EXtra-foam Documentation

Release 1.1.0dev

Jun Zhu

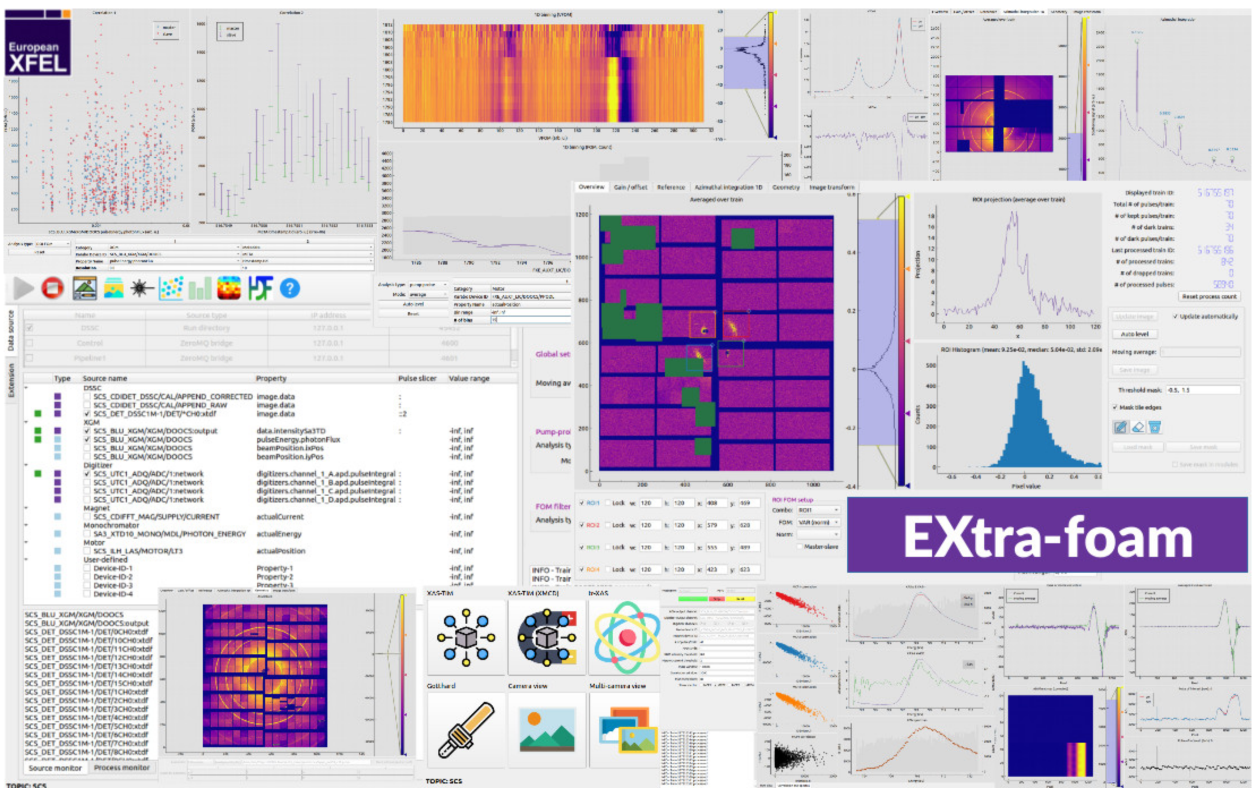
Jun 05, 2022

Contents

Contents:

CHAPTER 1

Introduction



EXtra-foam (**F**ast **O**nline **A**nalysis **M**onitor) is a framework that provides real-time and off-line data analysis (**detector geometry**, **pump-probe**, **azimuthal integration**, **ROI**, **statistics**, etc.) and visualization for experiments that use **2D area detectors** (*AGIPD*, *LPD*, *DSSC*, *FastCCD*, *JungFrau*, *ePix100*, etc.) and **1D detectors** (*Gotthard*, *XGM*, *digitizer*, etc.) at European XFEL.

	online		offline	
	raw	calibrated	raw	calibrated
AGIPD	Yes	Yes	Yes	Yes
LPD	Yes	Yes	Yes	Yes
DSSC	Yes	Yes	Yes	Yes
JungFrau	No	Yes	Yes	Yes
FastCCD	Yes	Yes	Yes	Yes
ePix100	Yes	Yes	Yes	Yes
Basler camera	Yes		Yes	
Gotthard	Yes		Yes	
Digitizer	Yes		Yes	

1.1 Why use EXtra-foam

1. It allows users to perform EDA (exploratory data analysis) in real time by 'probing' the data with a combination of different analysis tools, for instance, monitoring individual pulses in a train, checking correlation and trying different normalization methods, etc. This is particularly useful if users are not sure what the data really look like or want to have a sanity check;

2. It provides tailored data analysis configuration and visualization for specific experiments. For example, in *pump-probe* setup, it allows users to choose how the pump and probe pulses are distributed (e.g. in the same train or different train) by providing several typical "modes". It also integrates important plots in a single window so that users can gather abundant information in a glance;

3. It allows uses to **replay the experiment** with files. This is another very useful feature since for both newcomers and veterans. For newcomers, it helps to understand what whey will see during a real experiment by running the 'replay' with some sample/real data; for veterans, it helps to optimize the parameters which in turn provides a better real-time monitoring and feedback during experiments. Moreover, *When starting a new type of experiment, you may not be able to observe the expected signal during the first run. It could be worthy of double checking the analysis setup by replaying the experiment before searching for other reasons.* It is worth noting that the 'replay' result could be different from the real-time result if you are using the real-time calibration service from Karabo, since the offline calibration algorithms are more complicated than the real-time ones.

1.2 Performance

European XFEL can provide X-ray free-electron laser pulse trains (macropulse in accelerator terminology) at a maximum repetition rate of 10 Hz. These pulse trains can be filled with up to 2700 pulses (micropulse in accelerator terminology), corresponding to a maximum intra-train repetition rate of 4.5 MHz. Detectors at European XFEL can be categorized into pulse-resolved and train-resolved ones. Speaking of the performance of real-time analysis, we use the combination of repetition rate (trains/s) and frame rate (pulses/train). For instance, 10 Hz with 64 pulses/train on a DSSC detector means that 640 frames of 1M megapixel images can be preprocessed and analysed per second.

Table 1: Performance on the online cluster [72 Intel(R) Xeon(R) Gold 6140 CPU @ 2.30GHz, 754 GB RAM]

	pulse-resolved/multi-frame detectors	train-resolved/single-frame detectors
processing rate	> 10 Hz with 64 pulses/train	> 10 Hz

Note: Due to the limited performance of *PyQt*, the visualization rate could be slower than the processing rate if there are too many plots to render, especially for train-resolved detectors.

CHAPTER 2

Installation

If you want to use **EXtra-foam** on the online or *Maxwell* cluster, please check **GETTING STARTED**.

To install **EXtra-foam** in your own environment, you are encouraged to use Anaconda to run and build **EXtra-foam**.

2.1 Install EXtra-foam

```
$ git clone --recursive --branch <tag_name> https://github.com/European-XFEL/EXtra-
↪foam.git

# If you have cloned the repository without one or more of its submodules, run
$ git submodule update --init

$ cd EXtra-foam

# Create an Anaconda_ environment, by default it's named 'extra_foam'. If
# you want to install into an existing environment, use `conda env update` instead.
$ conda env create -f environment.yml

$ conda activate extra_foam
# We need to set this variable so that the libraries from the conda
# environment are loaded first, in particular libstdc++. If the system
# libstdc++ is loaded first and it's too old for the version extra-foam was
# compiled against, then we might get nasty loader errors.
$ conda env config vars set LD_LIBRARY_PATH=${CONDA_PREFIX}/lib:${LD_LIBRARY_PATH}
# Re-activate the environment so the new variables take effect
$ conda deactivate
$ conda activate extra_foam

# Install extra-foam
$ export CMAKE_PREFIX_PATH=${CONDA_PREFIX:-"$(dirname $(which conda))/../"}
$ pip install .
```

2.2 Install your own EXtra-foam kernel on the Maxwell cluster for offline analysis

For now, there is no documentation for the Python bindings of the C++ implementations in **EXtra-foam**. If you are interested in using those super fast C++ implementation to accelerate your analysis, please feel free to dig into the code and ask questions.

```
# ssh to the Maxwell cluster and then
$ module load anaconda3

# follow the previous steps to install EXtra-foam

$ conda install ipykernel nb_conda_kernels -y

# Now you should be able to load the newly created kernel on max-jhub.
```

2.3 Install C++ API only

Please check foamalgo.

CHAPTER 3

Getting started

3.1 Choose the correct version

EXtra-foam can be started on both online and *Maxwell* clusters. Currently, there are two versions of **EXtra-foam** deployed. Please always consult your contact person if you are not sure which version to use.

3.1.1 I. Latest version

This is the latest release of **EXtra-foam**. This version usually contains more features than the **stable** version.

```
module load exfel EXtra-foam/beta
extra-foam DETECTOR TOPIC
```

More info on command line arguments can be obtained as

```
[user@exflonc12 ~]$ extra-foam --help

 usage: extra-foam [-h] [-V] [--debug] [--redis_address REDIS_ADDRESS]
                   {AGIPD,LPD,DSSC,JUNGFRAUPR,JUNGFRAU,FASTCCD,EPIX100,BASLERCAMERA}
                   {SPB,FXE,SCS,SQS,MID,HED}

 positional arguments:
   {AGIPD,LPD,DSSC,JUNGFRAUPR,JUNGFRAU,FASTCCD,EPIX100,BASLERCAMERA}
                         detector name (case insensitive)
   {SPB,FXE,SCS,SQS,MID,HED}
                         Name of the instrument

 optional arguments:
   -h, --help            show this help message and exit
   -V, --version         show program's version number and exit
   --n_modules N_MODULES
                         Number of detector modules. It is only available for
                         using single-module detectors like JungFrau in a
```

(continues on next page)

(continued from previous page)

```
                        combined way. Not all single-module detectors are
                        supported.
  --debug               Run in debug mode
  --pipeline_slow_policy {0,1}
                        Pipeline policy when the processing rate is slower
                        than the arrival rate (0 for always process the latest
                        data and 1 for wait until processing of the current
                        data finishes)
  --redis_address REDIS_ADDRESS
                        Address of the Redis server
```

For more details about detector modules, please refer to *Geometry*.

Note: It sometime takes more than a minute to start **EXtra-foam** for the first time! This is actually an issue related to the infrastructure and not because **EXtra-foam** is slow.

Note: If you are connecting to the online or *Maxwell* clusters via SSH, you will need to enable X11 forwarding by including the -X option.

Note: In order to have a better experience with **EXtra-foam** on the *Maxwell* cluster, you should need FastX2 at max-display. There is also a link for downloading the desktop client on the bottom-right corner when you opened max-display and logged in. For more details, please refer to the official website for FastX2 at DESY.

3.1.2 II. Stable version

To start the **stable** version on online or *Maxwell* clusters:

```
module load exfel EXtra-foam
extra-foam DETECTOR TOPIC
```

3.1.3 III. Test version

To start the **test** version on online or *Maxwell* clusters:

```
module load exfel EXtra-foam/alpha
extra-foam DETECTOR TOPIC
```

Note: **test** version is not covered by OCD!

3.2 Data analysis in real time

For real-time data analysis, the (calibrated) data is streamed via a *Karabo bridge*, which is a *Karabo* device (*TrainMatcher*) running inside the control network.

Name	Source type	IP address	Port
DSSC	ZeroMQ bridge	10.253.0.140	4511
Control	ZeroMQ bridge	127.0.0.1	4600
Pipeline1	ZeroMQ bridge	127.0.0.1	4601
Pipeline2	ZeroMQ bridge	127.0.0.1	4602

Note: Please check the online-clusters available for users at different instruments.

Note: If you are an external user coming to XFEL, EXtra-foam should already have been all setup with all the relevant hostnames/ports. If you are an instrument scientist who wants to know where the Karabo bridges are, please get in touch your local DA contact (or email da-support@xfel.eu).

3.3 Data analysis with files

See *Stream data from files*

CHAPTER 4

Troubleshooting

4.1 Starting-up issues

4.1.1 Could not connect to display

While trying to run **EXtra-foam** remotely on the online cluster (exflonc12, etc), if you end up with error messages similar to,

```
qt.qpa.xcb: could not connect to display
qt.qpa.plugin: Could not load the Qt platform plugin "xcb" in "" even though it was
↪found.
This application failed to start because no Qt platform plugin could be initialized.
Reinstalling the application may fix this problem.
```

please make sure that you have done X11 forwarding while logging to the online cluster. Using **EXtra-foam** on Maxwell cluster, it is better to use FastX2 at max-display as explained in previous section.

- **Shut down the redis server?**

If you are prompted to warnings like,

```
[user@exflonc12 ~]$ extra-foam DSSC SCS

services.py - WARNING - Found Redis server for DSSC (started at 2020-02-06 12:50:03.
↪906872)
already running on this machine using port 6380!

You can choose to shut down the Redis server. Please note that the owner of the Redis
↪server
will be informed (your username and IP address).

Shut down the existing Redis server? (y/n)
```

EXtra-foam uses *Redis* as broker to pass meta information between different processes. By design, each type of detector has its unique *Redis* port so one can safely run more than one **EXtra-foam** instances for different detectors on the same machine. However, it is not allowed to run two instances with the same type of detector. Also, **EXtra-foam** receives data from **karabo bridge** and thus there can be data loss if there is any instance secretly running in the background, stealing the data.

In the instrument control room, there should be only one **EXtra-foam** instance for the detector that is running. Therefore, it is safe to type "y" to shut down the existing *Redis* server. However, if somebody wants to make a joke about you and did that remotely, you will get informed.

4.1.2 Config file is invalid

If you are prompted to warning like,

```
Traceback (most recent call last):
  File "/home/username/anaconda3/envs/foam/bin/extra-foam", line 11, in <module>
    load_entry_point('EXtra-foam', 'console_scripts', 'extra-foam')()
  File "/home/username/xfel-data-analyais/EXtra-foam/extra_foam/services.py", line␣
↪356, in application
    config.load(detector, topic)
  File "/home/username/xfel-data-analyais/EXtra-foam/extra_foam/config.py", line 456,␣
↪in load
    self._data.load(detector, topic)
  File "/home/username/xfel-data-analyais/EXtra-foam/extra_foam/config.py", line 382,␣
↪in load
    self.from_file(det, topic)
  File "/home/username/xfel-data-analyais/EXtra-foam/extra_foam/config.py", line 393,␣
↪in from_file
    raise OSError(msg)
OSError: Invalid config file: /home/username/.EXtra-foam/scs.config.yaml
ParserError('while parsing a block mapping', <yaml.error.Mark object at␣
↪0x7fcffbd84910>,
"expected <block end>, but found '<block mapping start>'", <yaml.error.Mark object at␣
↪0x7fcffbd84ed0>)
```

This error is triggered when the *Config file* is not valid. Please correct it if you have modified the default one. Alternatively, you can delete it and let the program generate a default one for you.

4.1.3 Incorrect dependencies

If you happen to have some Python packages which are used in **EXtra-foam** but in the meanwhile installed in your *~/.local* directory, anything bad can happen if they have different version numbers. In the best case, **EXtra-foam** crashes, for example, when it attempted to call a function which does not exist in your local installation, and you will immediately notice it. In the worst case, the software keeps running but the result is incorrect and the difference between the incorrect result and the correct one is unperceivable.

This is the downside of using Anaconda to deploy software. However, it is easy to track down the loathful Python package if you already have one or a few suspects. Assuming the module *EXtra-foam/beta* is loaded, you can check the suspect one-by-one by following the example below:

```
~ python
Python 3.7.3 (default, Mar 27 2019, 22:11:17)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import redis
>>> print(redis.__file__)
# expected result :-)
```

(continues on next page)

(continued from previous page)

```
/gpfs/exfel/sw/software/xfel_anaconda3/EXtra-foam-beta/lib/python3.7/site-packages/
↪redis/__init__.py
# This is bad!
/home/username/.local/lib/python3.7/site-packages/redis/__init__.py
```

The remedy is simply. Run *pip uninstall* to remove your local installation.

4.2 Operational issues

4.2.1 No data is received

If **EXtra-foam** opens up fine and running it by clicking on the *Start* button in the *Main GUI* does nothing, please make sure that relevant *TrainMatcher* device is properly configured, activated, and sending data.

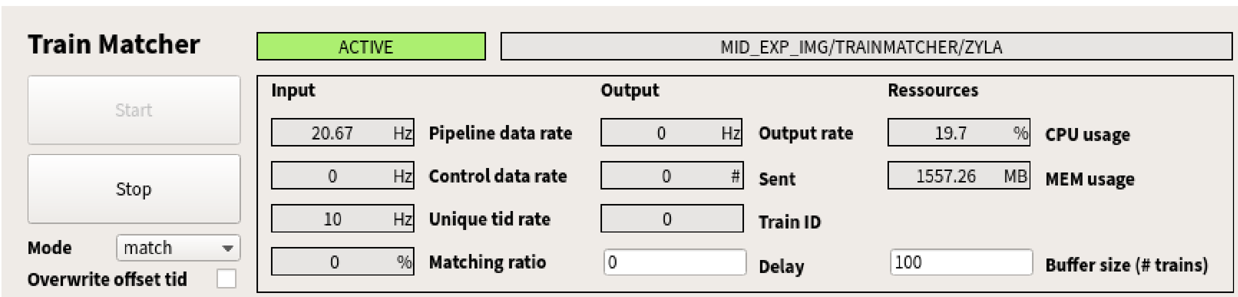


Fig. 1: Example of the matching/streaming status panel of a TrainMatcher.

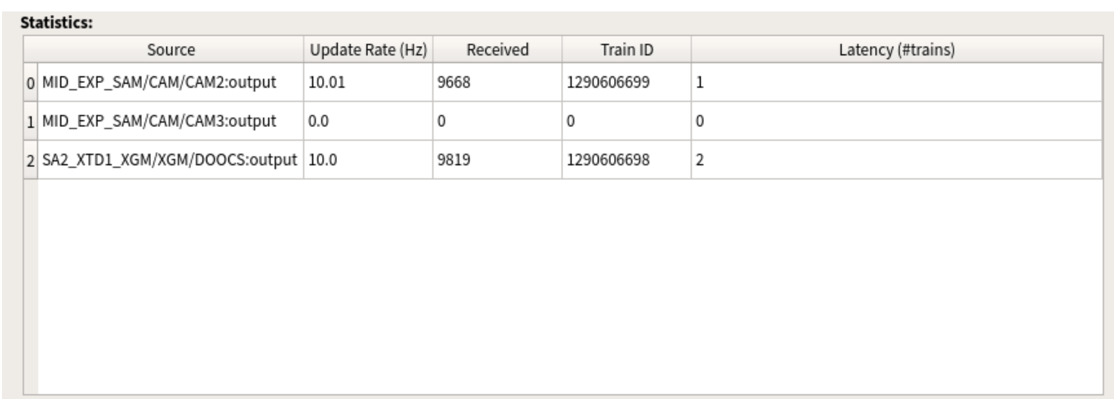
There is a wonderous number of things that could go wrong:

- Matching might be failing. This can be checked by looking at the matching ratio, in the above screenshot the matching ratio is 0%.

  This could be for a couple of reasons:

  – A source is not found. If the TrainMatcher cannot get data from a device for some reason, it will display a status message in the sources list.

  – A source is enabled that isn't sending data. Check this by looking at the statistics table in the TrainMatcher:

  Statistics:

	Source	Update Rate (Hz)	Received	Train ID	Latency (#trains)
0	MID_EXP_SAM/CAM/CAM2:output	10.01	9668	1290606699	1
1	MID_EXP_SAM/CAM/CAM3:output	0.0	0	0	0
2	SA2_XTD1_XGM/XGM/DOOCS:output	10.0	9819	1290606698	2

  In this case one of the cameras (`MID_EXP_SAM/CAM/CAM3:output`) isn't sending any data (its update rate is 0Hz), so disabling that camera would fix the matching. Of course, if the device is actually meant to be streaming data that would point to an issue with the device (e.g. maybe the camera isn't acquiring).

	Select	Source	Offset (#trains)	Status
0	☐ False	MID_EXP_DES/CAM/CAM1:output	0	
1	☐ False	MID_EXP_DES/CAM/CAM2:output	0	
2	☐ False	MID_EXP_IMG/CAM/ZYLA:output	0	
3	☐ False	MID_EXP_SAM/CAM/CAM1:output	0	
4	☑ True	MID_EXP_SAM/CAM/CAM2.schrodingersProperty	0	Property not found on device
5	☑ True	MID_EXP_SAM/CAM/CAM2:output	0	Monitoring
6	☐ False	MID_EXP_SAM/CAM/CAM3:output	0	
7	☑ True	MID_EXP_SAM/CAM/FAM2:output	0	Device not found
8	☑ True	SA2_XTD1_XGM/XGM/DOOCS:output	0	Monitoring

Fig. 2: Examples of errors the TrainMatcher shows for sources it can't connect to.

- The buffer size is too low. Data from different sources arrives at slightly different times, and unmatched trains are kept in a buffer which are continually checked and matched as new data comes in. If a source has unusually high latency and the buffer size is too small, old trains may be removed from the buffer before the data from the high-latency source has a chance to arrive.

  You can check this by looking at the latency column in the statistics table, and comparing it to the buffer size in the *matching/streaming status panel*. In the screenshot above of the statistics table, there's only a couple trains of latency so a buffer size of 100 should be plenty.

  Note: If the latency of sources is ridiculously high (e.g. hundreds of millions of trains), the TrainMatcher is probably running on a device server that hasn't been configured with a timeserver, so it's reporting the wrong latency. The device server will need to be reconfigured with a timeserver and restarted. This needs to be done by an expert, so ask your local DA contact or email da-support@xfel.eu.

- Sending might be failing. Check this with the `Output rate` and `Sent` fields in the *matching/streaming status panel*. If the output rate is 0Hz (and the matching ratio is is not 0Hz), then there's a couple of possibilities:

  - If there are no interfaces listed in the bridge outputs list in TrainMatcher, then check the `ZeroMQ configuration` property of the TrainMatcher.

    * If the property is empty the TrainMatcher doesn't yet have a bridge configured, and you should ask your local DA contact for help configuring the TrainMatcher (or email da-support@xfel.eu).

    * If it's not empty, then some other process might be using that port and blocking the TrainMatcher from binding to it. Check this by SSH'ing into the machine running the TrainMatcher (while the TrainMatcher device is shutdown) and searching for used ports:

```
# Search for the port that the TrainMatcher is configured to use
$ netstat -antlp | grep 45059
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
tcp        0      0 10.253.0.171:45059      0.0.0.0:*              ␣
↪LISTEN      24296/python3
tcp        0      0 10.253.0.171:45059      10.253.0.151:51012     ␣
↪ESTABLISHED 24296/python3
```

In this example, there's a `python3` process with PID `24296` bound to port `45059`. If you see a similar process, *do not* attempt to kill it yourself, but contact an expert.

However, in the past we've also observed a strange bug that occurs if a TrainMatcher crashes in the middle of operation while EXtra-foam is connected to it: EXtra-foam will keep the connection to the port, and even though it's not bound to it, that alone will block the TrainMatcher from binding to the port. In this case you'll see something like:

```
$ netstat -antlp | grep 45056
(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)
tcp        0      0 10.253.0.171:45056      10.253.0.171:45056     ␣
↪ESTABLISHED 133974/python
```

There's only a single `ESTABLISHED` connection to that port, which in this case is from EXtra-foam. Restarting EXtra-foam and then the TrainMatcher should fix this.

  – Otherwise, EXtra-foam is probably configured to read from the wrong hostname/port. Check that the *Bridge client connections* match that of the TrainMatcher.

- EXtra-foam cannot find the sources it's configured for in the data being streamed from the TrainMatcher. In this case, the TrainMatcher will show that data is being sent but EXtra-foam will do nothing.

  This could be because:

  – There's a typo in some source name in EXtra-foam, in which case you'll need to modify the *Data source tree*.

  – EXtra-foam has the right source name, but it hasn't been enabled in the TrainMatcher. You'll need to cross-check the enabled sources in EXtra-foam and the TrainMatcher.

CHAPTER 5

Config file

Users from different instruments should use the corresponding config file to config the instrument specific data sources, the detector specific setups and so on.

Each instrument has a default config file, which can be found in the github repository. We appreciate if the beamline scientists can help us keep the default config file updated.

Let's take *FXE* for example, when one starts a detector on topic *FXE* for the first by time:

```
extra-foam LPD FXE
```

, the system will create a new config file *$HOME/.EXtra-foam/fxe.config.yaml* using the default one. The first block of the config file looks like the following:

```
SOURCE:
    # Default source type: FILES or BRIDGE
    DEFAULT_TYPE: 1
    CATEGORY:
        LPD:
            PIPELINE:
                FXE_DET_LPD1M-1/CAL/APPEND_CORRECTED:
                    - image.data
                FXE_DET_LPD1M-1/DET/*CH0:xtdf:
                    - image.data

        JungFrau:
            PIPELINE:
                FXE_XAD_JF1M/DET/RECEIVER-1:display:
                    - data.adc
                FXE_XAD_JF1M/DET/RECEIVER-2:display:
                    - data.adc

        XGM:
            CONTROL:
                SA1_XTD2_XGM/DOOCS/MAIN:
                    - pulseEnergy.photonFlux
```

(continues on next page)

(continued from previous page)

```
                    - beamPosition.ixPos
                    - beamPosition.iyPos
            PIPELINE:
                SA1_XTD2_XGM/DOOCS/MAIN:output:
                    - data.intensitySa1TD

        Motor:
            CONTROL:
                FXE_SMS_USR/MOTOR/UM01:
                    - actualPosition
                FXE_SMS_USR/MOTOR/UM02:
                    - actualPosition
```

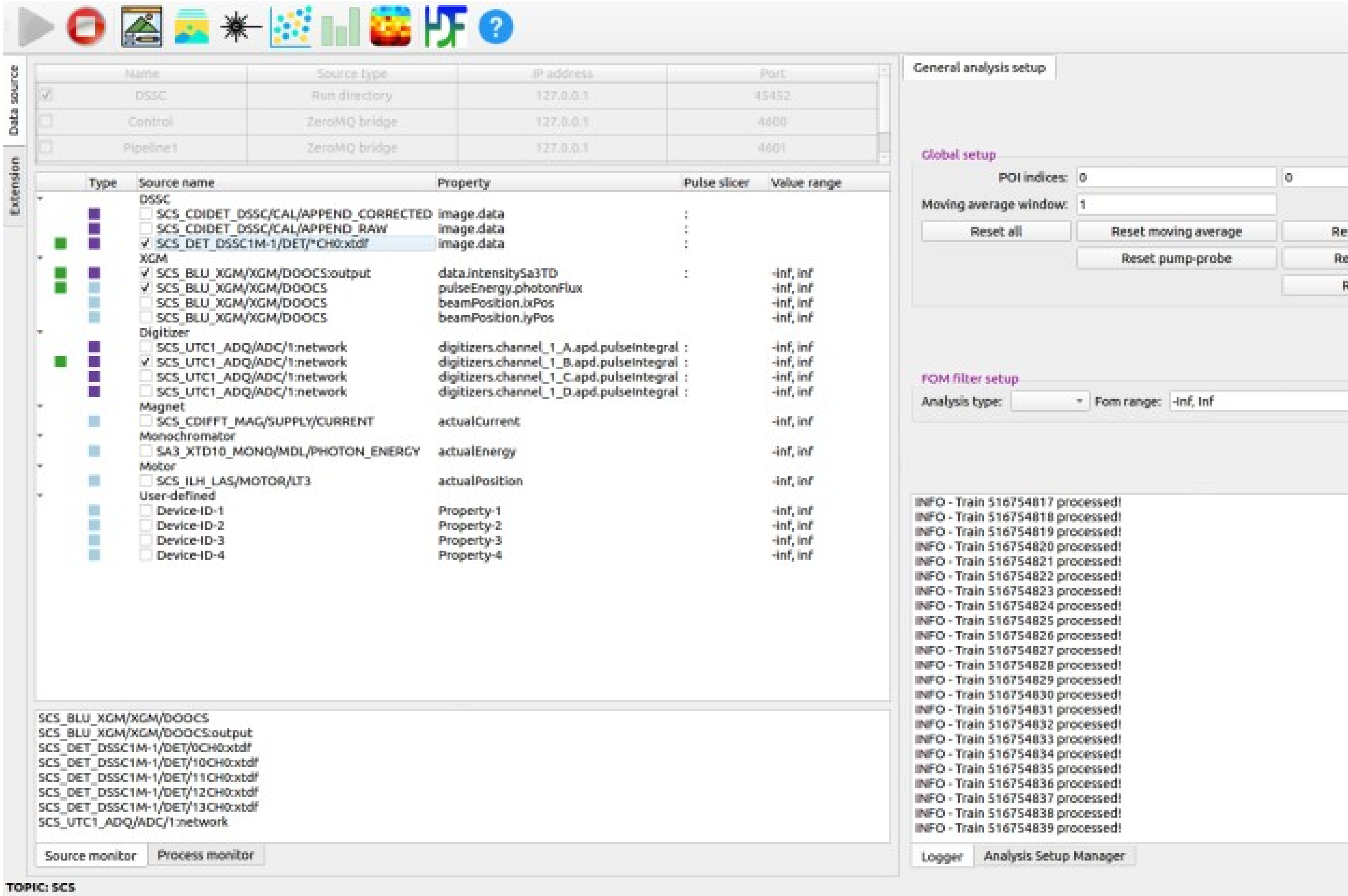
The next block of the config file looks like the following:

```
DETECTOR:
    LPD:
        GEOMETRY_FILE: lpd_mar_18_axesfixed.h5
        # - For lpd_mar_18.h5 and LPDGeometry in karabo_data
        # "QUAD_POSITIONS": [[-13.0, -299.0], [11.0, -8.0], [-254.0, 16.0], [-278.0, -
↪275.0]],
        # - For lpd_mar18_axesfixed.h5 and LPD_1MGeometry in karabo_data
        # The geometry uses XFEL standard coordinate directions.
        QUAD_POSITIONS:
            x1:   11.4
            y1:  299
            x2:  -11.5
            y2:    8
            x3:  254.5
            y3:  -16
            x4:  278.5
            y4:  275
        BRIDGE_ADDR: 10.253.0.53
        BRIDGE_PORT: 4501
        LOCAL_ADDR: 127.0.0.1
        LOCAL_PORT: 45451
        SAMPLE_DISTANCE": 0.4
        PHOTON_ENERGY": 9.3

    JungFrau:
        BRIDGE_ADDR: 10.253.0.53
        BRIDGE_PORT: 4501
        LOCAL_ADDR: 127.0.0.1
        LOCAL_PORT: 45453
        SAMPLE_DISTANCE: 2.0
        PHOTON_ENERGY: 9.3
```

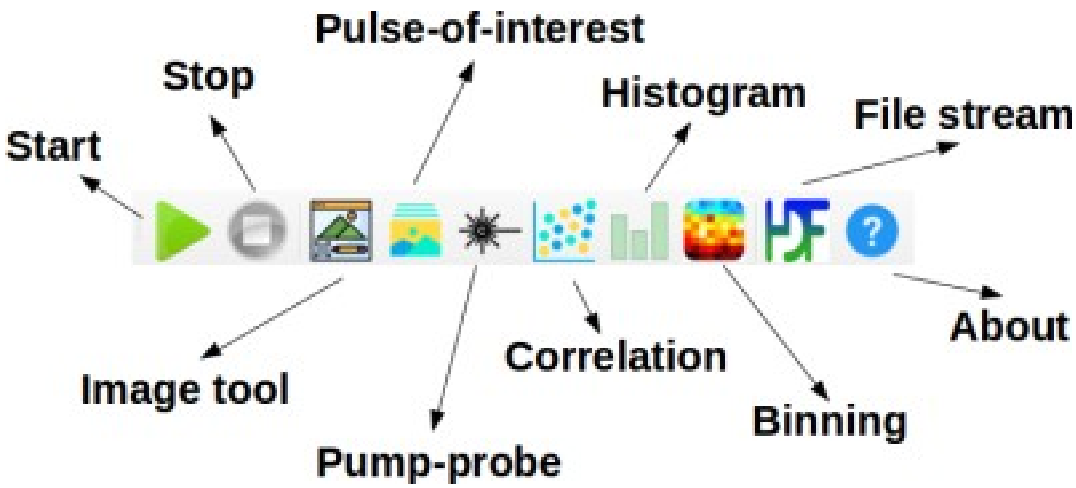
CHAPTER 6

Main GUI

The main GUI of **EXtra-foam** is divided into several control panels grouped by functionality and a log window.



6.1 Action bar



- **Start**

  Start acquiring, processing and visualizing data.

- **Stop**

  Stop acquiring and processing data.

- **Image tool**

  Open the *Image tool* for image related manipulation and analysis.

- **Pulse-of-interest**

  Open the window for monitoring up to two pulses of interest. *Only available for pulse-resolved detectors*

- **Pump-probe**

  Open the window for *Pump-probe analysis*.

- **Correlation**

  Open the window for *Correlation* analysis.

- **Histogram**

  Open the window for *Histogram* analysis.

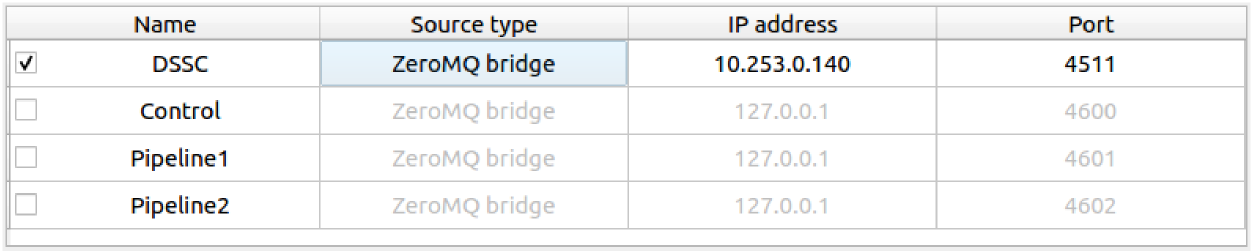
- **Binning**

  Open the window for *Binning* analysis.

- **File stream**

  Open the window for *Stream data from files*.

- **About**



	Name	Source type	IP address	Port
✓	DSSC	ZeroMQ bridge	10.253.0.140	4511
	Control	ZeroMQ bridge	127.0.0.1	4600
	Pipeline1	ZeroMQ bridge	127.0.0.1	4601
	Pipeline2	ZeroMQ bridge	127.0.0.1	4602

Fig. 1: Bridge settings in EXtra-foam.

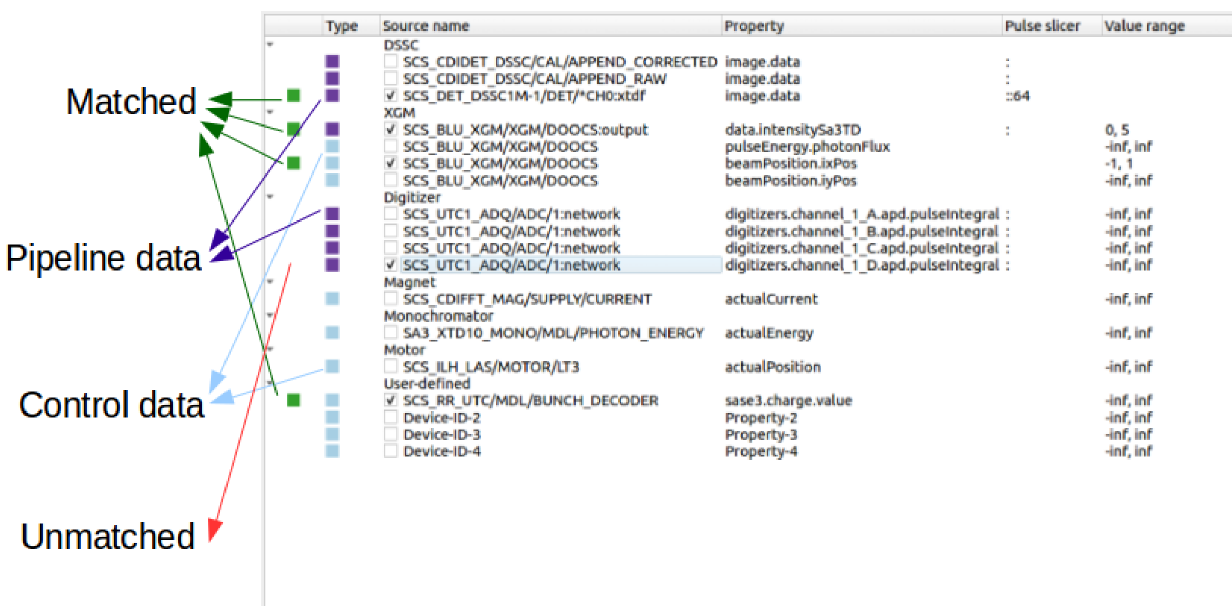
6.2 Data source

6.2.1 Bridge client connections

EXtra-foam supports different kinds of data flows. Unless you are very familiar with the **EXtra-foam** and Karabo, it is recommended to use the scenario 1 depicted in *Streaming from Karabo*. Namely, only one connection should be checked.

Input	Description
`Source type`	Receiving the data from • *ZeroMQ bridge*: mainly used for real-time analysis. The data will be sent from a *TrainMatcher* Karabo device; • *run directory*: used for replaying the experiment.
`IP address`	IP address of the TCP connection.
`Port`	Port number of the TCP connection.

6.2.2 Data source tree

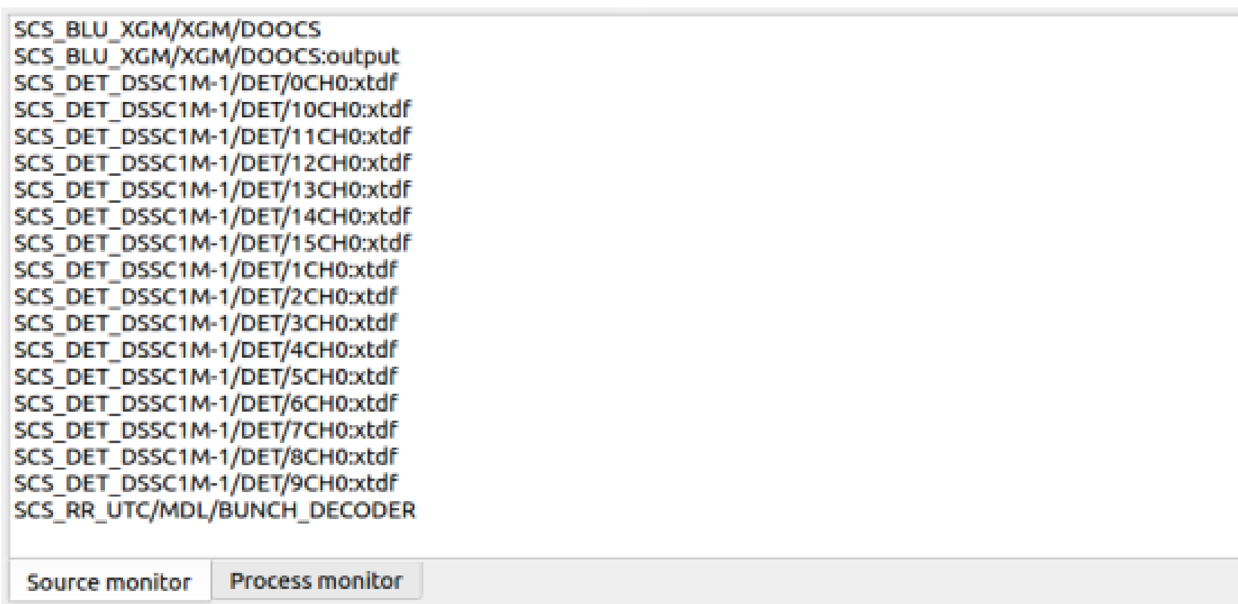


In the data source tree, one can select which data source items (the ID of a data source item is defined by both the *Source name* and the *Property*) are required in the analysis. Data source items can be configured via *Config file*. Users are not allowed to add/delete data source items in the GUI, i.e. add or delete rows in the tree. Alternatively, users can modify an existing data source item *when it is checked*.

When a data source item is checked and is matched (found together with other data source items with the same train ID), the indicator on the first column of the tree will turn green. The indicator will turn red if the data source item is not able to be matched for a while or it is unchecked.

Input	Description
`Type`	Pipeline data or control data.
`Source name`	Karabo device or output channel name
`Property`	Property of the given device or output channel
`Pulse slicer`	The input will be used to construct a *slice* object in Python which is used to select the specified pulse pattern in a train from a pipeline data.
`Value range`	Value range filter of the corresponding source. When it applies to a pipeline data, it will apply data filtering pulse-by-pulse. When it applies to a control data, it will then apply data filtering train-by-train. *Not all pipeline data support it*

6.2.3 Source monitor



Source monitor is used to display sources names (Karabo device and output channel) received by the Karabo bridge client. Different from the (match/unmatched) indicator in the data source tree, it also shows sources which do not exist in the data source tree. When a modular source (e.g. DSSC data) is selected and matched, it displays individual names of the received modules. However, the source monitor does not show property names for speed, as a Karabo device can have dozens of control data. The property names of a Karabo device or an output channel can be found in the Karabo GUI when you are performing online analysis in the hutch, or in the *File streamer* when you are streaming data from files.

Note: The Karabo bridge client will only start to receive data when the green *Start bridge* button is checked and the data source item of the main detector is selected.

6.3 General analysis

6.3.1 Global setup

Define analysis parameters used globally.

Input	Description
`POI indices`	Indices of the pulse of interest (POI) 1 and 2. It is used for visualizing a single image in the *Pulse-of-interest* window. **If 'Pulse slicer' is used to slice a portion of the pulses in the train, this index is indeed the index of the pulse in the sliced train**. *Pulse-resolved detector only.*
`Moving average window`	Moving average window size. If the moving average window size is larger than 1, moving average will be applied to all the registered analysis types. If a new window size is smaller than the old one, the moving average calculation will start from the scratch. Currently, this setup will affect the calculations of: • Pulse intensity of XGM • Pulse integral of Digitizer • Train-resolved (pump/probe) azimuthal integration • Train-resolved (pump/probe) ROI FOM • Train-resolved (pump/probe) ROI 1D projection
`Reset`	Reset the moving average counts of all registered analysis types.

Warning: Since version 1.10, another moving average was re-activated in *Image tool*. If both the moving averages are set, the result of azimuthal integration, ROI FOM and ROI 1D projection will be a moving average on top of the moving averaged image! This is due to some historical reason and it will be fixed in the future.

6.3.2 Pump-probe setup

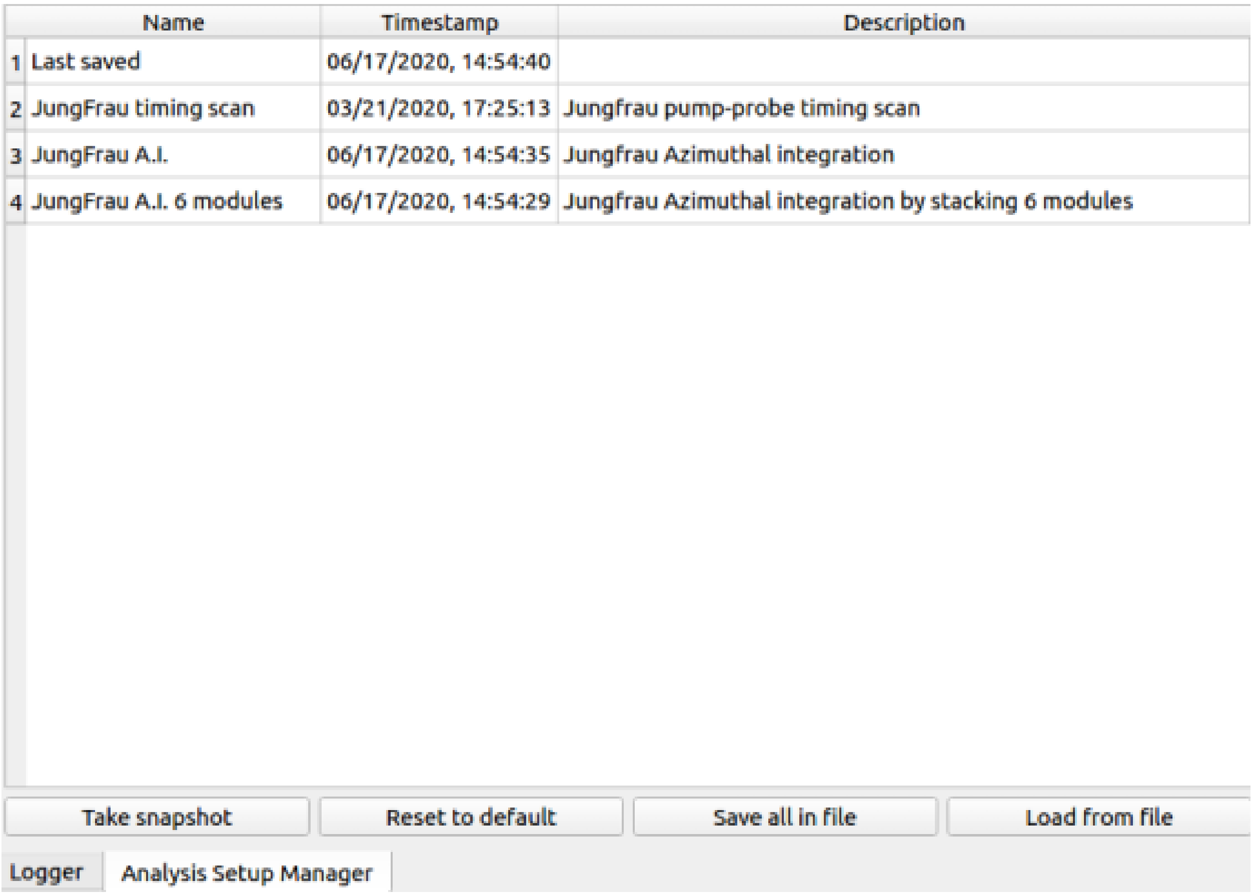
See *Pump-probe analysis*

6.3.3 Pulse filter setup

Apply data reduction by setting the lower and upper boundary of the specified FOM. Currently, it affects calculating the average of images in a train as well as the averages of images of ON-/Off- pulses in a train

Input	Description
`Analysis type`	See Analysis type.
`FOM range`	Number of bins of the histogram.
`pulse resolved`	This checkbox is only enabled for the pulse-resolved detectors. When it is checked, the filtering is pulse-wise. Otherwise, the filtering is train-wise.

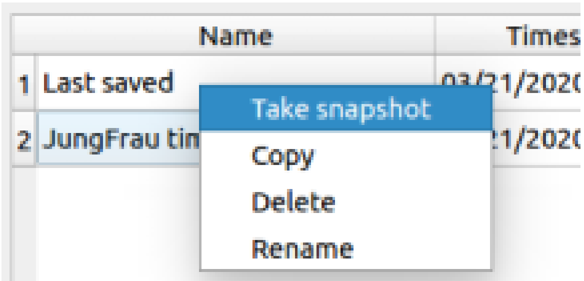
6.4 Analysis setup manager



Analysis setup manager is a new feature introduced in version 0.8.1, it allows users to save and load different analysis setups (a snapshot in the Redis database) conveniently. To apply a setup, simply **double-click** the name of the setup listed in the table. Please distinguish it from *Config file*, which is mainly used for data source management. Due to the historical reason, some setups in the *Config file* can also be saved and loaded via the *Analysis setup manager*, like `photon energy`, `sample distance`, etc. *Config file* defines the default setups which will be overwritten when a setup in the *Analysis setup manager* is applied. The default setups can be recovered by clicking the `Reset to default` button.

Input	Description
`Take snapshot`	Take a snapshot of the current parameters and store them in `Last saved`.
`Reset to default`	Reset the current parameters to default. `Last saved` will not be affected.
`Save all to file`	Save all the setups listed in the table to file. The data in the setup file will be overwritten.
`Load from file`	Load setups from file. In case of name conflict, the listed setups in the table will be overwritten.

When right-clicking the name of a snapshot, a context menu will show up:

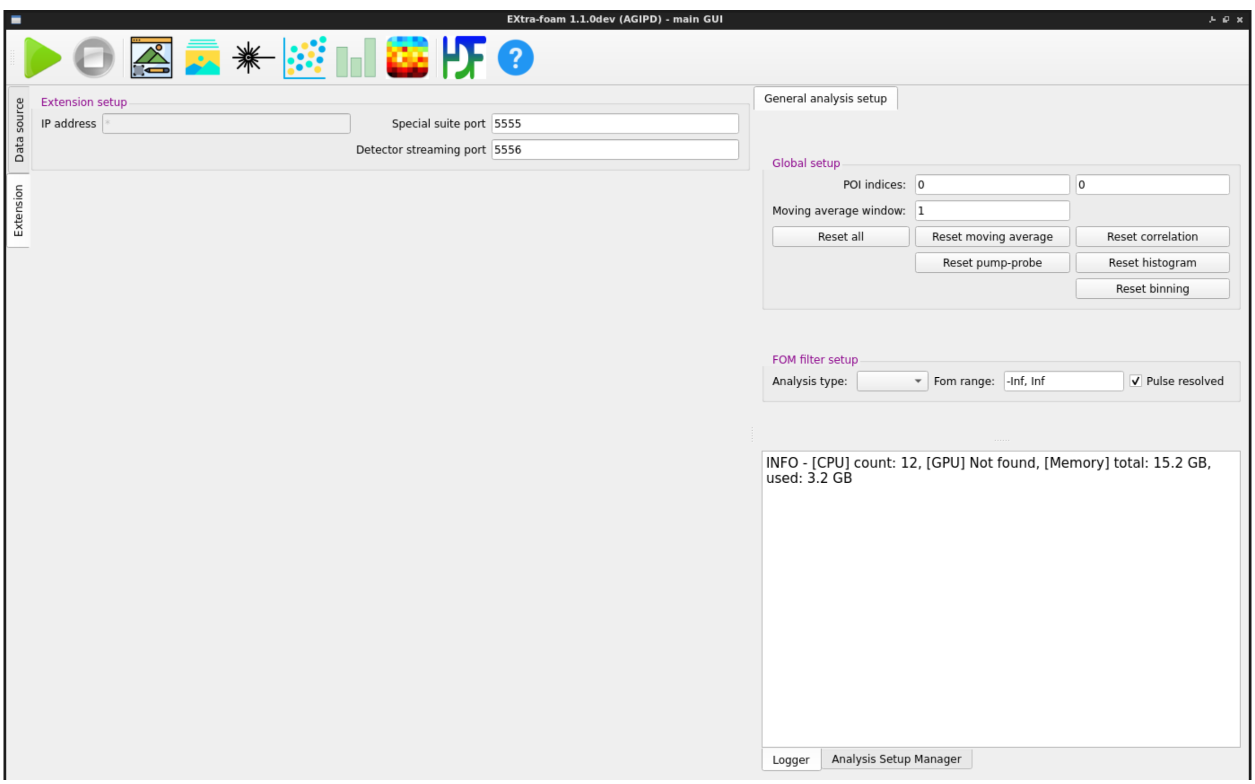


Input	Description
`Take snapshot`	Take a snapshot of the parameters and store them in both the `Last saved` and the selected analysis setups.
`Copy`	Make a copy of the selected analysis setup.
`Delete`	Delete the selected analysis setup.
`Rename`	Rename the selected analysis setup.

Warning: *Analysis setup manager* is still in the testing phase and we are collecting feedbacks from users. It should be noted that there is no backup recovery mechanism for now.

6.5 Extensions

On the left hand side there is a tab bar, one for the main GUI and the second for the extensions settings:



EXtra-foam supports streaming the processed data in two ways:

1. All processed data may be streamed to a *Special analysis suites*.
2. Just the processed and averaged detector image may be streamed over a Karabo bridge. This is could be helpful if, for example, an experiment requires using two detectors simultaneously. In this case EXtra-foam could be set up to read data from both a trainmatcher/Karabo bridge hosted in Karabo and another EXtra-foam instance. To get the detector data from a Karabo bridge client, use the key EF_<source_name>, where `source_name` is the source name of the main detector. For example, if streaming ePix data from the device `MID_EXP_EPIX-2/DET/RECEIVER:output` with property `data.image`, use `EF_MID_EXP_EPIX-2/DET/RECEIVER:output` and `data.image` at the client.

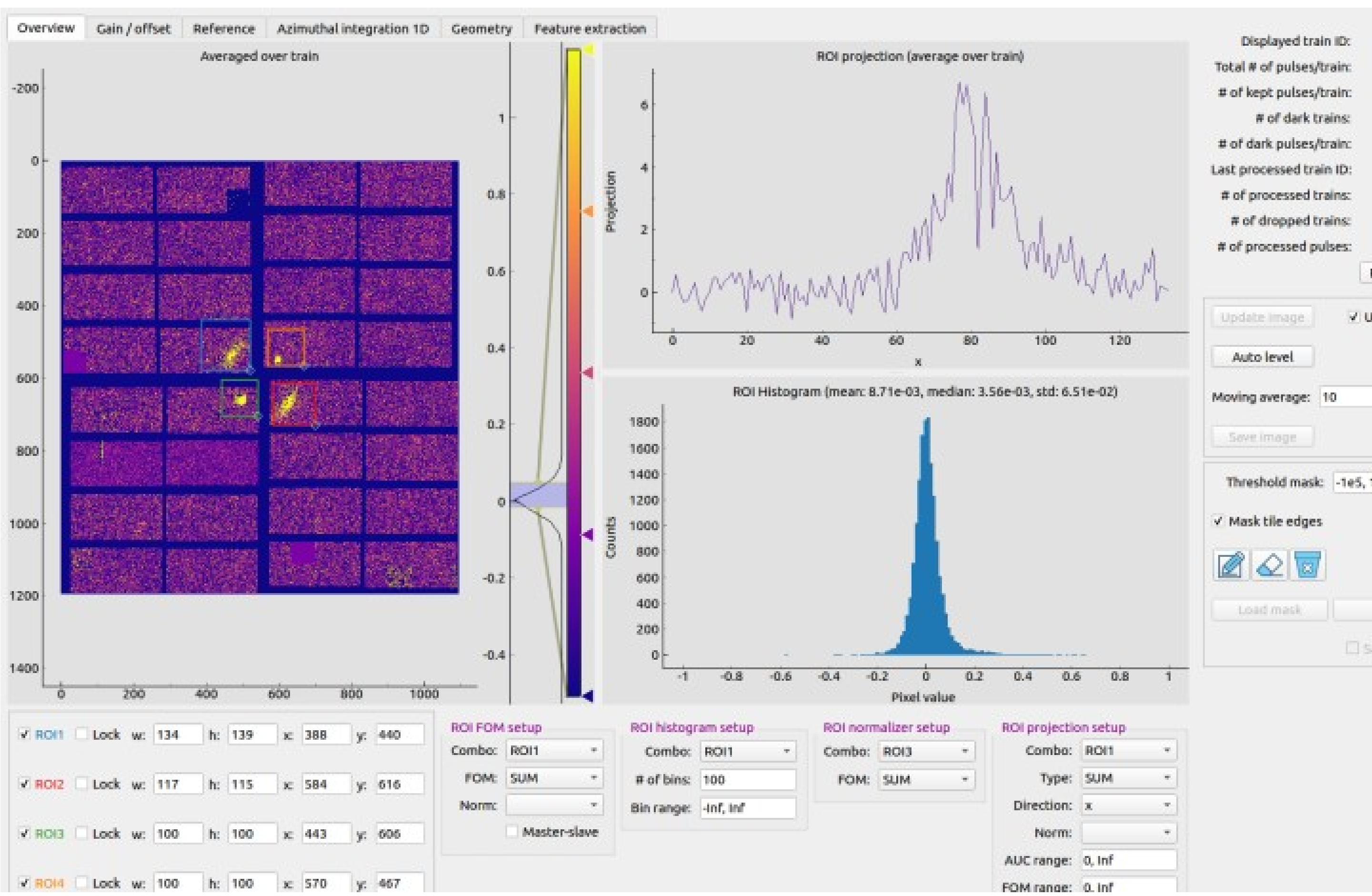
Warning: There are limited use-cases for streaming detector data to another EXtra-foam instance, contact da-support@xfel.eu if you would like to do this.

Input	Description
`Special suite port`	Port to use for streaming data for the special suites.
`Detector streaming port`	Port to use for streaming the processed 2D detector image.

CHAPTER 7

Image tool

The *ImageTool* window is the second control window which provides various image-related information and controls.



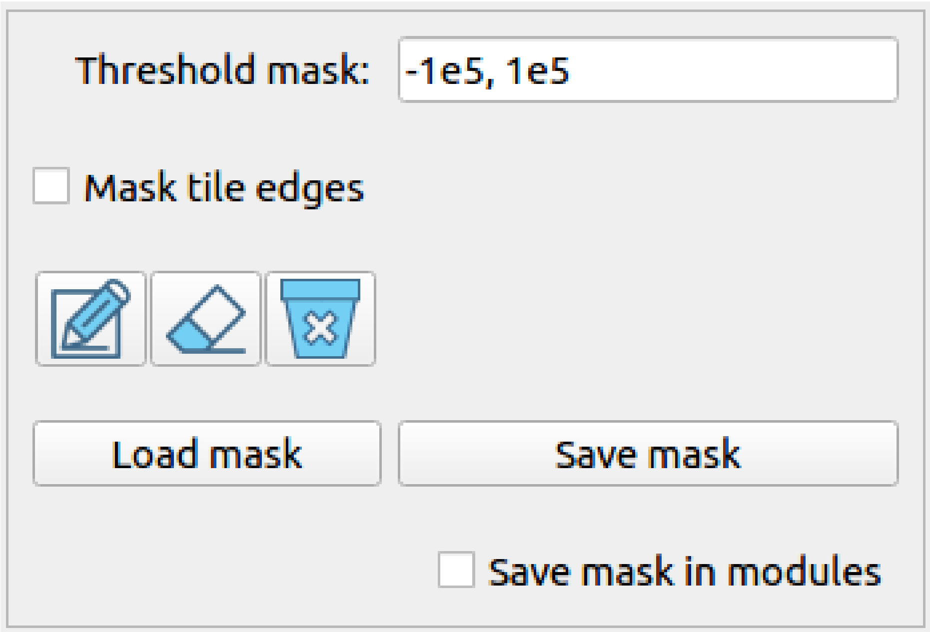
7.1 Image control

Input	Description
`Update image`	Manually update the current displayed image in the *ImageTool* window. Disabled if *Update automatically* is checked.
`Update automatically`	Automatically update the current displayed image in the *ImageTool* window.
`Moving average`	Apply moving average to the image data. It affects both the individual images in a train and the averaged image, as well as the subsequent analysis. If a new window size is smaller than the old one, the moving average calculation will start from the scratch.
`Auto level`	Update the detector images (not only in the *ImageTool* window, but also in other plot windows) by automatically selecting levels based on the maximum and minimum values in the data.
`Save image`	Save the current image to file. Please also see *ImageFileFormat*

Warning: The moving average here is not calculated by nanmean, which means that if a pixel of the image in a certain pulse is *NaN*, the moving average of that pixel will be *NaN* for that pulse.

Warning: Please be aware that there is another moving average setup in the *Global setup* in the main GUI.

7.1.1 Mask panel



Besides the nan pixels inherited from the calibration pipeline, users are allowed to mask additional pixels to nan in this panel with threshold mask, tile edge mask and image mask.

It should be noted that image mask is treated differently in **EXtra-foam**. One can draw and erase image mask at run time as well as save/load it as an assembled image or in modules if the detector has a geometry. Nan pixels outside the masked region of the image mask will not be saved and thus will also not be overwritten after loading an image mask from file.

Input	Description
`Threshold mask`	An interval that pixels with values outside the interval will be masked. Please distinguish *threshold mask* from clipping.
`Mask tile edges`	Mask the edge pixel of each tile. *Only applicable for AGIPD, LPD and DSSC if EXtra-foam is selected as the* `Assembler` *in* *Geometry*.
`Mask ASIC edges`	Mask the edge pixel of each ASIC. *Only applicable for JungFrau and ePix100.*
`Draw`	Draw mask in a rectangular region. *Only available in the Corrected panel.*
`Erase`	Erase mask in a rectangular region. *Only available in the Corrected panel.*
`Remove mask`	Remove the image mask.
`Load mask`	Load an image mask in *.npy* format. The dtype of the loaded numpy array will be casted into bool if it is not. For detectors with a geometry, it is allowed to load an image mask in modules, i.e., an array which has the shape (modules, ss, fs).
`Save mask`	Save the current image mask in *.npy* format.
`Save mask in modules`	Save image mask in modules. *Only applicable for AGIPD, LPD and DSSC if EXtra-foam is selected as the* `Assembler` *in* *Geometry*.

7.1.2 ROI manipulation

You can activate (tick **On**) up to 4 ROIs at the same time. One can change the size (**w**idth, **h**eight) and position (**x**, **y**) of an ROI by either dragging and moving the ROI on the image or entering numbers. You can avoid modifying an ROI unwittingly by **Lock**ing it.

7.1.3 ROI FOM setup

Input	Description
`Combo`	ROI combination, including *ROI1*, *ROI2*, *ROI1 + ROI2*, *ROI1 - ROI2*, and *ROI1 / ROI2*.
`FOM`	ROI FOM type, including *SUM*, *MEAN*, *MEDIAN*, *MIN*, *MAX*. *STD*, *VAR*, *STD (norm)* and *VAR (norm)*.
`Norm`	Normalizer of ROI FOM. *Only applicable for train-resolved and pump-probe analysis*.
`Master-slave`	Check to activate the *master-slave* model. This model is used exclusively in *Correlation* window. When it is activated, FOMs of ROI1 (master) and ROI2 (slave) will be plotted in the same correlation plot. For other statistics analysis like binning and histogram, only ROI1 FOM will be used.

- *STD (norm)* is defined as the ratio between the standard deviation and the mean.
- *VAR (norm)* is defined as the ratio between the variance and the square of the mean.

7.1.4 ROI histogram setup

Input	Description
`Combo`	ROI combination, e.g. *ROI1*, *ROI2*, *ROI1 + ROI2*, *ROI1 - ROI2*.
`Bin range`	Lower and upper boundaries of all the bins. In case of *+/- Inf*, the boundary will be calculated dynamically.
`# of bins`	Number of bins of the histogram.

7.1.5 ROI normalizer setup

The settings for this are on the *ROI normalizer settings* tab. The normalization source can either be the main detector, or some other 2D source image. To use a different source, such as a camera, add it as a pipeline source (purple square) in the *Data source tree* under the *User-defined* section, and if it is a 2D image it will be displayed as an option in the *ROI source* list.

Input	Description
`ROI source`	Source to compute the normalization factor from.
`Combo`	ROI combination, e.g. *ROI3*, *ROI4*, *ROI3 + ROI4*, *ROI3 - ROI4*.
`FOM`	ROI FOM type, e.g. *SUM*, *MEAN*, *MEDIAN*, *MIN*, *MAX*.

7.1.6 ROI projection setup

Define the 1D projection of ROI (region of interest) analysis setup.

Input	Description
`Combo`	ROI combination, e.g. *ROI1*, *ROI2*, *ROI1 + ROI2*, *ROI1 - ROI2*.
`Direction`	Direction of 1D projection (x or y).
`Norm`	Normalizer of the 1D-projection VFOM. *Only applicable for train-resolved and pump-probe analysis.*
`AUC range`	AUC (area under a curve) integration range.
`FOM range`	Integration range when calculating the figure-of-merit of 1D projection.

7.1.7 Photon binning setup

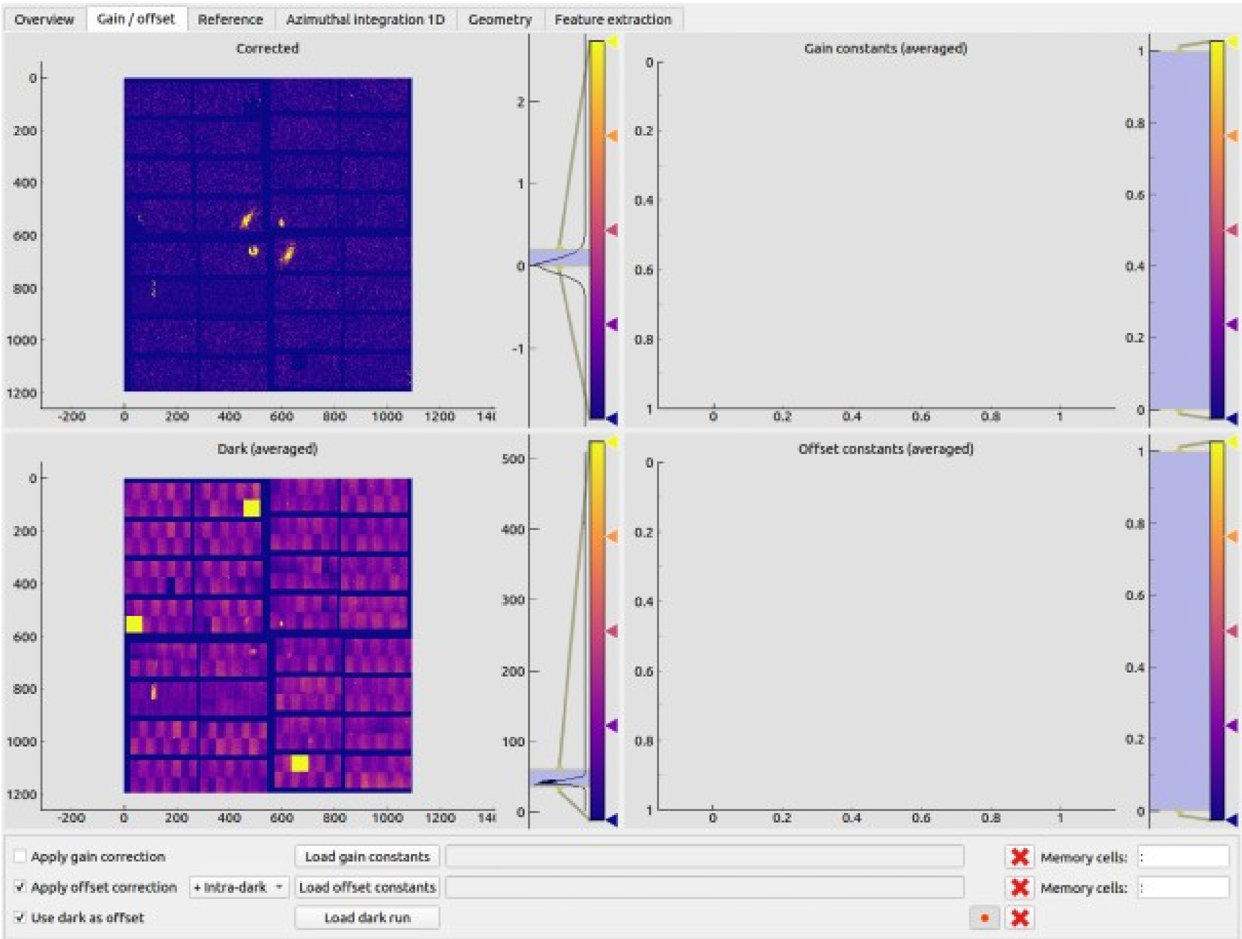
This bins the data from the detector based on an ADU threshold, which corresponds to the raw value recorded by a detector for one photon. The binning uses the formula:

$$bin = \left\lfloor \frac{\frac{T}{2} + A_{raw}}{T} \right\rfloor$$

$$A_{binned} = clamp(bin, 0, \infty)$$

Input	Description
`ADU threshold`	The ADU threshold to use for binning. A reasonable value for this could be found from the histogram of the image by looking for a visible ‘step’ between the counts of one and two photons.

7.2 Gain / offset



Apply pixel-wised gain and offset correction, where

$$A_{corrected} = (A_{raw} - I_{offset}) \cdot I_{gain}$$

Users can record a "dark run" whenever data is available. The dark run consists of a number of trains. The moving average of the each "dark pulse" in the train will be calculated, which will then be used to apply dark subtraction to image data pulse-by-pulse.

Input	Description
`Apply gain correction`	Check to activate gain correction.
`Apply offset correction`	Check to activate offset correction. Since version 1.10, a variation of offset correction has been introduced: • `+intra-dark`: After the pulse-by-pulse offset correction, every other pulse will be subtracted by the following one starting from the first pulse. For instance, imaging a pulse train consisting of four pulses *ABAB*, the extra intra-dark correction will result in a train *A'BA'B*, where *A' = A - B*. Then one can make use of the pulse slicer in the *Data source tree* to remove the intra-dark pulses.
`Use dark as offset`	Check to use recorded dark images as offset. The already loaded offset constants will be ignored.
`Record dark`	Start and stop dark run recording.
`Remove dark`	Remove the recorded dark run.

Warning: The moving average here is not calculated by nanmean, which means that if a pixel of the image in a certain pulse is *NaN*, the moving average of that pixel will be *NaN* for that pulse.

Note: Some detectors have its own special treatment for gain/offset correction:

- DSSC:

  Due to the readout issue, pixels with value 0 will be converted to 256.

7.3 Reference image

Input	Description
`Load reference`	Load a reference image from file. Please also see *ImageFileFormat*
`Set current as reference`	Set the current displayed image as a reference image. For now, reference image is used as a stationary off-image in the *predefined off* mode in *pump-probe* analysis.
`Record reference`	Record the received displayed images and perform a moving average until the `Stop` (recording) button has been toggled. The resulting image will be set as a reference image.
`Save reference`	Saves the reference image to a *NumPy* file.
`Remove reference`	Remove the reference image.

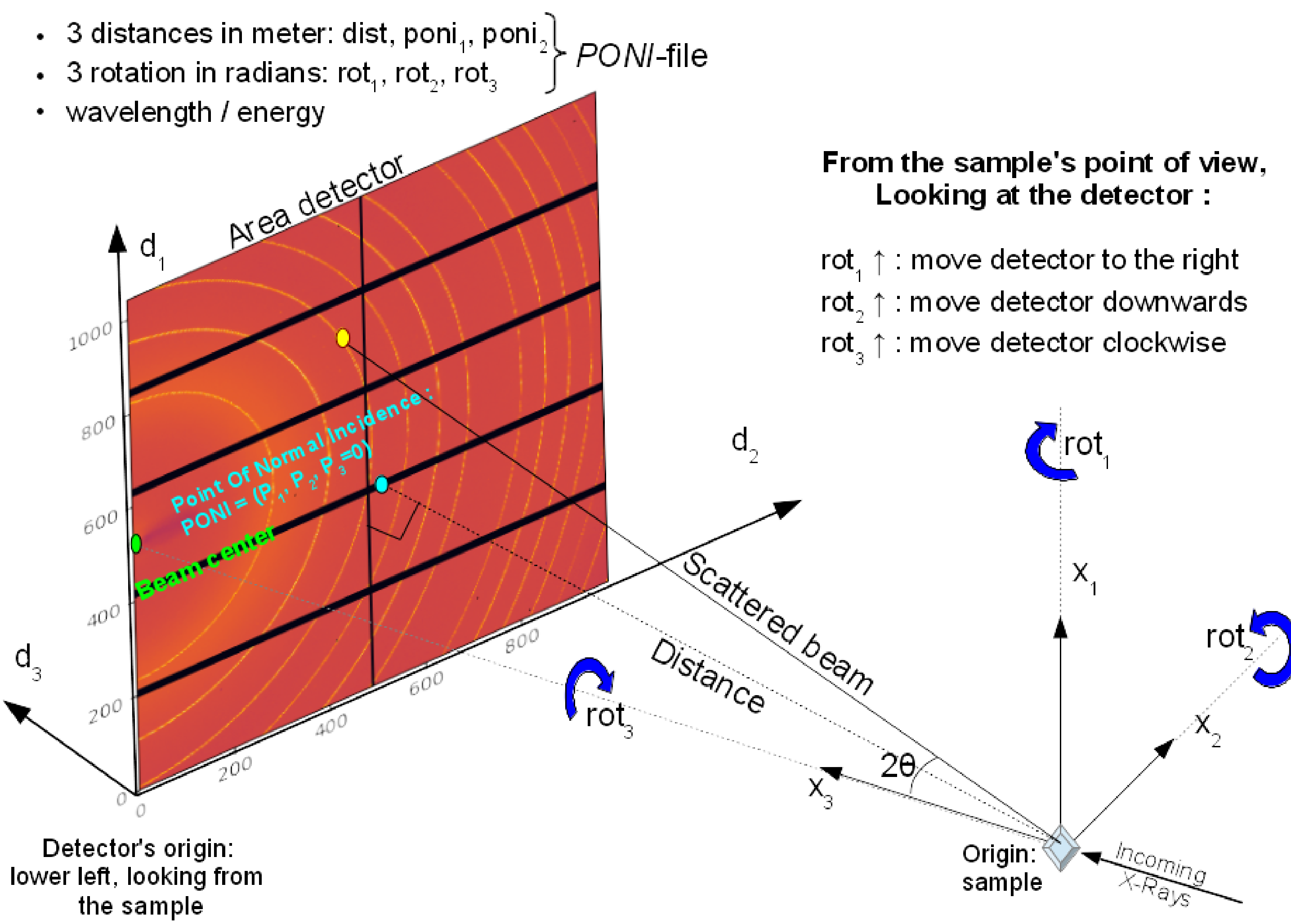
Note: Image file format

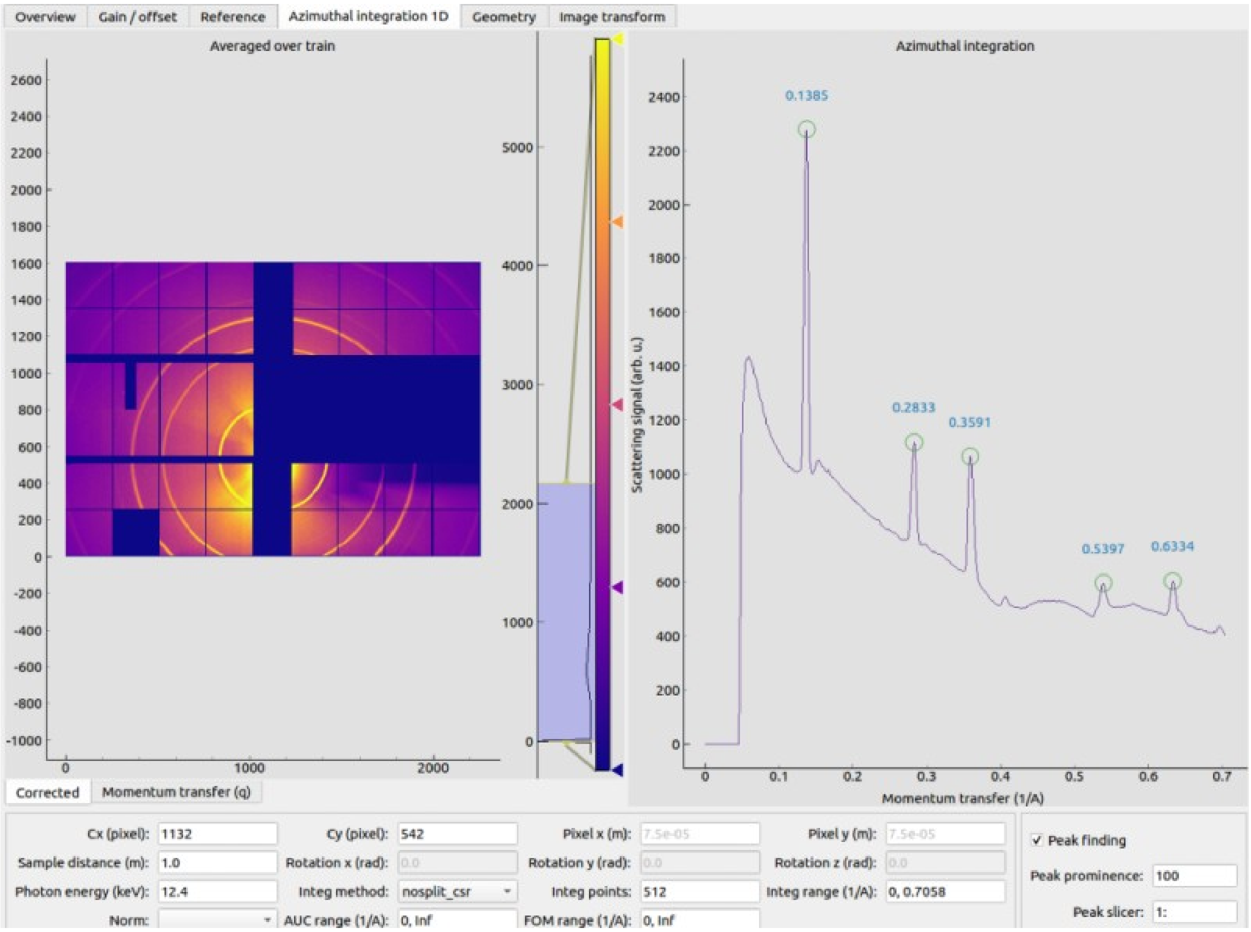
The two recommended image file formats are *.npy* and *.tif*. However, depending on the OS, the opened file dialog may allow you to enter any filename. Therefore, in principle, users can save and load any other image file formats supported by imageio. However, it can be wrong if one writes and then loads a *.png* file due to the auto scaling of pixel values.

7.4 Azimuthal integration 1D

EXtra-foam uses pyFAI to do azimuthal integration. As illustrated in the sketch below, the **origin** is located at the sample position, more precisely, where the X-ray beam crosses the main axis of the diffractometer. The detector is treated as a rigid body, and its position in space is described by six parameters: 3 translations and 3 rotations. The orthogonal projection of **origin** on the detector surface is called **PONI** (Point Of Normal Incidence). For non-planar detectors, **PONI** is defined in the plan with z=0 in the detector's coordinate system. It is worth noting that usually **PONI** is not the beam center on the detector surface.

The input parameters *Cx* and *Cy* correspond to *Poni2* and *Poni1* in the aforementioned coordinate system, respectively.





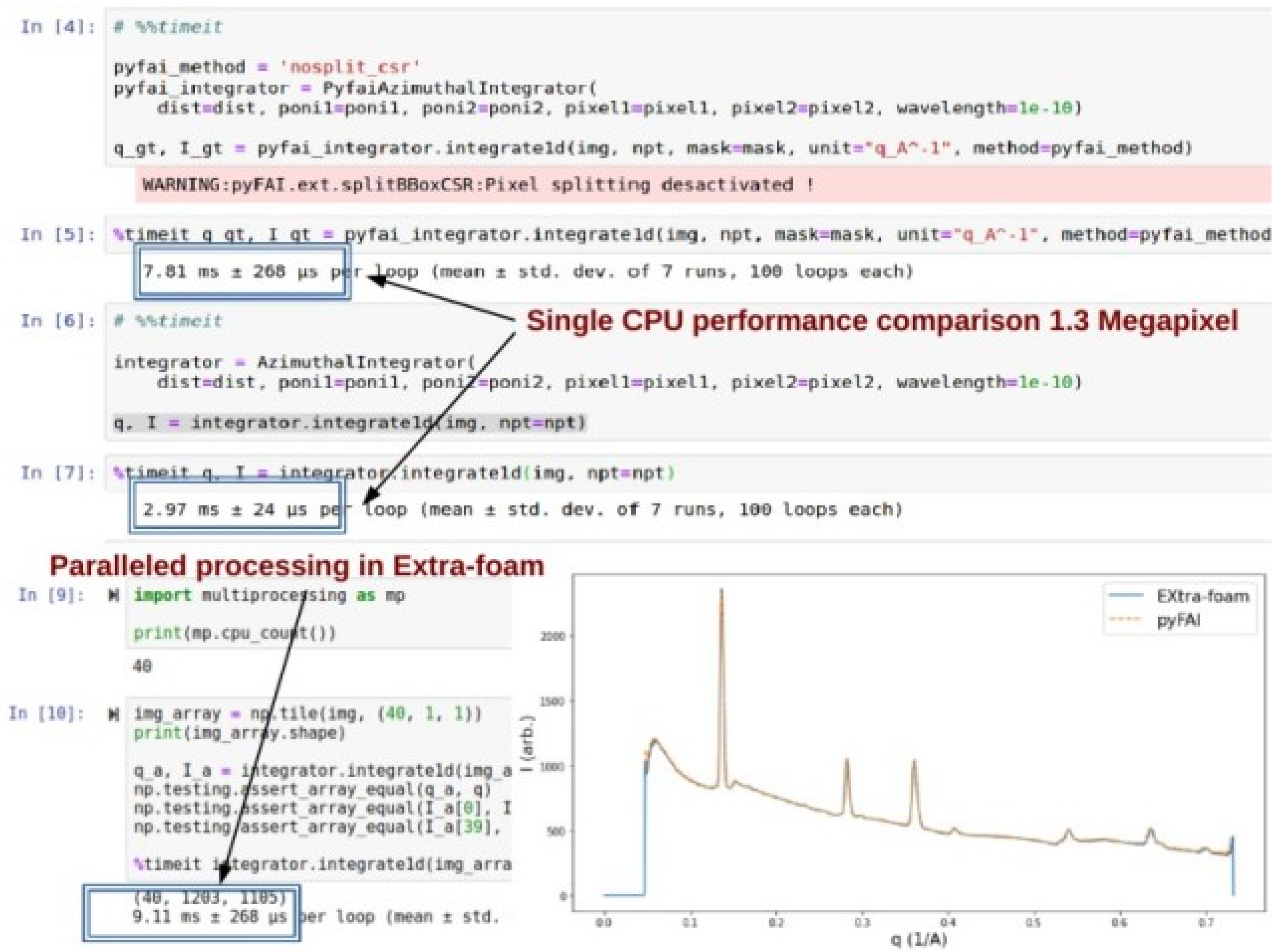
Input	Description
`Cx (pixel)`	Coordinate of the point of normal incidence along the detector's 2nd dimension.
`Cy (pixel)`	Coordinate of the point of normal incidence along the detector's 1st dimension.
`Pixel x (m)`	Pixel size along the detector's 2nd dimension.
`Pixel y (m)`	Pixel size along the detector's 1st dimension.
`Sample distance`	Sample-detector distance in m. Only used in azimuthal integration.
`Rotation x (rad)`	*Not used*
`Rotation y (rad)`	*Not used*
`Rotation z (rad)`	*Not used*
`Photon energy (keV)`	Photon energy in keV. Only used in azimuthal integration for now.
`Integ method`	Azimuthal integration methods provided by pyFAI.
`Integ points`	Number of points in the output pattern of azimuthal integration.
`Integ range (1/A)`	Azimuthal integration range.
`Norm`	Normalizer of the scattering curve. *Only applicable for train-resolved and pump-probe analysis*.
`AUC range (1/A)`	AUC (area under curve) range.
`FOM range (1/A)`	Integration range when calculating the figure-of-merit of the azimuthal integration result.

By default, peak finding is activated and peak positions will be annotated along the scattering curve if the number of detected peaks is between 1 and 10. There is no special reason for choosing 10 as the upper limit. Nevertheless, if there are two many peaks found, it may be due to a noisy scattering curve or some unreasonable peak-finding parameters.

For now, users can set prominence to refine the number of detected peaks and use a slicer to select part of them. The prominence of a peak measures how much a peak stands out from the surrounding baseline of the signal and is defined as the vertical distance between the peak and its lowest contour line. The slicer is useful when the scattering curve has some undesired structure, especially at the start and/or end of the curve.

Input	Description
`Peak finding`	Check to activate real-time peak finding and annotating.
`Peak prominence`	Minimum prominence of peaks.
`Peak slicer`	Pixel size along the detector's 2nd dimension.

EXtra-foam also has its own fast azimuthal integration implemented in C++. On a cluster with 40 cores, it takes about only 9 ms to integrate a train of 40 1.3-Megapixel images. Unfortunately, this implementation has not been integrated into the GUI for now.

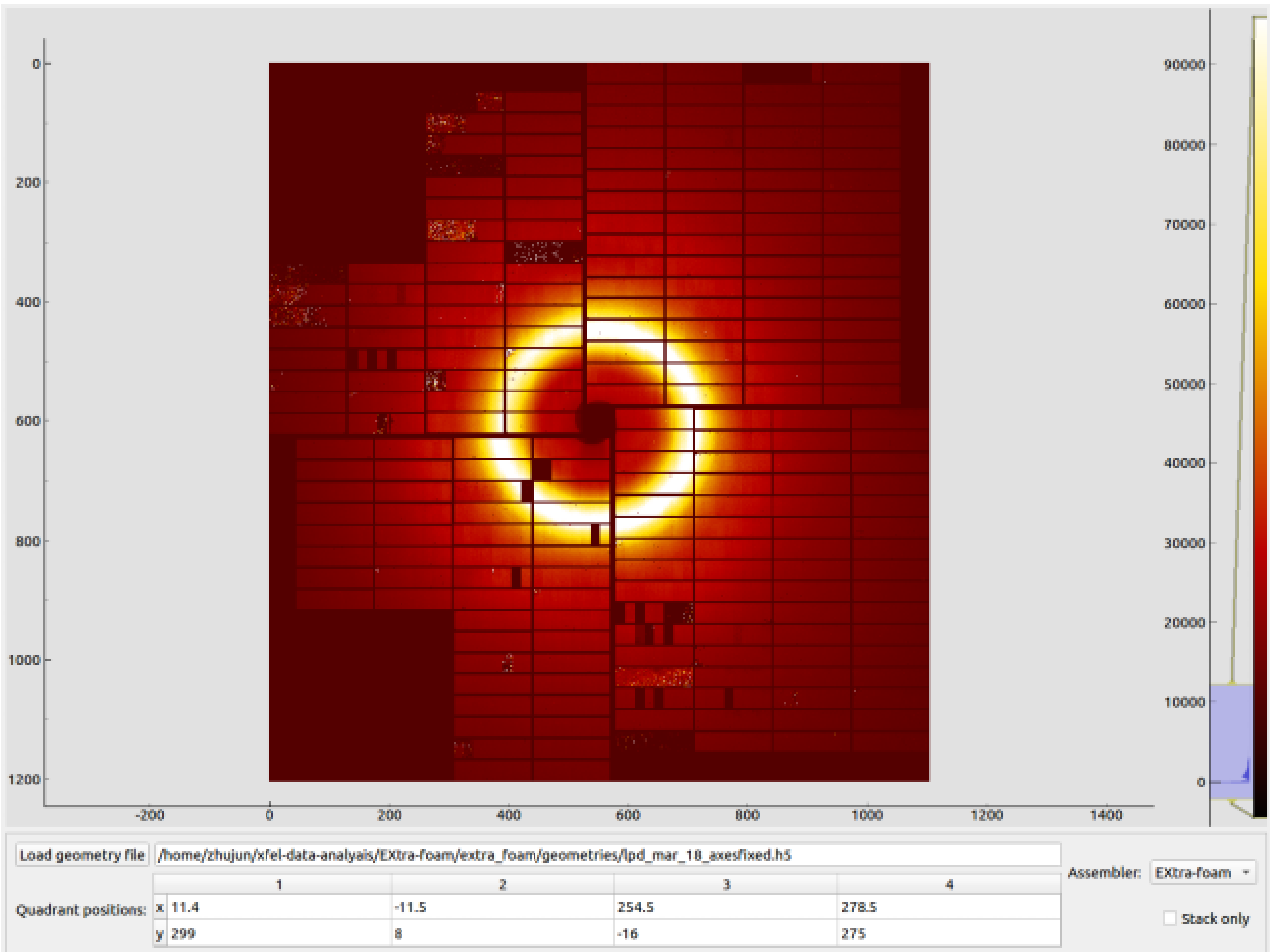


7.5 Geometry

Geometry is only available for the detector which requires a geometry to assemble the images from different modules, for example, AGIPD, LPD, DSSC as well as JungFrau and ePix100 used in a combined way.

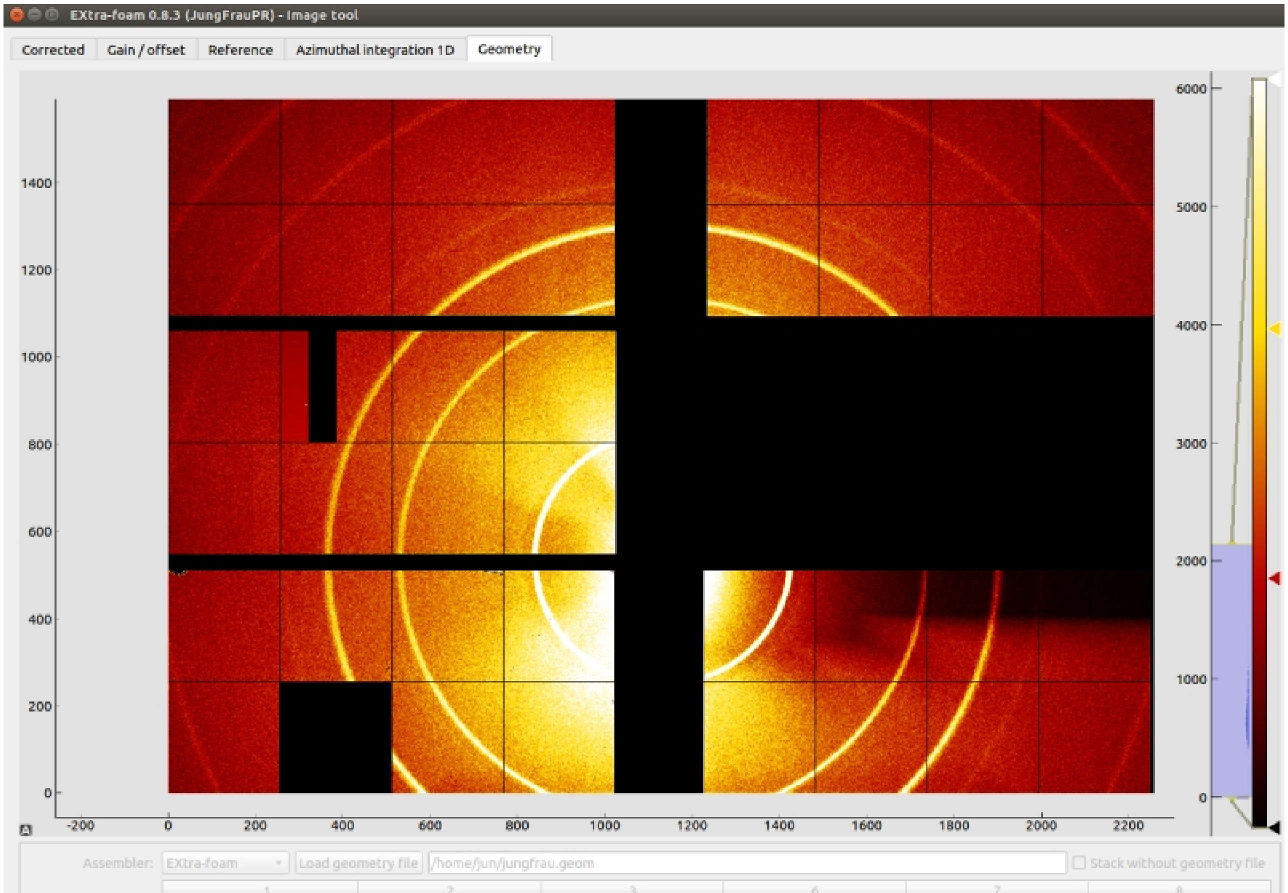
For details about geometries of AGIPD, LPD and DSSC, please refer to this documentation. It should be noted that the online and offline data format are different. For real-time data received from the *ZMQ bridge*, all the 16 modules have been stacked in a single array and the source name is usually a Karabo device name. However, for data streamed from files, modules data are distributed in different files and each module has a unique source name. For example, DSSC modules at SCS are named as *SCS_DET_DSSC1M-1/DET/0CH0:xtdf*, *SCS_DET_DSSC1M-1/DET/1CH0:xtdf*, . . . , *SCS_DET_DSSC1M-1/DET/15CH0:xtdf*. **EXtra-foam** relies on the "index" (0 - 15) in the source name to find the corresponding module. Accordingly, in the *Data source* tree, one should use *SCS_DET_DSSC1M-1/DET/*CH0:xtdf* as the source name, which has a '*' at the location where the module index is expected.

LPD-1M with 16 modules:

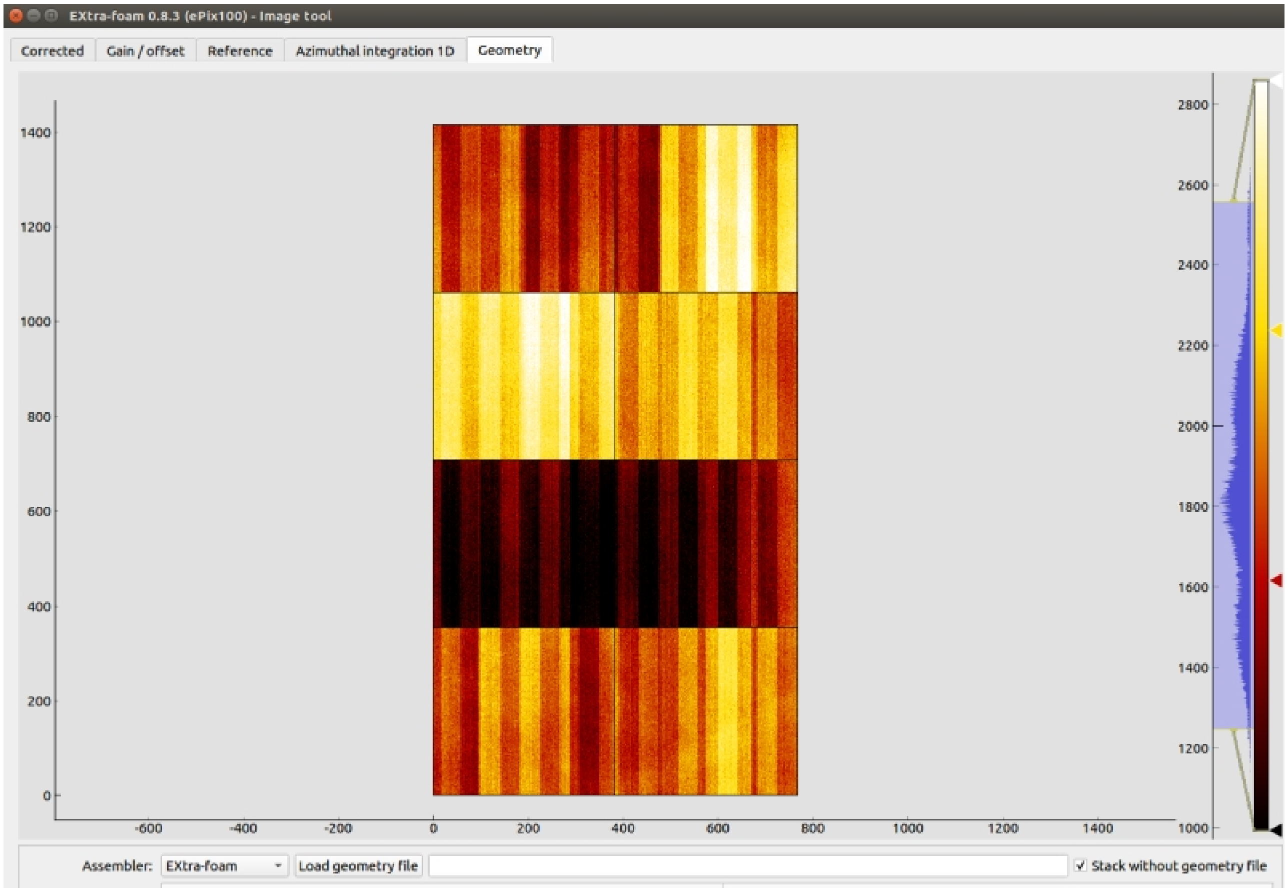


EXtra-foam implemented a generalized geometry for detectors like JungFrau and ePix100. To allow more than one modules, **one must explicitly specify the number of modules in the command line at startup**. Similar to AGIPD, LPD and DSSC, the online and offline data format can be different. For real-time data received from the *ZMQ bridge*, all the modules could have been stacked in a single array and the source name is usually a Karabo device name. However, it also supports data arriving in modules, as data streamed from files. Similarly, it relies on the "index" in the source name to find the corresponding module. Different from AGIPD, LPD and DSSC, **the module index starts from 1**. For example, JungFrau modules at SPB are named as *SPB_IRDA_JNGFR/DET/MODULE_1:daqOutput*, *SPB_IRDA_JNGFR/DET/MODULE_2:daqOutput*, . . . , *SPB_IRDA_JNGFR/DET/MODULE_8:daqOutput*. Similarly, in the *Data source* tree, one should use *SPB_IRDA_JNGFR/DET/MODULE_*:daqOutput* as the source name.

6-module JungFrau with geometry file in the CFEL format. Module 1 is located on the top-right corner and all modules (1, 2, 3, 6, 7, 8) are arranged in closewise order.



2-module ePix100 without geometry file. Module 1 is located on top of module 2.



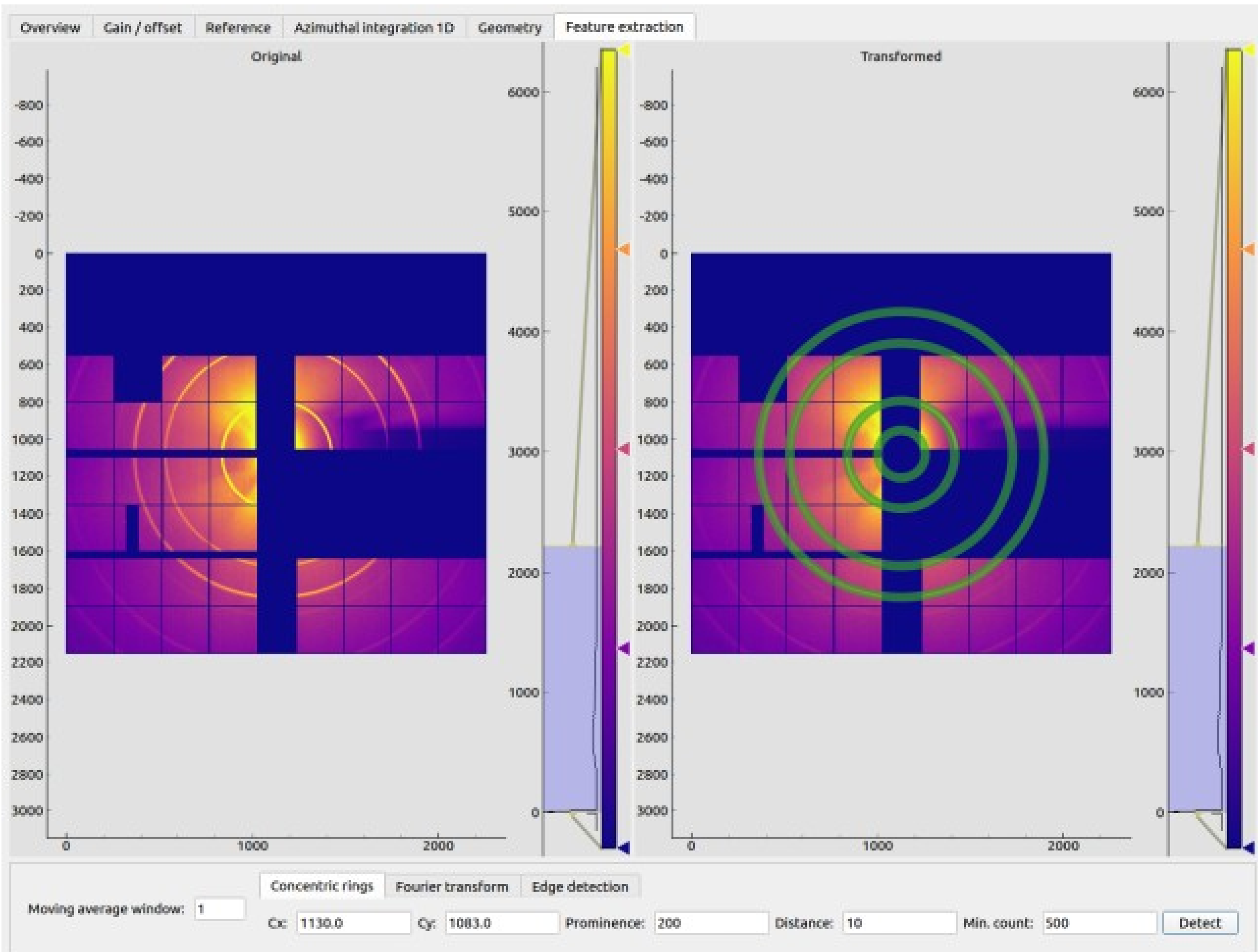
Input	Description
`Quadrant positions`	The first pixel of the first module in each quadrant, corresponding to data channels 0, 4, 8 and 12. *Only avaible for 1M detectors, i.e. AGIPD, LPD and DSSC, with non-CFEL format geometry file.*
`Module positions`	The first pixel of each module. *Only available for JungFrau and ePix100 with non-CFEL format geometry file. Not implemented yet*
`Load geometry file`	Open a *FileDialog* window to choose a geometry file from the local file system. *Ignored if* `Stack without geometry file` *is checked.*
`Assembler`	There are two assemblers available in *EXtra-foam* for AGIPD, LPD and DSSC. One is EXtra-geom implemented in Python and the other is the local C++ implementation. Indeed, the latter follows the assembling methodology implemented in the former but is much faster with multi-core processors.
`Stack without geometry file`	When the checkbox is checked, the modules will be seamlessly stacked together. Unfortunately, it does not mean that this will be faster than assembling with a geometry. It simply provides an alternative to check the data from different modules.

7.6 Feature Extraction

Here, one can visualize the original image and its transform side by side. The transformed image can be further used for feature extraction. A feature extraction analysis will be activated only if the corresponding control widget tab is activated. *Not all transformed images support feature extraction and not all feature extractions require a prior image* *transform*.

Input	Description
`Moving average window`	Use moving averaged image to suppress background noise and enhance features.

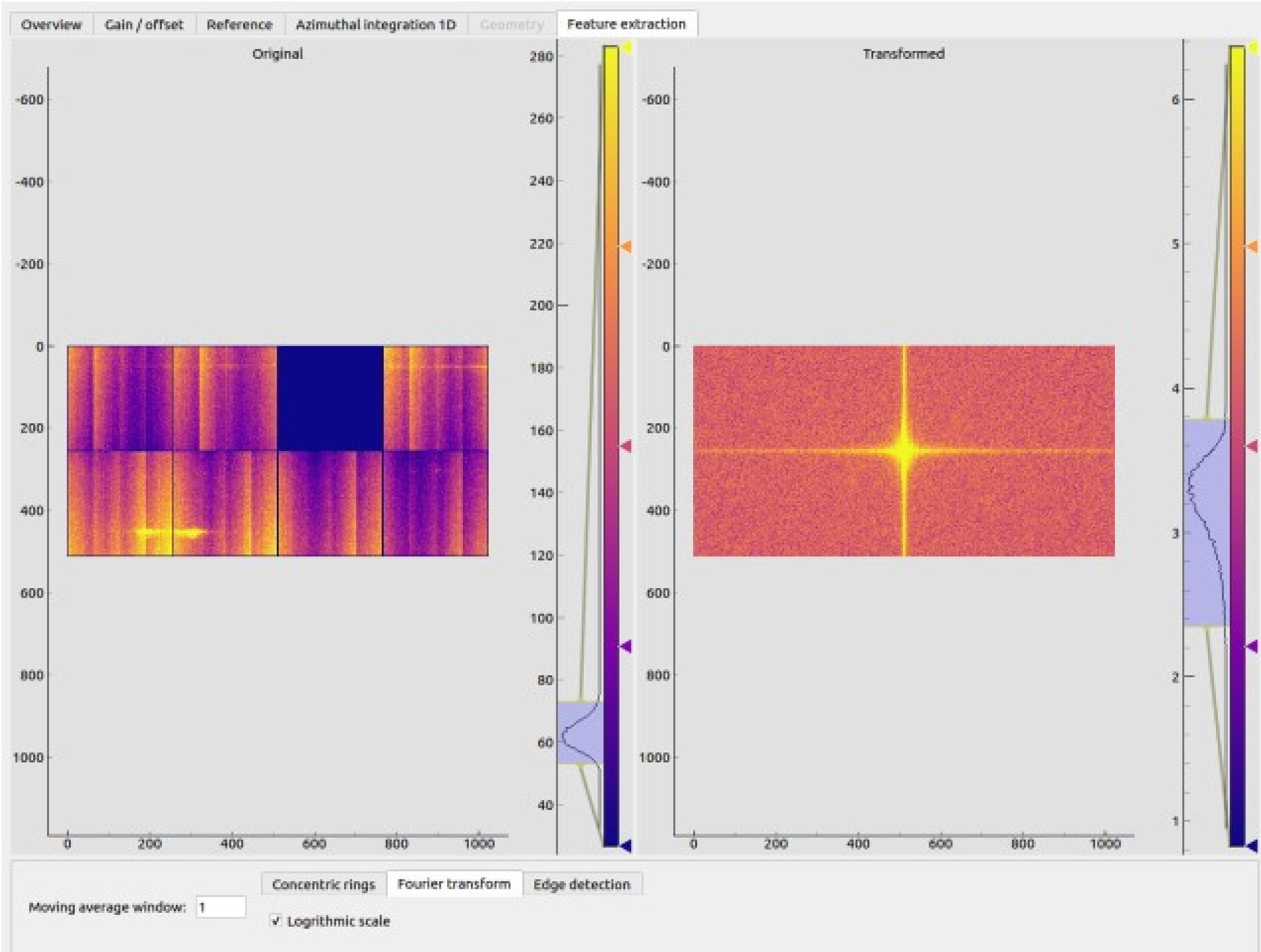
7.6.1 Concentric rings



Find the center of concentric rings in an image. It is typically used in finding the center for *Azimuthal integration 1D*. It is only available when the data processing pipeline is not running, i.e., it cannot be used in real-time analysis.

Input	Description
`Cx`	Initial guess for the x coordinate of the center, in pixel.
`Cy`	Initial guess for the y coordinate of the center, in pixel.
`Prominence`	Prominence of the ring.
`distance`	Minimum horizontal distance between neighbouring rings.
`Min. count`	Minimum number of valid pixels required for the ring. The nan pixels are excluded.
`Detect`	Click to find the optimized center. If found, the number in `Cx` and `Cy` will be updated and the detected rings will be marked in the transformed image.

7.6.2 Fourier transform



Apply 2D discrete Fourier Transform to the original image and shift the zero-frequency component to the center of the spectrum using fft package in scipy.

Input	Description
`Logrithmic scale`	Check to display the amplitude in logrithmic scale.

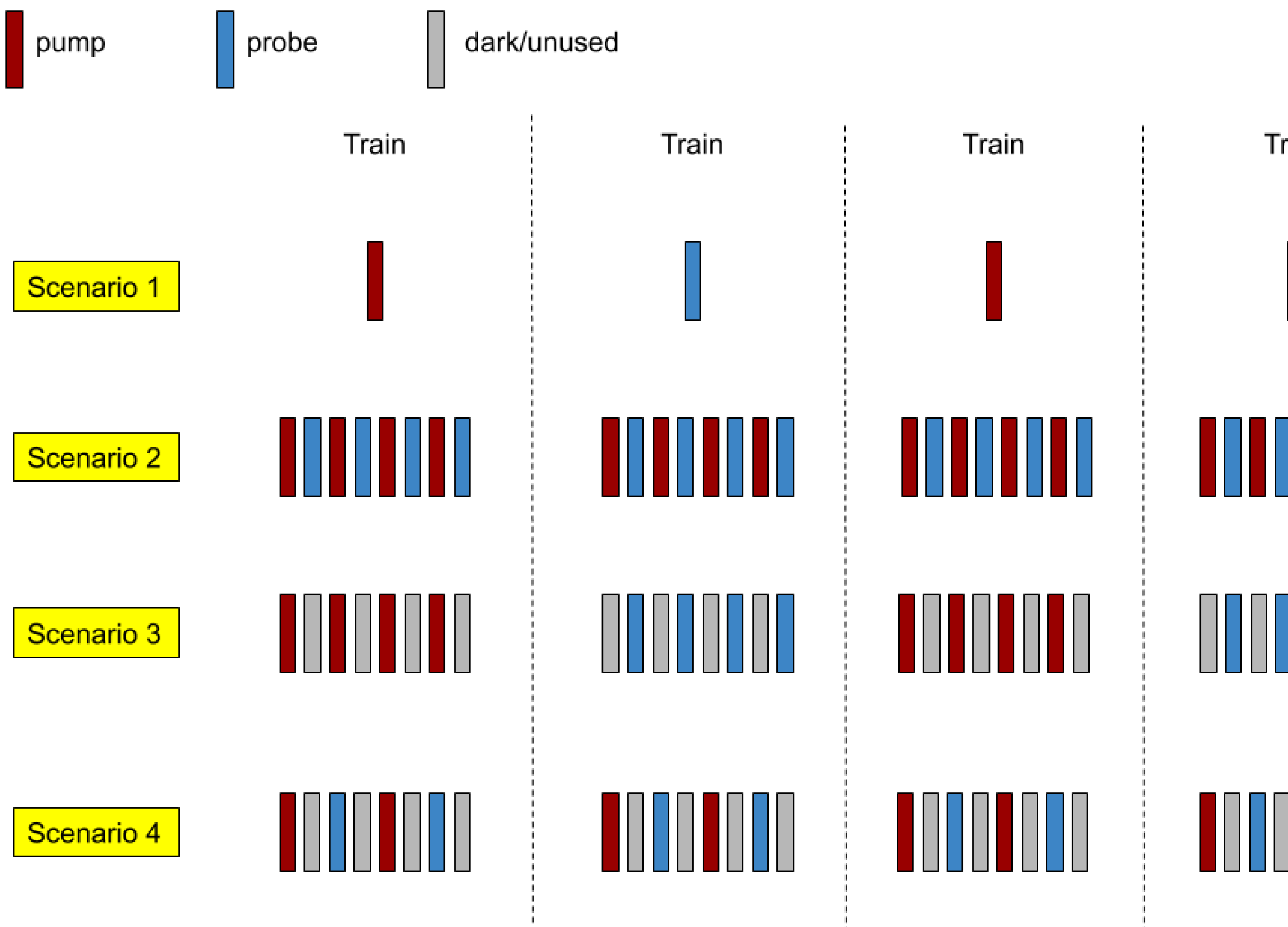
7.6.3 Edge detection

Detect edges in the original image and the transformed image is a binary image which shows the edge and non-edge pixels. **EXtra-foam** uses a similar algorithm to Canny edge detection to detect edges.

Input	Description
`Kernel size`	kernel size for Gaussian blur.
`Sigma`	Gaussian kernel standard deviation.
`Threshold`	(first, second) thresholds for the hysteresis procedure.

CHAPTER 8

Pump-probe analysis



In the *pump-probe* analysis, the average (nanmean) images of the on- and off- pulses in a train are calculated by

$$\bar{A}_{on} = \Sigma A_{on}/N_{on}$$
$$\bar{A}_{off} = \Sigma A_{off}/N_{off}.$$

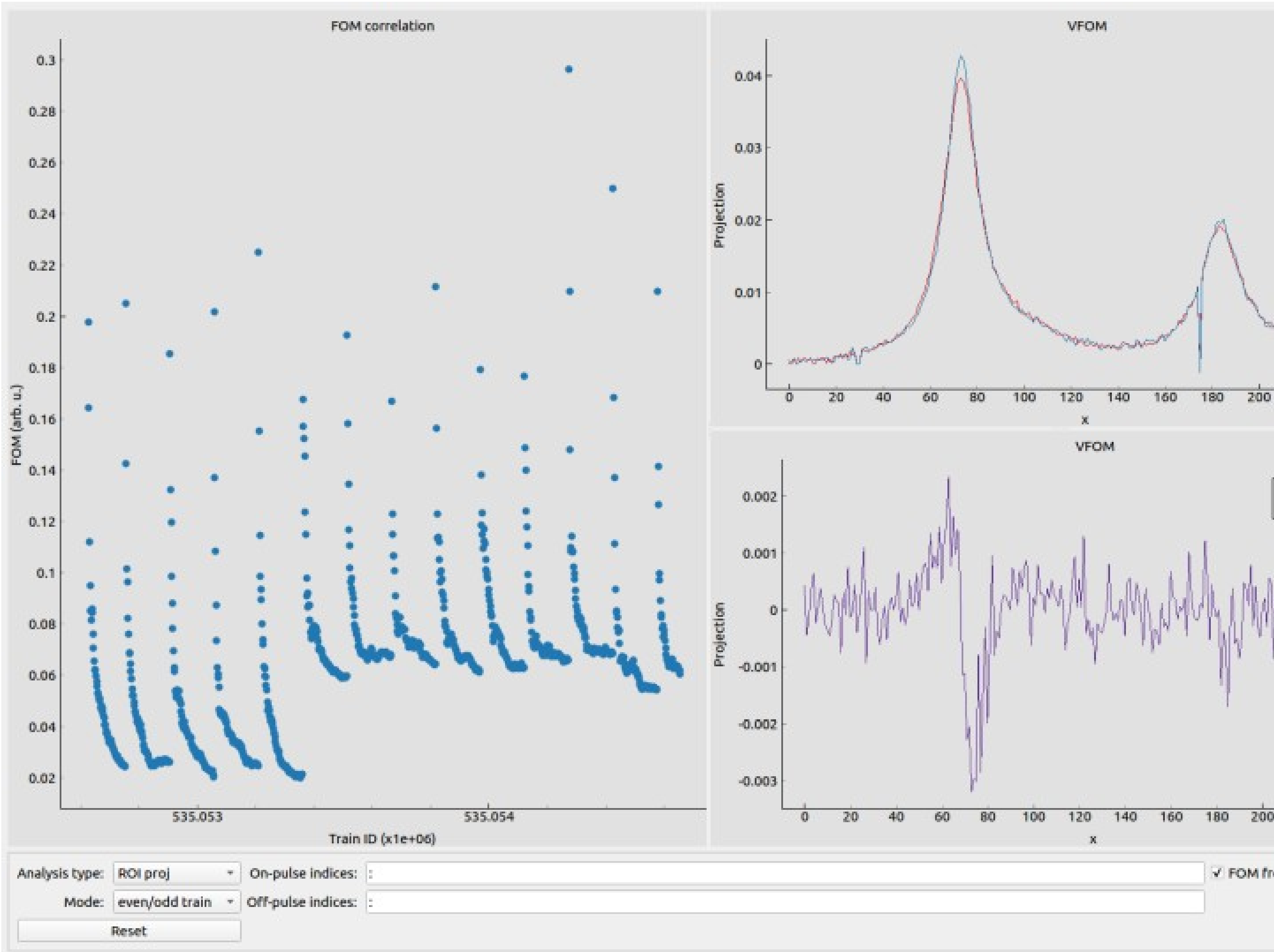
If the sub-Analysis type has VFOM, the VFOM K_v and FOM K are given by

$$K_v = f(\bar{A}_{on}) - f(\bar{A}_{off})$$
$$K = \Sigma K_v.$$

Otherwise, FOM K is given by

$$K = g(\bar{A}_{on}) - g(\bar{A}_{off}).$$

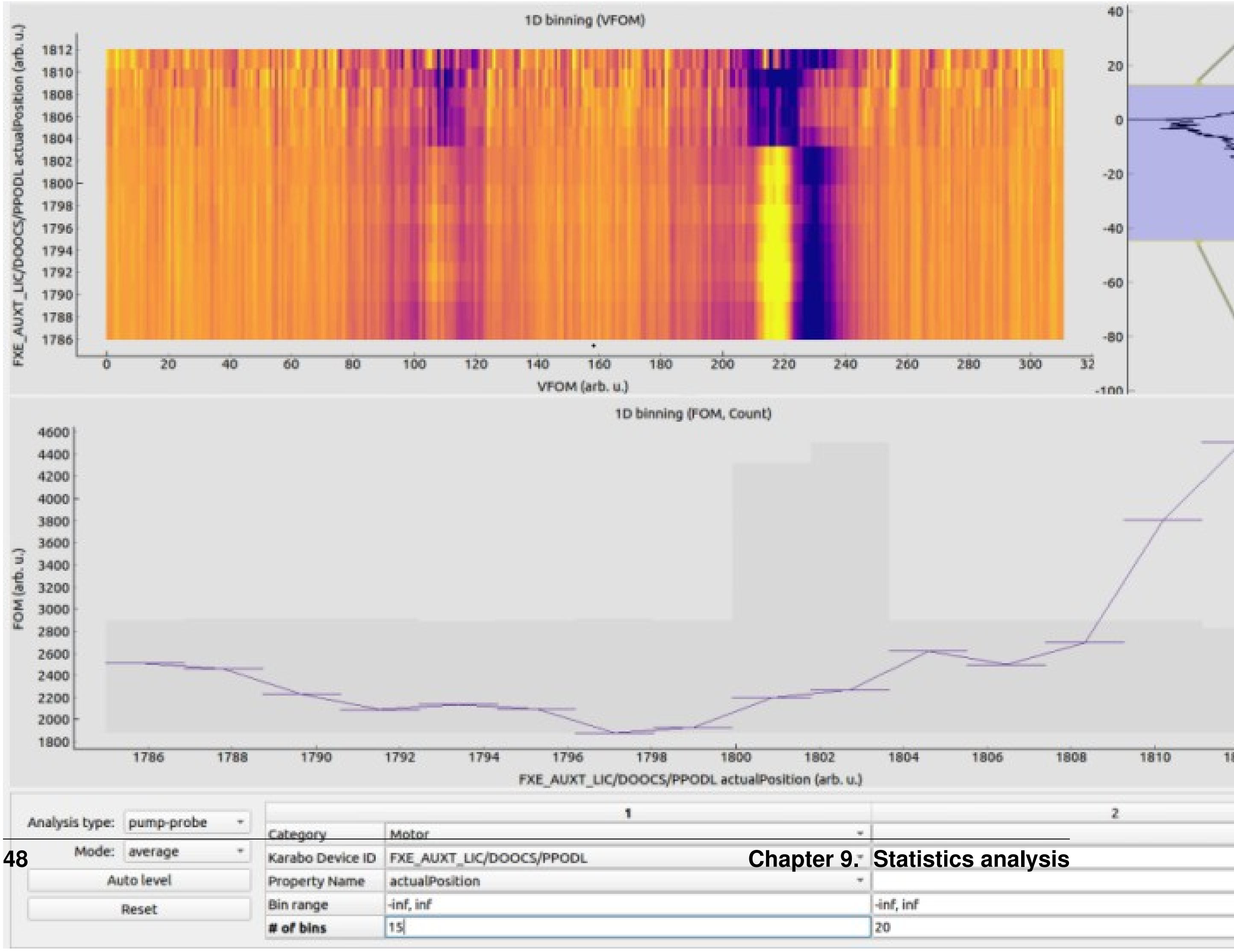
Input	Description
`On/off mode`	Pump-probe analysis mode: • `reference as off`: On-pulses will be taken from each train while the 'off' (reference image) is specified in the ImageTool. • `same train`: On-pulses and off-pulses will be taken from the same train. Not applicable to train-resolved detectors. • `even\/odd`: On-pulses will be taken from trains with even train IDs while off-pulses will be taken from trains with odd train IDs. • `odd\/even`: On-pulses will be taken from trains with odd train IDs while off-pulses will be taken from trains with even train IDs.
`Analysis type`	See Analysis type.
`On-pulse indices`	Indices of all on-pulses. **If 'Pulse slicer' is used to slice a portion of the pulses in the train, these indices are indeed the indices of the pulse in the sliced train**. *Pulse-resolved detector only.*
`Off-pulse indices`	Indices of all off-pulses. *Pulse-resolved detector only.*
`FOM from absolute on-off`	If this checkbox is ticked, the FOM will be calculated based on |*on - off*| (default). Otherwise *on - off*.
`Reset`	Reset the FOM plot in the *Pump-probe window* and the global moving average count.


FOM correlation
FOM (arb. u.)
Train ID (x1e+06)
535.053
535.054
VFOM
Projection
x
VFOM
Projection
x
Analysis type: ROI proj
On-pulse indices: :
Mode: even/odd train
Off-pulse indices: :
Reset

CHAPTER 9

Statistics analysis

9.1 Binning



The binning window allows to setup the visualization of 1D/2D binning of the FOM and VFOM for different analysis types. For now, it works only with train-resolved data. For instance, you can bin the sum of a ROI of the averaged frames in a train with respect to a motor position, while binning the sum of a ROI of each frame in a train with respect to a motor position or the XGM intensity in each corresponding pulse is not supported yet. As a result, internally it maintains a buffer with a length of **18,000**, which corresponds to the number of data points collected in 30 minutes. The old data point will be discarded when the buffer is full.

The first binning parameter is used for both 1D and 2D binning, while the second binning parameter is only used for 2D binning. When any of the `Mode`, `Bin range` and `# of bins` changes, the internally cached data will be re-binned. However, the program is smart enough to clear the cached data under certain conditions:

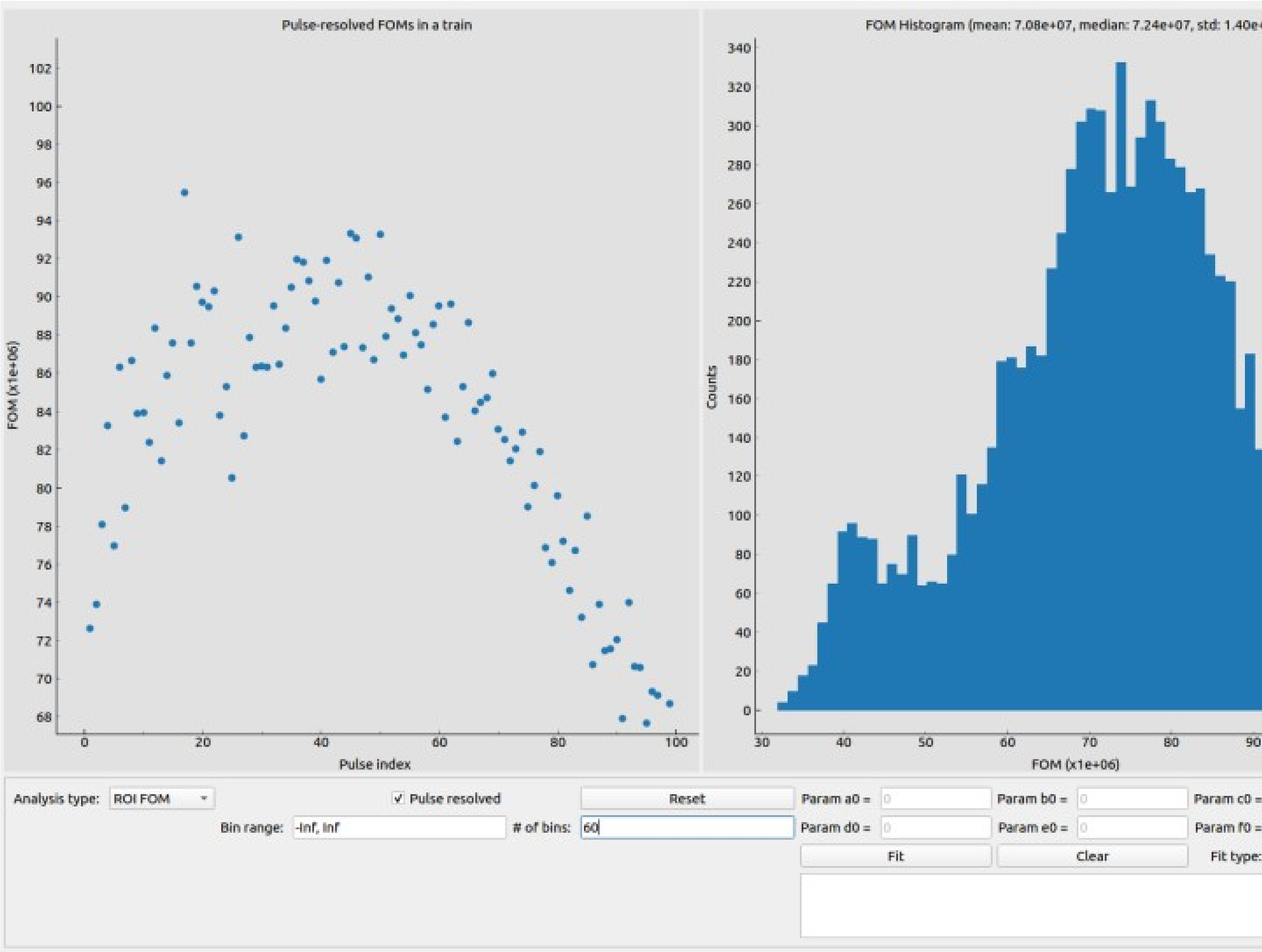
- `Analysis type` changed;
- The sub analysis type of the pump-probe analysis changed if the current analysis type if *pump-probe*;
- `Karabo Device ID` and/or `Property name` changed. It is worthy of mentioning that, if the second binning parameter was removed, the on-going 1D binning will not be affected, i.e. the cached data will not be cleared. However, if the second binning parameter was added, the on-going 1D binning will start from scratch. The rational behind the design is that the internally cached 1D and 2D binning data must be aligned;
- The length of VFOM changed. For instance, the shape of the ROI changed when the analysis type is `ROI proj`.

, since it does not make sense to mix the new and old data together. In this case, users don't need to remind themselves to click the reset button.

Input	Description
`Analysis type`	See Analysis type.
`Mode`	The data in each bin will be • *average*: averaged; • *accumulate*: summed up.
`Reset`	Reset the binning history.
`Category`	Category of the slow data.
`Karabo device ID`	ID of the Karabo device which produces the slow data.
`Property name`	Property name in the Karabo device.
`Bin range`	Lower and upper boundaries of all the bins. In case of +/- *Inf*, the boundary will be calculated dynamically.
`# of bins`	Number of bins.
`Auto level`	Reset the levels of heatmaps in the *Binning* window.

Warning: Explicitly setup the binning range whenever possible. Normally binning only takes less than 1 ms with a predefined bin range. However, the cost could increase to a few tens of ms with a dynamic bin range!

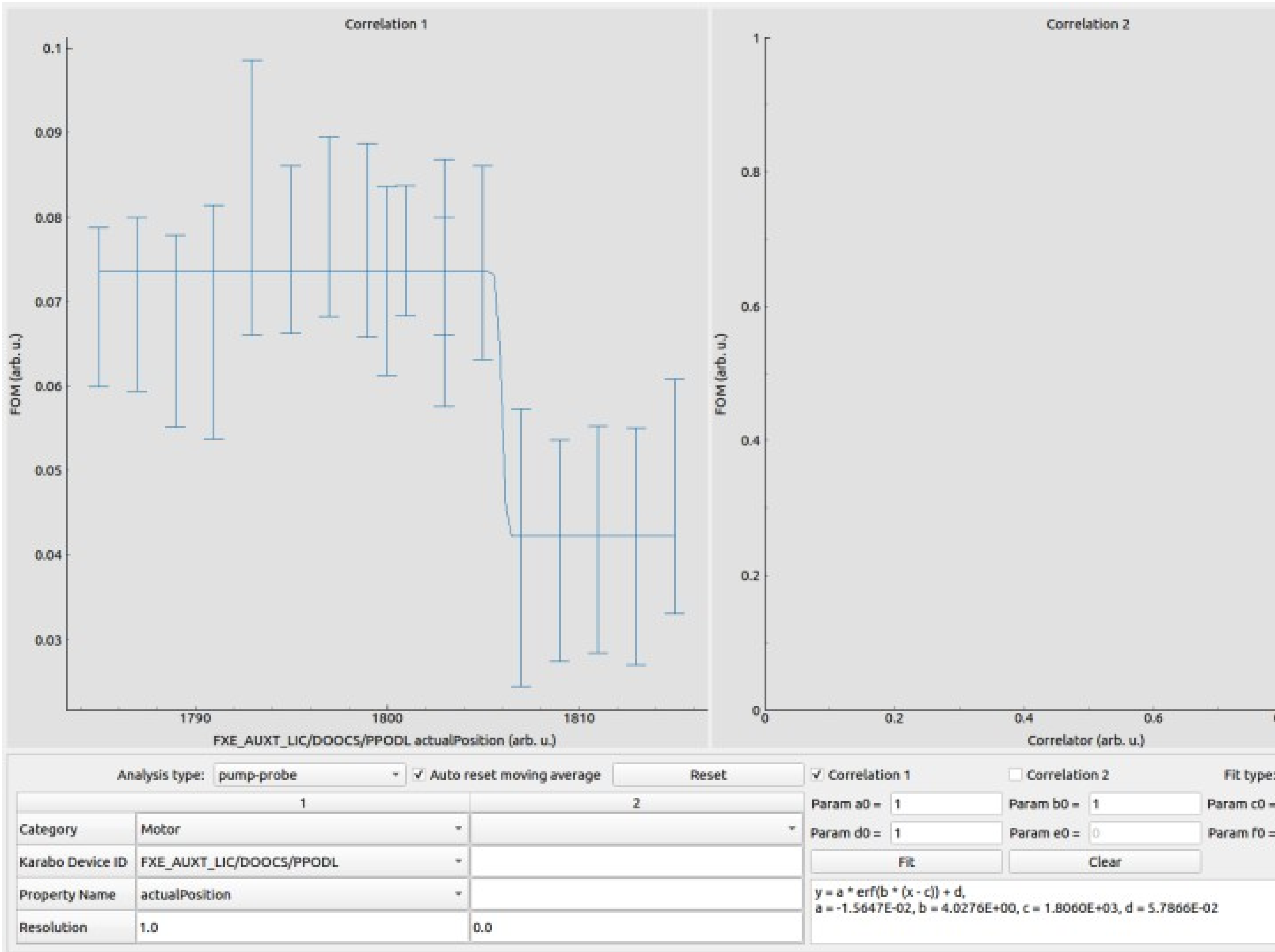
9.2 Histogram



Setup the visualization of pulse- / train- resolved statistics analysis.

Input	Description
`Analysis type`	See Analysis type.
`pulse resolved`	This checkbox is only enabled for the pulse-resolved detectors. When it is checked, the histogram is pulse-wise. Otherwise, the histogram is train-wise.
`Bin range`	Lower and upper boundaries of all the bins. In case of *+/- Inf*, the boundary will be calculated dynamically.
`# of bins`	Number of bins of the histogram.
`Reset`	Reset the histogram history.

9.3 Correlation



The correlation window allows to setup the visualization of correlations of a given FOM with various control data. Two plot types are supported in the correlation analysis:

- Scatter plot

  Scatter plot is the most commonly used plot type and it can be activated in the correlation window by setting `Resolution` to 0 (default).

- Statistics bar plot

  Statistics bar plot is very useful in the so-called "stop-and-scan" analysis: *a motor moves along a predefined path and it stays at each location for a certain period of time to collect enough data*. It is activated when `Resolution` is larger than 0. The y values of the upper and lower bars in the plot are $\bar{y} + \sigma$ and $\bar{y} - \sigma$, respectively, where $\bar{y}$ the mean and σ the standard deviation of the FOM values at that point. The widths of both bars are equal to `Resolution`. Specifically, assuming the current statistics point recorded an average x value of $\bar{x}$ and a new data (x, y) arrives, a new statistics point will be started if $|\bar{x} - x| > r$, where *r* is `Resolution`. Otherwise, the statistics of the current point will be updated. It should be noted that *if there is only 1 data at a statistics point, the point will be discarded.*

The statistics bar plot is very similar to 1D binning. However, different from 1D binning, a new statistics point will be recorded if the motor moves away and later comes back to the same location.

Note: One can change the value of resolution on-the-fly without resetting the whole data history.

Input	Description
`Analysis type`	See Analysis type.
`Category`	Category of the slow data.
`Karabo device ID`	ID of the Karabo device which produces the slow data.
`Property name`	Property name in the Karabo device.
`Resolution`	0 for scatter plot and any positive value for statistics bar plot.
`Reset`	Reset the correlation history.
`Auto reset moving average`	Check to automatically reset moving average in the "stop-and-scan" analysis when a new point starts. *Only apply to correlation 1.*

One can also plot FOMs of ROI1 and ROI2 together when the *master-slave* mode is activated in *ROI FOM setup*.

CHAPTER 10

Special analysis suites

10.1 Introduction

Special analysis suite is a new concept introduced in the version 0.8.2 to solve some issues which cannot be addressed in EXtra-foam core. Each suite is a group of applications (we will call it *app* in the rest part of this documentation) dedicated for specific online analysis. The following table compares EXtra-foam core and special suite in various aspects.

EXtra-foam core	EXtra-foam special suite
General purpose; rich features.	Special-purposed small apps.
Heavy computation.	Light computation.
Image detector oriented.	Any detector.
One instance per detector per cluster.	**No restriction on number of instances.**
GUI performance degrades with a lot of plots.	**Limited plots per app. GUI update rate can reach 10 Hz in most cases.**
Complicated.	Easy to implement.
Multi-process; distributed.	Single-process; multi-threading.
Python and C++.	Pure Python.
	Depends on EXtra-foam core; may need a core instance to run.

There are two different work flows for special analysis:

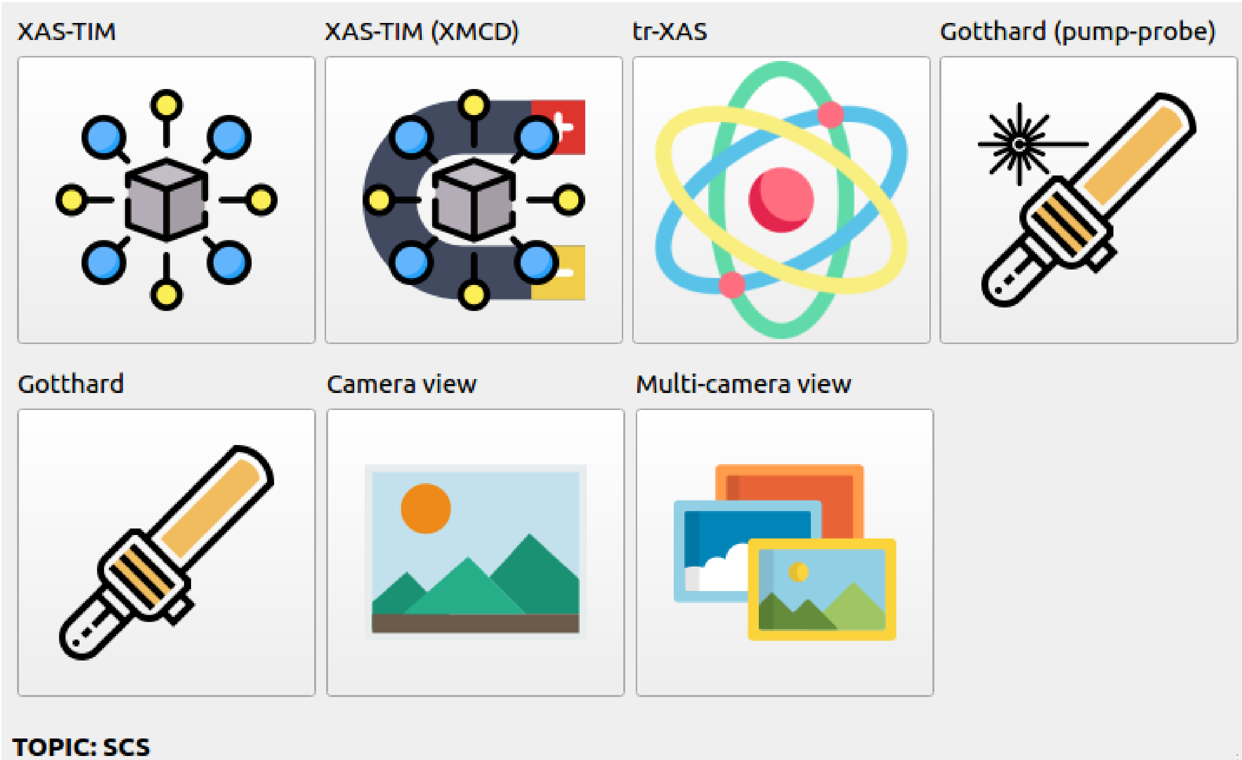
1. The special analysis receives processed data from a main **EXtra-foam** instance;
2. The special analysis is independent from the main instance and it directly receives data from a “Karabo bridge”.

The first work flow can be deemed as an extension to **EXtra-foam** core, while the second one is indeed similar to a mini **karaboFAI** prior to version 0.4.7.

The special analysis suite can be started by typing

```
extra-foam-special-suite TOPIC
```

in a terminal. Depending on the topic, you will see a facade window like



Click one of the app icons and the analysis window will show up and the facade will be closed. If one needs two instances of the same app, one can simply repeat the process. Apps in special suite can be further categorized into special-purposed (e.g. XAS-TIM, tr-XAS) and general purposed (e.g. Camera view, multi-camera view) apps. General purposed apps will appear in any topic while special-purposed apps will only appear in certain topics. Moreover, the same special-purposed app may behave differently in different topics. Namely, they are really customized for topics.

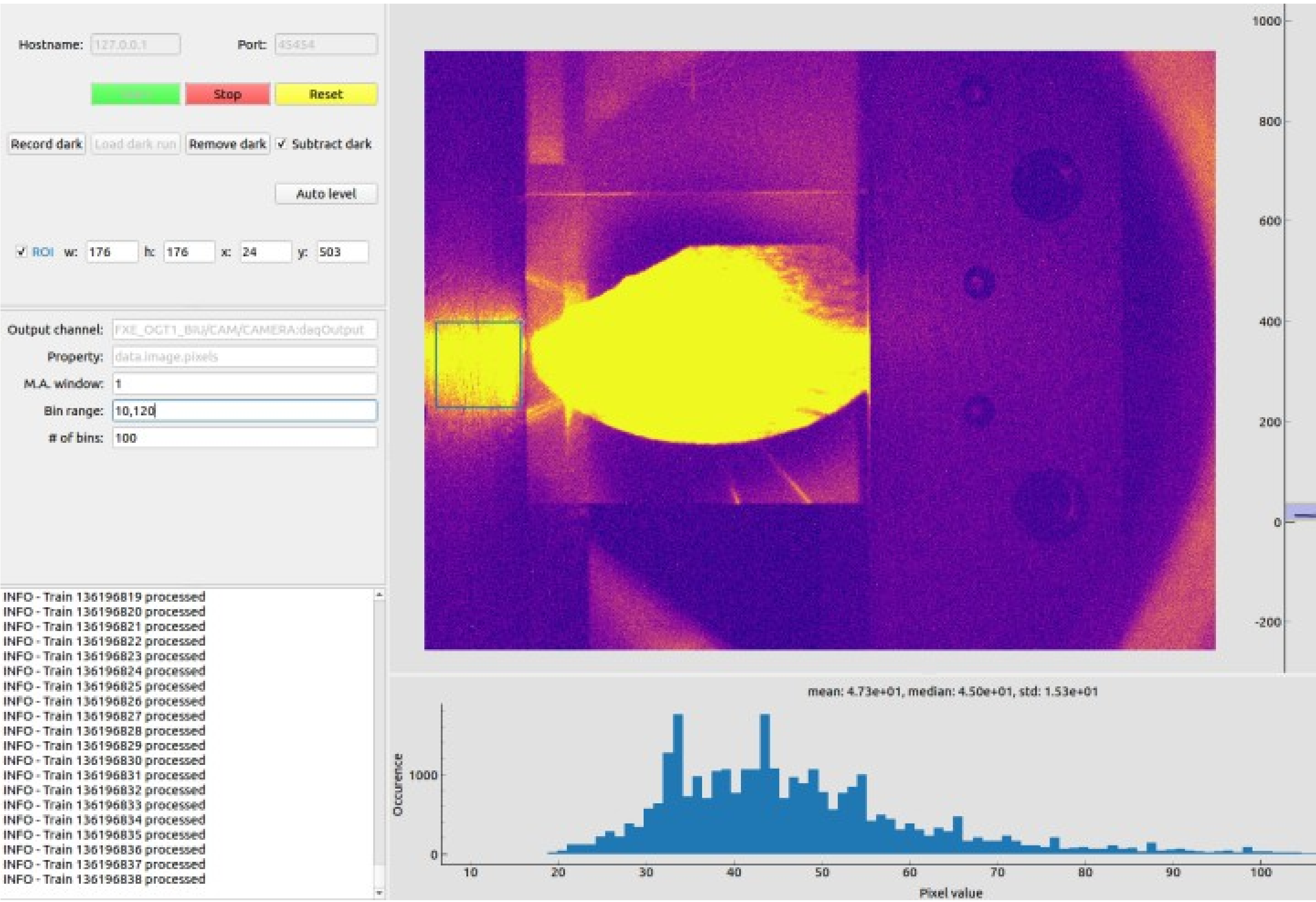
More info on command line arguments can be obtained as

```
[user@exflonc12 ~]$ extra-foam-special-suite --help

 usage: extra-foam-special-suite [-h] [-V] [--use-gate] [--debug]
                                 {SPB,FXE,SCS,SQS,MID,HED,DET}

 positional arguments:
   {SPB,FXE,SCS,SQS,MID,HED,DET}
                         Name of the topic

 optional arguments:
   -h, --help            show this help message and exit
   -V, --version         show program's version number and exit
   --debug               Run in debug mode
```

10.2 General purposed apps

Camera view, *Vector view*, multi-camera view

10.2.1 Camera view

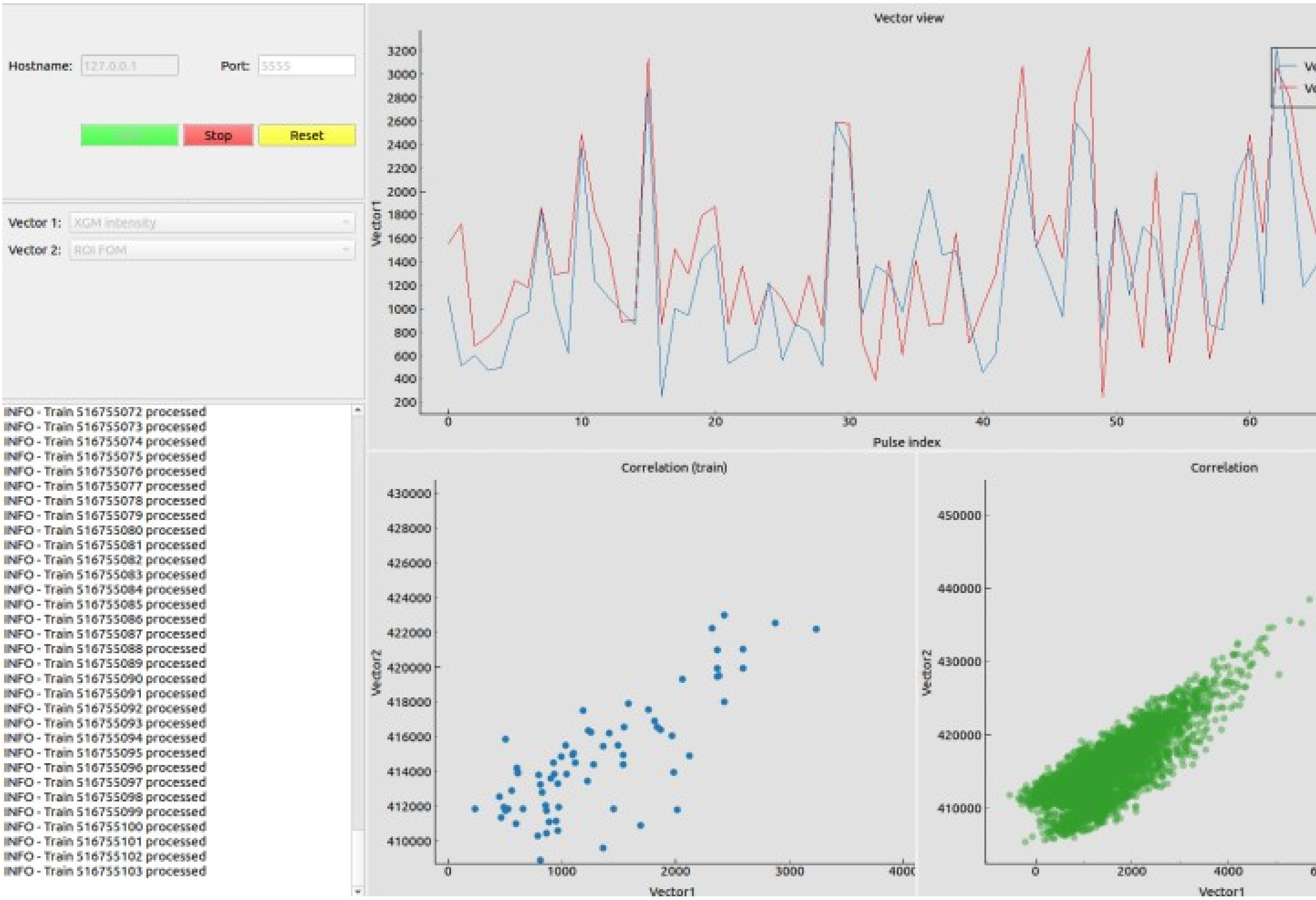


Camera view is a light-weight image tool for monitoring image data updated at 10 Hz or even faster. It also allows to monitor (ROI) histogram of the image. When the ROI is activated, histogram is calculated over the ROI instead of the whole image. It supports any train-resolved image data, i.e. an image data which has a shape of (y, x) or (1, y, x) or (y, x, 1).

More small features will be added in the future.

Input	Description
`Output channel`	Output channel name.
`Property`	Image data property name.
`M.A. window`	Slicer used to slice pulses in a train.
`Bin range`	Histogram bin range.
`# of bins`	Number of histogram bins.

10.2.2 Vector view



Vector view helps you monitor 1D data like XGM intensity, digitizer pulse integral, pulse-resolved ROI FOM and the correlation between them. **Vector view receives processed data from a main EXtra-foam instance.**

Input	Description
`Vector1`	Vector 1.
`Vector2`	Vector 2 (optional).

Note: In order to correlate two vectors, you will need to use the *Pulse slicer* in the data source tree located in the main GUI to select the data.

Warning: To have the pulse-resolved ROI FOM data, currently you must open the *Histogram* window in the main GUI and select *ROI FOM* as the analysis type. This will activate the pulse-resolved ROI FOM calculation, which is a little bit expensive. Also, make sure the *Pulse resolved* checkbox is checked.

10.3 Special purposed apps

- SPB

Gotthard

- FXE

Bragg diffraction peak, XES

- SCS

Gotthard, *Gotthard (pump-probe)*, *tr-XAS*, *XAS-TIM*, *XAS-TIM-XMCD*

- MID

Gotthard, *Gotthard (pump-probe)*

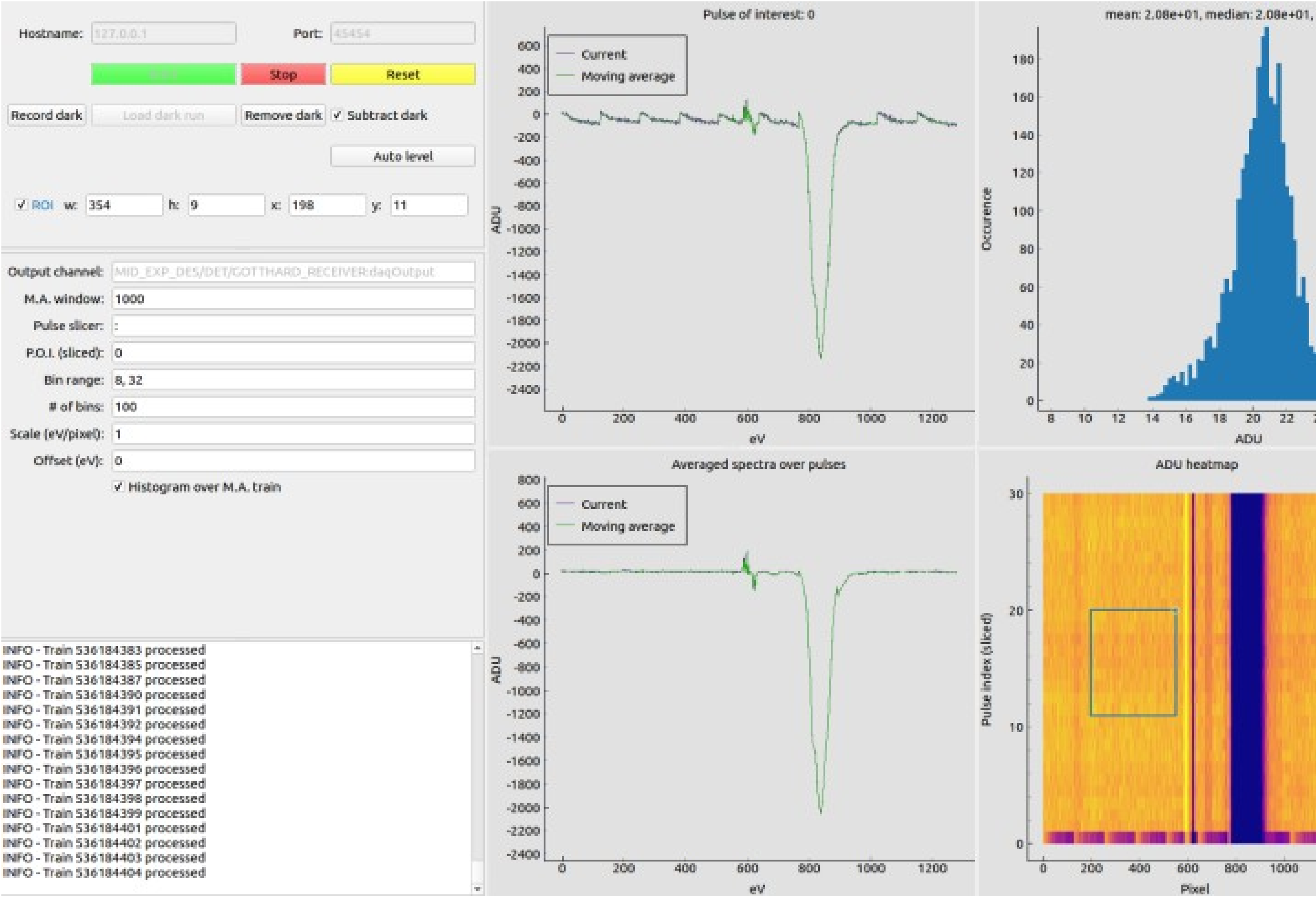
- HED
- DET

Multi-module scan

- XPD

Gotthard

10.3.1 Gotthard

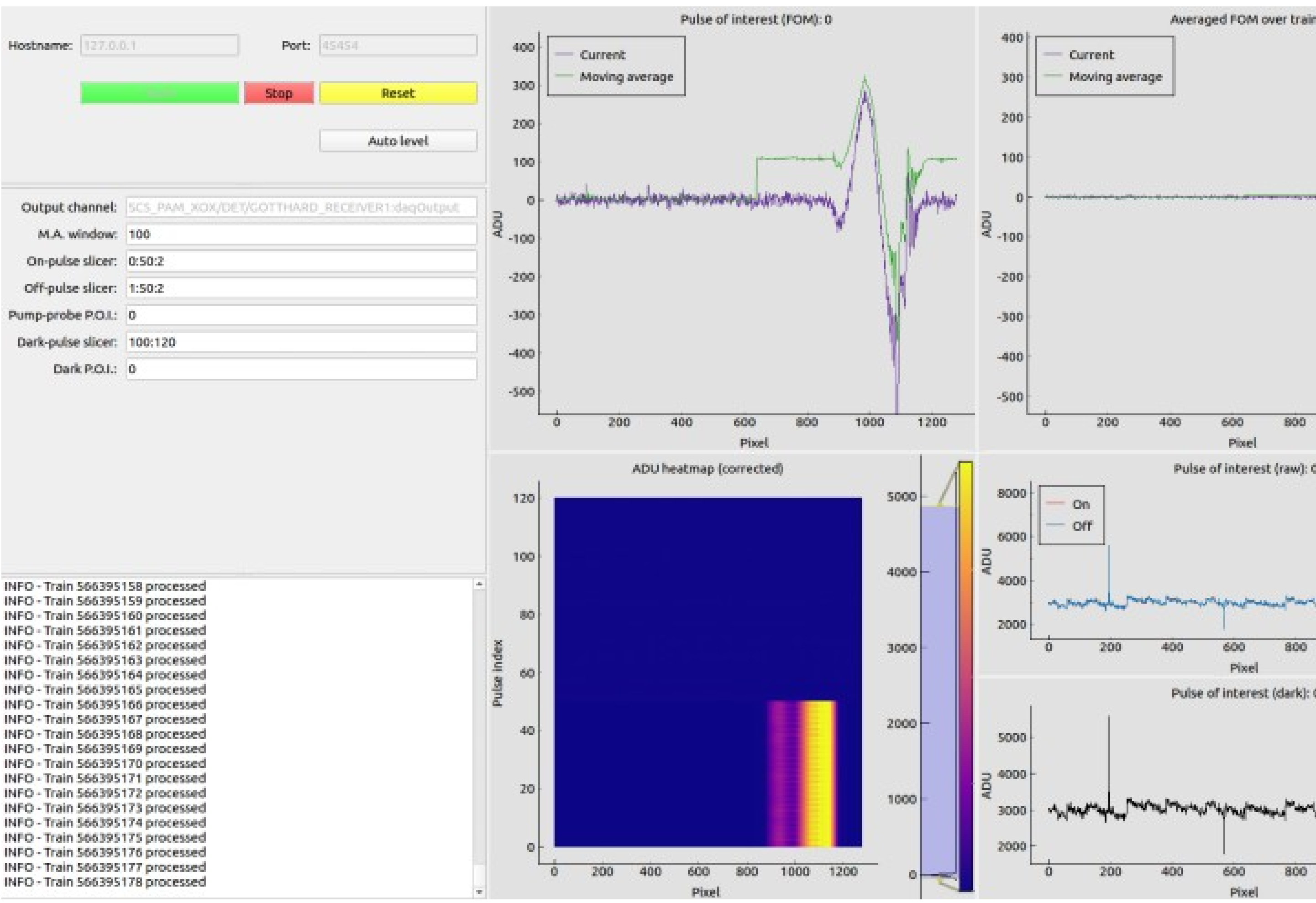


When the ROI is activated, histogram is calculated over the ROI instead of the whole image.

For applying dark subtraction, one can record the dark online or load a dark run from the file.

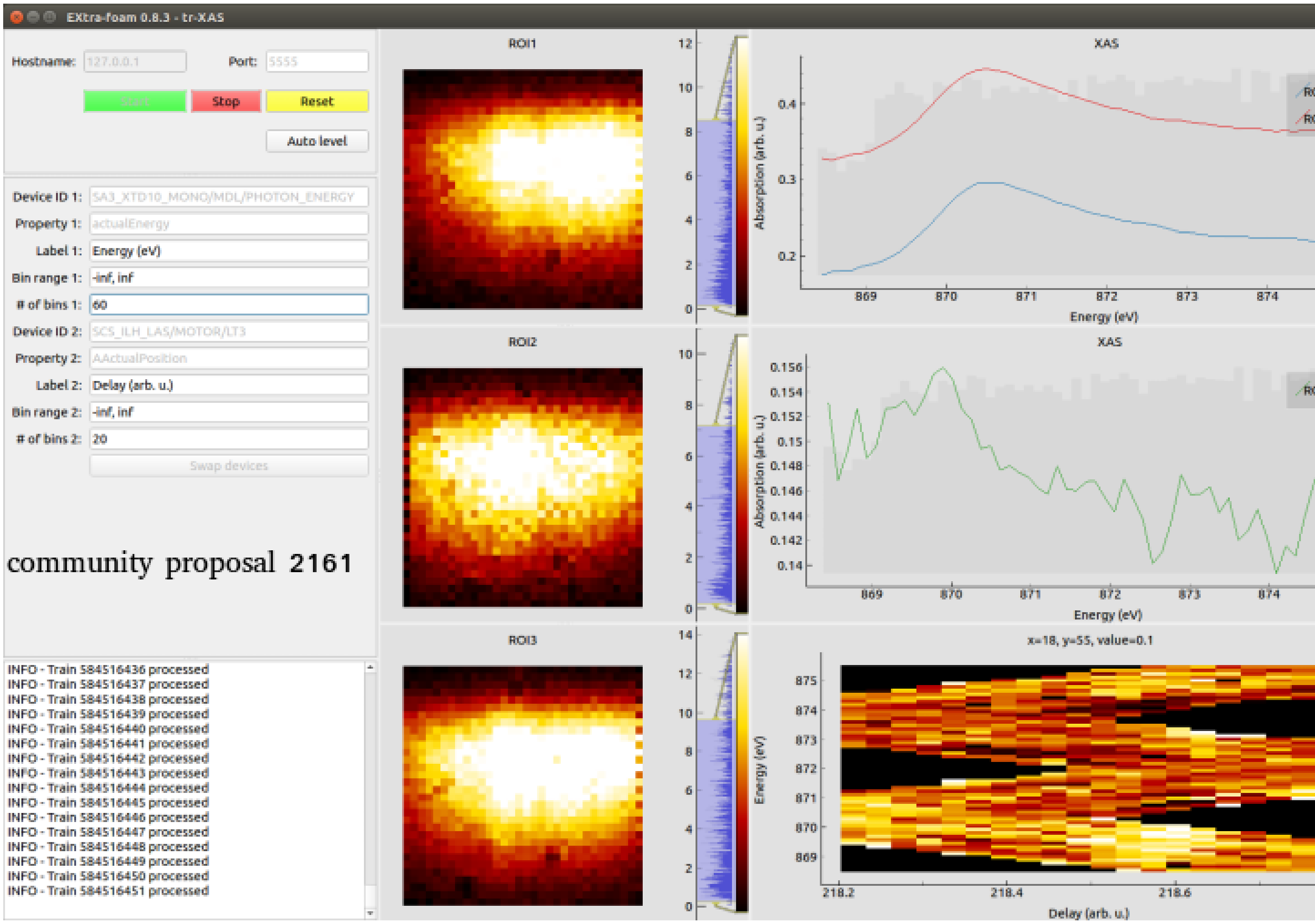
Input	Description
`Output channel`	Output channel name.
`M.A. window`	Moving average window size.
`Pulse slicer`	Slicer used to slice pulses in a train.
`P.O.I. (sliced)`	Index of the pulse of interest after pulse slicing. For example, if there are 60 pulses in a train and the slicer only selects a half of them. The index range will be from 0 to 29.
`Bin range`	Histogram bin range.
`# of bins`	Number of histogram bins.
`Scale (eV/pixel)`	Scale used for axis calibration. If it is zero, no calibration will be performed and the `Offset` will be ignored as well.
`Offset (eV)`	Offset used for x-axis calibration.
`Histogram over M.A.`	Check to calculate the histogram over the moving averaged data.

10.3.2 Gotthard (pump-probe)

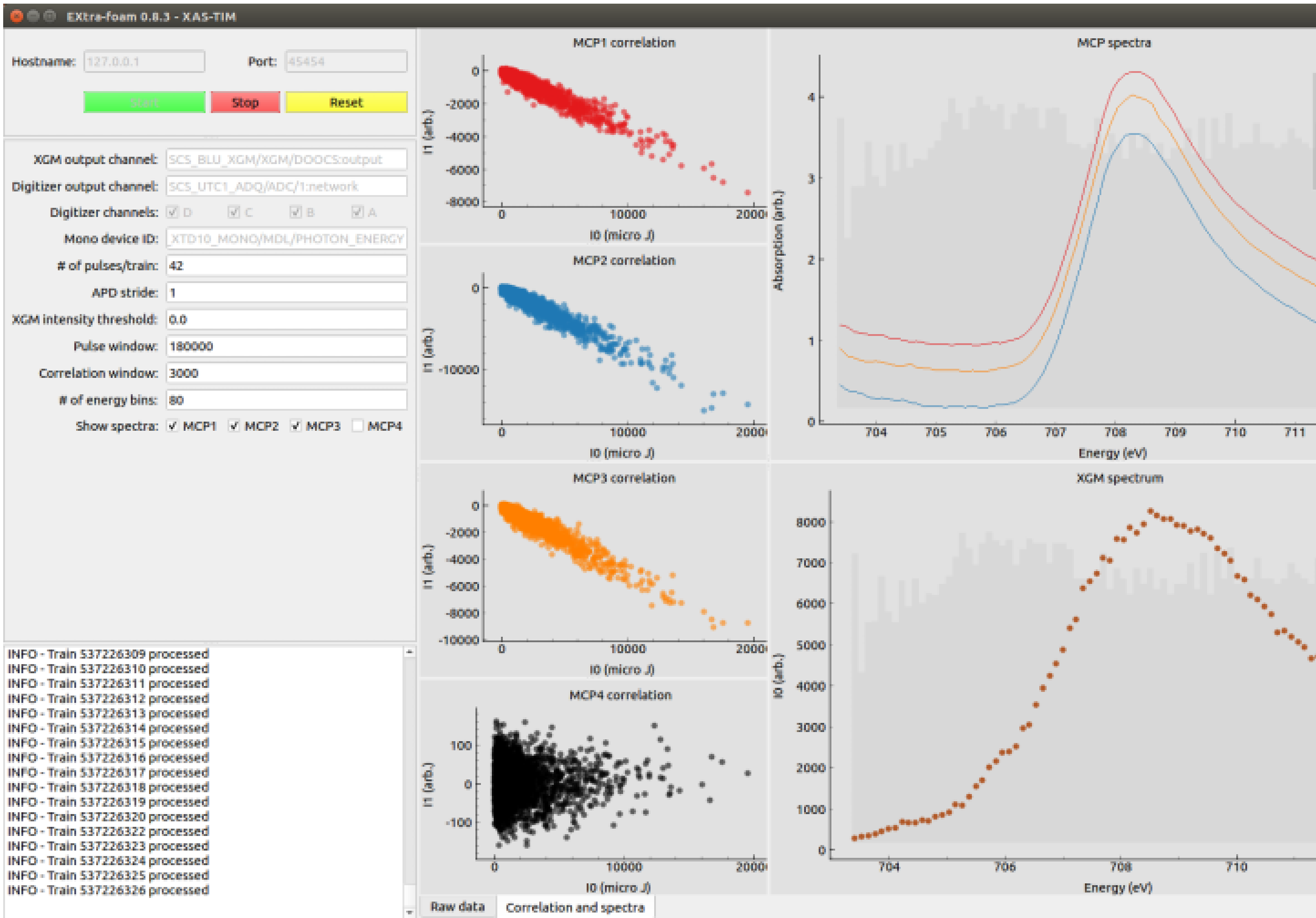


Input	Description
`Output channel`	Output channel name.
`M.A. window`	Moving average window size.
`On-pulse slicer`	Slicer used to slice on (pumped) pulses in a train.
`Off-pulse slicer`	Slicer used to slice off (unpumped) pulses in a train. *The numbers of the on and off pulses must be the same.*
`Pump-probe P.O.I.`	Index of the pump-probe pulse of interest. For example, if there are 10 on-pulses and 10 off-pulses, the range of the index is from 0 to 9.
`Dark-pulse slicer`	Slicer used to slice dark pulses in a train.
`Dark-pulse P.O.I.`	Index of the dark pulse of interest. For example, if there are 5 dark pulses and the total number pulses is 100, the index range is from 0 to 4 regardless of the dark pulse positions in the train.

10.3.3 tr-XAS

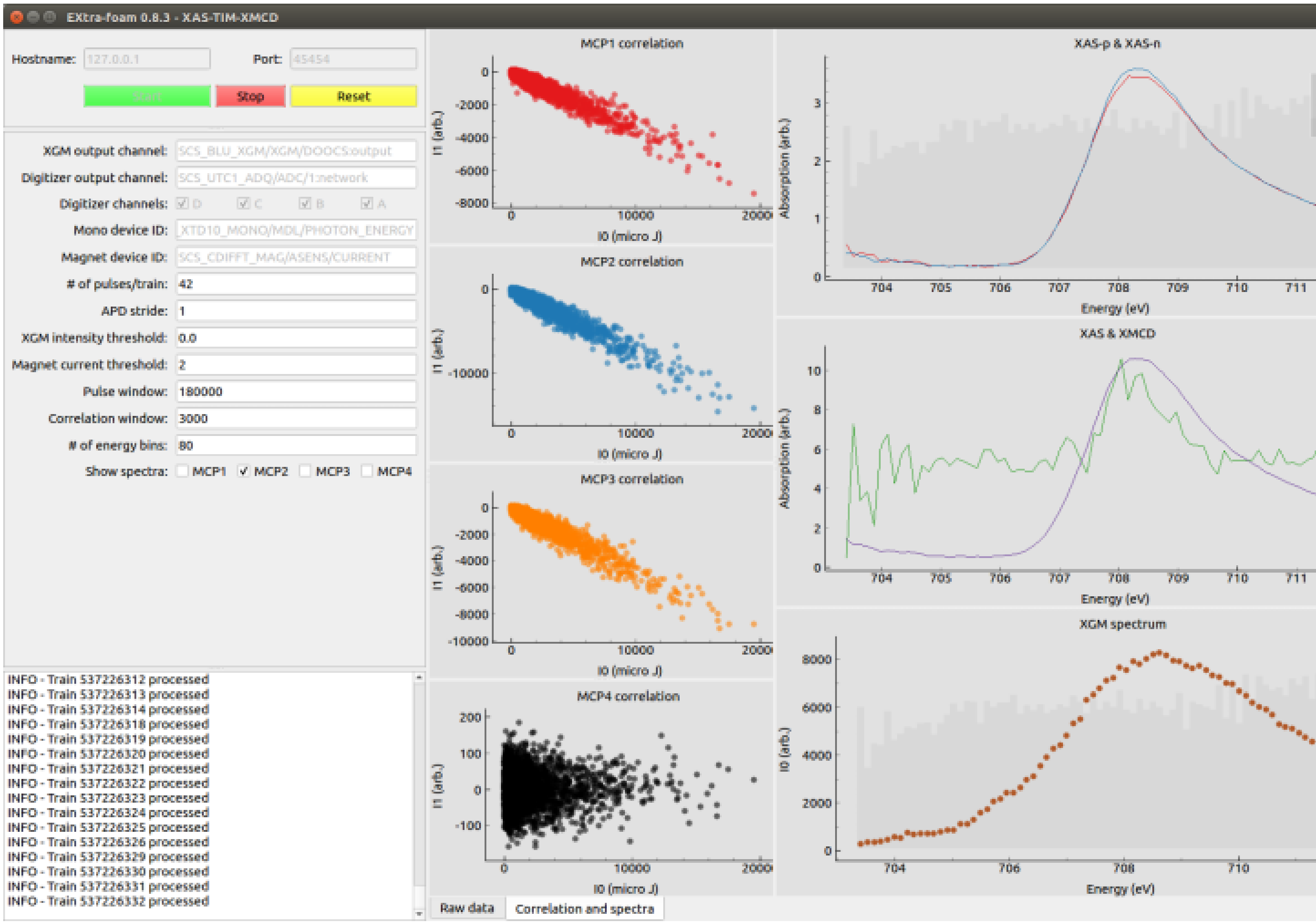


10.3.4 XAS-TIM



Input	Description
`XGM Output channel`	XGM output channel name.
`Digitizer output channel`	Digitizer output channel name.
`Digitizer channels`	Check to analyze data on the corresponding channel.
`Mono device ID`	Monochromator device ID.
`# of pulses/train`	Number of 'useful' XGM pulses in a train.
`APD stride`	Stride used in slice the digitizer APD data.
`XGM intensity threshold`	Pulses with XGM intensity below this threshold will be ignored.
`Pulse window`	Maximum number of pulses used in calculating the spectra.
`# of energy bins`	Number of energy bins for spectra calculation.
`Pulse window`	Maximum number of pulses used in I0/I1 correlation visualization.
`Show spectra`	Check to show spectra on the corresponding digitizer channel.

10.3.5 XAS-TIM-XMCD

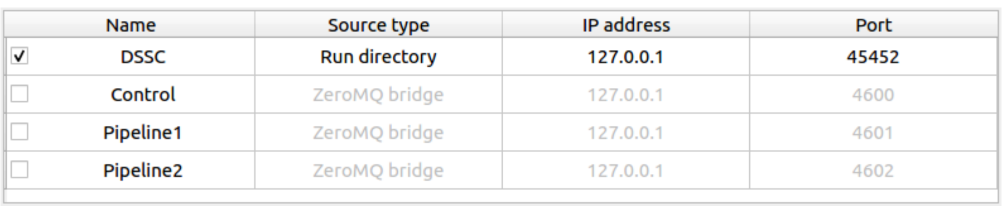


XAS-TIM-XMCD is very similar to *XAS-TIM*. For input not listed in the following table, please refer to the input table in *XAS-TIM*.

Input	Description
`Magnet device ID`	Output channel name of the device which provides the magnet current values.
`Magnet current threshold`	Pulses with an absolute magnet current value below this threshold will be ignored.

CHAPTER 11

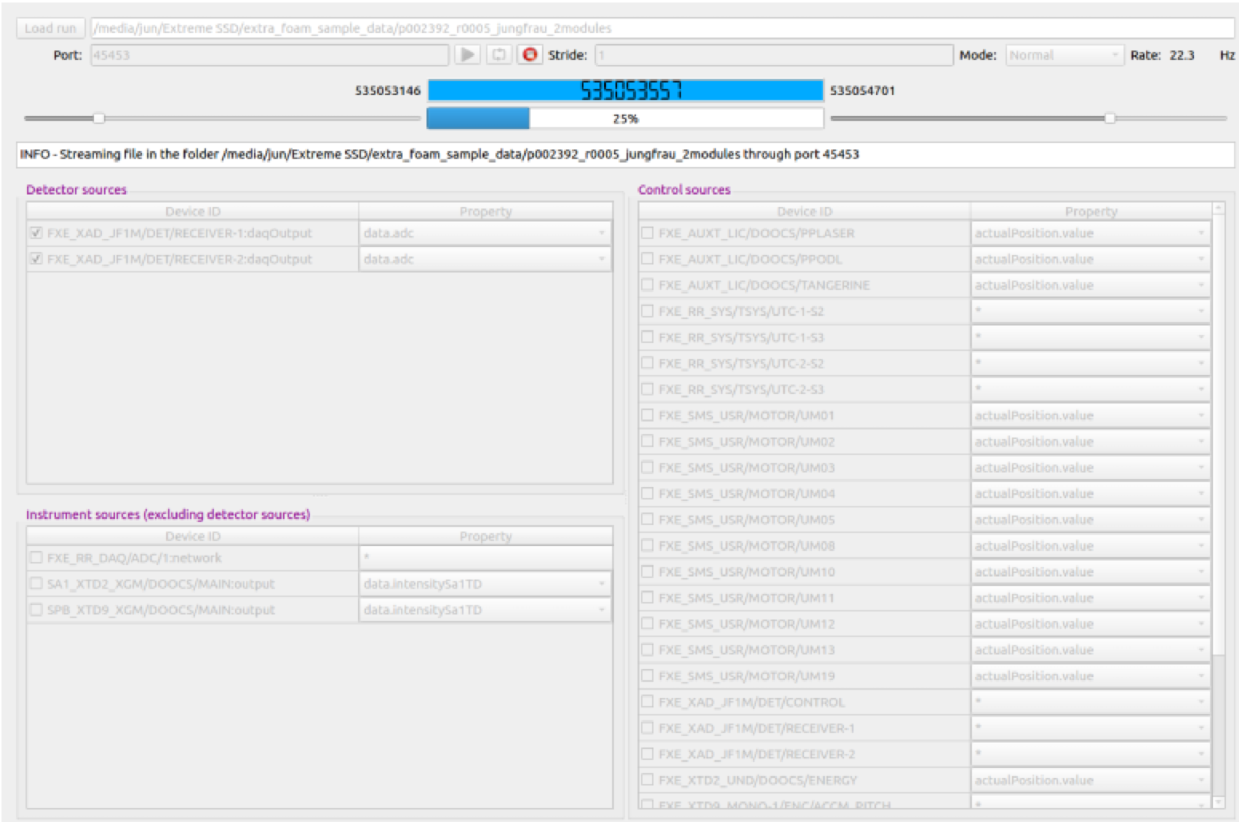
Stream data from files



	Name	Source type	IP address	Port
✓	DSSC	Run directory	127.0.0.1	45452
	Control	ZeroMQ bridge	127.0.0.1	4600
	Pipeline1	ZeroMQ bridge	127.0.0.1	4601
	Pipeline2	ZeroMQ bridge	127.0.0.1	4602

11.1 File streamer

EXtra-foam can be used to replay experiments with stored files. To start, click on the *File stream* icon on the tool bar to opens the following window:



Alternatively, one can type

```
extra-foam-stream
```

in another terminal to open the above window. **This is the recommended way since the streamer and the EXtra-foam instance are decoupled**. This is also useful for development since one does not have to set up the streamer again when restarting **EXtra-foam**.

Input	Description
`Load run`	Click to select a run folder. The run folder can also be specified via entering the full path.
`Port`	The TCP port from which the data is streamed. If the GUI is not opened from the terminal, the `Port` is displayed as '*' and internally it is the same as the port specified in the *Data source* panel in the main GUI.
`Stream once`	Press to stream the data in the run folder once.
`Stream repeatedly`	Press to stream the data in the run folder repeatedly. If the stream reaches the end of the data, the stream will restart from the beginning with a faked *train ID*, which ensures that the *train ID* continuously increases in the new cycle. This feature is only useful for developers.
`Stop stream`	Press to stop streaming.
`Mode`	Stream mode: • `Normal`: Sources in a train are streamed together. • `Random shuffle`: Sources in a train are streamed one by one and the order is random.
`First train ID`	Slide to change the first train ID to stream.
`Last train ID`	Slide to change the last train ID to stream.
`Stride`	Train ID stride used to slice trains to stream.

Note: If the specified run folder has a path structure as on *Maxwell GPFS* (*.../proc/runnumber/*). The loader will try to load the data (e.g. control data) other than the detector data from *.../raw/runnumber/* first and fall back to *.../proc/runnumber/* if *.../raw/runnumber/* is not a valid run directory. This is needed for large multi-module detectors, as the folder *.../proc/runnumber/* only stores the calibrated detector data.

11.2 Sample run directories

Note: Streaming files from the online cluster is pretty fast. However, it is sometimes unbearable to stream a large run from the *Maxwell* cluster. For development, it is recommended to copy a few files in a run to a local directory.

	Run directory	Description
AGIPD	/gpfs/exfel/exp/SPB/201831/p900039/proc/r0273	ring; 176 pulses
LPD	/gpfs/exfel/exp/FXE/201802/p002218/raw/r0229	ring, 100 pulses
	/gpfs/exfel/exp/FXE/201802/p002218/proc/r0229	ring, 100 pulses
DSSC	/gpfs/exfel/exp/SCS/201901/p002212/raw/r0061	pump-probe; 70 pulses
	/gpfs/exfel/exp/SCS/201901/p002212/raw/r0059	pump-probe (dark of r0061)
	/gpfs/exfel/exp/SCS/201901/p002161/raw/r0093	tr-XAS, single module, 50 pulses
	/gpfs/exfel/exp/SCS/201901/p002161/raw/r0095	tr-XAS (dark of r0093)
JungFrau	/gpfs/exfel/exp/FXE/201930/p900063/proc/r1051	pump-probe
	/gpfs/exfel/exp/FXE/201930/p900063/raw/r1051	pump-probe
	/gpfs/exfel/exp/SPB/201922/p002566/proc/r0061	Burst mode; ring; 6 modules
FastCCD	/gpfs/exfel/exp/SCS/201802/p002170/proc/r0141	
	/gpfs/exfel/exp/SCS/201802/p002170/raw/r0141	
Gotthard	/gpfs/exfel/exp/MID/201931/p900090/raw/r0395	Test data
	/gpfs/exfel/exp/MID/201931/p900090/raw/r0300	Test data (dark)
	/gpfs/exfel/exp/SCS/201931/p900094/raw/r0647	pump-probe
XAS-TIM	/gpfs/exfel/exp/SCS/201931/p900094/raw/r0491	XMCD, 42 pulses/train, APD stride = 1

CHAPTER 12

Streaming from Karabo

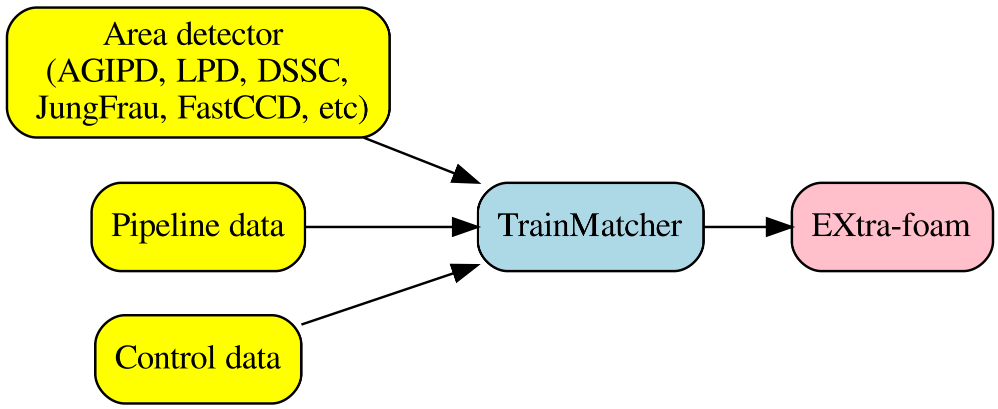
At the European XFEL, **EXtra-foam** receives data from the distributed control framework Karabo.

Data received from different sensors/detectors located at different locations need to be "aligned" for further analysis. At European XFEL, the accelerator produces 10 bunch trains per second. As a result, data are stamped and can only be aligned by **Train ID**. For big modular detectors like AGIPD, LPD, DSSC, etc., alignment of data from different modules are carried out in the so-called "calibration pipeline". However, in most use cases, users will also want to align the detector data with motor positions and/or some other detectors (e.g. XGM, digitizer).

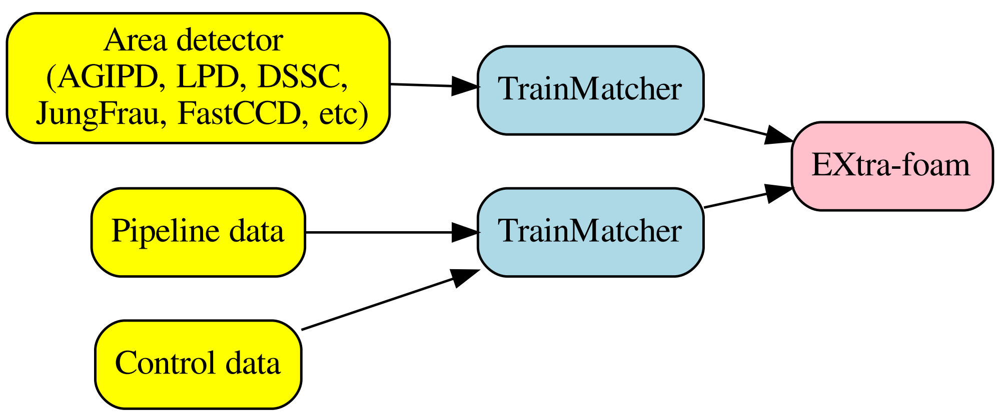
Fortunately, the aforementioned alignment is taken care of by *TrainMatcher* Karabo devices or EXtra-foam, users do not need to worry about that. After the data from different sources are aligned in a TrainMatcher, they will be serialized and streamed over ZeroMQ using the Karabo bridge protocol. EXtra-foam makes use of karabo-bridge-py, which is a Karabo bridge client implementation, to receive and deserialize the data.

There are two scenarios for operating EXtra-foam with TrainMatcher devices:

1. The most common scenario is where all necessary data is aligned by a single TrainMatcher, and then streamed to EXtra-foam.



2. One can also send different data items via different TrainMatchers and EXtra-foam will align everything automatically after receiving the data. If not all the requested data in a given train can be found, the train will be discarded.

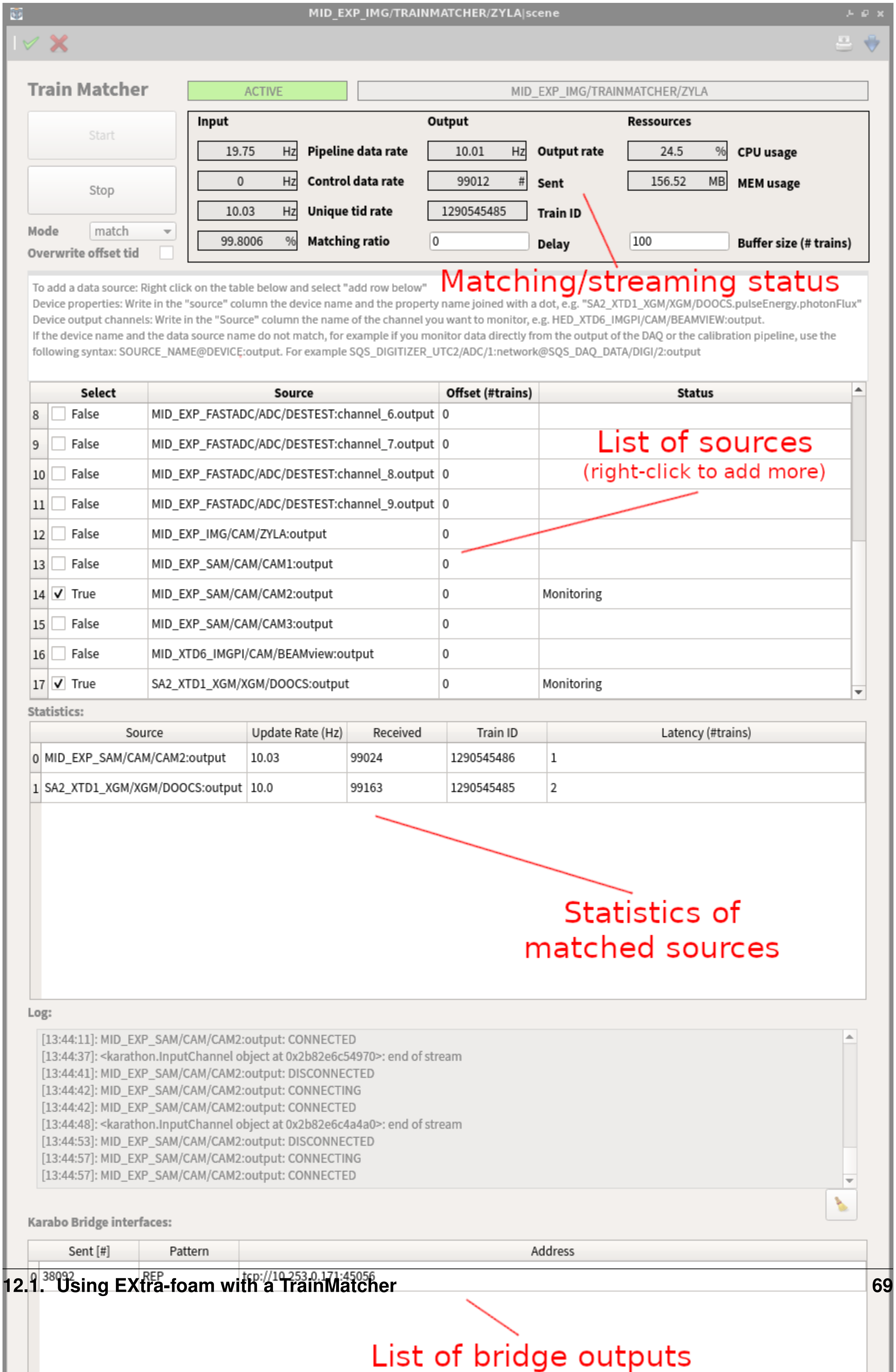


This should rarely be necessary. In most situations having a single TrainMatcher as in the first scenario will suffice.

12.1 Using EXtra-foam with a TrainMatcher

As mentioned in the previous section, a TrainMatcher is the Karabo device that matches data from different data sources by train ID. Your instrument should already have a TrainMatcher set up to use with EXtra-foam (if not, ask your local DA contact).

This is a screenshot of the TrainMatcher interface, with the particularly important parts for operation highlighted:


MID_EXP_IMG/TRAINMATCHER/ZYLA|scene
Train Matcher
ACTIVE
MID_EXP_IMG/TRAINMATCHER/ZYLA
Start
Stop
Mode
match
Overwrite offset tid
Input
19.75 Hz Pipeline data rate
0 Hz Control data rate
10.03 Hz Unique tid rate
99.8006 % Matching ratio
Output
10.01 Hz Output rate
99012 # Sent
1290545485 Train ID
0 Delay
Ressources
24.5 % CPU usage
156.52 MB MEM usage
100 Buffer size (# trains)
Matching/streaming status
To add a data source: Right click on the table below and select "add row below"
Device properties: Write in the "source" column the device name and the property name joined with a dot, e.g. "SA2_XTD1_XGM/XGM/DOOCS.pulseEnergy.photonFlux"
Device output channels: Write in the "Source" column the name of the channel you want to monitor, e.g. HED_XTD6_IMGPI/CAM/BEAMVIEW:output.
If the device name and the data source name do not match, for example if you monitor data directly from the output of the DAQ or the calibration pipeline, use the following syntax: SOURCE_NAME@DEVICE:output. For example SQS_DIGITIZER_UTC2/ADC/1:network@SQS_DAQ_DATA/DIGI/2:output
Select | Source | Offset (#trains) | Status
8 False MID_EXP_FASTADC/ADC/DESTEST:channel_6.output 0
9 False MID_EXP_FASTADC/ADC/DESTEST:channel_7.output 0
10 False MID_EXP_FASTADC/ADC/DESTEST:channel_8.output 0
11 False MID_EXP_FASTADC/ADC/DESTEST:channel_9.output 0
12 False MID_EXP_IMG/CAM/ZYLA:output 0
13 False MID_EXP_SAM/CAM/CAM1:output 0
14 True MID_EXP_SAM/CAM/CAM2:output 0 Monitoring
15 False MID_EXP_SAM/CAM/CAM3:output 0
16 False MID_XTD6_IMGPI/CAM/BEAMview:output 0
17 True SA2_XTD1_XGM/XGM/DOOCS:output 0 Monitoring
List of sources
(right-click to add more)
Statistics:
Source | Update Rate (Hz) | Received | Train ID | Latency (#trains)
0 MID_EXP_SAM/CAM/CAM2:output 10.03 99024 1290545486 1
1 SA2_XTD1_XGM/XGM/DOOCS:output 10.0 99163 1290545485 2
Statistics of
matched sources
Log:
[13:44:11]: MID_EXP_SAM/CAM/CAM2:output: CONNECTED
[13:44:37]: <karathon.InputChannel object at 0x2b82e6c54970>: end of stream
[13:44:41]: MID_EXP_SAM/CAM/CAM2:output: DISCONNECTED
[13:44:42]: MID_EXP_SAM/CAM/CAM2:output: CONNECTING
[13:44:42]: MID_EXP_SAM/CAM/CAM2:output: CONNECTED
[13:44:48]: <karathon.InputChannel object at 0x2b82e6c4a4a0>: end of stream
[13:44:53]: MID_EXP_SAM/CAM/CAM2:output: DISCONNECTED
[13:44:57]: MID_EXP_SAM/CAM/CAM2:output: CONNECTING
[13:44:57]: MID_EXP_SAM/CAM/CAM2:output: CONNECTED
Karabo Bridge interfaces:
Sent [#] | Pattern | Address
0 38092 REP tcp://10.253.0.171:45056
List of bridge outputs

In this case we have a TrainMatcher configured with lots of different sources (cameras, an XGM, digitizer channels, etc), but only two are being *monitored* (see the source list): `MID_EXP_SAM/CAM2:output` and `SA2_XTD1_XGM/XGM/DOOCS:output`. Both of these are pipeline sources and both are getting updated at 10Hz (see the statistics table), which means the TrainMatcher is matching and streaming data at 10Hz (see the matching/streaming status at the top). Here the train data is getting streamed over just one Karabo bridge interface (see the list of outputs at the bottom), but more could be added if necessary.

To configure a TrainMatcher and connect EXtra-foam to it:

1. Add the sources you want. This can be done by right-clicking the source list and adding new items, but sources added like this will not be saved across restarts. To permanently add a source, shutdown the device and modify the `Data Sources` property.
2. Select the sources you want by ticking the boxes for each source. When you're done, either hit the Enter key or click the green tick icon in the top left to apply the changes.
3. Click the `Start` button to start the TrainMatcher.

Note: The previous two steps (adding sources and enabling/disabling them) can be done while the TrainMatcher is running, there's no need to stop/start it while reconfiguring the sources.

4. Now we can connect EXtra-foam to the Karabo bridge of the TrainMatcher. In EXtra-foam's *Main GUI* check the bridge configuration:

Name	Source type	IP address	Port
☑ DSSC	ZeroMQ bridge	10.253.0.140	4511
☐ Control	ZeroMQ bridge	127.0.0.1	4600
☐ Pipeline1	ZeroMQ bridge	127.0.0.1	4601
☐ Pipeline2	ZeroMQ bridge	127.0.0.1	4602

The source type should be `ZeroMQ bridge`, and the IP address and port should match the one shown in the TrainMatcher (see the list of bridge outputs in the screenshot above).

If that matches, you should be able to click EXtra-foam's start button and begin processing data.

For common issues related to the TrainMatcher and EXtra-foam, see the *Operational issues* section.

CHAPTER 13

Streaming from saved runs

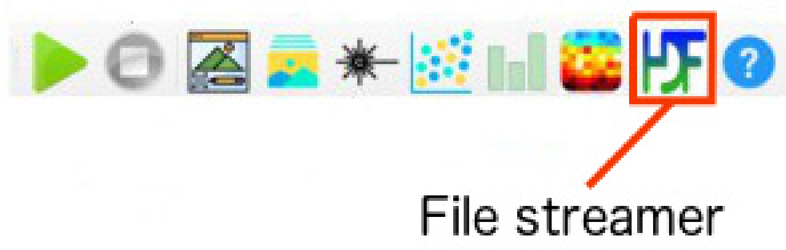
There is a stand-alone tool, the EXtra-foam file streamer, dedicated to provide a data stream from files of a run directory. This tool has its own GUI window.

13.1 Overview of user operation steps:

1. Open the file streamer (from the main GUI or command-line)
2. Select and load the run directory
3. Select the sources from the files to be streamed
4. In the main GUI, change the streaming source type (mode)
5. Select the appropriate source name the in the main GUI
6. Start sending the stream in the streaming window
7. Start processing of data in the main GUI

13.2 Opening the file streamer

The task bar on top of the main window consists of 10 icon buttons for various tasks, typically opening a separate window. The second button from the right, showing HDF (1) can be clicked to start the file streamer tool.

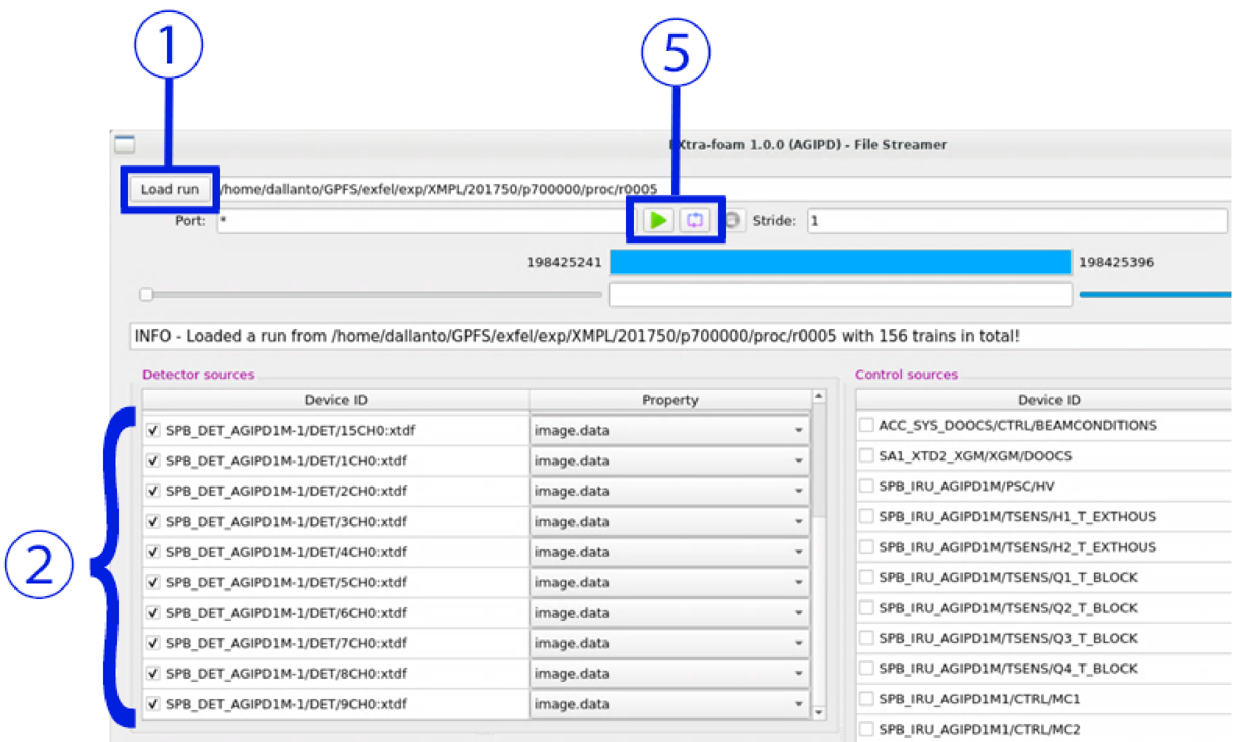


Alternatively, one can use the command line and type

```
extra-foam-stream
```

13.3 Selecting the run directory

In the open file streamer window, push the `Load run` button (1) in order to open a standard file browser dialog, and navigate to the target run directory.



Alternatively, a given run path can also be put into the text field by typing, respectively copy/paste.

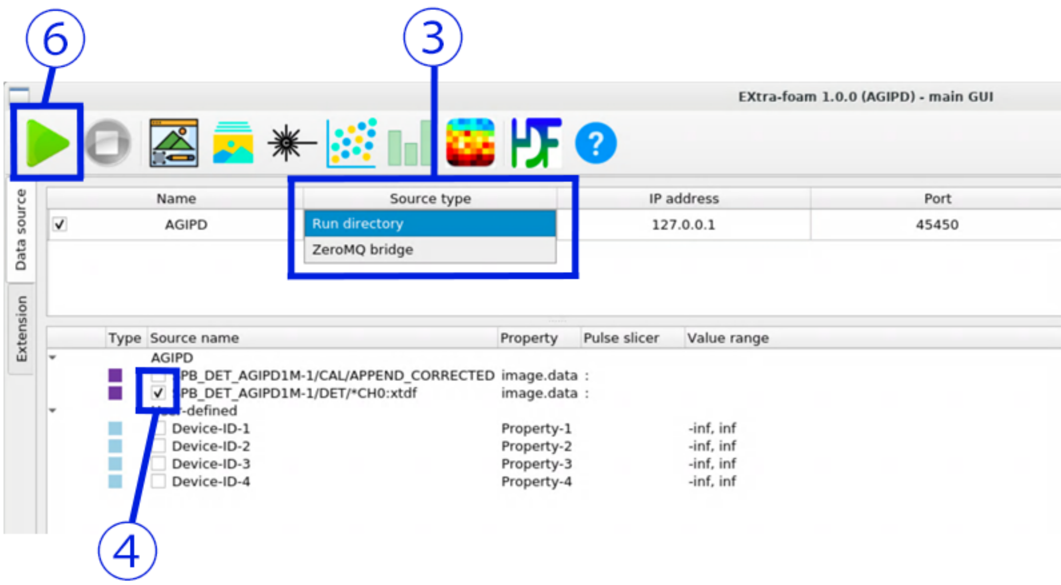
13.4 Selecting the sources from file

The figure depicts the source display after loading a run with AGIPD-1M detector data. Select all 16 detector modules by marking every check-button (2).

13.5 Changing the streaming source type and name

Back in the main window, double-click the drop-down button in the `Data source` panel, below the `Source type` label (3). It reads `ZeroMQ bridge` by default, select `Run directory` instead.

In the normal case, the `IP address` and `Port` will adapt automatically.



Next, mark the appropriate check-button in the `Source name` column (4) of the panel below. For big detectors, this DAQ name typically ends with `*CH0:xtdf` and corresponds to the source names as identified from the run folder, i. e. the selections in (2).

13.6 Starting the file-stream broadcast

Now that the settings are made, start the data stream from file by pressing one of the play buttons in the streamer (5), which are for train-wise single-pass through the data, as per green play-arrow, or continuous loop for "endless" playback, as per loop symbol. Train-IDs are steadily increased for subsequent loops.

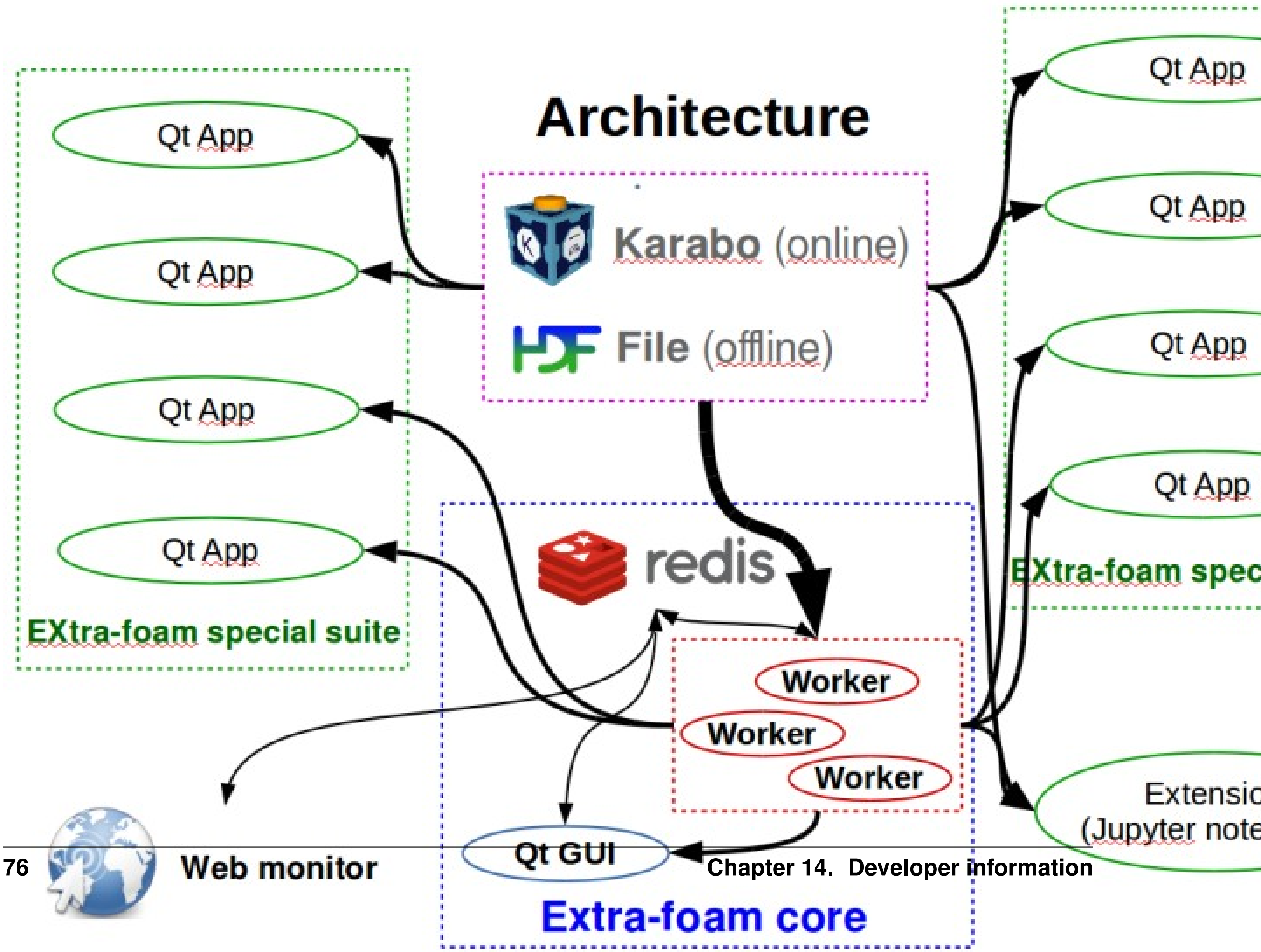
13.7 Starting the data processing

Start the actual EXtra-foam data processing/analysis by pressing the play button (6) in the main window, which will take the data train-by-train from the file stream. Select your type of analysis by choosing the corresponding task button from the task bar.

CHAPTER 14

Developer information

14.1 Design



14.2 Build and Test

Before running the Python unittest, rebuild the c++ code if it was updated. One can do either

```
$ pip install -e . -v
```

or

```
$ python setup.py build_ext --with-tests --inplace
```

Then, run the both C++ and Python unittests:

```
$ python setup.py test
```

To only run the Python unittest:

```
$ python -m pytest extra_foam -v -s
```

To build and run the c++ unittest only (we use GoogleTest):

```
$ mkdir build && cd build
$ cmake -DBUILD_FOAM_TESTS .. && make ftest
```

To run the Python benchmark:

```
$ python setup.py benchmark
```

14.3 Release EXtra-foam

- Update the **changeLog**;
- Update the version number in *extra_foam/__init__.py*;
- Merge the above change into the *dev* branch;
- Merge the *dev* branch into the *master* branch;
- Tag the *master* branch;
- Create a new branch from the new *master* branch and update the version number in *extra_foam/__init__.py*. For example, if the latest release version is "0.8.0", the new version number should be "0.8.1dev", supposing the next release is 0.8.1. Also, the name of the new branch should be "0.8.1dev";
- Merge the new branch into the *dev* branch;

14.4 Deployment on EuXFEL Anaconda Environment

EXtra-foam deployment on exfel anaconda environments should be done using **xsoft** account. Use the following anaconda environments to deploy particular versions of **EXtra-foam**

Version	Deployment environment
Latest	EXtra-foam/beta
Stable	EXtra-foam
Test	EXtra-foam/alpha

```
$ ssh xsoft@max-display.desy.de
$ cd workspace
$ git clone --recursive --branch <tag_name> https://github.com/European-XFEL/EXtra-
↪foam.git EXtra-foam-<tag_name>
$ cd EXtra-foam-<tag_name>
$ module load exfel <environment_name>
$ which pip
/gpfs/exfel/sw/software/<environment_name>/bin/pip
$ pip install . -v
```

Note: ssh to the Maxwell and online cluster with your own account, respectively, and launch **EXtra-foam** there to double check the deployed version.

CHAPTER 15

Changelog

15.1 1.0.0 (31 July 2020)

- **Improvement**
  - Improve FFT of image in feature extraction. PR #268
  - Improve fitting interface. PR #265
  - Add reset panel in the main GUI. PR #264
  - Update default config files. PR #262
  - Improve azimuthal integration performance. PR #261, PR #268
  - Revert y-axis orientation change. PR #260
  - Add Python binding for double dtype in generalized geometry. PR #259
- **New Feature**
  - Implemented pulse-resolved correlation in special suite. PR #270
  - Add curve fitting in histogram window. PR #267
  - Support streaming Basler camera from files. PR #266
  - Add more fitting types, Gaussian, Lorentzian and erf. PR #265
  - Add pump-probe FOM to histogram window. PR #263
  - Reset moving average in correlation scan mode. PR #263

15.2 0.9.1 (15 July 2020)

- **Bug Fix**
  - Fix transform type update in TransformView in ImageTool. PR #251

- **Improvement**
  - PlotItem will not be shown in legend if its name is empty. PR #254
  - Improve y2-axis plot implementation. PR #253
  - Improve stack detector modules. PR #247
  - Implement ScatterPlotItem to replace the pyqtgraph one. PR #238
- **New Feature**
  - Implement curve fitting in correlation window. PR #255
  - Implement azimuthal integration and concentric ring detection in C++. PR #252

15.3 0.9.0 (30 June 2020)

- **Bug Fix**
  - Fix bug in data source tree introduced in 0.8.4. PR #235
  - Fix transparent Nan pixel with the latest pyqtgraph version. PR #231
- **Improvement**
  - Improve performance of PlotCurveItem and add benchmarks for PlotItems and ImageViewF. PR #243
  - Make ctrl widgets in Imagetool more compact. PR #241
  - Improve C++ code quality and suppress TBB deprecate warning. PR #239
  - Add meter bar to plot area. PR #236
  - Reimplement ImageItem and PlotItem to replace the pyqtgraph ones. PR #232, PR #242
  - Improve data source updating and color encoding matched source items in the data source tree. PR #230, PR #234
  - Rename configurator to analysis setup manager and allow take snapshot for each setup. PR #228
  - Update data source and trouble shooting in documentation. PR #227
  - Add summary of compiler flags for EXtra-foam-python. PR #226
  - Update installation (gcc7) and CI (gcc8). PR #226
  - Rename misleading mouse mode in the right-click context menu. PR #211
  - Update pyqtgraph to the latest master branch. PR #206, PR #211, PR #233
- **New Feature**
  - Annotate peaks in azimuthal integration view. PR #240
  - Enable Legend for all the plot items. PR #206, PR #237
  - Implement logarithmic X/Y scale in right-click context menu. PR #206
  - Enable C++ API installation and add examples. PR #227

15.4 0.8.4 (8 June 2020)

- **Bug Fix**
  - Beam size in the bar plot will change when the resolution changes in the correlation analysis. PR #209
- **Improvement**
  - Update Redis versions: server -> 6.0.5; redis-py -> 3.5.2; hiredis-py -> 1.0.1. PR #220
  - Slightly improve image proc C++ code performance on machines with few threads. PR #219
  - Visualize data type in data source Tree. PR #218
  - Improve the performance of pulse filter. PR #214
  - Improve setup.py. PR #212
  - Keep data when resolution changes in correlation analysis; move sequence classes into algorithms/data_structures. PR #209
  - Mask ASIC edge when n_modules == 1; simplify geometry binding code. PR #207
  - Add benchmark for generalized geometry. PR #205
  - Make special suite more self-contained. PR #204
  - Mask JungFrau and JungFrauPR. PR #200
  - Move statistics ctrl widgets into corresponding windows. PR #199
- **New Feature**
  - Color encoding matched sources in the data source tree. PR #220
  - Add processed pulse counter in ImageTool. PR #216
  - Add support for C++ API. PR #213
  - Add xtensor-blas as a submodule. PR #210
  - Implement image transform processor and view (FFT, edge detection). PR #203
  - Integrate Karabo gate (PipeToEXtraFoam device) which allows request pipeline and control data in special suite. PR #168

15.5 0.8.3 (11 May 2020)

- **Breaking change**
  - In the terminal, "–n_modules 2" is required to run JungFrauPR with two modules. PR #41
- **Bug Fix**
  - Change pixel size of ePix100 from 0.11 mm to 0.05 mm. PR #189
- **Improvement**
  - Mask tile/ASIC edges by default. PR #192
  - Improve geometry 1M and its unittest. PR #190
  - Invert y axis for displayed image. PR #187
  - Rename geometry to geometry_1m in C++. PR #186

  - Improve tr-XAS analysis in special suite. PR #163, PR #183
  - Improve correlating error message. PR #182
  - Improve documentation for special suite. PR #177
  - New reset interface in special suite. PR #170
  - Regularize names of methods and attributes in special suite. PR #167
  - Add new mode, start/end train ID control and progress bar, etc. in FileStreamer. PR #166
  - Move definition of meta source from config to SourceCatalog. PR #165
  - Use correlated queue in special suite. PR #164
  - Improve shape comparing error message in C++. PR #160
  - Improve mask image data implementation and interface. PR #157
  - Move image assembler into image processor. PR #155
  - Refactor masking code. PR #149
  - Implement generic binding for nansum and nanmean. PR #114

- **New Feature**
  - Add axis calibration in Gotthard analysis. PR #179
  - Implement generalized geometry for multi-module detectors. PR #175, PR #196
  - Implement streaming JungFrauPR data from files. PR #174
  - Implement Gotthard pump-probe analysis in special suite. PR #173, PR #178
  - Add ROI histogram in CameraView in special suite. PR #172
  - Add ROI control in special suite. PR #171
  - Implement XAS-TIM-XMCD in special suite. PR #162
  - Implement MultiCameraView in special suite. PR #147
  - Implement XAS-TIM in special suite. PR #146
  - Implement load and save mask in pixel coordinates. PR #132, PR #154, PR #185, PR #191, PR #197

15.6 0.8.2 (8 April 2020)

- **Bug Fix**
  - Fix not able to close file stream process when closing, if the file stream window is opened through the main GUI. PR #122
  - Fix offset correction switch between dark and offset. PR #141
- **Improvement**
  - Move mouse hover (x, y, v) display implementation to ImageViewF. PR #148
  - Visualize dark and offset separately. PR #141
  - Improve loading reference image and calibration constants. PR #141
  - Implement smart auto levels of image. PR #138
  - Enhance SourceCatalog.add_item. PR #137

- Improve class init with moving average descriptor. PR #136
- Bump EXtra-data version and remove duplicated code. PR #131
- Tweak assembling code in C++ to make the result exactly the same as EXtra-geom. PR #129
- Simplify ImageProc binding code. PR #125
- Update dependencies. PR #118
- Update documentation. PR #115, PR #130
- Move tr-XAS analysis to special suite. PR #89

- **New Feature**
  - Generalize file stream. PR #122
  - Add standard deviation, variance and speckle contrast into ROI FOM. PR #119
  - Implement tile edge mask for modular detectors. PR #110
  - Add support for fast ADC as a digitizer source. PR #101
  - Implement Camera view (special suite). PR #89
  - Implement Gotthard analysis (special suite) for MID. PR #89
  - Implement interface and examples for special analysis suite. PR #89

15.7 0.8.1 (16 March 2020)

- **Improvement**
  - Automatically reset empty image mask with inconsistent shape. PR #104
- **New Feature**
  - Implement AGIPD 1M geometry in C++. PR #102
  - Add ROI1_DIV_ROI2 as an option for ROI FOM. PR #103
  - Implement normalization for ROI FOM. PR #96
  - Implement ROI FOM master-slave scan. PR #93
  - Add branch-based CI and Singularity image deployment. PR #92
  - Add support for ePix100 detector. PR #90
  - Implement save and load metadata. PR #87

15.8 0.8.0.1 (3 March 2020)

- **Bug Fix**
  - Fix display bug in ImageTool PR #85

15.9 0.8.0 (2 March 2020)

- **Improvement**
  - Get rid of the artifact induced by masking pixel to zero when calculating statistics, e.g. mean, median, std.
  - Provide a mask to pyFAI to perform azimuthal integration. PR #61
  - New C++ implementation to mask pixel in Nan and/or return a boolean mask. PR #61
  - ROI pulse FOM and NORM will only be calculated after registration. PR #61
- **New Feature**
  - Enable train-resolved FOM filter. PR #78
  - Display numbers of processed and dropped trains. PR #77
  - Support online single module data from a modular detector. PR #72
  - Allow type selection for 1D projection (sum or mean). PR #71
  - Implement mouse cursor value indicator for PlotWidgetF. PR #66
  - Preliminary implementation of nanmean and nansum in C++. PR #61
- **Bug Fix**
  - Fix pulse-filter in digitizer. PR #80
  - Fix gain/offset slicer for train-resolved detectors. PR #76
  - Use nansum in Tr-XAS analysis. PR #75
  - Fix typo in unittest. PR #74
  - Fix changing device ID in data source on the fly. PR #69

15.10 0.7.3 (24 February 2020)

- **Breaking change**
  - In the terminal, "–topic" becomes a positional argument. PR #41
- **Improvement**
  - Reimplement Color classes. mkPen and mkBrush from pyqtgraph are not needed anymore. PR #53
  - Allow select pipeline policy (wait or drop) via commandline. The default is wait since the data arrival speed is slower than the processing speed during online analysis. PR #45
  - Replace Python's build-in queue.Queue to speed up data transfer. PR #45
  - Improve the visualization of heatmap. PR #44
  - Allow starting instances with different detectors without warning message. PR #41
  - Allow to shutdown others' Redis server to avoid zombie Redis server occupying the port. PR #41
  - Implement Fast assembling for LPD and DSSC in C++. PR #40
  - Resign the config code. Now each instrument will has its own config file, e.g. scs.config.yaml, fxe.config.yaml. All the instrument sources will be set up in the config file. PR #38
  - Implement streaming raw (AGIPD, LPD) data from files and also 'confirmed' streaming raw (AGIPD, LPD) data online. PR #38

- **New Feature**
  - Allow specific bin range of histogram. PR #56
  - Provide ROI histogram for train-resolved detectors; Provide ROI histogram for the averaged image of pulse-resolved detectors. PR #56
  - Display *mean*, *median* and *std* for all histogram plots. PR #56
  - ROI histogram for pulse-resolved detectors. PR #55
  - Double-y plot for 1D binning. PR #53
  - Support normalizing by digitizer (TIM). PR #52
  - Support multiple ZMQ endpoints connections. PR #45
  - Automatically correlate data from the same/different endpoints with train ID. PR #45
  - Allow automatically choosing bin range. PR #44
  - Also add an option to stack the detectors (LPD and DSSC) without assembling. PR #40
  - Control required sources in the DataSourceTree. PR #38
  - Allow filtering by value for all non-detector data sources. PR #38
  - Implement AdqDigitizer processor. PR #38
- **Bug Fix**
  - Fix default AGIPD geometry. PR #62
  - Disable pulse slicer for train-resolved detectors in DataSourceTree and gain/offset correction. PR #56
  - Fix logger level. PR #41
  - Fix extra-foam-kill. PR #41

15.11 0.7.2 (16 January 2020)

- **Improvement**
  - Remove 'AZIMUTHAL_INTEG_RANGE' from configuration PR #32
  - Remove 'process monitor' from action and make it a tab in DataSourceWidget PR #32
  - Reduce the update frequency of plots which accumulates data, for example, correlation, histogram, heatmap, etc., to 1 Hz PR #31
  - Improve Redis server configuration PR #29
  - Allow ImageViewF.setImage(None) PR #28
  - Provide better interface for users to call C++ code PR #25
  - Log geometry change and remove 'AZIMUTHAL_INTEG_POINTS", "CENTER_X", "CENTER_Y" from configuration PR #24
  - Rearrange C++ code and separate benchmark code from unittest PR #15
  - Re-implement PairData -> SimplePairSequence and AccumulatedData -> OneWayAccuPairSequence PR #14
  - Re-implement BinProcessor. Now, data history is stored and users can re-bin it at anytime PR #14
  - Reduce MAX_QUEUE_SIZE from 5 to 2 to reduce latency PR #14

  - Remove 'update_hist' in PumpProbeData and CorrelationData. Now GUI update is completely decoupled from processors PR #14
  - Merge CorrelationWindow into StatisticsWindow. Rename the old statistics widgets to histogram widgets; add a new tab in the MainGUI which is dedicated for 'statistics' control PR #14
  - Update dependencies PR #11
  - Simplify ThreadLogger code PR #10
- **New Feature**
  - Implement q-map visualization PR #32
  - Implement pixel-wise gain-offset correction by loading numpy array from files PR #25
  - New ROI analysis interface (enable different FOMs of ROI; enable pulse-resolved ROI normalizer; enable pulse-resolved ROI1 +/- ROI2 FOM; enable visualization of ROI projection and pulse-resolved ROI FOM in ImageTool) PR #12
- **Bug Fix**
  - Fix a bug in MovingAverageScalar and MovingAverageArray. Setting a new value of None will reset the moving average instead of being ignored PR #14

15.12 0.7.1 (4 December 2019)

This is the first release after migrating from EuXFEL gitlab to github!!!

- **Improvement**
  - Rename omissive fai to foam and change config folder from karaboFAI to EXtra-foam PR #6
- **Test**
  - Migrate CI from EuXFEL gitlab to public github PR #1

15.13 0.7.0 (25 November 2019)

- **Improvement**
  - Change supporting email, (long) description and header content in each file #174
  - Regularize Qt imports #173
  - Re-arange the GUI interface and move image related control into ImageTool #171
  - Add hiredis-py as dependency and improve redis connection infrastructure #170
  - Remove (canvas, dockarea, flowchart, multiprocess) from pyqtgraph code base #155
- **New Feature**
  - Support online FCCD raw data analysis #169
  - Publish available data sources in Redis and improve infrastructure in client proxy #166
- **Bug Fix**
  - Clean-up thread logger gracefully #170

15.14 0.6.2 (15 November 2019)

- **Improvement**
  - Code clean up and improve base classes in GUI #164
  - Improve image processing code in cpp (align with xfai) #159
  - Enhance ImageTool interface (integrate functions in DarkRunWindow and OverviewWindow) #158
- **New Feature**
  - Introduce special analysis interface (implement tr-XAS) #165
  - Add an option to not normalize VFOM #162
- **Bug Fix**
  - Pulse slicer will also slice the stored dark images #165

15.15 0.6.1 (28 October 2019)

- **Improvement**
  - Remove XAS related code (GUI, processor, etc.) !154
  - Update import location of ZMQStreamer !151
  - Improve system information summary interface and enable detecting GPU resources !138
- **New Feature**
  - Implement normalization by XGM pipeline data !157
  - New data source management interface !157
  - Implemented web monitor in Dash !152

15.16 0.6.0 (31 August 2019)

- **Bug Fix**
  - Assembling image from files, when non-detector source available in data !140
  - Add mid specific data sources in ctrl widget !139
- **Improvement**
  - Code clean-up ! 138
  - Remove moving average of images !128
  - Display number of filtered pulses/train in OverviewWindow !128
  - Raise StopPipelineError in ImageProcessorPulse instead of ProcessingError !128
- **New Feature**
- **Test**

15.17 0.5.5 (26 August 2019)

- **Bug Fix**
  - Fix user defined control data in 1D binning analysis !134
  - Fix image mask in pulse-resolved ROI !133
- **Improvement**
  - Allow instrument sources to stream apart from DET !135
  - Allow shutdown idling karaboFAI instance remotely !130
  - Rearrange plot widgets !121
  - Improve the API for C++ image processing code !116 !129
  - AGIPD also works with bridge data with 'ONDA' format !115
- **New Feature**
  - Add statistics plot for pulse of interest !127
- **Test**

15.18 0.5.4 (20 August 2019)

- **Bug Fix**
  - Fix bug if shape changes when using out array for assembling !122
- **Improvement**
- **New Feature**
  - Support pulse-resolved and two-module JungFrau !83
- **Test**

15.19 0.5.3 (16 August 2019)

- **Bug Fix**
  - Fix series nan mean two images !106
- **Improvement**
  - Introduce 'TOPIC' to separate instrument specific sources !114
  - Implement masking image in cpp !110
- **New Feature**
  - Implement DarkRunWindow !109
  - Allow save image and load reference in ImageTool !107
- **Test**
  - Integrate cpp unittest into setuptools and CI (both parallel and series) !110

15.20 0.5.2 (9 August 2019)

- **Bug Fix**
- **Improvement**
  - Prevent costly GUI updating from blocking data acquisition !101
  - Improve nanmean performance when simple slice is not applicable !97
  - Add output array in image assembly !85
- **New Feature**
  - List critical information of a run in FileStreamer window !103
  - Implement AboutWindow !102
  - Pulse slicing and data reduction !99
  - New widget SmartSliceLineEdit !98
- **Test**

15.21 0.5.1 (5 August 2019)

- **Bug Fix**
  - Capture exception when trying to kill others' instance !93
  - Add AGPID detector in FileServer !90
  - Fix when a new detector key cannot be found in an old config file !87
- **Improvement**
  - Implement parallel version of xt_nanmean_images !91
  - Delete detector data in raw data after Assembling !88
  - Update geometry file and default quad positins for DSSC !86
  - Make compiling with TBB and XSIMD default !84
- **New Feature**
  - Added MID_DET… source to list in AGIPD dict in config.py !94
- **Test**
  - Unittest statistics #82
  - Unittest for command proxy #81

CHAPTER 16

Indices and tables

- genindex
- modindex
- search