Skywalker

Automated Alignment at LCLS

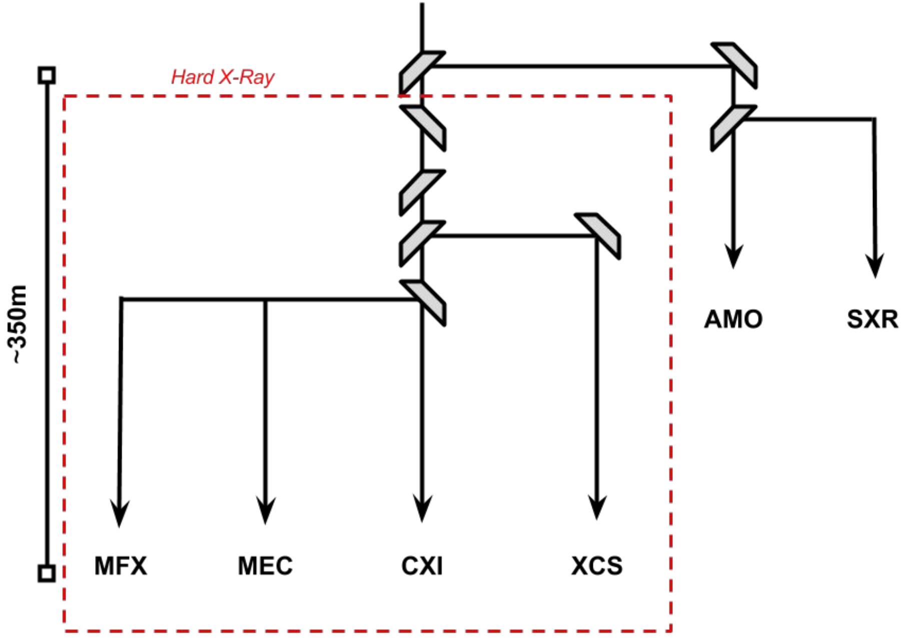
Zachary Lentz

U.S. DEPARTMENT OF ENERGY
Office of Science

SLAC NATIONAL ACCELERATOR LABORATORY

Motivation for Automatic Mirror Alignment

- Seven experimental hutches, each requiring unique pointing of flat mirror systems
  - Each with sensitive downstream optics
  - Common for experiments to run in serial
- LCLS-II upgrade in 2019 adds more mirrors and more endstations
  - Emphasis on automation

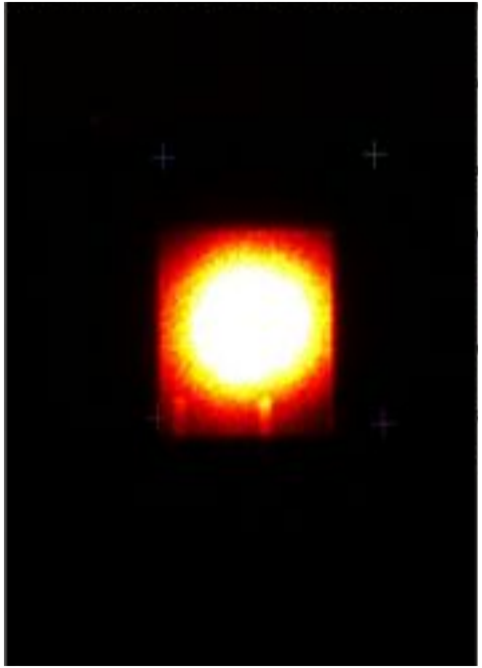


Project Skywalker

- **Set positions and pre-alignment are not good enough**
  - Pointing of the FEL is not repeatable enough for `set and forget` values
    - No diagnostics sensitive enough to determine undulator pointing for optics 350m away
  - Over 190 devices in the common areas of the beamline
- **Skywalker Deliverables**
  - Operators should be able to `single-click` align to any of Hard X-Ray endstations
    - Should be done faster than manual alignment (or time claimed by operators for manual alignment)
  - Deal with dynamic target selection
  - Full automation
    - Watch for drops in FEL energy
    - Clear the beamline of obstructions
    - Durable against day to day operation
  - Create a suite of tools for future automation projects

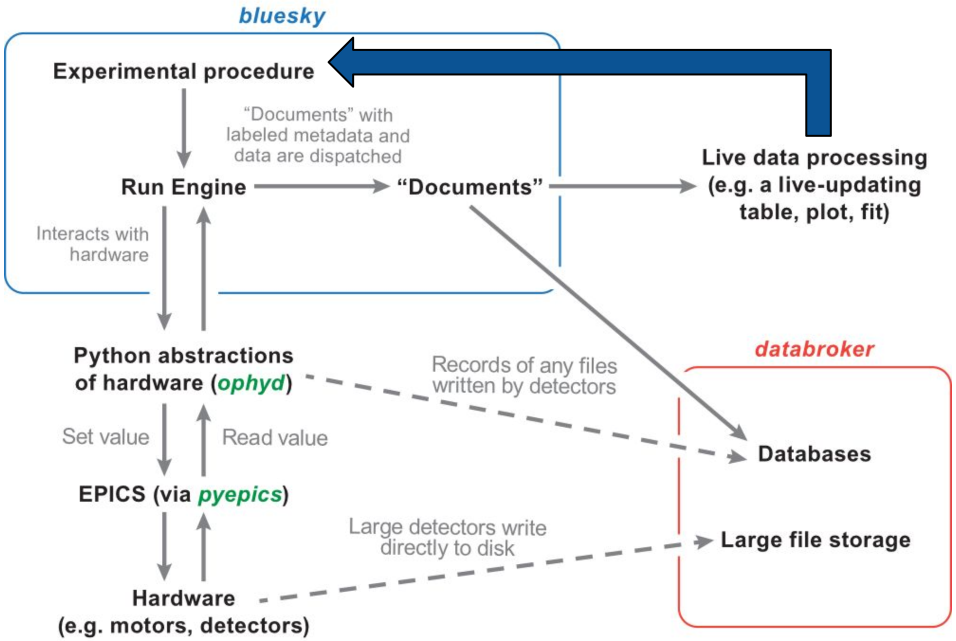
The Tools

- Flat Mirror System
  - Controlled by ELMO Drives with a Beckhoff PLC Ethercat Master
  - Stepper and piezo in series on the pitch mechanism
- 4 Jaw-Slits
  - Produced by JJ-XRAY
  - Use EPICS motor record
- Imagers
  - YAG crystal fluoresces when X-Rays are incident
  - EPICS AreaDetector
- Software
  - Python 3.5 and up
  - Ophyd and Bluesky



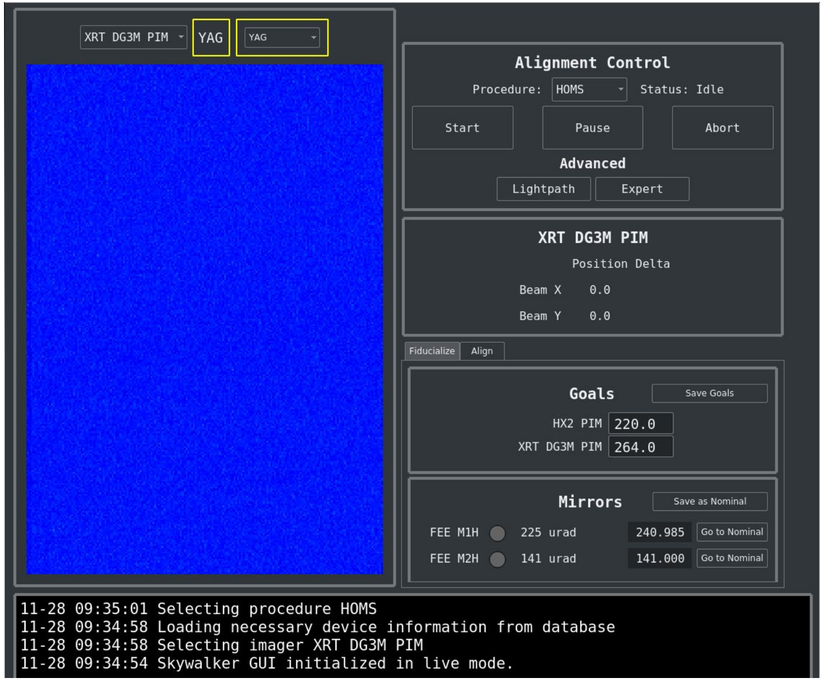
Bluesky and Automation: Motivation

- An appealing combination because we already have Ophyd devices
- Bluesky provides flow control
  - Start, stop, pause, rewind, resume, suspend
- Bluesky has an interface that lends itself well to re-using well-tested code in new ways.
  - Use documents interface to feed information back into the plan

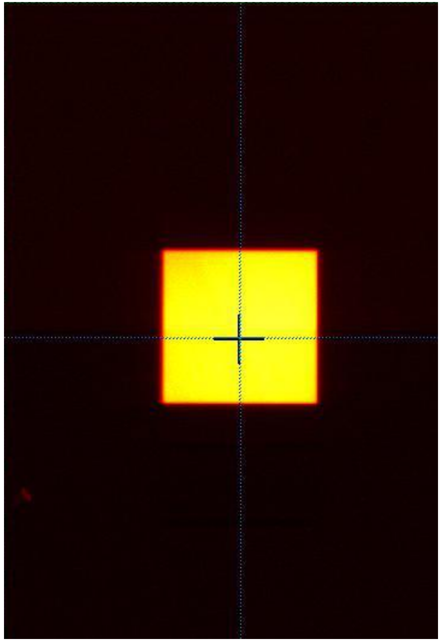


PyDM User Interface

- Allows user to select procedure
  - Select hutch for beam delivery
  - Tune tolerances and averaging
- Full control of RunEngine
- Live feedback for mirror and imager systems
- Displays logging output



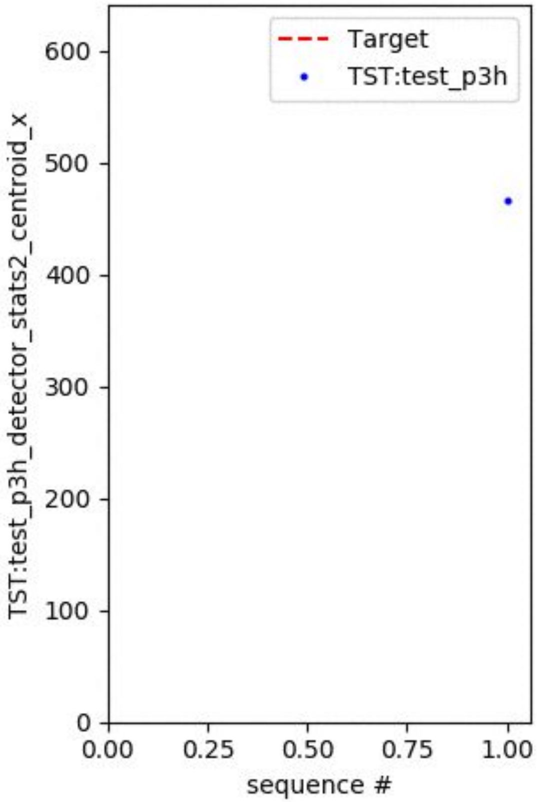
Defining Alignment

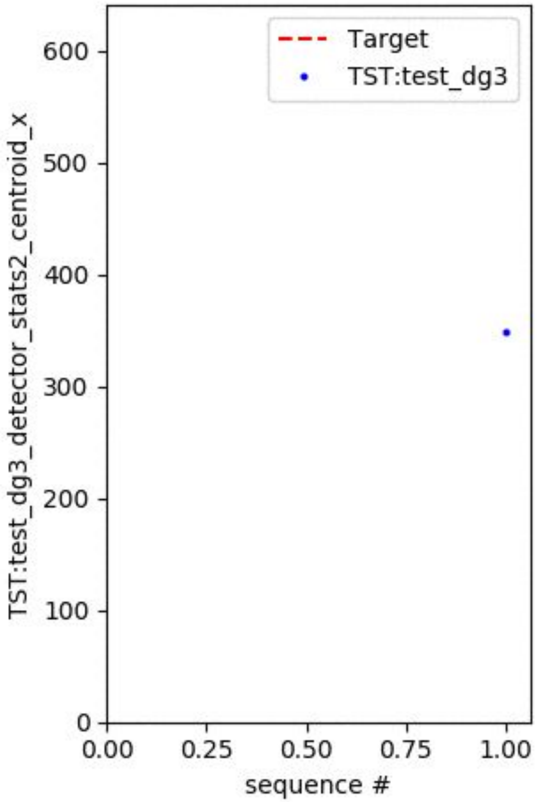


- Fiducialize imager by using previously aligned 4-Jaw slits
  - Trim the beam down to only a small subsection that is aligned
  - Calculate the centroid of this subsection
  - Expand slits and difference between the open and closed centroid is the error in our pointing
  - Fiducialization at 2 points defines our target beam axis

2-Mirror Alignment Procedure

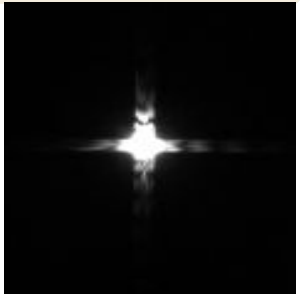
- Basic iterative process
- Match first mirror to a near imager, second mirror to a far imager
- Align each mirror-imager pair to their goal pixel, which slightly misaligns the other pair. Repeat.
- Use linear fit to minimize number of steps
- Take average of many blueky reads from the area detector beam centroid after each move

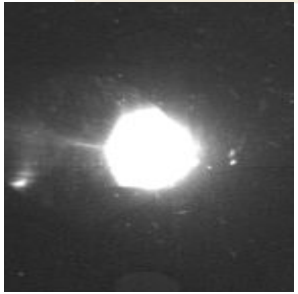


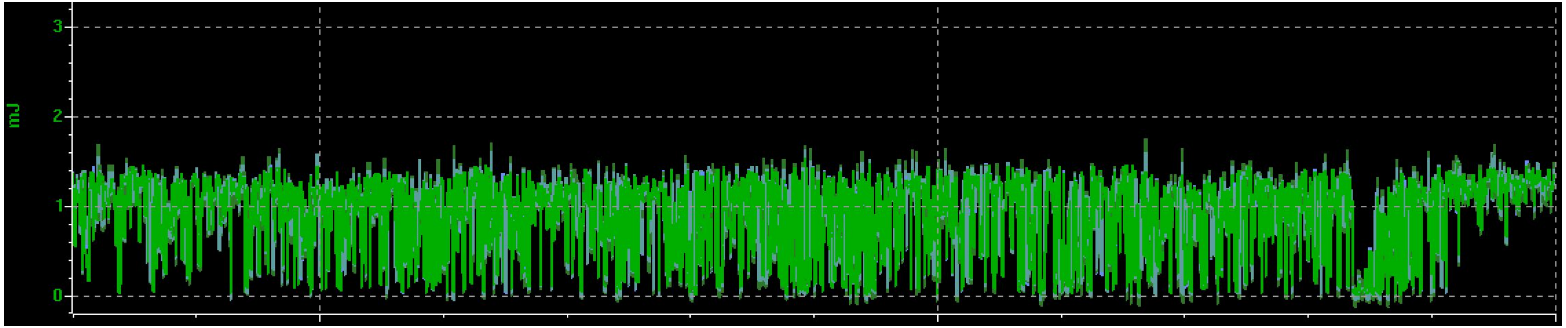


Hurdles: It’s never easy

- Unstable FEL
  - Heavy use of Bluesky suspenders
  - Bad events must be filtered
- Optical Phenomena
  - Sometimes, more complex image filtering may be needed… but you may not know until taking beam.







Commissioning Results

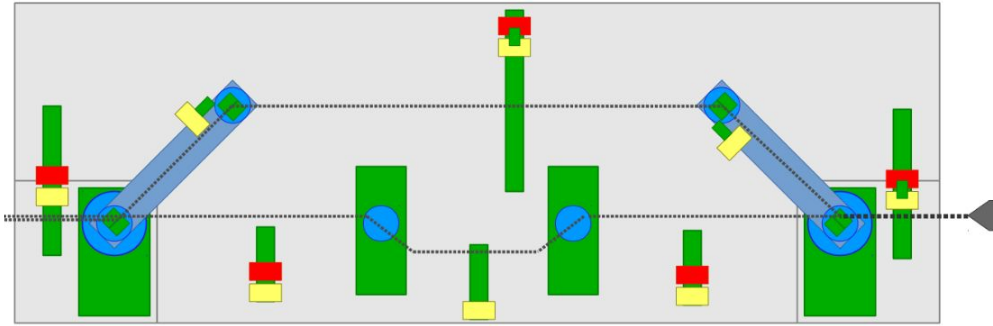
- Single Mirror Alignment
  - No prior knowledge
  - Capable of pixel precision (3.4 um)
    - Accomplished in roughly 60 seconds
  - Majority of time spent during fine adjustment
    - Within 10 pixel tolerance after first step
  - Jupyter Notebook
- Two Mirror Alignment
  - Able to reliably find solutions within a pixel
  - Starting with no prior knowledge, no beam on imagers
    - 7 minutes
  - Seeding the run with approximate values
    - 2 minutes

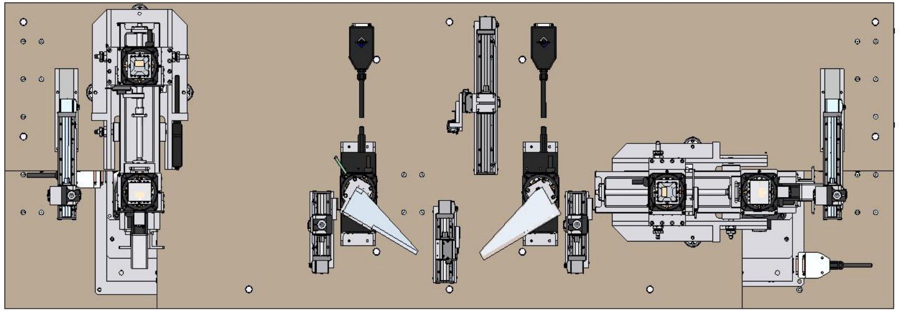
Bluesky and Automation: The Verdict

- Bluesky is well-suited to procedures that need both dynamic flow control and data feedback.
  - Similar level of complexity as doing it functionally, but you get a lot of extra control in a clean way.
  - No need to implement your own interrupt handler, your own pause/resume, your own device abstraction schema...
- Can re-use any of the data collection and visualization code
- There is room for growth
  - Suspenders are powerful and have huge potential for new features e.g. message-dependent suspension
  - We’ve only scratched the surface of adaptive plans

Follow-up Project: Split and Delay

- LCLS installed a new Split and Delay assembly
- Delay line is sensitive to small transverse errors: these have strong effects on the beam overlap.
- Re-used code from Skywalker to automatically calibrate for repeatable transverse error in the delay branches
  - Use extra rotation axis to correct for the pointing error
- With the calibration, we can quickly scan across delay intervals.





Related Upcoming Projects

- Automated mirror centering
- Automated lens focusing
- Automated laser/x-ray overlap, with specified laser parameters tuned automatically
- Record data from alignment routines to optimize the alignment and analyze beam statistics over time

Links

Documentation

- https://pswww.slac.stanford.edu/swdoc/releases/skywalker
- https://nsls-ii.github.io/ophyd
- https://nsls-ii.github.io/bluesky

Source

- https://github.com/slaclab/skywalker
- https://github.com/slaclab/pswalker
- https://github.com/slaclab/lightpath
- https://github.com/slaclab/happi
- https://github.com/slaclab/psbeam
- https://github.com/NSLS-II/ophyd
- https://github.com/NSLS-I/bluesky

Special Thanks

- Daniel Flath
- Alex Wallace
- Teddy Rendahl
- Abdullah Ahmed Rashed
- Hugo Slepicka
- Diling Zhu
- Karl Gumerlock
- Daniel Allan
- Thomas Caswell
- Ken Lauer
- Matt Gibbs