TECHNICAL INFORMATION

Supporting Material for Visionary-S CX

3D machine vision

SICK
Sensor Intelligence.

Valid for products

Visionary-S CX

Manufacturer

SICK AG
Erwin-Sick-Str. 1
79183 Waldkirch
Germany

Legal information

Original document

This document is an original document of SICK AG.

Content

1 Introduction and Download

1.1 Introduction

Information material and the necessary software are available to assist you in setting up the device. You will find all of these on sick.com.

In the following you will find an overview and explanations of the individual contents.

2 Contents

2.1 DOC

2.1.1 Product introduction and data sheet

- Short overview about the device and its technical specifications.

2.1.2 SOPAS Installation & Embedding

- Documentation how to install SOPAS ET and how to establish a connection to Visionary-S with it

2.1.3 Device configuration user guide

- Detailed description of the Visionary-S settings and filters that can be adjusted via the SOPAS ET GUI. Tips about how to configure your device dependent on your circumstances.

2.1.4 Diagnostics description

- Description of the diagnostic tools in SOPAS ET.

2.1.5 Trigger mode description

- Description of how to use the trigger mode of Visionary-S.

2.1.6 Firmware update guide

- Description of how to update the Visionary-S device.

2.1.7 3D Coordinate transformation

- Detailed background information on 3D point cloud transformation of Visionary-S data. Also covers formulas and programming hands on.

2.1.8 Emulator description

- Describes usage of the Visionary emulator manager for offline development with SSR files.

2.1.9 SD Card Cloning

- Description on how to export and import configuration jobs using only a SD card

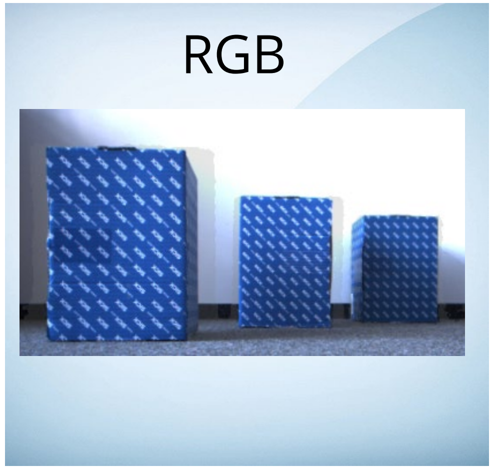
2.1.1
Product introduction

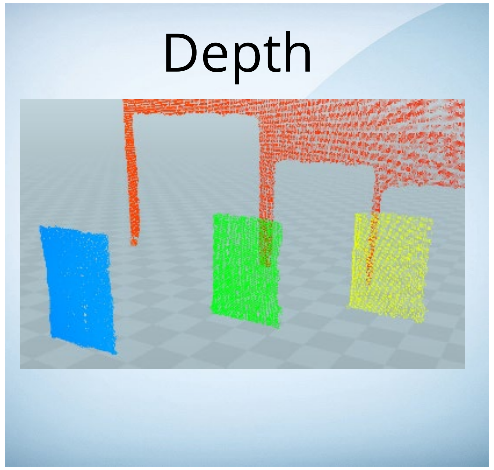
SICK AG – Mobile Perception – 3D Snapshot

3D Snapshot Technologies

Introduction

3D snapshot means capturing a static scene three-dimensionally in one shot without moving mechanical parts inside the device





Technologies:

- **Visionary-S: Precise 3D structured light stereo**
- Visionary-T: Accurate 3D time-of-flight
- Visionary-B: Rugged 3D stereo vision
- Diverse combinations of technologies mentioned above

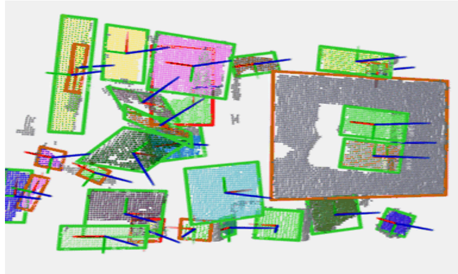
SICK - 3D Snapshot

VISIONARY-S (structured light stereo) , **VISIONARY-T** (3D time-of-flight) and **VISIONARY-B** (passive stereo)

No matter which 3D snapshot vision technology is required, SICK offers you what you need for your application.

Visionary-S CX

Benefits



- Robust camera for industrial use with an optimal working range of 0.5 to 2.5m
- Over 320,000 distance and color values in one shot
  → 3D information even for stationary objects
- High frame rate coupled with very high accuracy
- Quicker results: output of color and depth values directly from the camera
- High quality color and depth data enable reliable solutions for the factory and logistic automation
- Flexibility: the camera is compatible with most of the common programming languages and easy to configure with SOPAS ET
- Highly efficient illumination: reliable depth data even under ambient light conditions

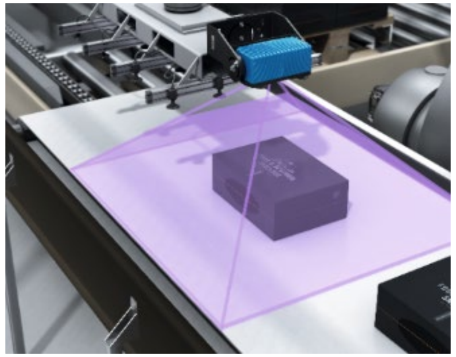
Visionary-S CX

Possible applications

SICK
Sensor Intelligence.

Robotics , Intralogistics, Consumer goods, Machine tools

- Bin picking
- Navigation, positioning of robot arm
- Quality inspection, e.g. secondary packaging
- Palletizing & depalletizing
- Goods dimensioning
- Further applications..



Visionary-S CX

Key Parameters

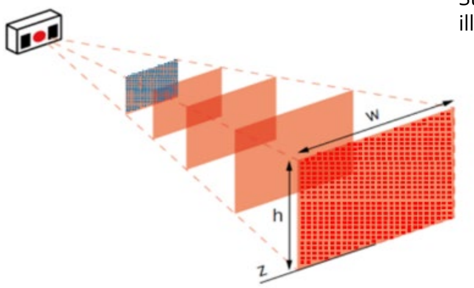
Key parameters	
Detection distance	0.5m - 2.5m
Angle of view	~60° x 50°
Resolution 3D / RGB	~640px x 512px
Sunlight robustness	Up to 40kLux
Frame rate	Up to 30 fps
IP class	IP67
Supply voltage	24 V +- 15%
Power consumption	19 W / 1.6 A (peak)
Weight	1.7 kg (2.2 kg)[1]
Dimensions (W x H x D)	162 x 93 x 78 mm (162 x 116 x 104 mm)[1]
Temp range	0°...40°C (0°...50°C)[1]
Interface	Distance and RGB data via TCP/IP
Laser safety	Class 1 (λ: 808 nm); EN/IEC 60825-1:2014; EN/IEC 60825-1:2007
Shock resistance	According to EN 60068-2-27:2009
Vibration resistance	According to EN 60068-2-6 and 60068-2-64
Electromagnetic compatibility (EMC)	EN 61000-6-2:2005-08 EN 61000-6-4:2007-01

[1]with cooling fins

[2]Average value at 90% remission

Distance (z)	FOV (w x h)	Pixel size
@0.5m	~45 cm x 45 cm	~1 mm
@1m	~100 cm x 90 cm	~2 mm
@1.5m	~160 cm x 130 cm	~3 mm
@2m	~220 cm x 180 cm	~4 mm
@2.5m	~280 cm x 230 cm	~5 mm

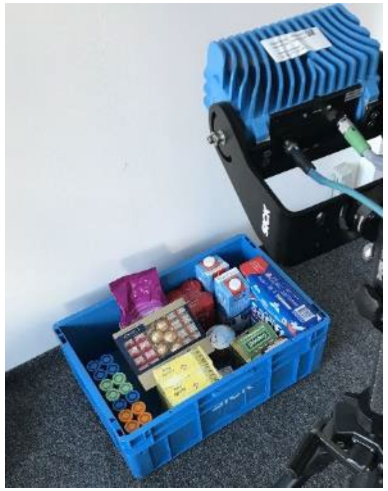
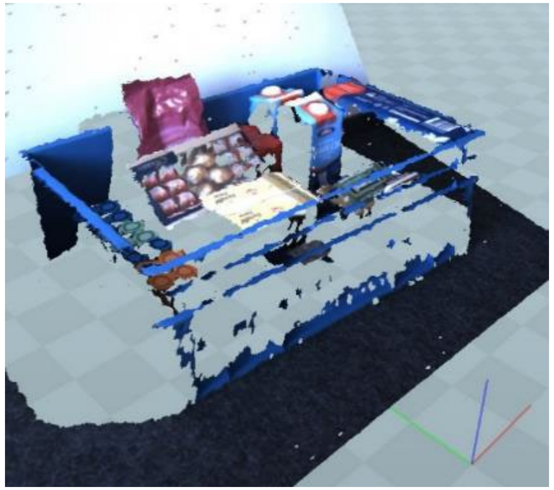
Distance (z)	Accuracy[2] Δz	Precision[2] σz
@0.5m	1.5 mm	0.25 mm
@1m	2.5 mm	0.6 mm
@1.5m	3 mm	1.5 mm
@2m	4.5 mm	2 mm
@2.5m	6 mm	4 mm

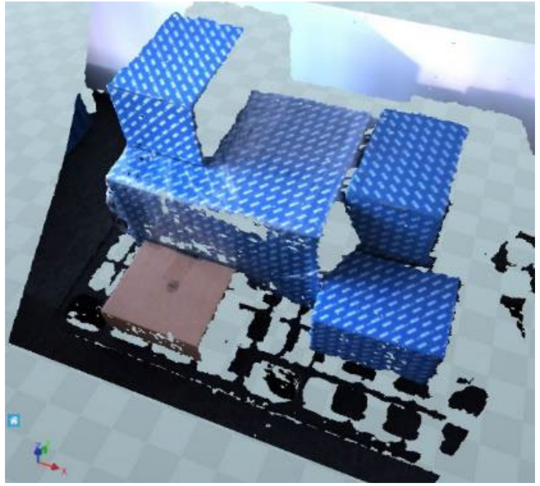
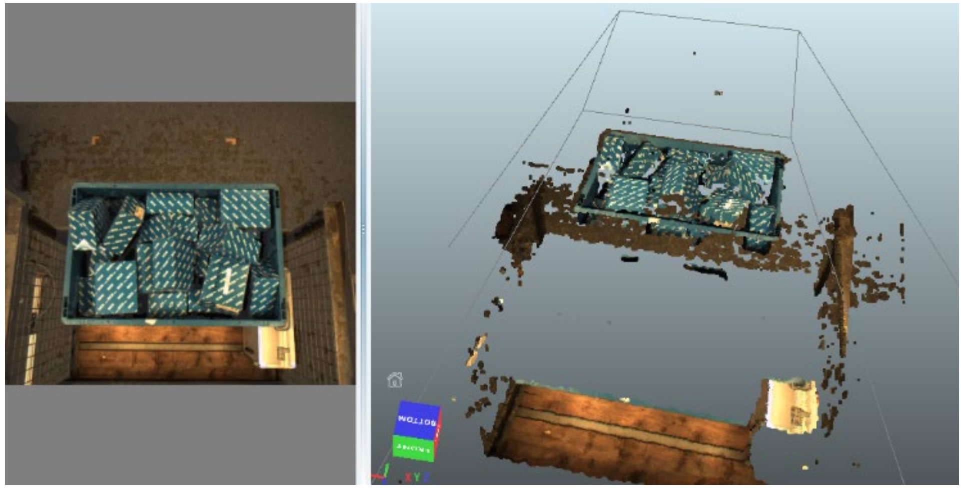
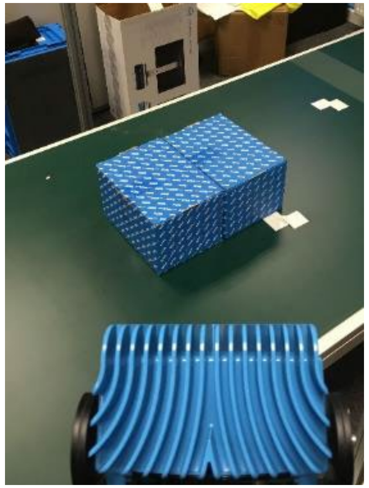
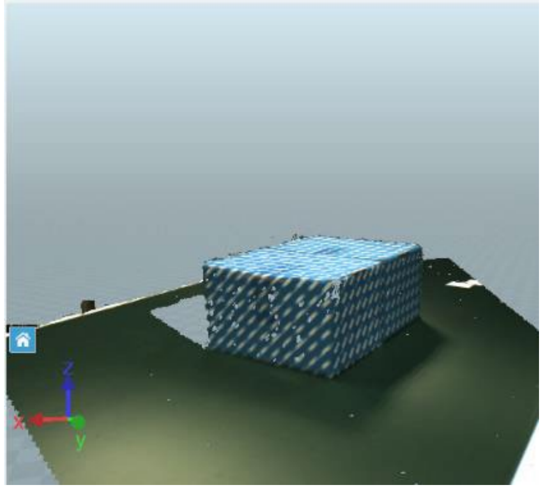


For more information, please refer the SICK website or contact SICK sales

Visionary-S CX
Data examples

SICK
Sensor Intelligence.

Thank you for your attention.

SICK AG – Mobile Perception – 3D Snapshot

2.1.2
SOPAS Installation & Embedding

SICK AG – Mobile Perception – 3D Snapshot

Content

SICK Sensor Intelligence.

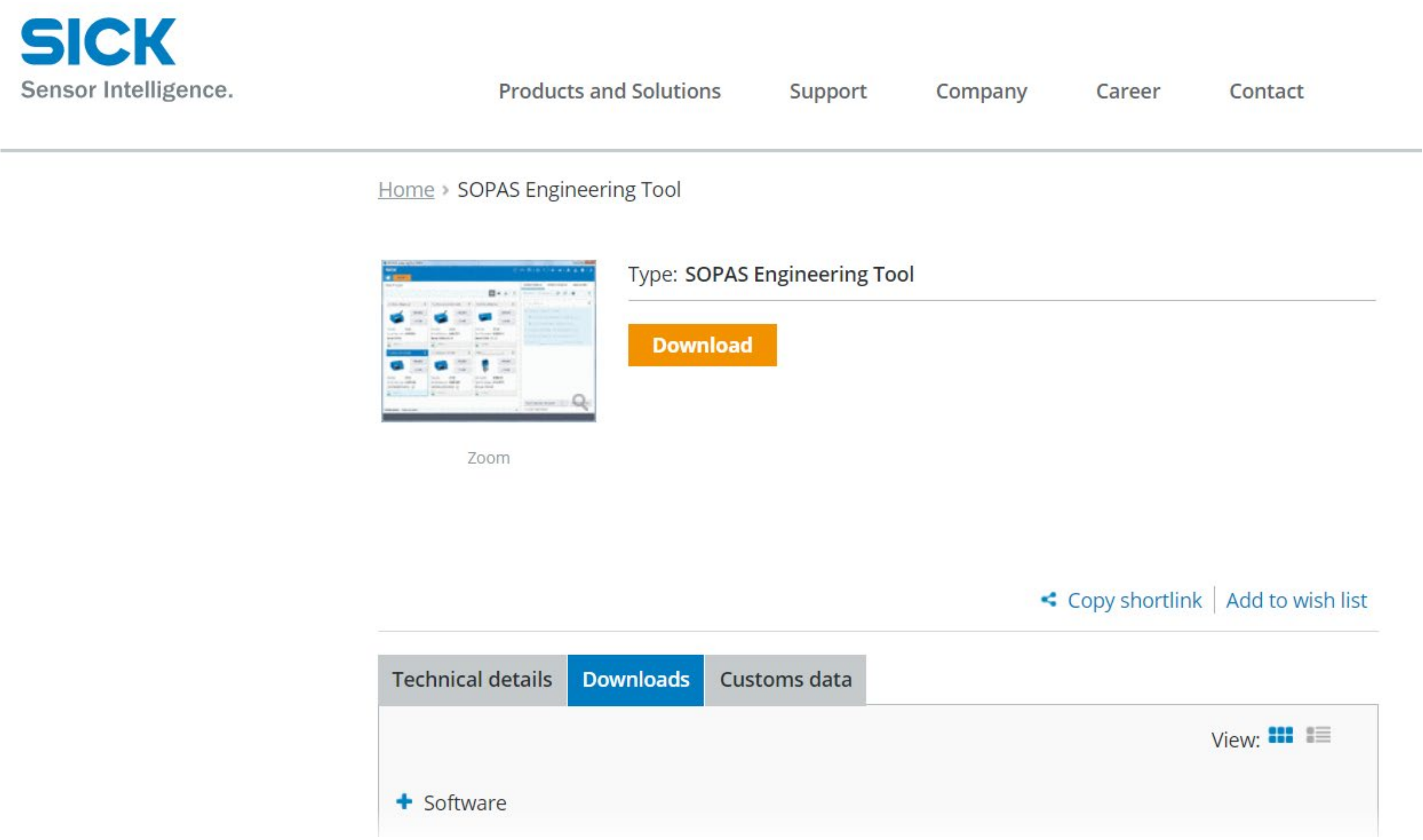
Prerequisites
MINIMAL SYSTEM REQUIREMENTS

Minimal system requirements	
Processor:	Intel® Core™ i5 2,6 GHz
RAM:	4 GB RAM
Interface:	Hardware communication channels such as serial interfaces, USB or Ethernet, depending on the SICK device
Operating system:	Windows 10, Windows 7 (32 bit/64 bit), Windows 8 (32 bit/64 bit)
Graphic interface:	e.g. Intel® HD Graphics 3000 (or NVIDIA® NVS 3100M 512MB gDDR3) and OpenGL 2.0 Support
Monitor:	Min. 256 colors - recommended 65,536 colors (16 bit Hi color)
Screen resolution:	1024 x 768 px
Hard disk space:	450 MB
Ethernet:	>1 Gbit/s

Install SOPAS ET

SICK
Sensor Intelligence.

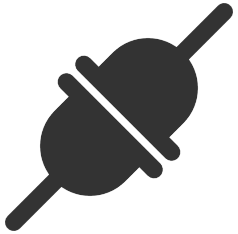
- SOPAS ET is available on https://www.sick.com/SOPAS_ET



STEP BY STEP
CHOOSE YOUR WAY TO INSTALL DEVICE

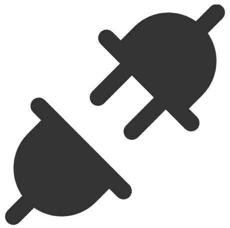
Two ways to embed a Visionary device into SOPAS

Online - requires a Visionary device



- Getting properly connected
- Install device description from device
- Go online and explore the GUI

Offline - requires one or more SOPAS device description (sdd) file(s)

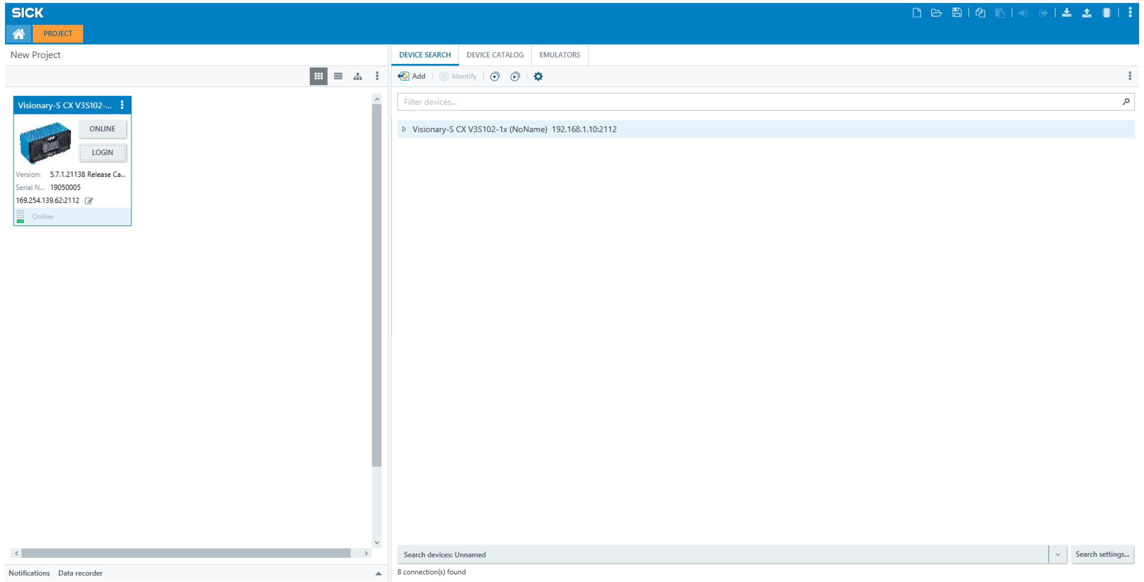


- Add SOPAS device description (sdd) from SICK.com or the Supporting material
- Open and explore the GUI in the offline mode (no data stream)

Getting properly connected

START SOPAS ET

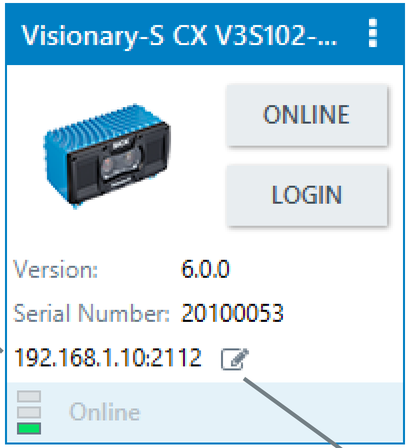
- Connect your device via Ethernet to your local PC
- Connect the unit to the power supply and wait until it has booted up
- Start SOPAS ET
- The device should be found and added automatically to the project



Getting properly connected
CHANGE IP ADDRESS

- If necessary, change the IP address of the device according your local network
- It’s also possible to change between static IP address or dynamic IP address via DHCP server

IP address of the device is displayed in the device tile.

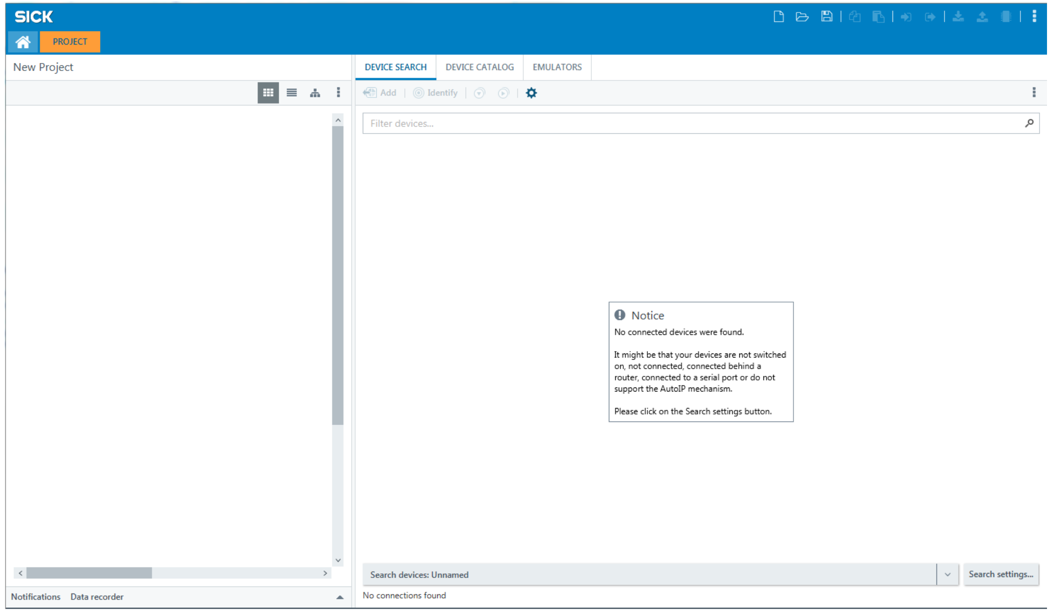


Clicking the pen allows to modify the IP address settings

Getting properly connected
DEVICE NOT FOUND

If the device was not found, check the following:

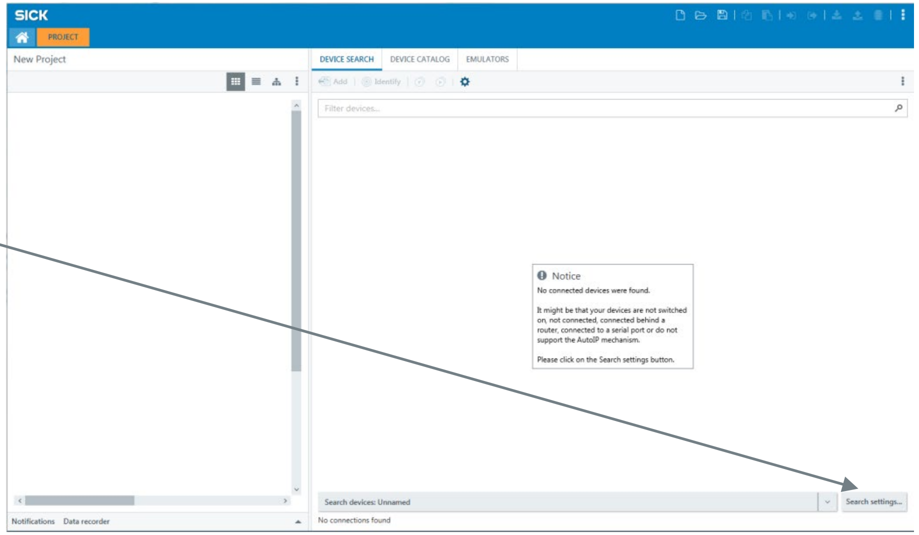
- First connect and power the device, after that start SOPAS ET.
- Check your local network settings.
- Default IP of the Visionary devices is 192.168.1.10



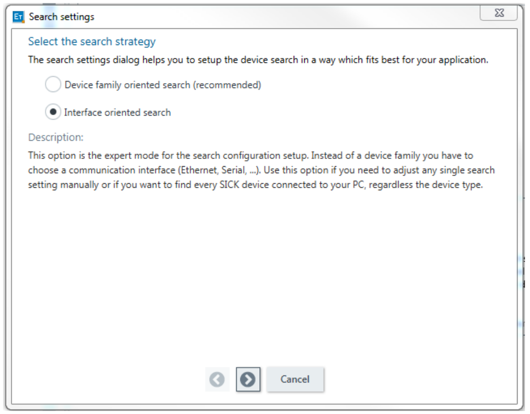
Getting properly connected
CHANGE SEARCH SETTINGS

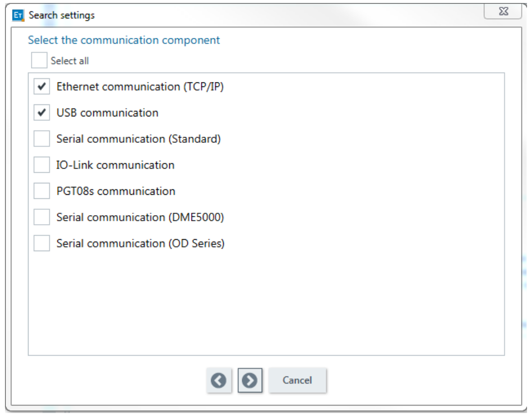
If the device was not found, check the following:

- Change search settings



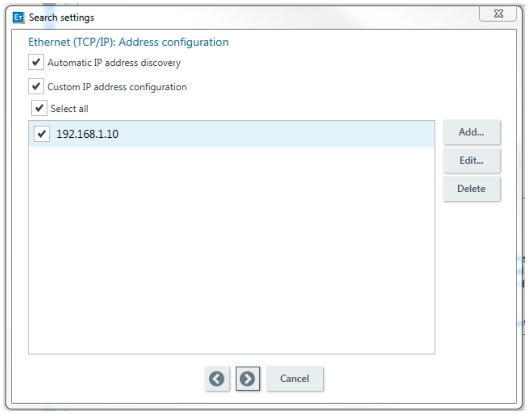
- Follow the standard SOPAS ET wizard.





Getting properly connected
CHANGE SEARCH SETTINGS

- Some local network settings or hardware may block the automatic IP address discovery scan which based on broadcast messages.
- Add the default IP **192.168.1.10** address to the search list.

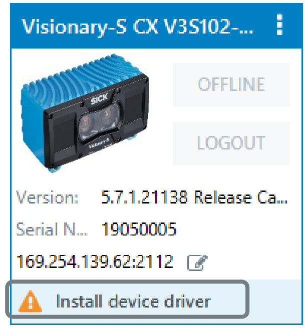


- If your device is not detected automatically, check your firewall settings and used hardware between your local host and the device.
- **Please make sure your firewall allows communication to the TCP-ports 2112, 2113, 2114**
- **In addition, the camera uses the UDP-port 30718 for AUTO-IP-Scan.**
  For this purpose, *Broadcast* must be enabled
- Duplicate IP addresses, firewall settings or used network components may also block the change of the IP address.

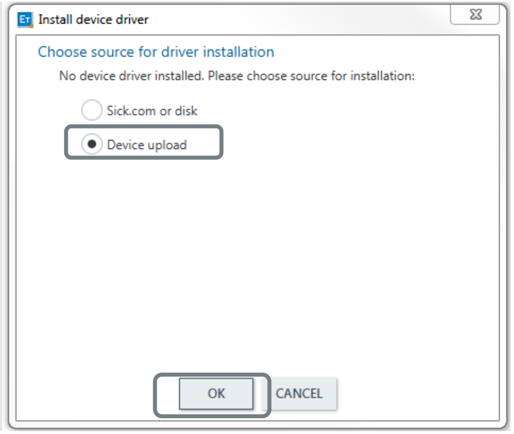
Install SOPAS Device description
(SDD) FROM DEVICE

After a successful connection, the driver might be missing.

- Click on *Install device driver*



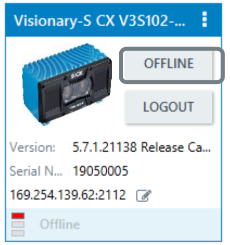
- Choose *Device upload*



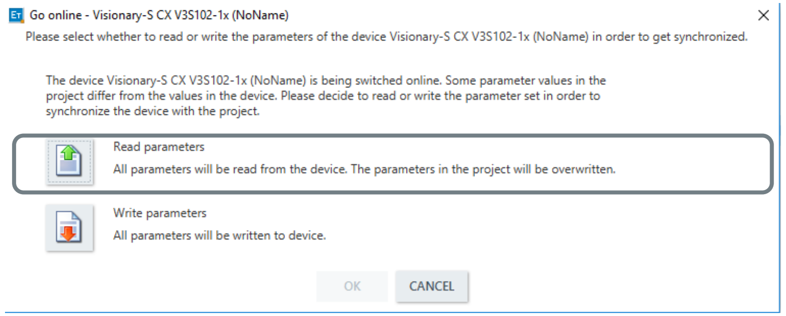
Go Online
(IF NOT AUTOMATICALLY DONE)

SICK
Sensor Intelligence.

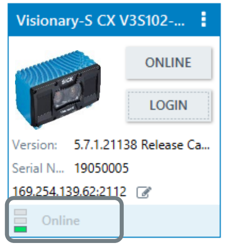
- Click *Offline* to go online



- Choose *Read parameters*



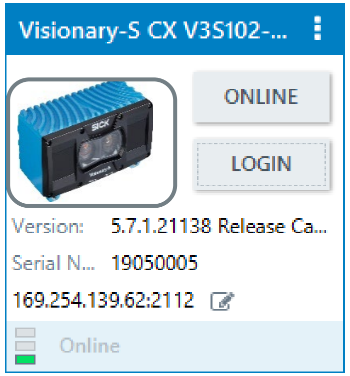
- Success!



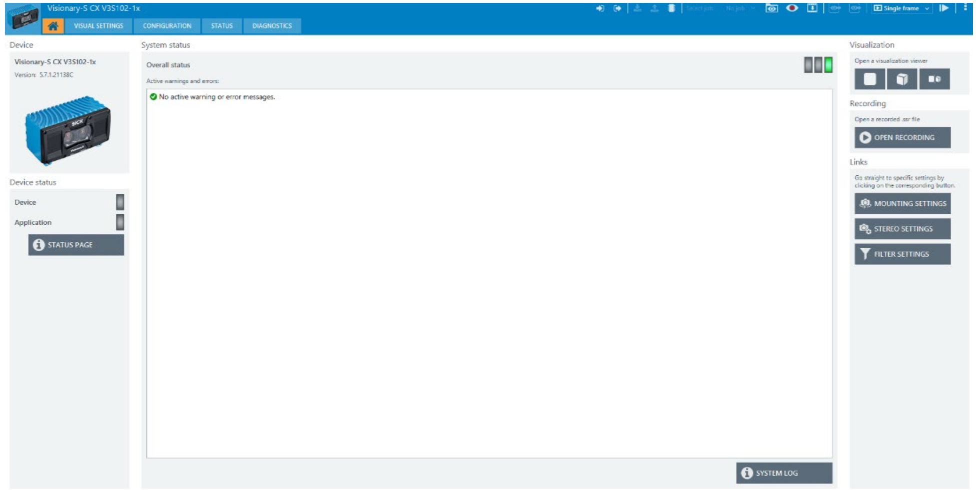
Device Window

OPEN DEVICE WINDOW

- Double click on the device tile to open the device window



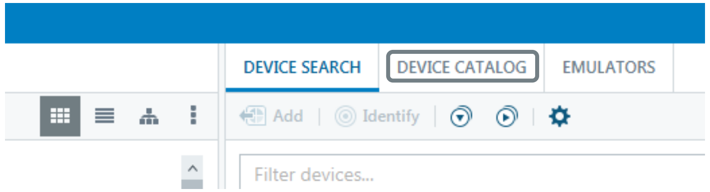
- Continue with GUI Configuration presentation for more details and examples



Install Device description

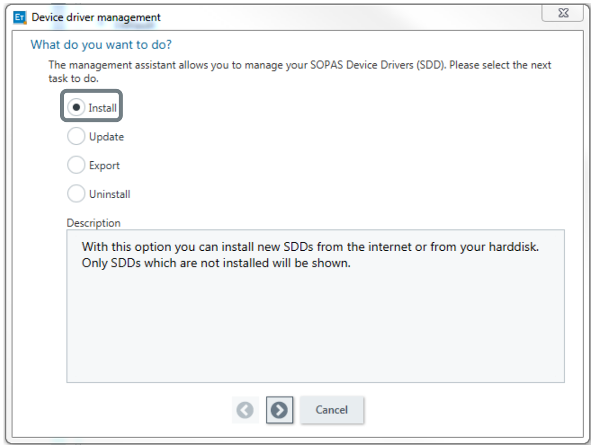
SDD FROM SICK.COM OR DISK

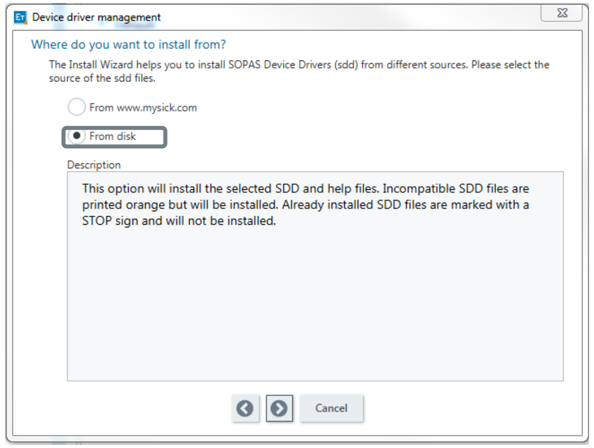
- Open *Device catalog*



- Start the device driver management

- Choose *Install →From disk*

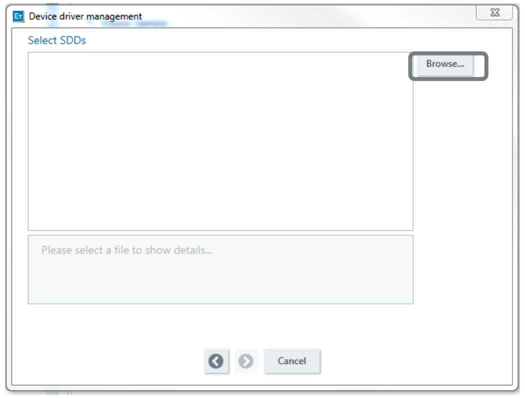




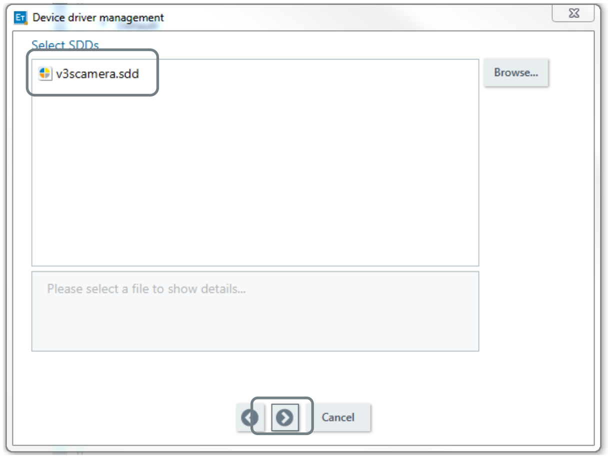
Install Device driver
SDD FROM SICK.COM OR DISK

SICK
Sensor Intelligence.

- Search and find *V3SCamera.sdd.*
  The SDD is included in the Supporting material



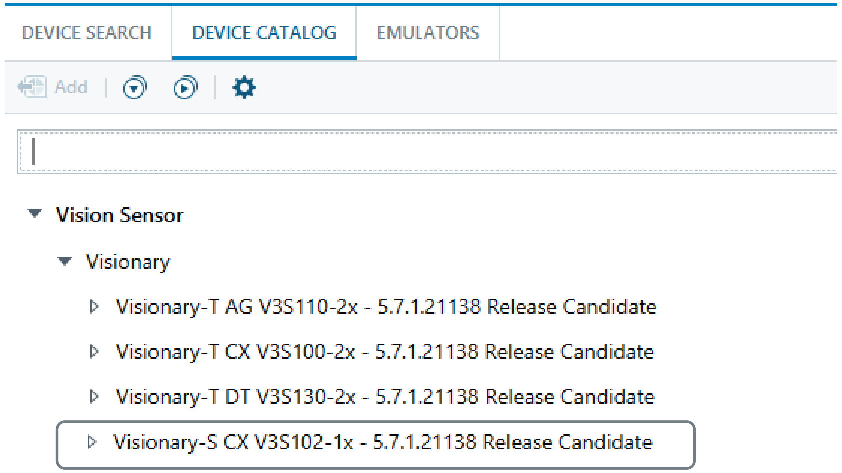
- Confirm selection



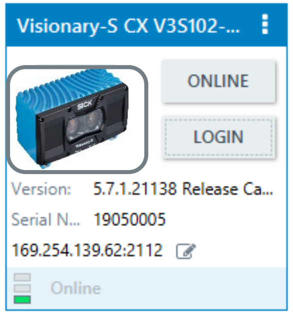
Open offline device

OPEN DEVICE CATALOG

- Double click on your Visionary Version
  - If more than one versions available, choose the latest one.



- Double click on the device tile to open the device window (no data stream)
- Continue with the Device configuration presentation for more details and examples



Thank you for your attention.

SICK AG – Mobile Perception - 3D Snapshot

2.1.3
Device configuration user guide

SICK AG – Mobile Perception - 3D Snapshot

Content

Content

Visionary-S Device Page

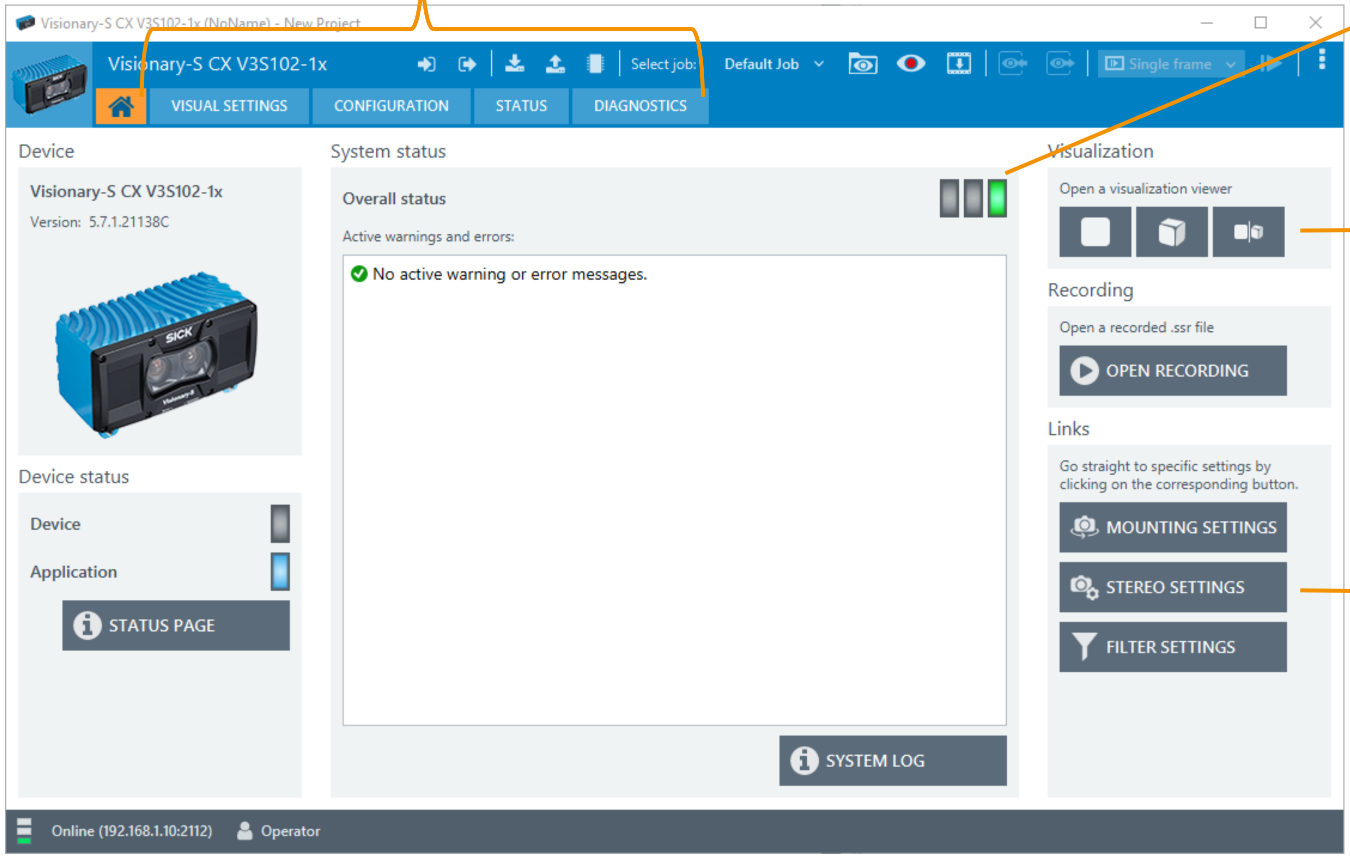
Overview

Use the main navigation tabs for quick navigation

Check if the system works without any errors

Start the visualization with one of the viewers

Use the links to navigate quickly to important settings

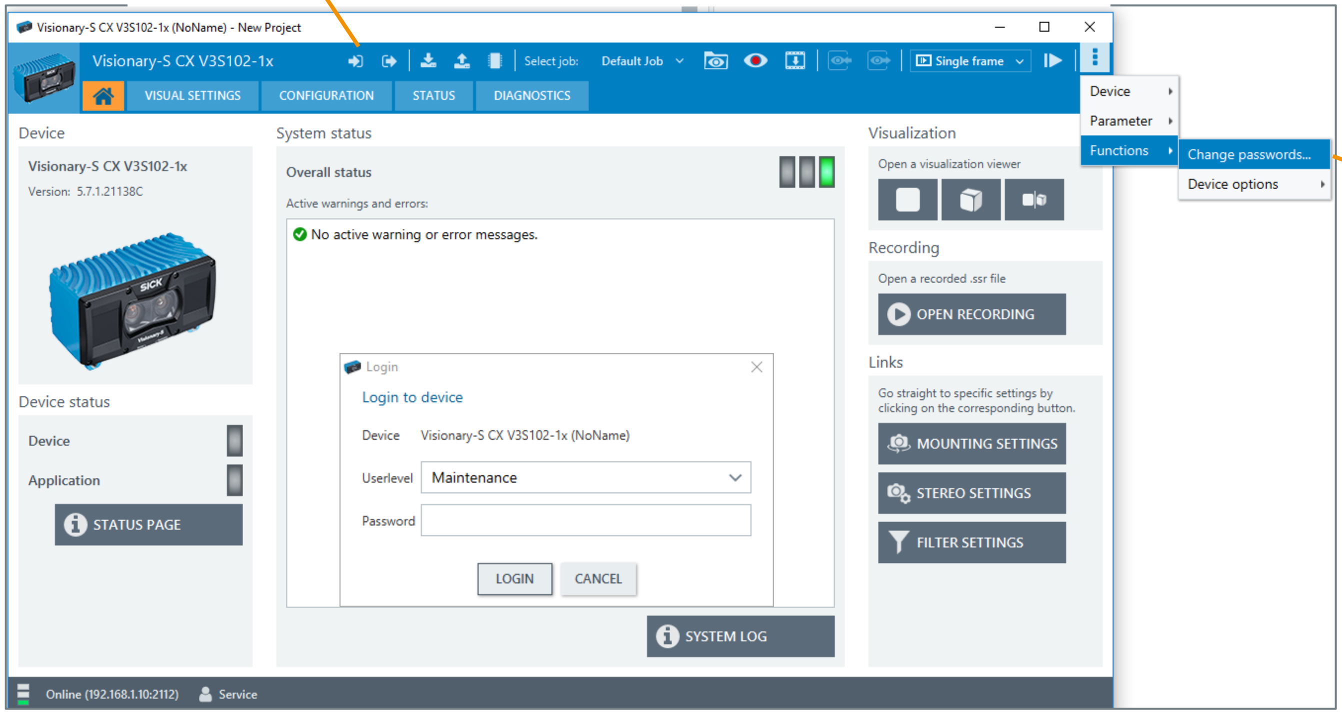


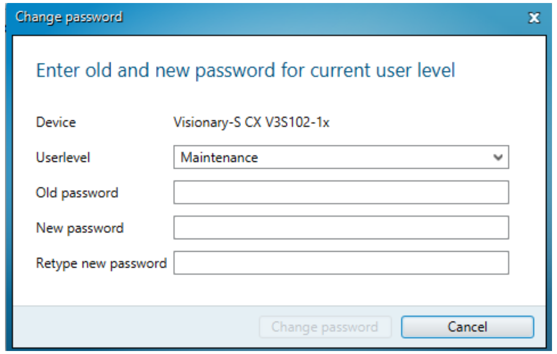
Visionary-S Device Page

Login

To change different parameters you have to log in on “Authorized Client” or “Service” user level with the corresponding password:

Userlevel	Password
Authorized Client	CLIENT
Service	CUST_SERV





It is possible to give personal passwords for different user levels. We urge you to do so for security reasons!

Visionary-S Device Page

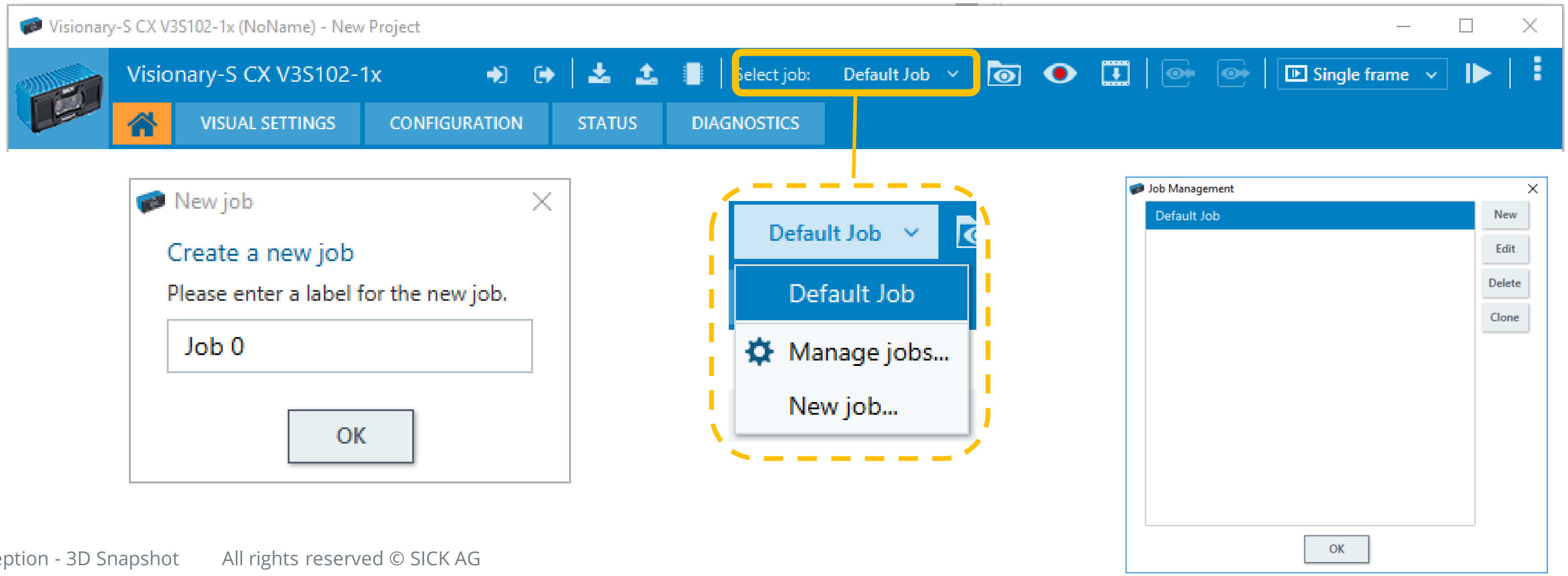
Managing jobs

The device configuration is stored to a Job

Multiple jobs on one device are supported

Job management via drop down menu

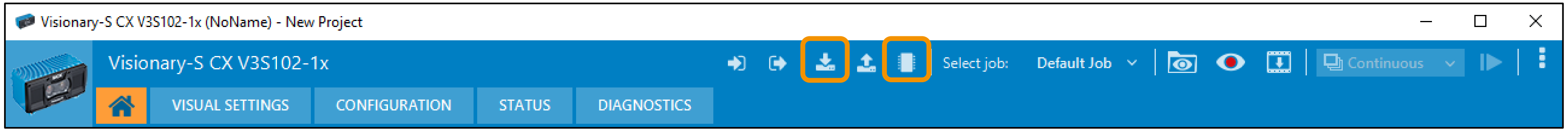
- A new job can be created
- A job can be cloned, e.g. to modify the parameters
- Jobs can be renamed



Visionary-S Device Page

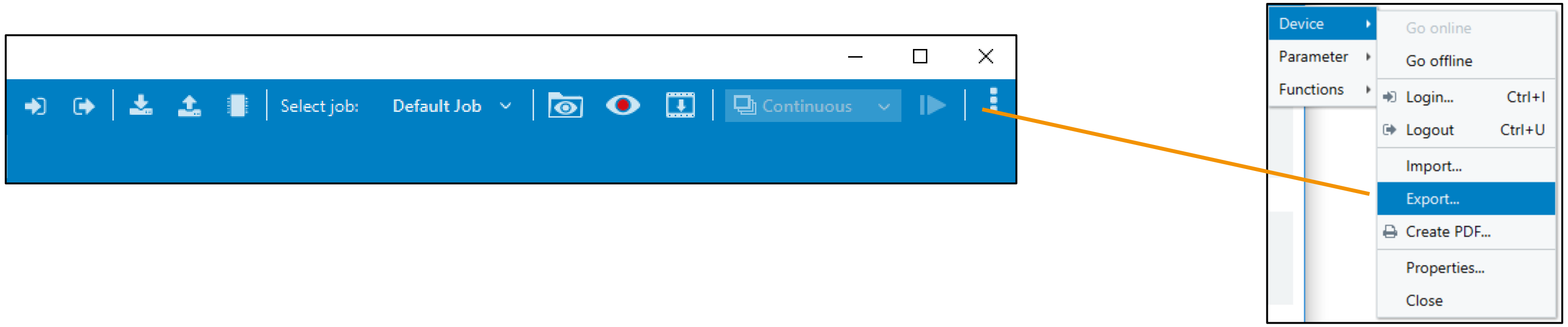
Save configuration

To save the configurations/jobs permanently, <u>first</u> press the „**write parameters to device**“ button, then in a second step press the „**save parameters permanently**“(otherwise the configurations are lost when switching off the device)



All device settings can be exported by the “Device” menu of SOPAS ET.

The settings can be stored for later use or to multiply the settings on different devices. Note that these settings contain all jobs.



Visionary-S Device Page

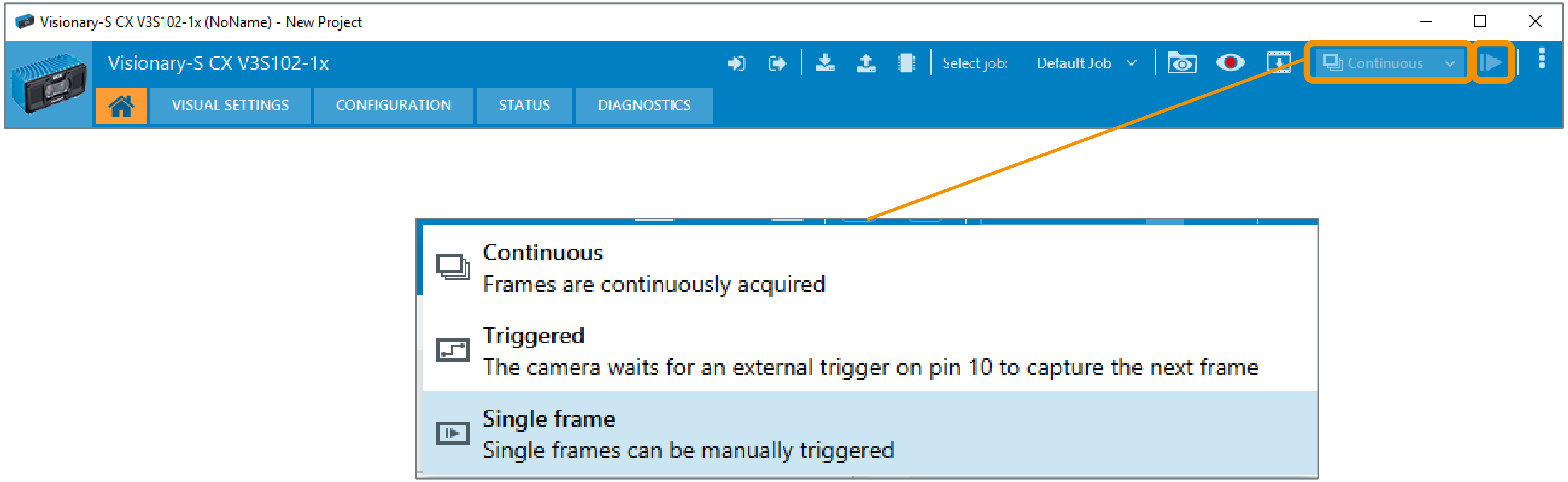
Acquisition mode

The camera can operate in three different acquisition modes:

Continuous (default): Frames are continuously acquired and visualized/streamed

Triggered: The camera waits for an external trigger on pin 10 to capture next frame.

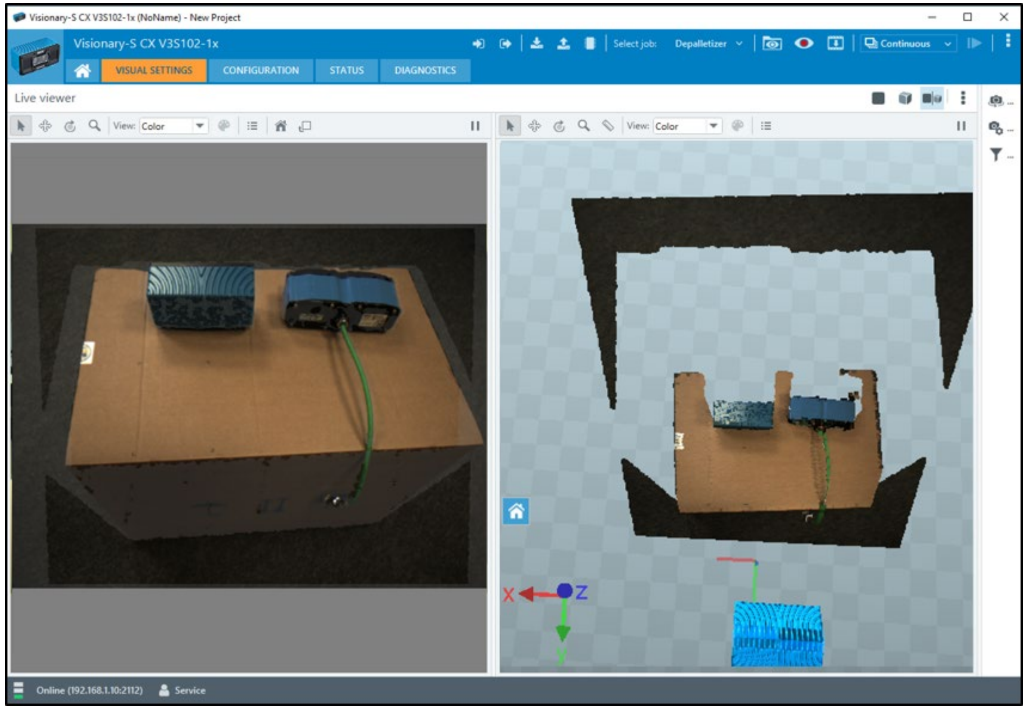
Single frame:
Single frames can be triggered by software command either by SOPAS „**trigger next frame**“ button or by sending a command e.g. API or telegram



Visionary-S Device Page

Record – Store ssr file on local disk

› Available in the device window toolbar

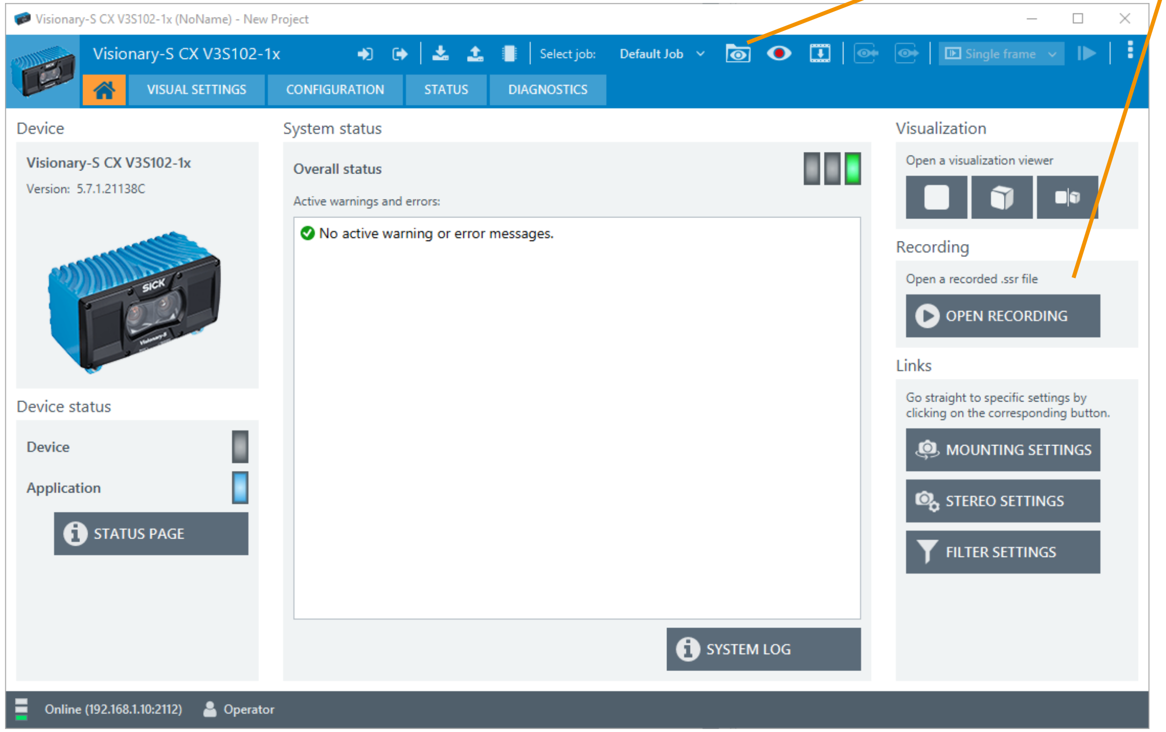


- Press *Start recording* button
- Active recording is shown by this symbol REC in the upper right corner and the Start recording button turns from dark to light blue
- Press again to stop recording
- File saving dialog opens automatically
- Select your directory, name the SSR file and save it
- The file size of a recording increases for about **35 MBytes per second**. Hence the file size can be very large and the recording is automatically stopped when the file size is about to exceed **2 GBytes**.
- For those large files saving might take some time. Please note that the live view is shown after pressing **Save** even if the file itself might not be completely saved yet.

Visionary-S Device Page

Replay - Load, *.ssr file from local disk

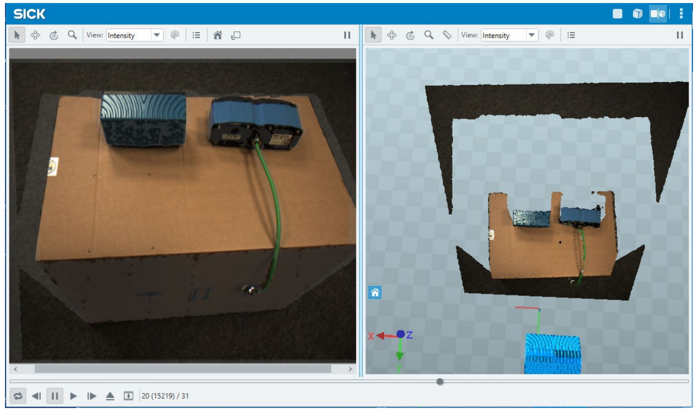
- Available in the device window toolbar
- Note that to increase the performance of the playback window the live viewer is automatically set to *Pause*



Press Open recording file either  in the device window toolbar or  on the homepage

Select your .SSR and choose open

A File Playback window opens and offers (nearly) the same options like a 2D or 3D viewer



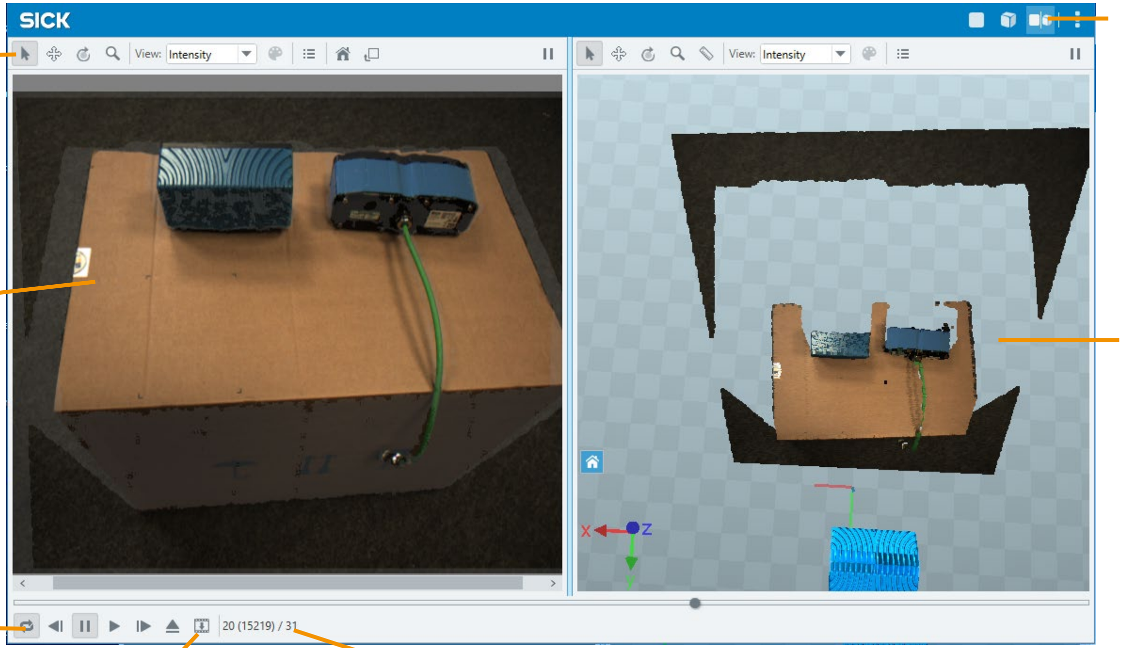
Visionary-S Device Page

Replay - File playback

Additional to the 2D or 3D features a playback bar is available

For the remaining symbols please look into the GUI configuration (page 9)

The File Playback window will always open in in 2D/3D split view. Of course you can switch to sole 2D or 3D



2D view

3D view

Replay mode
Single or in a loop

Save current frame as 3D point cloud (PCD or PLY) file

Play

Pause

Frame counter

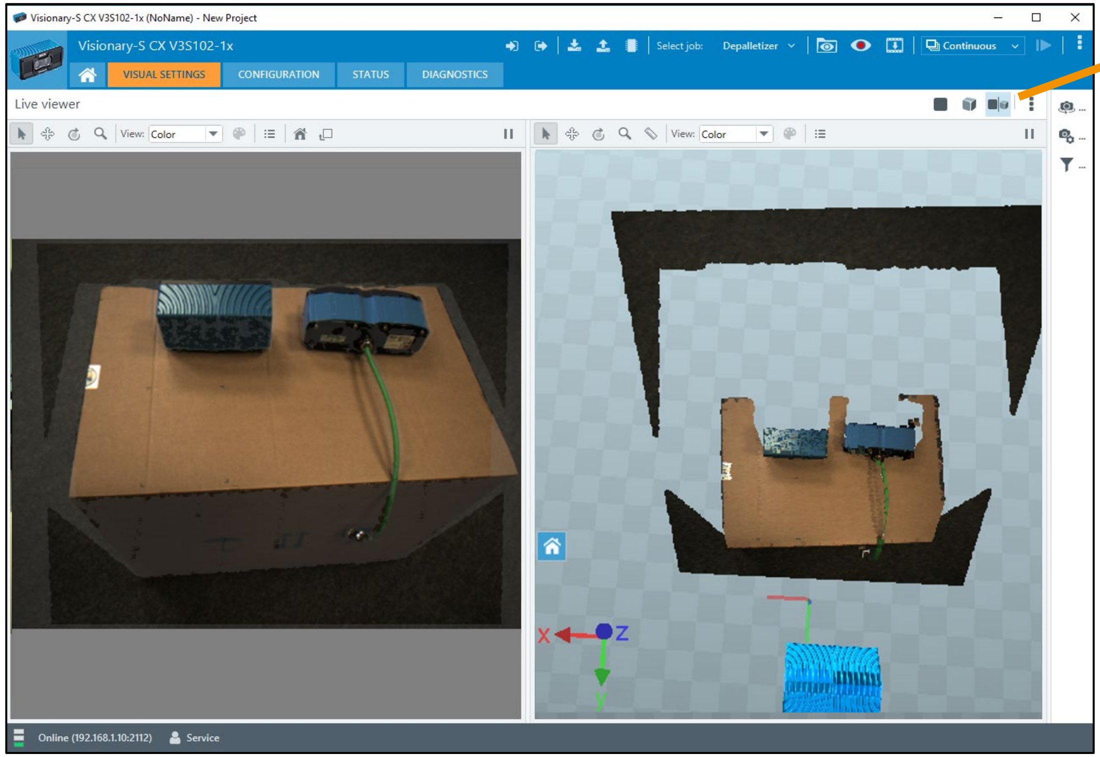
Single Step
(Forward or backward)

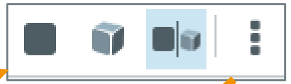
Open file (Open another SSR file)

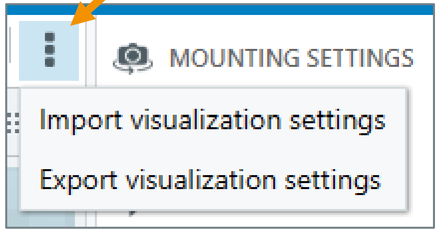
Visualization

Features

Use the SOPAS ET feature on the upper right corner to switch between the 2D -, 3D-viewer or both at the same time.







This scene shows a box where sensor heads of the Visionary-T and Visionary-B are placed. On the left you can see the 2D, on right the 3D point cloud view of this scene. Both views contain the same information

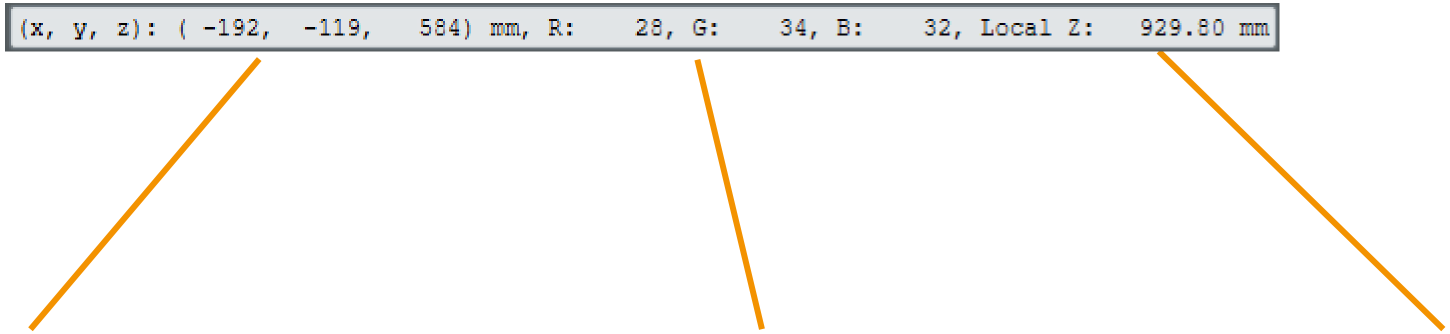
The refresh rate of the visualization depends on the computer’s performance

You can also export and import your visual settings (e.g. viewing angle and zoom)

Visualization

Measurement bar - overview

- Available in 2D/3D Viewer
- Use selection tool to enable the measurement feature
- You may use the pause mode to freeze the data
- The measurement bar is visible if the mouse pointer is in the surrounding of data points
- Hovering with the mouse pointer over a specific point gives additional information about the data



X,Y,Z in [mm]
Data in world coordinates. See device page mounting for more information

R,G,B parameters
Displays the color intensities for each pixel.

Perpendicular distance in [mm].
(only available in 2D viewer)

Visualization

2D Viewer

Selection tool. Hovering over the image shows the measurement bar

Move. Available by pressing shift key as well

Rotate. Available by pressing ctrl key as well

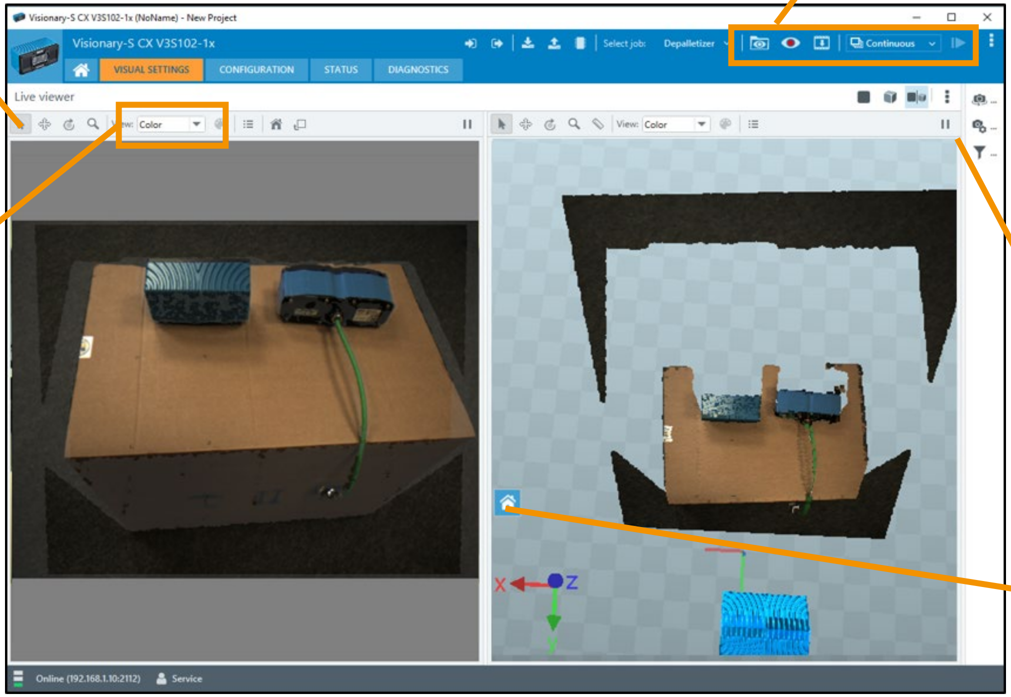
Zoom. Available by mouse wheel as well

Open a new window to play a SSR file stored on local disk

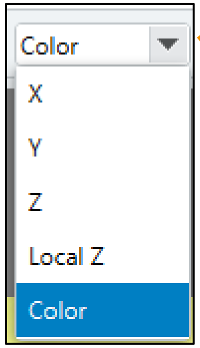
Record data in a SSR file on local disk

Save current frame as 3D point cloud (PCD or PLY) file

Trigger next frame (not active in continuous mode)



View selection



Pause Freezes current image

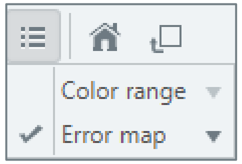
Home Reset zoom and position settings

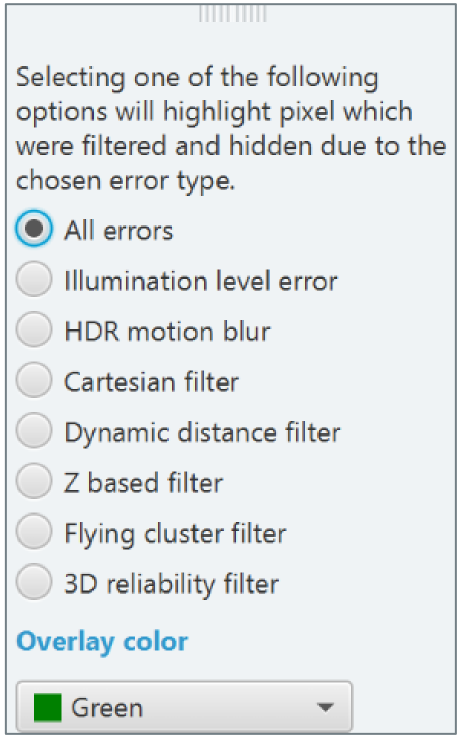
Visualization

2D Viewer – Error map

Various circumstances can lead to a loss in data values. The error map visualization helps to find the reason for this loss by highlighting omitted pixels in a defined color (default: green):

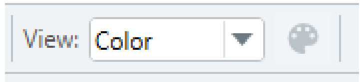
- **All errors**: all pixel values which were omitted from the final frame are highlighted
- **Illumination level error**: highlights pixels with illumination problem. Change the 3D exposure to improve the situation.
- **HDR motion blur**: highlights pixels where depth values changed during HDR/HQM frame acquisition. Reduce dynamics in target scene or increase reliability filter setting.
- **Cartesian filter**: highlights pixels which are omitted due to the settings of this filter. Optimize filter settings if necessary.
- **Dynamic mode**: highlights pixels which are omitted due to the settings of this filter. Optimize filter settings if necessary.
- **Z based filter**: highlights pixels which are omitted due to the settings of this filter. Optimize filter settings if necessary.
- **3D reliability filter**: highlights pixels which are omitted due to the settings of this filter. Optimize filter settings if necessary.



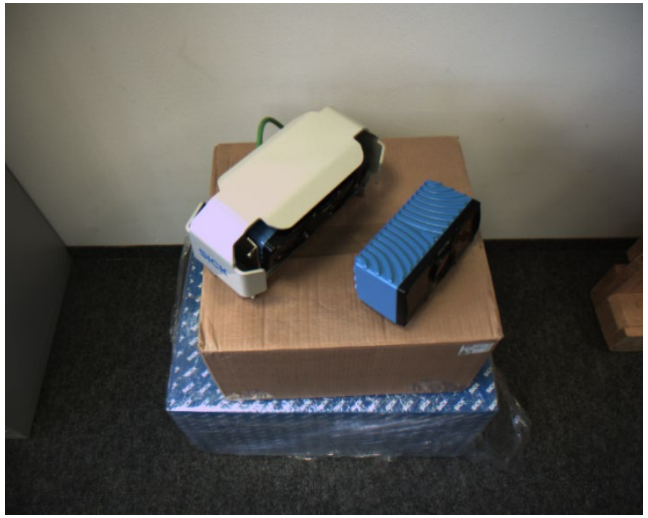


Visualization

2D Viewer – Visualization settings

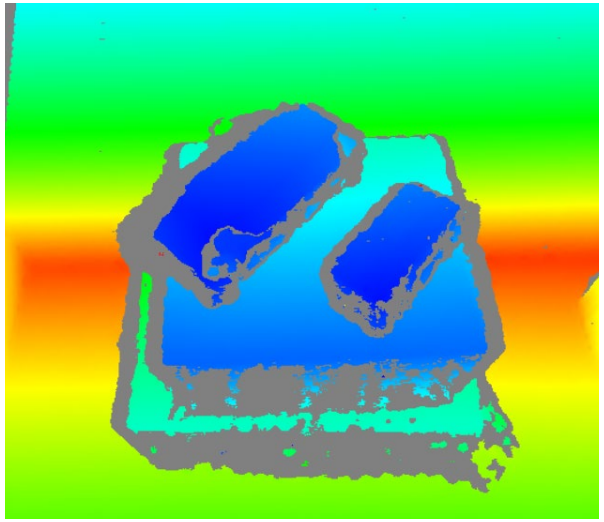


Color



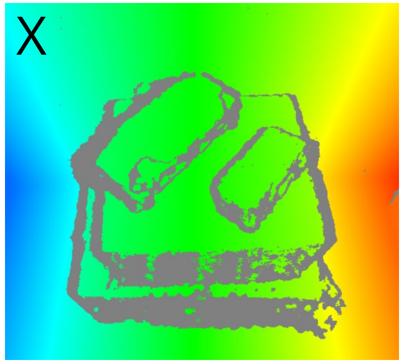
Perfect for visual identification of the scene.

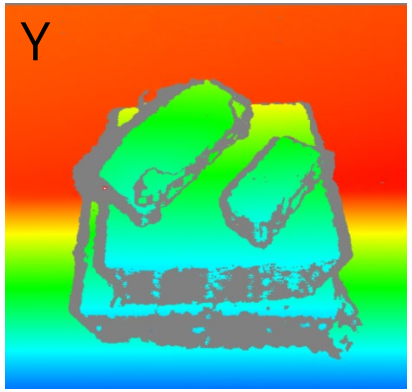
Local Z

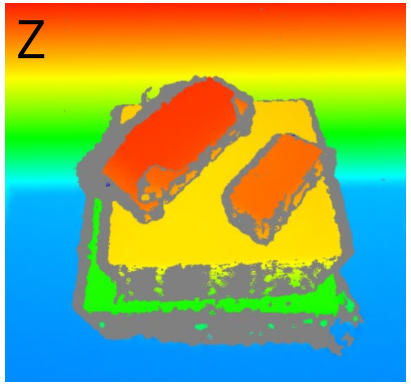


Depicts distance to camera. Grayed pixels do not posses depth information (see error map).

Cartesian visualization/ Heightmaps





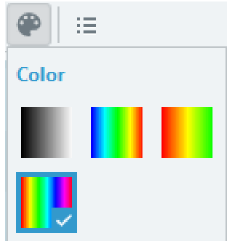


X, Y and Z visualize position of pixel in the world coordinate system.

Visualization

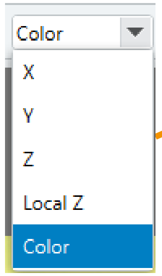
3D viewer

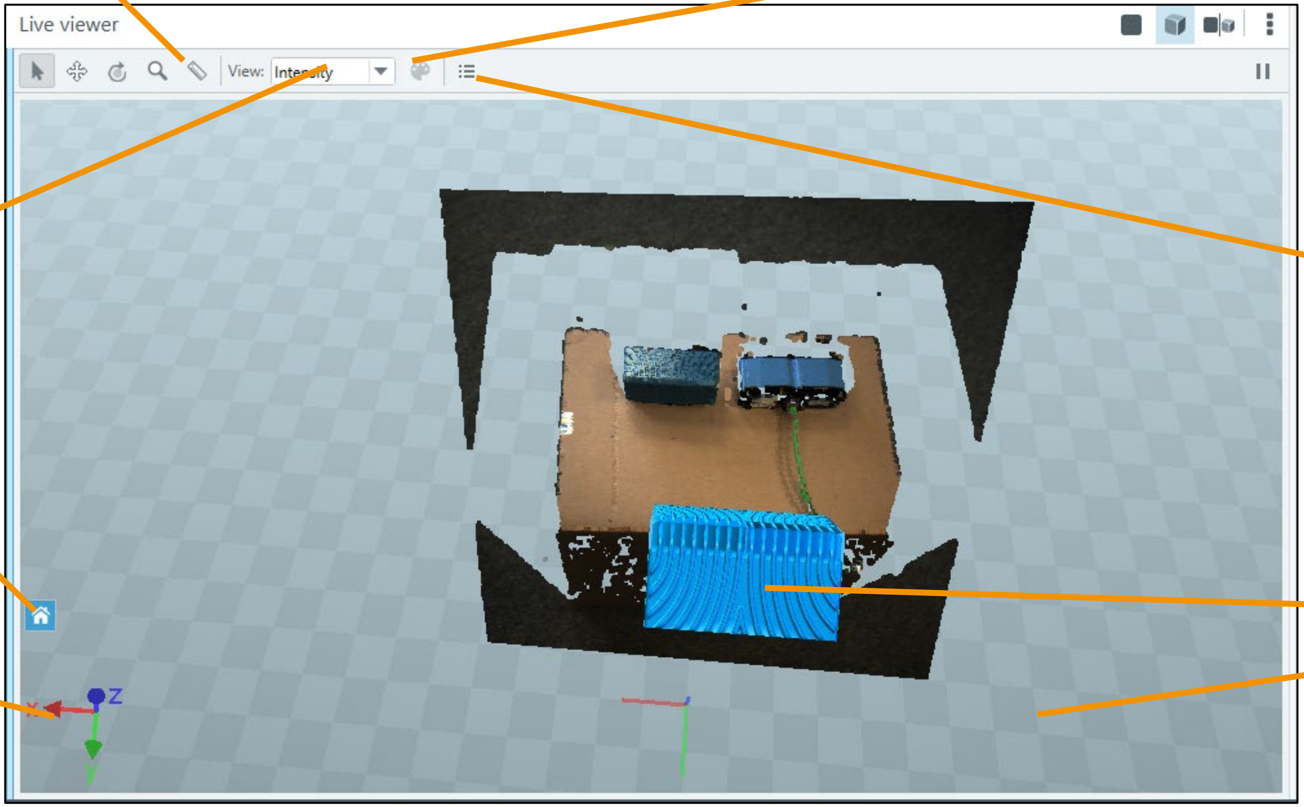
Ruler. Measure the distance between two points.

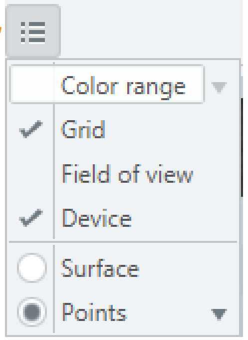


Coloring will range from:

- Black to white
- Blue to red
- Red to green
- Red to purple (selected)







Reset perspective

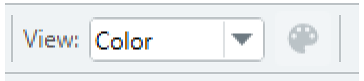
X, Y, Z orientation of the current perspective

Device

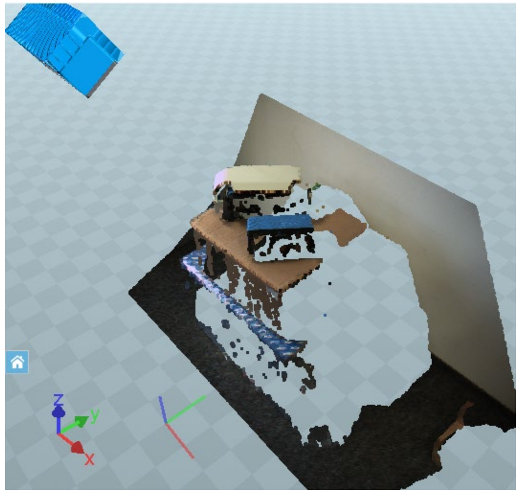
Grid

Visualization

3D viewer - Visualization settings

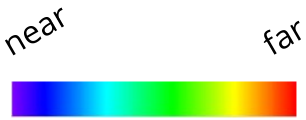


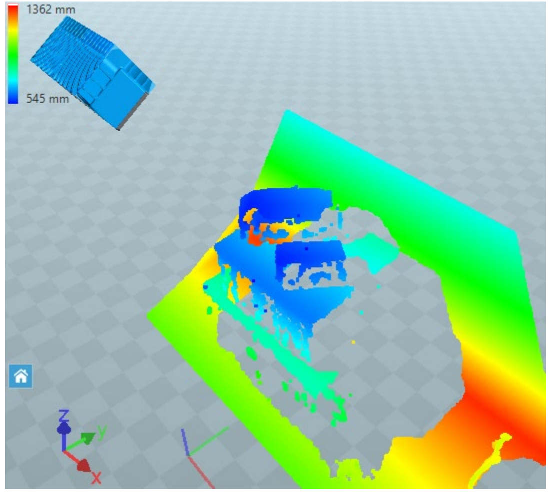
Color



Perfect for visual identification of the scene.

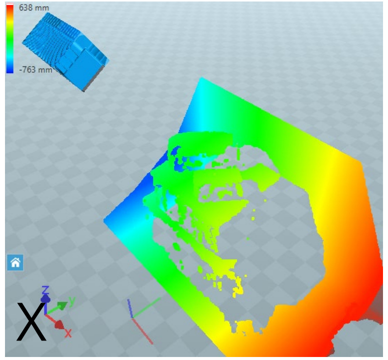
Local Z

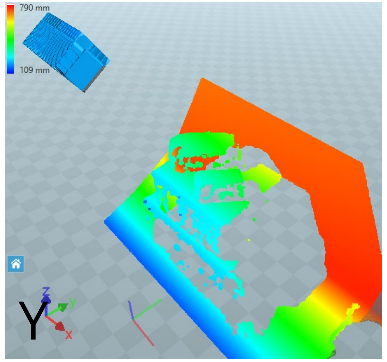


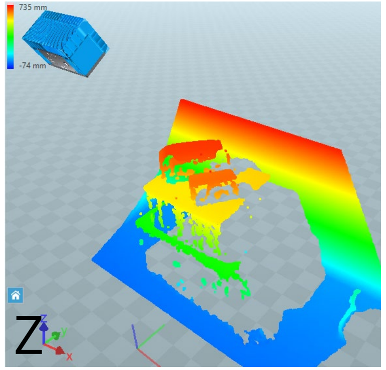


Depicts distance to camera

Cartesian visualization







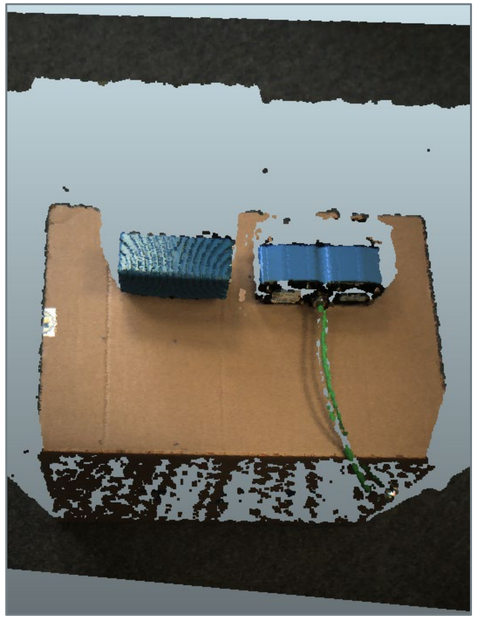
X, Y and Z visualize position of pixel in the world coordinate system.

Visualization

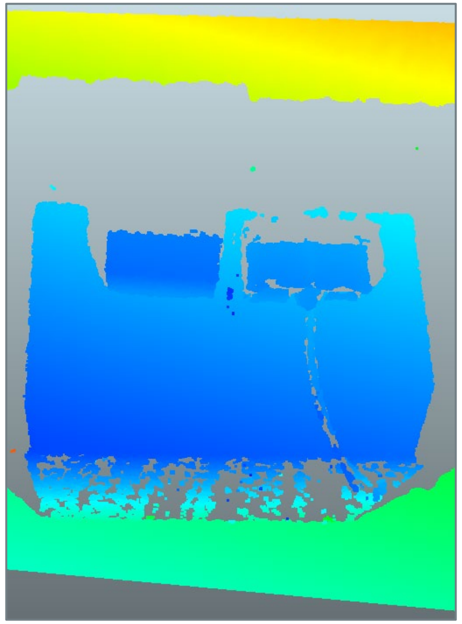
3D viewer - Visualization settings

- Available for live camera view or saved recordings
- The 3D visualization always renders the point cloud according to the given mounting settings (see settings menu)
- Use the “View” drop down menu to choose different kind of data for the visualization

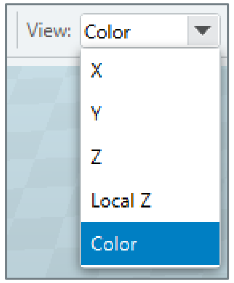
Example:



The data of the color map is used for coloring each value of the point cloud



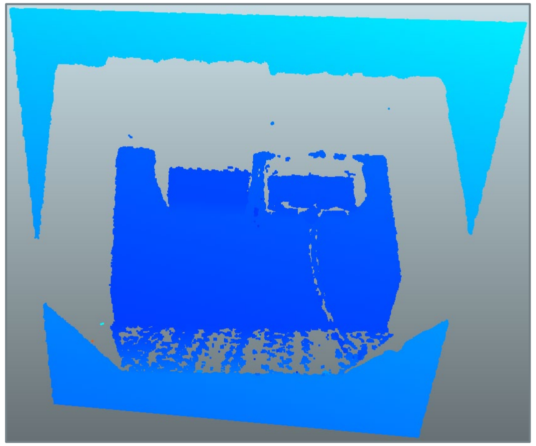
The data of the local Z map is used for coloring each value of the point cloud

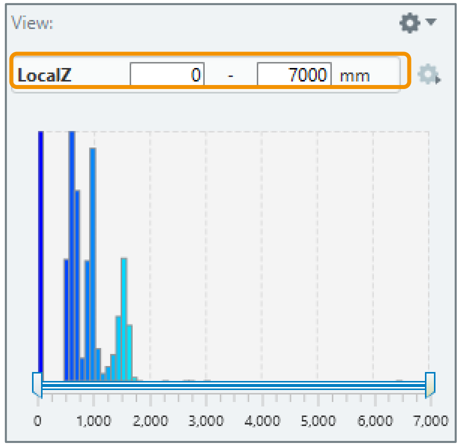


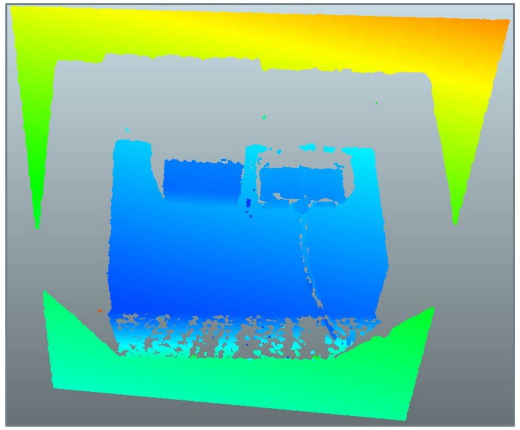
Visualization

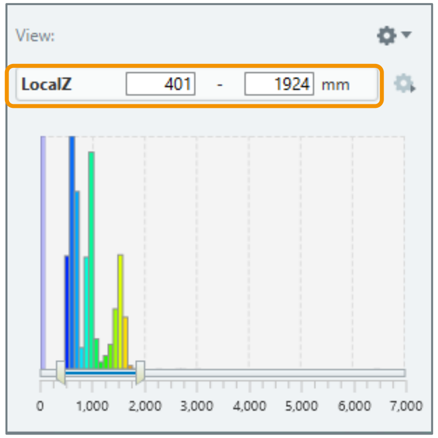
3D viewer - Visualization settings

- The color range settings allow optimizing the coloring
- Color range settings are available for each data source and all of them are active at the same time
- Points out of range will be grayed out
- A histogram is shown to support data analysis and easily selecting the range of interest
- The limits of the color range settings will be calculated during opening the 3D viewer. If the scene changes completely, close and open the 3D viewer to recalculate the limits
- Example (Local Z coloring): adjustment of the color range accordingly (oriented to the diagram)





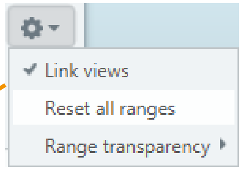




Visualization

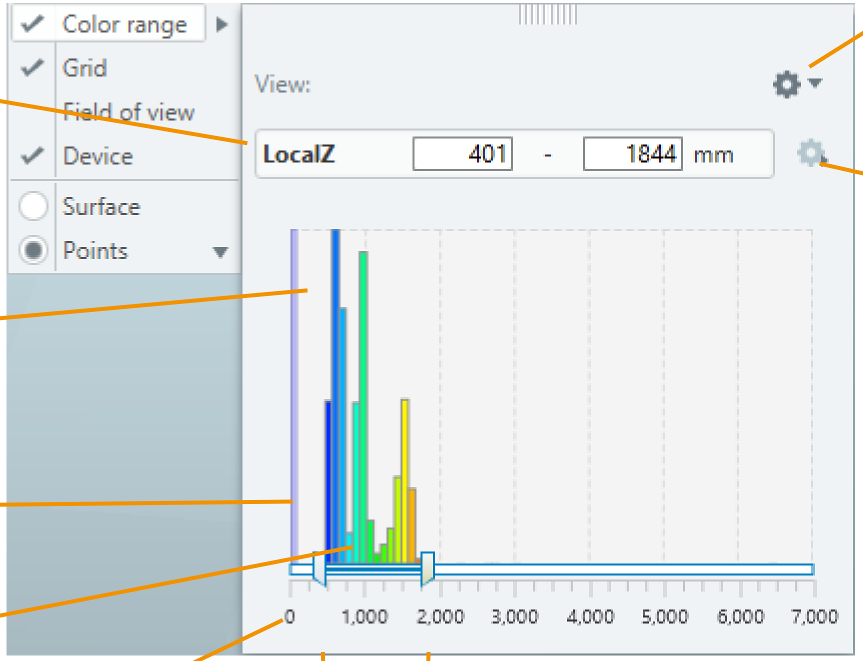
3D viewer - Visualization settings

The range settings are available for 2D and 3D viewer



Reset.
All range settings

Source selection.
Use for histogram only



Reset.
Selected color settings

Histogram.
Graphical representation of the distribution of data

Data out of range.
Grayed out

Coloring will be optimized for the selected range

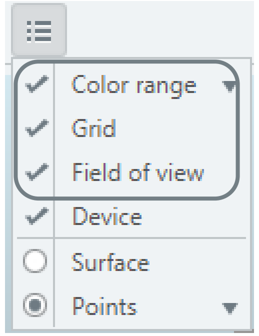
Scaling. Depends on selected source

Slider.
Used for data selection

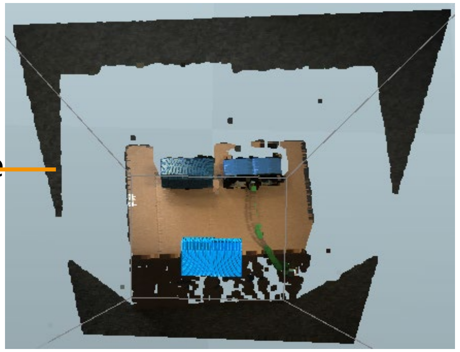
Visualization

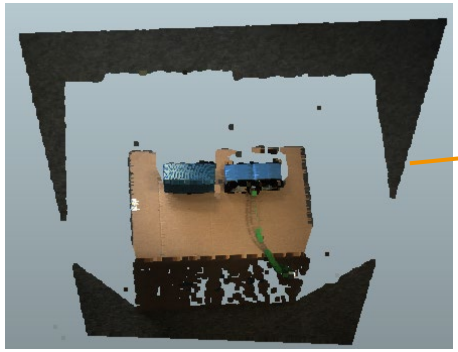
3D viewer - Visualization settings

- For easy orientation a grid and a device model can be visualized as an overlay



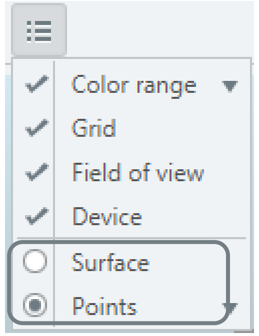
With grid, field of view and device mode

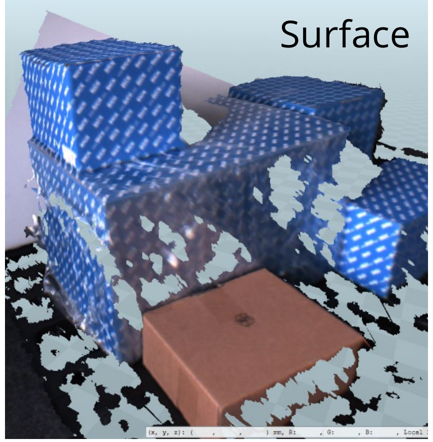


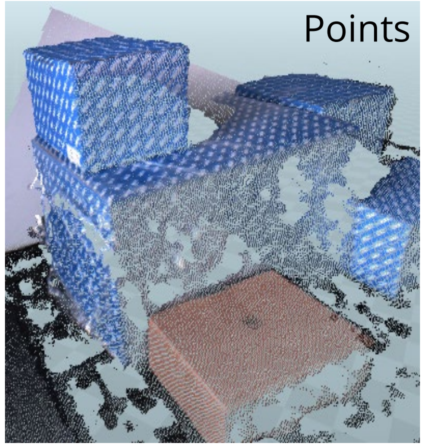


Without grid, field of view and device model

- Choose *Surface* or *Points* to visualize the scene as required







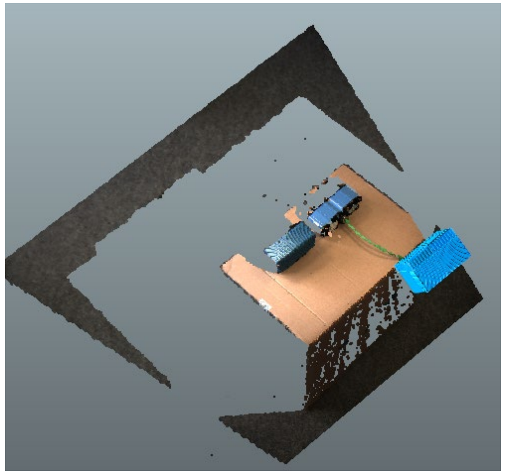
Visualization

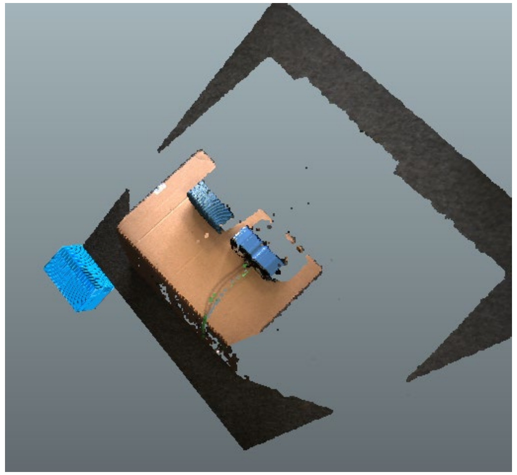
3D viewer - Navigation

You can analyze the scene with different points of view

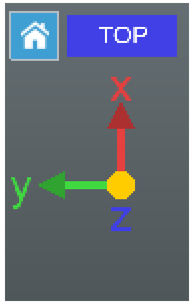
Navigate via mouse and keyboard

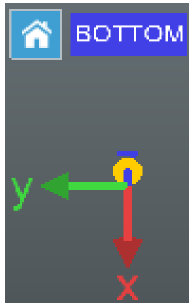
- Use mouse wheel to zoom in and out
- Use *ctrl* and *shift* key to change tool

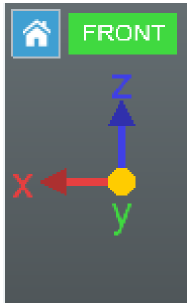




- Click on the home symbol to go to standard perspective behind the camera
- Click on the arrow tips to select a pre-defined perspective





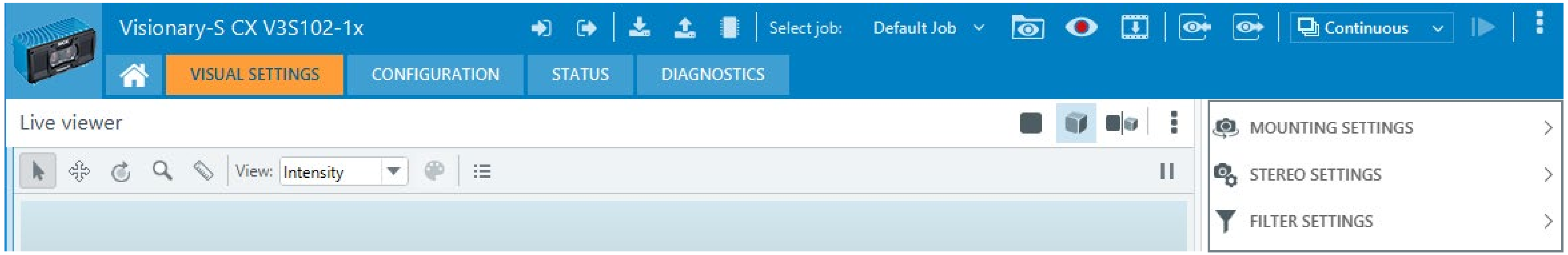


These labels (TOP, BOTTOM, FRONT) are only for your orientation in the 3D visualization and do not correspond (necessarily) to the orientation of your device. The colors of the arrows are kept constant for easy orientation.

Settings Menu

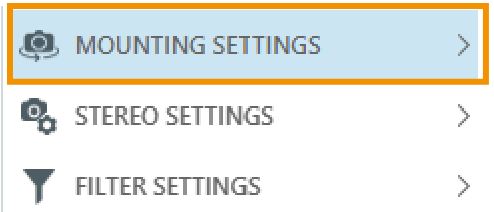
Overview

- The next slides will guide you through the Settings-menu on the right
- To change the settings, make sure you are logged in with the appropriate user level (see SOPAS Start Page)
- It's also possible to store the setting changes inside of the device, e.g. with 



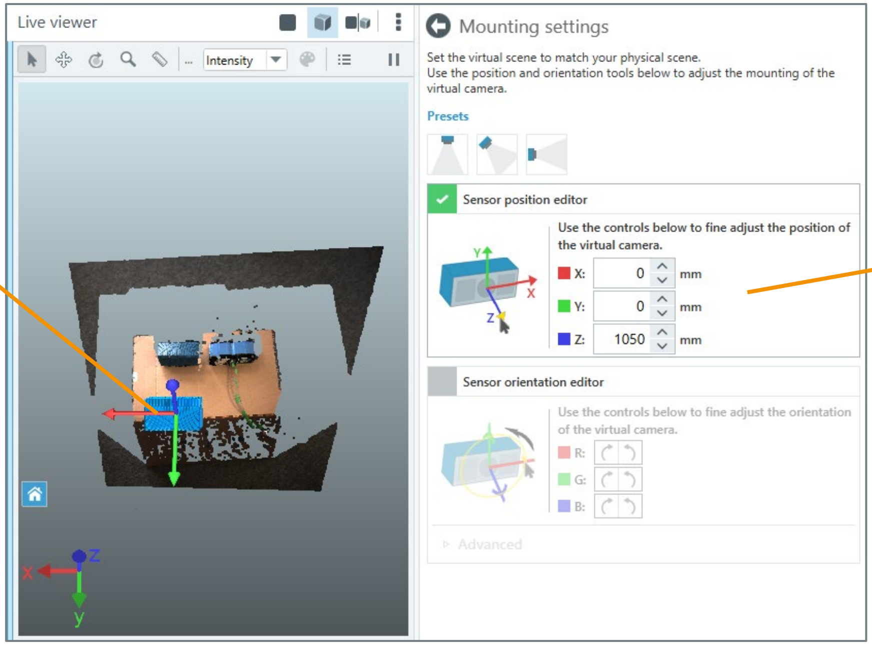
Settings Menu

Mounting settings



- The mounting settings allow the user to set the sensor pose in world coordinates
  - There are six parameters, three for each sensor position and orientation
- The mounting settings are stored inside of the device
- The values are used for visualization and are available via the API for further calculations

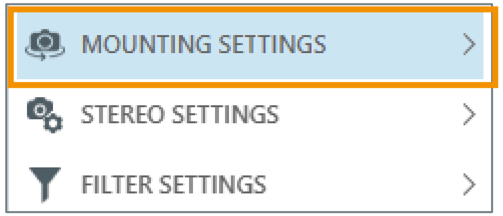
3D View.
The device model overlay visualizes the given position and rotation information



Mounting information.
Changes are used immediately in the 3D Viewer

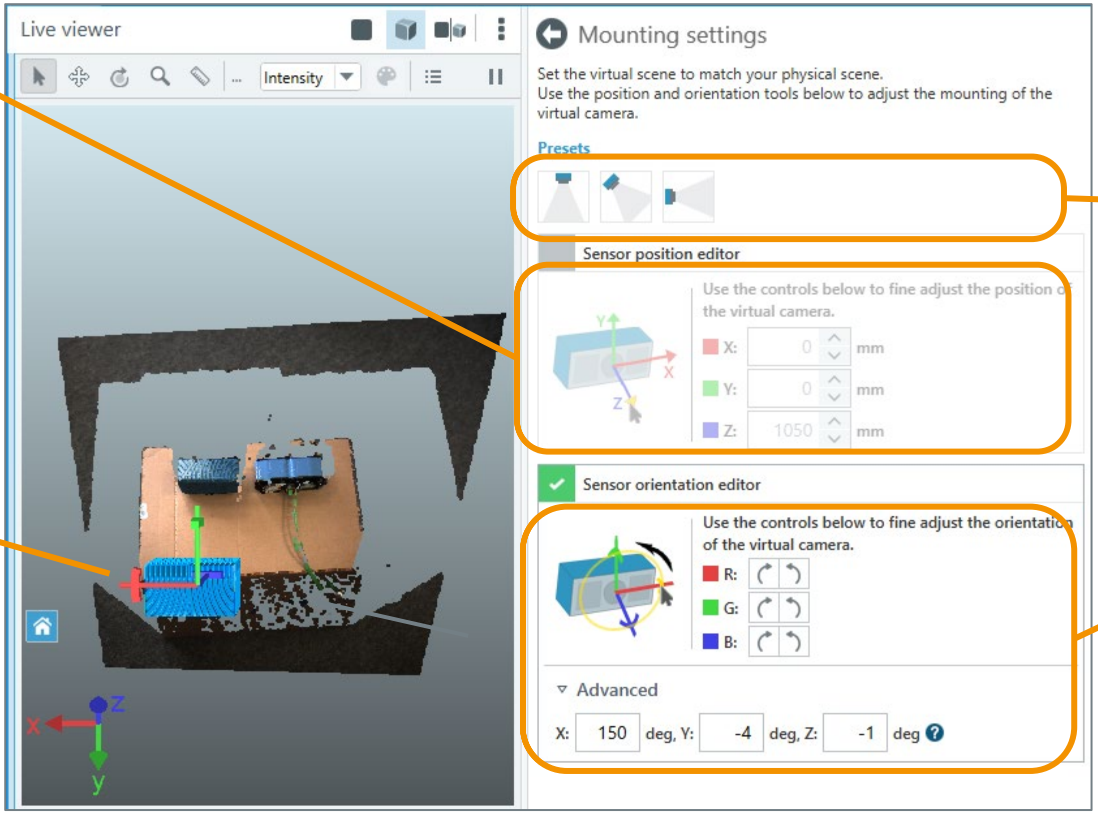
Settings Menu

Mounting settings



Sensor position: You may enter the values or use the buttons

You can move and rotate the virtual camera in the 3D scene.
Pick the handles by using the selection tool



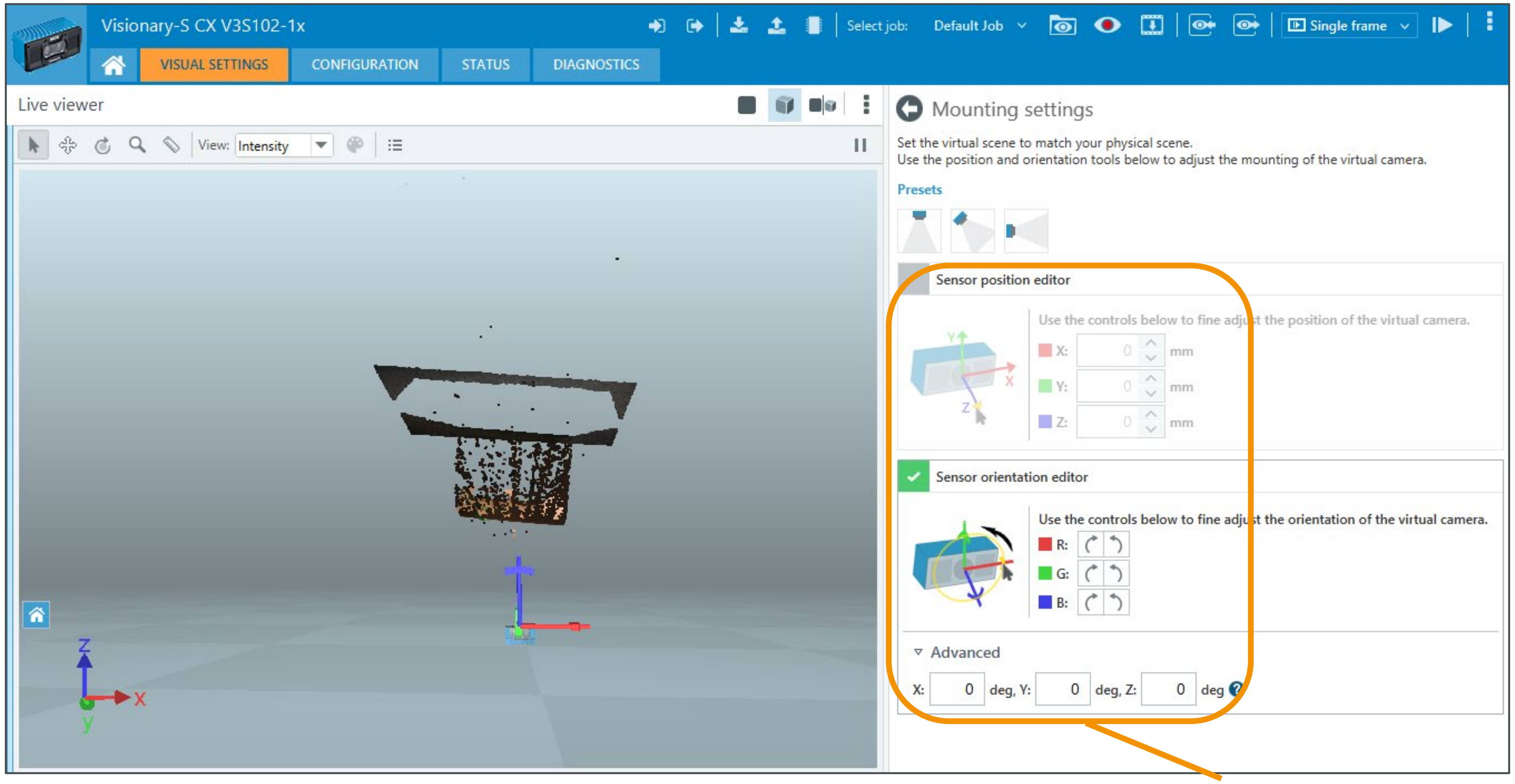
Pressing the presets will set the most common values for the three depicted mounting scenarios

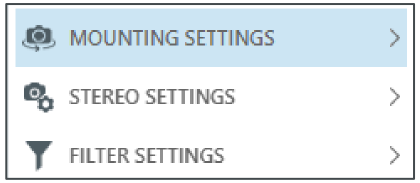
Sensor orientation.
You can use the rotation buttons or the advanced menu to enter numeric values

Settings Menu

Mounting settings

In the scene illustrated here, the physical sensor is mounted 1,015m above the ground in reality, but visualization settings are different and have to be adjusted.





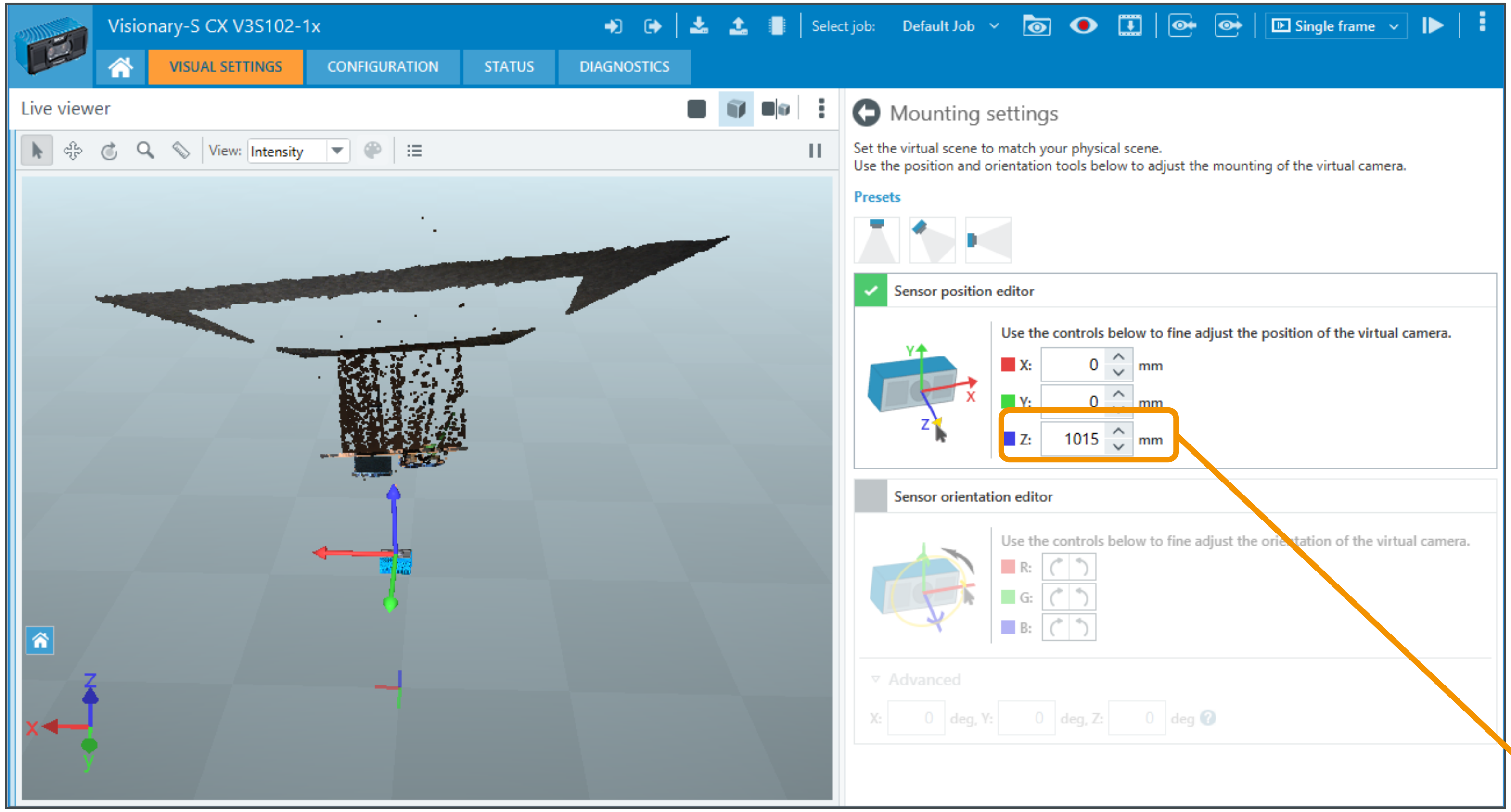

Real mounting

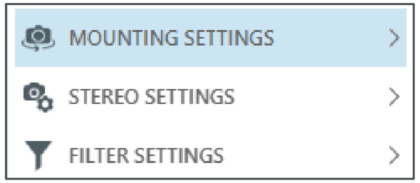
Default position and orientation: all values are set to zero.
That means: “Sensor is placed on the floor/grid and looks to the ceiling.”

Settings Menu

Mounting settings

In reality the sensor X angle is tilted 149° and the Y angle is tilted -5° and looks to the corner





Real mounting

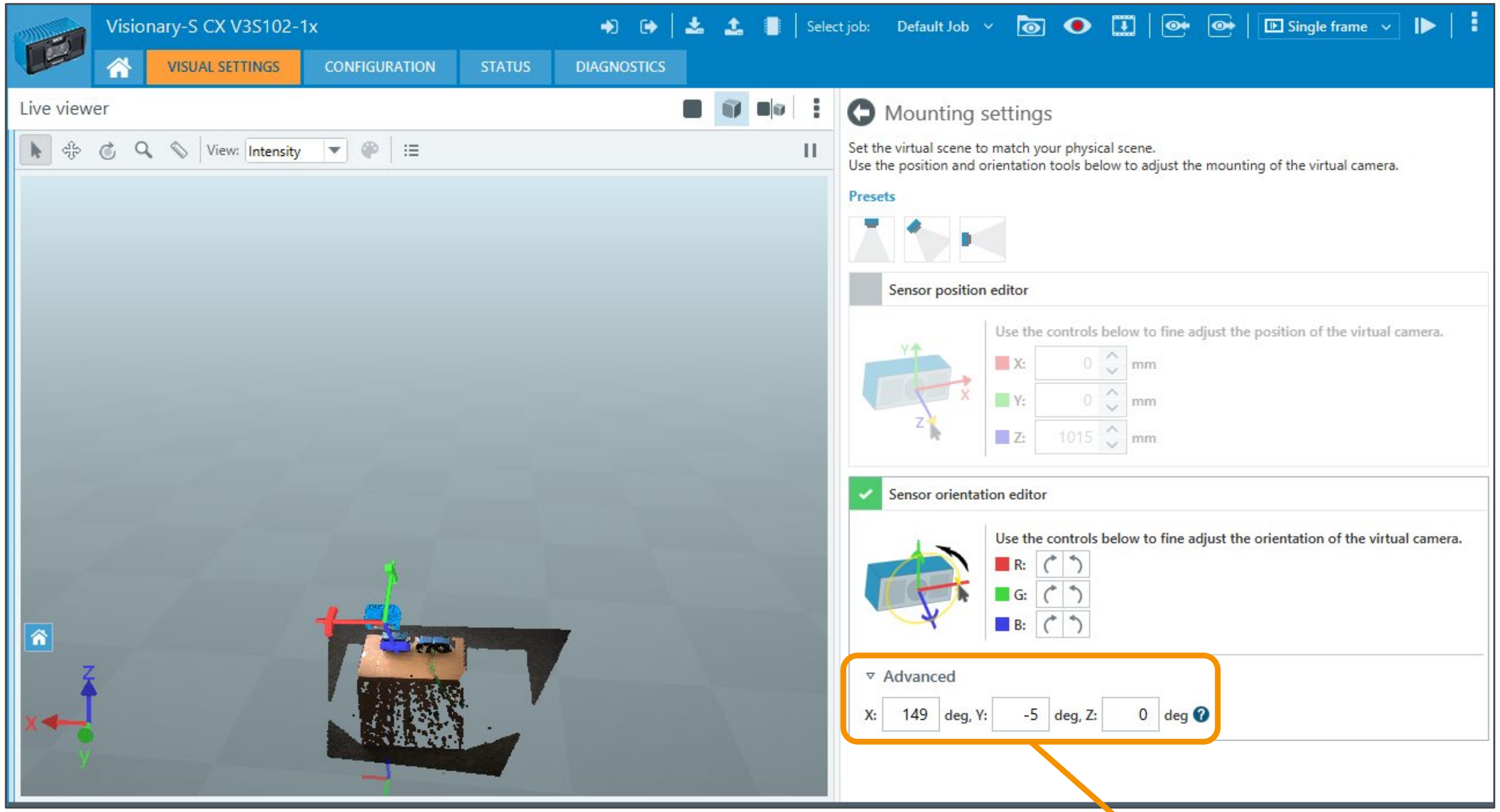
Z-Position changed to 1015mm.
That means: “Sensor is mounted 1015mm above the floor and looks normal to the grid (ceiling)”

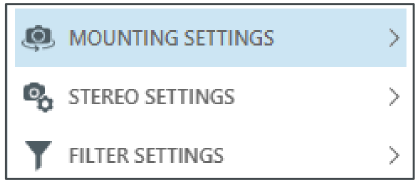
Settings Menu

Mounting settings

SICK Sensor Intelligence.

Now the settings fit to the real scene





Real mounting

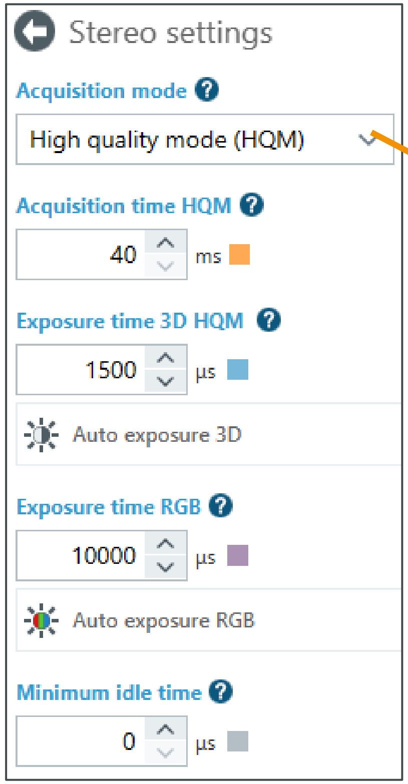
X-Angle changed to 149 degrees.
Y-Angle changed to -5 degrees

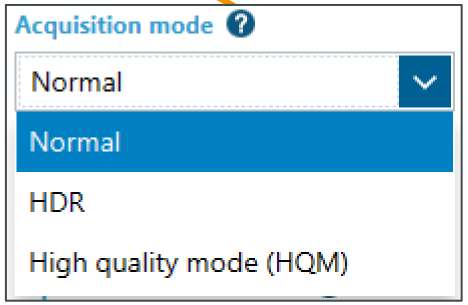
Settings Menu

Stereo Settings – acquisition mode

The acquisition mode includes three different options for choice

- **Normal mode:** all stereo settings can be manually configured, e.g. exposure (a.k.a integration) time. Max. frame rate = 30 FPS
- **HDR mode:** in this mode two different exposure times are used for every single frame. Max. frame rate = 25 FPS
  - Especially useful if dark and shiny objects are present in one scene. Overexposed pixels of one exposure are combined with underexposed pixels from a second exposure with another integration time → optimal exposure of the scene
- **High quality mode:** is used, if higher repeatability of depth values is required. Max. frame rate = 25 FPS





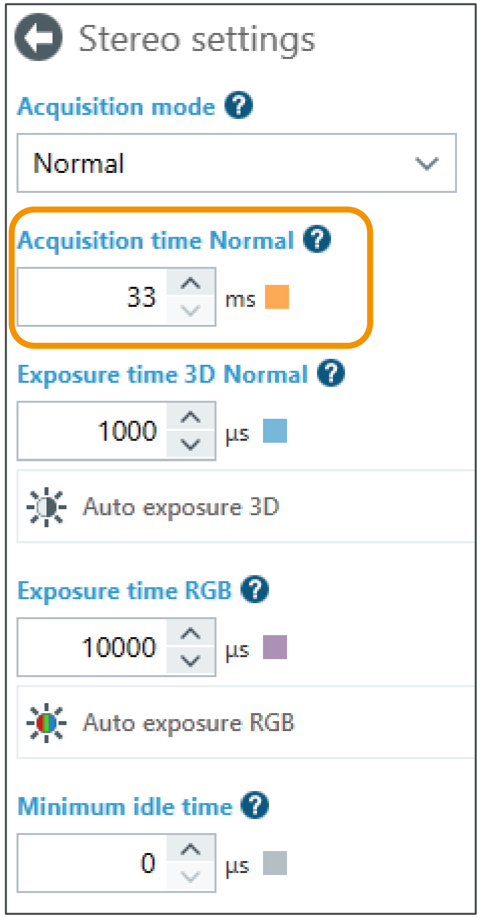
SICK APP SPACE

- **Visionary-S AP**:
- Image.Provider.Camera.V3SXX2_1Config.setAcquisitionMode(config, “**Normal**")
- Image.Provider.Camera.V3SXX2_1Config.setAcquisitionMode(config, "**HDR**")
- Image.Provider.Camera.V3SXX2_1Config.setAcquisitionMode(config, “**HQM**")

Settings Menu

Stereo settings – acquisition time

- The acquisition time determines the duration for a complete frame acquisition
- The time setting has to be at least 33 ms
  (Visionary-S CX max frame rate = 30 FPS)
- Any higher acquisition time results in a lower frame rate



SICK APP SPACE

- **Visionary-S AP (max frame rate = 50 FPS = 20ms)**:
- Image.Provider.Camera.V3SXX2_1Config.setFramePeriod(config, **20**)

Settings Menu

Stereo settings – Exposure time 3D

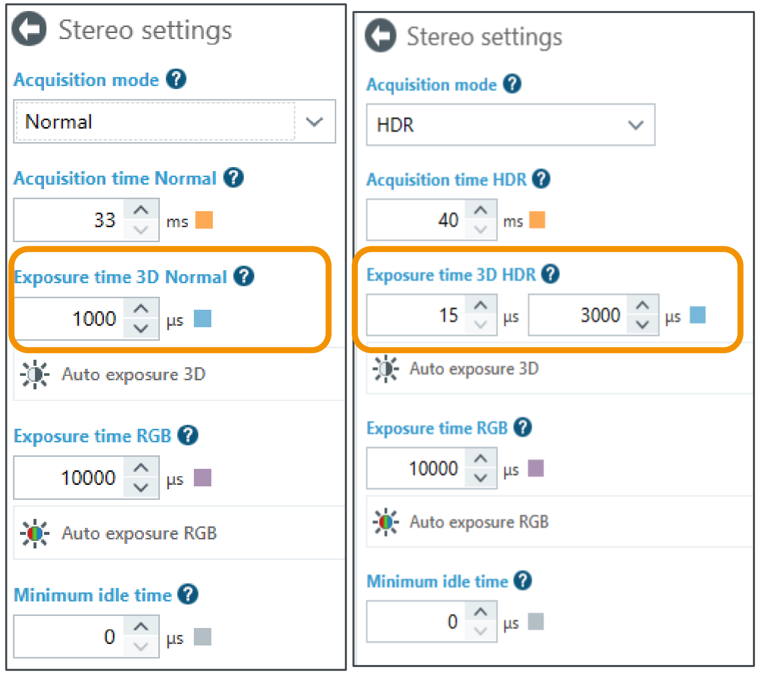
Set the 3D exposure time that fits your application the best.

- **The default setting is 1000 µs**
- **The range is 15 µs up to 9323 µs**

Note, higher values are advantageous if darker objects are of the main interest. However, be aware that setting higher values can lead more easily to saturated values for objects with high remission and to larger blur effects for dynamic scenes. Integration times above 5ms combined with higher frame rates (>15 FPS) can lead to temperature errors

HDR Mode:

- Set the low and high exposure times, which fits your application needs.
- Wider spread exposure times result in higher dynamic range
- Use a moderate to high setting of the reliability filter for optimal HDR performance

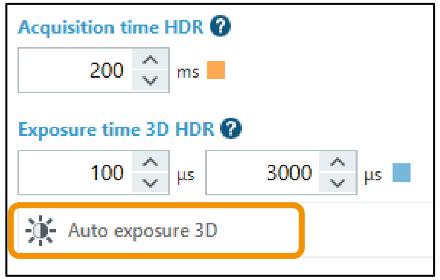


SICK APP SPACE

- **Visionary-S AP**:
- *Normal:* Image.Provider.Camera.V3SXX2_1Config.setStereoIntegrationTime(config, **1000**)
- *HDR short:* Image.Provider.Camera.V3SXX2_1Config.setStereoIntegrationTime(config, **15**)
  HDR long: Image.Provider.Camera.V3SXX2_1Config.setStereoSecondIntegrationTime(config, **3000**)
- *HQM:* setStereoIntegrationTime(config, **1000**) **&** setStereoSecondIntegrationTime(config, **1000**)

Settings Menu

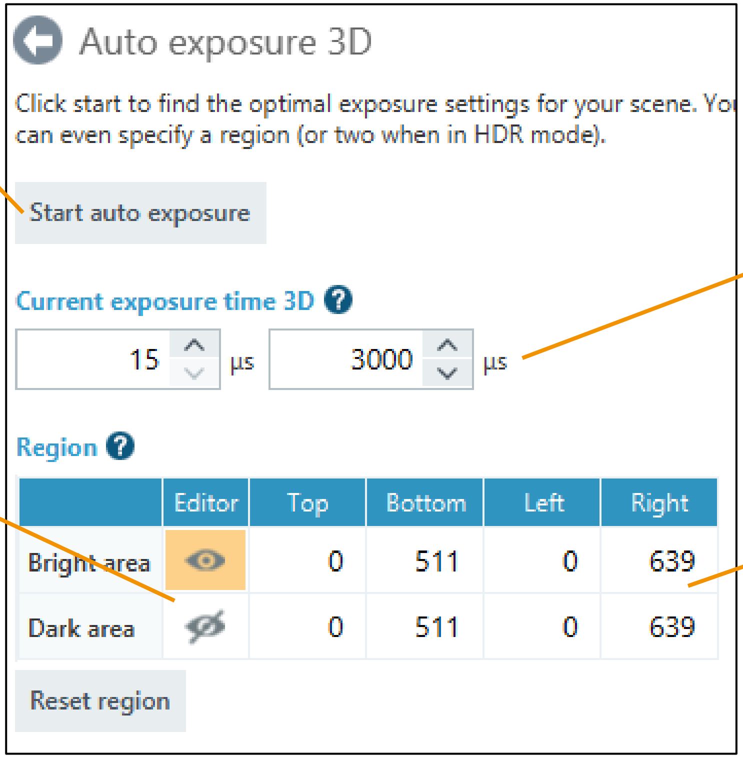
Stereo settings – Auto Exposure 3D



Use the auto exposure tool to automatically determine the optimal exposure settings

Click start to run the auto exposure on the defined region (default: full field of view)

Click on the eye symbol to define the region with the editor tool in the live view



The current exposure time(s) can be seen/set here

By typing in the pixel values you can specify the region manually

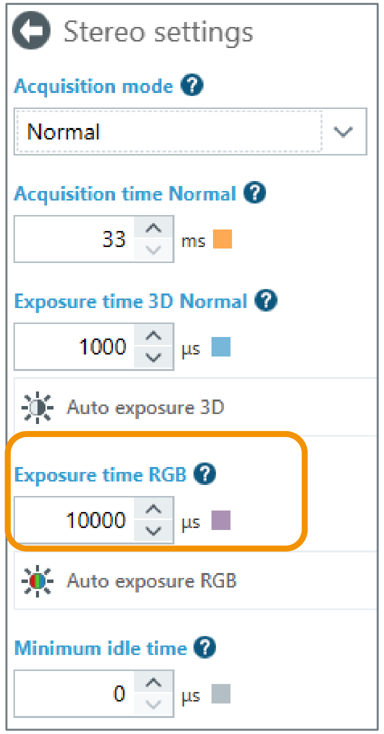
Settings Menu

Stereo settings – Exposure time RGB

Set the exposure time RGB to optimize the brightness of the RGB image according to the ambient light. The default setting is 1000µs

The range is 15 µs up to 1.000.000 µs

Note, higher values are advantageous if darker objects are of the main interest.
However be aware that setting higher values can easily lead to both saturated values for objects with high remission and strong blur effects for dynamic scenes



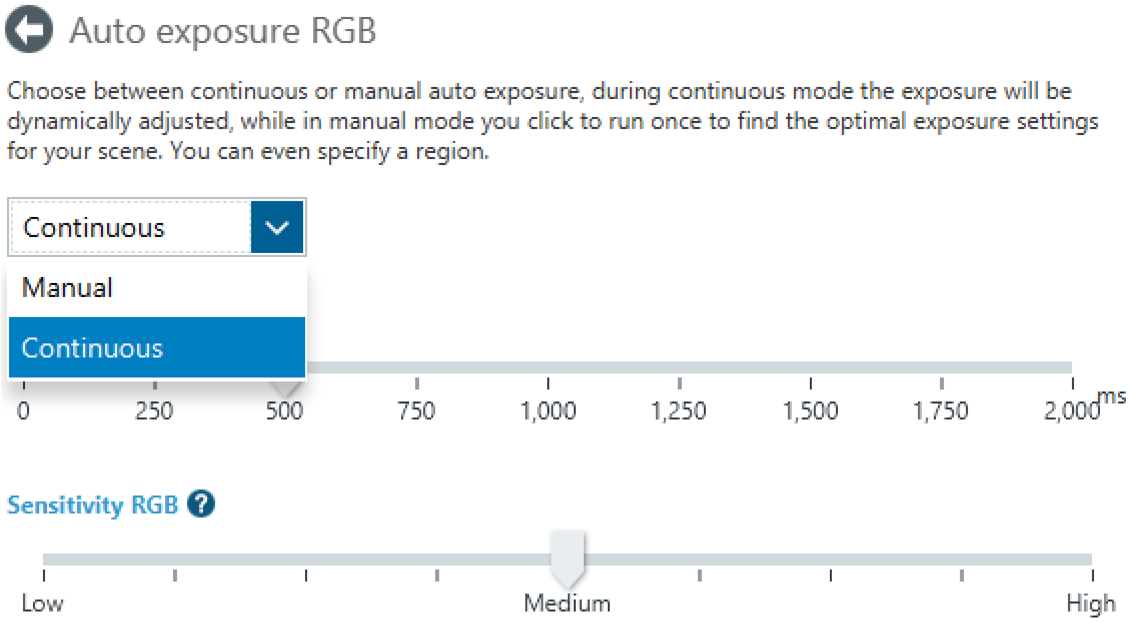
- **Visionary-S AP:**
- Image.Provider.Camera.V3SXX2_1Config.setColorIntegrationTime(config, **10000**)

Settings Menu

Stereo settings – Auto Exposure RGB

Use the auto exposure tool to automatically determine the optimal exposure settings

You can select between manual or continuous mode



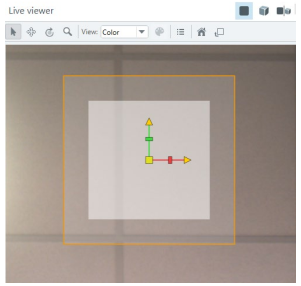
Settings Menu

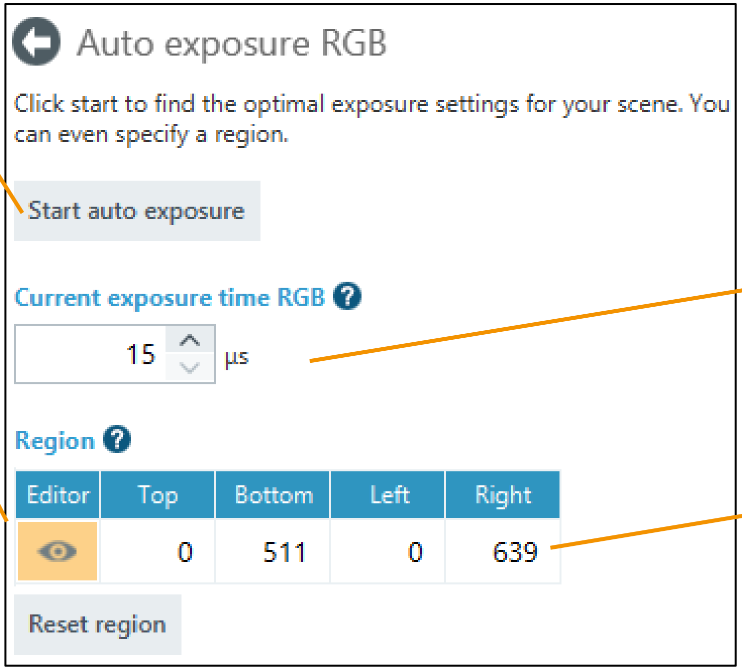
Stereo settings – Auto Exposure RGB – Manual mode

Use the auto exposure **in manual mode** to automatically determine the optimal exposure settings when needed

Click start to run the auto exposure on the defined region (default: full field of view)

Use the editor tool in the live view to define the region





The current exposure time(s) can be seen/set here

By typing in the pixel values you can specify the region manually

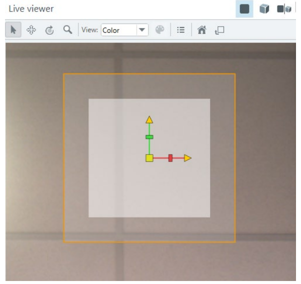
Settings Menu

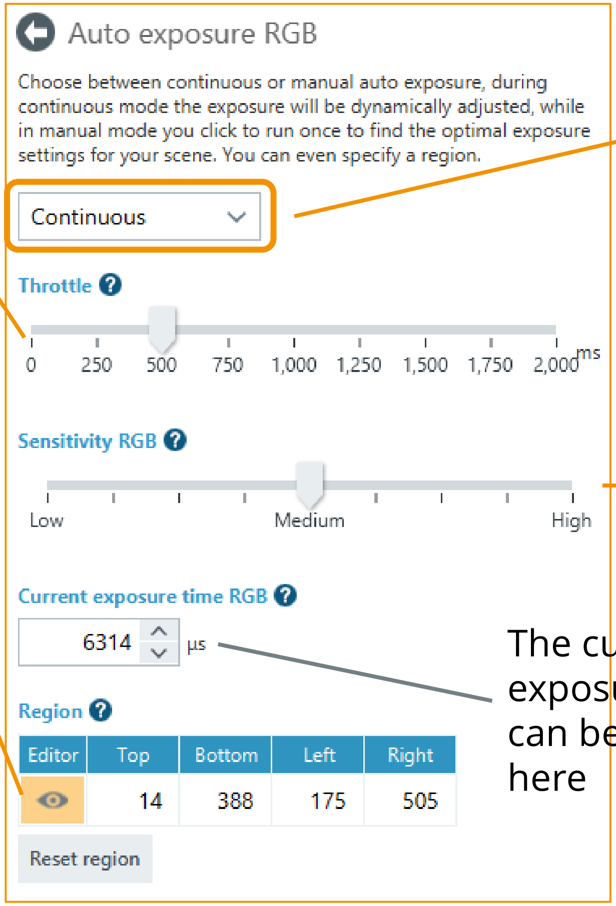
Stereo settings – Auto Exposure RGB – Continuous mode

Use the auto exposure control **in continuous mode** to automatically determine the optimal exposure settings periodically

Defines pause until a next image for RGB exposure time calculation is used

Use the editor tool in the live view to define the region





Once continuous mode is selected the auto exposure on the defined region is dynamically adapted (default: full field of view)

Sensitivity for automatic exposure change.

- Low sensitivity will reduce continuous automatic changes (higher tolerance)
- High sensitivity will increase the adjustment frequency

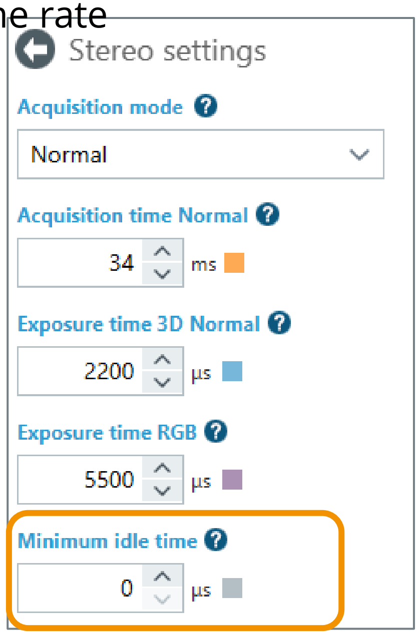
The current exposure time can be seen here

Settings Menu

STEREO SETTINGS – MINIMUM IDLE TIME

SICK
Sensor Intelligence.

- Can be set in order to extend or to guarantee a time duration between two frames without capturing
- During idle time, actions which occur in the field of view are not captured (optimal time slot for object manipulation)
- Increasing the value can result in a lower frame rate



SICK APP SPACE

- **Visionary-S AP**:
- Image.Provider.Camera.V3SXX2_1Config.setMinimumIdleTime(config, **0**)

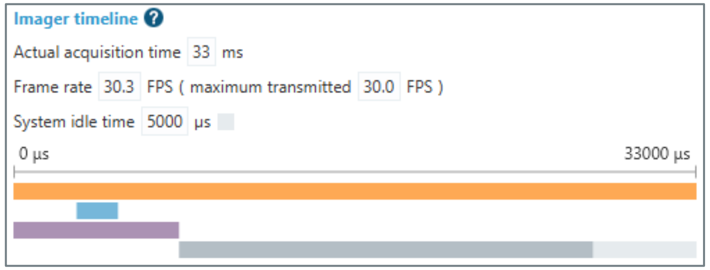
Settings Menu

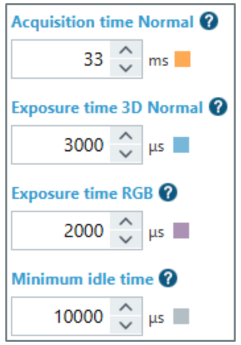
Stereo settings – Overview acquisition timings

- The “overview acquisition timings” visualizes the timings of various settings influencing the frame rate
- You can see the total acquisition time for capturing one frame and the resulting frame rate

The graphical display helps to show when certain processes during the acquisition happen and how they influence the actual acquisition time for each frame.

- The orange bar represents the time value defined in “Acquisition time”
- The blue bar represents the time value defined in “Exposure time 3D”
- The violet bar represents the time value defined in “Exposure time RGB”
- The light gray bar represents system idle time for data processing (which varies and cannot be changed)
- In case a “Minimum idle time” has been set, the dark gray bar fills the light gray bar
- Note that bars and corresponding input fields are highlighted if hovered by the mouse pointer



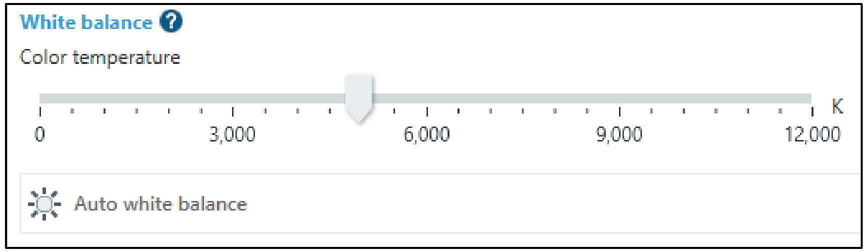


Settings Menu

Stereo settings – White balance

› Different ambient light sources can lead to a red or blue shifted RGB image

› Shifting the white balance to smaller or higher values will counteract the effect of this color shift

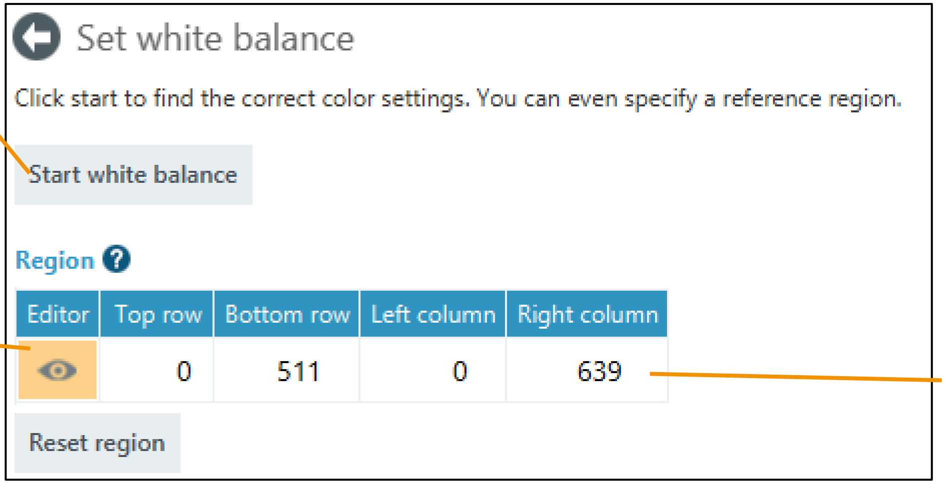


› With the auto white balance tool the color correction can be automated based on a white reference object

› Define a region only consisting of the white object before pressing start!

› Click start to run the auto exposure on the defined region (default: full field of view)

› Use the editor tool in the live view to define the region



› By typing in the pixel values you can specify the region manually

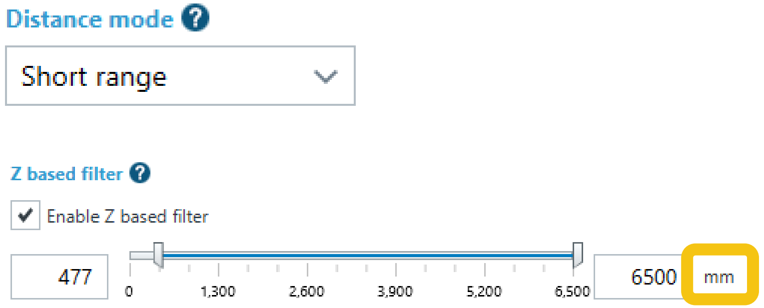
Settings Menu

Stereo settings – Distance mode

For distance mode “Short range” and “Long range” can be defined

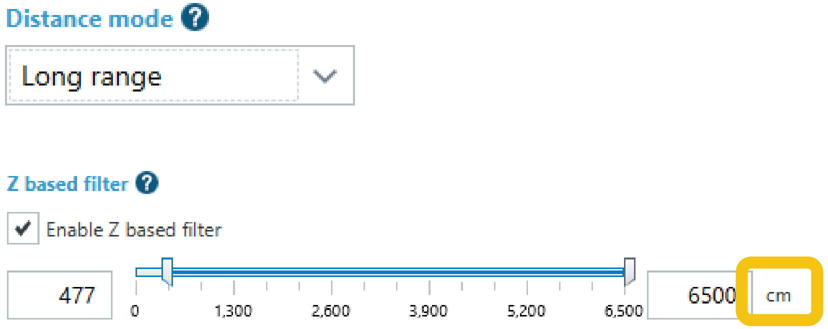
Short range mode is recommended for short range applications.

Depth range is limited to 0.5 …6.5m and provides sub millimeter precision



Long range mode is recommended for applications with working distances above 6.5m.

The working range is increased to 0.5m …65m



Be aware if Z based filter is active and changes in distance mode are set:

- **Short range** mode provides sub mm precision therefore values for Z based filter are in **[mm]**
- **Long range** mode increases working range therefore values for Z based filter are in **[cm]**

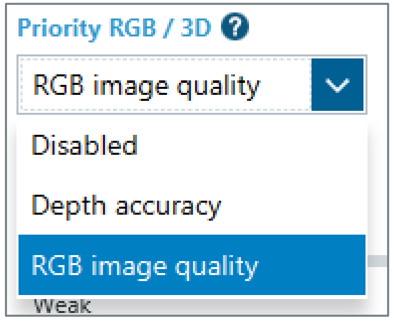
The defined numerical values themselves does not change when changing “Distance mode” only the unit changes !

Settings Menu

Stereo settings – Priority RGB / 3D

SICK
Sensor Intelligence.

- The Visionary-S acquires both depth and color information of its surroundings which can be internally matched and output as a single frame.
- **Depth accuracy**: Use this mode, if only those pixels with valid depth data are of interest to you. Each pixel with a valid depth value will get the best fitting RGB value. Pixels with invalid depth value will distort the RGB image and should be occluded.
- **RGB image quality**: Use this mode, if a correct and complete RGB map is your priority. Each pixel of the RGB map will get the best fitting depth value (or none).
- **Disabled**: Use this mode if you need both RGB and 3D values to be as accurate as possible, while a matching of both information is not required. E.g. recognition of printed label on a single box while measuring its dimensions as accurate as possible.



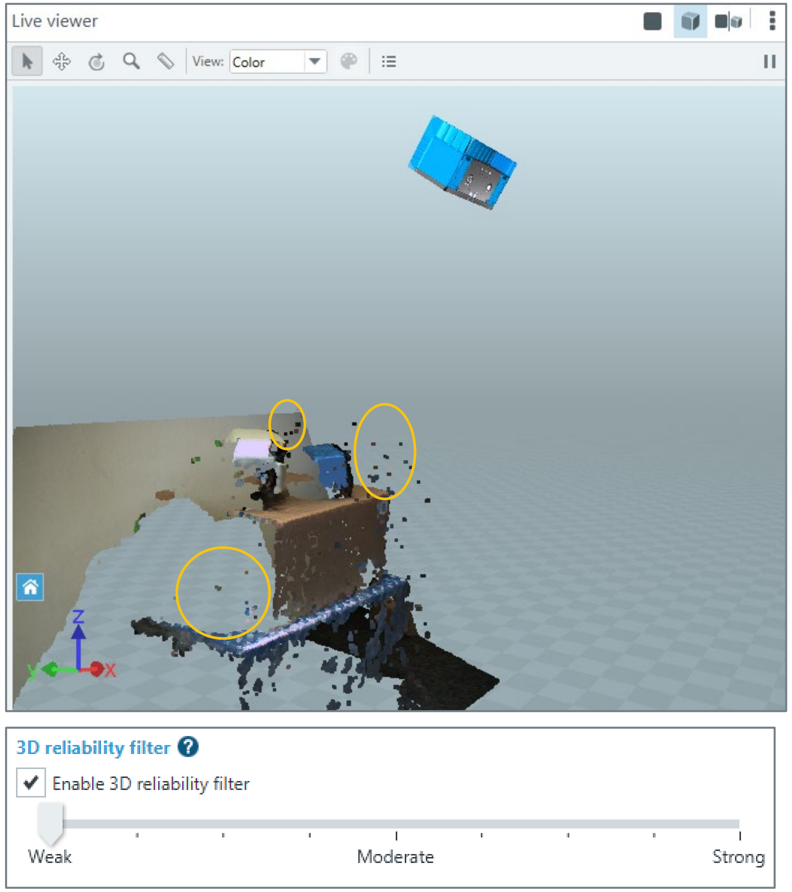
SICK APP SPACE

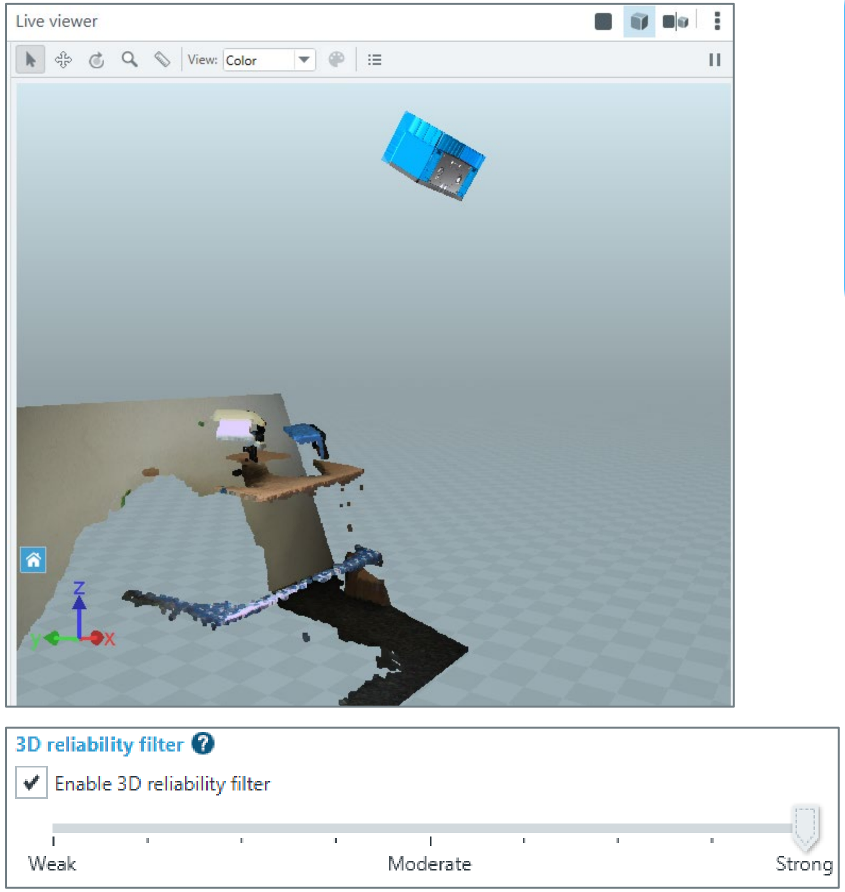
- **Visionary-S AP**:
- *Disabled:* Image.Provider.Camera.V3SXX2_1Config.setColorMappingMode(config,'**Disabled**')
- *Depth accuracy:* Image.Provider.Camera.V3SXX2_1Config.setColorMappingMode(config,'**Disparity_On_Rgb**')
- *RGB image quality:* Image.Provider.Camera.V3SXX2_1Config.setColorMappingMode(config,'**Rgb_On_Disparity**')

Settings Menu

Stereo settings – Reliability filter

- If activated (recommended!), the reliability filter for each pixel is assessed
- Depending on the setting, pixels with lower reliability are hidden in the visualization





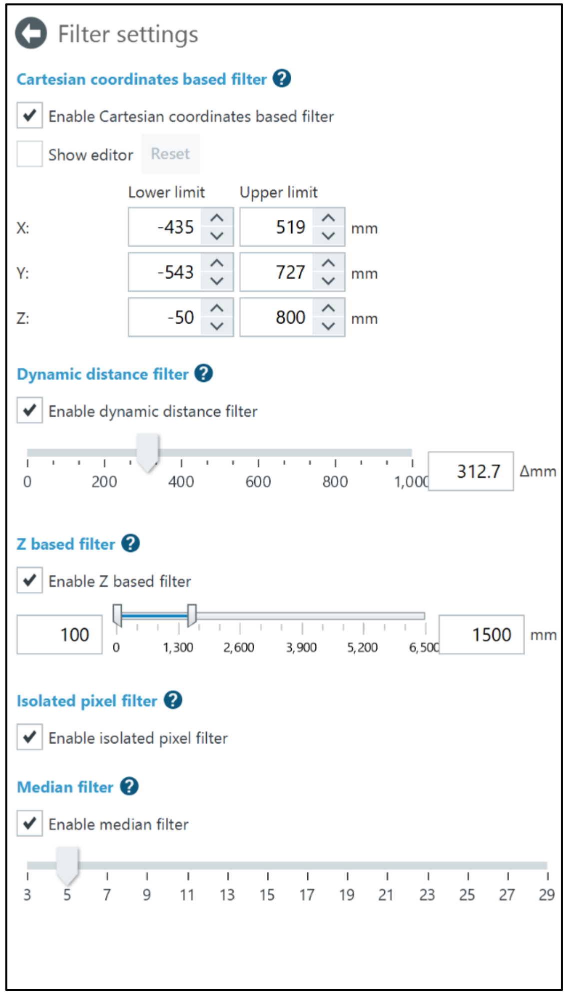
SICK APP SPACE

- **Visionary-S AP:**
- Image.Provider.Camera.V3SXX2_1Config.enableDepthValidation(config, **true**)
- *Weak*: config:setDepthValidationLevel(**1**)
- *Strong*: config:setDepthValidationLevel(**10**)

Settings Menu

Data filter settings

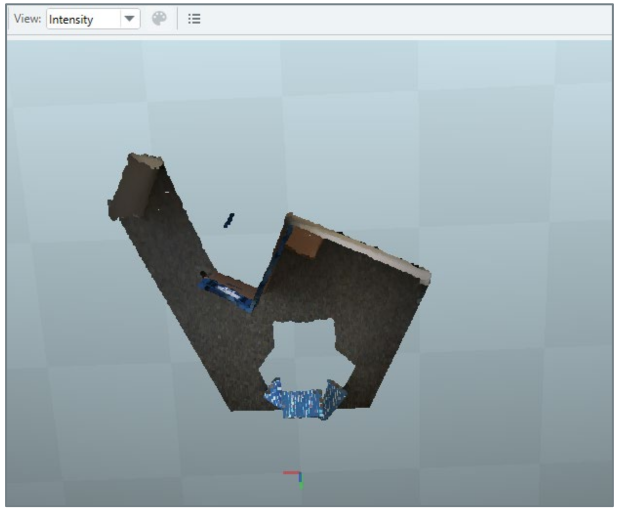
Next slides will show what effects data filter settings will have on the data stream (and visualization)

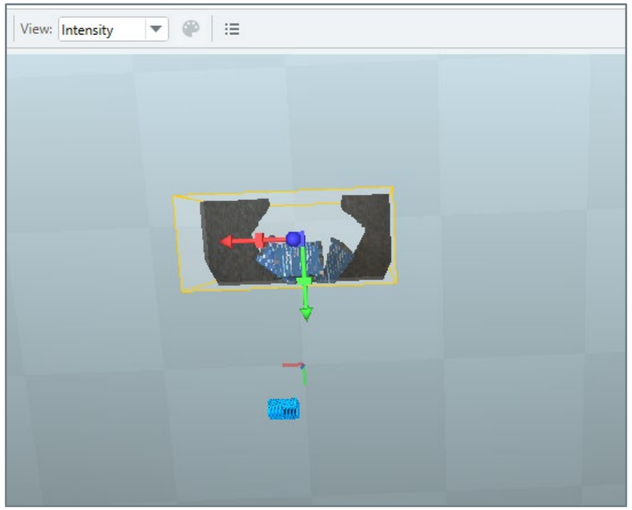


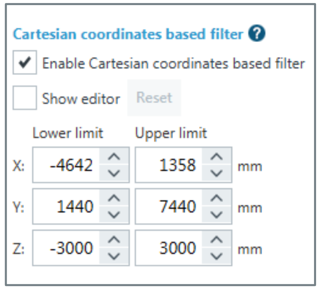
Settings Menu

Filter settings – Cartesian mode

- Helps to crop away background objects in a scene which might disturb object recognition in the foreground (best for fixed camera positions)
- This filter crops the 3D point cloud in X-, Y- and Z world coordinates. Distance values outside this cartesian volume are set to zero .
- The adjustment of the cartesian volume can be done by using the text input fields or by using the graphical editor (activated by show editor)





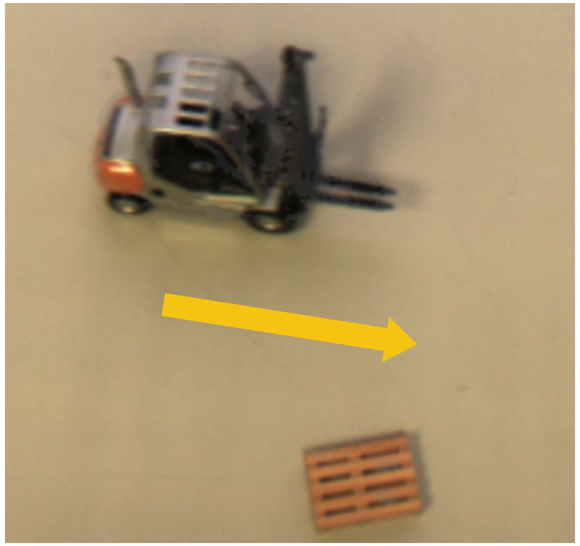


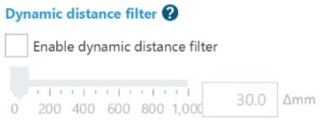
Settings Menu

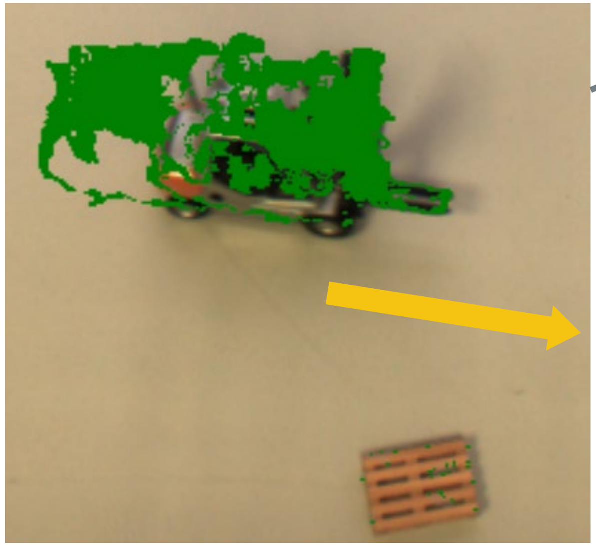
Filter settings – Dynamic distance filter

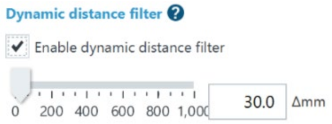
- This filter helps,
- When fast moving objects should get removed from the scene
- When pixels with low depth value repeatability should be suppressed
- Compares distance value of a pixel in the previous and current frame and if greater than the threshold this pixel is considered invalid and its distance value set to zero → filtered out

- Filtered pixels can be visualized using the "error map" of the 2D viewer









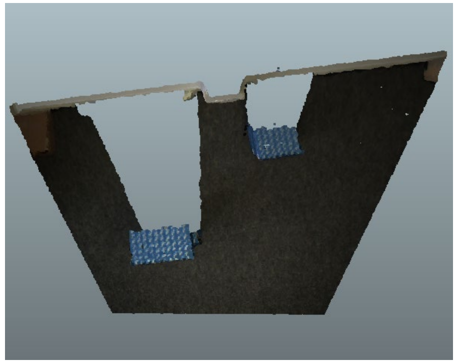
Be aware when using dynamic distance filter in single frame and triggered mode:
If position (distance) of an object changes from one to next frame the 3D data of the object itself can be filtered out due to dynamic distance filter settings.

Settings Menu

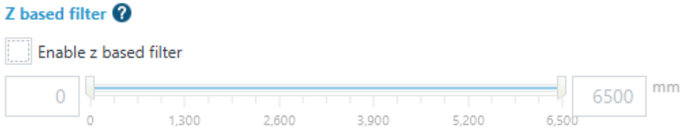
Filter settings – Z-based filter

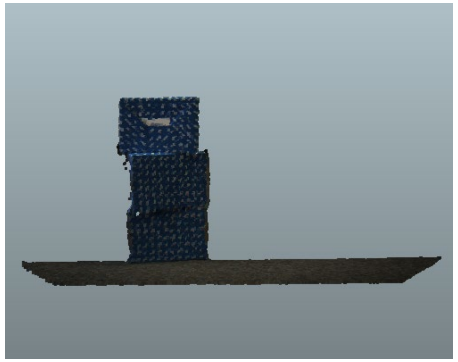
- Helps to crop away background objects in a scene which might disturb object recognition in the foreground (best for dynamic camera positions)
- Set values to zero which are out of the defined local Z range
- In the example boxes are placed in different distances to the camera. Setting the upper threshold to 1430 mm removes all boxes except the boxes in the foreground

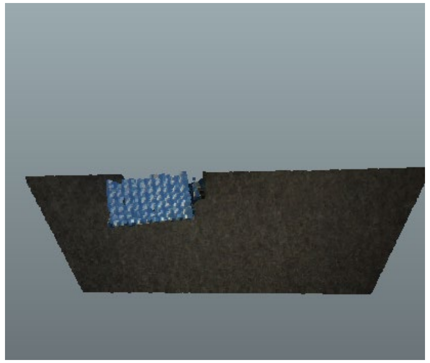
Real scene



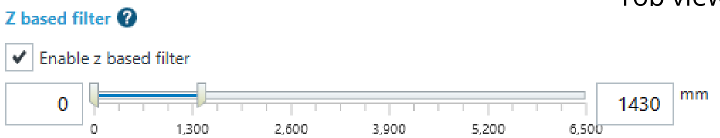
Top view







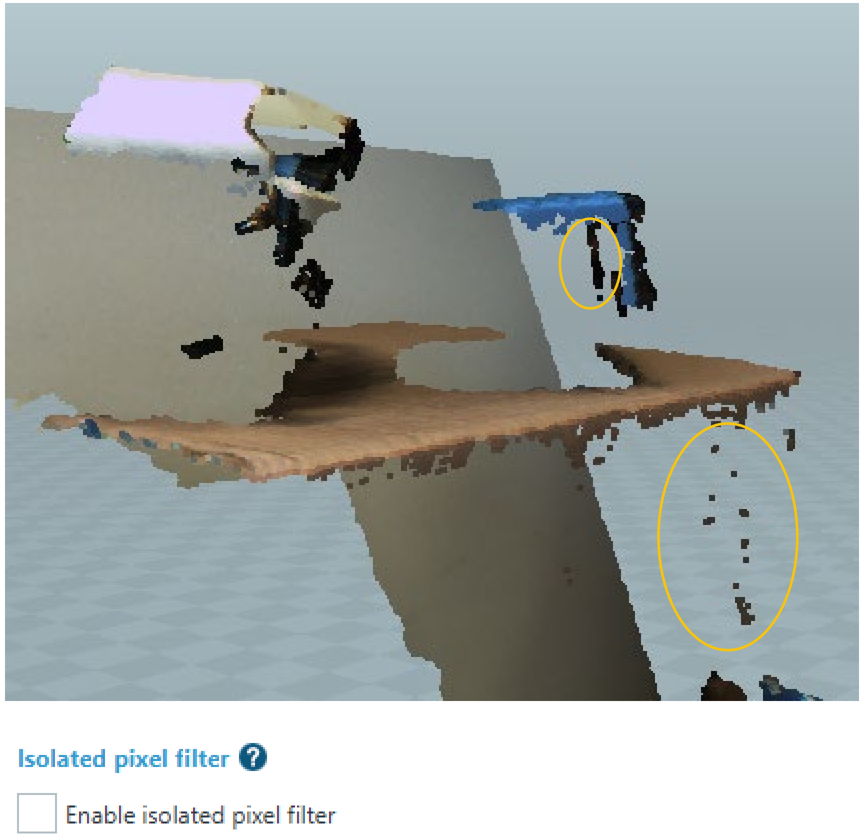
Top view



Settings Menu

Filter settings – Isolated pixel filter

- This filter detects and removes isolated pixels or even smaller clusters of isolated pixels



- Filtered pixels can be visualized using the "error map" of the 2D viewer

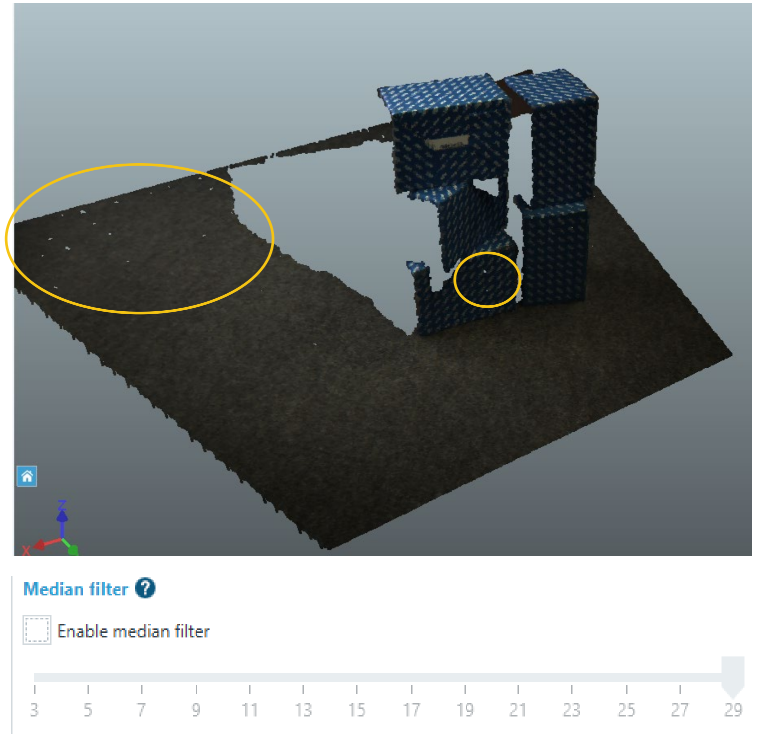
SICK APP SPACE

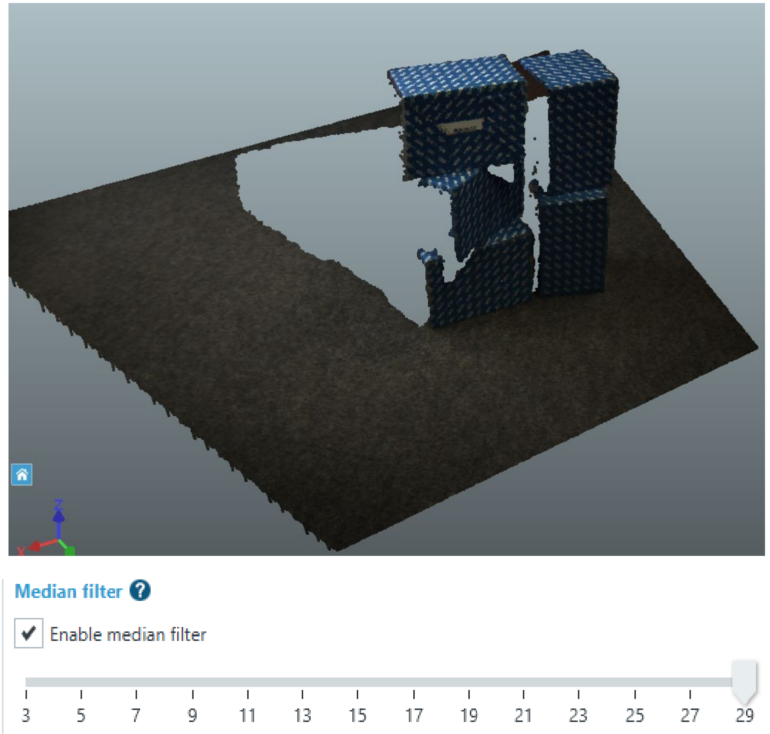
- Visionary-S AP: enableFlyingClusterFilter()

Settings menu

Filter settings – Median filter

- Improves data quality by closing gaps in the data map and increasing the repeatability of depths values
- Returns the median for each pixel over the selected median level of frames.
- Continuous mode: The frame rate is reduced according to the median level
- Single frame and triggered mode: The number of captured images must be the same as the median level
- E.g. three single acquisition commands at a selected median of 3

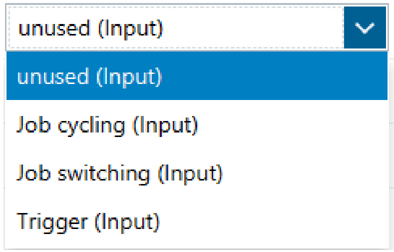


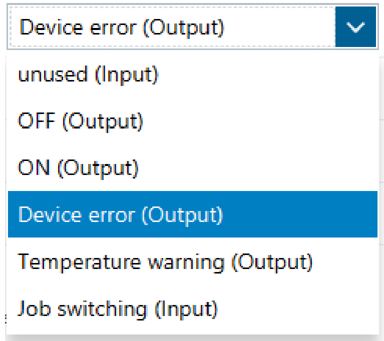


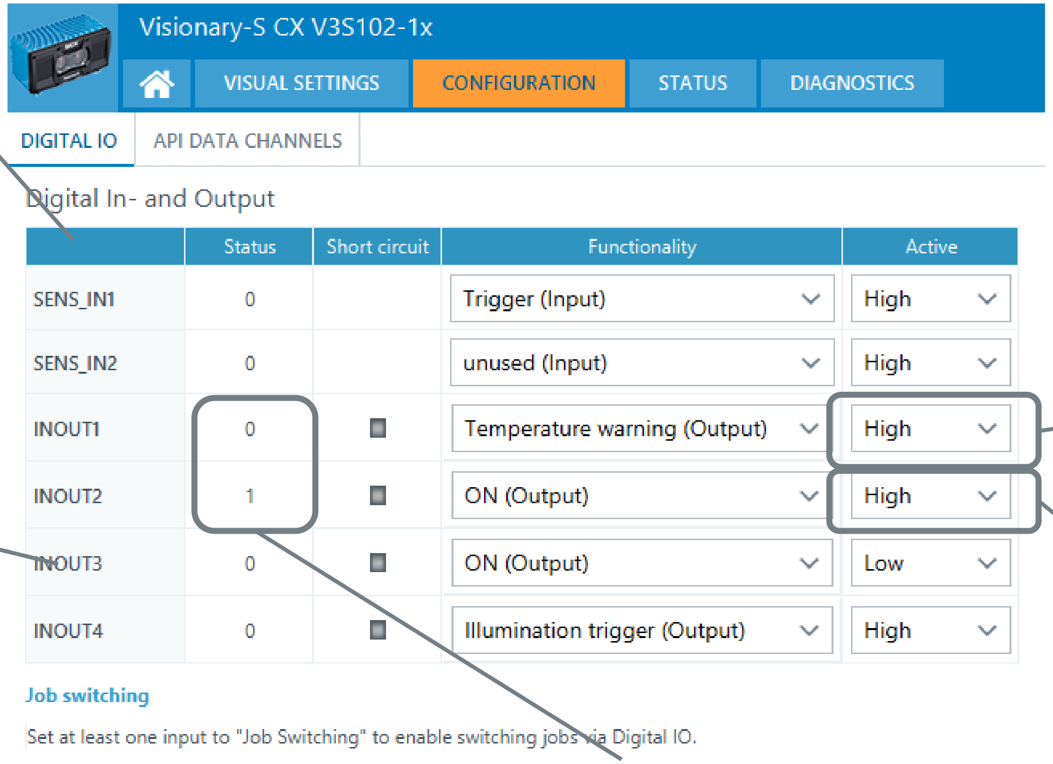
Configuration

Digital IO settings

Current IO status and the active configuration is displayed on the same page







- The in- and output behavior can be inversed. In this example:
  1) output is on HIGH when camera is on
  2) output is on LOW when camera is on

- Status column corresponds with out/input signal:
  0 = no input or output
  1 = detected input or active output signal

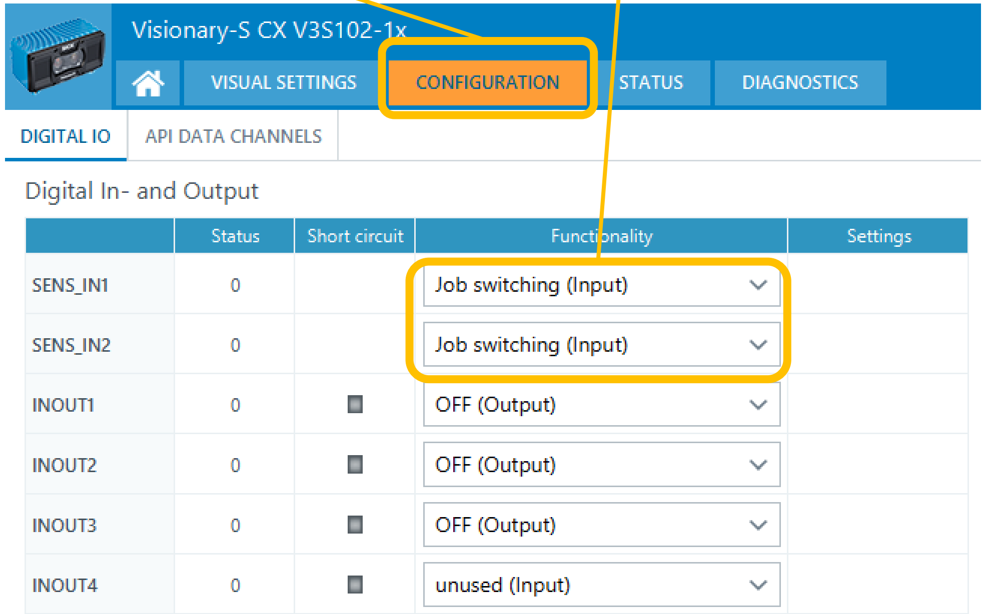
Configuration

Job Switching

The two digital inputs can be used to activate the preconfigured setups/jobs. There are two ways to do this, either with „Job switching“ or „Job cycling“

Job Switching (use if four or less jobs need to be addressed)

On the „Configuration“ page, activate „Job switching“ for both inputs (SENS_IN)



Job switching

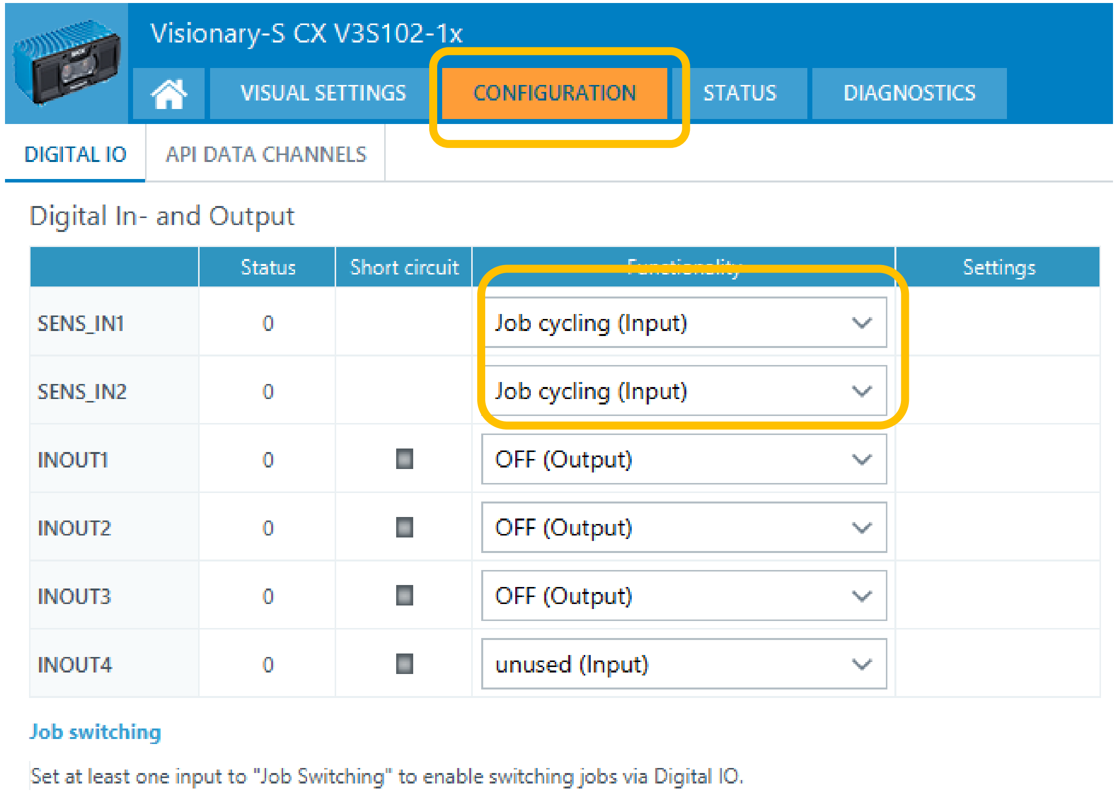
SENS_IN1	SENS_IN2	Job
0	0	Select job...
0	1	Select job...
1	0	Select job...
1	1	Select job...

Configuration
Job Cycling

SICK
Sensor Intelligence.

Job Cycling (use if more than four Jobs need to be addressed via digital inputs)

On the „Configuration“ page, activate „**Job cycling**“ for both inputs (SENS_IN)



Switching Setups/Jobs using Digital inputs

Job Cycling 2

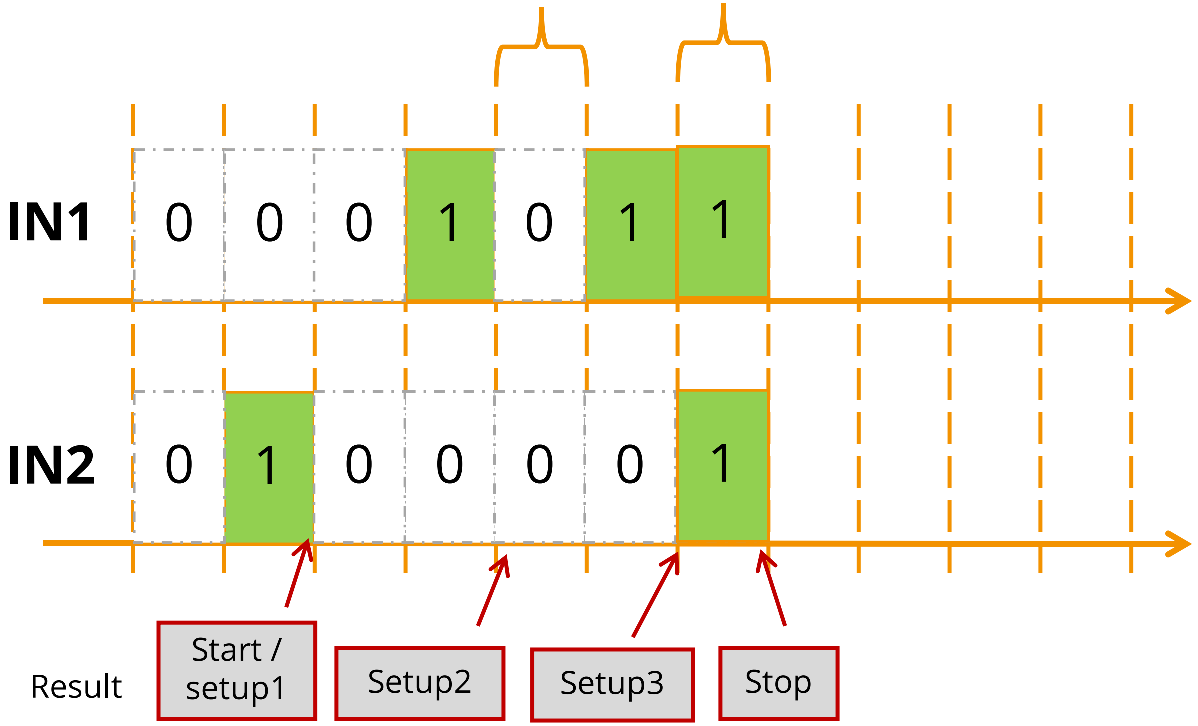
The combination of both digital inputs is interpreted as one of four commands.

Commands:

Input 1 (SENS_IN1)	Input 2 (SENS_iN2)	Result
Low (=0)	Low (=0)	neutral, no action
Low (=0)	High (=1)	start sequence, reset to first job ID
High (=1)	Low (=0)	increase job ID by 1
High (=1)	High (=1)	stop sequence

Minimum command time for each slot is 6 ms, recommended are 10 ms

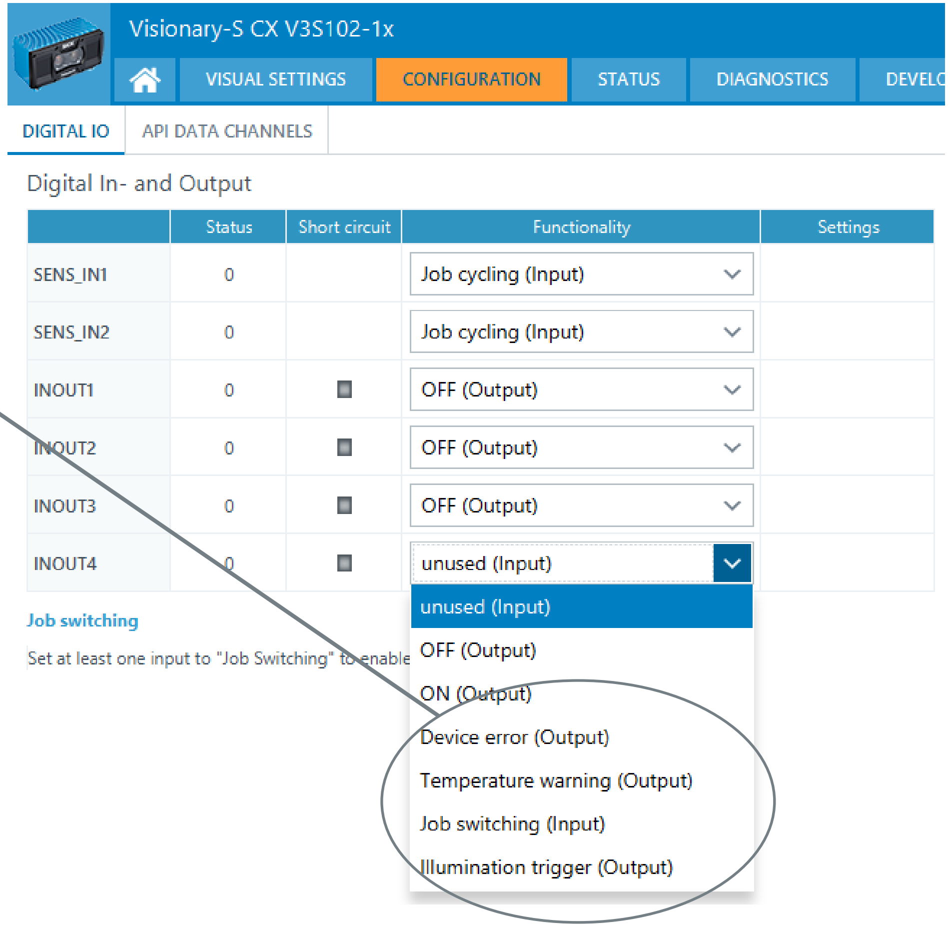
Example sequence, choosing Setup 3:



Configuration

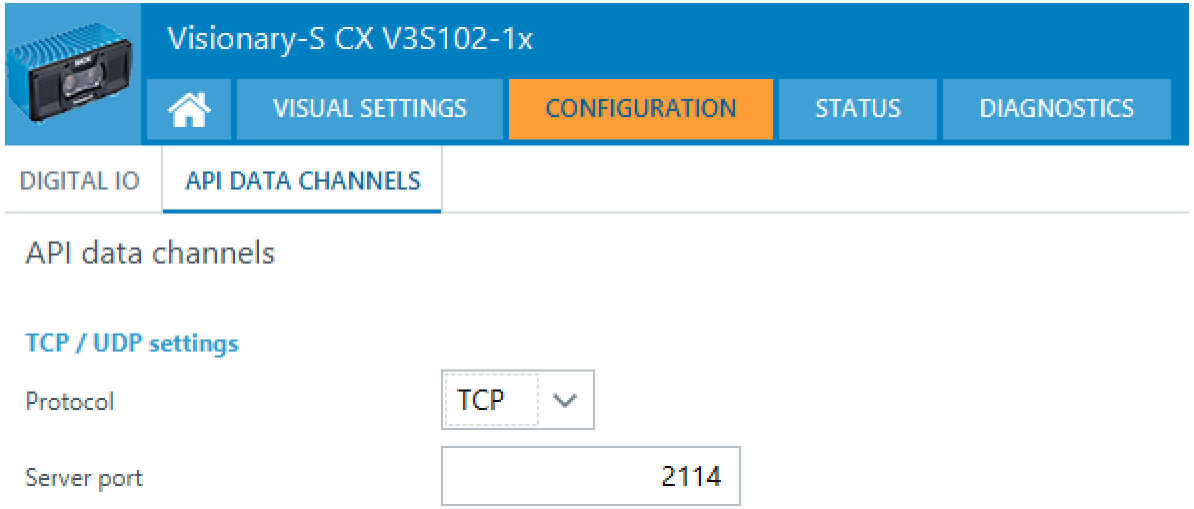
Digital IO settings

- Device error (Output): Output high in case of critical hardware or software error
- Temperature warning(Output): high when critical temperature of device is reached
- INOUT4 only –
  Illumination Trigger (Output): Output signal is on high during RGB acquisition. Allows for synchronization with external light source



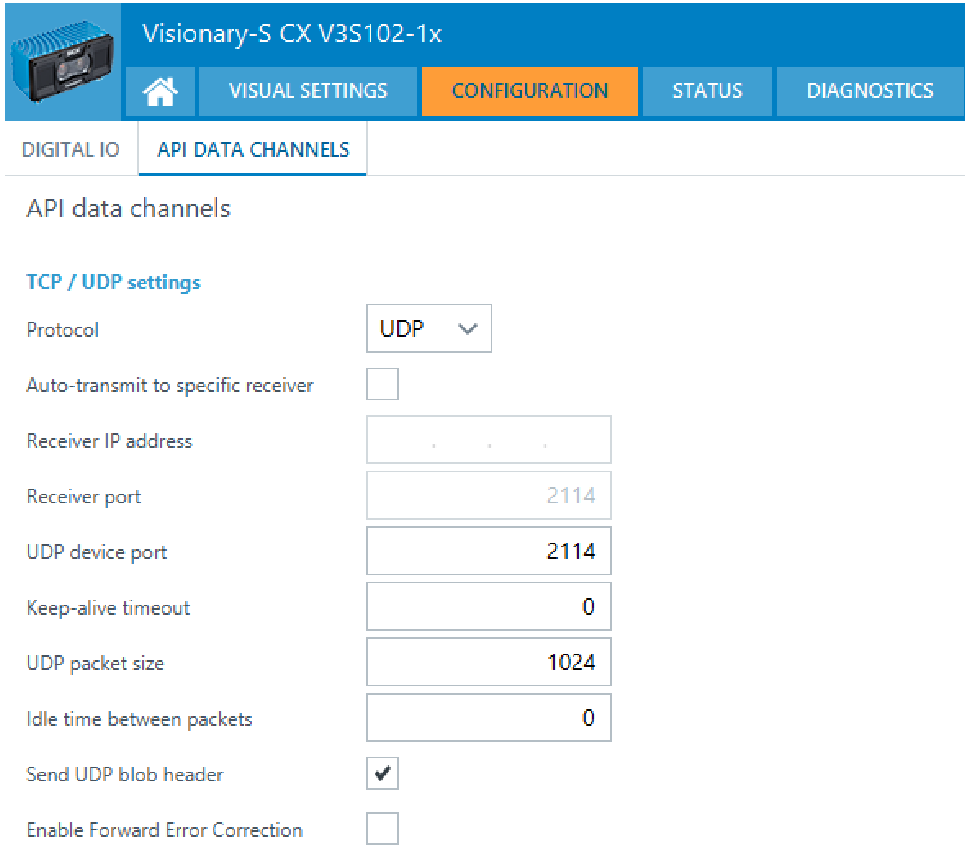
Configuration
API DATA CHANNELS

- Can be used to change TCP/UDP settings
- Port settings can be changed
- Shows the current configuration status



Configuration
API DATA CHANNELS

- Can be used to change TCP/UDP settings
- Port settings can be changed
- Shows the current configuration status



Status/Diagnostics

SICK
Sensor Intelligence.

The status page provides an overview of all relevant parameters

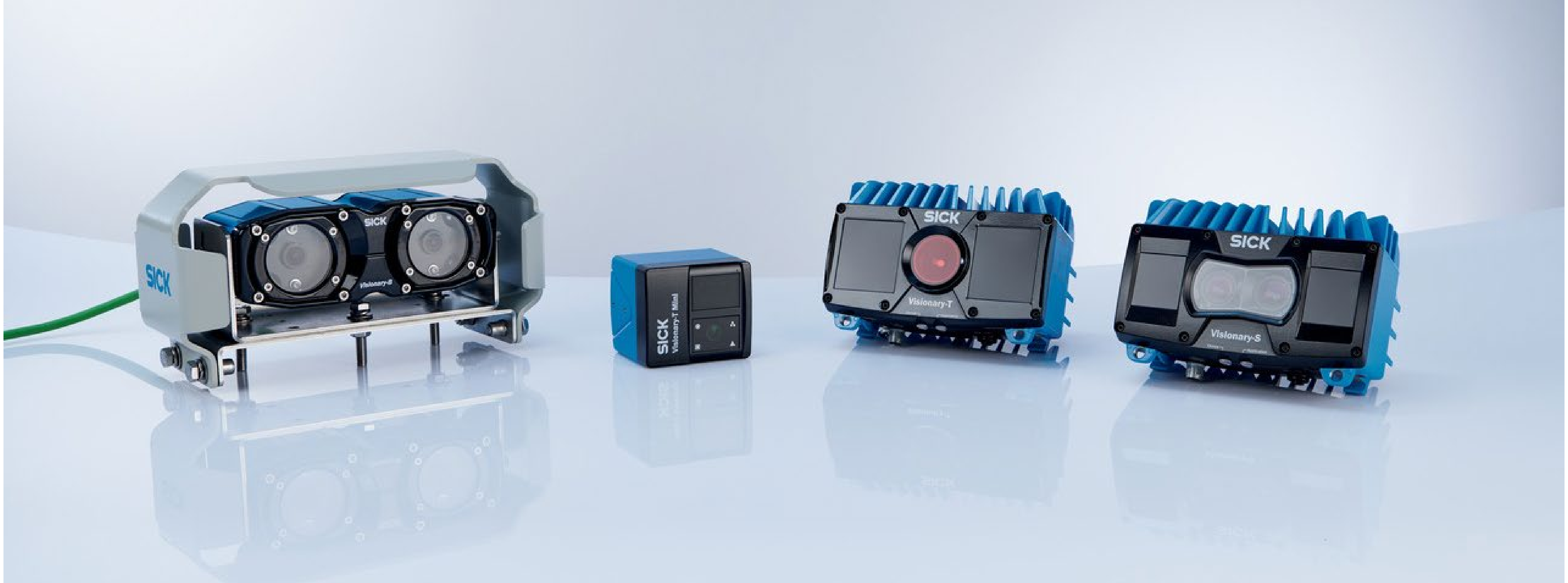
- The traffic light signal next to the name shows if everything is working correctly
- Click on the name of the parameters to get detailed information
- To see detailed numbers/values use the link in the opened window. The link guides you to the diagnostics page
- It is also possible to navigate to the diagnostics page by using the main navigation tabs

SICK PRODUCT PORTFOLIO

VISIONARY-T (3D time-of-flight), VISIONARY-B (Passive stereo) AND VISIONARY-S (Structured light stereo)

SICK
Sensor Intelligence.

3D Snapshot Cameras



No matter which **3D snapshot vision technology** is behind it, SICK offers the suitable one for **your** application

Thank you for your attention.

SICK AG – Mobile Perception – 3D Snapshot

SICK
Sensor Intelligence.

2.1.4 Diagnostics Description

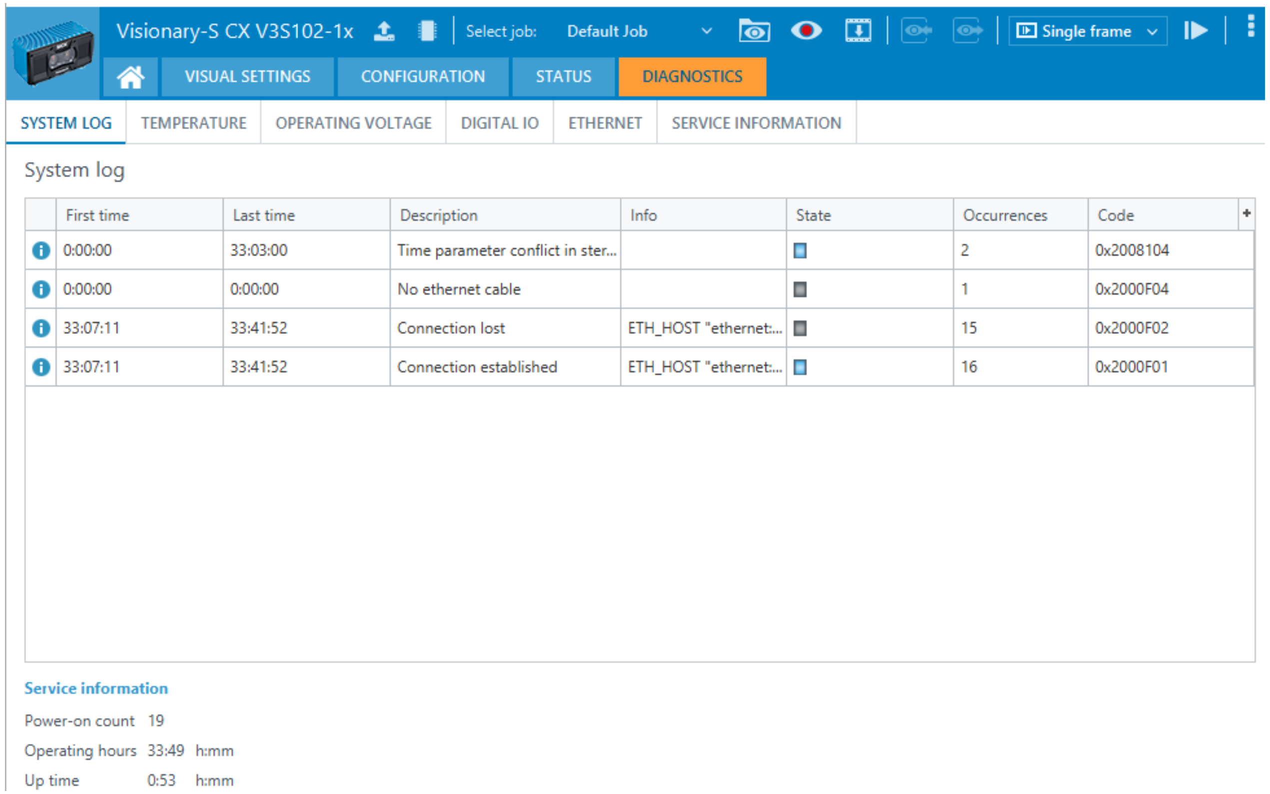
SICK AG – Mobile Perception – 3D Snapshot

Content

Visionary-S CX
DIAGNOSTICS TOOL

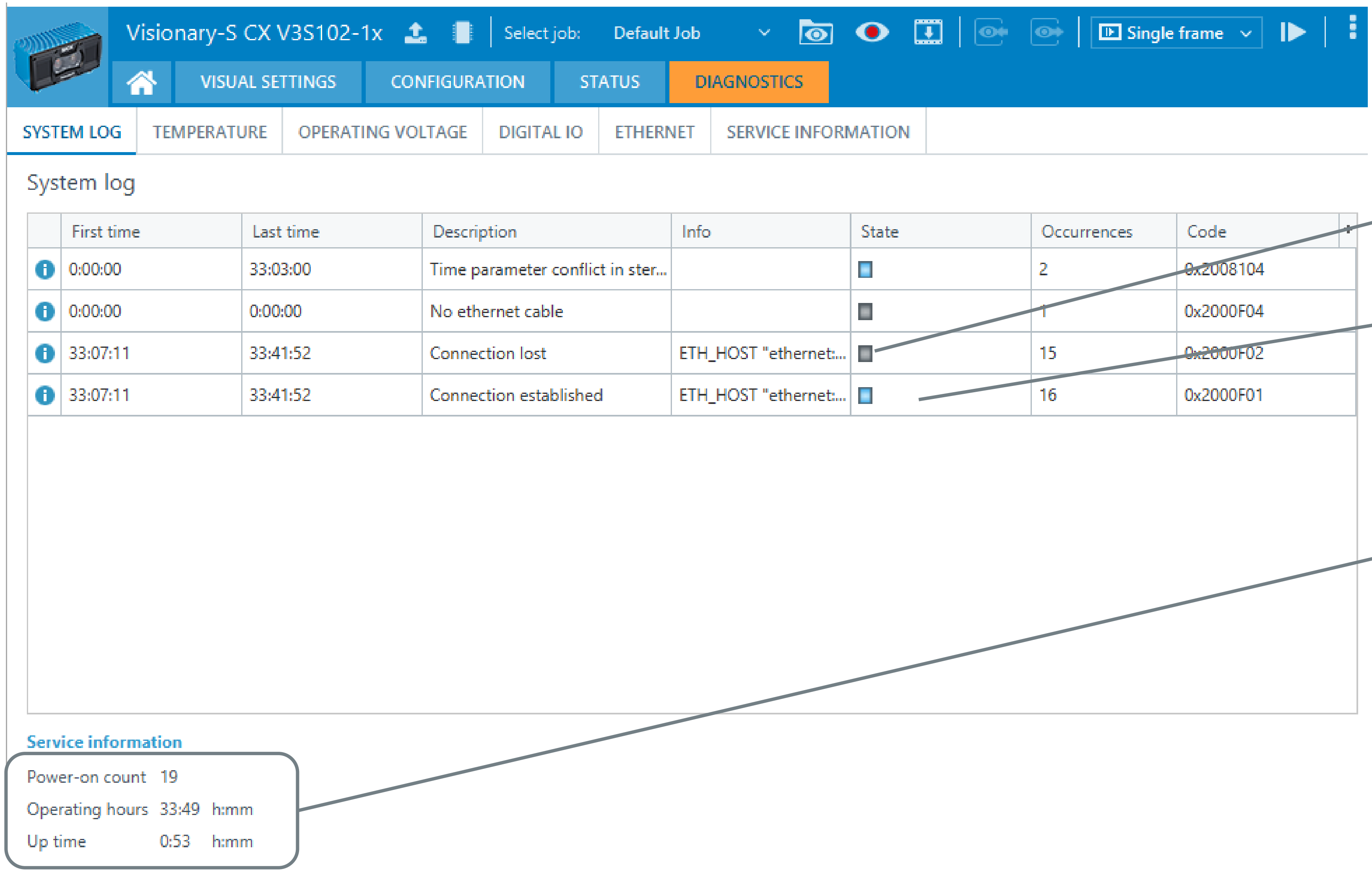
SICK
Sensor Intelligence.

- **Diagnostics tool for service purposes**
  - System log (e.g. critical occasions)
  - Temperature (of different device modules)
  - Operating voltage
  - Digital In- and Output (short circuit)
  - Ethernet
    (frame and error counter)
  - Service information
    (relevant for service)



Visionary-S CX
SYSTEM STATUS

Active and inactive messages since last power-up



Inactive message.

Active message.

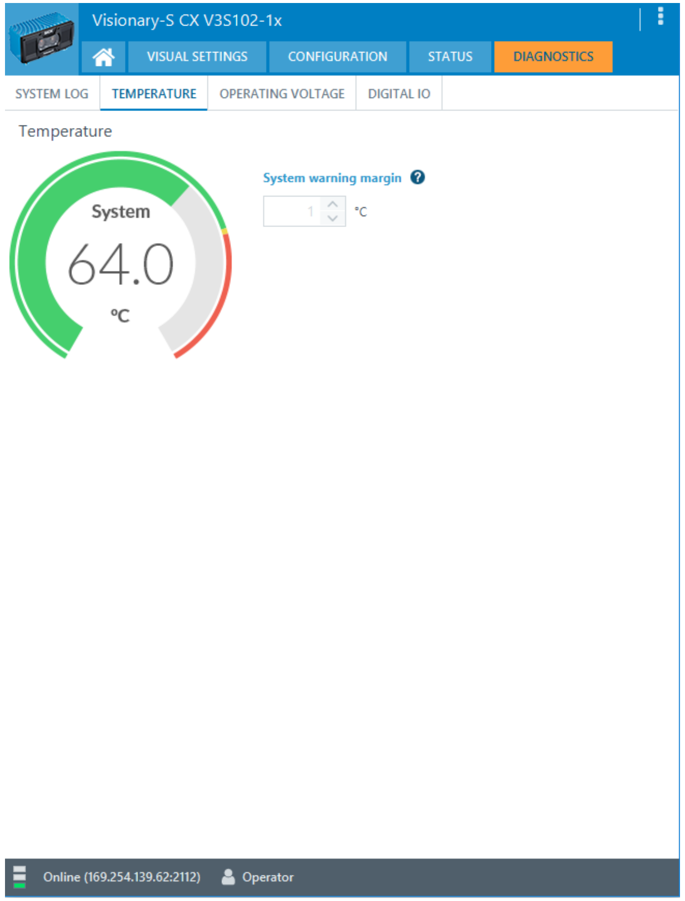
Counter of Power-ons, operating hours and up time

Visionary-S CX
TEMPERATURE

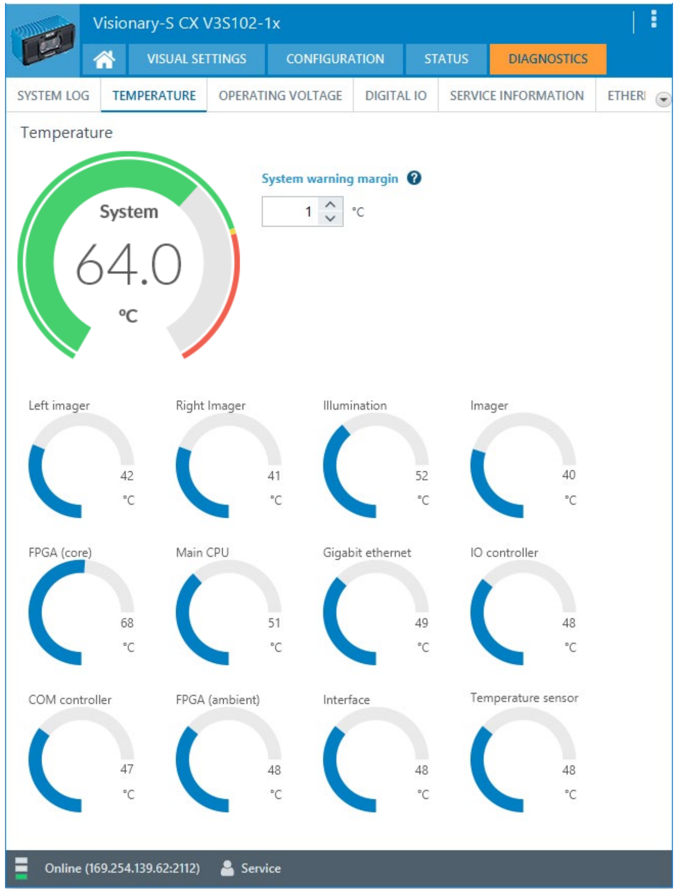
Login as *Service* to get detailed information

- Password: CUST_SERV

Userlevel Operator



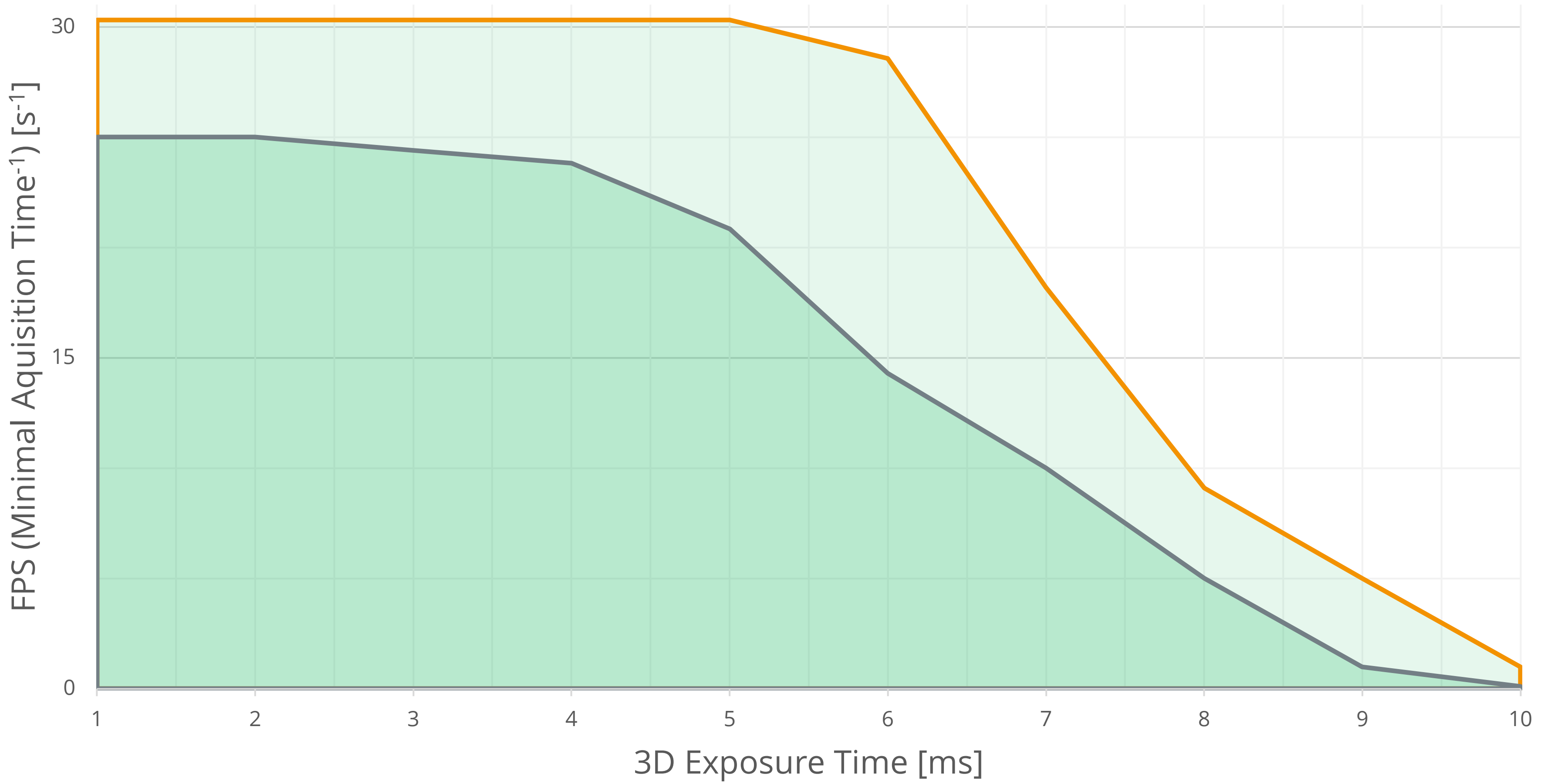
Userlevel Service



Visionary-S CX

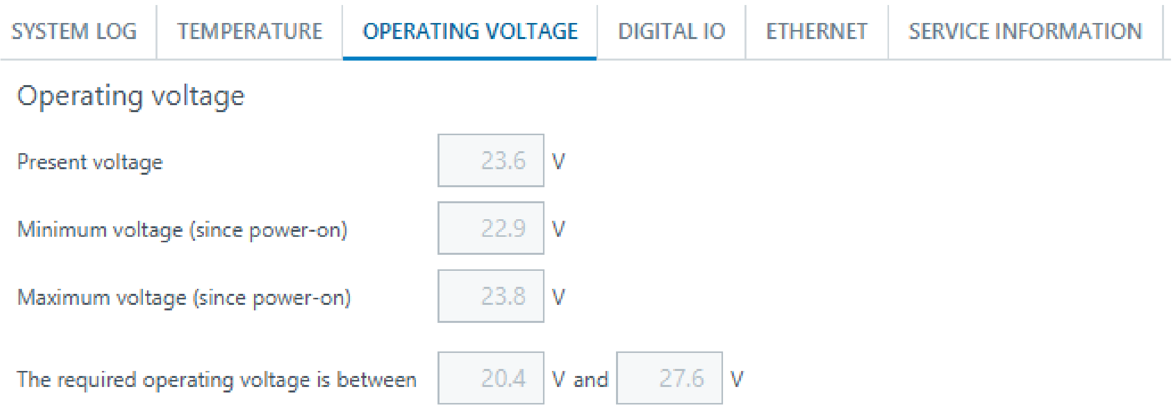
TEMPERATURE: STEREO SETTINGS GUIDELINE

3D exposure settings > 5ms can lead to gradual increase of device temperature when acquiring with full frame rate. Settings which are part of the green area of the graph below (e.g. 5 ms @ 30FPS Normal) will work over the full ambient temperature range specified for the device (45 / 50°C)



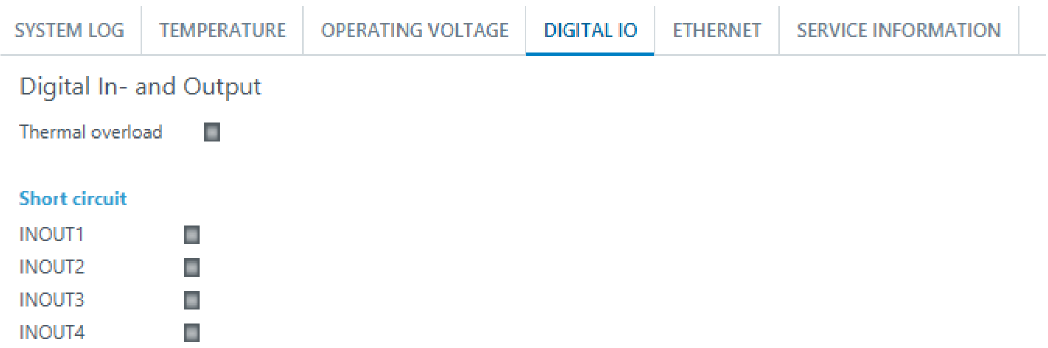
Visionary-S CX
OPERATING VOLTAGE

Monitoring of the operating voltage



Visionary-S CX
DIGITAL IO

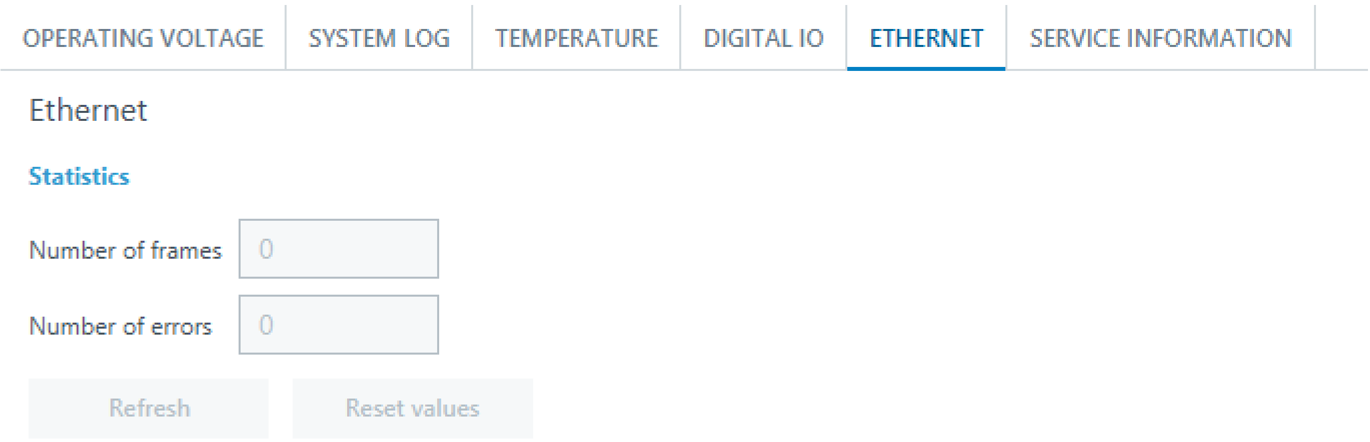
Status of the digital outputs. To reset IO errors a device restart might be necessary



Visionary-S CX
ETHERNET

Statistics and information about Ethernet interface

- Userlevel "Service" required



Visionary-S CX
SERVICE INFORMATION

Sensor Intelligence.

Type code, serial number, version numbers

- Userlevel "Service" required

SYSTEM LOG | TEMPERATURE | OPERATING VOLTAGE | DIGITAL IO | ETHERNET | SERVICE INFORMATION

Service Information

Device information

Manufacturer	SICK AG	**Firmware version**	5.7.0.21157C
Device type	V3S102-1AAAAAA	**SDD version**	5.7.1.21138C
Order number	2092957	**Serial number**	19050005

Software component versions

Kernel	4.9.30-rt20 #1 SMP Tue Ja
Bootloader	U-Boot 2018.03-05935-g
IO Controller	IS020549
FPGA Bitstream	10.9.0.20816
LMC Version	LMC20585

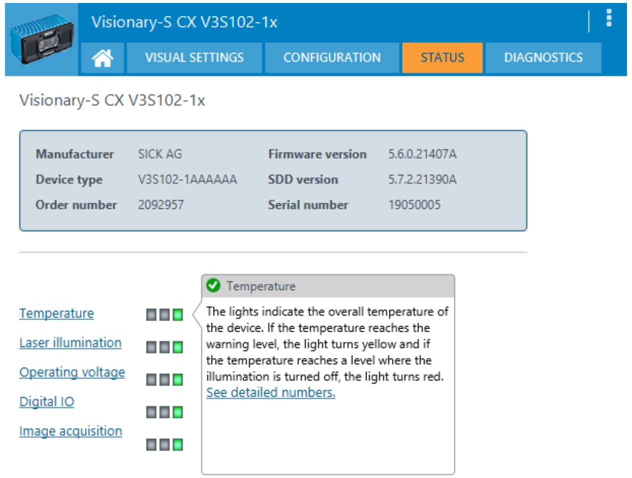
Visionary-S CX

DEVICE STATUS

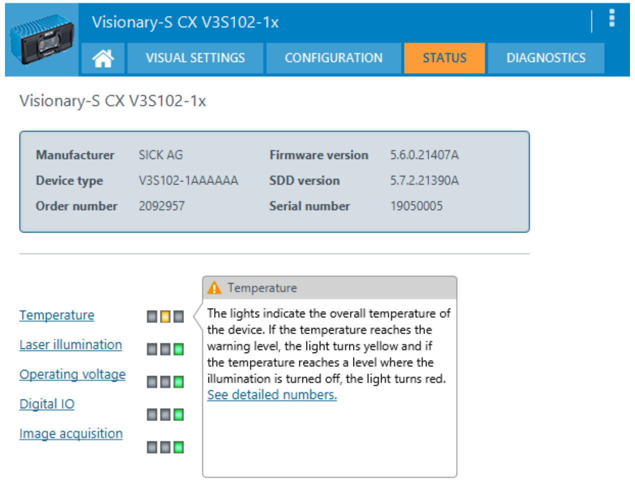
Additional information about the health status of the device hardware

Status of the device temperature

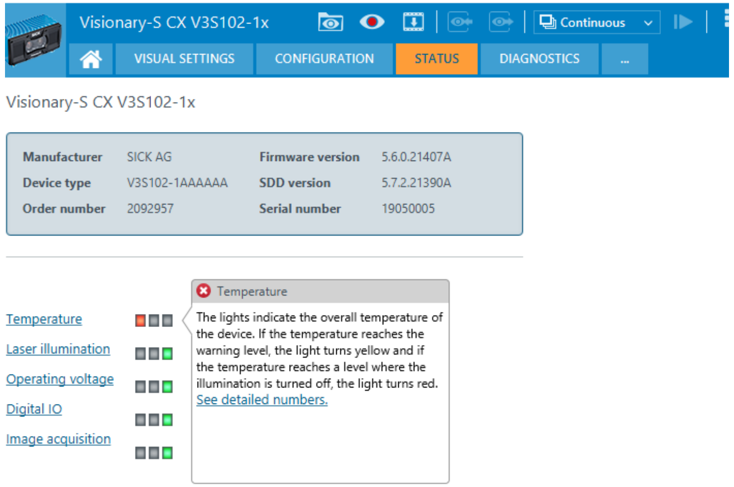
Fine.



Warning level. Additional cooling, reduction of frame rate or 3D exposure recommended.



Error level. Overheating. Illumination is switched off automatically.



NOTE:
Illumination will turn on again when device temperature reaches normal temperature (Temp < warning and error temperature level)

Thank you for your attention.

SICK AG – Mobile Perception – 3D Snapshot

2.1.5
Trigger mode description

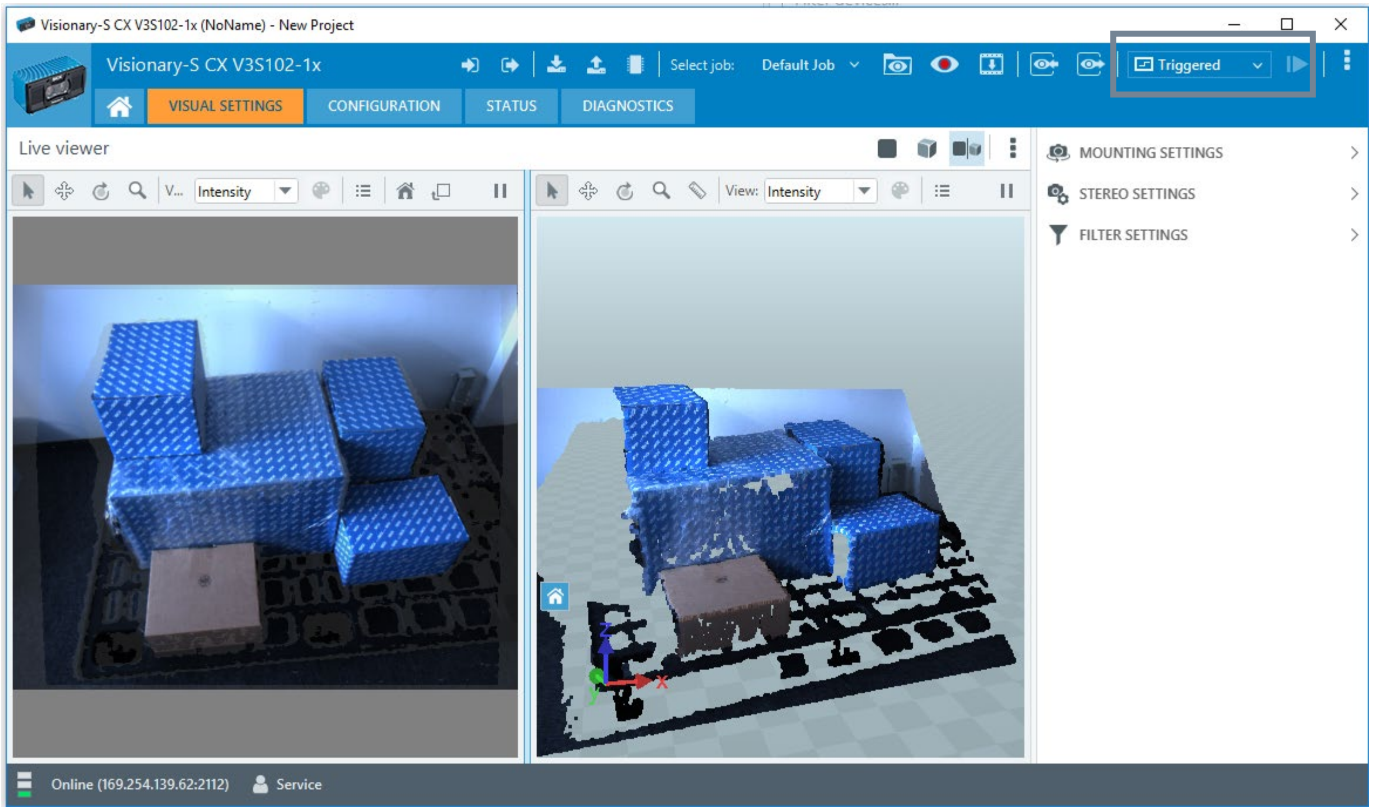
SICK AG – Mobile Perception – 3D Snapshot

Content

Trigger mode
OVERVIEW

SICK Sensor Intelligence.

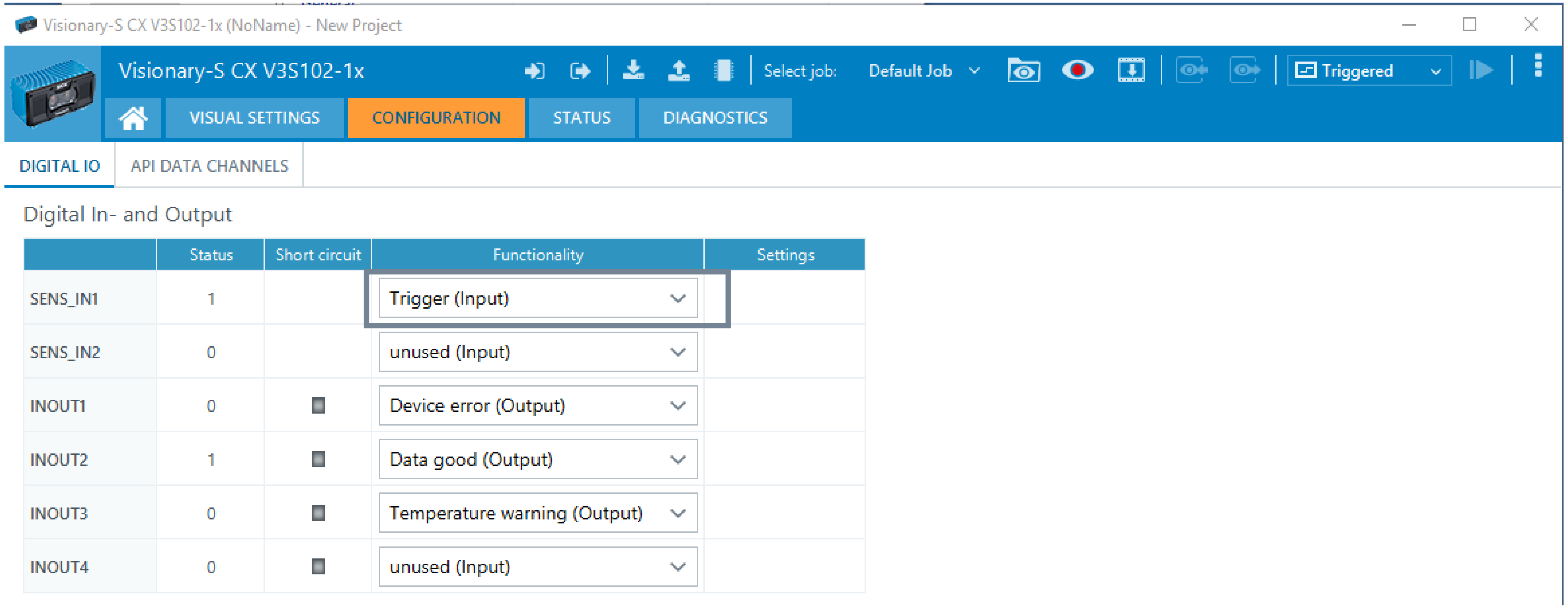
- In trigger mode (“**Triggered**”) the camera does not acquire images until externally triggered
- After a trigger, the frame is captured and transferred to the host
- To simulate the trigger behavior in SOPAS, change to „**Single frame**“ and press the button to trigger the next frame.



Trigger mode

ACTIVATING THE TRIGGER MODE

- To activate the trigger mode, select „**Triggered**“ from the drop-down menu
- Make sure that in the CONFIGURATION tab „Trigger (Input)“ is checked on „SENS_IN1“



- **Note:** Commands send via telegram are ignored in “**Triggered**” mode

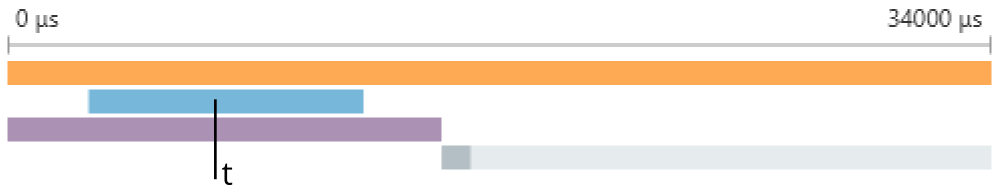
Trigger mode
GENERAL

- Visionary-S comes with a hardware trigger which offers minimal delay between trigger input and frame capture (~500 μs)
- A trigger signal can only be configured on „**SENS_IN1**“
- Trigger pulses <80us are ignored → trigger pulse should be > 200us to ensure correct detection of trigger request
- The Visionary-S will not accept any new trigger during the processing of a previous trigger.
- For synchronizing the trigger behavior, you can use “Illumination trigger (Output)” which is only available on “**INOUT4**”

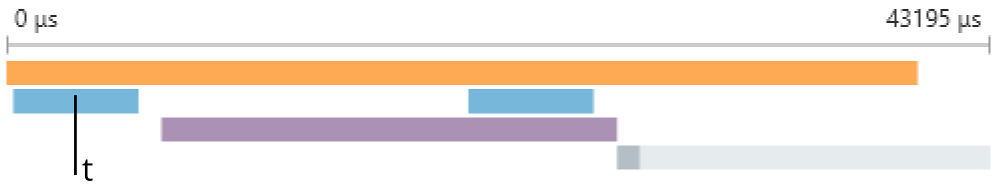
- **Note:**
  - The illumination output is on high when no RGB acquisition is ongoing
  - The illumination output is on low during RGB acquisition
  - RGB acquisition always ends after 3D acquisition has finished
  - No image capturing between Illumination trigger (Output) on high and the next Trigger (Input)

Trigger mode
TIMESTAMP

- In normal mode, the timestamp equals the center of the 3D acquisition

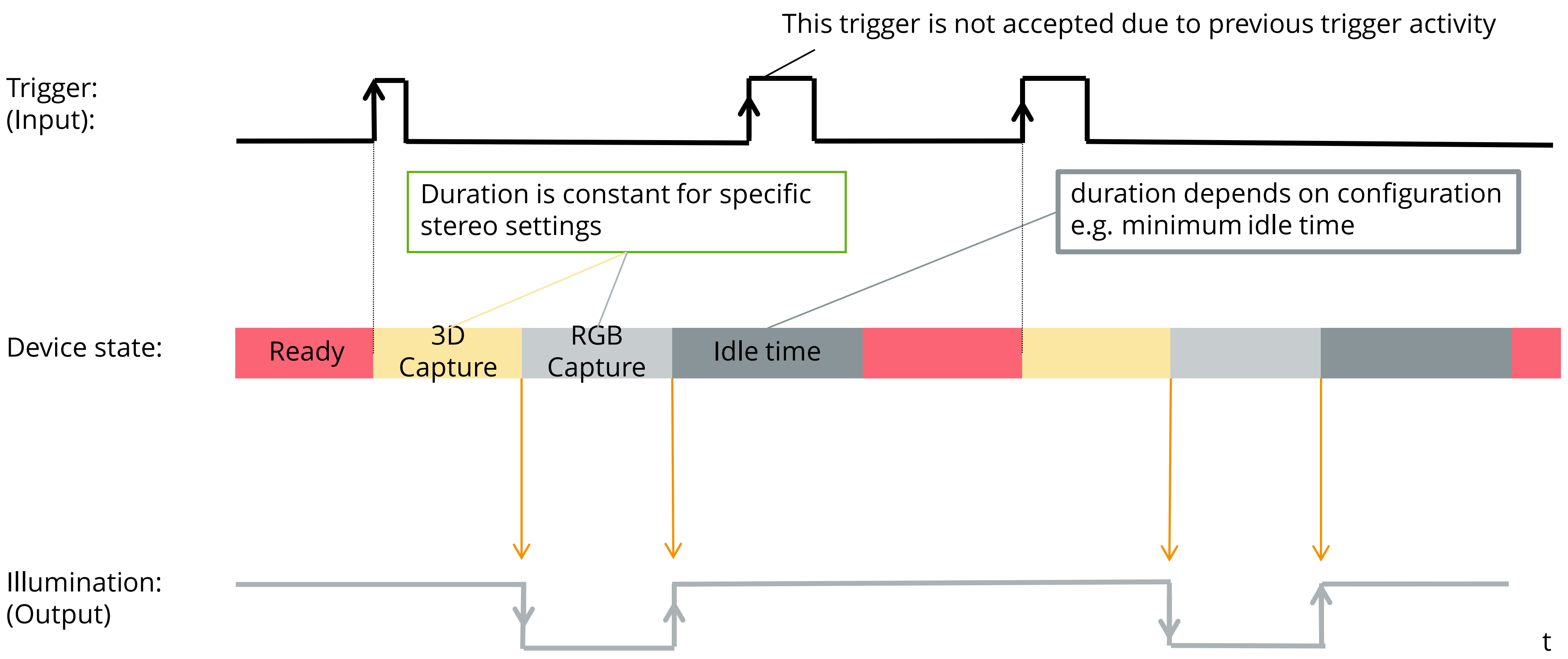


- In HDR/HQM mode, the timestamp equals the center of the first 3D acquisition



- The duration between trigger input and timestamp depends on the acquisition settings

Trigger mode
VISUALIZATION



Thank you for your attention.

SICK AG – Mobile Perception – 3D Snapshot

2.1.6
Firmware update guide

SICK AG – Mobile Perception – 3D Snapshot

Content

SICK Sensor Intelligence.

Introduction

SICK
Sensor Intelligence.

- This document will guide you through the firmware update process.

- We highly recommend to use the latest firmware versions for your Visionary device in order to enable the latest device features.

- Make sure to save your device configuration as *.sopas file before you update your Visionary device.
  - For that, simply import/export the *.sopas file to/from the device via SOPAS
  - NOTE: When re-importing *.sopas after a firmware update please refer to the separate documentation for this case

- Disclaimer:
  - The device configuration may be lost after an update.
  - The graphical user interface may change after an update.
  - The latest firmware may support only the latest SOPAS ET version.

Preparation

SICK Sensor Intelligence.

- Install the latest version of SOPAS ET, e.g. 2021.1 or higher
  - SOPAS ET is available on www.sick.com

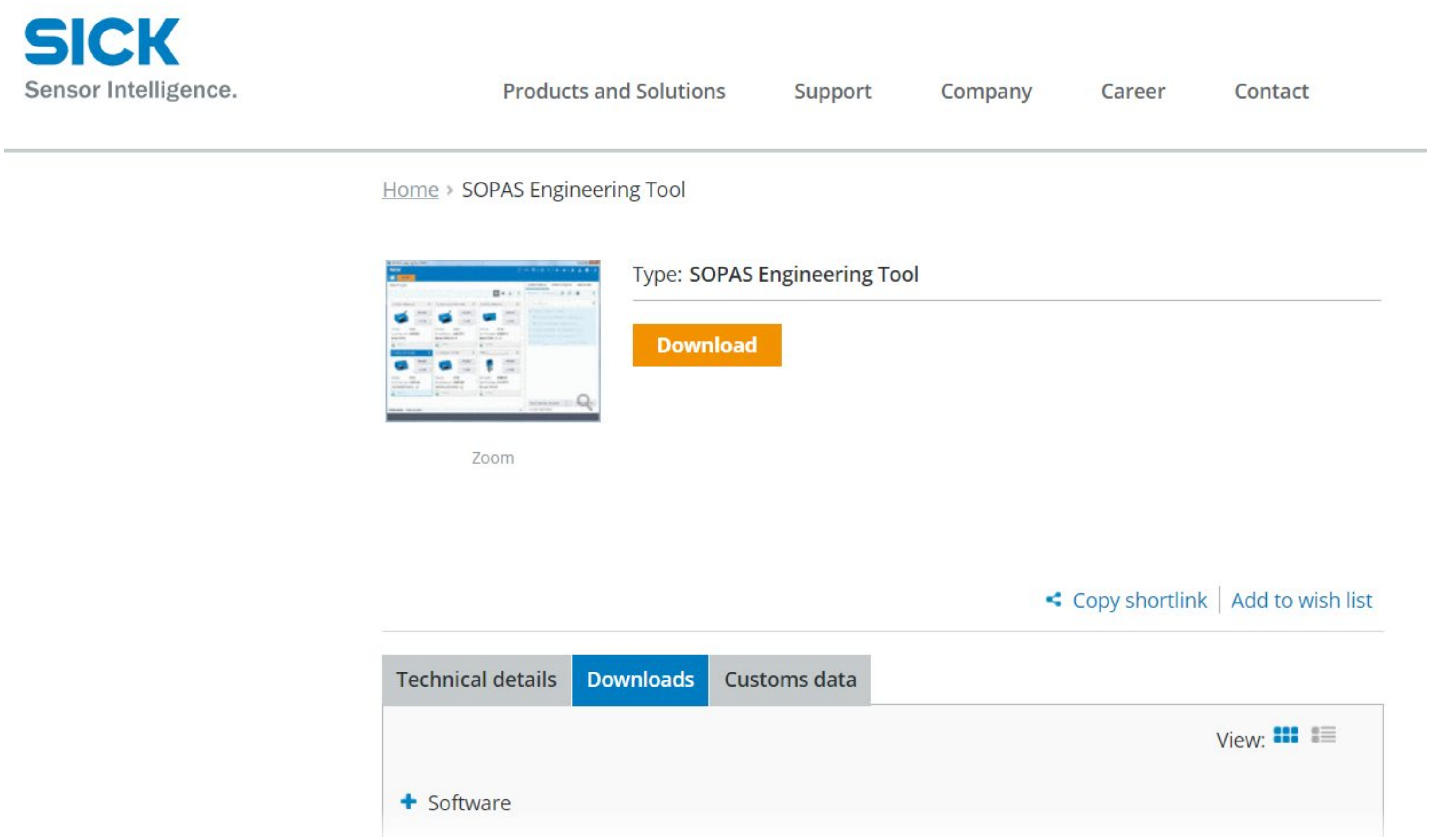
- Be sure that your power connection works properly and is well connected to the device

- Connect the Visionary device with your PC:
  - Connection via Ethernet
  - Connection proved e.g. by receiving 3D data

- Download the latest device firmware (.spk) from sick.com
  - https://www.sick.com/ag/en/p/p677696

Install SOPAS ET

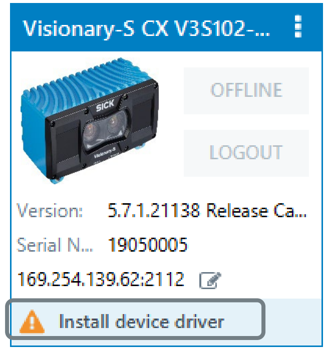
- SOPAS ET is available on https://www.sick.com/SOPAS_ET



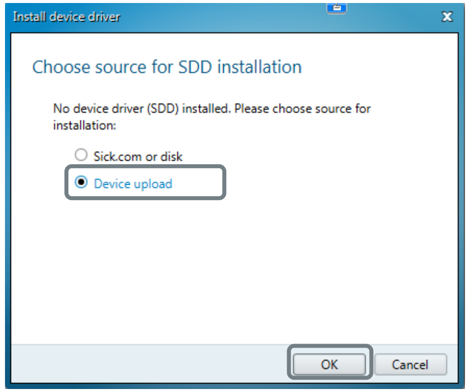
Install SDD
(IF NOT ALREADY DONE)

- After a successful connection the driver might be missing.

- Click on *Install device driver*



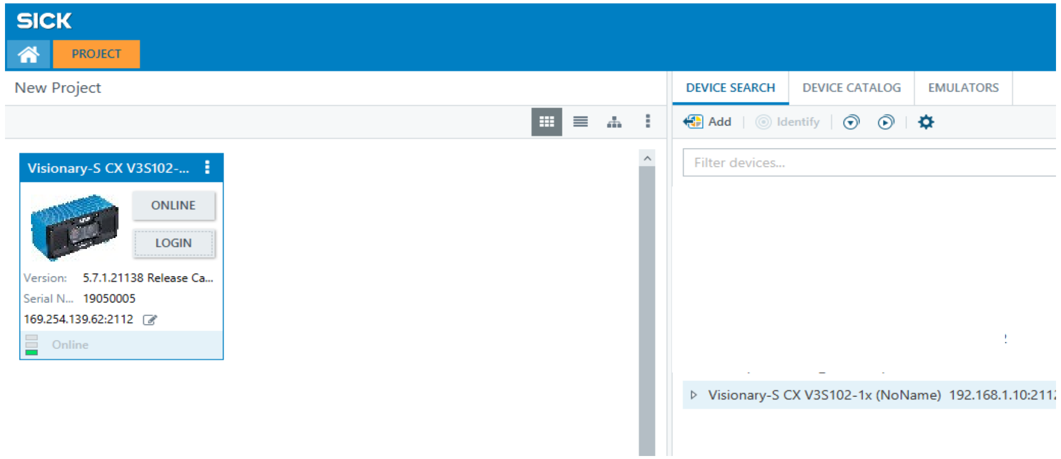
- Choose *Device upload*



Start

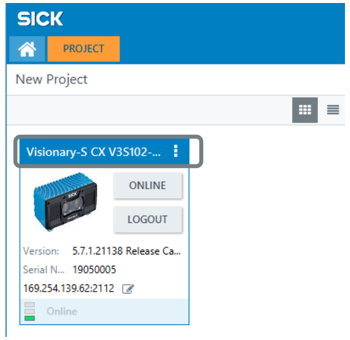
- Precondition: Device is online.

- If not, please check:
  - Physical connection
  - IP address
  - SDD version
    (uninstall and upload form device again)
  - Always start device first, then SOPAS ET
    (to ensure that the communication interface is active)

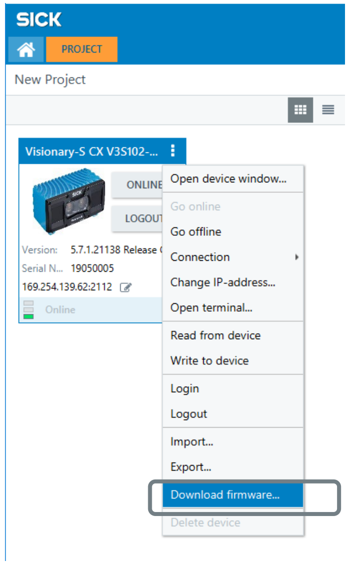


Update
SELECT FILES

- Device must be highlighted:

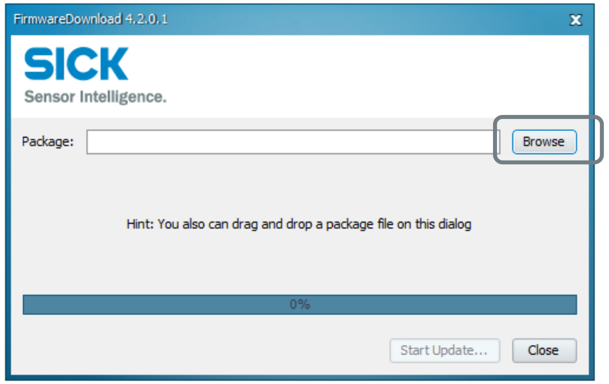


- Select Download firmware:

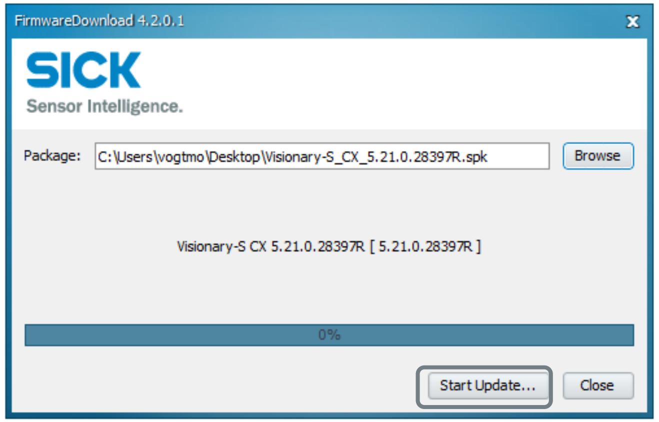


Update
SELECT FILES

- Select a *.spk file

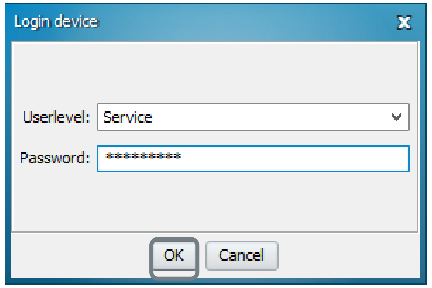


- Start Update

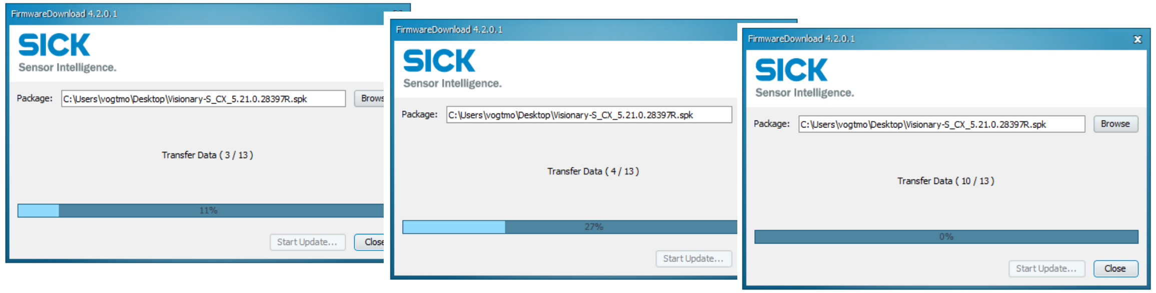


Update
SELECT FILES

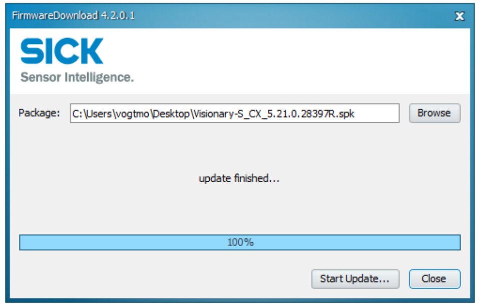
- Confirm **Service** level with password
- Password: **CUST_SERV**



- Wait …

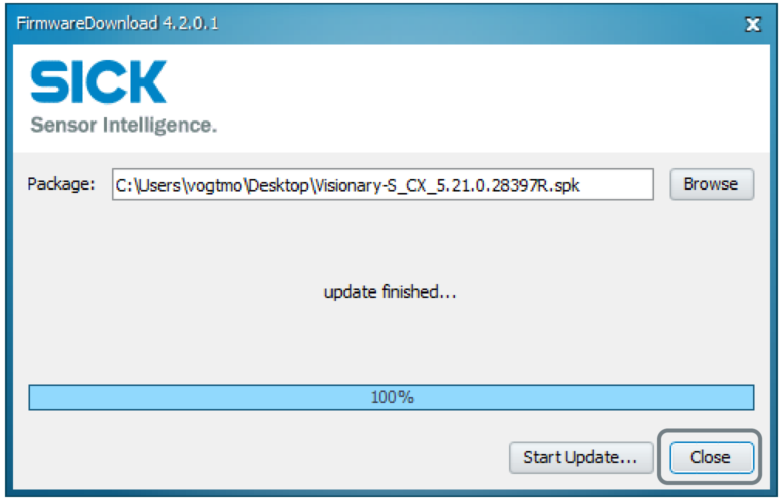


- … until download succeeds

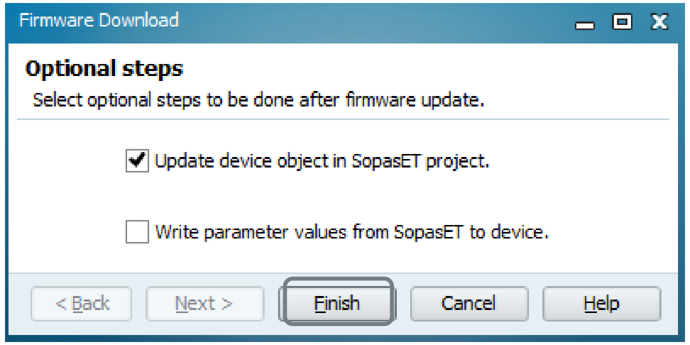


Update
FINISH UPDATE

- Close *Firmware Download* dialog

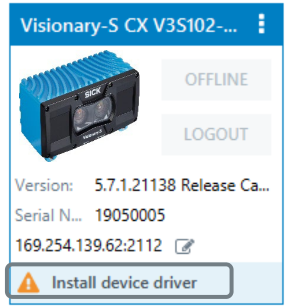


- Choose *Finish*

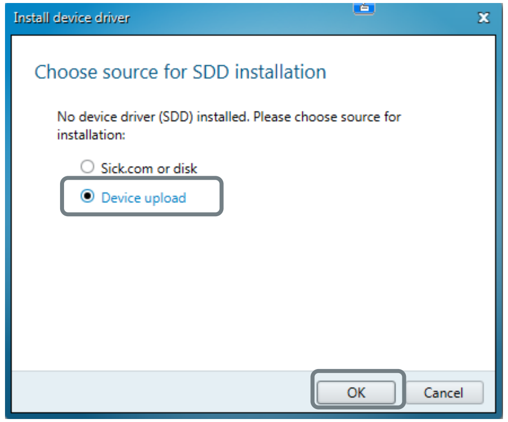


Update
INSTALL NEW DEVICE DESCRIPTION

- After a successful connection the driver might be missing.
- Click on *No driver installed*

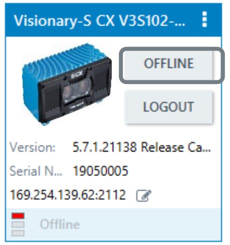


- Choose *Device upload*

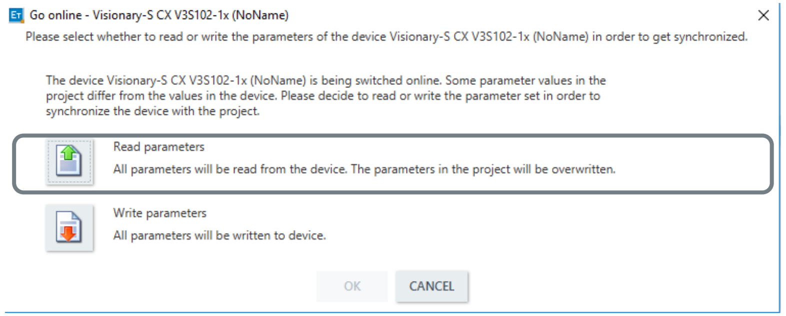


Go Online

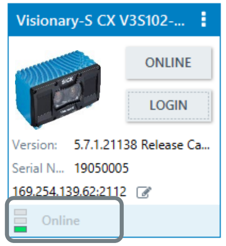
- Click *Offline* to go online



- Choose *Read parameters*



- Success!



Thank you for your attention.

SICK AG – Mobile Perception – 3D Snapshot

2.1.7
3D Coordinate Transformation

Introduction

The SICK snapshot vision sensors have in common that they produce data that can be transformed into 3D point cloud format. However, this computation differs between all these technologies. This document explains how to transform the distance information from the stereo camera Visionary-S into 3D point cloud data.

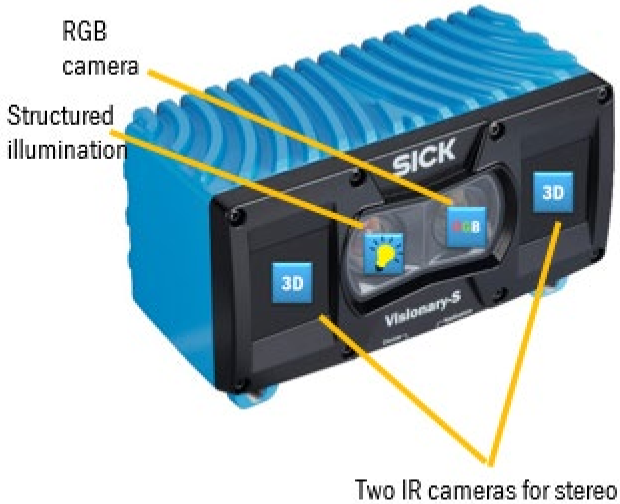
Table of Contents

General concept

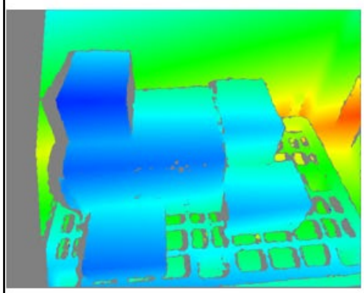
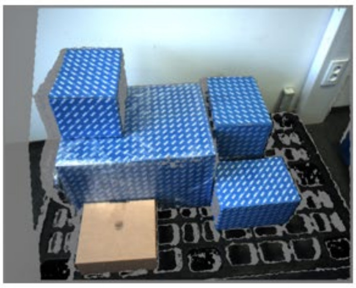
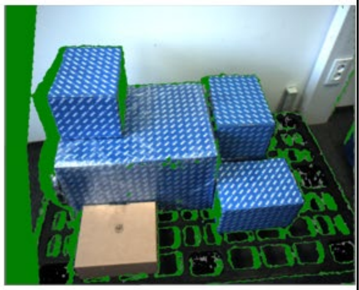
This chapter gather well know imaging concepts that we will follow in order to create 3D point clouds. If you are more interested in how the transformation has to be computed, have a look at From Z-map to point cloud. If you are interested in how to use the example code provided with the camera on the USB stick, please have a look at Hands-On: Using the programing examples.

Image data

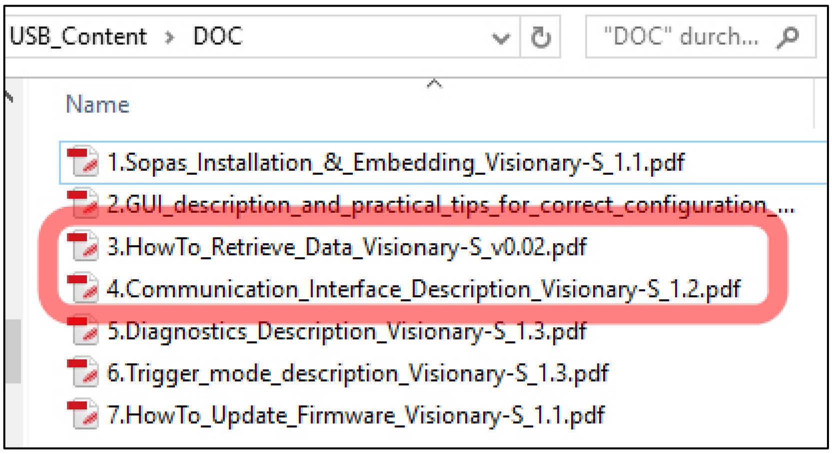
The Visionary-S combines both color and depth perception.



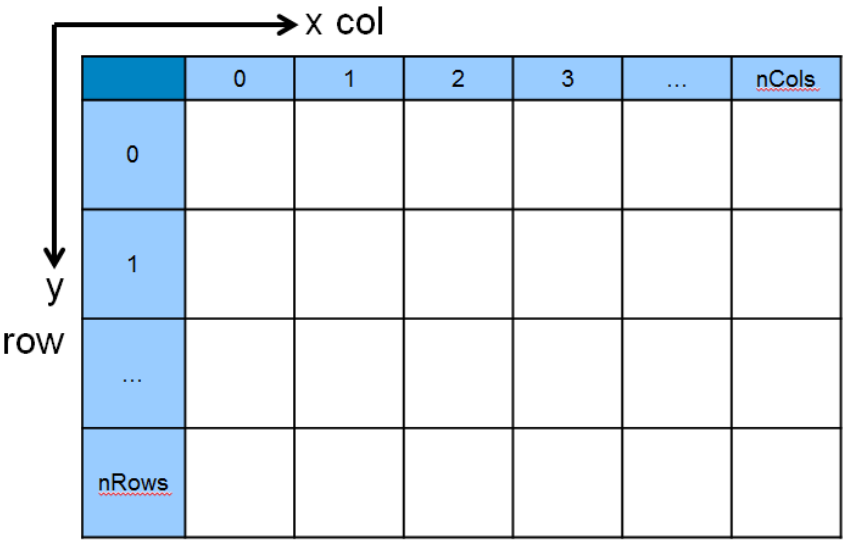
Both depth and color information, plus information about the validity of a pixel are provided by the Visionary-S as three distinctive maps.

Distance values	Intensity RGB values	Filter Map
		
Here color represents the distance to the sensor	Here the color represents the true color of the captured scene	This map contains filter information about the reliability of the pixel values

For a detailed description, how to connect with the Visionary-S and how to grab the data, please have a look at the content of the USB drive, which is delivered together with the Visionary-S.

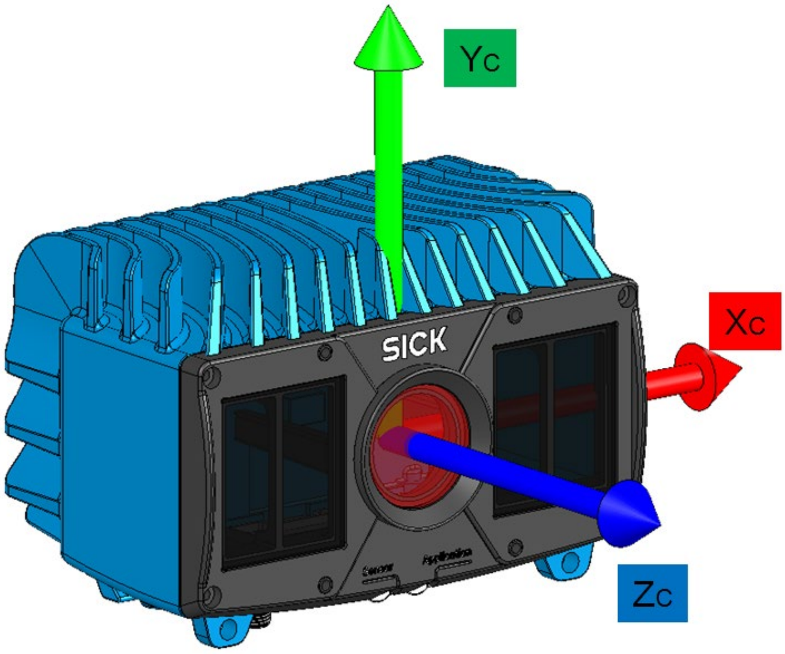


To keep things simple, we will concentrate on the distance values first. See the following figure for an illustration of the pixel matrix. Note that the column and row indices can be named **x** resp. **y** and **col** resp. **row**. We use both nominations.



Camera and local coordinate system

The camera coordinate system (sensor coordinate system) is defined as a right-handed system with Z_C coordinate increasing along the axis from the back to the front side of the sensor, Y_C coordinate increasing vertically upwards and X_C coordinate increasing horizontally to the left, all from the point of view of the sensor(or someone standing behind it). The origin of this coordinate system is the focal point of the image sensor and its z-axis is coincidental with the optical axis.



Derived from this is the local coordinate system. The origin of the coordinate system (0, 0, 0) is a specific reference point, specifically the center of the front face of the camera. To convert from camera to local coordinate system, specific offsets and rotations need to be applied. The local coordinate system is indicated by the index 'L' (x_L, y_L, z_L).

Intrinsic parameters

The intrinsic parameters describe how the data is mapped from the imager chip into the camera coordinate system. Those parameters describe the optical properties of the imaging system. They do not change as long as the optics are fixed, which is the case for our sensors. As those parameters include a mapping from pixel indices (pixel positions on the imager chip) into a metric coordinate system, we have to define the values either in metric units or in relation to the pixel indices.

We use the standard pin-hole camera model as used in OpenCV:camera calibration and 3d reconstruction and described in Szeliski, Richard:Computer Vision Algorithms and Applications. The values provided are the coordinates (c_x, c_y) of the principal point in pixels and the focal lengths f_x, f_y in pixels.

Real lenses have distortions values that need to be compensated. For this reason up to 3 radial distortion coefficients k_1 ... k_3 and 2 tangential distortion coefficients p_1 ... p_2 are provided. The exact number differs between different sensor devices, unused coefficents are always 0. The used model is compatible to the one used by the camera calibration of OpenCV.

Extrinsic parameters

The extrinsic parameters or extrinsic matrix of a sensor describes its pose in the 3D world space. Pose includes position and orientation. We use the term extrinsic matrix in its original mathematical meaning as the transformation matrix from world space to camera space. We refer to a clear description made by Kyle Simek[1].

As the sensor is used in real world applications, the operator should be able to reconsider the camera parameters. We assume that the pose of the sensor is easy to reconsider during the setup. Hence, we use a "camera-centric" parameterization, which describes how the camera changes relative to the world. These parameters correspond to elements of the inverse extrinsic camera matrix which we call *CameraToWorldTransform* (in the BLOB metadata XML description) or resp. *CameraToWorldMatrix*:

$$\begin{matrix} R_C & C \\ 0 & 1 \end{matrix}$$

Note that R_C is a 3x3 rotation matrix that describes the sensor orientation in world coordinates and $C = (C_X, C_Y, C_Z)$ is the 3D translation vector describing the sensor position.

We further define the following:

- n Rotation occurs about the camera's position (more precisely the center of the front face of the sensor which is the origin of the local coordinate system).
- n Rotation occurs in a mathematical positive sense (counterclockwise).
- n Translation and rotation is executed in the following order:

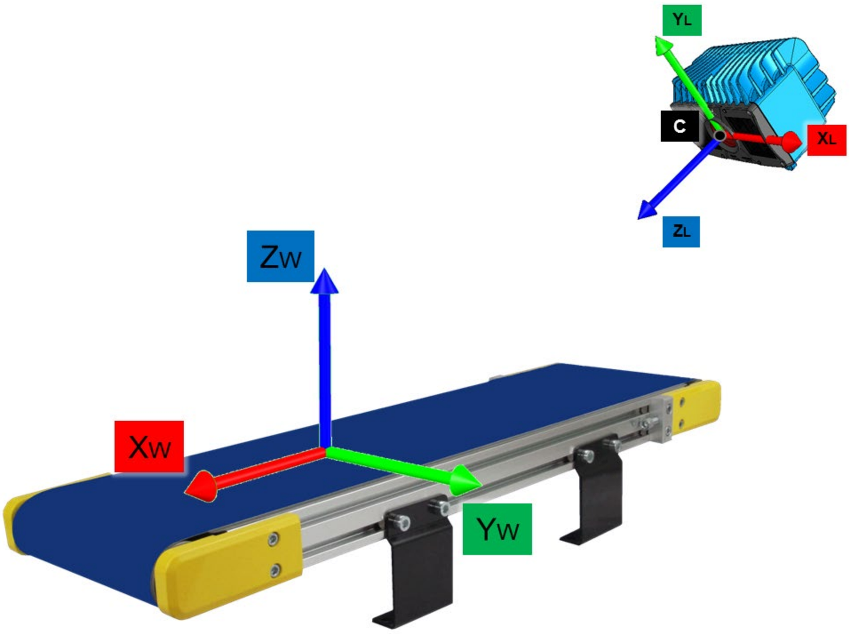
  a) Sensor is in the world origin, axes are aligned with world axes (see Figure 3.8)
  b) Yaw (Rotate around sensor Z_L-axis)
  c) Pitch (rotate around sensor Y_L-axis)
  d) Roll (rotate around sensor X_L-axis)
  e) Apply the translation (add the translation vector to the rotated position)

For the Visionary-S the standard bracket (article no 2077710; illustrated in the figure below) allows a rotation of the device around an axis parallel to the X_L-axis. As this rotation axis is parallel but has not the same origin, the physical rotation leads to a change of the device position.

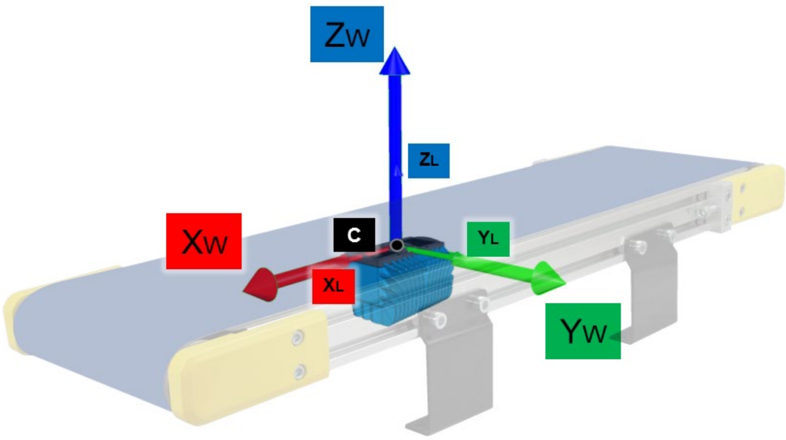
Internally, we only store the CameraToWorldMatrix. This leads to the effect that the mounting orientation angles are not unambiguous. If the user stores a pose in the device, it is stored in the CameraToWorldMatrix. If the orientation angles are loaded from the device, they are computed from the CameraToWorldMatrix. The result of this computation for the three angles might be different from what the user has defined before, however the pose remains the same. It is possible to create a specific orientation by various combinations of rotations.

World coordinate system

The world coordinate system is defined by the user and it is the one the data is visualized in the 3D viewer. We assume that usually the floor or conveyor belt corresponds to the xy-plane and hence the floor normal points in z-direction (see figure below).



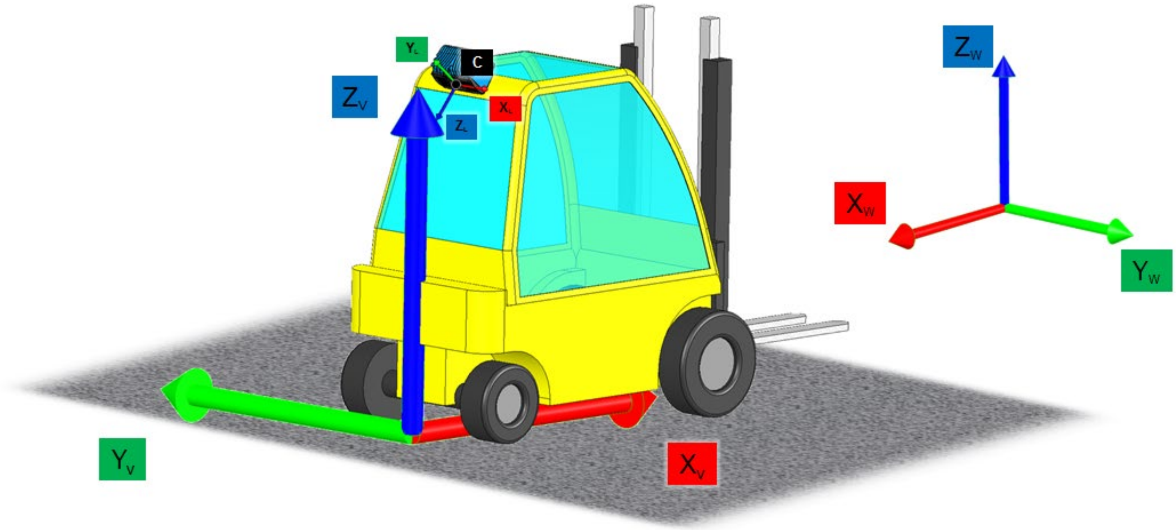
The home position of the sensor is aligned with the world coordinate system so that the viewing direction is along the z-axis. However, the front plane cannot practically be aligned with the xy-plane so that there is an offset of the size of the sensor housing to the local coordinate system and half of the housing to the camera coordinate system.



Vehicle coordinate system

If we know that the device will be attached to some vehicle, we introduce another coordinate system as the pose in the world coordinate system is variable. We call this system vehicle coordinate system (X_V, Y_V, Z_V) and we follow the definitions from Wikipedia Axes conventions/conventions for land vehicles. In the application the pose of the sensor will usually be fixed in relation to the vehicle coordinate system and variable in the world coordinate system.

If the vehicle knows its position in the world coordinate system, it can send this information to the sensor so that it can provide a 3D point cloud in world coordinates as output.



Transformations

Changing the transformation direction

If we have a world to sensor transformation matrix $M_{c \to w}$ (as contained in the camera model), and want to generate world-coordinates from a sensor coordinate point, we need to calculate the inverse of the $M_{c \to w}$.

If $M_{c \to w} = \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix}$

then the inverse $M_{c \to w}^{-1}$ is $\begin{pmatrix} R^T & -R^T t \\ 0 & 1 \end{pmatrix}$

where R^T is the transposed matrix of R.

Composition of transformations

If we have two transformation T_1 and T_2 and want to apply T_2 after T_1 has been applied, the resulting transformation T can be calculated as $T = T2 \cdot T1$ where $\cdot$ is the matrix product. (Note that the later transformation is the first factor).

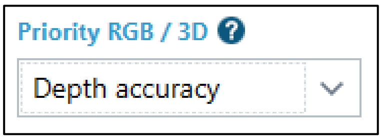
From Z-map to point cloud

A stereo image camera contains two synchronized imagers that capture the same scenery using the same exposure settings at the same point in time. In the following text the two acquired images are called left and right image. The stereo imager uses the effect that the displacement between the same point in the left and the right image is a measure for the z-distance of the point to the imagers. This displacement is called the disparity of the point.

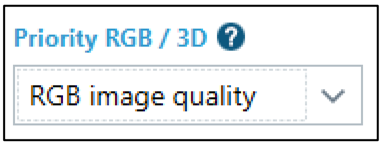
To obtain this disparities in both images first the lens distortion is corrected. Then a common reference system is used and both images are converted to this reference.

In our stereo camera two references are used, depending on how the color information and the distance information are mapped to each other:

- if the color is mapped onto the distance, the focal point and axis of the left imager is used as reference,



- if the distance is mapped onto the color image, the focal point and axis of the color imager is used.



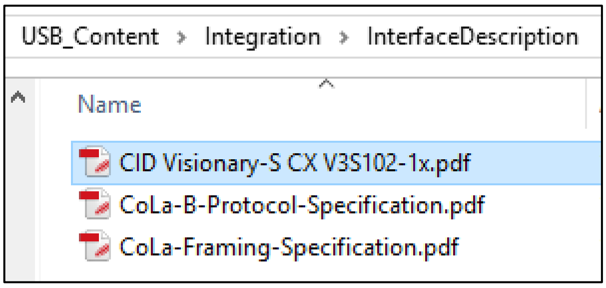
The offset between these two reference points will be corrected using the transformations in the next chapter. Keep in mind, that using the distance maps without this corrections will lead to a visible X shift in the data when toggling the RGB/3D priority modes.

Both left, right and color image are projected into the new reference system. Using the projected left and right images a new image is generated that contains the z-distances calculated from the disparities between the left and right images.

This z image follows a hole camera model whose parameters are those of the reference system. Since the images already are lens-corrected, we do not need any lens correction parameters. The following section explains how to transform the z image back into 3D point coordinates.

Using streamed data

The data format of the camera and how to access it is explained in detail in the CID document located on the USB stick shipped with your Visionary-S



The BLOB XML description section contains the following values:

- the 4x4 camera to world matrix $M_{c \rightarrow w}$
(in 2D row-major layout, see memory layout of multi-dimensional arrays)
- the intrinsic matrix values (c_x, c_y, f_x, f_y) of the reference system.

Further we have the Z-map Z of size n_{rows} x n_{cols}. The maps are in 2D row-major layout as well.

To calculate the camera coordinates (x_C, y_C, z_C) of the point p_C at index (row, col) (row: 0…n_{rows}, col: 0…n_{cols}) the following code can be used.

Distance map to local point coordinates

```
const double xp = (cx - col) / fx;
const double yp = (cy - row) / fy;

const double zc = zmap[row][col];                // zmap is the Z-map
const double xc = xp * zc;
const double yc = yp * zc;
```

Now we need to transform from camera coordinates to world coordinates. The camera to world matrix also includes our camera-to-local transformation, so this means our point in world coordinates (x_W, y_W, z_W) is $p_W = M_{c \rightarrow w} \cdot p_C$.

Sensor to world coordinate system

```
const double xw = m_c2w[0][3] + zc * m_c2w[0][2]  + yc * m_c2w[0][1] + xc * m_c2w[0][0];
const double yw = m_c2w[1][3] + zc * m_c2w[1][2]  + yc * m_c2w[1][1] + xc * m_c2w[1][0];
const double zw = m_c2w[2][3] + zc * m_c2w[2][2]  + yc * m_c2w[2][1] + xc * m_c2w[2][0];
```

If we have not set mounting settings in the HMU, we can omit the (very small) tilt and rotation correction angles for the imager and only concentrate on the displacement relative to the focal point (that are stored in the last column (column 3 if we use a zero based index) of the camera to world matrix.

Simplified sensor to world coordinate system

```
const double xw = m_c2w[0][3] + xc;
const double yw = m_c2w[1][3] + yc;
const double zw = m_c2w[2][3] + zc;
```

Measuring in the Z-map

When using the distance map directly for measuring, be aware that there are some effects that you need to consider.

First: the Z-map is *not* centered around the devices reference point, it is centered around its focus point O_C (x_C, y_C, z_C). This point varies depending on whether you use "color to z" or "z to color" mapping. To obtain the point that represents this focus point, you can use the the translation information from the device reference point (that is the origin of the local coordinate system - for details see the section about the local coordinate system in 3D coordinate transformation) to the focus point O_C. This translation is stored in the last column of the camera to world transformation matrix.

Reference point to center of Z

```
const double dx = m_c2w[0][3];
const double dy = m_c2w[1][3];
const double dz = m_c2w[2][3];
```

What can we do with this information?

If we subtract dz from the Z values in the z-map, we get the distance from the device reference point to the object in the scene. The tuple (dx, dy) where real optical axis (and thus the center of the Z-map) is relative to the device reference point. As an approximation:

- if the color to Z mapping was selected, the optical axis is that of the left image sensor
- same is true if no mapping was selected
- if the Z to color mapping was selected, the optical axis is that of the color image sensor

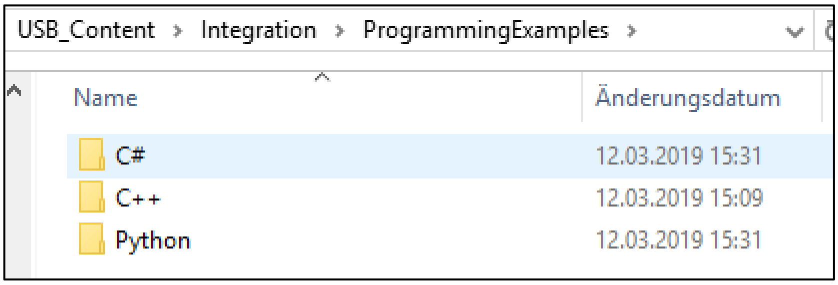
Second: there is an additional error though due to the fact that the intersection point (the principal point) of optical axis of the lens and the image plane of the imager is not perfectly above the center of the image. Due to manufacturing tolerance there is a small variance, that is published via the c_x and c_y values of the intrinsic matrix. This effect will result in a lateral error that grows linear with distance and will be approximately 10mm at 2.5m distance.

To summarize, measuring using the Z-map directly can be used for very fast checks when precision can be sacrificed for speed. In this case the translation between reference point and focal point needs to be considered to interpret the measurements in a sensible way. Only information about the distance between the imager plane and an object (the z coordinate) is readily available. Lateral information (x, y) has the lateral error described above and needs to be interpreted carefully since due to perspective objects further away of course appear smaller than those closer to the camera.

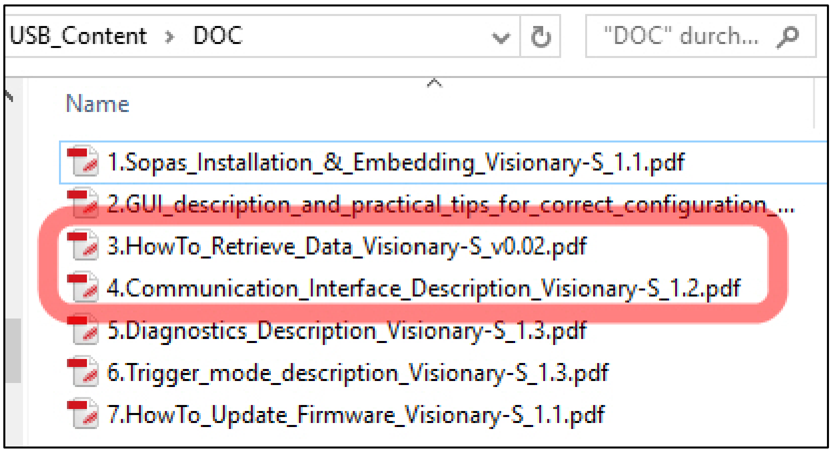
Hands-On: Using the programing examples

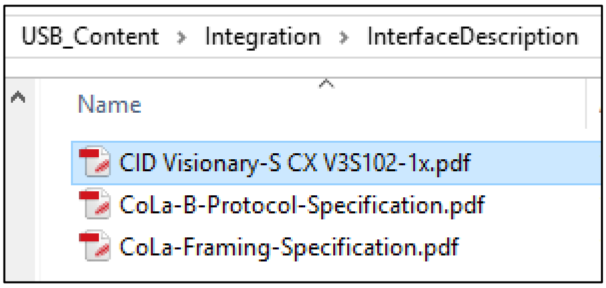
Helpful USB content

For convenience SICK provides programing examples for streaming devices, which contain code to connect, configure and use the camera. They can be found in the USB stick shipped with the Visionary.



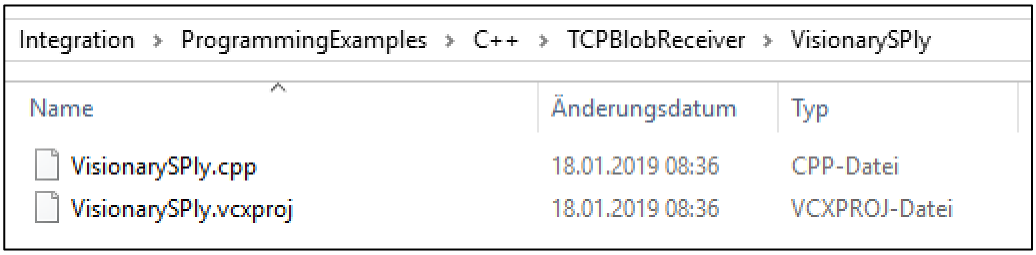
For an overview and references to the available functions, please have a look at the documents described above:



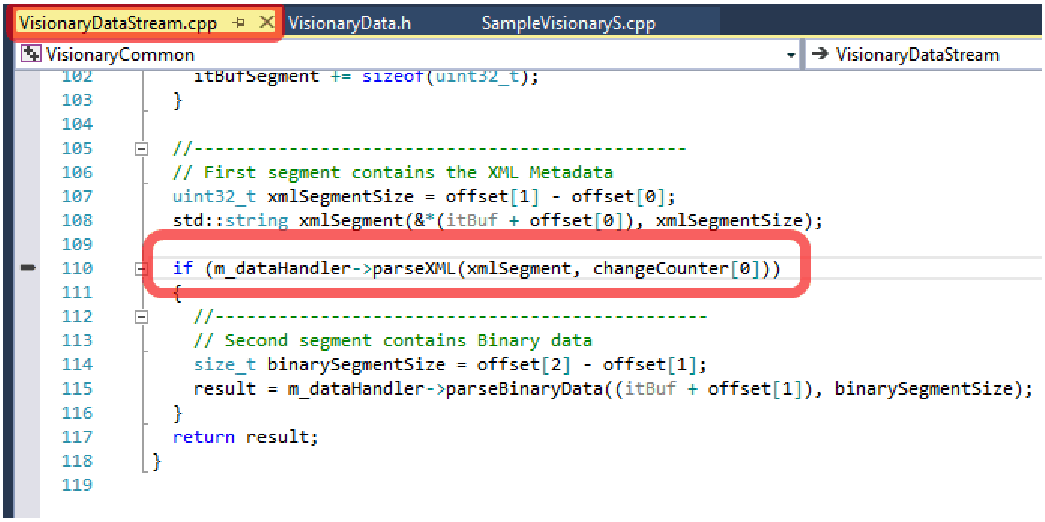


Using C++ to compute point cloud data

An easy and straight forward to present 3D point cloud conversion is the example to convert the camera data to a PLY file:



In the example, the image data and the important XML header containing the camera parameters for the correct camera to world transformation is extracted:



How the XML is extracted from the blob is shown here:

VisionarySData.cpp | VisionaryDataStream.cpp | VisionaryData.h | SampleVisionaryS.cpp

VisionaryCommon | (Globaler Gültigkeitsbereich)

```
}

bool VisionarySData::parseXML(const std::string & xmlString, uint32_t changeCounter)
{
  //-----------------------------------------------
  // Check if the segment data changed since last receive
  if (m_changeCounter == changeCounter)
  {
    return true;  //Same XML content as on last received blob
  }
  m_changeCounter = changeCounter;
  m_preCalcCamInfoType = VisionaryData::UNKNOWN;

  //-----------------------------------------------
  // Build boost::property_tree for easy XML handling
  boost::property_tree::ptree xmlTree;
  std::istringstream ss(xmlString);
  try {
    boost::property_tree::xml_parser::read_xml(ss, xmlTree);
  }
  catch (...) {
    wprintf(L"Reading XML tree in BLOB failed.");
    return false;
  }

  boost::property_tree::ptree dataStreamTree = xmlTree.get_child("SickRecord.DataSets.DataSetStereo.FormatDescriptionDepthMap.DataStream");

  //-----------------------------------------------
  // Extract information stored in XML with boost::property_tree
  m_cameraParams.width = dataStreamTree.get<int>("Width", 0);
  m_cameraParams.height = dataStreamTree.get<int>("Height", 0);

  int i = 0;
  BOOST_FOREACH(const boost::property_tree::ptree::value_type &item, dataStreamTree.get_child("CameraToWorldTransform")) {
    m_cameraParams.cam2worldMatrix[i] = item.second.get_value<double>(0.);
    ++i;
  }

  m_cameraParams.fx = dataStreamTree.get<double>("CameraMatrix.FX", 0.0);
  m_cameraParams.fy = dataStreamTree.get<double>("CameraMatrix.FY", 0.0);
  m_cameraParams.cx = dataStreamTree.get<double>("CameraMatrix.CX", 0.0);
  m_cameraParams.cy = dataStreamTree.get<double>("CameraMatrix.CY", 0.0);

  m_cameraParams.k1 = dataStreamTree.get<double>("CameraDistortionParams.K1", 0.0);
  m_cameraParams.k2 = dataStreamTree.get<double>("CameraDistortionParams.K2", 0.0);
  m_cameraParams.p1 = dataStreamTree.get<double>("CameraDistortionParams.P1", 0.0);
  m_cameraParams.p2 = dataStreamTree.get<double>("CameraDistortionParams.P2", 0.0);
  m_cameraParams.k3 = dataStreamTree.get<double>("CameraDistortionParams.K3", 0.0);

  m_cameraParams.f2rc = dataStreamTree.get<double>("FocalToRayCross", 0.0);

  m_zByteDepth = getItemLength(dataStreamTree.get<std::string>("Z", ""));
  m_rgbaByteDepth = getItemLength(dataStreamTree.get<std::string>("Intensity", ""));
  m_confidenceByteDepth = getItemLength(dataStreamTree.get<std::string>("Confidence", ""));

  m_distanceDecimalExponent = dataStreamTree.get<int>("Z.<xmlattr>.decimalexponent", 0);

  return true;
```

Having extracted the XML information, the XYZ transformation can be done as shown below. The code below was written to work with both the Visionary-S and the Visionary-T, which have different technologies. For Visionary-S, the function will work out like the formula shown in Distance map to local point coordinates.

```
void VisionaryData::preCalcCamInfo(const ImageType& imgType)
{
  assert(imgType != UNKNOWN);     // Unknown image type for the point cloud transformation

  m_preCalcCamInfo.reserve(m_cameraParams.height * m_cameraParams.width);

  //-----------------------------------------------
  // transform each pixel into Cartesian coordinates
  for (int row = 0; row < m_cameraParams.height; row++)
  {
    double yp = (m_cameraParams.cy - row) / m_cameraParams.fy;
    double yp2 = yp * yp;

    for (int col = 0; col < m_cameraParams.width; col++)
    {
      // we map from image coordinates with origin top left and x
      // horizontal (right) and y vertical
      // (downwards) to camera coordinates with origin in center and x
      // to the left and y upwards (seen
      // from the sensor position)
      const double xp = (m_cameraParams.cx - col) / m_cameraParams.fx;

      // correct the camera distortion
      const double r2 = xp * xp + yp2;
      const double r4 = r2 * r2;
      const double k = 1 + m_cameraParams.k1 * r2 + m_cameraParams.k2 * r4;

      // Undistorted direction vector of the point
      const float x = static_cast<float>(xp * k);
      const float y = static_cast<float>(yp * k);
      const float z = 1.0f;
      double s0 = 0;
      if (RADIAL == imgType)
      {
        s0 = sqrt(x * x + y * y + z * z) * 1000;
      }
      else if (PLANAR == imgType)
      {
        s0 = 1000;
      }
      else
      {
        assert(!"Unknown image type for the point cloud transformation");
      }
      PointXYZ point;
      point.x = static_cast<float>(x / s0);
      point.y = static_cast<float>(y / s0);
      point.z = static_cast<float>(z / s0);

      m_preCalcCamInfo.push_back(point);
    }
  }
```

And as final step the Sensor to world coordinate system:

```cpp
void VisionaryData::transformPointCloud(std::vector<PointXYZ> &pointCloud) const
{
  // turn cam 2 world translations from [m] to [mm]
  const double tx  = m_cameraParams.cam2worldMatrix[3]  / 1000.;
  const double ty  = m_cameraParams.cam2worldMatrix[7]  / 1000.;
  const double tz  = m_cameraParams.cam2worldMatrix[11] / 1000.;

  for (std::vector<PointXYZ>::iterator it = pointCloud.begin(), itEnd = pointCloud.end(); it != itEnd; ++it)
  {
    const double x = it->x;
    const double y = it->y;
    const double z = it->z;

    it->x = static_cast<float>(x * m_cameraParams.cam2worldMatrix[0] + y * m_cameraParams.cam2worldMatrix[1] + z * m_cameraParams.cam2worldMatrix[2] + tx);
    it->y = static_cast<float>(x * m_cameraParams.cam2worldMatrix[4] + y * m_cameraParams.cam2worldMatrix[5] + z * m_cameraParams.cam2worldMatrix[6] + ty);
    it->z = static_cast<float>(x * m_cameraParams.cam2worldMatrix[8] + y * m_cameraParams.cam2worldMatrix[9] + z * m_cameraParams.cam2worldMatrix[10] + tz);
  }
}
```

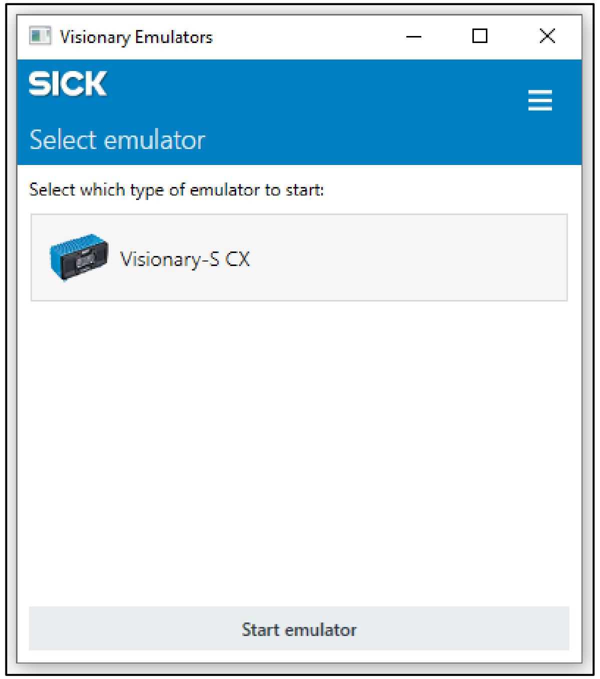
2.1.8
Emulator Description

SICK AG – Mobile Perception – 3D Snapshot

Content

Emulator
OVERVIEW

- Using the emulator a previously recorded SSR file can be played back to simulate a Visionary camera.
- Typical scenes can be recorded to find optimal filter settings
- Solution development without the need of a physical device close by
- Convenient to repetitively test data algorithmson real life test data



Run the emulator
OPEN THE EMULATOR

SICK
Sensor Intelligence.

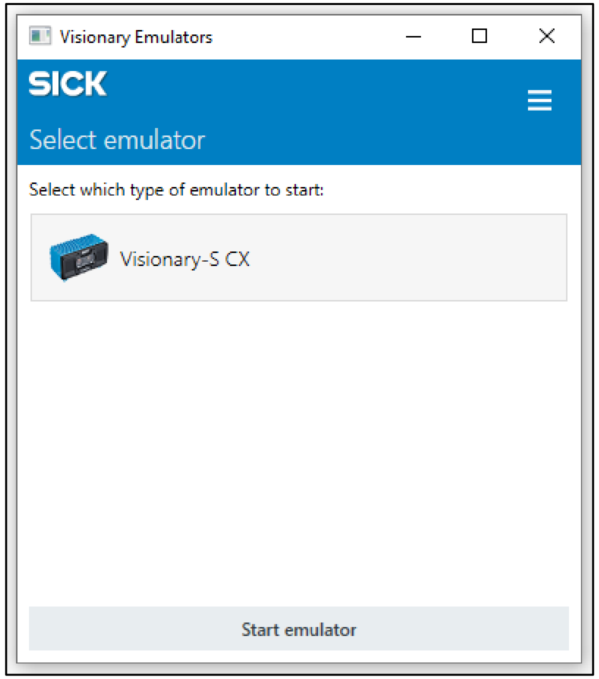
1. Locate the folder “EmulatorManager” on the USB drive

2. Start “*V3SEmulatorManager.exe*”

   - Assets
   - Emulators
   - config.json
   - debug.log
   - Newtonsoft.Json.dll
   - NLog.config
   - NLog.dll
   - V3SEmulatorManager.exe
   - V3SEmulatorManager.exe.config
   - Visionary.NET.Shared.dll

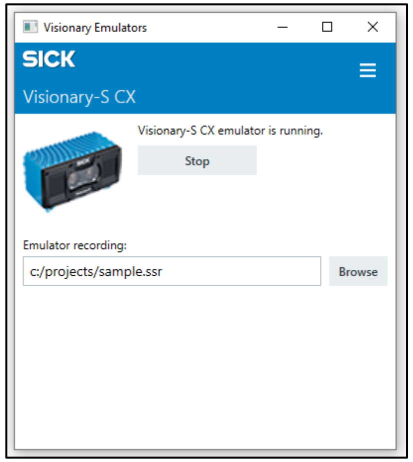
3. Select the device you want to emulate

4. Press “*Start emulator*”



Run the emulator
SELECT RECORDS

5. Type in the path to the SSR file you want to use for emulation or click "*Browse*"



6. Open "Sopas ET"

Use the emulator in SOPAS ET

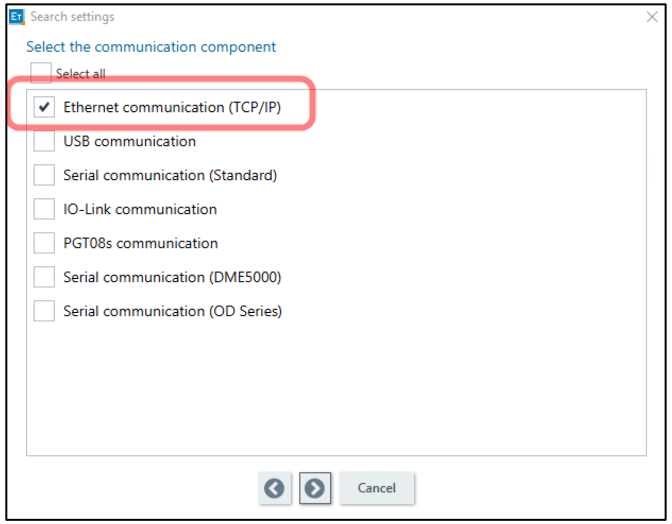
FIND EMULATED DEVICE

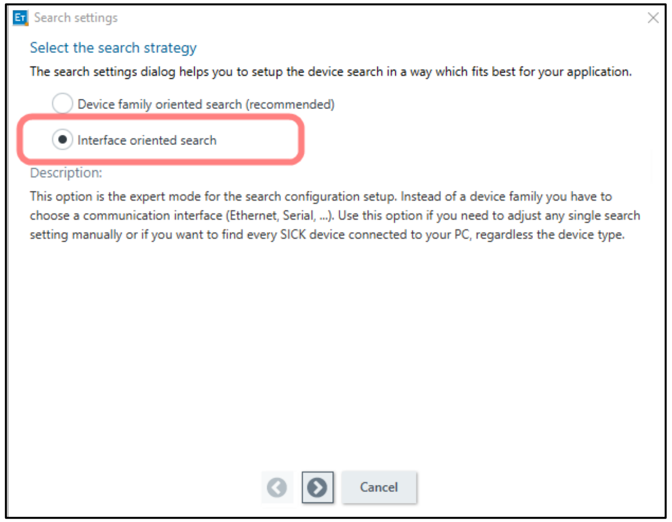
7. By default, the emulated device does not show when starting SOPAS ET

8. Click on "*Search settings*" (lower right corner)

9. Select "*Interface oriented search*"

10. Select "*Ethernet communication (TCP/IP)*"





Use the emulator in SOPAS ET

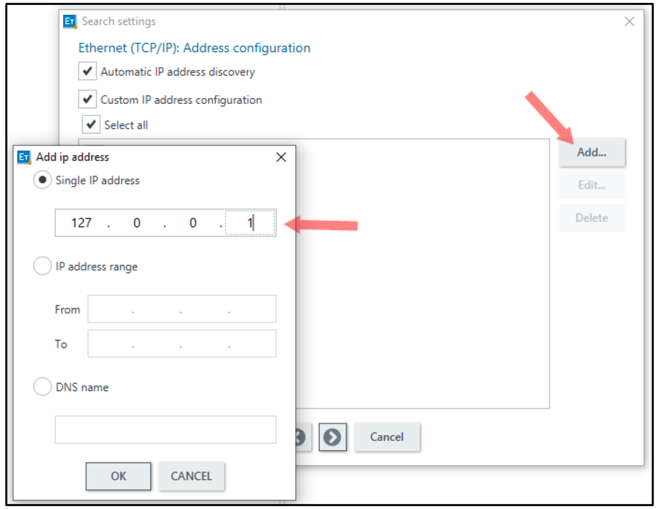
FIND EMULATED DEVICE

11. Check mark "*Custom IP address configuration*"

12. Press "*Add...*"

13. Type in local host IP address **127.0.0.1**

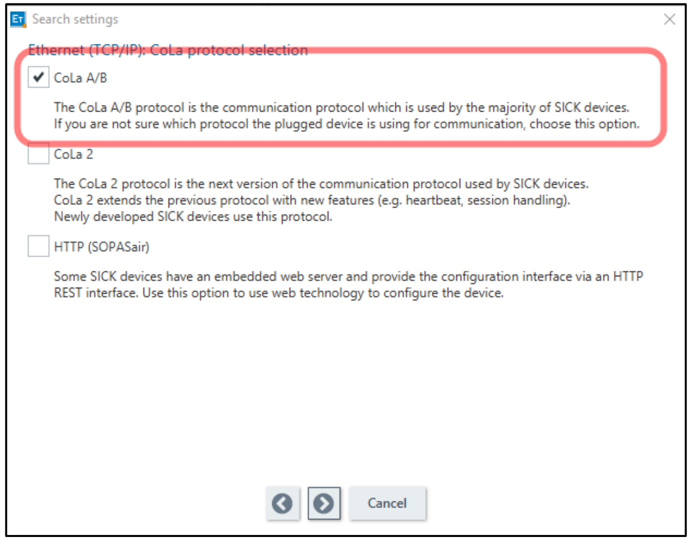
14. Press OK



Use the emulator in SOPAS ET

FIND EMULATED DEVICE

11. Choose “CoLa A/B” protocol



12. Confirm “Advanced search settings”

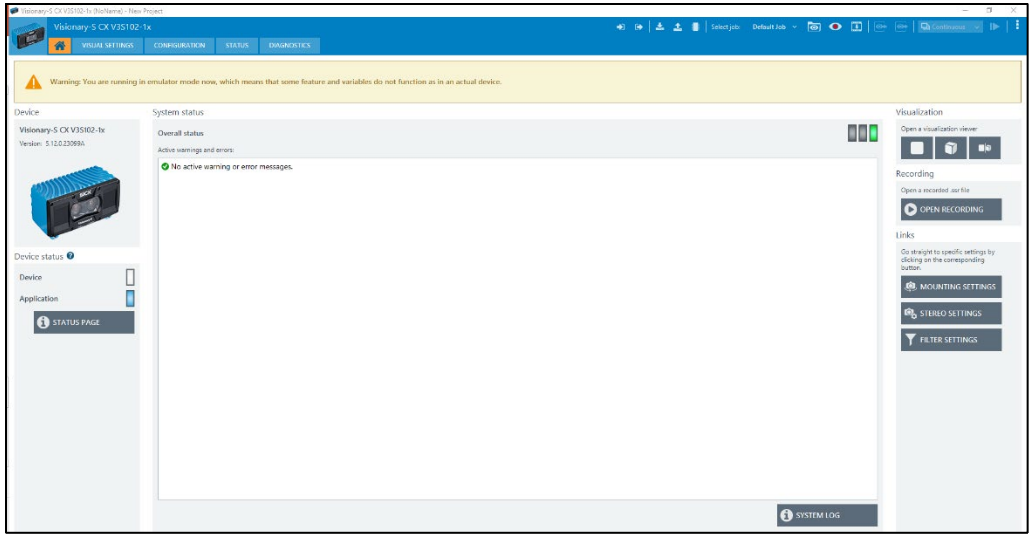
13. Name the search and click “Finish”

Use the emulator in SOPAS ET

INSTALL DEVICE DRIVERS

14. Install device drivers (use device upload)

15. Open device page (you should see a “emulator in use warning”)



Use the emulator in SOPAS ET
FEATURES AND RESTRICTIONS

Features

- Change mounting settings
- Add or change filter
- Grab raw data stream

Known restrictions

- Some settings which cannot be adjusted:
  - Time-of-flight/Stereo settings
  - Hardware based filter settings (Reliability filter, median filter…)
  - Hardware trigger in- and outputs

Thank you for your attention.

SICK AG – Mobile Perception – 3D Snapshot

SICK
Sensor Intelligence.

2.1.9
SD Card Cloning

SICK AG – Mobile Perception – 3D Snapshot

Content

SD Card Cloning

DISCLAIMER

- Only qualified personal is allowed to open the “Service” cover.
- Please make sure that the Visionary is not supplied with power and running. Otherwise please turn it off before opening the “Service” cover.
- The IP67 protection class can fail in case of improperly use.
- Please make sure that the SD Card is formatted in a linux compatible file system e.g. FAT, EXT
- The file system NTFS is not supported.

SD Card Cloning
OVERVIEW

SD Card Cloning:

- Easily clone available configuration on different devices.
- User friendly tool for cost effective service.
- Completes effective exchange device service.

- **Note:**
  Please make sure that the Visionary is not supplied with power and running.
  Otherwise please turn it off before opening the “Service” cover.

SD Card Cloning
CLONE CONFIGURATION

Clone configuration:

- Open the “Service” cover with a TX8 Tool and insert a mini SD card including a valid configuration set for the Visionary.
- Start the device.
- The configuration on the SD card will be transfer to the device automatically during the start up of the device.
- Turn off device and remove the SD card.
- Fix both M2 torx screw with 60Ncm torque

SD Card Cloning

SAVE CONFIGURATION

Save configuration:

- Open the “Service” cover with a TX8 Tool and insert an **empty** mini-SD card.
- Start the device.
- The configuration of the Visionary will be saved to the SD card automatically during the start up of the device or press the SOPAS button to write the configuration to the SD card.
- Turn off device and remove the SD card.
- Fix both M2 torx screw with 60Ncm torque
- It is possible to keep the SD card within the Visionary.
- Please make sure to keep the “Service” cover always closed during the operation.

Thank you for your attention.

SICK AG – Mobile Perception – 3D Snapshot