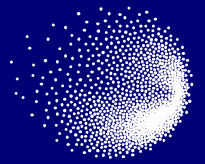


PSI Center for Photon Science

The *Pixelator* Instrument Control System

Benjamin Watts
PolLux Beamline Scientist

The Pixelator Project

- History
- Structure:
  - Controller
  - Realtime
  - GUI options
  - Scripting
- Scan implementation details:
  - Interferometric position metrology
  - Scan modes and patterns
  - Monitors and further data sources
- NeXus data files
- Future developments

Pixelator History

2006: PolLux STXM installed at SLS

- ALS design purchased from Accel (now Axilon, sort of)
- Controlled via *STXM Control* by Tolek Tyliszczak

2010: Begin *Pixelator* project to replace *STXM Control*

- Collaboration agreement with *Max-Planck-Institut für Intelligente Systeme*

2011: Contract *Semafor Informatik & Energie AG* to write most of the code

2012: Begin "production" use of *Pixelator* at PolLux (and writing *NeXus/NXstxm* files)

2014: *NXstxm* definition ratified by *NeXus*

2014: Implemented at I08@Diamond (since switched to *GDA*)

2015: Implemented at HERMES@Soleil

2023: Switch to *Orocos* realtime system (previously *Orchestra*)

2023: Implemented at SOPHIE@SLS (Stationed at MAXIV 2024+2025)

2023: Implemented at MYSTIIC@BESSYII

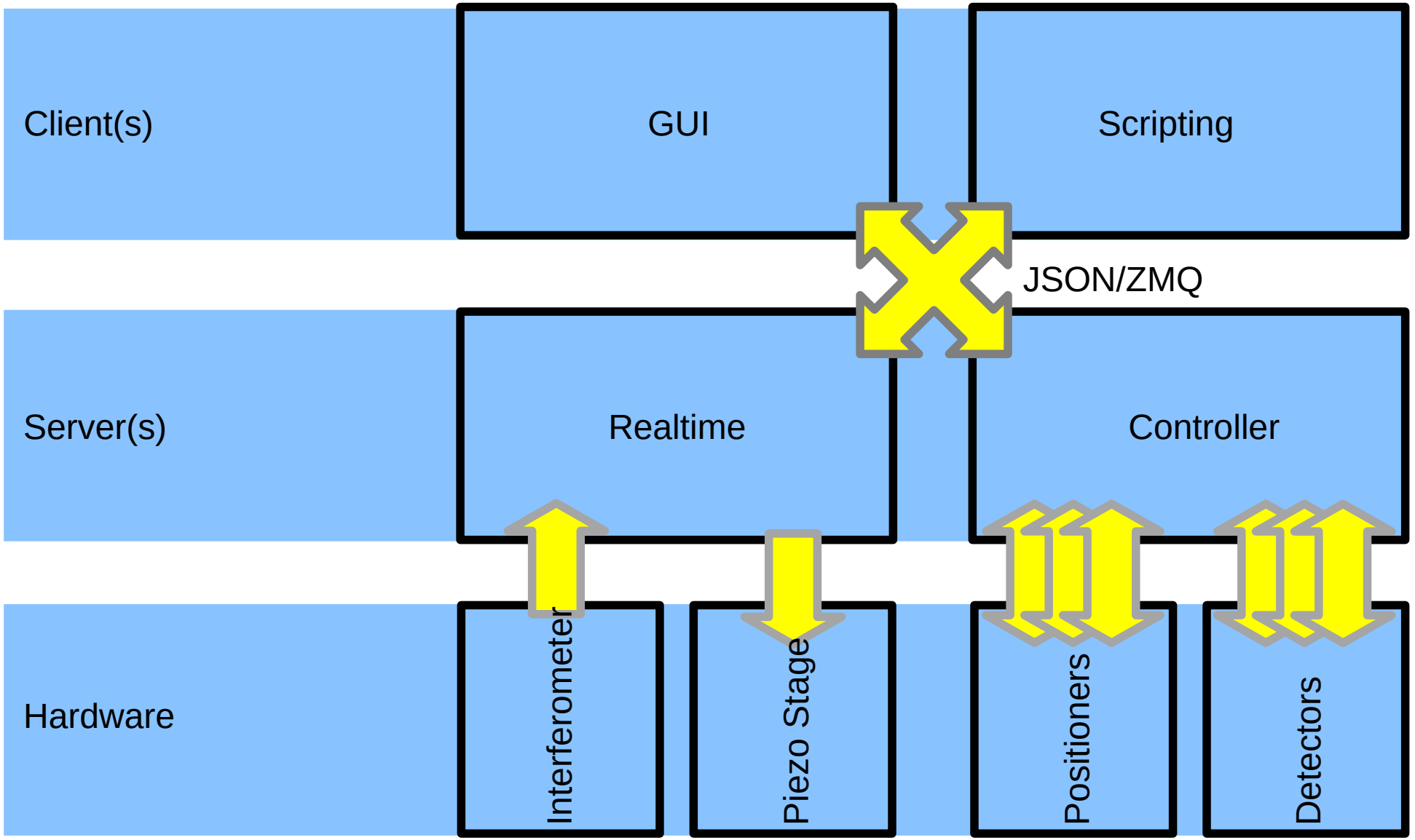
2024: Implemented at MAXYMUS@BESSYII

2026: Tested at CHIARA/SoftiMAX@MAXIV with pyStxm as GUI

Pixelator Philosophy

- Strong server/client separation with ZMQ messaging in between
- Modular hardware interface to make it easy to implement new hardware
- Rely on messaging rather than assuming actions are performed
- Avoid code forks
  - Beamline implementations differ only by configuration settings
  - Features implemented in a general way in order to not limit possibilities
- Focus on supporting STXM & ptychography instruments, but applicable to a wide variety of scanning instruments
- Standard file format (NeXus/HDF5)
  - Write lots of metadata
  - Write interferometer positions

Pixelator System Components



Pixelator Controller

C++ program with:

- Core logic
- All scan logic
- Configuration

Hardware modules consist of:

- **Controllers**
  - Handles communication with actual hardware
  - Implements communication protocols
- **Positioners**
  - Has a value (not always a physical position)
- **Detectors**
  - Provide value(s) at each scan point

Listen for client commands via JSON/ZMQ:

- Scan requests
- Movement requests

Publish information to clients via JSON/ZMQ:

- Scan data
- All info relevant to a GUI

Pixelator Controller

Supported hardware modules include:

- **Motor Controllers**
  - NewPort (TCP)
  - PI NexLine (TCP)
  - SmarAct (Serial, USB, or TCP)
  - Zaber (Serial)
  - ZMI (via EPICS)
- **Industrial Control**
  - EPICS
  - TANGO
- **Special**
  - PandAbox (TCP)
  - Transform (combined positioners)
- **Detectors**
  - Counter input (e.g. for PMT)
  - Analog input
  - EPICS areaDetector
  - Andor CCD (TCP)
  - ZMQ interface
- **Interferometers**
  - Agilent/Keysight
  - Zygo
  - Attocube
  - Renishaw
  - Generic AquadB interface
- **PCI Cards**
  - NI-6602 Counter/Timer
  - NI-6733 Analog Output
  - NI-6289 Analog Input

Pixel Clock

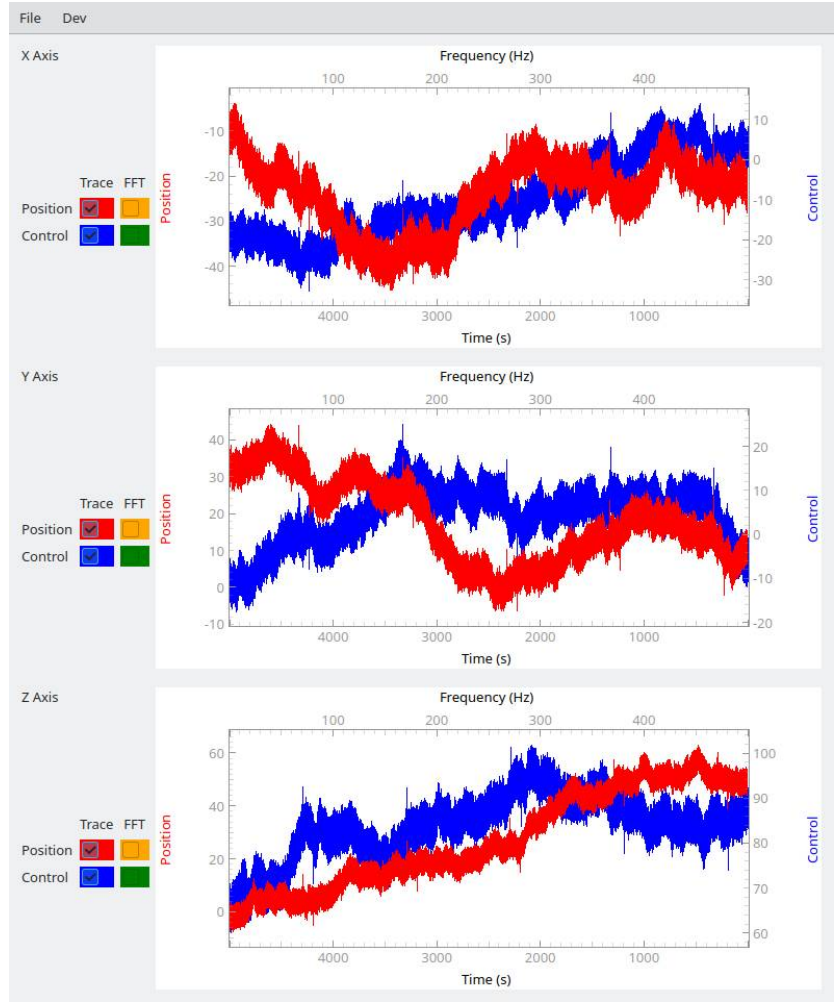
Pixelator Realtime

Based on Orocos toolchain:
`https://www.orocos.org/toolchain.html`

Realtime environment for time-critical tasks

- Configuration of endstation done in the `start.ops` script
  - Various interferometer axes
  - Different geometries
- Interferometeric sample position feedback
  - Positions and control voltages published via ZMQ
- Fast data sampling
- Coordination of combined-axis (i.e. tilted) scans
- Software correction of energy-drift

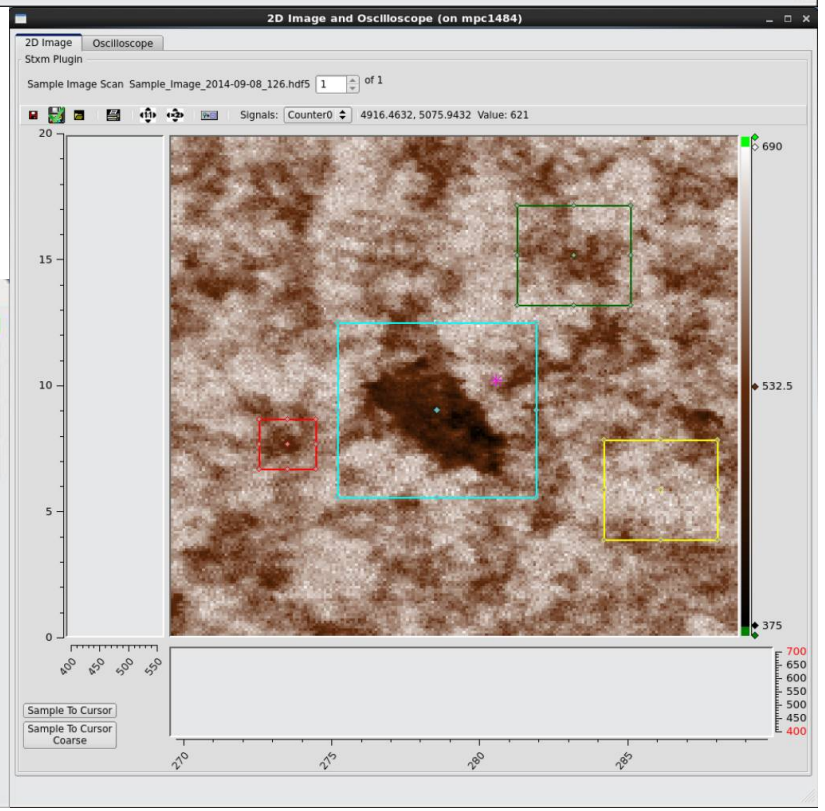
Typically run in a Shellbox and interact with via telnet.



Pixelator GUI

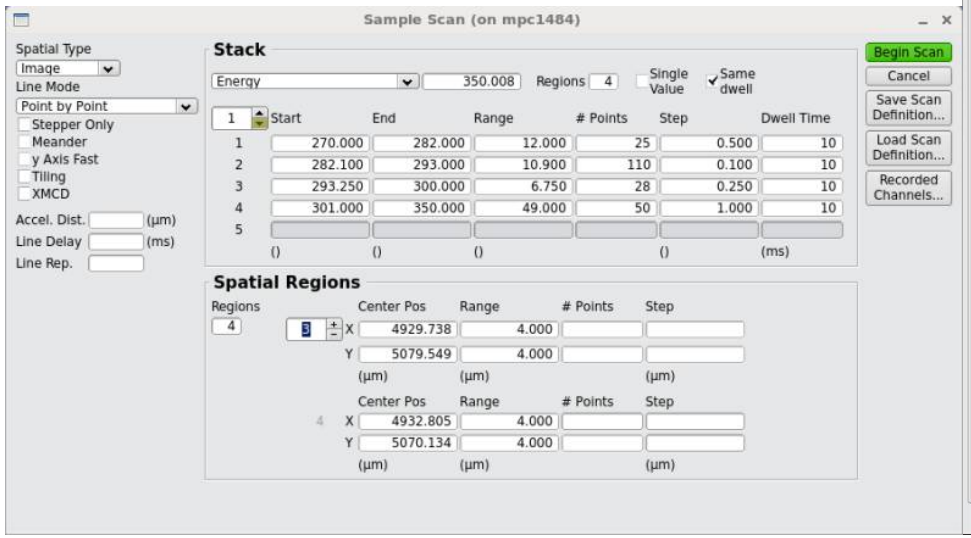
GUI design:

- Minimal logic, just forward requests
- Plot published data
- Configuration provided by *Controller*:
  - *Beamline Control* section
  - Can hide positioners



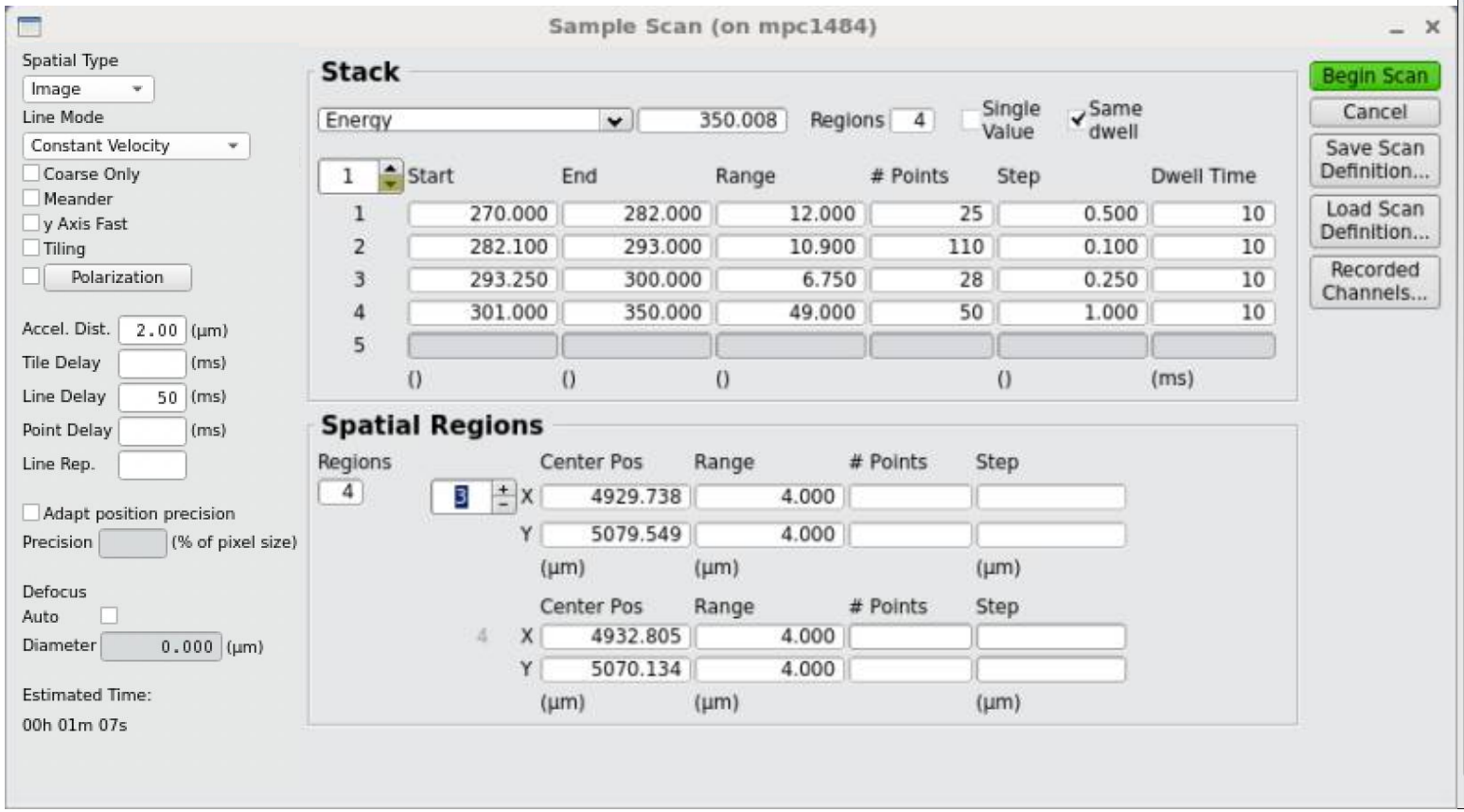
Original GUI by Semafor AG:

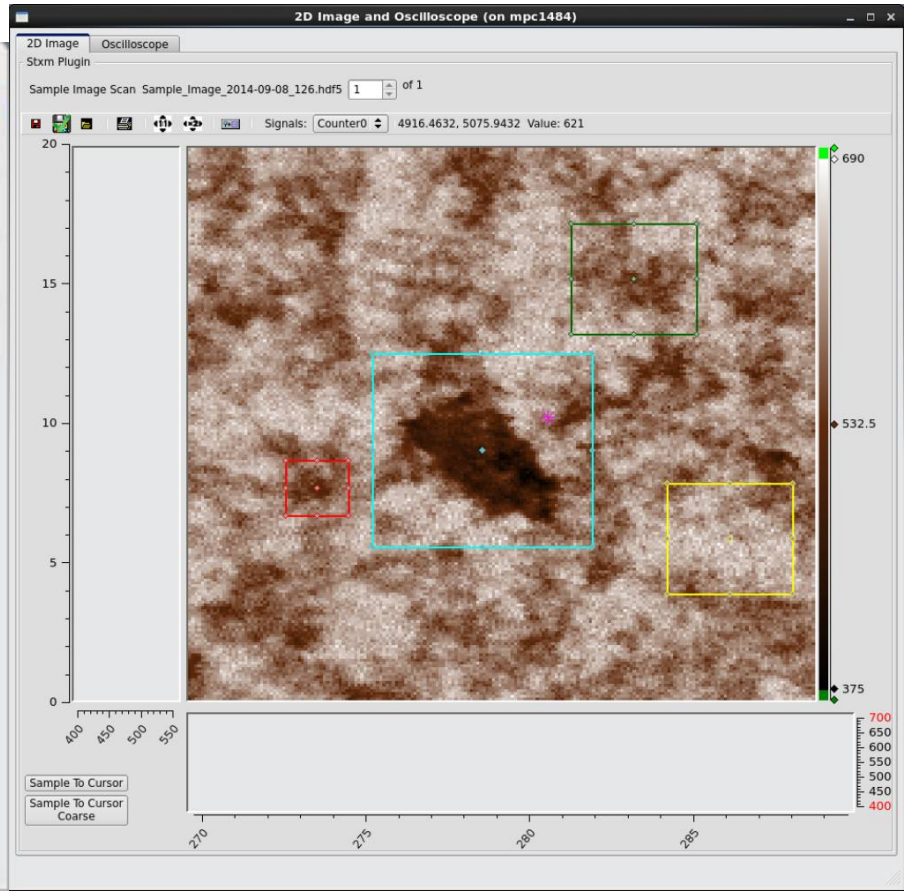
- Qt widgets
- *Intens* scripting language
- Very similar to the ALS *StxmControl* GUI



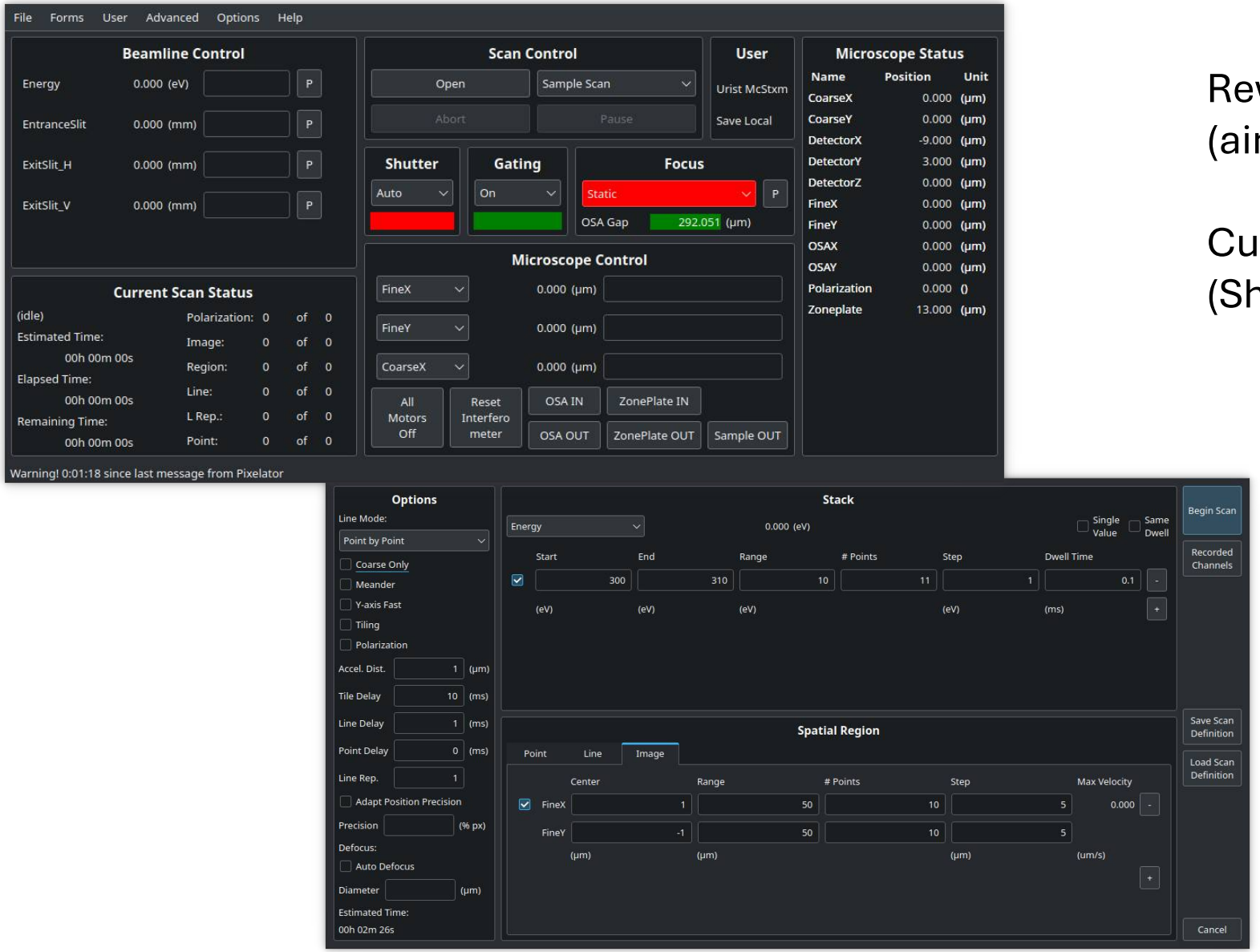
Pixelator GUI

- Many scan options
- Thin client – all calculations done in the Controller





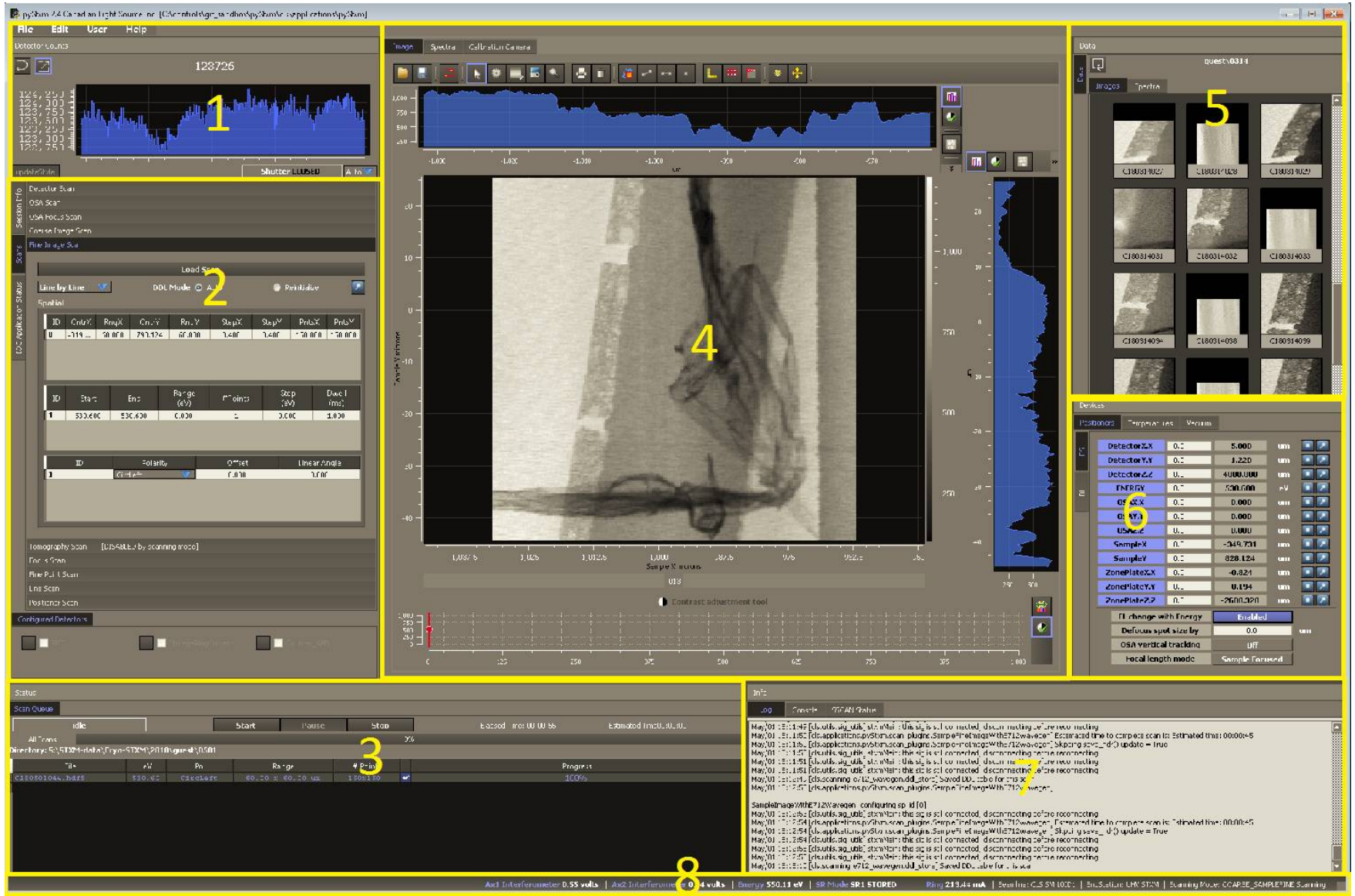
Pyxcellent GUI



Rewrite of standard GUI in Python
(aim to replace Intens GUI)

Currently work in progress
(Shown here with dark theme on KDE)

PyStxm GUI – Russ Berg (CLS)



1) Live counts
   Shutter control
2) Scan Parameters
   Device monitor
   Preferences
3) Scan Queue
4) Data visualisation
5) Data browser
6) Devices overview
7) Log
   Python console
8) Status bar

Compatibility work is in progress

Pixelator Script

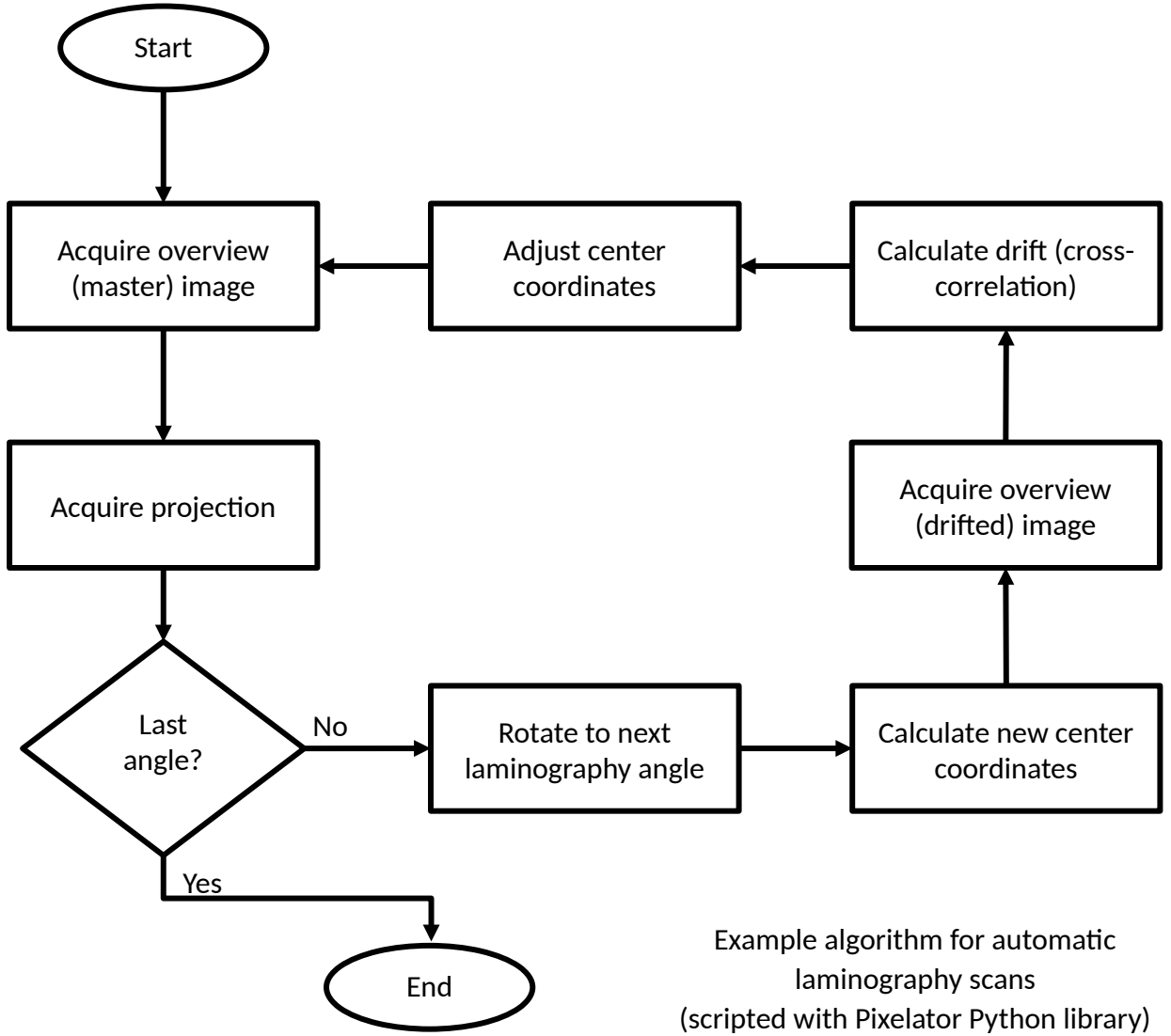
Python module

Send command requests over ZMQ:

- Scans
- Positioner motions
- Simple analysis:
  - Sample position tracking
  - Focus adjustment

Mostly used for repetitive experiments:

- Laminography
- XMCD hysteresis loops

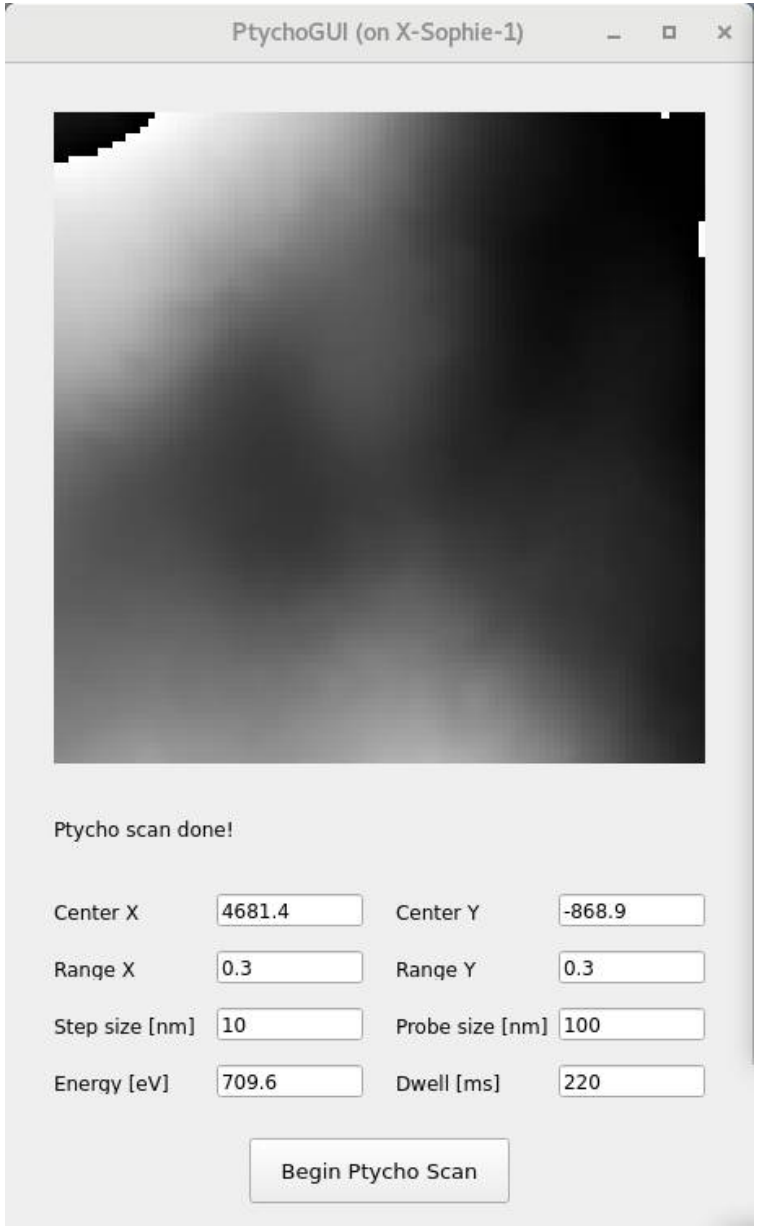


Example algorithm for automatic laminography scans (scripted with Pixelator Python library)

Pixelator GUI: PtychoGUI

GUI specifically for launching ptychography scans:

- Thin GUI using the scripting library
- Calculates points in Fermat spiral pattern
- Points are re-ordered into a roughly rectangular raster path
- Point list is given to Pixelator in the scan request
- Published scan data is regridded for plotting



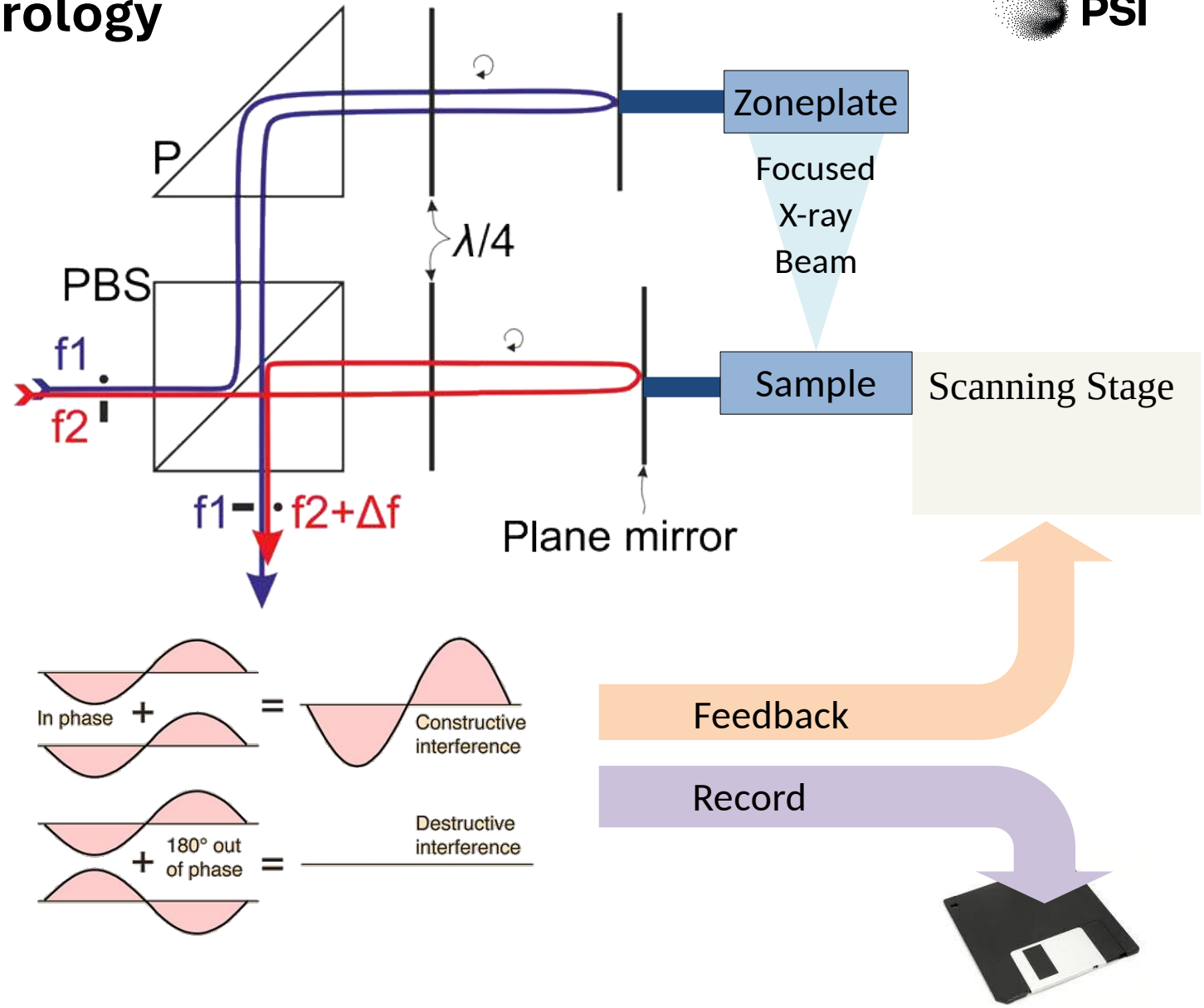
Interferometric Position Metrology

On the nanometer scale:

- All materials bend and wobble like Jello
- Stage motions are imprecise

Laser interferometers provide stable position metrology:

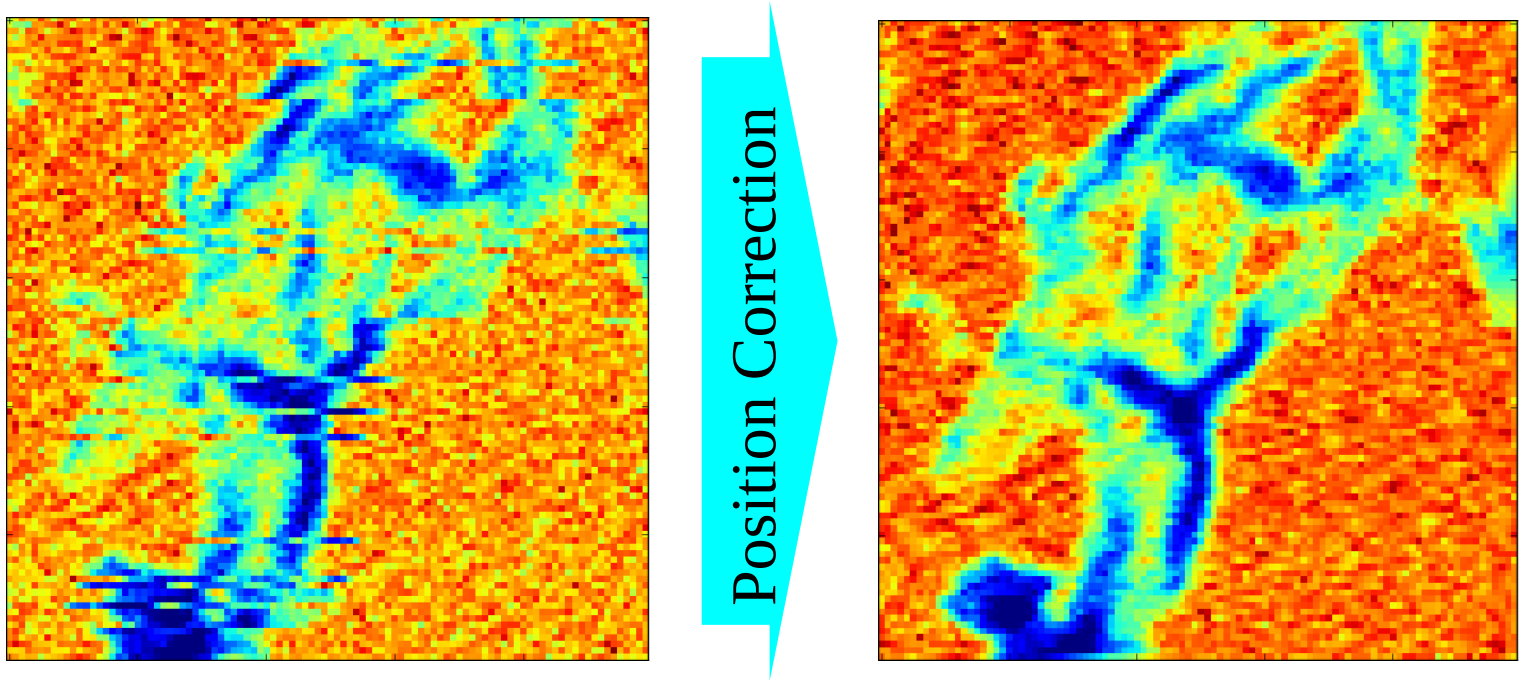
1) Split beam
2) Bounce beam off mirrors attached to reference objects
3) Recombine beams
4) Use variations in interference to calculate changes in relative position



Interferometric Position Metrology

Heterodyne laser interferometer:

- Resolution: 0.3 nm
- non-contact, long range
- exteroceptive:
  include thermal drifts in the measurement
- linear, accurate and stable scale



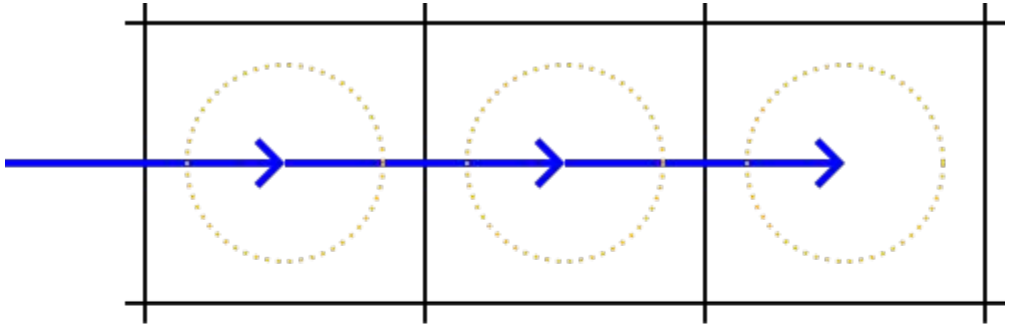
Assumed pixel positions

Measured pixel positions

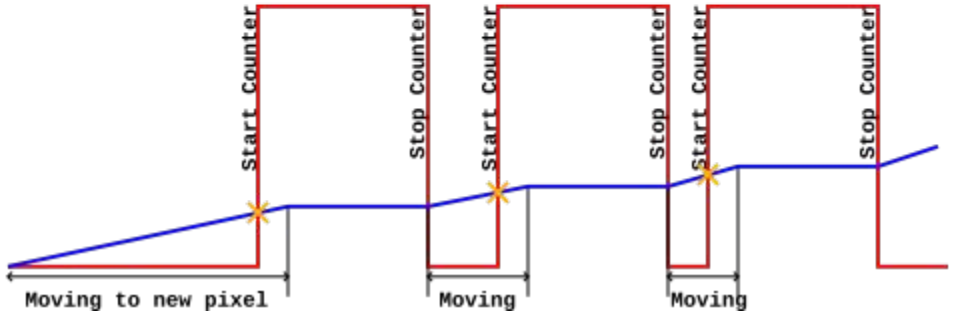
Point-by-Point Scan

Stop at each pixel position:

- High positional accuracy
- Start counter timer when arriving close enough to pixel center
- Dead time when moving between pixels
- Lots of acceleration/deceleration



Spatial Pixels

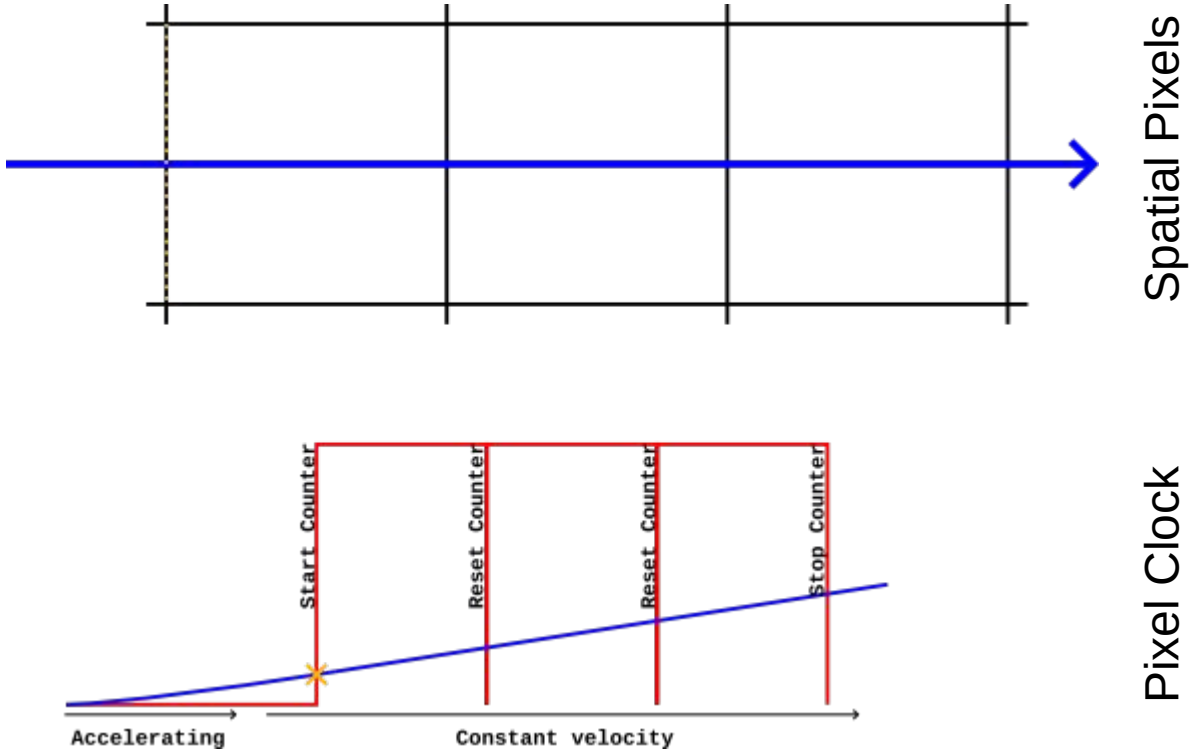


Pixel Clock

Contant-Velocity Scan

Measurements on the fly:

- Lower positional accuracy
- Give some run-up distance (dead time)
- Start pixel train at line start
- Divide counts by time to assign to pixels
- Minimised acceleration/deceleration



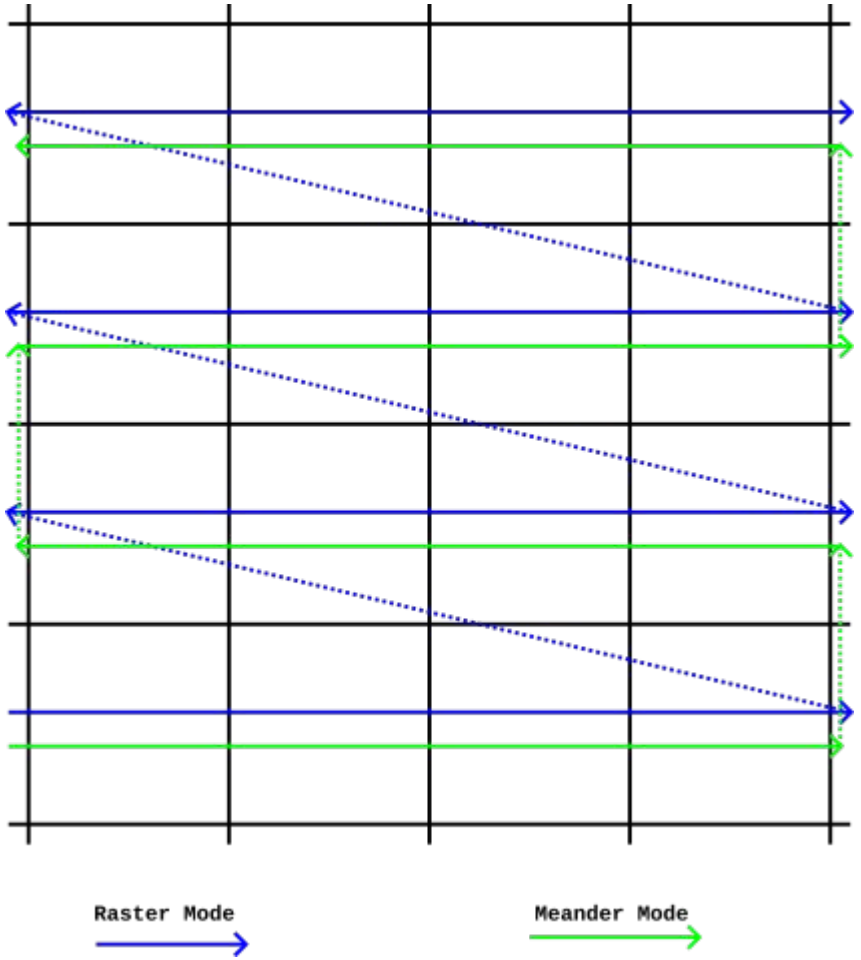
Scan Patterns

STXM scan patterns:

- Sawtooth raster (scan row then rewind)
- Meander (scan odd rows to right, even rows to left)
- Y-axis fast sawtooth raster (scan column then rewind)
- Y-axis fast meander (scan odd columns up, even columns down)

Ptychography scan patterns:

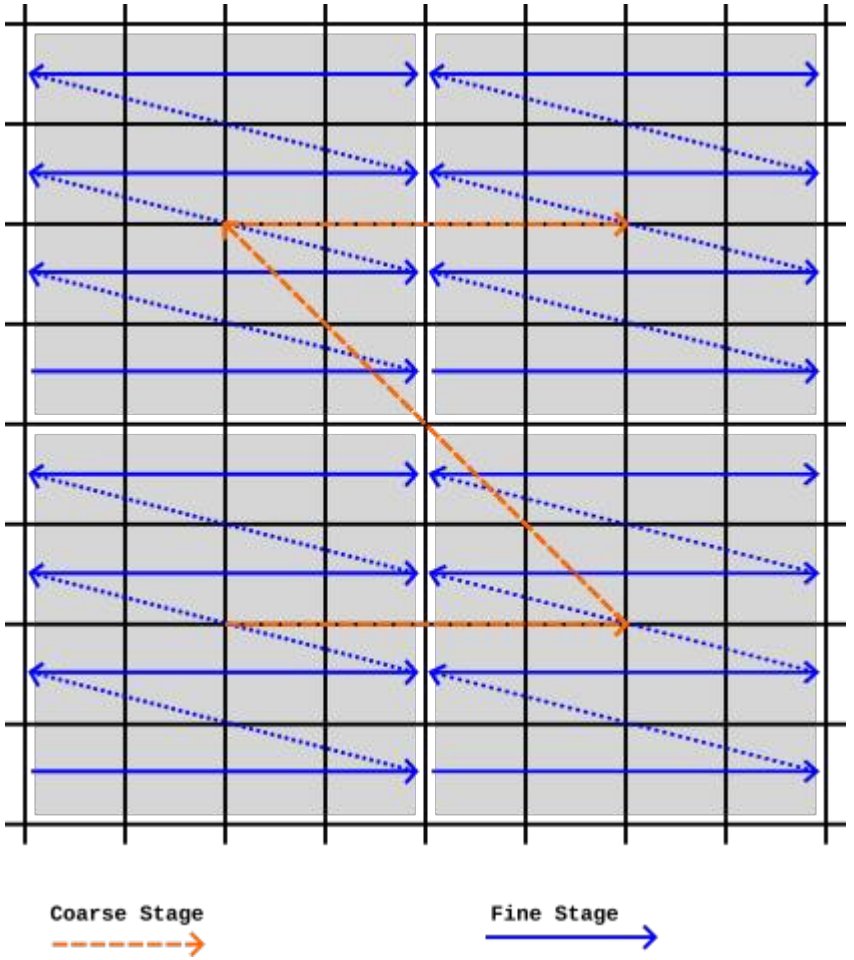
- Fermat spiral (reordered to quasi-rectangular scan)
- Arbitrary list of points



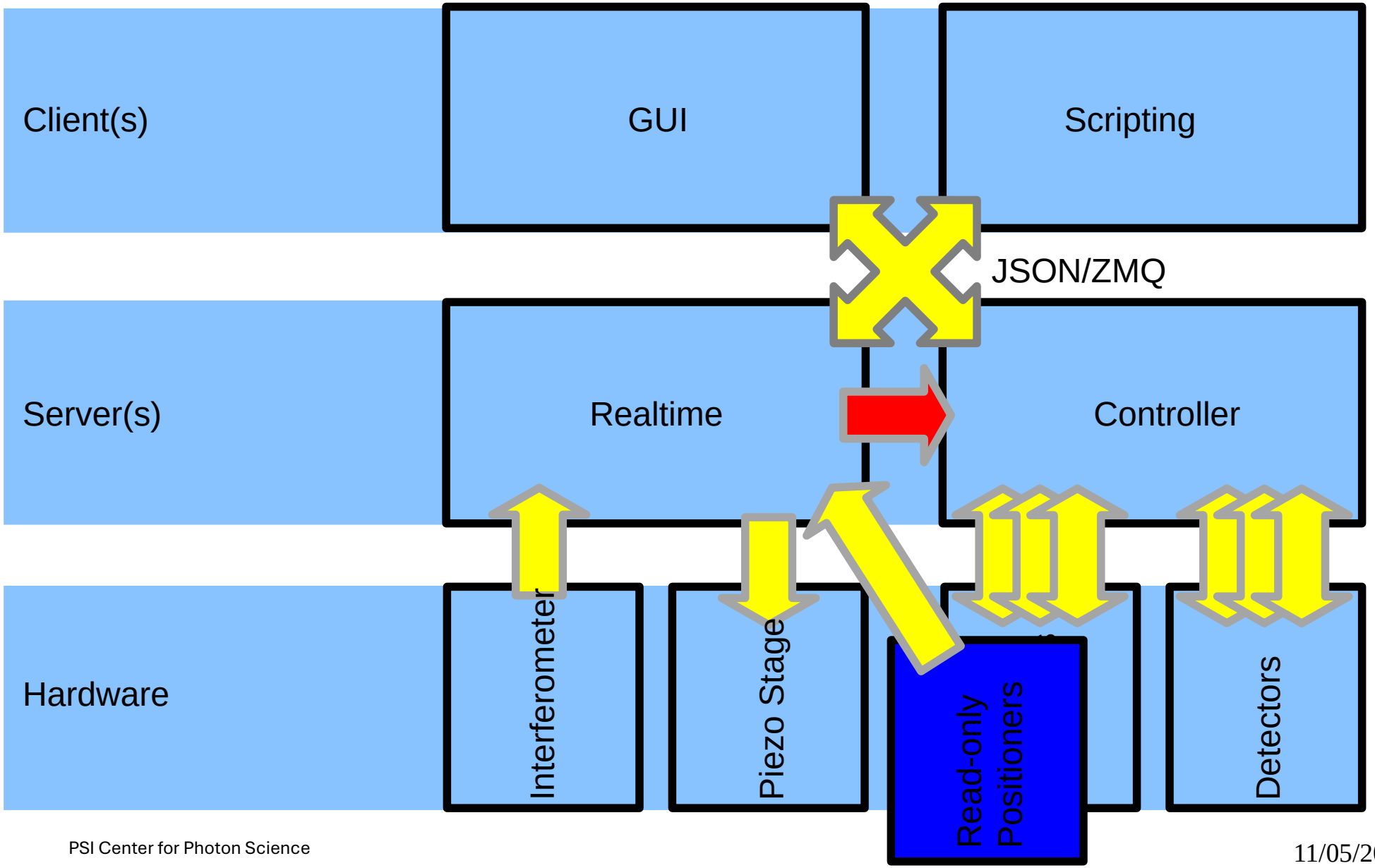
Tiling Mode

Divide scan into “tiles” that fit within the range of the sample fine stage:

- Minimise coarse stage movements
- Less scan overhead
- Interferometric positioning makes stitching precise



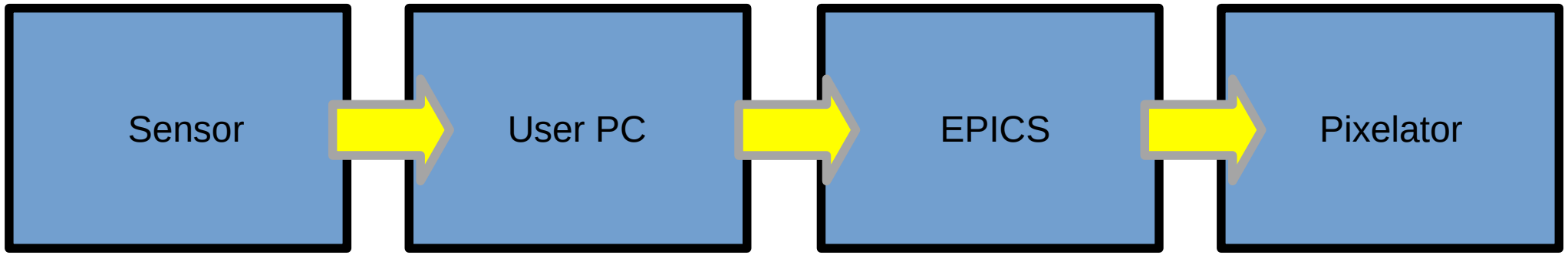
Fast Buffering of Slow Devices



Per-Pixel Recording of Arbitrary Data

Fast buffering of the “read-only positioners”
allows per-pixel monitoring of EPICS channels:

- Storage ring current
- Sample environment, e.g.:
  - Temperature
  - Field strength



Pixelator *NXstxm* Data Files

NeXus data format in HDF5 files

- Well documented standard:
  https://manual.nexusformat.org/classes/applications/NXstxm.html
- F.A.I.R. data

- Simple view of data in NXdata groups
- Listed data for image reconstructions in NXdetector groups
- Per-pixel values of any channel
- Can inspect file contents with:
  https://myhdf5.hdfgroup.org/

root of HDF5 file
Root level of a NeXus data file

NXentry — One per spatial region
All data belonging to one scan or run.
A given NeXus file can contain multiple related scans or runs

NXinstrument
The data needed to describe an instrument.
Contains groups for each relevant instrument component.

- **NXsource**
- **NXcollimator**
- **NXattenuator**
- **NXdetector** — One per detector, list
- ...

NXsample
All the information about the sample

NXmonitor — Ring Current
Incident intensity monitor

NXuser
User information

NXdata — One per detector, E.Y.X array
Links to plottable data in the NXdetector group --
one instance for each detector bank. Provides support
for generating a view of the data automatically.

...

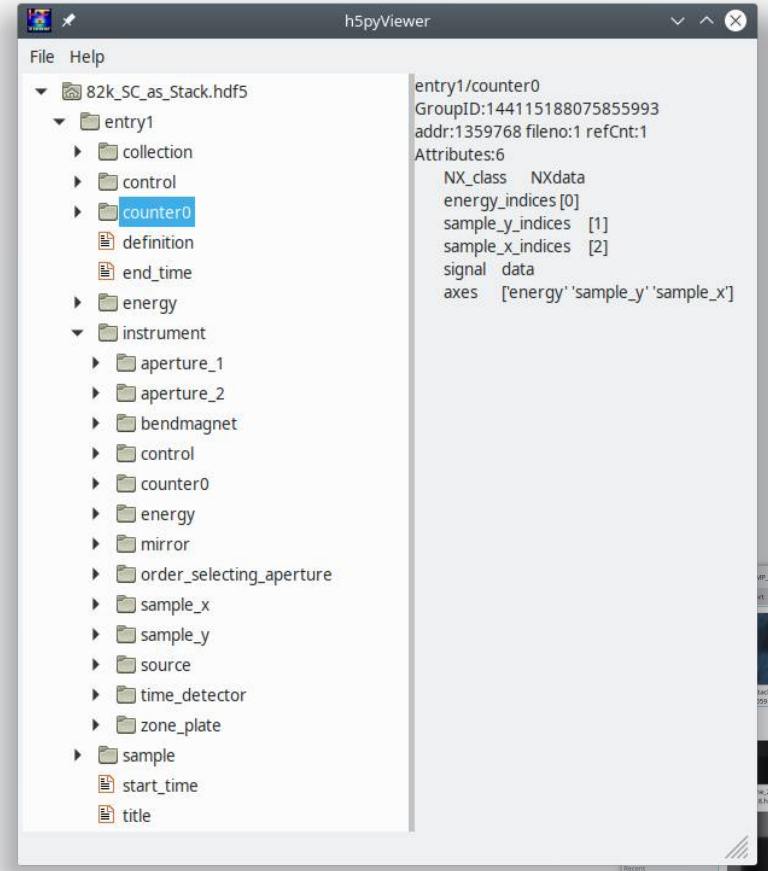
NXentry
Additional NXentry groups are optional

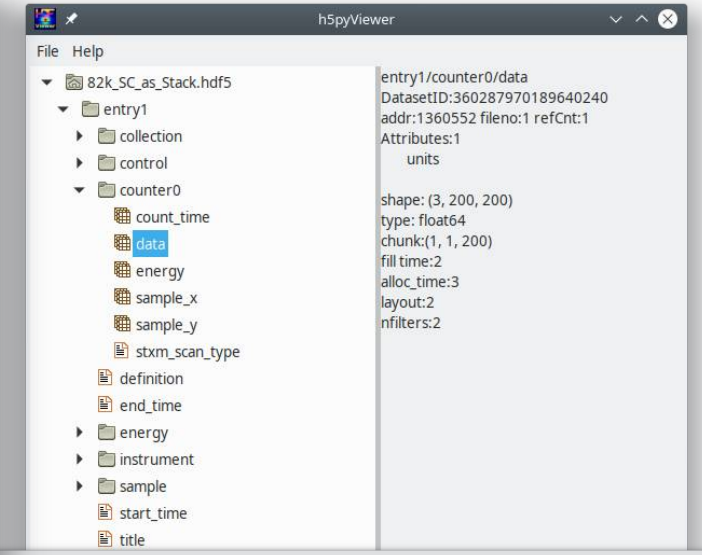
Pixelator *NXstxm* Data Files

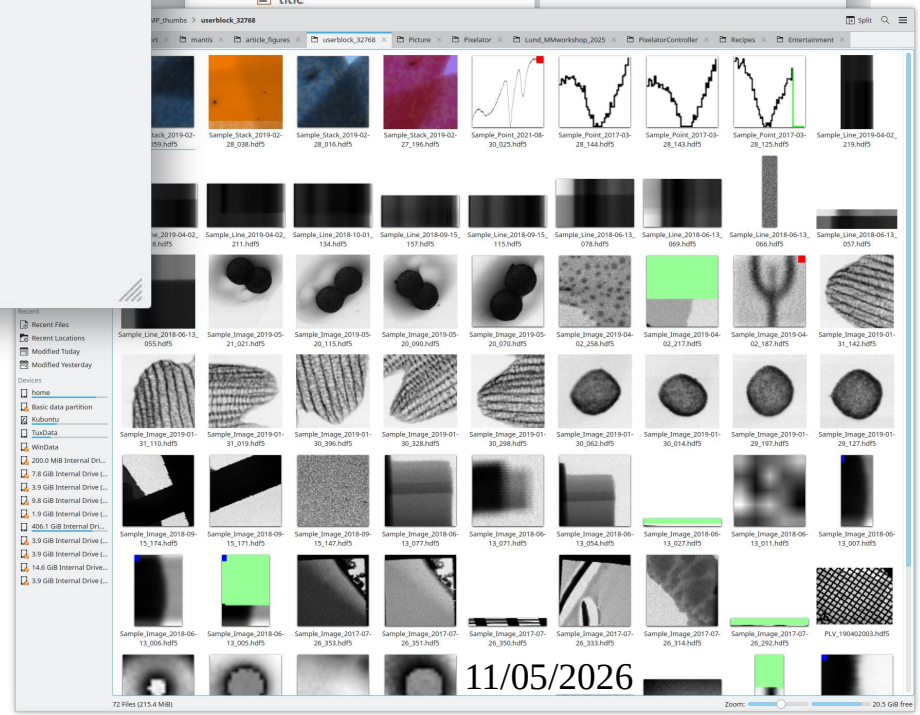
NeXus data format in HDF5 files

- Well documented standard
  https://doi.org/10.1063/1.4937536
- F.A.I.R. data

- Configurable metadata
- Script inserts thumbnail data
- File browser plugins for previews



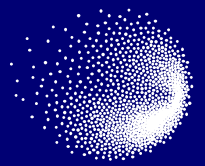




Pixelator Development

Currently planned development goals

- Improve support of *PandAbox* FPGA
- Integrate super-sampling scan mode
- Interoperate with Orphyd/bluesky and BEC
- Improve interoperability with pyStxm
- Improve documentation



PSI Center for Photon Science

With thanks to:

- Jörg Raabe (PSI)
- Simone Finizio (PSI)
- Simon Bader (Semafor)
- Markus Weigand (MPI)
- Russ Berg (CLS)