of files.

6.1.9 Upload File

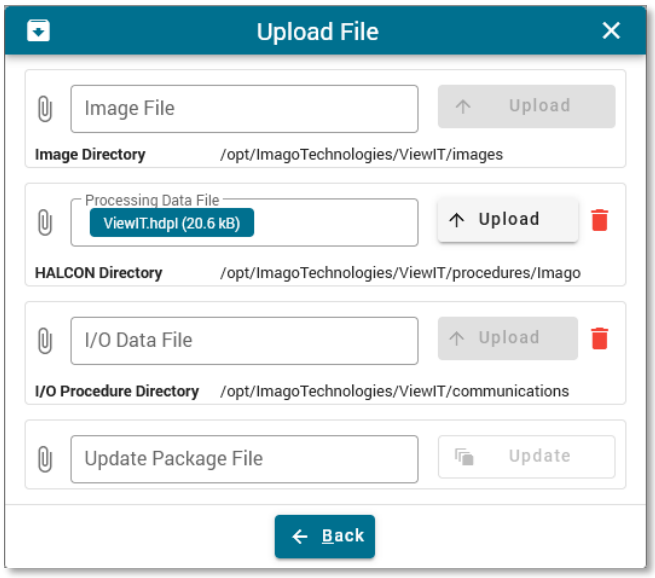


Figure 46: Upload File Dialog

This dialog allows transferring files to the Vision Cam.

Image Files are written to the Image directory, Processing Data Files to the image processing procedure path, and I/O Data Files to the communication procedure path. This can be HALCON procedure libraries, C++ libraries, Python scripts, and data like model definitions for pattern matching.

Select a file and press ↑ Upload to send a file. You can also drag a file from the Windows Explorer into a group.

opens a dialog that allows deleting a selected file.

← Back closes the dialog.

Update Packages can be Debian packages (`.deb`) like SDK updates or new versions of ViewIT (`.out`). After pressing Update the file is checked and a message containing file information appears. Now press ✓ Update to start the installation. After that, the application will be restarted.

6.1.11 AI.go

AI.go uses a pre-trained neural network for classifying objects of all kinds. A new model is created by assigning images to different classes like “dog” or “cat” and starting a light training process. Different models can be used and created. Model and image data is stored in separate working directories.

The AI.go dialog manages the data for the deep learning. It has two tabs – File Management and Training.

Back closes the dialog.

6.1.11.1 File Management Tab

This tab has three groups – Working Directories, Classes, and Images.

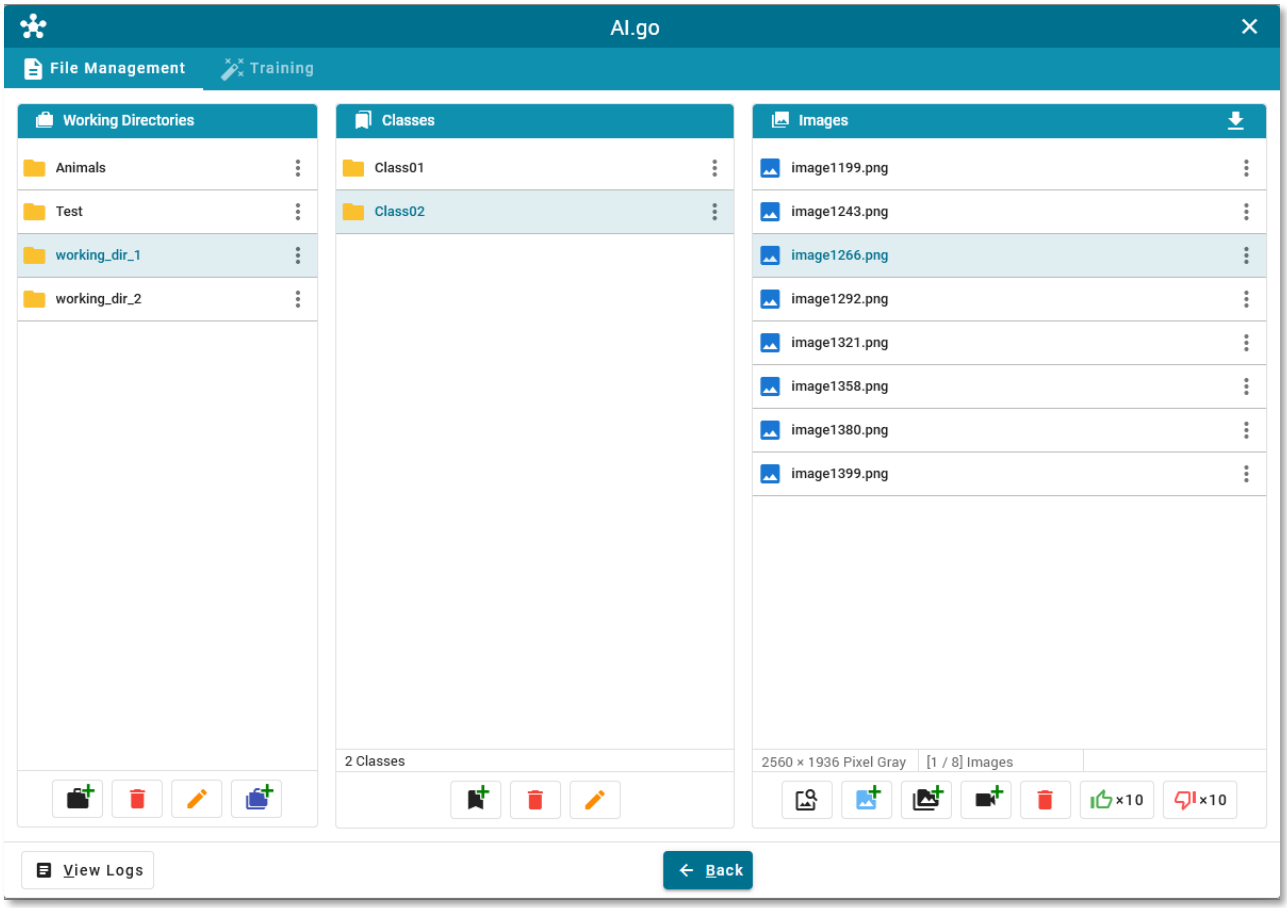


Figure 47: Vision AI.go File Management Tab

Click on the ⋮ symbol to the right of each entry to open a context menu.

View Logs opens a dialog that shows all log files which have been created while training and processing. Select a file to view its contents.

The data is structured as follows:

Working Directories

A working directory contains all files and folders which are used to train and work with a neural network.

creates a new workings directory.
The user is asked for a new name.

deletes the selected working directory and all its contents.

renames the selected working directory.

copies the contents of the selected working directory to a new one.

Each working directory can contain several Classes.

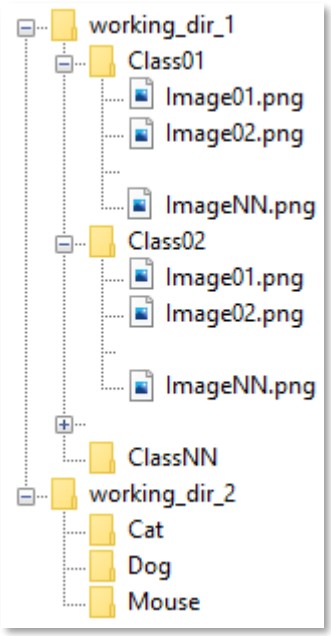


Figure 48: File Hierarchy Example

Classes

Selecting a working directory updates the list of included classes.

creates a new class folder.

deletes the selected class and its contents.

renames the selected class.

Each class can contain several Images.

Images

Selecting a class folder updates the list of included images.

Dragging an image file from the Windows Explorer and dropping to the image group copies the file to the class folder. All images should have the same size and format.

shows the selected image in the image viewer dialog.

opens the upload image dialog.

deletes the selected images. Use the shift key to select multiple images.

×10 copies the images from the good buffer to the class folder.
The “10” indicates the number of images in the buffer.

×10 copies the images from the bad buffer to the class folder.

opens the live view dialog which allows adding a grabbed camera image to the class folder.

6.1.11.1.1 Live View Dialog

Use the **Live View** dialog to add a grabbed image to the training data set.

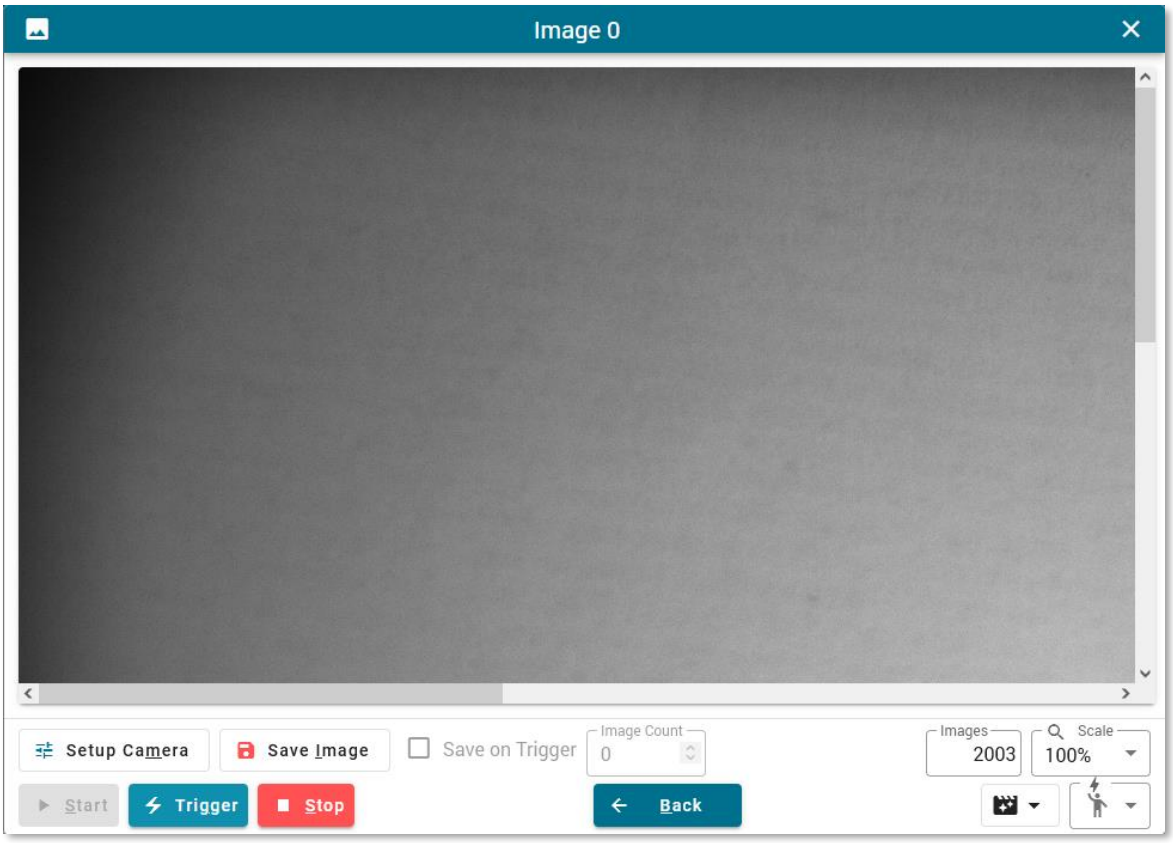


Figure 49: Live View Dialog

Setup Camera opens the Camera Setup dialog (see section 6.1.5.3.1).

Save Image saves the current image with the image number into the class folder.

Save on Trigger is available if Software or Triggered mode is enabled. In this mode each trigger signal saves an image with ascendant name until Image Count reaches 0.

Scale changes the magnification factor of the displayed camera image.

The action line contains the following controls:

Start enables live view.

Trigger manually generates a trigger signal. (Only available in Software mode).

Stop disables live view.

Back closes the dialog.

disables or enables the processing steps which are executed on each image.

Software , Triggered, Free-Run selects the trigger mode.

6.1.11.2 Training Tab

The **Training** tab is similar to the **File Management** tab. It has three groups – Working Directories, Models, and Information.

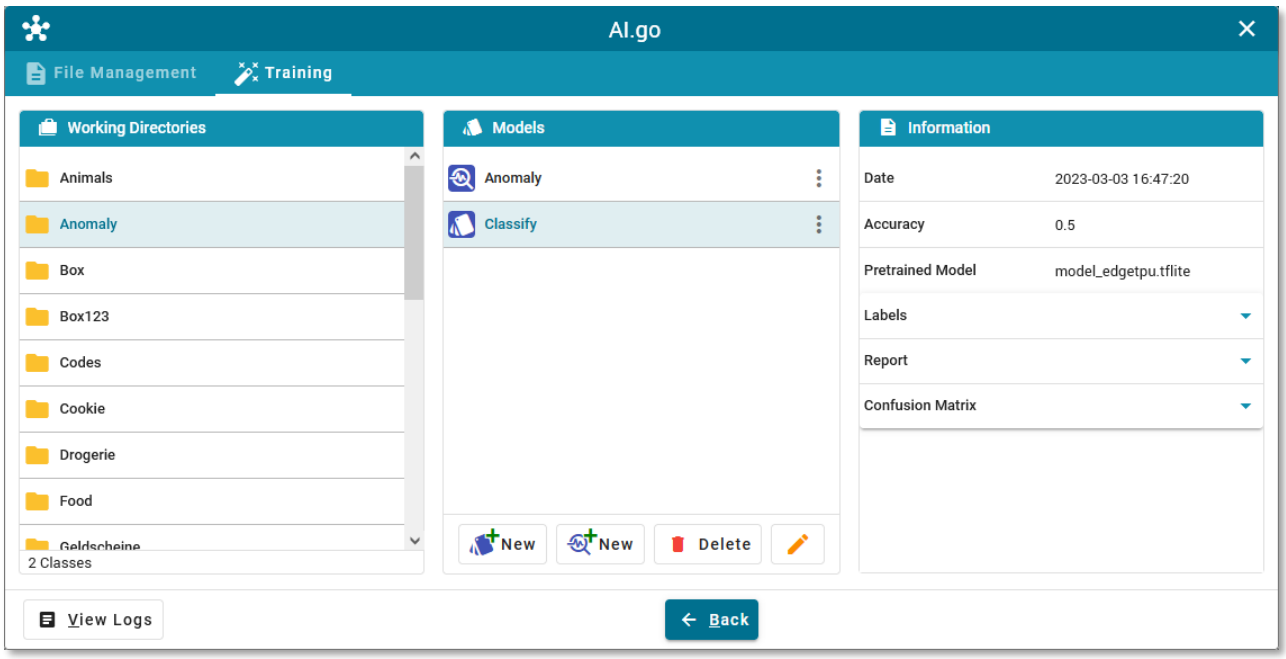


Figure 50: Vision AI.go Training Tab

The list of working directories is identical to the list in the File Management Tab (see 6.1.11.1). After selecting a working directory, the list of previously created model is updated.

Models

The type of model is indicated by (Classification) or (Anomaly). A click on the ⋮ symbol to the right of each entry opens a context menu.

New creates a new **Classification** model.
After choosing a name for the model, the training starts. Depending on the number of images and classes this can take some minutes. A bar indicates the progress.

New creates a new **Anomaly** model.
This model type is only available if a Class "good" has been created.

Delete deletes the selected model.

renames the selected model.

Information

This column shows information like creation Date and Labels. An anomaly model also contains a value for Threshold which indicates the boundary between good and bad.

6.1.11.3 Training and Inference

The following sequence describes how to create a new **Classification** model:

- Open AI.go dialog and go to File Management tab.
- Select in the Working Directory group to create a new working directory.
  - Enter a name like "Animals" in the dialog and press Save.
- Select in the Class group to create a new class.
  - Enter a name like "Cat" in the dialog and press Save.
- Select in the Class group to create another class.
  - Enter a name like "Dog" in the dialog and press Save.
- You can create more classes. Minimum number is 2, maximum 5.
- Now drag image files to the corresponding Images group.
  At least 8 images are required for each class.
  You can also open the Live View dialog by pressing .
  - Grab and save images according to the class (see 6.1.11.1.1).
- After filling up the image data for all classes switch to the Training tab.
- Select your previously created working directory.
- Select to create a new model.
  - Enter a name like "Animals01" in the dialog and press Save.
  - If requirements like number of classes and images are not satisfied a warning dialog appears and the training is aborted. A detailed error message is printed to the event log (see 6.1.1.1).
  - Now the model is created. The progress is shown in a dialog.
    The training process takes between 10 seconds and some minutes.
- After creating a model, the details are shown in the information group.

The creation of a new **Anomaly Detection** model is similar. But only one Class "good" needs to be created.

Select the new model for image processing:

- Open the Configuration dialog and go to Image Processing tab.
- In Processing Framework select Python API.
- In Procedure Name select `ClassificationInference` for **Classification** or `AnomalyInference` for **Anomaly Detection**.
- Open the Parameters dialog box.
- Go to Init Parameters by shifting the parameter groups to the left.
  - Set the parameter tflite_model to "`model_edgetpu`" respectively "`anomaly_model_edgetpu`" when using **Anomaly Detection**.
  - Set model to "`Animals >> Animals01`"
    (First part is Working Directory, second the name of the model).
  - Set logger_level to `Warning` to reduce the amount of logging.
- When using anomaly detection go to Input Parameters.
  - A threshold of 1000 will use the default value created from training. A higher anomaly_score indicates a greater difference to the "good" images. If the anomaly_score of an image is larger than threshold, it is considered as anomaly.
  - Set display_anomaly_heatmap to 1 to draw a heatmap on top of the image.
  - heatmap_alpha sets the transparency of the heatmap.

Select ✓ Use to accept the new configuration.

6.1.12 About

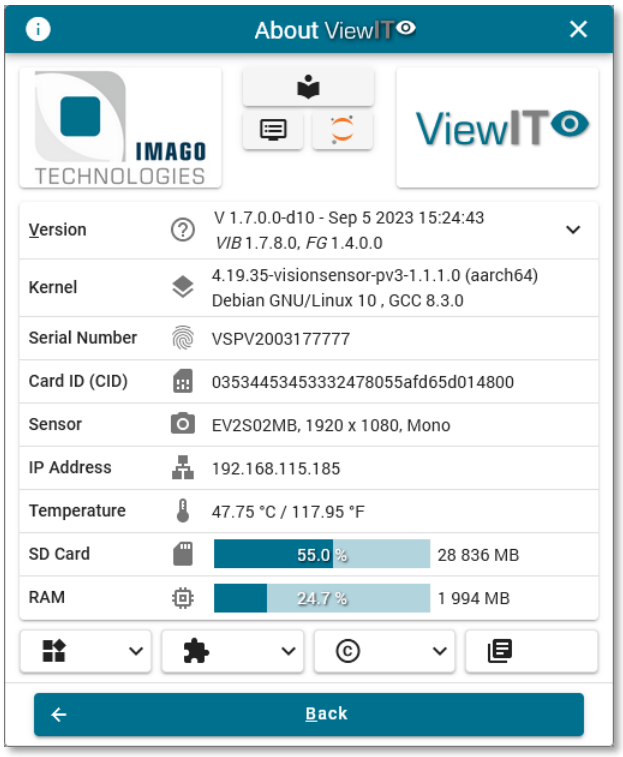


Figure 51: Information Dialog

This dialog shows information like program version, serial number of the device, operating system, available capacity of the flash card, etc.

opens a new browser tab which gives access to the **REST API** (see section 6.3).

opens a dialog window containing this documentation. The user can move the document to a new browser tab by pressing in the title bar. Use in the bottom left to switch between
- Manual,
- Plug-In API, the documentation how to develop a plug-in,
- OpenAPI, the interactive documentation of the REST API,
- IMAGO SDK documentation.

opens a new browser tab and starts a JupyterLab server. (This feature is not available in all editions.)

opens a card with a list of features.

opens a card with a list of used components and their version numbers.

opens a list with copyright information for the used components.

shows the change log.

Back closes the dialog.

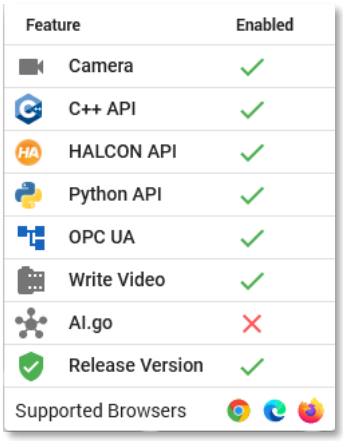


Figure 52: Available Features

Pressing the features button opens a card showing all available features for the running application. The features depend on the installed ViewIT edition.

The editions including HALCON API and OPC UA support require an appropriate license file.

6.1.13 Exit

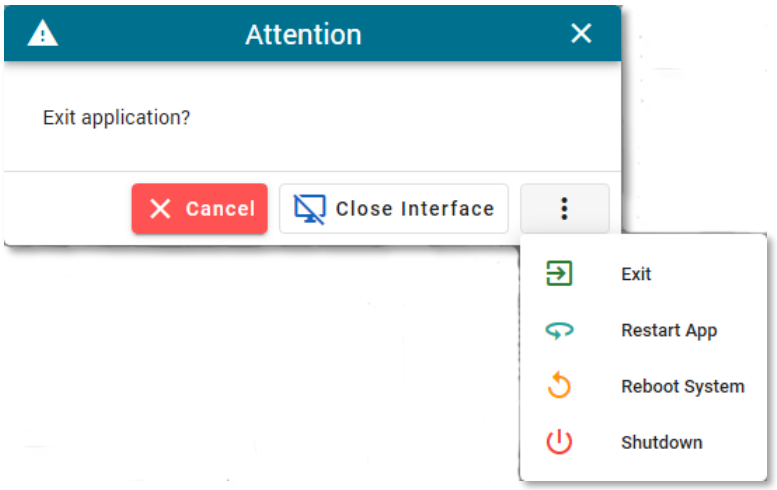


Figure 53: Exit Dialog

Cancel closes the dialog.

Close Interface stops the web interface. In this mode the camera grabs and processes image without visualization.

⋮ unfolds the following options:

Exit quits the application

Restart App restarts the program.

Reboot System quits the application and reboots the camera.

Shutdown quits the application and switches camera off.

Figure 54: Closed Interface

In closed web interface mode, you can open the main menu again by pressing

Restart.

Production opens the **Production Page** in a separate browser tab.

6.2 Production Page

The production page can be opened by entering the IP address of the Vision Cam into the address field of a web browser and adding `/production`, e. g. `http://192.168.115.86/production`.

It is independent from the main web interface.

This page shows the current state from the image processing in a well-arranged layout without impact to the CPU load. The values of output parameters and statistics are updated five times per second, the scaled down camera image only once every five seconds.

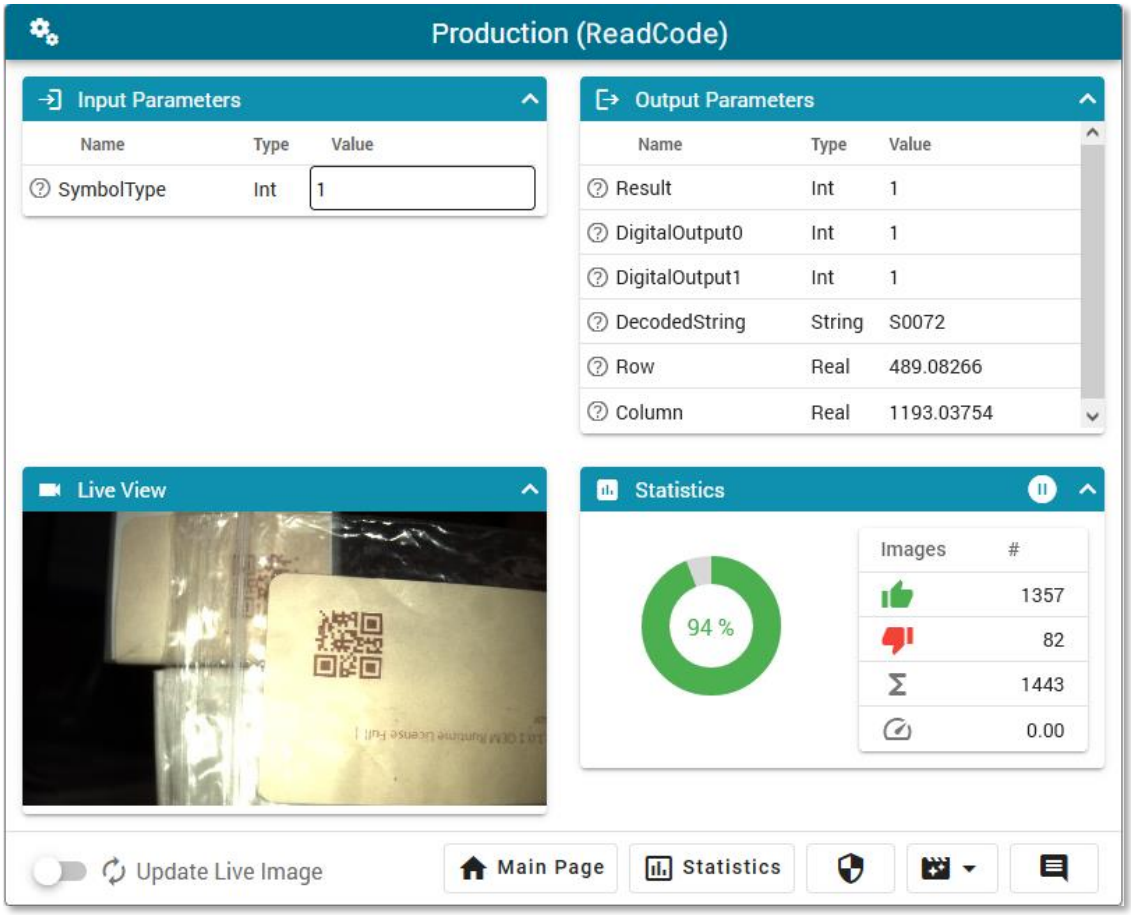


Figure 55: Production Page

Update Live Image enables / disables refreshing of the shown image.

Main Page opens the main screen of the web interface (see section 6.1).

Statistics opens the statistics dialog (see section 6.1.6).

opens the administrator login dialog (see section 6.1.1.2).

changes the processing step and shows its related the parameters.

shows the current log data.

At administrator level the user can change the values of the input parameters.

6.3 OPC UA Interface

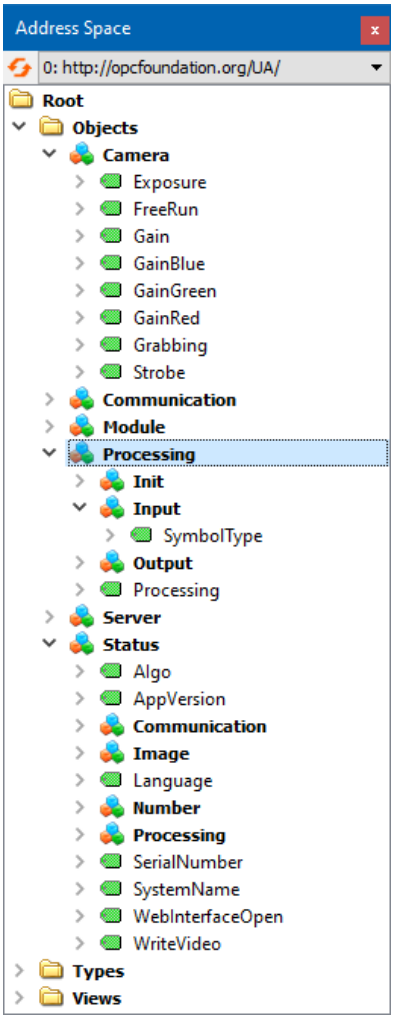


Figure 56: OPC UA Address Space Tree

OPC Unified Architecture (OPC UA) is a machine-to-machine communication protocol (https://opcfoundation.org/ about/what-is-opc/).

All data in the address space is represented hierarchically in a structure of files and folders.

An OPC UA client can subscribe to elements in the address space and monitor changes of the data. The client can also write data based on access permissions.

ViewIT contains an OPC UA server which automatically creates an address space that contains `Camera`, `Communication`, `Module`, `Processing` and `Status` data. The `Processing` and `Communication` folders and sub-folders are generated by the parameters of the used image processing and I / O functions.

The OPC UA client can read all information from the tree. Data in the folders `Camera`, `Communication` and `Processing` → `Init` / `Input` is also writable.

The connection can be encrypted by a self-signed certificate with different security levels like Basic 256 SHA256 or AES128 SHA256 RSA-OAEP.

6.4 REST API

REST API means **RE**presentational **S**tate **T**ransfer **A**pplication **P**rogramming **I**nterface. This interface is based on the behavior of world wide web (WWW) and is a basic approach for the communication between client and server. This implementation exchanges data via JSON (**J**ava**S**cript **O**bject **N**otation).
The REST API is accessed by calling a specific URL (**U**niform **R**esource **L**ocator).
The URL consists of a scheme, the host address, an optional port number and a path.

Example: http://192.168.115.188:80/help
Scheme://Host:Port/Path

IP address and port number can be changed in the General Tab of the Configuration Dialog (see section 6.1.5).

The interactive OpenAPI documentation that is available from the **About Dialog** contains more information about the several commands. The web page is accessible by `http://ip/rest-api/rest_api.html`.

6.4.1 HTML Output

When entering the URL with path `/help` (for example `http://192.168.115.189/help`) the information page of the REST API is loaded.

This page gives an overview of available paths. A path can link to a simple web page that is representing the information in human readable form or to the data in JSON or XML representation.

HTTP Interface

Post Commands

Feature	Action	URL	ID	Value
☑ Use Strobe	Set	UseStrobe	useStrobe	true / false
☑ Free-Run	Set	UseFreeRun	useFreeRun	true / false
☑ Processing	Set	UseProcessing	useProcessing	true / false
☐ I/O	Set	UseIo	useIo	true / false
☐ Grabbing active	Set	IsGrabbing	isGrabbing	true / false
Get list of available modules		GetModules		
Set camera module[1]		SetModuleCamera	module	filename
Set image processing module[1]		SetModuleProcessing	module	filename
Set I/O module[1]		SetModuleIo	module	filename
Set OPC UA module[1]		SetModuleOpcUa	module	filename
Set modules from recipe[1]		SetModuleReicpe	module	filename
Get settings value		GetSetting	camera / data / io / processing	parameter

[1] Only if web interface is closed and camera stopped

POST http://192.168.115.189/URL 'Content-Type: application/json' {"ID":value}

Examples (Windows):

```
curl -X POST http://192.168.115.189/IsGrabbing -d "{""isGrabbing"":false}"
curl -X POST http://192.168.115.189/GetSetting -d "{""data"":null}"
curl -X POST http://192.168.115.189/GetSetting -d "{""processing"":{""procedureName"":null}}"
curl -X POST http://192.168.115.189/GetSetting -d "{""camera"":{""exposure"":null,""gain"":null}}"
```

Web Links

Web Link	JSON Link	XML Link	Description
/image/Live.jpg /image/Live.png /image/Live.bmp			Current grabbed image
/images	/Json/images /images.json	/Xml/images /images.xml	List of saved images
/videos	/Json/videos /videos.json	/Xml/videos /videos.xml	List of saved videos
/log/Current.htm	/Json/currentlog /currentlog.json	/Xml/currentlog /currentlog.xml	Current log
/logs	/Json/logs /logs.json	/Xml/logs /logs.xml	List of saved log files
/inits	/Json/inits /inits.json	/Xml/inits /inits.xml	List of processing initialization parameters
/inputs	/Json/inputs /inputs.json	/Xml/inputs /inputs.xml	List of processing *Input* parameters
/outputs	/Json/outputs /outputs.json	/Xml/outputs /outputs.xml	List of processing *Output* parameters
/cominits	/Json/cominits /cominits.json	/Xml/cominits /cominits.xml	List of communication initialization parameters
/pres	/Json/pres /pres.json	/Xml/pres /pres.xml	List of communication *Pre* parameters
/posts	/Json/posts /posts.json	/Xml/posts /posts.xml	List of communication *Post* parameters
	/Json/parameters /parameters.json	/Xml/parameters /parameters.xml	List of all parameters
/status	/Json/status /status.json	/Xml/status /status.xml	List of status values
/results	/Json/results /results.json	/Xml/results /results.xml	List of results
/buffers	/Json/buffers /buffers.json	/Xml/buffers /buffers.xml	List of image buffers
	/Json/modules /modules.json	/Xml/modules /modules.xml	List of available modules
/main			Main application page
/production			Production page
/processing			Image processing procedure page
192.168.115.189:8888/lab			Open jupyterlab page

Figure 57: REST API Help Page

6.4.3 JSON Output


```
{
   "image" :
   {
      "index" : 104,
      "name" : "/buffers/index/104.png",
      "time" : "2020-05-27 11:33:12,844"
   },
   "outputs" :
   [
      {
       "name" : "Result",
       "type" : "Int",
       "value" : 1
      },
      {
       "name" : "DigitalOutput0",
       "type" : "Int",
       "value" : 1
      },
      {
       "name" : "DigitalOutput1",
       "type" : "Int",
       "value" : 1
      },
      {
       "name" : "DecodedString",
       "type" : "String",
       "value" : "S0072"
      },
      {
       "name" : "Row",
       "type" : "Real",
       "value" : 530.06800969011329
      },
      {
       "name" : "Column",
       "type" : "Real",
       "value" : 828.17220237854565
      }
   ]
}
```


The JSON output uses human readable text to store data objects consisting of attribute / value pairs and arrays of data.

The format was derived from JavaScript, but many tools and programming languages can generate and parse data in this format.

Figure 58: JSON output

7 Configuration

7.1 Configuration Files

The configuration files are stored in the folder `/opt/ImagoTechnologies/ViewIT/.viewit/`. They are also using the JSON data format. The general configuration is handled by the file `ViewIT.ini`. It contains among other things the values for the `AdminPassword` and browser `Authentication` with `Username` and `Password`.

It is possible to modify the settings files manually. But this should only be made by advanced users.
Each settings type has its own file extension:

Type	Extension
Camera setting	`.camset`
Processing setting	`.procset`
I/O setting	.ioset
OPC UA setting	.opcset
Recipe containing all settings	.recipe

7.2 Image Processing Plug-Ins

Plug-ins for image processing are stored in the folder `/opt/ImagoTechnologies/ViewIT/procedures/`. The process cycle is described in section **Fehler! Verweisquelle konnte nicht gefunden werden.**. You can create a plug-in either with HALCON, C++ or Python. For more information about function interfaces and detailed parameter lists see the additional document `ViewIT_Plug-In_API.pdf`.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Processing</title>
  <link rel="stylesheet" href="fromres/vuetify.css" />
  <link rel="stylesheet" href="fromres/extra.css" />
</head>
<body>
  <div id="app" class="v-application">
    <div class="mx-auto mt-8 pa-0 v-alert elevation-2">
      <div class="pa-4">
        <i class="v-icon material-icons green--text mr-2"
           style="font-size:32px">info
        </i>
        Web page created by
        <div id="name" class="source"
          style="display:inline">
        </div>
      </div>
      <hr id="divider" class="v-divider" />
      <div class="pa-4">
        <i class="v-icon material-icons grey--text mr-3">
           message
        </i>
```

Each image processing procedure can have an individual user designed **web page**. Make a folder with the name of the procedure in the `procedures` folder and create a file named `index.html`.

The page has access to the internal web resources of ViewIT. You can link to the stylesheets by `fromres/vuetify.css` and `fromres/extra.css`.

When you include the Vue.js framework by `<script src="fromres/vue.js">` and Vuetify by `"fromres/vuetify.js"` you can create user interface components like buttons, sliders, or dialogs.

In the same way you can link to your own resource files located in the folder, but without the prefixed `fromres` path. See the included examples for more information how to create interactive user interfaces.

```
        <div id="comment" class="ml-11 mt-n8 text-body-2">
        </div>
      </div>
    </div>
  </div>

  <script>
function JsonRequest(method, url, value, callBack) {
  let req = new XMLHttpRequest();

  req.responseType = 'json';
  req.onreadystatechange = () => {
    if (req.readyState === 4 && req.status === 200)
      callBack(req.response);
  };
  req.open(method, '../' + url, true);
  req.setRequestHeader('Content-type', 'application/json');
  req.send(JSON.stringify(value));
}

JsonRequest('POST', 'GetSetting', { 'processing': null},
  response => {
  let name = response.processing.procedureName.value;

  document.getElementById('name').innerHTML = name;
  document.getElementById('comment').innerHTML =
    response.processing.procedureComment.value;
  document.title = name;
});
  </script>
</body>
</html>
```

The adjacent example uses a JavaScript function `JsonRequest()` to retrieve information via the REST API. It posts a `GetSetting` message and receives a response containing data from the `processing` section in form of a JSON object. This information is used to modify the contents of the HTML elements with ID `name` and `comment`.

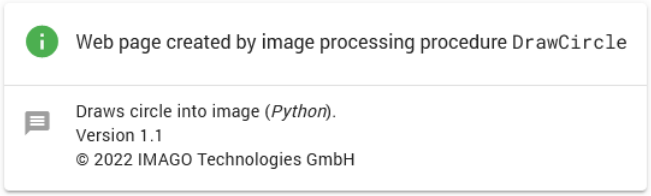


Figure 59: User Web Page

7.2.1 HALCON Procedures

The image processing uses the **HDevEngine** from HALCON. It allows executing HALCON scripts / procedures by applications outside of **HDevelop**.

The user created functions are stored in the `procedures` folder or in subfolder which is given in the HALCON settings dialog (6.1.5.4.2). The ViewIT application scans this folder for available procedures when starting. There is a library file `ViewIT.hdpl` that contains some examples. It is possible to edit this file in HDevelop. You can modify existing procedures and create new ones. You can also create a new procedure library file and copy it to the folder.

There must be three procedures in each library file: `Init`(), `CleanUp`() and the **User Procedure** that is selected in Configuration Image Processing Tab (see section 6.1.5.4**Fehler! Verweisquelle konnte nicht gefunden werden.**).

7.2.2 General C++ API

It is possible to create dynamic libraries containing a user procedure. It uses the process cycle described in section **Fehler! Verweisquelle konnte nicht gefunden werden.**. The ViewIT application will detect all `lib???.so` files stored in the folder `procedures` folder.

There is an example project which contains the class `MyProcessing`. This class has some fixed functions that are necessary for linking and managing the library. The user can modify the functions and is also allowed to add own member variables to the class.

7.2.3 Python API

Python scripts stored in the folder `/opt/ImagoTechnologies/ViewIT/procedures/` can be executed. The same process cycle is used as in HALCON and C++ procedures.

The data from grabbed images is transferred to the Python module as a multidimensional array (`numpy.ndarray`). Therefore, importing the Python package `numpy` is required. Also, the packet `python3-opencv` is useful.

7.3 Communication Plug-Ins

The communication plug-ins are similar to those for image processing. The files are stored in the folder `/opt/ImagoTechnologies/ViewIT/communications/`. In contrast to image processing these plug-ins have two working functions – `Pre`() and `Post`(). The first function is called before image processing, the second after.

7.4 SSL Key

When using a **HTTPS** connection ViewIT is using a self-signed certificate. It is also possible to use your own **key pair**. For this, you need following files in the folder `/opt/ImagoTechnologies/ViewIT/.viewit/`:

1. `certificate.crt` which contains a certificate in PEM format and Base64 ASCII encoding.
2. `certificate.key` which holds the RSA private key in the same format.

If you do not have a key pair, you can generate it.

At first create the file `localhost.ext` with domain information:

```
authorityKeyIdentifier = keyid,issuer
basicConstraints = CA:FALSE
keyUsage = digitalSignature, nonRepudiation, keyEncipherment, dataEncipherment
subjectAltName = @alt_names

[alt_names]
DNS.1 = localhost
IP.1 = 127.0.0.1
```

Now use a Linux shell to create the key files:

- Generate CA (Certificate Authority) certificate:
  - Generate private key:
    `openssl genrsa -out CA.key -des3 2048`
    Enter a pass phrase for CA key and enter again for verification.
  - Generate root CA certificate:
    `openssl req -x509 -sha256 -new -nodes -days 3650 -key CA.key -out CA.pem`
    Re-enter pass phrase and enter information for country, locality, organization, etc.
- Generate certificate:
  - Create localhost
    `openssl genrsa -out localhost.key -des3 2048`
    Enter a pass phrase for key and enter again for verification.
  - Generate CSR
    `openssl req -new -key localhost.key -out localhost.csr`
    Re-enter pass phrase and enter information for country, locality, organization, etc.
  - Generate certificate
    ```
    openssl x509 -req -in localhost.csr -CA CA.pem -CAkey CA.key
     -CAcreateserial -days 3650 -sha256 -extfile localhost.ext
     -out certificate.crt
    ```
    Re-enter pass phrase.
  - Generate RSA private key
    `openssl rsa -in localhost.key -out certificate.key`
    Re-enter pass phrase.
- Copy `certificate.crt` and `certificate.key` files:
  `cp certificate.* /opt/ImagoTechnologies/ViewIT/.viewit/`

After re-starting ViewIT and connecting via HTTPS you will be asked again by the browser to trust the certificate of the web page.

7.5 JupyterLab

JupyterLab is a next-generation web-based user interface for **Project Jupyter**. JupyterLab enables you to work with documents and activities such as Jupyter Notebooks, text editors, terminals, and custom components in a flexible, integrated, and extensible manner.

Jupyter Notebook is a web-based interactive environment that can be used to create Jupyter Notebook documents. Figures, text, graphics, and executable program code can be summarized in it and made available to users. The user accesses a notebook from the Jupyter notebook server via a web browser and can interact with the data and information. Typical application areas for the notebooks are scientific data analysis and visualization, such as those used in statistical environments, business intelligence applications, or machine learning.

Jupyter notebooks are always useful when interactive data analyses and visualizations are to be made available in a simple and elegant way. Especially for ViewIT, this is a very valuable way to easily visualize and analyze the output of communication and image processing plug-ins, which communicate via the Python API. A Jupyter Notebook can be converted from the browser interface into different formats.

Two examples are preinstalled to show how to access grabbed images from the camera and call an image processing procedure.

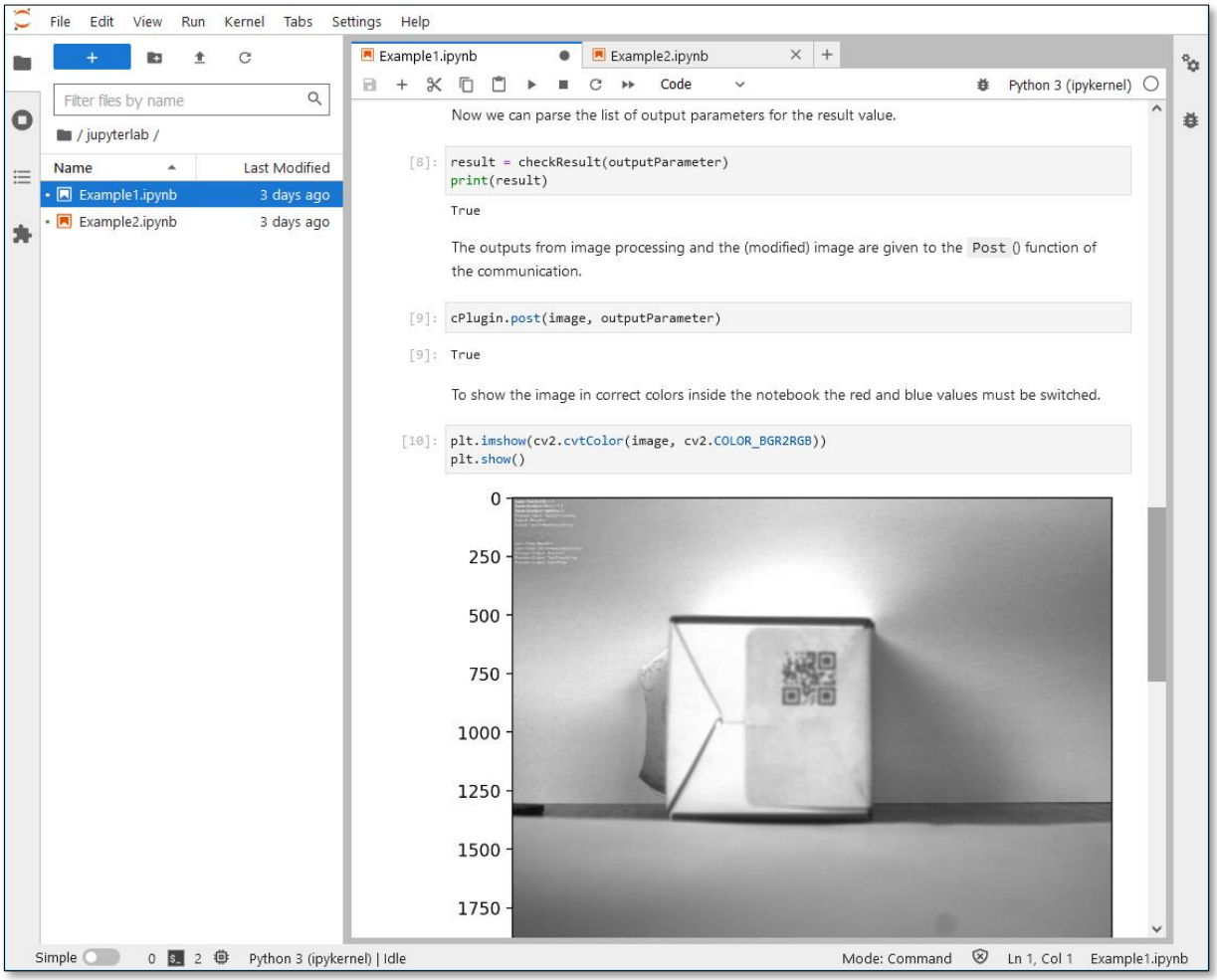


Figure 60: JupyterLab

8 Support

Finally, if you have any open questions, the IMAGO support team is happy to assist you in any cases. For direct contact to the support, please use our ticket system: https://imago.freshdesk.com

Also, visit our IMAGO Download Portal: https://www.imago-technologies.com/support

9 History

Revision	Date	Changes
1.0	Nov-15-2021	First release
1.7	December 2023	Update for ViewIT version 1.7