Taprisiot Application User Manual

Abstract	This documents is the user manual of the ConoscopeDemo application
Version	0.4
Status	Draft
Date	2020/11/18

Revision history

Version	Date	Content
0.1	2020/06/03	Initial version
0.2	2020/06/25	Add "log file" chapter
0.3	2020/07/01	Add export file naming options
		Add ROI
		Add AutoExposure Measurement Area
		Add AutoExposure level and exposure time thresholds
		Add CaptureSequence SaveCapture option
		Add ConvertRaw feature
	2020/07/02	Add AE exposure time granularity
	2020/07/06	Add <AeExpoGran> tag in ExportFormat name
0.4	2020/11/18	Add streaming options

ELDIM
1185 Rue d'Epron (Ancienne)
14200 Hérouville Saint-Clair
France

ELDIM, the ELDIM logo and other product names referenced herein are trademarks of ELDIM.
Other product names, designations, logos, and symbols are trademarks or registered trademarks of their respective owners.

Table of contents

1   Introduction

Conoscope Application drives the taprisiot device.

The application handles following features:
- Calibration data
- Taprisiot configuration: temperature and filter selection
- Capture: with specific exposure time or auto exposure option
- Export data: Raw Data or Processed data
- Capture Sequence: Capture required images to generate X, Y, Z images

2   Install

2.1   Requirement

Euresys drivers to drive the Frame Grabber: coaxlink-win10-x86_64-11.0.3.82.exe
Once install, launch GenICam browser. The application may request to update frame grabber firmware.
Please select Coaxlink Quad G2 (2 camera), update firmware. Once done shut down and unplug the PC to be sure the frame grabber board is properly switched off.

2.2   Application

The application is **ConoscopeDemo.exe**. The folder contains all required dependencies.

iconengines	27/05/2020 16:35	File folder	
imageformats	27/05/2020 16:35	File folder	
platforms	27/05/2020 16:35	File folder	
styles	27/05/2020 16:35	File folder	
translations	27/05/2020 16:35	File folder	
CaptureSequenceExposureTime.json	06/04/2020 11:51	JSON File	1 KB
config.json	28/04/2020 18:09	JSON File	2 KB
ConoscopeDemo.exe	27/05/2020 16:34	Application	357 KB
ConoscopeLib.dll	27/05/2020 16:33	Application extens...	780 KB
D3Dcompiler_47.dll	11/03/2014 11:54	Application extens...	4 077 KB
demo.json	28/04/2020 18:09	JSON File	1 KB
libEGL.dll	27/03/2020 14:43	Application extens...	24 KB
libGLESv2.dll	27/03/2020 14:43	Application extens...	3 519 KB
opengl32sw.dll	14/06/2016 14:00	Application extens...	20 433 KB
PipelineLib.dll	27/05/2020 16:32	Application extens...	49 KB
Qt5Core.dll	27/03/2020 14:43	Application extens...	6 217 KB
Qt5Gui.dll	27/03/2020 14:43	Application extens...	6 961 KB
Qt5Svg.dll	27/03/2020 19:47	Application extens...	329 KB
Qt5Widgets.dll	27/03/2020 14:43	Application extens...	5 425 KB
Qt5Xml.dll	27/03/2020 14:43	Application extens...	200 KB

2.3 User interface
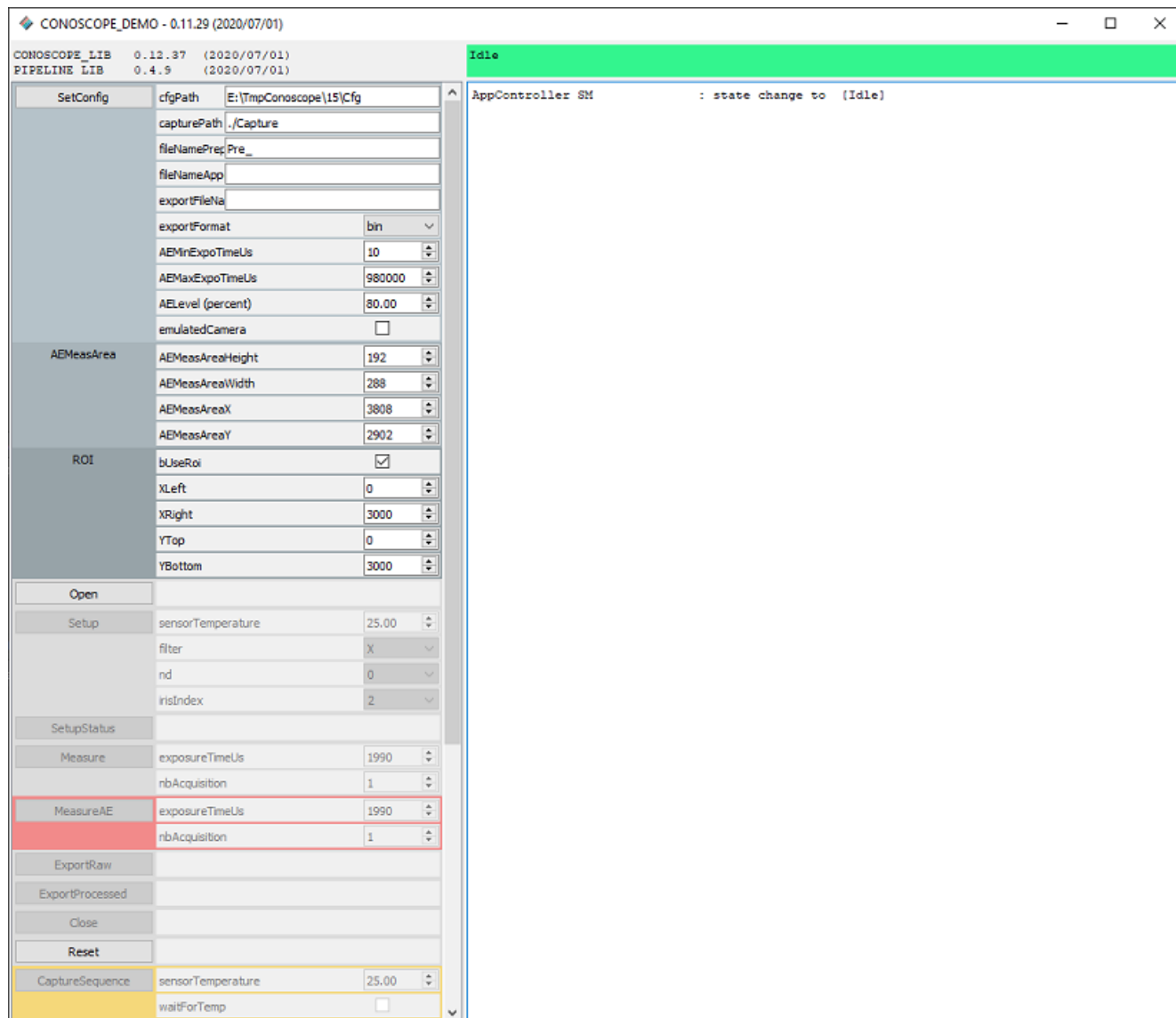
Warning:
When launching the application, it might be possible that an error pops-up indicating that a dll is missing.
Please install vcredist_x64.exe and/or vc_redist.x64.exe

3 Application version

The application version and libraries version are indicated on the interface:
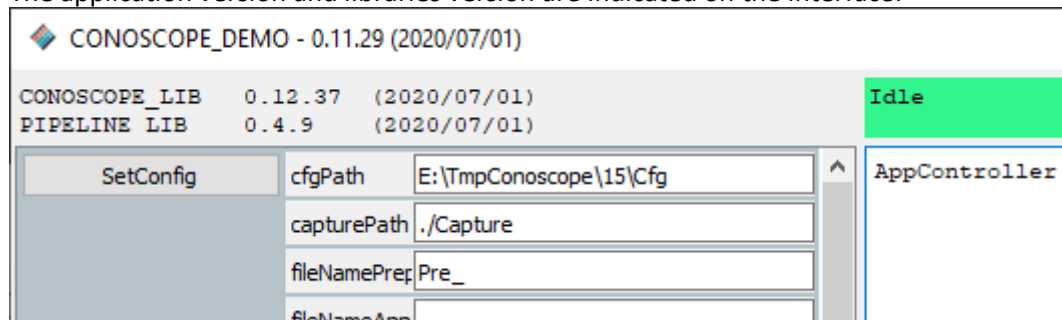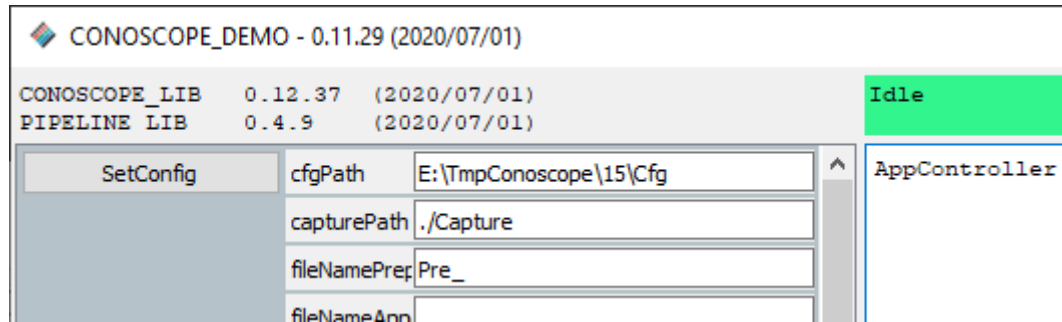
4   Calibration Data

To run properly, the application must access calibration data of the Taprisiot.
The folder where the calibration data are stored can be modified using SetConfig command:
set path in **cfgPath** and press **SetConfig**

Note: It is only possible to change the Calibration data folder after application start-up. Once Taprisiot is open, the folder can not be changed any more.

Calibration Data folder is populated by 2 ways:

- Manually. The user has calibration data and can copy it in the appropriate folder
- From Taprisiot: Data are stored in the memory of the Taprisiot and can be retrieved.

4.1   Manually

Copy calibration data into the folder.

remark: the subfolder matches with the camera SN (not the serial number of the taprisiot)

To know the camera SN, press **Open** command. The Camera SN is displayed in the log.
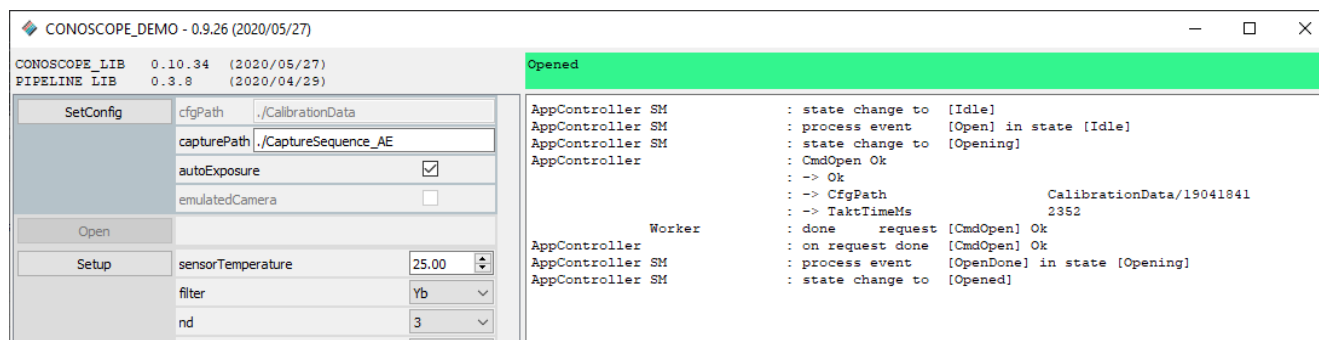
Name	Date modified	Type	Size
CAMERA_19041841.cfg	26/03/2020 12:14	CFG File	27 KB
CAMERA_19041841_prnu.bin	26/03/2020 12:14	BIN File	92 875 KB
FlatField_iris_2_filter_IrCut.bin	06/03/2020 10:52	BIN File	70 336 KB
FlatField_iris_2_filter_X.bin	04/03/2020 17:20	BIN File	70 336 KB
FlatField_iris_2_filter_Xz.bin	04/03/2020 17:20	BIN File	70 336 KB
FlatField_iris_2_filter_Ya.bin	04/03/2020 17:20	BIN File	70 336 KB
FlatField_iris_2_filter_Yb.bin	04/03/2020 17:20	BIN File	70 336 KB
FlatField_iris_2_filter_Z.bin	04/03/2020 17:20	BIN File	70 336 KB
FlatField_iris_3_filter_IrCut.bin	06/03/2020 10:53	BIN File	70 336 KB
FlatField_iris_3_filter_X.bin	04/03/2020 17:20	BIN File	70 336 KB
FlatField_iris_3_filter_Xz.bin	04/03/2020 17:20	BIN File	70 336 KB
FlatField_iris_3_filter_Ya.bin	04/03/2020 17:20	BIN File	70 336 KB
FlatField_iris_3_filter_Yb.bin	04/03/2020 17:20	BIN File	70 336 KB
FlatField_iris_3_filter_Z.bin	04/03/2020 17:20	BIN File	70 336 KB
FlatField_iris_4_filter_IrCut.bin	06/03/2020 10:53	BIN File	70 336 KB
FlatField_iris_4_filter_X.bin	04/03/2020 17:20	BIN File	70 336 KB
FlatField_iris_4_filter_Xz.bin	04/03/2020 17:20	BIN File	70 336 KB
FlatField_iris_4_filter_Ya.bin	04/03/2020 17:20	BIN File	70 336 KB
FlatField_iris_4_filter_Yb.bin	04/03/2020 17:20	BIN File	70 336 KB
FlatField_iris_4_filter_Z.bin	04/03/2020 17:21	BIN File	70 336 KB
FlatField_iris_5_filter_IrCut.bin	06/03/2020 10:53	BIN File	70 336 KB
FlatField_iris_5_filter_X.bin	04/03/2020 17:21	BIN File	70 336 KB
FlatField_iris_5_filter_Xz.bin	04/03/2020 17:21	BIN File	70 336 KB
FlatField_iris_5_filter_Ya.bin	04/03/2020 17:21	BIN File	70 336 KB
FlatField_iris_5_filter_Yb.bin	04/03/2020 17:21	BIN File	70 336 KB
FlatField_iris_5_filter_Z.bin	04/03/2020 17:21	BIN File	70 336 KB
OpticalColumn.xml	09/03/2020 12:55	XML Document	8 KB

4.2 Download Calibration data from Taprisiot device

calibration data is stored in the camera. It is possible to populate the folder with those values.

Remark: This upload takes a lot of time (around 30 minutes)

4.2.1 open
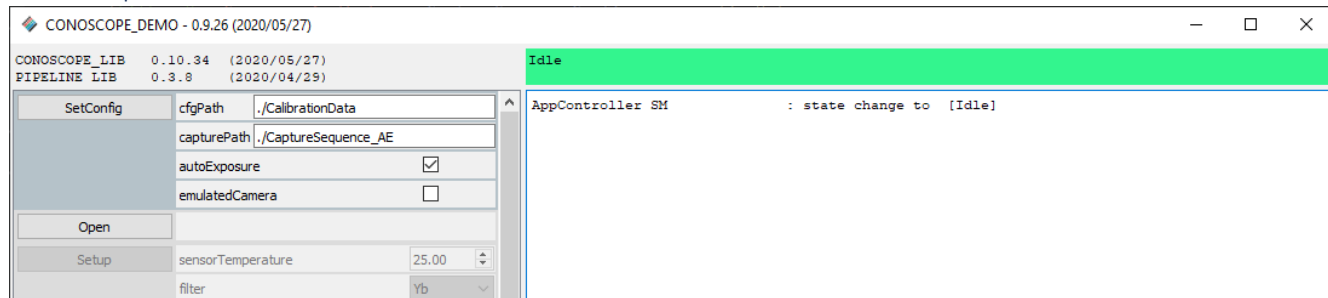
4.2.2 Read data

Launch read using **CfgFileRead command**
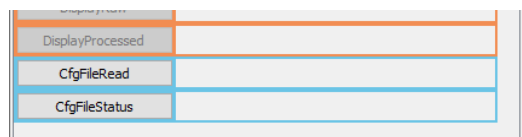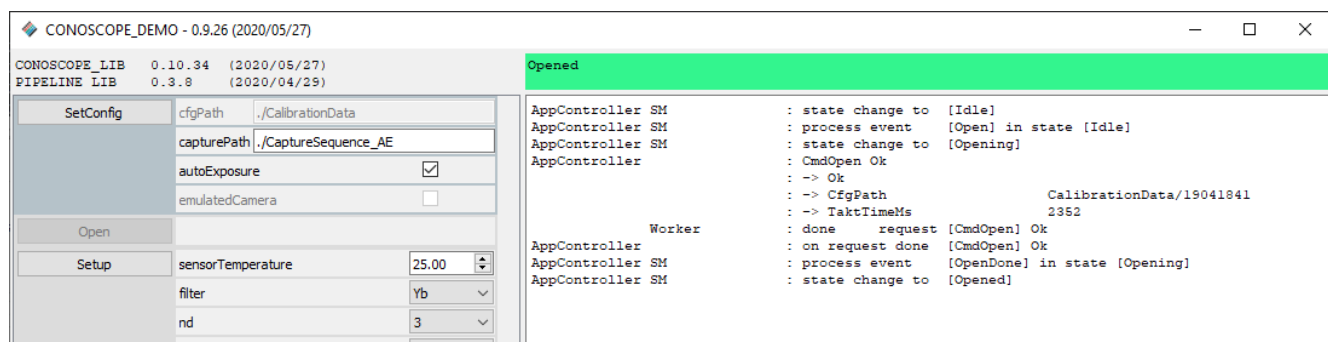
4.2.3 Read Status

Then regularly check Cfg with **CfgFileStatus** command until the process is complete.

5 Capturing an image

5.1 Configuring capture folder

When an image is captured, it is stored in a specific folder.

Change the path in **capturePath** and press **SetConfig**
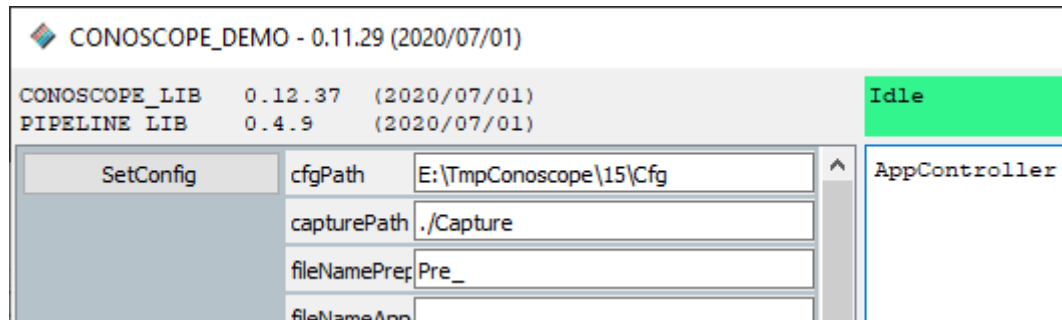
5.2 Open the device

Initially Taprisiot is in **Idle** state.

Press **Open** to bring the device in opened state.
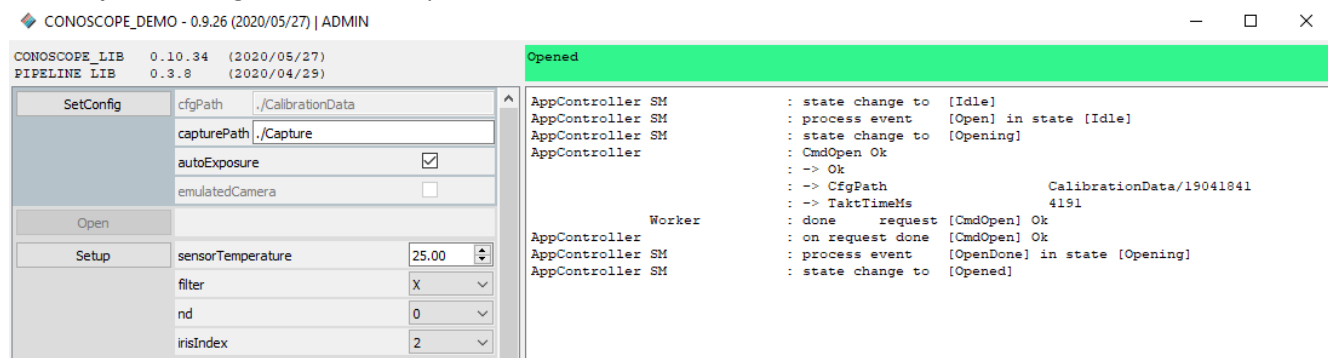
5.3 Setup

Once opened, it is possible to setup the device

- Targeted temperature: the device will be regulated at this temperature
- Filter: BK7, Mirror, X, Xz, Ya, Yb, Z, IrCut
- Nd filter: density 0, 1, 2, 3, 4
- Iris: aperture: 2mm, 3mm, 4mm, 5mm. **Aperture must be changed manually**.

Press **Setup** to go in ready state

Current configuration can be retrieved with **SetupStatus** command

5.4 Image Capture

5.4.1 Measure command

Measure command will capture the image at the specified ExposureTime

5.4.2 MeasureAE command

MeasureAE will capture the image after an AutoExposure sequence.

During MeasureAE processing, it is possible to know the status of the processing (**MeasureAEStatus**) and to stop the processing (**MeasureAECancel**)

MeasureAE command works with several iterations and will take more time than Measure command. During the algorithm, an image is captured and processed.
The brightest pixel of the measurement area is measured, and the exposure time is tuned so this value matches a specified level of saturation. By default, the exposure time is tuned so the return value is 80% of saturation.

Measure AE Parameters:

Measurement Area | Defines the area where the measurement is done.
note: the bigger the area is, the slower the algorithm is.

AEMeasAreaHeight   [0, 6004]
AEMeasAreaWidth   [0, 7920]
AEMeasAreaX   [0, 7920]
AEMeasAreaY   [0, 6004]

Set All parameters to 0 to disable

Level | Defines the target of the algorithm.
If Level is set to 80%, the exposure time is calculated so the brightest pixel of the measurement area is 80% of the saturation
AELevel (percent)

Threshold | Boundary in the exposure time (in micro seconds)
AEMinExpoTimeUs
AEMaxExpoTimeUs

Granularity | Defines the granularity of the Exposure Time calculated by the algorithm.
AEExpoTimeGranularityUs

5.4.3 Measure output

Measure Commands hold data internally.
To store capture, use command **ExportRaw** or **ExportProcessed**.
To display capture, use **DisplayRaw** or **DisplayProcessed**.

5.5 Export Capture

From state MeasureDone, use Commands **ExportRaw** and **ExportProcessed** to store capture in binary format. Those commands do not perform a capture (Measure commands does, see previous chapter).

5.5.1 Export Capture options:

```
◆ CONOSCOPE_DEMO - 0.11.29 (2020/07/01)

CONOSCOPE_LIB   0.12.37  (2020/07/01)          Opened
PIPELINE LIB    0.4.9    (2020/07/01)
```

SetConfig	cfgPath	E:\TmpConoscope\15\Cfg
	capturePath	./Capture
	fileNamePrep	Pre_
	fileNameApp	
	exportFileNa	
	exportFormat	bin
	AEMinExpoTimeUs	10
	AEMaxExpoTimeUs	980000
	AELevel (percent)	80.00
	emulatedCamera	☐
AEMeasArea	AEMeasAreaHeight	192
	AEMeasAreaWidth	288
	AEMeasAreaX	3808
	AEMeasAreaY	2902
ROI	bUseRoi	☑
	XLeft	0
	XRight	3000
	YTop	0
	YBottom	3000

capturePath   Folder where capture are exported

ROI   Defines the Region of interest of the exported image. (this apply only on processed capture)

```
        0    XLeft         XRight
    0   +---+------------+---+
        |                    |
YTop +      ++++++++++++     |
        |   ++++++++++++     |
        |   ++++++++++++     |
        |   ++++++++++++     |
YBot +      ++++++++++++     |
        |                    |
        +--------------------+
```

Each value in range [0, 6001]

fileNamePrepend	Add a string the beginning of the exported file name
fileNameAppend	Append a string at the end of the exported file name
ExportFormat	Specify the format of the exported file name

                    <TimeStamp>
                    <Filter>
                    <Nd>
                    <Iris>
                    <ExpoTime>
                    <NbAcq>
                    <Height>
                    <Width>
                    <SatFlag>
                    <SatLevel>
                    <AeExpoGran>
              default format
                    <TimeStamp>_filt_<Filter>_nd_<Nd>_iris_<Iris>

5.5.2   Raw Data Format

A capture is .bin file associated to a .json file containing information about the configuration of the device.
A binary file can be opened using ImageJ

File > Import > Raw
    Image type :   16 bits unsigned
    Width :   7920
    Height: 6004
    Little endian byte order
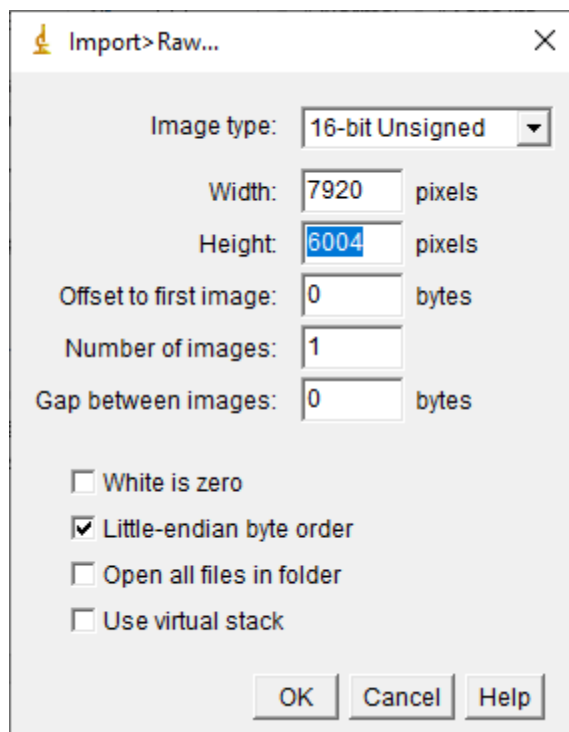
5.5.3   Processed Data Format

Image processing:

    Bias compensation

    Sensor defect correction

    Sensor PRNU correction

    Linearisation (depends on Setup configuration)

    FlatField (depends on Setup configuration)

File > Import > Raw

    Image type :   16 bits signed

    Width :   6001

    Height: 6001

    Little endian byte order
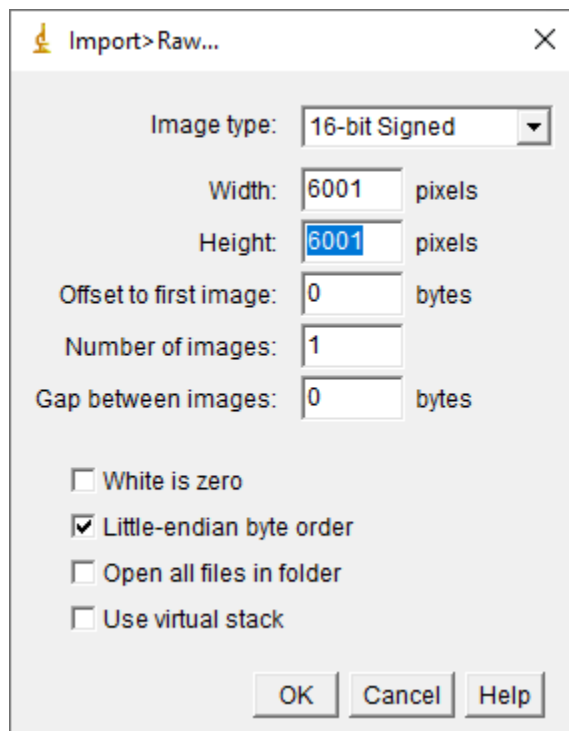
5.6   Display Capture

DisplayRaw and **DisplayProcessed** allow to display the image captured with Measure command

<u>Those commands do not perform a capture</u>

5.7   Next Step

Once a capture has been done, it is possible Measure again or to Setup and Measure

6   Closing the device / Reset

In any state, the device can be closed or reset.
Reset command will perform a power sequence on the Taprisiot (it takes about 2 minutes)

7   Streaming

7.1   Usage

When the device is in Ready state (after a **Setup** command). It is possible to display stream by pressing **DisplayStream**.

DisplayStream	AE	☑
	bProcessed	☑
	exposureTimeUs	90000

Streaming options: AutoExposure: AE parameters are set with SetOption command
                      bProcessed: Capture data are processed.
                                    By default, only defect pixels are removed.
                                    Configuration can be changed in DisplayStreamOption.json file

7.2   Display options:

Zoom   use the mouse wheel.
Cursors   right click on the image and select the option

---
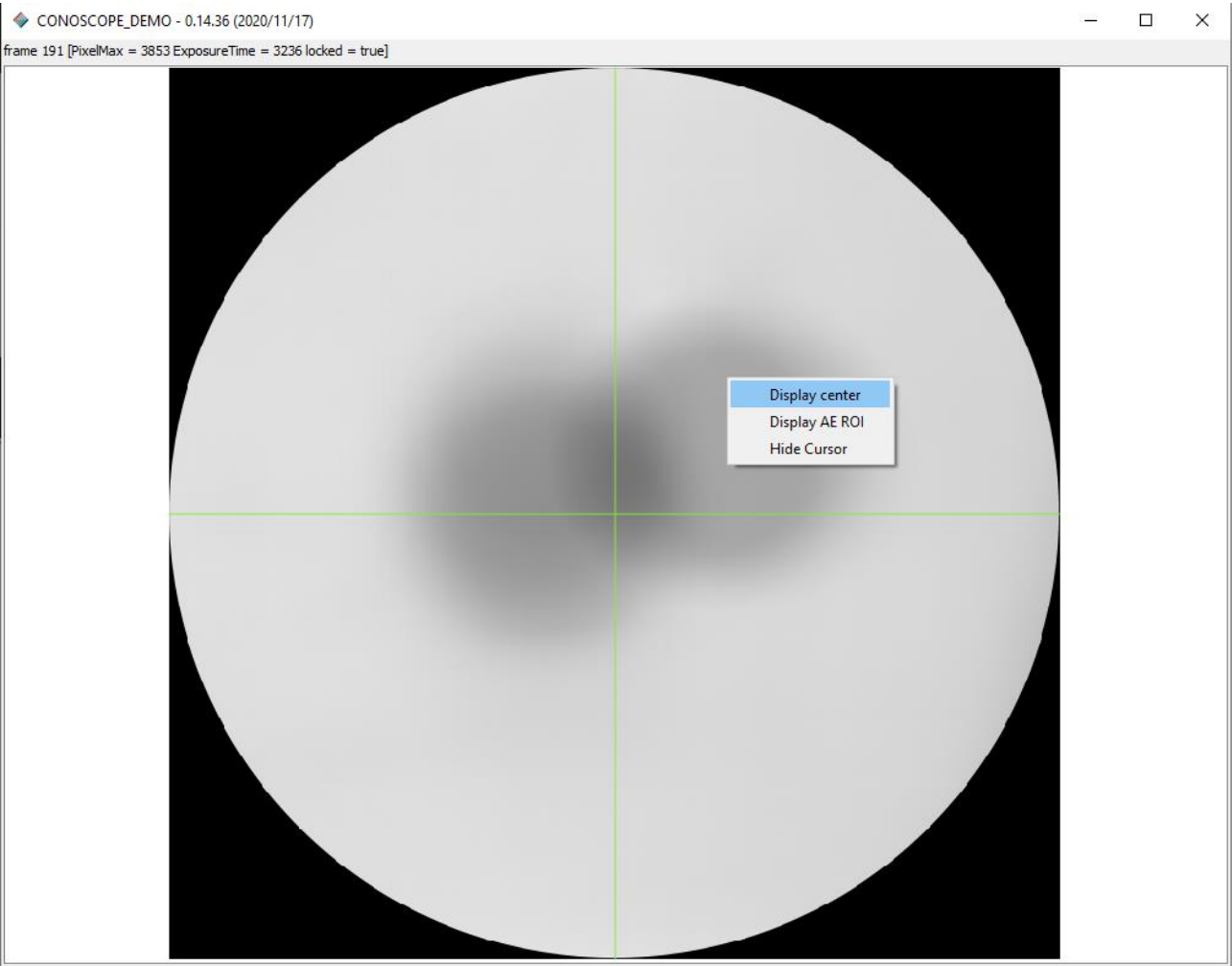
8   Capture Sequence

8.1   Description

Capture sequence captures necessary images to generate X, Y, Z images.
5 capture are used:

      - filter X
      - filter Xz
      - filter Ya
      - filter Yb
      - filter Z

8.2   Command

Command **CaptureSequence** can be called from **Opened** state.

Parameters:

sensorTemperature:	Target temperature for the Taprisiot
waitForTemp:	The sequence waits till the target temperature is reached
nd:	Setup the ND filter value
iris:	Indicate the iris installed on the Taprisiot

exposure time: option 1:   manual exposure, all captures are done
                                        with the same exposure time

CaptureSequence	sensorTemperature	25.00
	waitForTemp	☐
	nd	0
	irisIndex	2
	autoExpo	☐
	useExpoFile	☐
	exposureTimeUs	10000
	nbAcquisition	1

option 2:   exposure time of each capture is defined in
                       CaptureSequenceExposureTime.json

CaptureSequence	sensorTemperature	25.00
	waitForTemp	☐
	nd	0
	irisIndex	2
	autoExpo	☐
	useExpoFile	☑
	nbAcquisition	1

option 3:   AutoExposure

CaptureSequence	sensorTemperature	25.00
	waitForTemp	☐
	nd	0
	irisIndex	2
	autoExpo	☑
	nbAcquisition	1

CaptureSequenceExposureTime.json the exposure time for each filter and is in the application directory

```
{
        "ExposureTimeUs": {
                "Filter_X": 10000,
                "Filter_Xz": 10000,
                "Filter_Ya": 10000,
                "Filter_Yb": 10000,
                "Filter_Z": 10000
        }
}
```

Once the processing is started, it is possible to cancel it with **CaptureSequenceCancel** command

8.3 Result

Once the process is done, X, Y, Z images can be found in capture folder (SetConfig)

20200603_100031_filt_X_nd_0_iris_2_proc_1.bin
20200603_100031_filt_X_nd_0_iris_2_proc_1.json
20200603_100057_filt_Xz_nd_0_iris_2_proc_1.bin
20200603_100057_filt_Xz_nd_0_iris_2_proc_1.json
20200603_100126_filt_Ya_nd_0_iris_2_proc_1.bin
20200603_100126_filt_Ya_nd_0_iris_2_proc_1.json
20200603_100152_filt_Yb_nd_0_iris_2_proc_1.bin
20200603_100152_filt_Yb_nd_0_iris_2_proc_1.json
20200603_100218_filt_Z_nd_0_iris_2_proc_1.bin
20200603_100218_filt_Z_nd_0_iris_2_proc_1.json
20200603_100218_nd_0_iris_2_X_float.bin
20200603_100218_nd_0_iris_2_Y_float.bin
20200603_100218_nd_0_iris_2_Z_float.bin

Note: If bSaveCaptures is checked, all intermediate captures are stored. Then, they can be processed latter.

Images can be opened with ImageJ
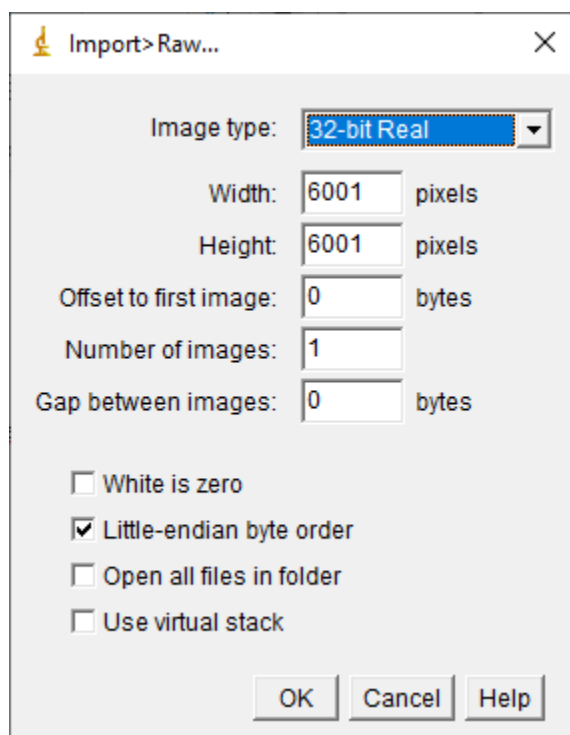
9 Processing captured data

It is possible to process raw data already captured.

From Idle state, set **emulatedCamera** option in **SetConfig** command (press SetConfig button to set)

9.1 Processing RawData

Follow the steps described in chapter "Capturing an image".

Parameter for **Measure** command is the path of the bin capture to process.

(note the associated json file must be present)

Then **ExportProcessed** command will generate the processed image using the calibration data.

9.2 Processing CaptureSequence

Capture paths to be processed are defined in CaptureSequenceCaptures.json (in the application folder)

```
{
        "FilePath": {
                "Filter_X": "capture_X.bin",
                "Filter_Xz": "capture_Xz.bin",
                "Filter_Ya": "capture_Ya.bin",
                "Filter_Yb": "capture_Yb.bin",
                "Filter_Z": "capture_Z.bin"
        }
}
```

Remark: by default, CaptureSequenceCaptures.json does not exists. When **CaptureSequence** is lanched without this file, it will generate a sample file. Then it is possible to update it.

To process the data, follow the steps described in chapter "Capture Sequence"

10 Log file

10.1 Introduction
Log information can be saved in file.

10.2 Configuration
In installation folder:

Name	Date modified	Type	Size
Cfg	04/06/2020 15:06	File folder	
iconengines	04/06/2020 15:04	File folder	
imageformats	04/06/2020 15:04	File folder	
platforms	04/06/2020 15:04	File folder	
styles	04/06/2020 15:04	File folder	
translations	04/06/2020 15:04	File folder	
CaptureSequenceExposureTime.json	06/04/2020 11:51	JSON File	1 KB
config.json	04/06/2020 15:07	JSON File	2 KB
ConoscopeDemo.exe	04/06/2020 15:03	Application	360 KB
ConoscopeLib.dll	04/06/2020 15:03	Application extens...	791 KB
D3Dcompiler_47.dll	11/03/2014 11:54	Application extens...	4 077 KB
demo.json	04/06/2020 15:06	JSON File	1 KB
libEGL.dll	27/03/2020 14:43	Application extens...	24 KB
libGLESv2.dll	27/03/2020 14:43	Application extens...	3 519 KB
LogConoscope.txt	04/06/2020 15:07	Text Document	12 KB
opengl32sw.dll	14/06/2016 14:00	Application extens...	20 433 KB
PipelineLib.dll	04/06/2020 12:42	Application extens...	49 KB
Qt5Core.dll	27/03/2020 14:43	Application extens...	6 217 KB
Qt5Gui.dll	27/03/2020 14:43	Application extens...	6 961 KB
Qt5Svg.dll	27/03/2020 19:47	Application extens...	329 KB
Qt5Widgets.dll	27/03/2020 14:43	Application extens...	5 425 KB
Qt5Xml.dll	27/03/2020 14:43	Application extens...	200 KB

Edit demo.json file and add LogMasks:

```
{
    "Application": {
        "LogMasks": [
            "State",
            "StateMachine",
            "Worker"
        ],
```

10.3 Log file
Log files are:

LogConoscope.txt
LogPipeline.txt