BEATS User Manual Documentation

Release 0.1

SESAME

May 08, 2024

CONTENTS

Welcome to SESAME ID10-BEATS User Manual! This guide is designed to help you explore the instructions, features and functionalities of the BEATS beamline. Whether you are a first-time or an experienced user, this manual will provide you with step-by-step instructions, tips, and troubleshooting advice to make the most out of your beamtime at SESAME BEATS.

CHAPTER

ONE

CONTENT

1.1 About

The BEATS beamline is located at port I10 of the SESAME storage ring and operates a hard X-ray full-field radiography and micro Computed Tomography (CT) station. Hard x-ray imaging can be applied to numerous scientific areas.

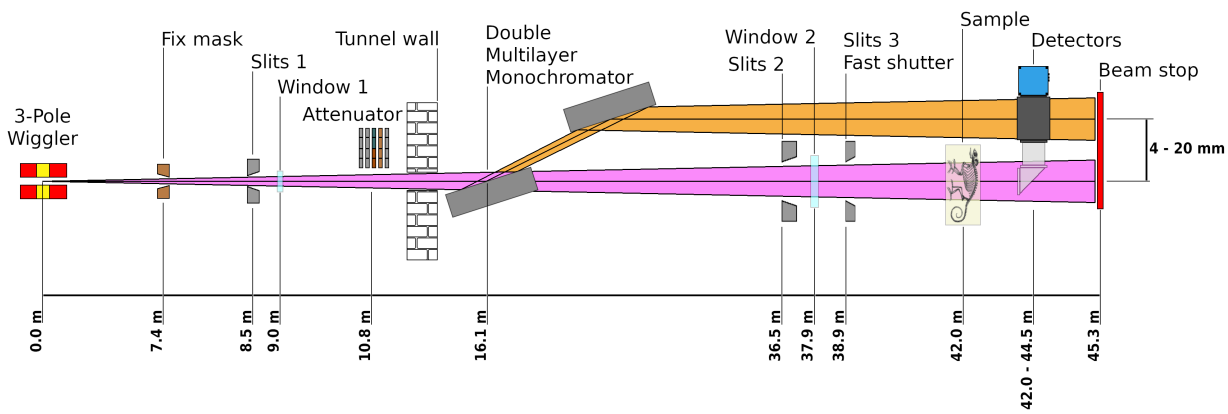


Fig. 1: *Layout of the ID-10 BEATS beamline.*

The total length of the BEATS beamline is 45.3 m. The insertion device is a 2.9 T 3-Pole Wiggler ($E_{crytical}$ = 12 keV) that provides photon energies well above 80 keV. The beamline can be operated in two modalities: filtered white beam, and monochromatic beam using a Double Multilayer Monochromator (DMM).

The usable beam size at the sample position is 70 x 15 mm^2, with estimated flux as high as 1×10^{10} Ph/s/mm^2 in 0.1% of the source bandwidth (white beam configuration).

The beamline endstation is composed of a sample air-bearing tomography manipulator, and of indirect X-ray detectors coupled to sCMOS cameras. Object pixel sizes in the range from 13.0 m down to 0.65 m are achieved.

1.2 General Information

1.2.1 Apply for beamtime

Visit the SESAME User Guide for information regarding proposal submission and calls for proposals.

SESAME User Office contact: useroffice@sesame.org.jo

1.2.2 Useful Links

More information about the BEATS project:

- Beamline specifications on Wayforlight.eu
- SESAME ID10-BEATS beamline web page

1.2.3 The BEATS project

The European Horizon 2020 project BEAmline for Tomography at SESAME (BEATS) had the objective to design, procure, construct and commission a beamline for hard X-ray full-field tomography at the SESAME synchrotron in Jordan. The project was launched in January 2019 and completed in June 2023 with the inauguration of the beamline.

BEATS involved leading research facilities in the Middle East (SESAME and the Cyprus Institute), and European synchrotron radiation facilities ALBA-CELLS (Spain), DESY (Germany), the ESRF (France), Elettra (Italy), INFN (Italy), PSI (Switzerland), SESAME (Jordan) and SOLARIS (Poland).

The initiative was funded by the European Union's Horizon 2020 research and innovation programme. The project was coordinated by the ESRF.

More information about the BEATS project:

- BEATS project webpage
- BEATS Science Case
- BEATS Technical Design Report

1.3 Prepare for beamtime

1.3.1 Travel and guesthouse arrangement

SESAME User Office will be in contact with you to organize your visit, including guesthouse reservation, providing you with an access card, and coordinating your safety training. Read the points below before your visit. This will help you have a smooth start of your beamtime at BEATS.

Your stay at SESAME's guesthouse

Please bring your soap/shampoo, and lotions. Towels are available in the guesthouse. **SESAME does not offer a cafeteria service at the moment, and there is no restaurant or café in the vicinity of the institute.** There are a few shops nearby that can be reached in a couple of minutes by car. We recommend you to bring food that you can cook at the guesthouse. You can also ask the taxi driver to stop at a supermarket on your way from the airport (this is generally included in the transportation cost).

1.3.2 Visualize and inspect your 3D data

To use your beamtime efficiently, you must be able to inspect the recorded data. We expect you to know how to do basic operations in ImageJ! For instructions on how to install and use ImageJ see the section *Load reconstructed volume with ImageJ*. ImageJ is also a powerful tool for image processing and analysis of your datasets after the beamtime! If you need more tools, visit our list of 3D *Data analysis software*.

1.3.3 Bring enough storage

Following SESAME's data policy, we will do our best to archive and maintain your data for at least 5 years after the experiment.

> **Warning:** Due to the massive size of tomography datasets, you will not be able to access data remotely after the beamtime. You are responsible for taking a copy of all data with you before leaving SESAME. For this, we ask you to bring external drive(s) with at least 1 TeraByte of free space.

1.3.4 Beamline essentials

Please take a look at the following sections of the BEATS beamline manual to familiarize with the beamline operation. In particular, take a look at:

- the *Hutch Search Procedure*, and at
- the *Tomographic reconstruction at BEATS* section.

1.4 Quick start guide

Welcome to ID-10 BEATS. Below is a coarse overview of the most important steps to setup and perform your first scan. To keep to page simple but provide all information needed, links to in depth descriptions are provided in the corresponding sections.

Note: The beamline is setup for your experiment at the start of the beamtime together with the beamline staff. This includes energy adjustment and mounting detector and optics. Most of the times, there will be little to change from one scan to the next, besides aligning the sample. Ask the beamline staff before changing any beamline setting.

1.4.1 Getting started

The beamline control GUI allows to control the in-vacuum instrumentation in the beamline front-end, optics hutch, and experimental hutch.

Here are things users should be comfortable doing during their experiments.

In particular, you might need to adjust the configuration of:

- One or more of the beamline slits
- Attenuator system
- search the hutch and open the shutters
- adjust the sample on the stage
- start a scan

The heartpiece of beamline operation is the `beats-qt` GUI.

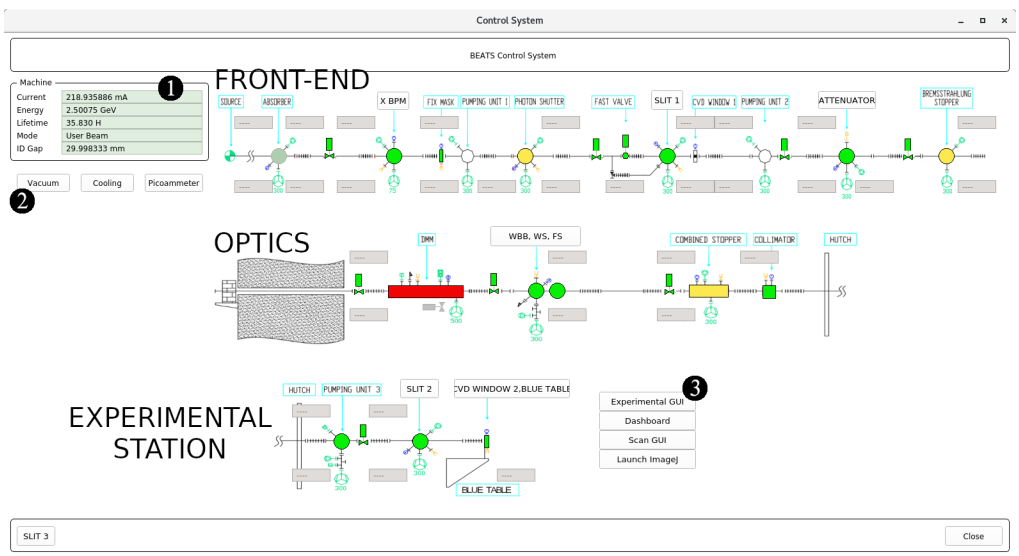


Fig. 2: *Figure 1: BEATS beamline control GUI.*

If it's not already started or gets closed accidentally, it can be started from a linux console on the control computer (e.g. by pressing ctrl + alt + t) and typing

```
$ beats-qt
```

The BEATS beamline GUI allows to #. display the machine status and parameters. #. launch the BEATS *Beamline vacuum GUI*. The vacuum GUI is used to open the shutters of the beamline. #. launch the BEATS *Experimental GUI*. This GUI controls the sample manipulator and detection systems.

1.4.2 Beamline vacuum GUI

To be able to acquire meaningful data, you need to have X-rays on your sample. Therefore, the shutters need to be opened at least. Open the shutters only when needed and you want to measure (also to protect your sample). Openening the shutters can only be performed after the hutch has been searched.

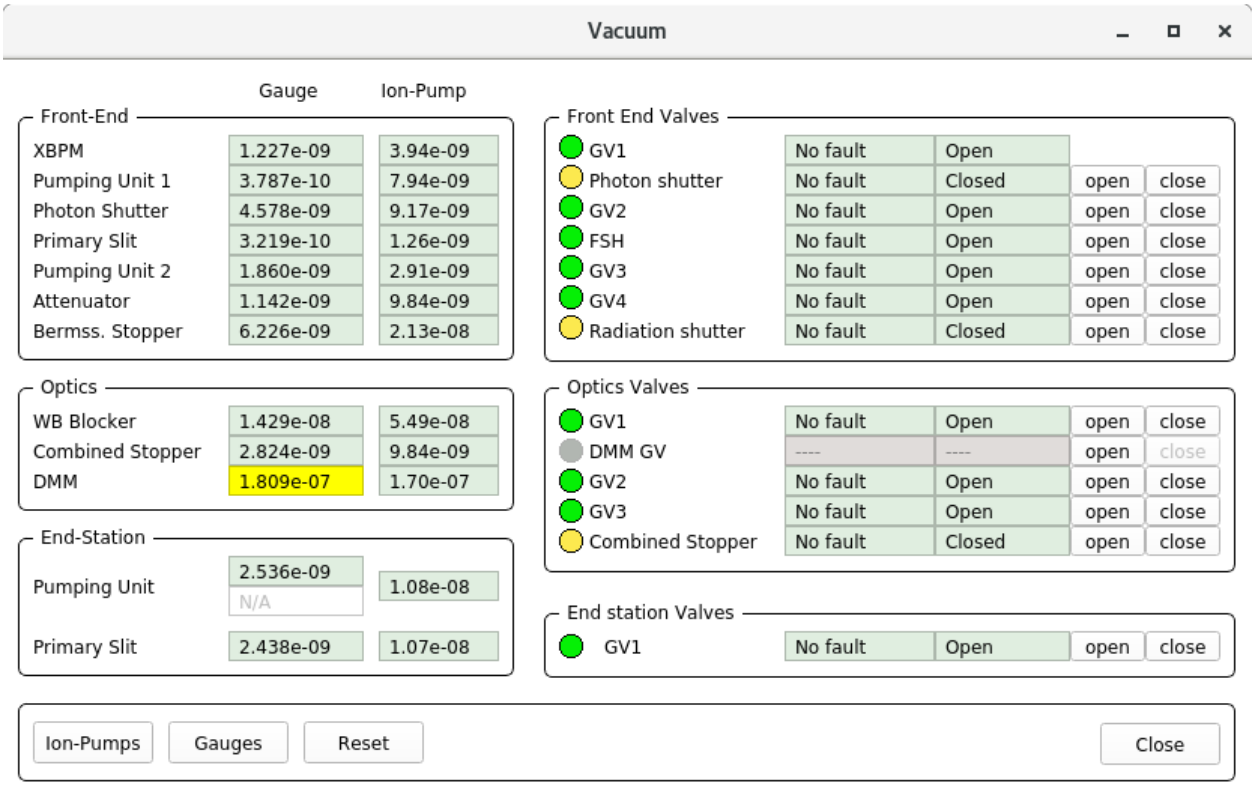


Fig. 3: *Figure 1: BEATS vacuum GUI.*

To open the beamline light path, the shutters must be opened in the following order:

1. Radiation shutter.
2. Photon shutter.
3. Combined stopper.

Note: It is only allowed to open the beamline shutters once the hutches have been searched. You will see a red button besides one of the shutter if opening is not allowed.

Warning: Before opening the beamline shutters for the first time, verify with your local contact the setup of the beamline (slits, attenuator, …).

1.4.3 Search the hutch

Hutch Search Procedure

The **Personnel safety system (PSS)** operates as an access control system for the shielded enclosed areas to prevent access to dangerous levels of radiation. One of the main PSS procedures is the search of the hutches, with the purpose to make sure that all staff left before the x-ray beam is brought into the hutch. Only when the search is completed (all search buttons have been pressed, confirming that no person has been found in the respective area) and the searcher left the hutch, the beamline shutters can be opened.

Fig. 4: *(Left) PSS hutch search button. (Center) PSS cabinet; hutch is ready to start search. (Right) PSS showing hutch door interlock and beamline shutters OPEN.*

Note: The purpose of the search is to make sure that nobody is in the hutch when the x-ray beam enters it. During the search, all corners of the hutch must be inspected and search buttons pressed by the person performing the search. All doors must remain closed during this procedure.

To preform the search of the hutch:

1. Do a pre-check to make sure that there are no people inside the hutch
2. Close the hutch door
3. Make sure that the safety officer and coordinators keys are in ON position, then press start search button. The red warning lights in the hutch will start blinking with variable tone audible warning to notify personnel that a search is in progress
4. Open the door, go inside the hutch then close the door. If you or someone opened the door after this step and before finishing the search, the search will fail and you have to repeat the procedure
5. Go to the first search point, look for people, if there are no people then press Search Button 1
6. Go to the second search point, look for people, if there are no people then press Search Button 2.
7. After searching the hutch and pressing all the buttons, leave the hutch and close the door. The door will be locked automatically
8. Press finish search button. The audible warning changes to an intermittent tone before enabling the opening of the shutters

Note: Note: The blinking frequency of the search buttons will change and become faster after a few seconds. It's at this time that you can press them.

1.4.4 Start streaming data

To access the BEATS Dashboard, type the following command:

```
$ BEATS_DAQ_Control_Monitor
```

the main GUI will appear:

For a very detailed description of the procedure go to *BEATS Dashboard*!

1.4.5 Set up your experiment

Set up your experiment

Warning: **Collision danger**: Only perform this operation together with the beamline staff. You must always pay attention to the position of endstation, detectors and sample, while performing the alignment. **Always move small steps when endstation and detector are close to each other!**

Preliminary steps

1. Mount your sample on top of the tomography endstation (Figure 1 RIGHT)
2. Turn on the alignment lasers
3. Use the laptop close to the endstation to:
    1. Pre-align sample on the intersection of the laser planes (you can verify this also moving the ROT stage)
    2. Pre-align the detector scintillator on the same line
    3. Set the distance between sample and detector to the desired value

Sample mount

- Samples can be mounted on the tomography rotation stage with M4 screws as shown Figure 2.
- A set of standard kinematic mounts from Newport is also available: M-BK-1A (download -> drawing).

Fig. 5: *Figure 1: BEATS Dashboard main window*

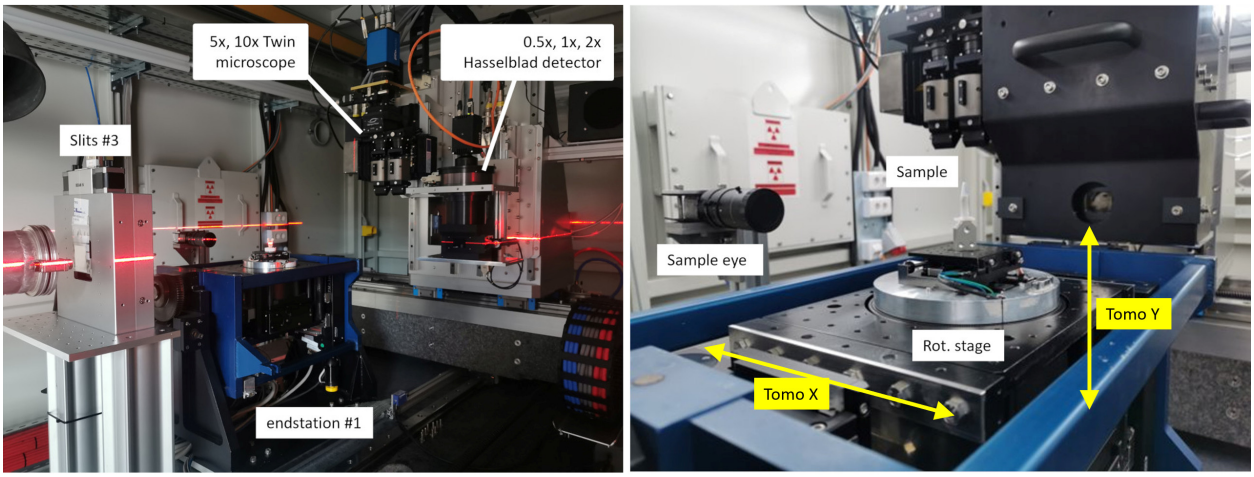


Fig. 6: *Figure 1: BEATS experimental station. Two laser lines are used to pre-align sample and detector on the beam.*

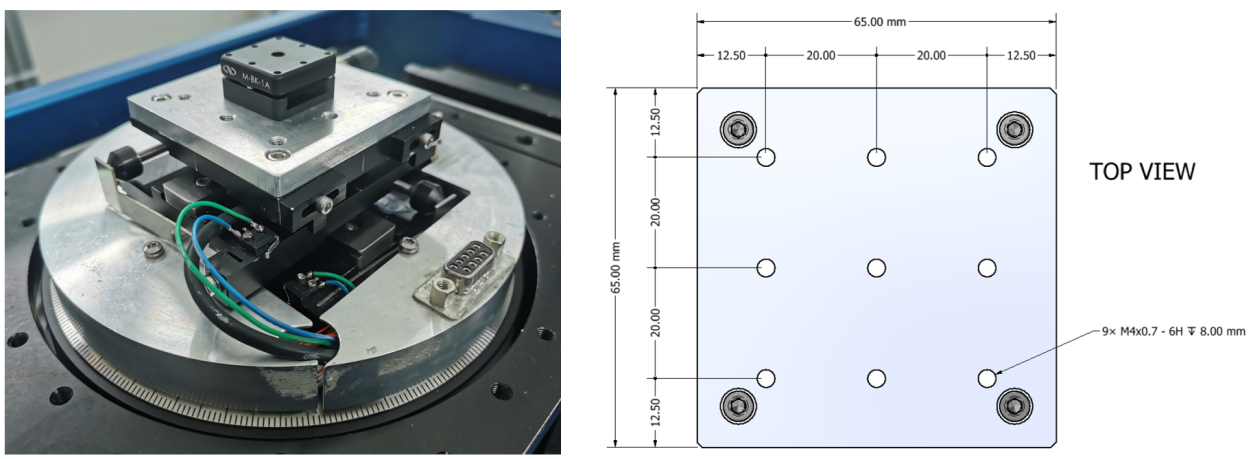


Fig. 7: *Figure 2: (LEFT) Detail of sample tomography stage. (RIGHT) The sample plate has 9 M4 holes that can be used for custom sample support.*

Sample alignment

In order to align the sample on the center of rotation of the rotary stage 4 motorized axis are needed:

• **TOMO_Y** (vertical motion)
• **TOMO_X** (horizontal motion perpendicular to the beam)
• **SAMPLE_SX** (horizontal motion above the rotary stage)
• **SAMPLE_SZ** (horizontal motion normal to "sample top X" above the rotary stage)

The motion range of the tomography endstation (Figure 1 RIGHT) is:

• **TOMO_X**: 60.0 mm
• **TOMO_Y**: 47.0 mm

Sample alignment procedure

Load the sample on the kinematic mount (for automatic alignemt of the endstation with tomoalign use the tungsten wire available at the beamline as sample) then:

1. Perform the hutch_search
2. Open the shutters using the vacuum
3. Use the *Experimental GUI* to move the sample up/down until the sample is in the field of view of detector.

Experimental GUI

img/exp_gui.png

Fig. 8: *Figure 3: BEATS experimental GUI.*

1.4.6 Reconstruct your data

Quick CT reconstruction

Use the Jupyter Notebook pipeline at the link below for reconstructing and visualizing data collected at BEATS. The notebook uses TomoPy, and is a good starting point for processing your data. You can make a copy of it, and modify the code based on your requirements.

BEATS CT reconstruction pipeline

Minimal TomoPy reconstruction pipeline for data collected at the BEATS beamline of SESAME

Created on: 23.05.2021 Last update: 09.10.2023

- By: Gianluca Iori, 2023
- Code license: MIT
- Narrative license: CC-BY-NC-SA

Type `Ctrl` + `Enter` on a single cell to run it.

Load experiment data

Enter the **sample_name** and the correct **output_dir**

```
sample_name = "bee_yazeed-20231001T170032"
work_dir = "/mnt/PETRA/SED/BEATS/IH/"+sample_name
h5file = work_dir+"/"+sample_name+".h5"

output_dir = "/mnt/PETRA/SED/BEATS/IH/scratch/tmp/"
recon_dir = output_dir+sample_name+"/recon/"
cor_dir = output_dir+sample_name+"/cor/"
```

Load the complete dataset (or)

```
# projs, flats, darks, theta = dxchange.read_aps_32id(h5file, exchange_rank=0)
```

Read a portion of the dataset

- `sino` controls the vertical detector lines to read - `sino=(1360, 2160, 1)`
- `proj` defines the range of projections - `proj=(1, 1001, 1)`

```
sino=(1360, 2160, 1)
proj=(1, 1001, 1)
```

```
projs, flats, darks, _ = dxchange.read_aps_32id(h5file, exchange_rank=0, sino=sino,␣
↪proj=proj)
theta = np.radians(dxchange.read_hdf5(h5file, 'exchange/theta', slc=(proj,)))

print("Dataset size: ", projs[:, :, :].shape[:], " - dtype: ", projs.dtype)
print("Flat fields size: ", flats[:, :, :].shape[:])
print("Dark fields size: ", darks[:, :, :].shape[:])
print("Theta array size: ", theta.shape[:])
```

```
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/bee_yazeed-
↪20231001T170032/bee_yazeed-20231001T170032.h5
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/bee_yazeed-
```

(continues on next page)

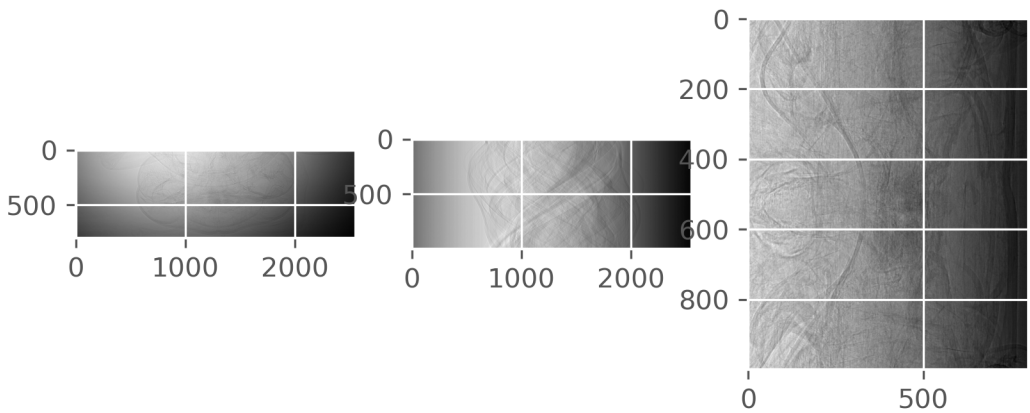
(continued from previous page)

```
↪20231001T170032/bee_yazeed-20231001T170032.h5
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/bee_yazeed-
↪20231001T170032/bee_yazeed-20231001T170032.h5
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/bee_yazeed-
↪20231001T170032/bee_yazeed-20231001T170032.h5
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/bee_yazeed-
↪20231001T170032/bee_yazeed-20231001T170032.h5
```

```
Dataset size:  (1000, 800, 2560)  - dtype:  uint16
Flat fields size:  (102, 800, 2560)
Dark fields size:  (102, 800, 2560)
Theta array size:  (1000,)
```

At any time you can take a look at your 3D data with `ru.plotmidplanes(data)`

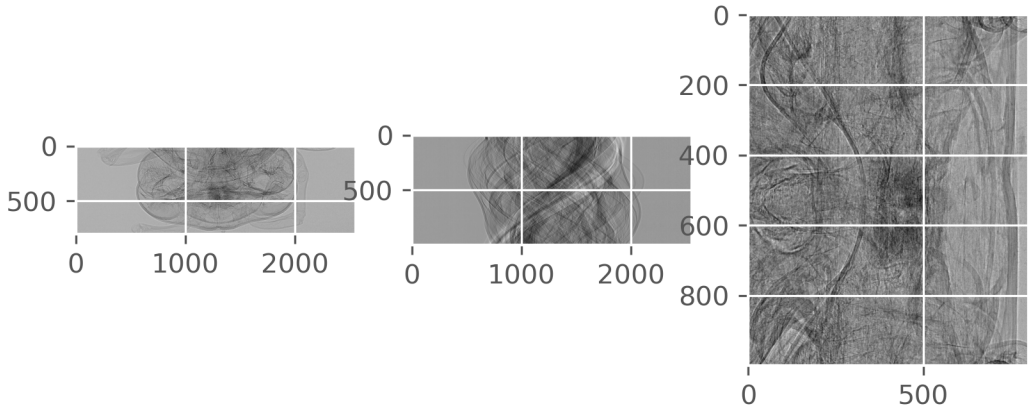
```
ru.plot_midplanes(projs)
```



Flat field correction

Normalize the image background.

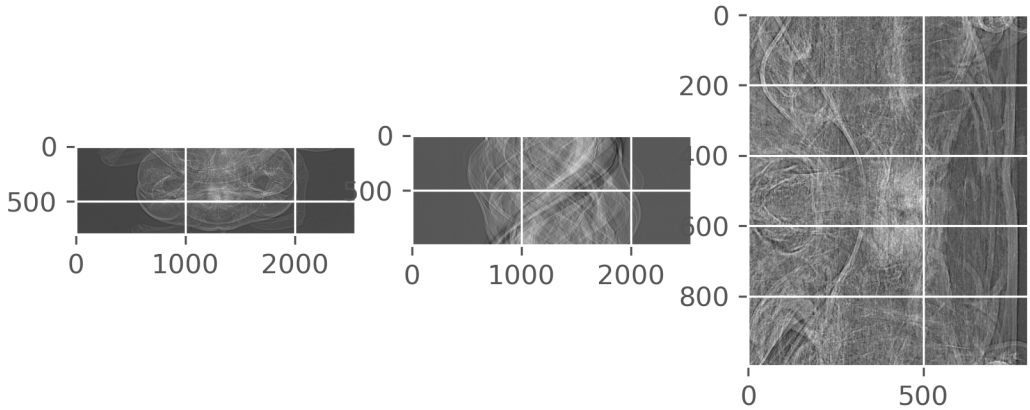
```
projs = tomopy.normalize(projs, flats, darks, ncore=ncore, averaging='median')
ru.plot_midplanes(projs)
```



Log transform

Calculate $ -log(projs) $ to linearize transmission tomography data.

```
projs = tomopy.minus_log(projs, ncore=ncore)
ru.plot_midplanes(projs)
```



Center Of Rotation (COR)

Save images reconstructed with COR range

```
cor_range = [1298, 1304, 0.5]
```

```
tomopy.write_center(projs, theta, cor_dir, cor_range)
```

```
INFO:tomopy.recon.algorithm:Reconstructing 12 slice groups with 12 master threads...
```

View them in Fiji

```
# os.system(Fiji_exe_stack + cor_dir+'{:04.2f}'.format(COR[0])+'.tiff &')
```

Manually insert the best COR

```
COR = 1301
```

Automatic detect COR

```
# COR = tomopy.find_center(projs, theta, init=projs.shape[2]/2, tol=0.5) # ind=200,
# print("Automatic detected COR: ", COR, " - tomopy.find_center")

COR = tomopy.find_center_vo(projs, ncore=ncore)
print("Automatic detected COR: ", COR, " - tomopy.find_center_vo")
```

```
Automatic detected COR:  1301.5  - tomopy.find_center_vo
```

Reconstruction

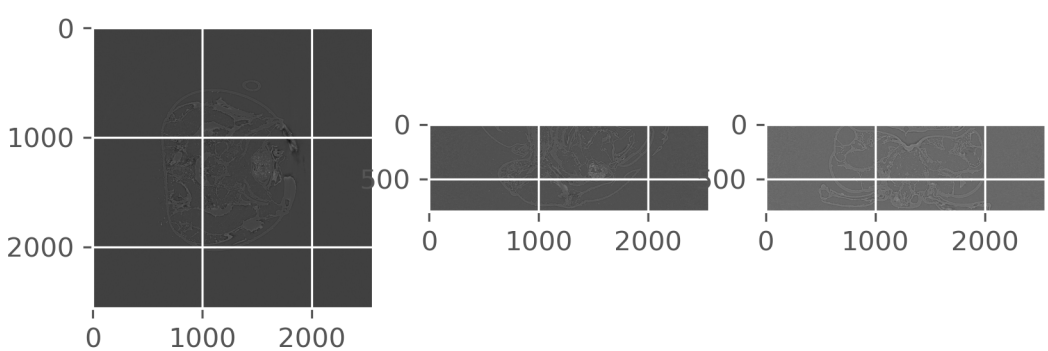
CPU reconstruction of the entire dataset

Algorithm	gridrec

```
recon = tomopy.recon(projs,
                     theta,
                     center=COR,
                     algorithm='gridrec',
                     sinogram_order=False,
                     ncore=ncore)
```

```
INFO:tomopy.recon.algorithm:Reconstructing 36 slice groups with 36 master threads...
```

```
ru.plot_midplanes(recon)
```



GPU reconstruction with the ASTRA toolbox

Algorithm	fbp CUDA ASTRA

```
# options = {'proj_type': 'cuda', 'method': 'FBP_CUDA'}

# recon = tomopy.recon(projs,
#                      theta,
#                      center=COR,
#                      algorithm=tomopy.astra,
#                      options=options,
#                      ncore=1)
```

Write reconstructed dataset

Write output tiff stack as `float32`

```
fileout = recon_dir+'slice.tiff'
dxchange.writer.write_tiff_stack(recon, fname=fileout, axis=0, digit=4, start=0,␣
↪overwrite=True)
```

Open virtual stack in ImageJ

```
# os.system(Fiji_exe_stack + fileout + ' &')
```

For further examples and a comprehensive description see *Tomographic reconstruction at BEATS*.

1.5 Data Acquisition (DAQ)

1.5.1 DAQ System

TomoScan

Tomoscan installation reference

Tomoscan reference

BEATS Tomoscan

Installing BEATS TomoScan

This page includes information about the needed packages to run the DAQ system.

Prerequisites

The following should be installed on the computer before running the system:

- Linux redhat based OS (This work has been done under CentOS 7.4, however, there should be no reason to not work on other distributions)
- EPICS BEATS IOCs (motion and scan IOCs)
- Python 3.9
- QT 4.1.0 based on 5.9.7.

Python virtual environment

venv module of Python is being used as a virtual environment for this setup.

The venv module of python provides support for creating **virtual environments** that is isolated from system site directories. Normally, each virtual environment has its own Python binary (which matches the version of the binary that was used to create this environment) and can have its own independent set of installed Python packages in its site directories.

to install and create venv:

```
$ pip3.9 install virtualenv
$ python3.9 -m venv ${Home}/DAQ/SW/venv3.9
```

to create alias of you environment:

```
$ vi ~/.bashrc
```

add the following line to the file:

```
alias p3='source ${Home}/DAQ/SW/venv3.9/bin/activate'
```

resource your bashrc:

```
source ~/.bashrc
```

Packages and libraries

The tool needs set of python packages and Qt libraries installed and configured.

Pyhon packages:

The list below contains the list of python packages needed for the scanning tool to run. After activating the python virtual environment (by typing **p3** in the terminal), you can use **pip** to install them in the virtual environment or you can copy this list in a text file (requirements.txt) and install them at once using this command (pip install -r requirements.txt)

```
pymsgbox
pyepics
h5py
pvapy
paramiko
colorama
PyQt5
```

Qt and its libraries

1. Install epics from SESAME's local repo.
2. Download Qt creator: https://drive.sesame.org.jo/owncloud/index.php/s/LO3GLyDkPMWZKU9.
3. Install qt-creator-opensource-linux-x86_64-4.13.3.run.
4. Install epics-qt, qt5, qwt, or anything related to *qt* packages by `yum` command.
5. Go to `.bashrc` and copy the following:

```
export EPICS_BASE='/opt/epics/base'
export EPICS_HOST_ARCH=linux-x86_64
export PATH=${PATH}:/opt/qtcreator-4.13.3/bin/
export QWT_ROOT=/usr/local/qwt-6.1.3
export QWT_INCLUDE_PATH=${QWT_ROOT}/include
export QE_TARGET_DIR=/usr/local/epics-qt
export PATH=${EPICS_BASE}/bin/$EPICS_HOST_ARCH:${QE_TARGET_DIR}/bin/${EPICS_
↪HOST_ARCH}:/usr/lib64/qt5/bin:${PATH}
export LD_LIBRARY_PATH=${EPICS_BASE}/lib/${EPICS_HOST_ARCH}:/usr/local/qwt-6.1.
↪3/lib:${QE_TARGET_DIR}/lib/${EPICS_HOST_ARCH}:${QE_TARGET_DIR}/lib/${EPICS_
↪HOST_ARCH}/designer
export QT_PLUGIN_PATH=${QT_PLUGIN_PATH}:${QWT_ROOT}/plugins:$QE_TARGET_DIR/lib/
↪$EPICS_HOST_ARCH
```

6. `source .bashrc`
7. To validate your setup, create a new project and open the designer, you should get qwt and epics qt widgets shown.

Clone the tomoScan DAQ system

Note: Make sure that the python environment is activated before proceeding with this section.

The scanning tool (BEATS_tomoscan) is available on github. The most recent version can be found on this link: https://github.com/SESAME-Synchrotron/BEATS_tomoscan.git. To clone and run, launch your terminal then do the follwoing:

```
$ cd /opt/epics/support
$ git clone git@github.com:SESAME-Synchrotron/BEATS_tomoscan.git
```

check configure/RELEASE all the epics directories are correct:

```
$ EPICS_BASE=/opt/epics/base
$ SUPPORT=/opt/epics/support
$ BUSY=$(SUPPORT)/busy
$ AUTOSAVE=$(SUPPORT)/autosave
$ ASYN=$(SUPPORT)/asyn
```

Run the following commands on BEATS_tomoscan

```
$ make
$ python setup.py install
```

Postrequisites

The following should be installed on the computer after installing the BEATS_tomoscan:

- BEATS H5 Writer refer to:
- PETRA/SED/BEATS/ sharing file system should be mounted on the local station.
- BEATS_Dashboard refering to: *Installing BEATS Dashboard*
- clone the SEDSS package to this directory as:

```
$ cd /${Home}/DAQ/SW/venv3.9/lib/python3.9/site-packages/tomoscan-0.1-py3.9.egg
$ git clone git@github.com:SESAME-Synchrotron/SEDSS.git
```

Run experiment and collect data

in order to running the experiment, go to: *BEATS Dashboard*

Welcome to BEATS Dashboard documentation!

This is the documentation for the BEATS dashboard, which manages the BEATS IOCs and initiates scanning based on the camera type and scanning methodology (step and continuous scan).The tool has been developed by data collection and analysis team of SESAME in collaboration with control team.

This documentation is targeting scientists and developers.

The dashboard is GUI based which enables users to quickly choose the scanning parameters, and thus saving the time of explaining how to get BEATS experimental data.

Dashboard

Current features and future plans

Current featues

At the moment, the BEATS dashboard has the following features:

- User friendly GUI
- Control and monitor the IOCs and scanning methodology (Tomoscan) based on the camera type.
- Online logging (last log).
- Shutters Status Monitoring.
- Single Shot Image GUI, and SSCAN tool.
- Public documentation targeting end-users (how-to).

Scanning tool | future plan

- Adding the available motion stages that are accessible for the user to choose from.
- Adding the motor records values.

Installing BEATS Dashboard

This page includes information about the needed packages to run the BEATS Dashboard.

Prerequisites

The following should be installed on the computer before running the scanning tool:

- Linux redhat based OS (This work has been done under CentOS 7.4, however, there should be no reason to not work on other distributions)
- Python 3.9
- QT 4.1.0 based on 5.9.7. *Qt and its libraries*
- Tomoscan refer to: *Installing BEATS TomoScan*
- tmux

Clone and run the BEATS dashboard

The BEATS dashboard is available on github. The most recent version can be found on this link: https://github.com/SESAME-Synchrotron/BEATS_Dashboard.git. To clone and run, launch your terminal then do the follwoing:

```
$ cd /home/control/DAQ/operation
$ git clone git@github.com:SESAME-Synchrotron/BEATS_Dashboard.git
open and build the project in ``qtcreator``
$ cd /BEATS_Dashboard/BEATS_DAQ_Control_Monitor/
$ make distclean
$ qmake
$ make
$ BEATS_DAQ_Control_Monitor
```

Warning: If all is fine, you should see the GUI pops up; otherwise, an error occurred during the installation.

Note: Create the hosts names in `/etc/hosts` according to `Scripts/BEATS_GUI_Bash_Start`.

Note: You have to check and edit the directories and hosts in the `Scripts/` folder according to your environment.

Note: You have to check the `EPICS_CA_MAX_ARRAY_BYTES` and `EPICS_CA_ADDR_LIST` in `.bashrc` before starting the sacanning process.

BEATS Dashboard

This dashboard has been developed to help in the starting tomoscan scanning process by allowing the user to monitor and control everything associated with tomoscan.

To access the BEATS Dashboard, type the following command:

```
$ BEATS_DAQ_Control_Monitor
```

the main GUI will appear:

The BEATS Dashboard shown below is divided into three sections for controlling and five sections for monitoring:

Controlling Sections:

- 1) **Common IOCs: These are the mandatory EPICS BEATS IOCs for scanning.**
  -Shutter IOC. -Motor IOC. -TomoScan Support IOC. -Writer Support IOC. -SSCAN IOC.
- 2) **Detector Type: The available detectors for the scanning.**
  -Detector Status (indicate if the (hardware, Software) is connected or not (Power, Ethernet, IOC)).
  -Detector IOC. -Detector Driver.
- 3) **Scanning Methodology: The available scanning techniques for BEATS beamline.**

  -Step Scan:

  - TomoScan IOC.

Fig. 9: *Figure 1: BEATS Dashboard main window*

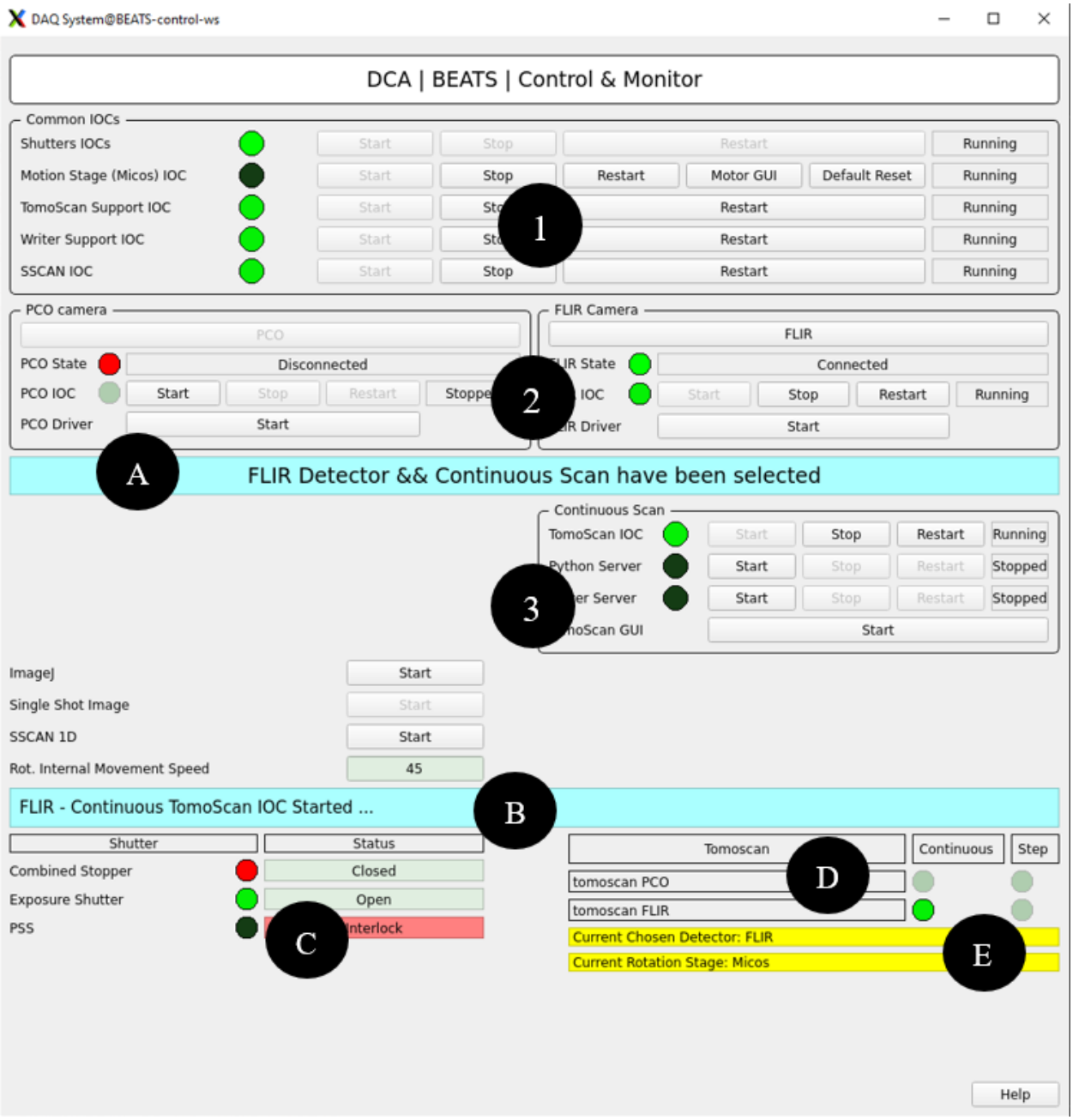


Fig. 10: *Figure 2: BEATS Dashboard sections*

- Python Server.
- writer Server.
- Filter Plugin Feature.
- MEDM (TomoScan MEDM)

-Continuous Scan:

- TomoScan IOC.
- Python Server.
- writer Server.
- MEDM (TomoScan MEDM)

Monitoring Sections:

- A) The detector type and the scanning technique are chosen.
- B) The online logging (last log).
- C) Shutters Status.
- D) Current tomoscan mode.
- E) The current detector chosen & current rotation stage.

Other Features

The following new features have been added to the BEATS dashboard:

1. ImageJ: starts imageJ viewer.
2. *Single Shot Image*.
3. Rot. Internal Movement Speed: the speed of the micos rotary stage that tomoscan set when go to max speed.

Selecting Process

The user has the option to choose the detector after opening the main window. Once the user has selected a detector, the *Current Chosen Detector* will display their selection. Following that, the user has the option to choose the scanning technique. Once the user has selected a scanning method (started the Tomoscan IOC), the *detector type and scanning technique* will display their choice. Additionally, as shown below, the other types of detectors will be disabled and the other scanning techniques hidden.

To change the detector type or scanning technique, the current process (TomoScan) must be stopped.

Note: All operations will be opened in tmux sessions, to attach any session, write the following commands:

```
$ tmux ls
$ tmux a -t "session name"
$ Esc, Ctrl b, d (to detach the session)
```

Fig. 11: *Figure 3: Selecting Process of Scanning Technique*

Warning: Make sure the TCPServerSocket.py is running on the server.

Warning: There is an interlocking between (Start, Stop, Restart) for all operations, depending on the status of the IOCs, whether they are running or not.

Warning: If one of the common IOCs is stopped (except SSCAN IOC), the other controlling sections will be disabled until all the common IOCs are running.

Warning: If the combined stopper shutter has a fault or the PSS is interlocked, the DAQ Tomoscan will be available only in *Testing Mode*.

Note: In the scanning techniques section, the python server (start button) is disabled until the tomoscan IOC is started.

Warning: There is an interlocking between the scanning techniques. This means that if any other scanning is started while the first one is still running, the first scanning will be automatically halted.

Warning: If the detector's IOC is stopped and you select any detector type, you cannot start the scan until the IOC is running.

Note: If the GUI is unexpectedly closed and then reopened, selecting one of the detectors will show the current choice if one of the other sections is hidden or disabled.

Single Shot Image

The fundamental idea behind a single shot image is to capture one or more frames based on the capturing type chosen. To begin this process, once opened, its features will be disabled as shown in the figure.4, and you must type the detector's prefix (TEST-PCO: or FLIR:) to be able to proceed as shown in the figure.5.

Note: The Single Shot Image main window button will be disabled if any tomoscan mode is running.

The redout and collect sections, which contain the detector's parameters, become active once you type the prefix.

The available capture modes are as follows:

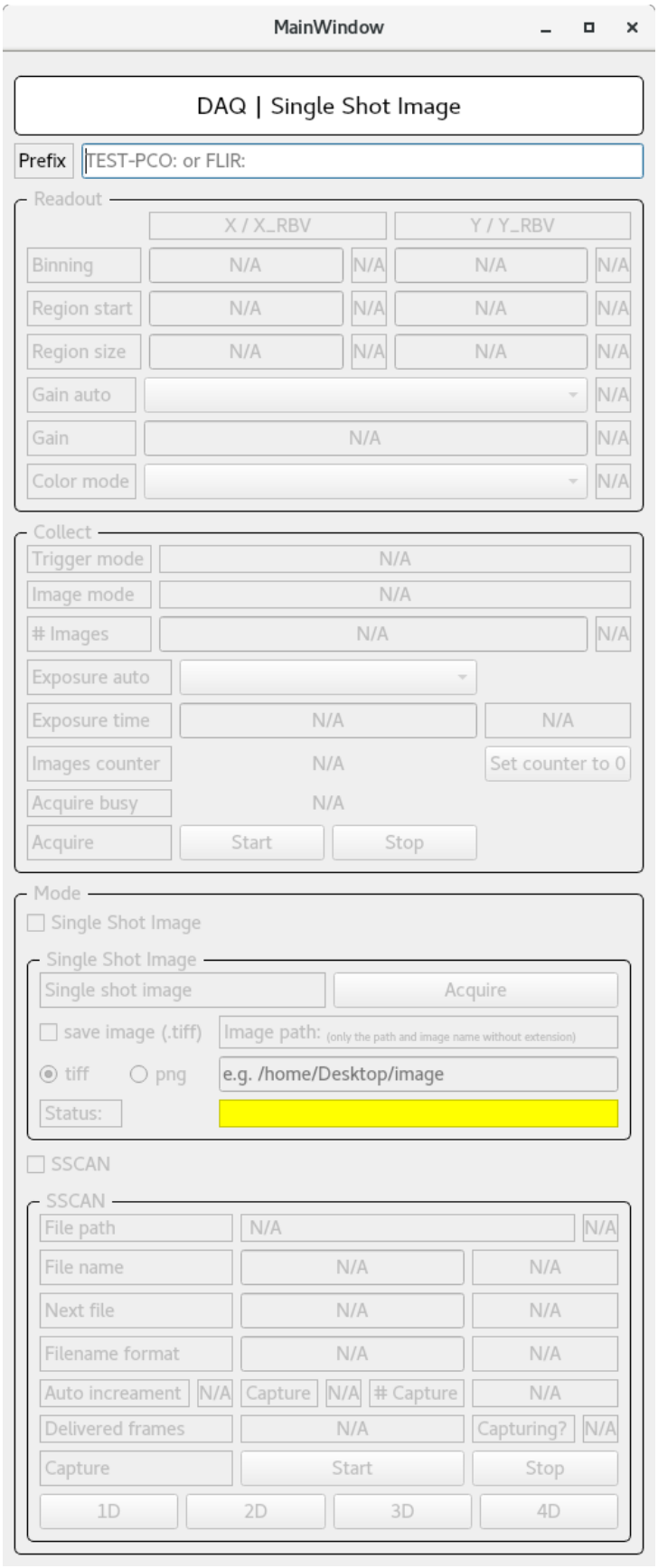


Fig. 12: *Figure 4: Single Shot Image -Main Window-*

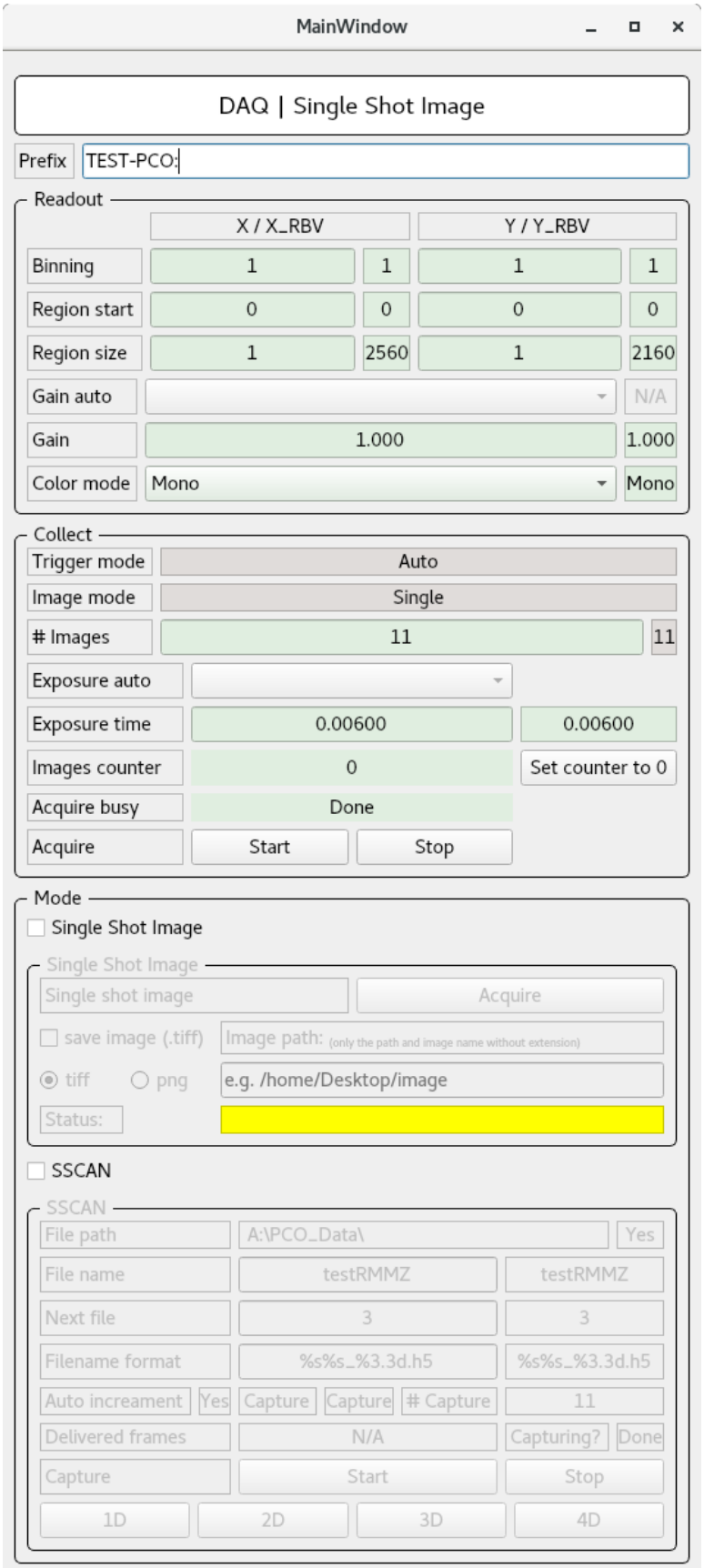


Fig. 13: *Figure 5: Single Shot Image -Available Prefixes-*

1. Single Image Acquiring

The idea behind this mode is to open the exposure shutter, take one shot, and then close the exposure shutter.

Both clicking the "Acquire" button or using the "Space" key will initiate the acquisition process. The image can also be saved (TIFF or PNG format).

Note: To save the image, you have to determine the path and define the image name only *without any extension*. Moreover you will be alerted if the path is not valid.

Note: The acquiring process is shown in the main terminal as figure below. Moreover, the *Status yellow field* shows the last log.

```
open exposure shutter...
191050
191051
close exposure shutter
reshape image to 2560 X 2160
image reshaped!
saving image to /home/control/DAQ/test.tiff
image /home/control/DAQ/test.tiff saved!
```

Fig. 15: *Figure 7: Single Shot Image -Acquiring Process-*

---

2. SSCAN

The idea behind this mode is to collect multiple images for each motion step. more info can be found here: SSCAN reference

The main *write fields* parameters of SSCAN section are:

- File name
- File format (the main format is h5 file)
- Next file number

The figure below will appear after clicking on the desired SSCAN dimension; you can start SSCAN up to 4 dimensions.

Note: The trigger PVs to start acquiring for both detectors are: - for PCO: TEST-PCO:cam1:Acquire - for FLIR: FLIR:cam1:Acquire

---

Note: Very Important! You must ensure that the data from the detector are gathered; the value for the *Capturing?* field should be (Capture yellow Colored instead of Done).

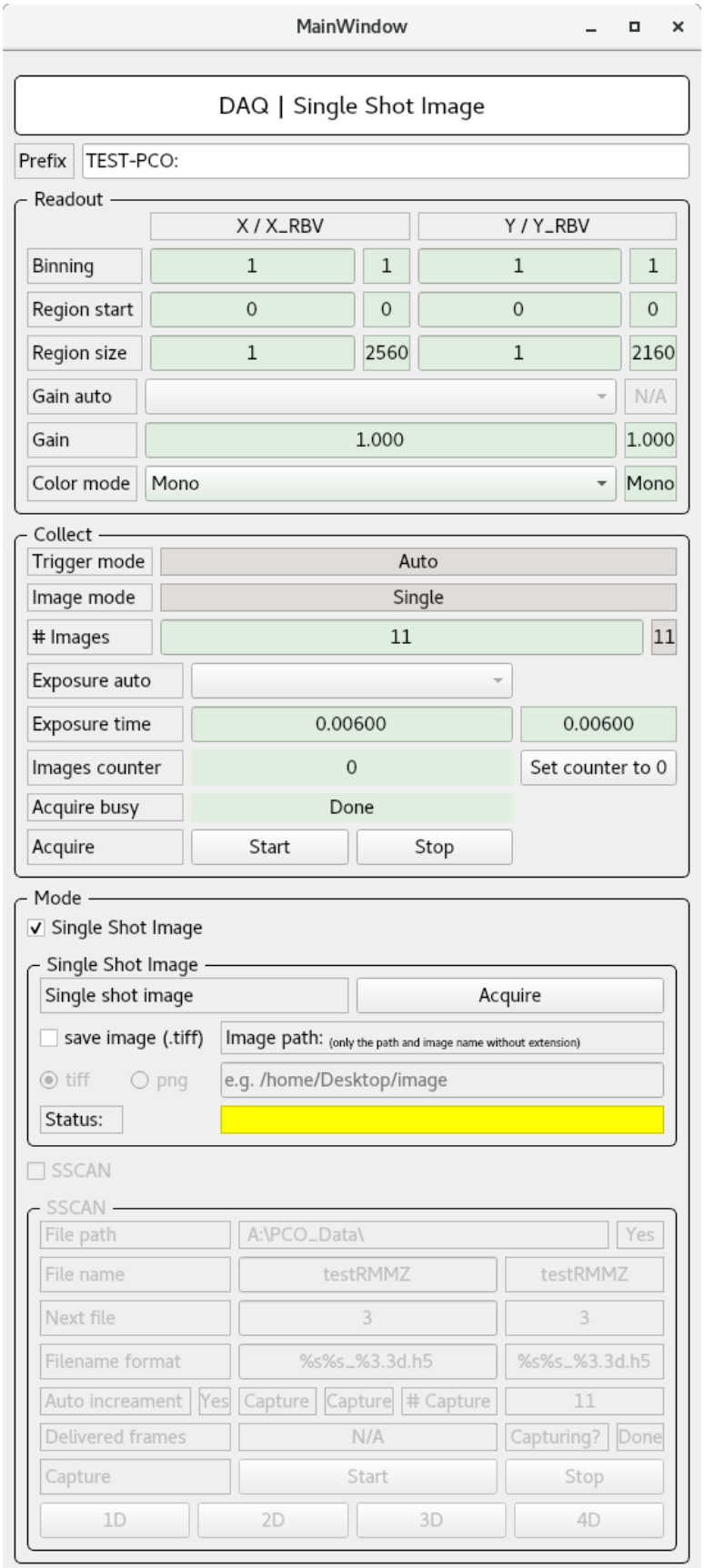


Fig. 14: *Figure 6: Single Shot Image -Main Parameters-*

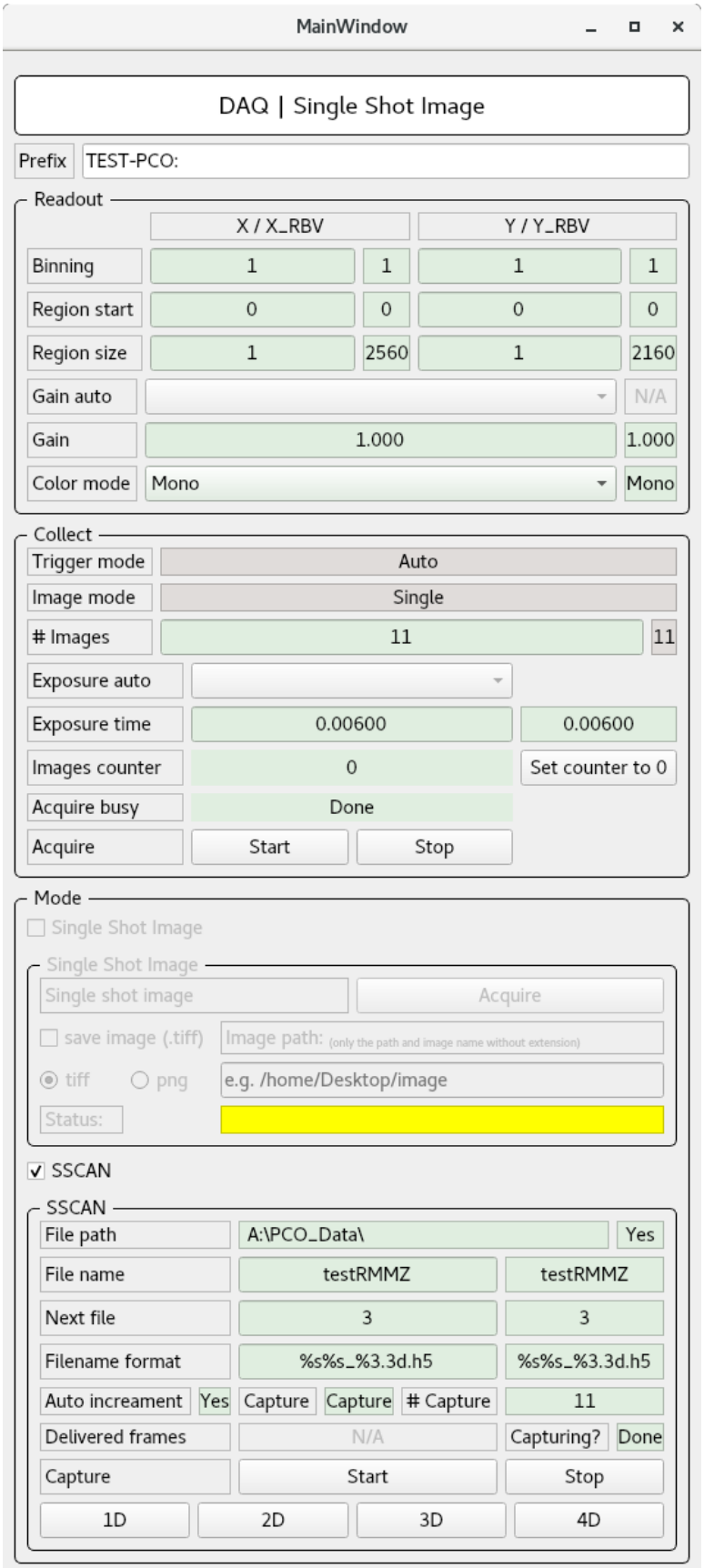


Fig. 16: *Figure 8: SSCAN -Main Parameters-*

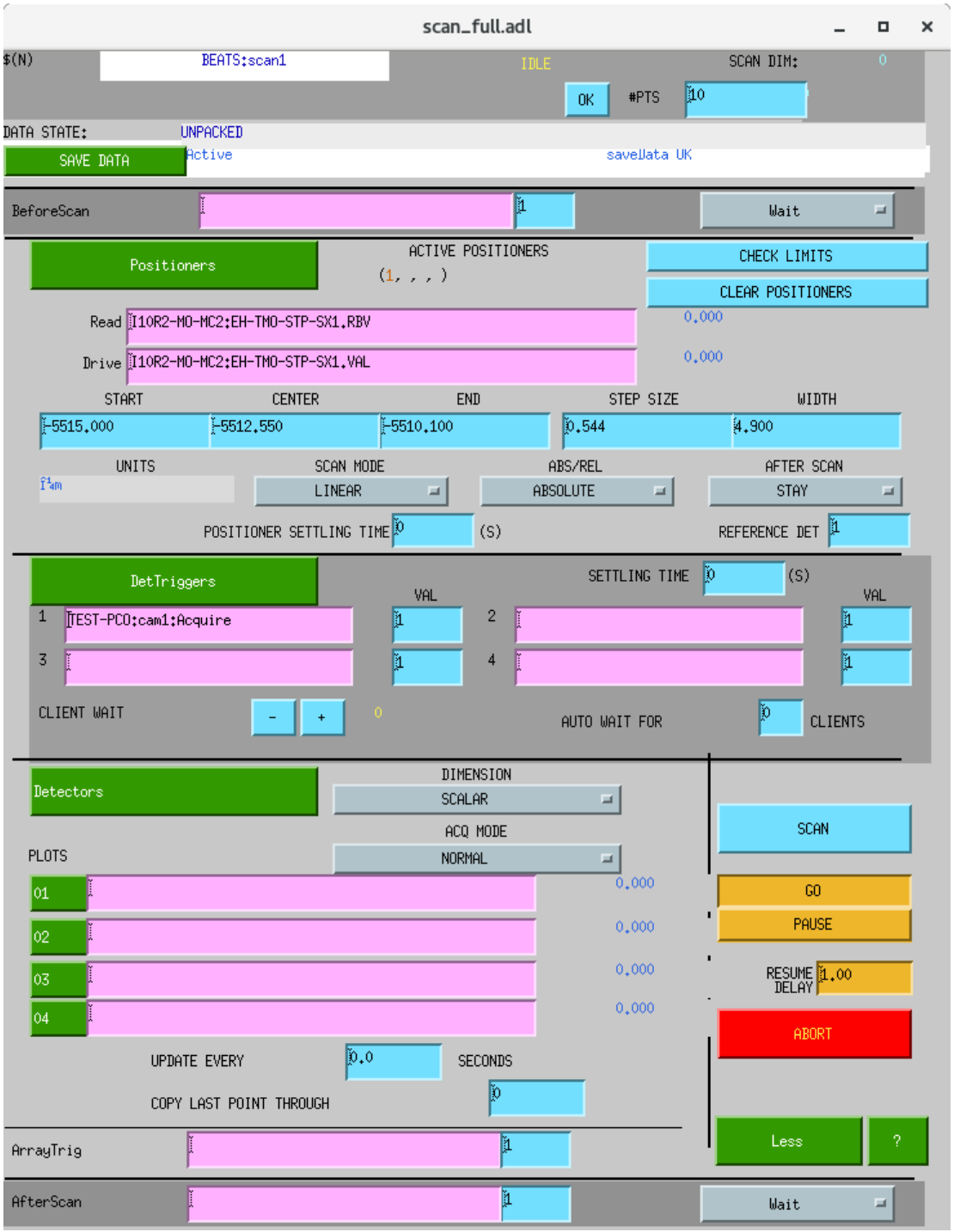


Fig. 17: *Figure 9: SSCAN -Main Window-*

Note: The file extension of SSCAN outout is binary format (.mda), to read it you have to convert it to txt file.

```
cd /home/control/Desktop/SSCAN_Data
./SSCAN -f (file name)  e.g. ./SSCAN -f test1.mda
```

Note: The data of SSCAN (output h5 files) are located in shared folder.

```
cd /home/control/Desktop/SSCAN
cd PCO_Data or FLIR_Data
```

1.6 Tomographic reconstruction at BEATS

This section contains examples of tomographic reconstruction pipelines using TomoPy.

Note: Reconstruction pipelines at BEATS can be also executed from the command line, or as Slurm jobs on `rum@sesame.org.jo`. The header of each example shows the shell command to launch the pipeline as a script. Visit the page *BEATS Computing Infrastructure* for more information.

1.6.1 *BEATS CT reconstruction pipeline - Paganin phase retrieval*

Minimal TomoPy reconstruction pipeline for **phase contrast** data collected at the BEATS beamline of SESAME.

Created on: 23.05.2021 Last update: 09.10.2023

- By: Gianluca Iori, 2023
- Code license: MIT
- Narrative license: CC-BY-NC-SA

Type `Ctrl` + `Enter` on a single cell to run it.

Load experiment data

Enter the **sample_name** and the correct **output_dir**

```
sample_name = "bee_yazeed-20231001T170032"
work_dir = "/mnt/PETRA/SED/BEATS/IH/"+sample_name
h5file = work_dir+"/"+sample_name+".h5"

output_dir = "/mnt/PETRA/SED/BEATS/IH/scratch/tmp/"
recon_dir = output_dir+sample_name+"/recon/"
cor_dir = output_dir+sample_name+"/cor/"
```

Load the complete dataset (or)

```
# projs, flats, darks, theta = dxchange.read_aps_32id(h5file, exchange_rank=0)
```

Read a portion of the dataset

- `sino` controls the vertical detector lines to read - `sino=(1360, 2160, 1)`
- `proj` defines the range of projections - `proj=(1, 1001, 1)`

```
sino=(1360, 2160, 1)
proj=(1, 1001, 1)
```

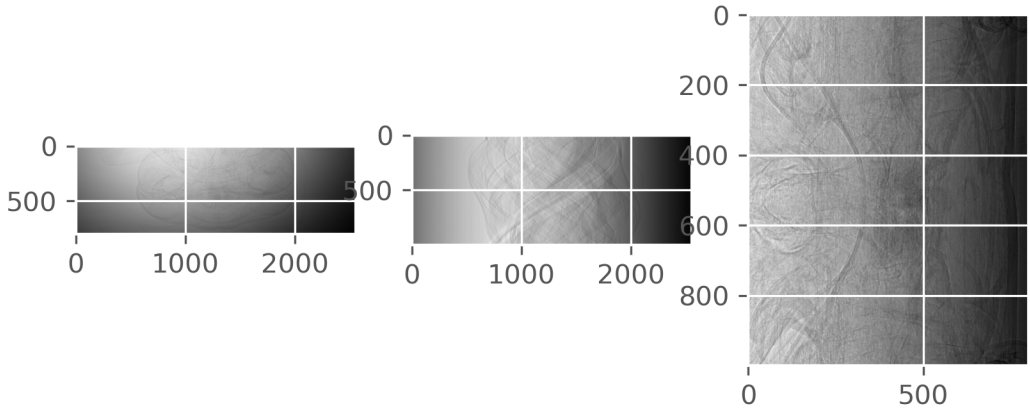
```
projs, flats, darks, _ = dxchange.read_aps_32id(h5file, exchange_rank=0, sino=sino,␣
↪proj=proj)
theta = np.radians(dxchange.read_hdf5(h5file, 'exchange/theta', slc=(proj,)))

print("Dataset size: ", projs[:, :, :].shape[:], " - dtype: ", projs.dtype)
print("Flat fields size: ", flats[:, :, :].shape[:])
print("Dark fields size: ", darks[:, :, :].shape[:])
print("Theta array size: ", theta.shape[:])
```

```
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/bee_yazeed-
↪20231001T170032/bee_yazeed-20231001T170032.h5
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/bee_yazeed-
↪20231001T170032/bee_yazeed-20231001T170032.h5
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/bee_yazeed-
↪20231001T170032/bee_yazeed-20231001T170032.h5
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/bee_yazeed-
↪20231001T170032/bee_yazeed-20231001T170032.h5
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/bee_yazeed-
↪20231001T170032/bee_yazeed-20231001T170032.h5
```

```
Dataset size:  (1000, 800, 2560)  - dtype:  uint16
Flat fields size:  (102, 800, 2560)
Dark fields size:  (102, 800, 2560)
Theta array size:  (1000,)
```

At any time you can take a look at your 3D data with `ru.plotmidplanes(data)`

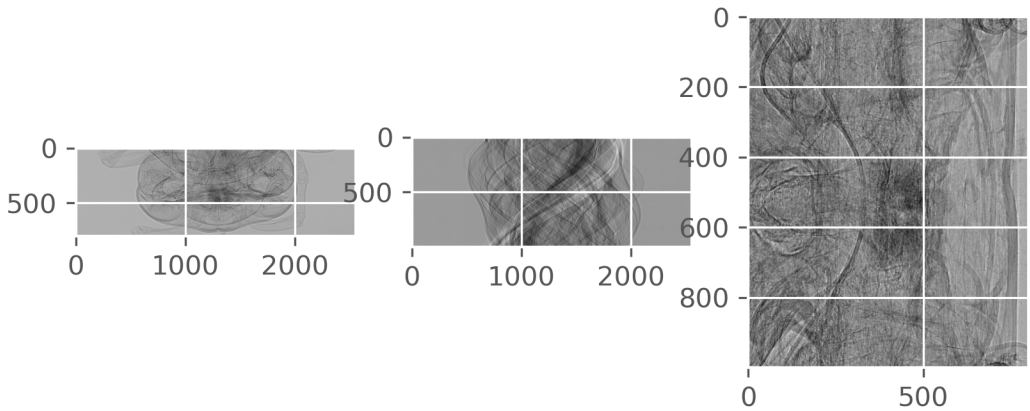
```
ru.plot_midplanes(projs)
```



Flat field correction

Normalize the image background.

```
projs = tomopy.normalize(projs, flats, darks, ncore=ncore, averaging='median')
ru.plot_midplanes(projs)
```



Phase retrieval

```
delta_beta = 250 # ratio between real and imaginary part of the refractive index
alpha=1./(4*3.141592**2 * delta_beta)
print("alpha: ", alpha)
```

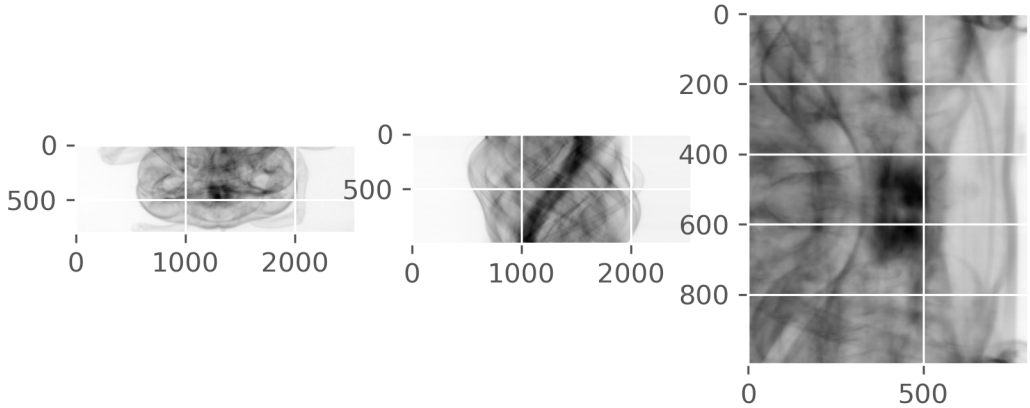
```
alpha:  0.00010132122580089835
```

```
projs_phase = tomopy.retrieve_phase(projs[:, :, :],
                                    pixel_size=1e-4*0.65,
                                    dist=2,
                                    energy=10,
                                    alpha=alpha,
                                    ncore=ncore,
```

(continues on next page)

(continued from previous page)

```
                                        pad=True)
# nchunk=None,

ru.plot_midplanes(projs_phase)
```



Center Of Rotation (COR)

Save images reconstructed with COR range

```
cor_range = [1260, 1300, 1]
```

```
tomopy.write_center(projs, theta, cor_dir, cor_range)
```

```
INFO:tomopy.recon.algorithm:Reconstructing 40 slice groups with 36 master threads...
```

View them in Fiji

```
# os.system(Fiji_exe_stack + cor_dir+'{:04.2f}'.format(COR[0])+'.tiff &')
```

Manually insert the best COR

```
COR = 1301
```

Automatic detect COR

```
# COR = tomopy.find_center(projs, theta, init=projs.shape[2]/2, tol=0.5) # ind=200,
# print("Automatic detected COR: ", COR, " - tomopy.find_center")

COR = tomopy.find_center_vo(projs, ncore=ncore)
print("Automatic detected COR: ", COR, " - tomopy.find_center_vo")
```

```
Automatic detected COR:  1301.5  - tomopy.find_center_vo
```

Reconstruction

GPU reconstruction with the ASTRA toolbox

Algorithm	fbp CUDA ASTRA

```
options = {'proj_type': 'cuda', 'method': 'FBP_CUDA'}

recon = tomopy.recon(projs_phase,
                     theta,
                     center=COR,
                     algorithm=tomopy.astra,
                     options=options,
                     ncore=1)
```

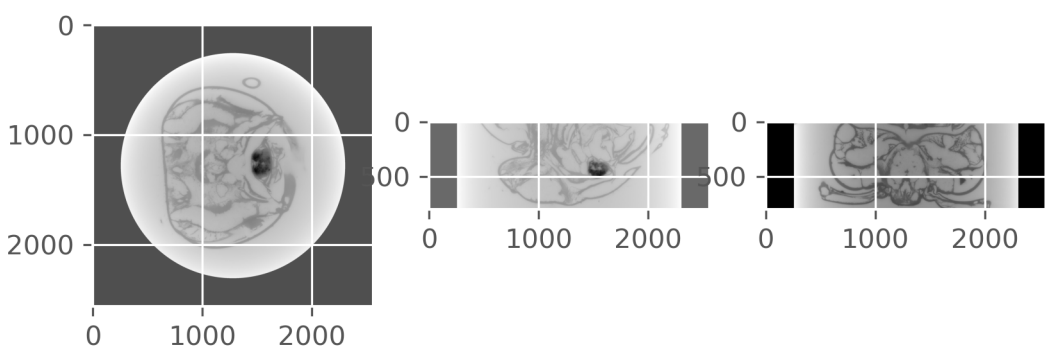
```
INFO:tomopy.recon.algorithm:Reconstructing 1 slice groups with 1 master threads...
```

Post-processing

Apply circular mask

```
recon = tomopy.circ_mask(recon,
                         axis=0,
                         ratio=0.8,
                         ncore=ncore)
```

```
ru.plot_midplanes(recon)
```



Write reconstructed dataset

Write output tiff stack as `float32`

```
fileout = recon_dir+'slice.tiff'
dxchange.writer.write_tiff_stack(recon, fname=fileout, axis=0, digit=4, start=0,␣
↪overwrite=True)
```

Open virtual stack in ImageJ

```
# os.system(Fiji_exe_stack + fileout + ' &')
```

1.6.2 Inspect 3D dataset with napari

napari is a Python library for n-dimensional image visualisation, annotation, and analysis.

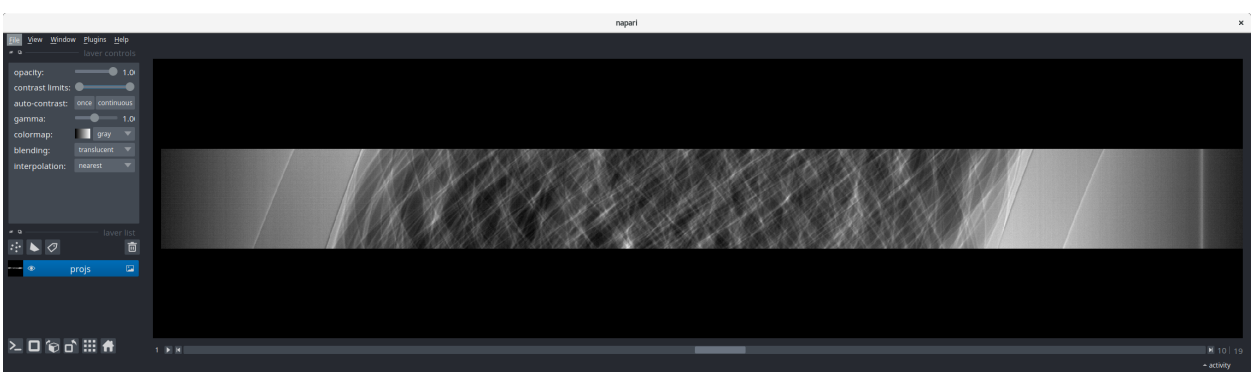
Load `HDF5` dataset

```
h5file = "/mnt/PETRA/SED/BEATS/IH/BEATS_first_scan-20230511T170626/BEATS_first_scan-
↪20230511T170626.h5"
projs, _, _, _ = dxchange.read_aps_32id(h5file, sino=(800, 820, 1), proj=(500, 700, 1),␣
↪exchange_rank=0)
```

```
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/BEATS_first_
↪scan-20230511T170626/BEATS_first_scan-20230511T170626.h5
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/BEATS_first_
↪scan-20230511T170626/BEATS_first_scan-20230511T170626.h5
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/BEATS_first_
↪scan-20230511T170626/BEATS_first_scan-20230511T170626.h5
ERROR:dxchange.reader:Unrecognized hdf5 dataset: "exchange/theta"
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/BEATS_first_
↪scan-20230511T170626/BEATS_first_scan-20230511T170626.h5
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/BEATS_first_
↪scan-20230511T170626/BEATS_first_scan-20230511T170626.h5
INFO:dxchange.reader:Data successfully imported: /mnt/PETRA/SED/BEATS/IH/BEATS_first_
↪scan-20230511T170626/BEATS_first_scan-20230511T170626.h5
WARNING:dxchange.exchange:num_angles(2001) is not the same as tomo.shape[0](200)
```

```
import napari
```

```
viewer = napari.view_image(projs)
```



1.6.3 Read theta array from encoder readout

- Angular information (theta array) is required for accurate tomographic reconstruction.
- If theta is not stored in the raw data, it is possible to simulate it as equally spaced between 0 and 180 degrees.
- Alternatively, we can read the rotation stage encoder readout during the scan (beta)

A. Load `theta` array stored in the `HDF5` dataset

This is a calculated array of the ideal angles during the scan.

```
h5file = "/mnt/PETRA/SED/BEATS/IH/BEATS_first_scan-20230511T170626/BEATS_first_scan-
↪20230511T170626.h5"
projs, _, _, theta = dxchange.read_aps_32id(h5file, exchange_rank=0, sino=(695, 705, 1))
```

```
print("Theta array size: ", theta.shape[:])
```

B. Simulate `theta` array

If `theta` is empty, we can retrieve it using the `tomopy.angles` method:

```
theta_tomopy = tomopy.angles(projs.shape[0])
```

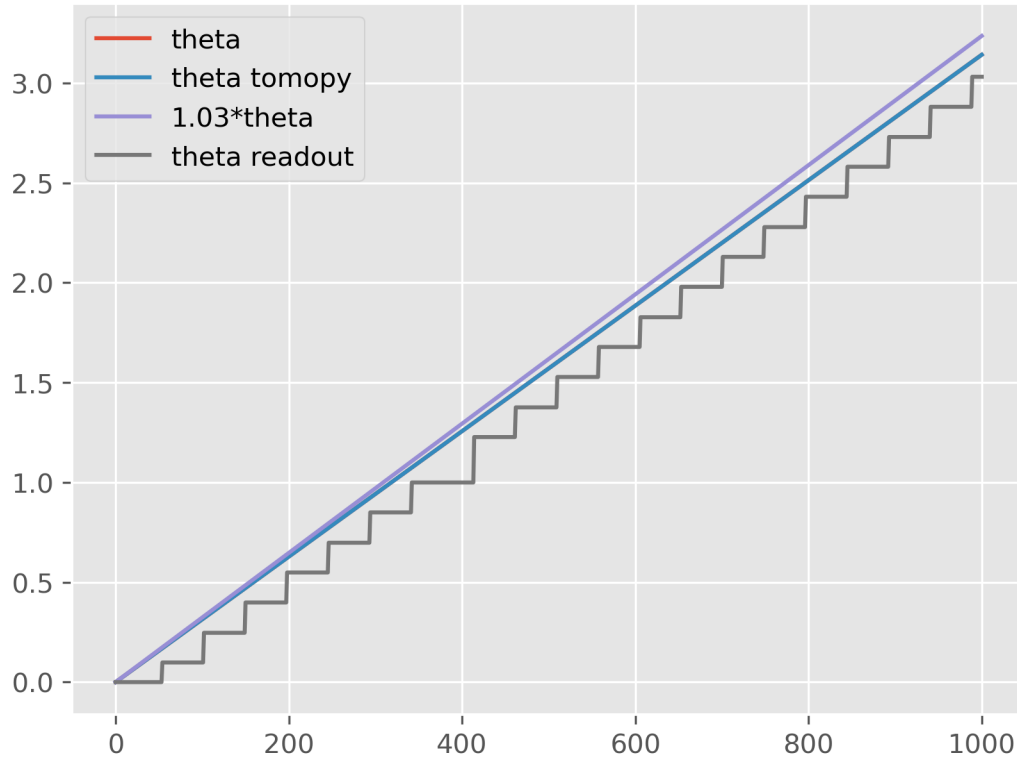
C. Read `theta_readout` from encoder

Another option is to read the angle readout from the rotation stage:

```
theta_readout = dxchange.read_hdf5(h5file, '/exchange/theta_readout')
```

```
plt.plot(theta, label='theta')
plt.plot(theta_tomopy, label='theta tomopy')
# plt.plot(1.03*theta, label='1.03*theta')
plt.plot(theta_readout/180*np.pi, label='theta readout')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7fcf106c5110>
```



1.6.4 Convert 360-degrees extended Field Of View scan to standard 180-degree

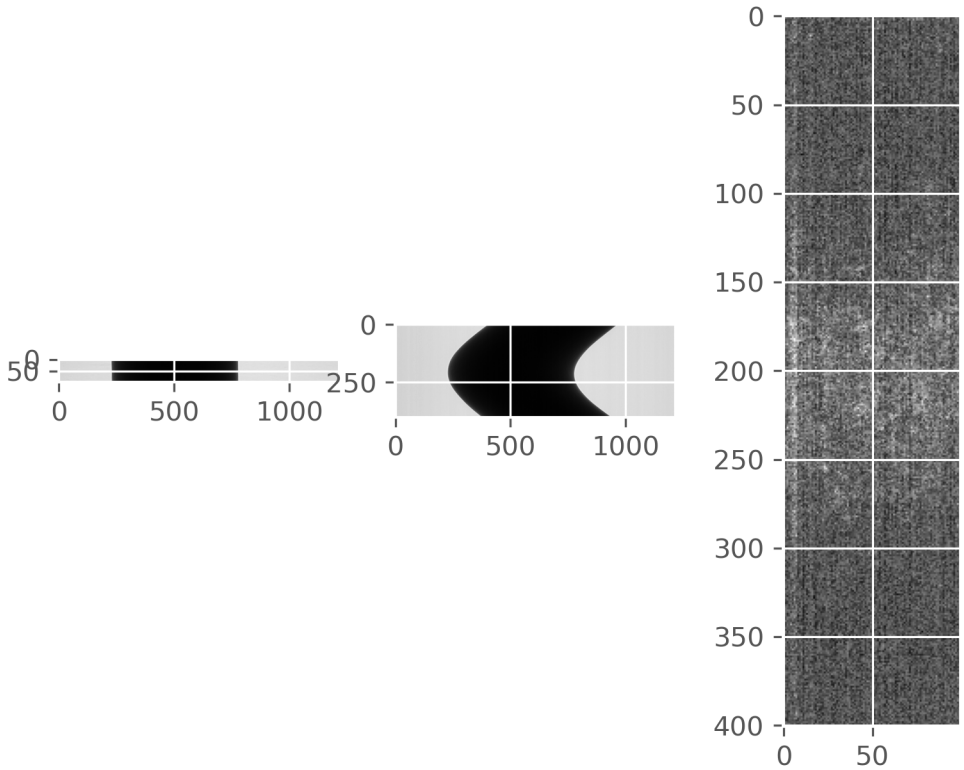
Visit the `tomopy.sino_360_to_180` for more information.

Load `HDF5` dataset of extended FOV scan

```
h5file = "/mnt/PETRA/SED/BEATS/IH/BEATS_first_scan-20230511T170626/BEATS_first_scan-
↪20230511T170626.h5"
projs, flats, darks, theta = dxchange.read_aps_32id(h5file, exchange_rank=0, sino=(600,␣
↪800, 1))
```

```
print("Dataset size: ", projs[:, :, :].shape[:], " - dtype: ", projs.dtype)
print("Flat fields size: ", flats[:, :, :].shape[:])
print("Dark fields size: ", darks[:, :, :].shape[:])
print("Theta array size: ", theta.shape[:])
```

```
ru.plot_midplanes(projs)
```

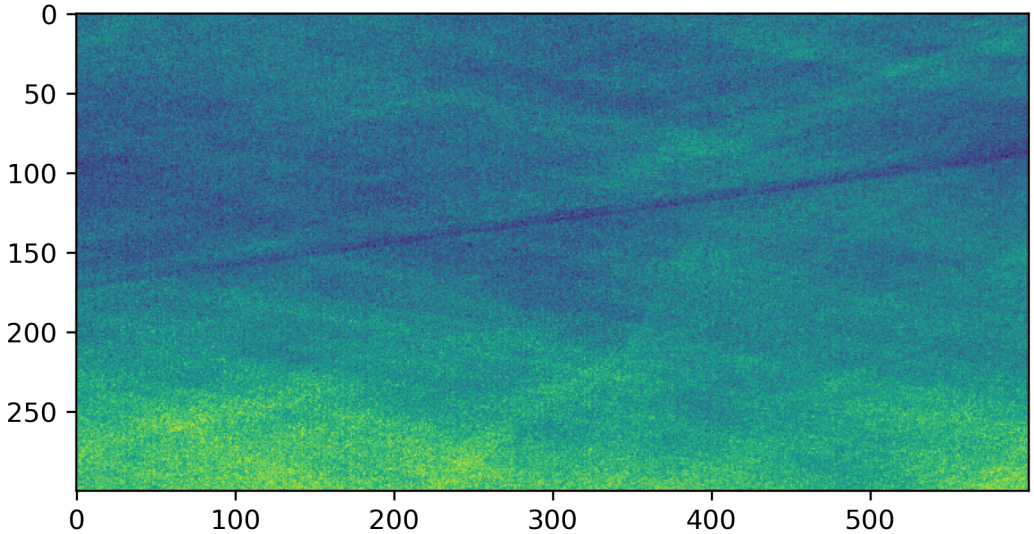



```
projs_180 = tomopy.sino_360_to_180(projs, overlap=600)
```

- Check for continuity of sinogram features.
- Fine-tune the `overlap` parameter.
- The correctly processed sinogram shows no edge but only continuous lines

```
plt.imshow(projs_180[600:900, 600, 2700:3300])
```

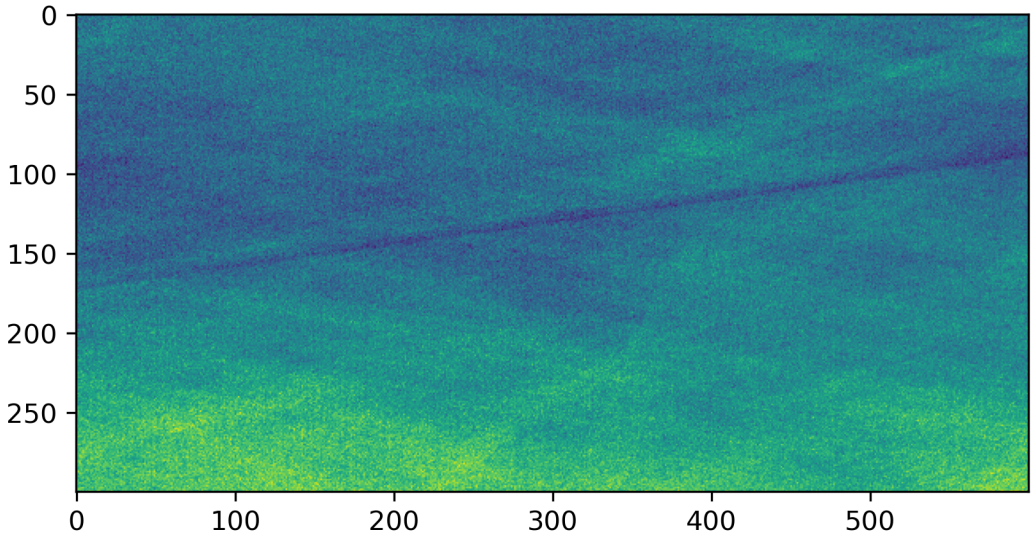
```
<matplotlib.image.AxesImage at 0x7fbfc88121d0>
```




```
projs_180 = tomopy.sino_360_to_180(projs, overlap=700)
```

```
plt.imshow(projs_180[600:900, 600, 2700:3300])
```

```
<matplotlib.image.AxesImage at 0x7fbfc88121d0>
```



1.7 Safety

This section contains basic safety information required to run your experiments at the beamline. At the start of your beamtime, you will receive a general safety training from the SESAME safety office.

1.7.1 Radiation protection

Radiation shielding assures the limitation of the exposure to ionizing radiation of all SESAME staff members and visitors to values legally defined by the Local Jordanian Law of Radiation Protection (Nuclear Safety and Security Law No. 43 for the year 2007 and its amendments). The expected dose received by staff members and users must be less than the dose limit for the general public for 2000 working hours. This implies shielding measures limiting the dose rate to values below a maximum of 0.5 µSv/h under normal operation conditions.

Warning: It is strictly prohibited to move, open, make holes, or modify in any way the beamline radiation shielding (i.e. lead, tungsten, polyethylene panels).

Fig. 18: *Radiation shielding chicanes and radiation monitor at BEATS. Modifying, moving or changing in any way the radiation shield as well as any other safety equipment is strictly forbidden.*

1.8 Beamline hardware

This section is under construction..

1.8.1 3-pole wiggler

1.8.2 Double Multilayer Monochromator

1.8.3 Tomography endstation 1

1.8.4 Detectors

Table of the detectors installed at the beamline: click here.

The table lists all detectors available at the beamline and shows the magnification, pixelsize and Field Of View (FOV) obtained with each combination of detector and camera. The second sheet contains a calculator of the optimal scintillator thickness.

1.8.5 Cameras

ToDo: Create a documentation for changing cameras and optics/magnifications.

1.9 Documentation of procedures

The intention of this section is to enable new beamline staff or trainees as quickly as possible to work autonomously at/with the beamline. Non-routine tasks.

ToDo: Create a documentation for changing cameras and optics/magnifications.

Put in filters and observe the beam for the first time. -> adjustment of the beam for the experiment.

1.9.1 alignment too

Here should be another reference *sample alignment*

tomoalign

1.9.2 after shutdown

- do search of the optics hutch
- are all chillers and other devices turned on (e.g. the chiller of the monochromator), are all symbols in the dashboad green? If everything seems ok click reset to refresh them and see if the shutters can be opened.
- open the gauge valves
- check with Yazeed if any motors need to be homed, especially if there was a power cut
- experimental hutch: CVD Window 2: water flow for cooling should be >= 3.5, below 3 it gives an interlock

1.9.3 Changing cameras

1.9.4 Selecting detector, start of IOC, getting initial images

FLIR needs to be plugged in PCO needs to be switched on on top, its chiller is in "follow-mode".

1.9.5 Refining slit settings and detector posotion

1.9.6 Potential obstacles

could be that wire scanner, diagnostics screen, white beam blocker (WBB), CVD window are "in the beam" and give artefacts

1.9.7 Change radiation and energy settings

The beamline energy can be changed using the Energy CLI.

Start the Energy CLI:

```
cd /home/control/energy/iocBoot/iocEnergy_2BM
python3 -i start_energy.py
```

Set energy to 20.0 keV (W/B4C stripe):

```
energy set --energy 20
```

Set energy to 20.0 keV (Ru/B4C stripe):

```
energy set --energy 19.99
```

— maybe list more settings + filtered radiation?

1.9.8 changing detectors

Videos + photos

How to switch on the cameras. : plug in power supply or simply switch on

increase distance between sample and detector to e.g. 1m

Det1/Det3 - optique peter:
remove lead box untighten small crews place the camera in a safe place (e.g. a hanging, CLEAN bag) connections

Det3 (poly radiation): connect all five available cables Det3 (mono radiation with objectives): connect Foc1/Rot1

Det2 - Hasselblad:
open black housing (heavy, perhaps use the crane)

1.9.9 changing optics/magnification

Videos + photos

1.9.10 cleaning the scintillator

1.9.11 Mount proposal folders

Warning: The following commands are for the beamline staff only.

Mount proposal `ExpData` and `recon` folders on BL-BEATS-WS01:

```
cd ~
./petra_prop_mounter.sh
```

Check mount points:

```
df -h
```

Unmount proposal folders:

```
umount /PETRA/SED/BEATS/SEM_6/20235010
umount /PETRA/SED/BEATS/SEM_6_recon/20235010
```

Mount proposal `ExpData` and `recon` folders on Win Data Dispenser and Dragonfly VizServer:

```
./petra_prop_recon_smb_mounter.sh
```

Note: For proposals belonging to a different semester the scripts `petra_prop_mounter.sh` and `petra_prop_recon_smb_mounter.sh` must be modified.

1.9.12 Endstation alignment

Note: The endstation is aligned by the beamline staff at the start of your beamtime. Generally, you don't need to repeat these operation and you can jump to *sample alignment*

1. Endstation pitch
2. Endstation X-axis
3. Camera rotation
4. Detector focus

1.10 BEATS Computing Infrastructure

1.10.1 DAQ workstation - BEATS-control-ws

The data acquisition (DAQ) workstation is used to control the beamline and scan settings. Visit the section *Data Acquisition (DAQ)* for more information.

Useful commands

Mount PETRA:

```
sudo mount -t nfs 10.1.14.100:/PETRA/SED/BEATS/IH /PETRA/SED/BEATS/IH
```

SMB mount of SSCAN data folder:

```
sudo mount -t cifs -o vers=3,username=beats.smb '\\10.1.14.100\pco-flir-ws' /home/
↪control/Desktop/SSCAN
```

Start the Energy GUI:

```
cd /home/control/energy/iocBoot/iocEnergy_2BM
python3 -i start_energy.py
```

1.10.2 Data analysis workstation - BL-BEATS-WS01

The data analysis workstation is used for several purposes including:

- Inspection of sinograms and CT reconstruction
- Submit reconstruction jobs on the cluster `rum@sesame.org.jo`
- 3D image visualization and processing

The list of software available on the workstation is listed in the section on :ref:'Data analysis software' below.

Useful commands

Start alrecon CT reconstruction environment:

```
conda activate tomopy
solara run alrecon.pages --host localhost
```

Start reconstruction pipeline on Jupyter Lab. Available pipelines are described in section *Tomographic reconstruction at BEATS*:

```
conda activate tomopy
jupyter lab
```

Mount proposal `ExpData` and `recon` folders on BL-BEATS-WS01:

```
cd ~
./petra_prop_mounter.sh
```

Check mount points:

```
df -h
```

Unmount proposal folders:

```
umount /PETRA/SED/BEATS/SEM_6/20235010
umount /PETRA/SED/BEATS/SEM_6_recon/20235010
```

Mount proposal `ExpData` and `recon` folders on Win Data Dispenser and Dragonfly VizServer:

```
./petra_prop_recon_smb_mounter.sh
```

Note: For proposals belonging to a different semester the scripts `petra_prop_mounter.sh` and `petra_prop_recon_smb_mounter.sh` must be modified.

1.10.3 Data analysis software

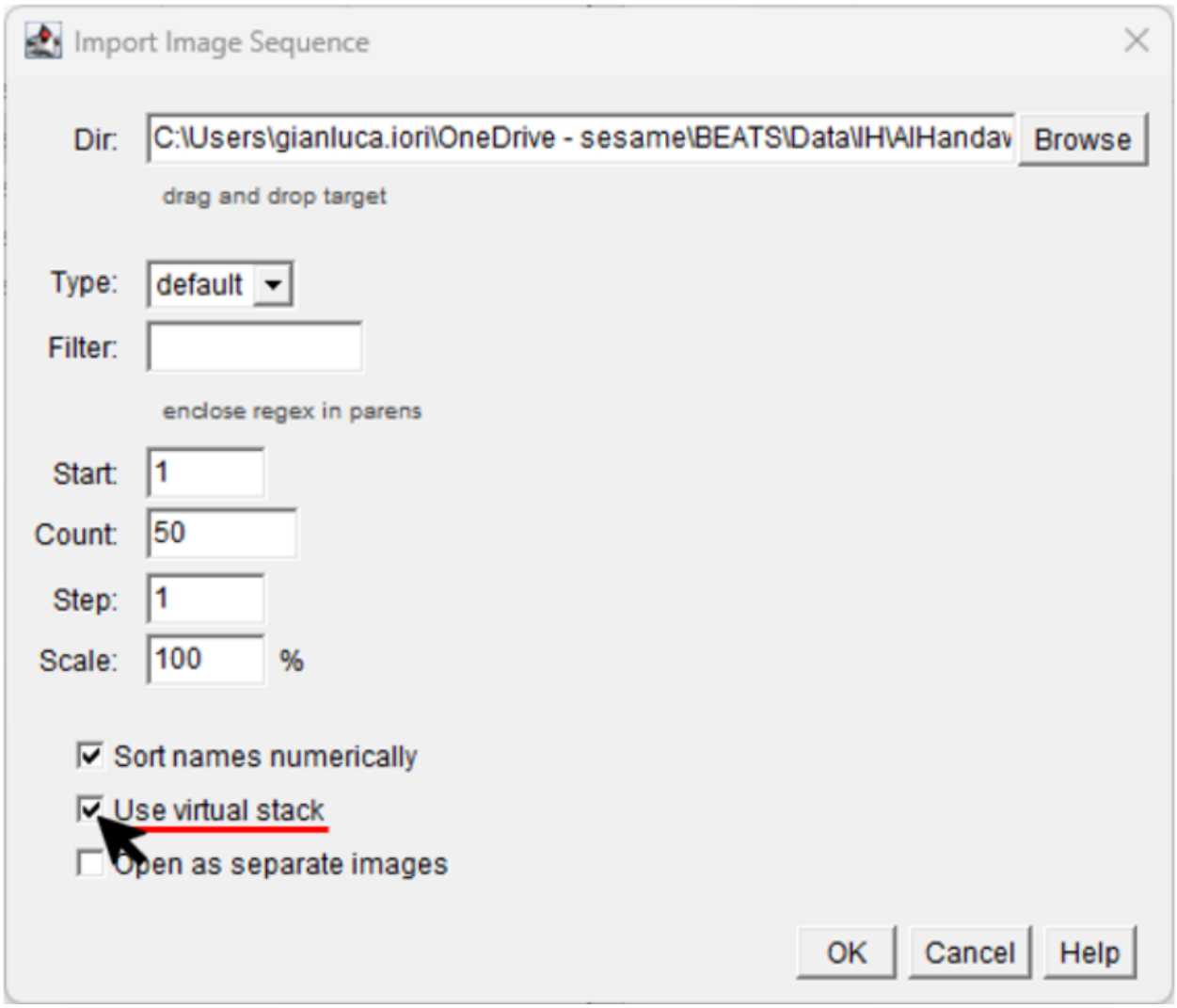
The software in the table below can be used to inspect and process 3D image data (sinograms and CT reconstructions) at SESAME BEATS.

Name	URL	Open source	Features
ImageJ	https://fiji.sc/	yes	Essential for data collection and reconstruction
Paraview	https://www.paraview.org/	yes	3D image rendering
Dragonfly	https://www.theobjects.com/dragonfly/index.html	no	3D image analysis and visualization
3D Slicer	https://www.slicer.org/	yes	3D image analysis and visualization
TomoPy	https://tomopy.readthedocs.io/en/stable/	yes	CT reconstruction in Python
Alrecon	https://github.com/gianthk/alrecon/tree/master	yes	Web app for CT reconstruction
Jupyter	https://jupyter.org/	yes	Interface for Python reconstruction pipelines (notebooks)

Load reconstructed volume with ImageJ

Reconstructions at SESAME BEATS are generally saved as a stack of `.TIFF` images contained in a reconstruction folder. To load a reconstruction in ImageJ use the command `File › Import › Image Sequence`. You can follow this video for a detailed explanation on how to import image sequences.

Note: Always select the option `Use Virtual Stack` when you import large image stacks in ImageJ!


Import Image Sequence
Dir:
C:\Users\gianluca.iori\OneDrive - sesame\BEATS\Data\IH\AIHandav
Browse
drag and drop target
Type:
default
Filter:
enclose regex in parens
Start:
1
Count:
50
Step:
1
Scale:
100
%
Sort names numerically
Use virtual stack
Open as separate images
OK
Cancel
Help

1.10.4 rum - BEATS reconstruction cluster

Access the reconstruction cluster `rum@sesame.org.jo` with:

```
ssh -X beatsbs@rum.sesame.org.jo
```

1.10.5 Data dispenser PC

1.10.6 Dragonfly VizServer

1.10.7 SESAME data portal