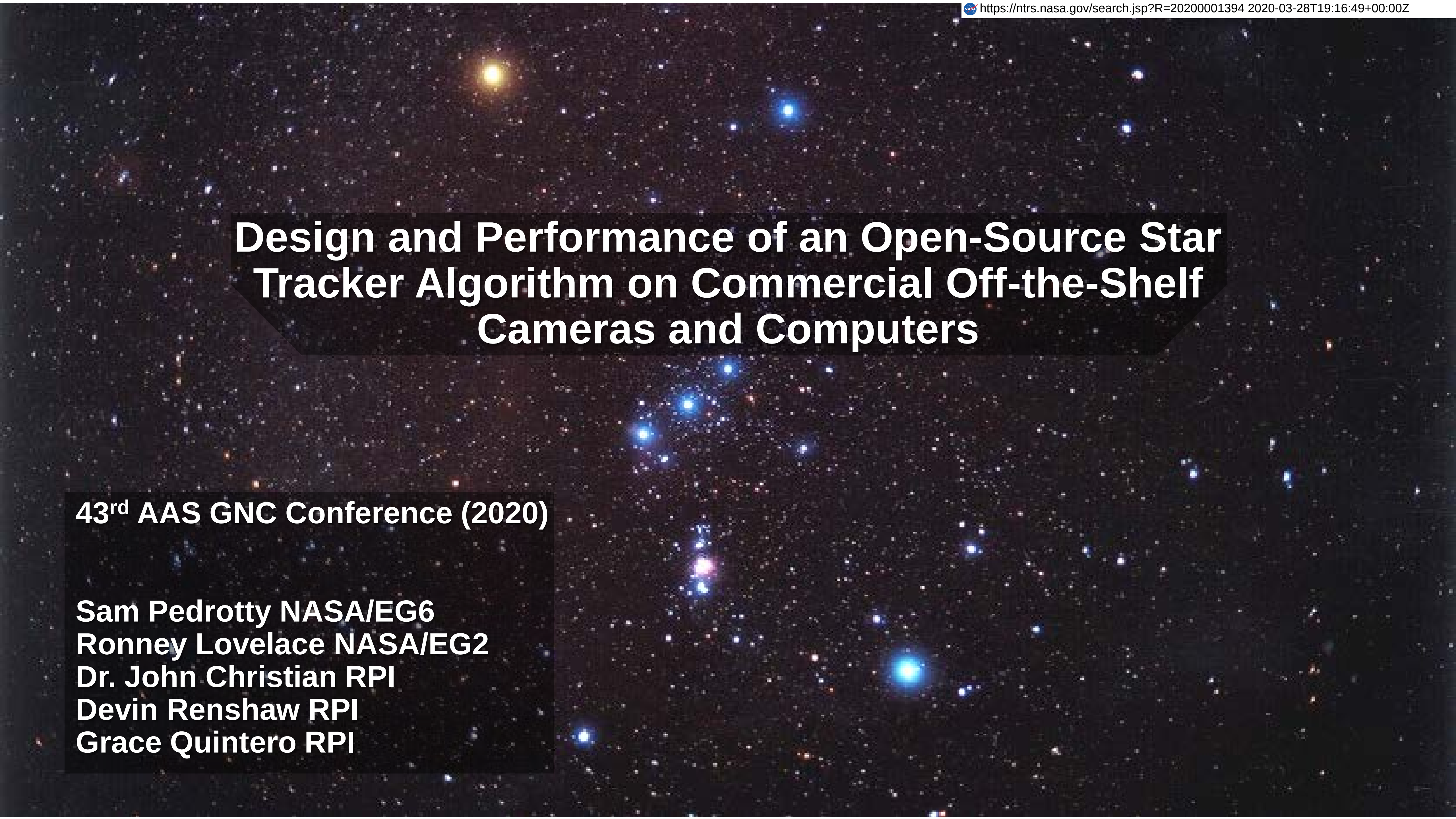


Design and Performance of an Open-Source Star Tracker Algorithm on Commercial Off-the-Shelf Cameras and Computers

43rd AAS GNC Conference (2020)

Sam Pedrotty NASA/EG6
Ronney Lovelace NASA/EG2
Dr. John Christian RPI
Devin Renshaw RPI
Grace Quintero RPI

Why are we here?

- We like space
- Accelerate spaceflight
- Share our work and planned release of a star tracker algorithm designed for and demonstrated on COTS cameras and computers
- Gauge interest
- Create partnerships

Content

- Star Tracker Overview
- Goal
- Design
- Development Path
- Tested Hardware Components
- Performance
- Challenges
- Forward Plan

Star Trackers

- □ Big
- □ Expensive
- □ Long lead-time
- □ High accuracy
- □ Developing CubeSat market

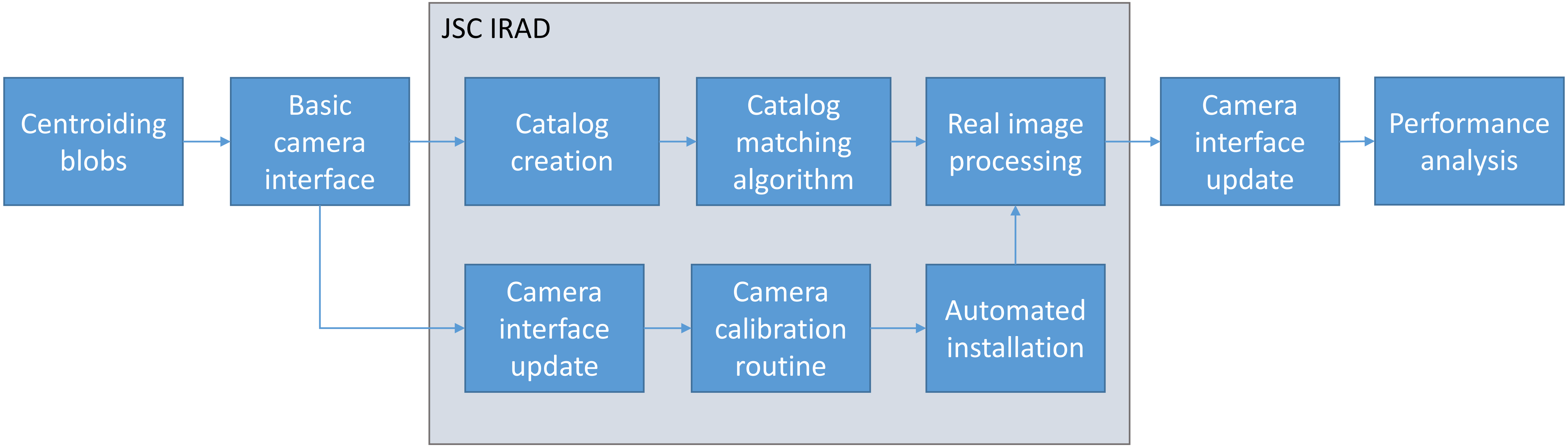
The Dream

- ❑ **Open-source**
- ❑ **Computer/Operating System agnostic**
- ❑ **Camera agnostic**
- ❑ **Good accuracy (but not ITAR good)**
  - Not in the open-source stuff, anyway
- ❑ **Enables improved spacecraft performance/capability, reduces cost and schedule**
- ❑ **Improved by the community**

Design

- ❑ **Written in Python (2 or 3)**
- ❑ **Built more like “proof of concept” as opposed to “optimized flight design”**
- ❑ **Automated installation/setup**
- ❑ **Algorithms**
  - RPI-provided algorithms open-source under 3-Clause BSD License
  - Camera calibration
    - openCV checkerboard standard
  - Catalog creation
    - Uses Hipparcos catalog
    - User-defined brightness threshold
    - Creates k-vector with inter-star angles
  - Dark frame creation
    - Either “averages” existing star images or uses a single image
  - Centroiding
    - Convert image to greyscale
    - Convert to binary image and threshold
    - Contour blobs, centroid those greater than user-input size
  - Catalog matching and attitude determination
    - Determine candidate triads, calculate inter-star angles
    - Search across k-vector to find a match
    - Singular value decomposition solution of the Wahba problem for each star matched
    - Returned attitude if matched stars greater than user-input threshold

Development Path



System Hardware Components

❑ Single Board Computers

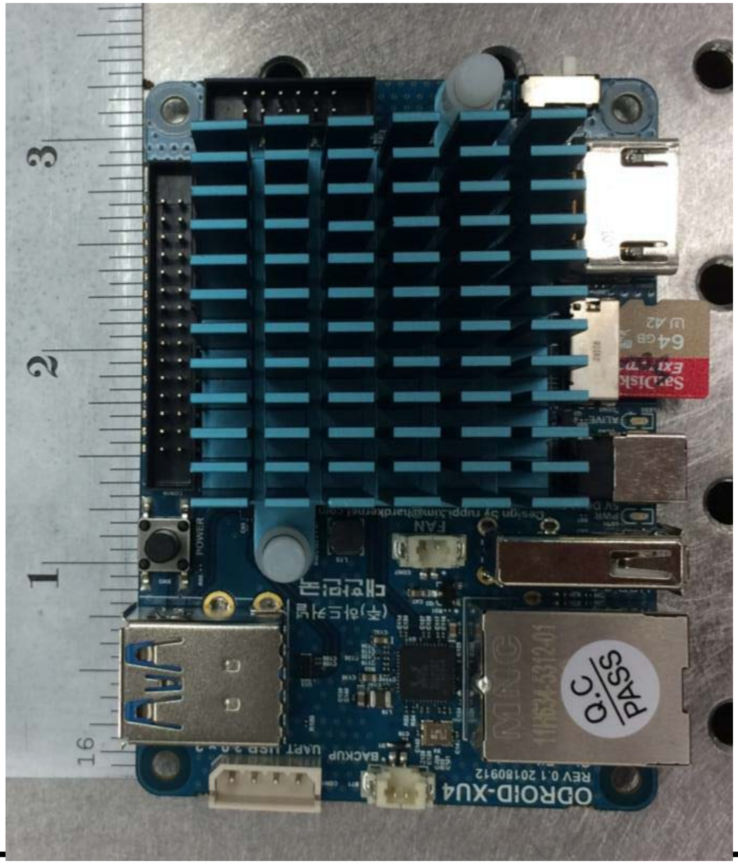
- Odroid XU4Q
- Raspberry Pi 3B+

❑ Cameras

- IDS UI-3180CP-M-GL R2
- Ximea MQ013MG-E2

❑ Lenses

- Fujinon HF12.5HA-1B
- Fujinon HF35HA-1B



Ximea Sky View

- ❑ **Max gain, 200ms exposure, 1280x1024 px, jpeg, HF12.5HA-1B lens**

Ximea Sky View (belt zoom)

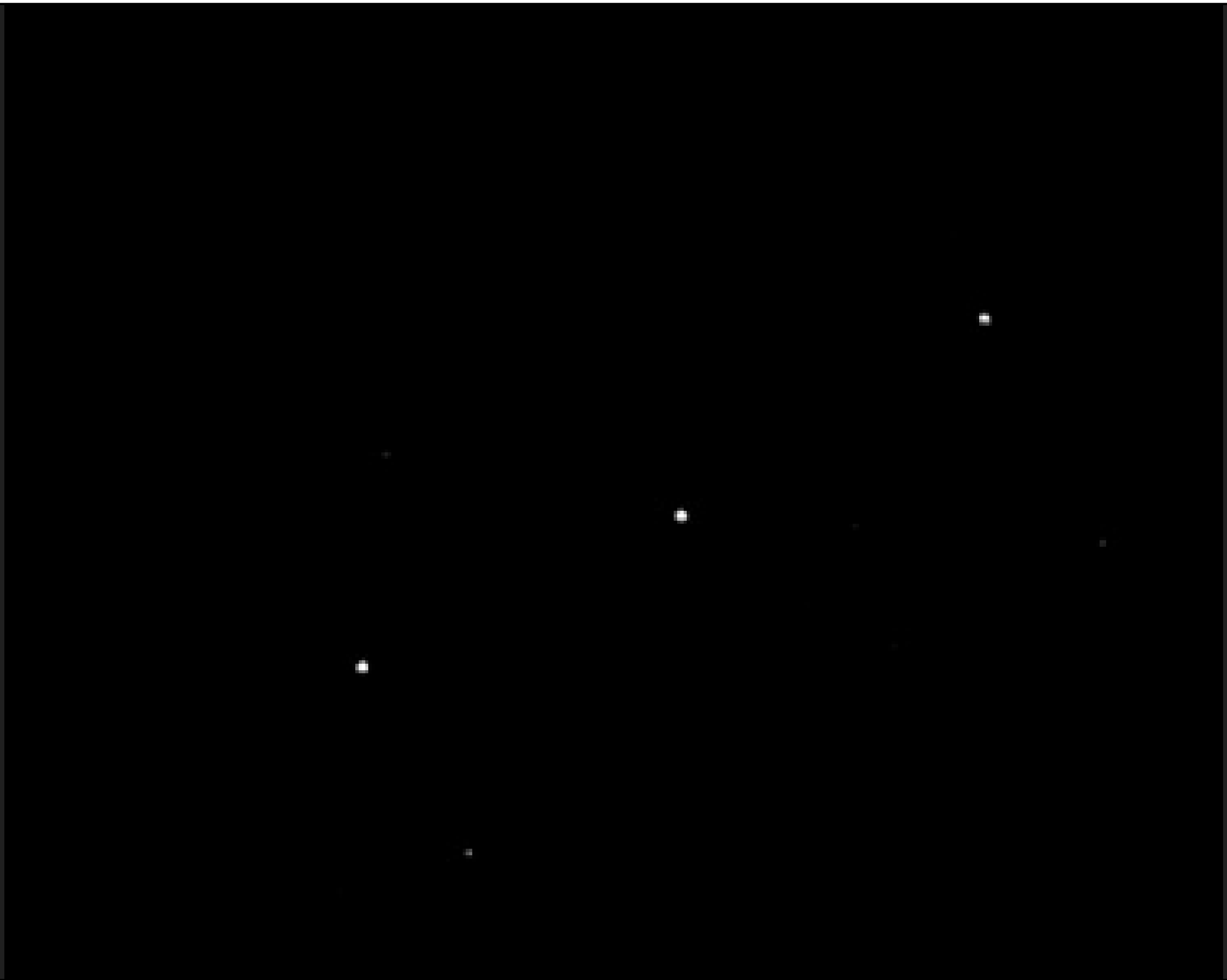
- ❑ **Max gain, 200ms exposure, 1280x1024 px, jpeg, HF12.5HA-1B lens**

IDS Sky View

- ❑ **Max gain, 200ms exposure, 2592x2048 px, jpeg, HF35HA-1B lens**

IDS Sky View (belt zoom)

- ❑ **Max gain, 200ms exposure, 2592x2048 px, jpeg, HF35HA-1B lens**

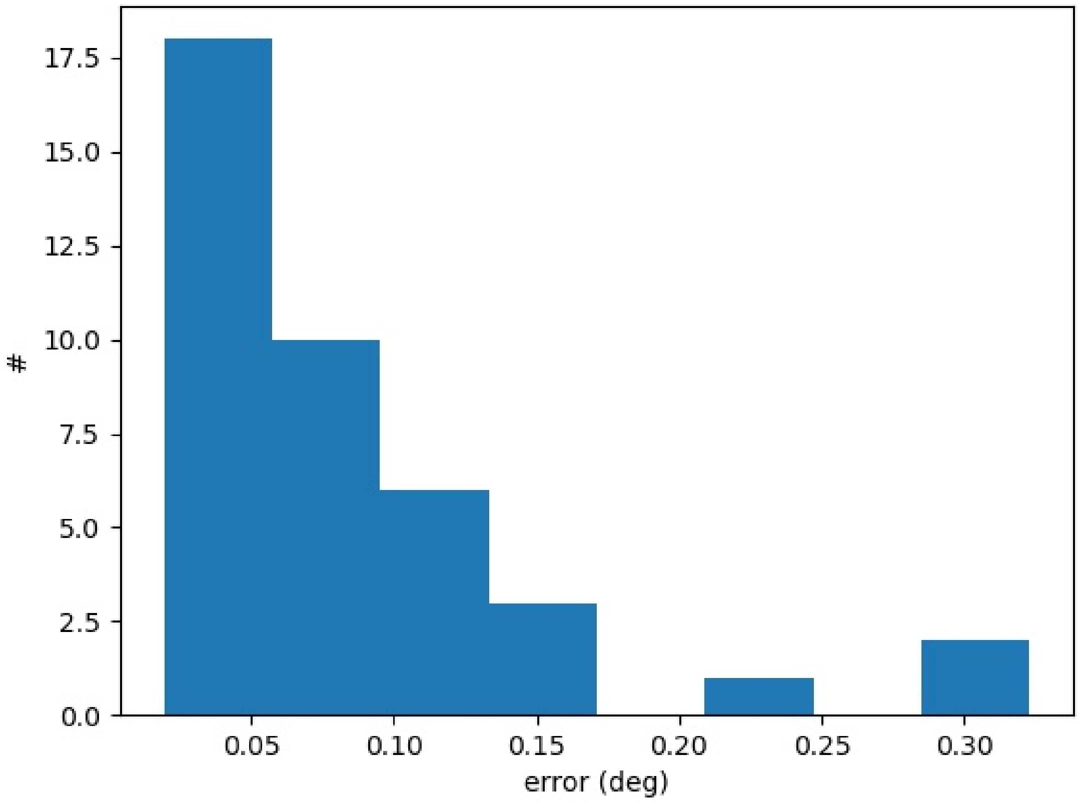


Performance

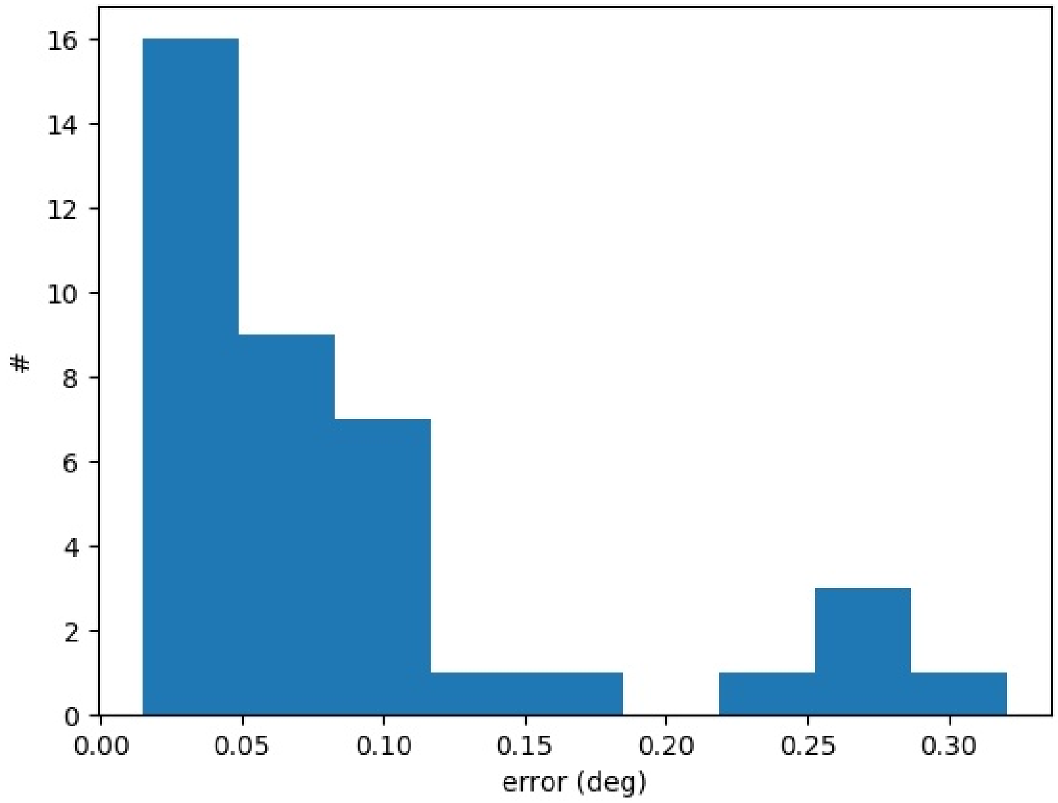
- ❑ **Two data sets available for accuracy, run time, and resource assessment**
  - Houston Orion Test Hardware (HOTH) rig data set
    - 41 tiff images (including darkframe)
    - Associated “truth” data and camera cal file
  - ISS DTO data set (250ms exposure)
    - 109 tiff images (including darkframe)
    - Associated “truth” data and camera cal file
    - “truth” data is created from another computer vision algorithm and shows a bias
- ❑ **Live-sky available for run time and resource assessment**
  - All sets taken at JSC with camera pointed at human-visible stars
  - 104 ximea jpeg images (not including darkframe)
  - 112 IDS jpeg images (not including darkframe

HOTH data accuracy

Windows machine HOTH imagery total error (deg)

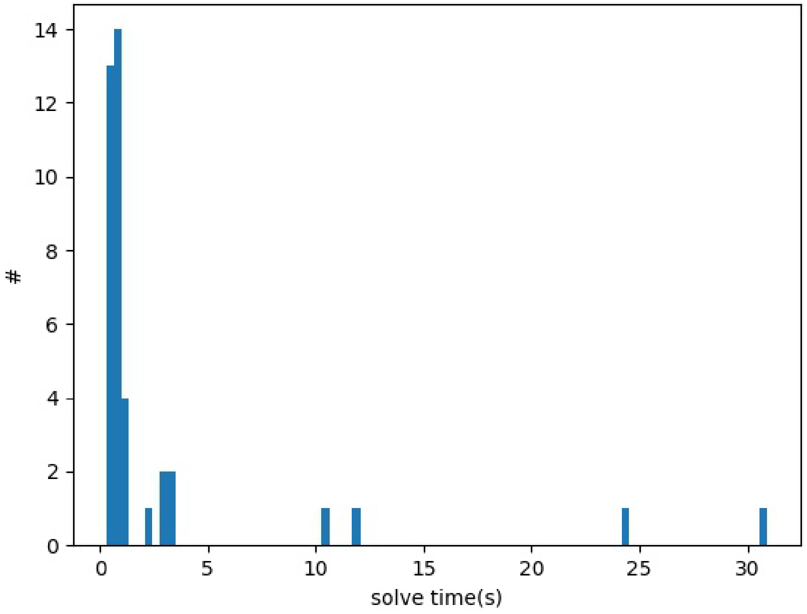


Odroid HOTH imagery total error (deg)

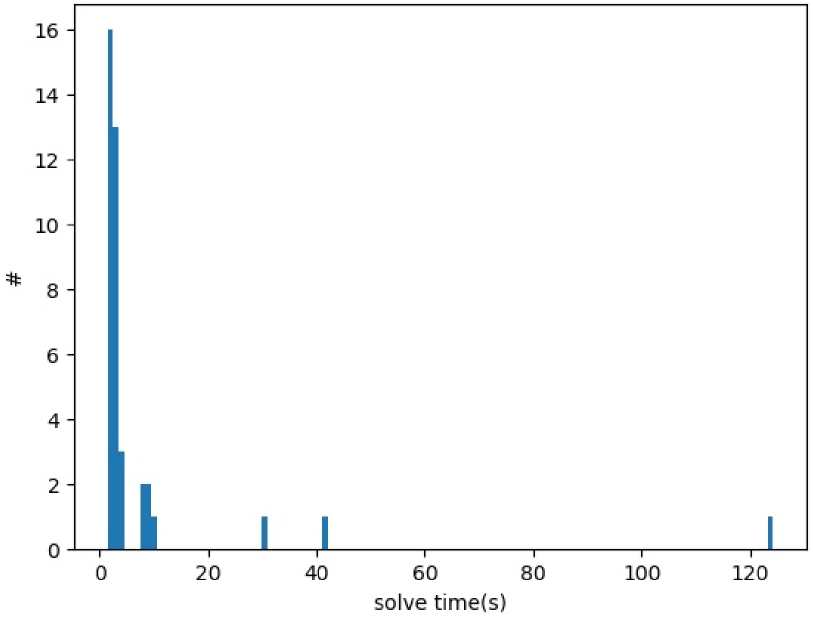


HOTH data solve time

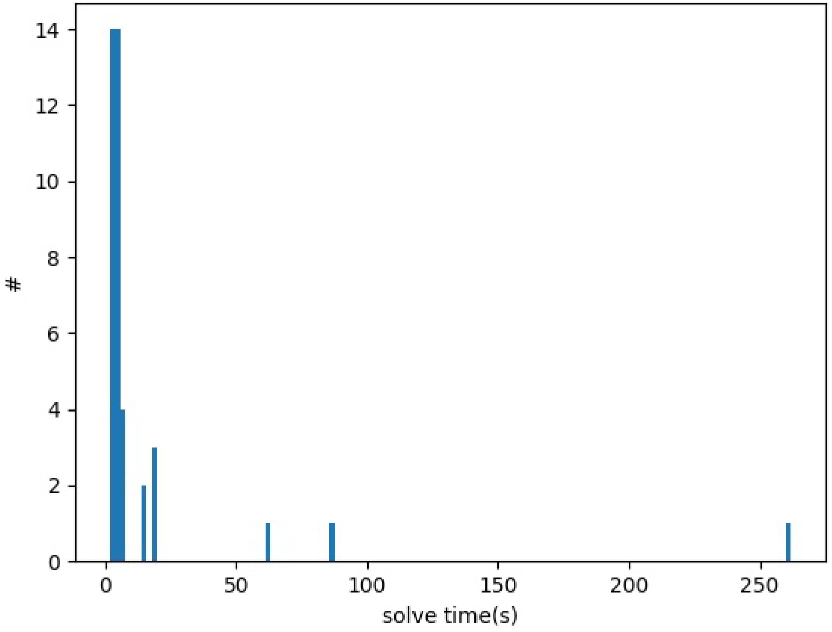
Windows machine HOTH imagery solve time (s)



Odroid HOTH imagery solve time (s)



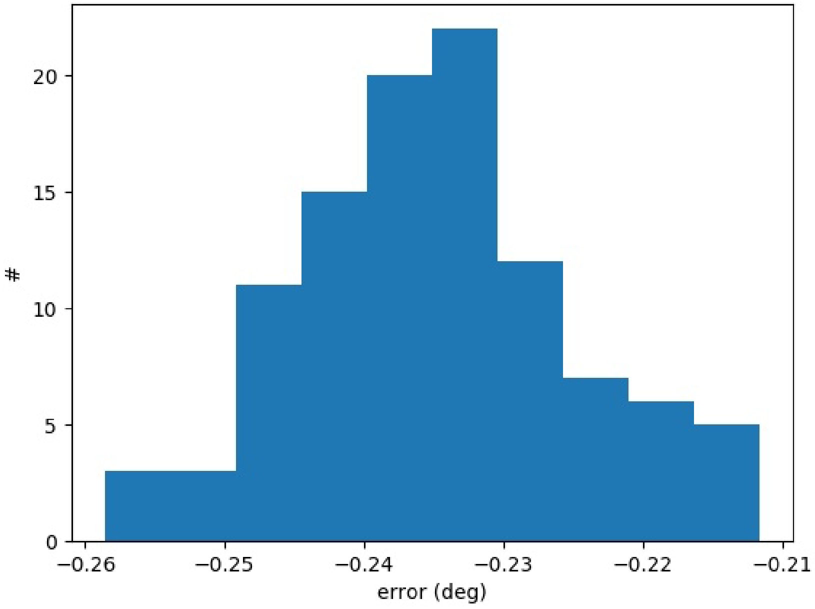
Pi3 HOTH imagery solve time (s)



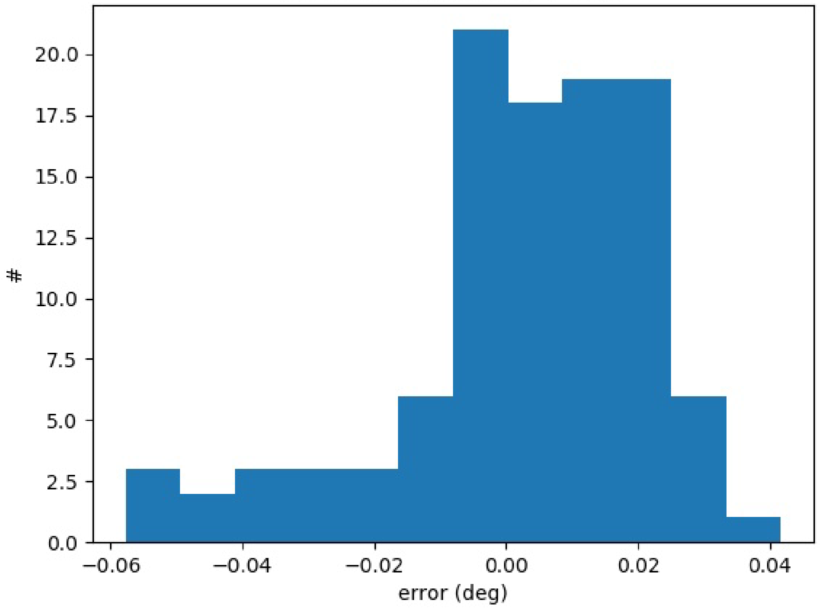
ISS DTO data accuracy

❑ **Component-wise view, highlighting bias issue in “truth” data**

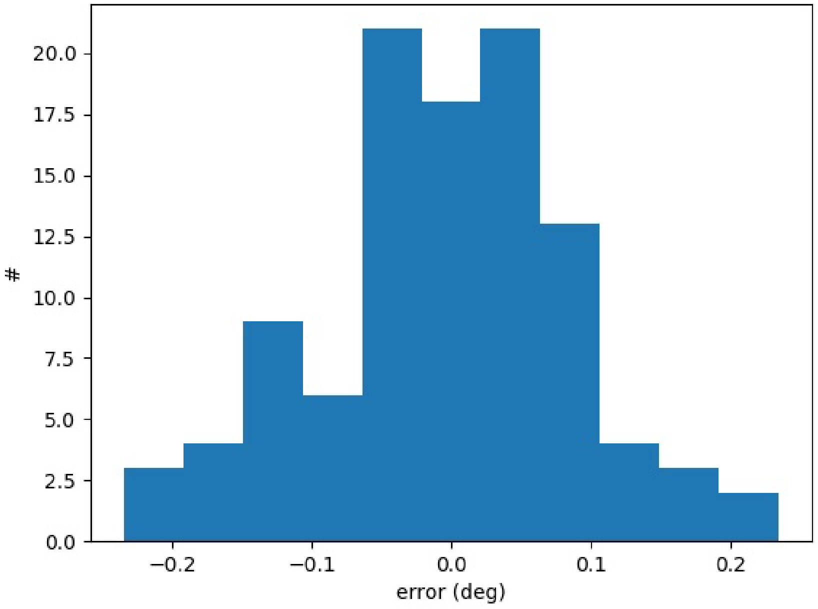
Odroid ISS imagery X error (deg)



Odroid ISS imagery Y error (deg)

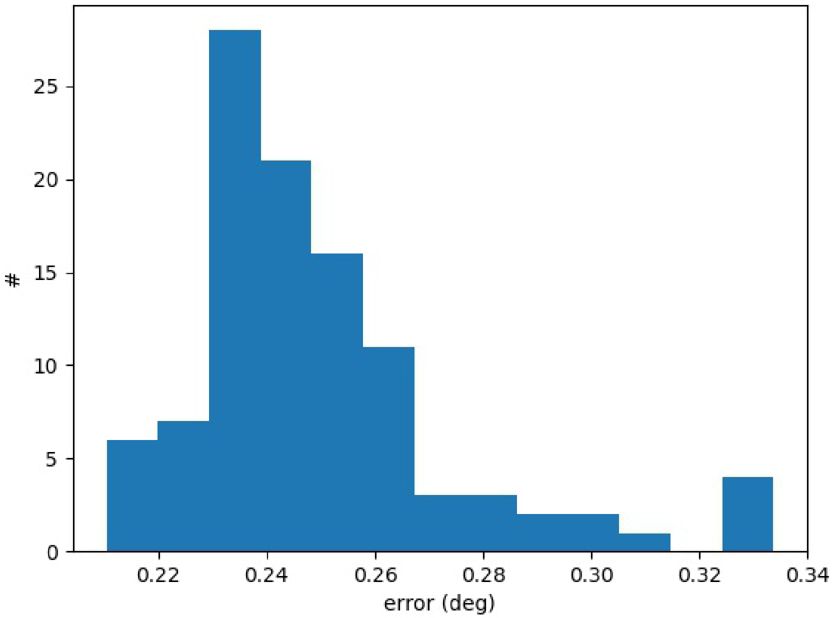


Odroid ISS imagery Z error (deg)

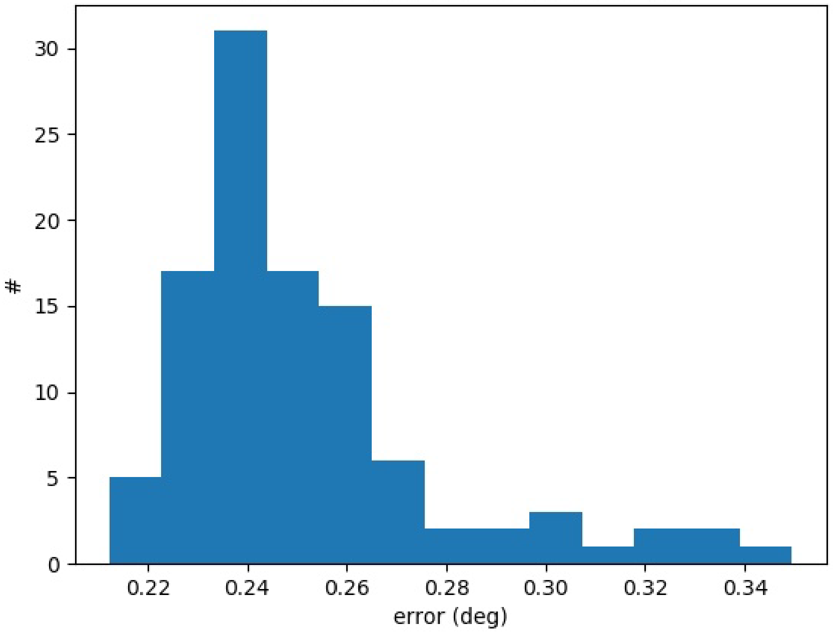


ISS DTO data accuracy

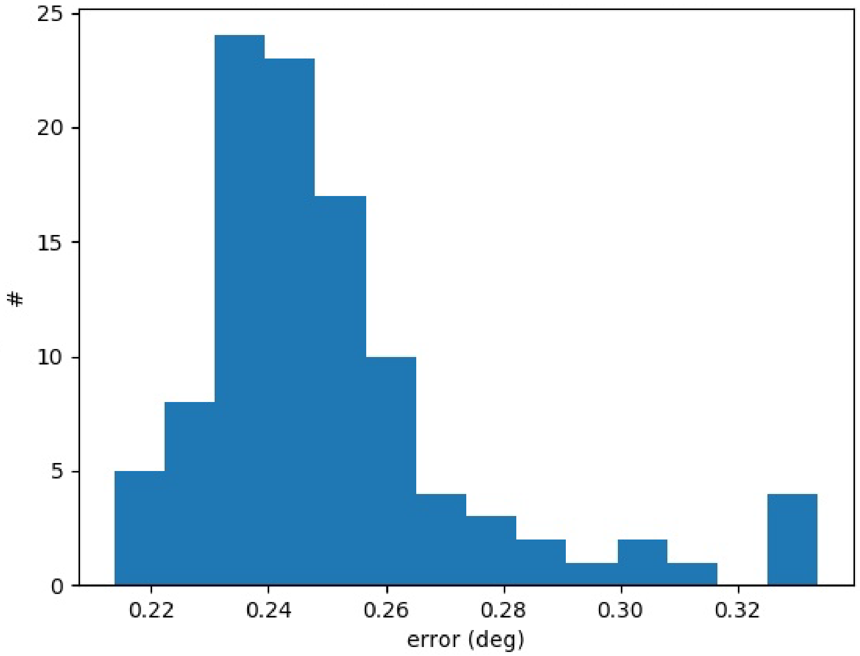
Windows machine ISS imagery total error (deg)



Odroid ISS imagery total error (deg)

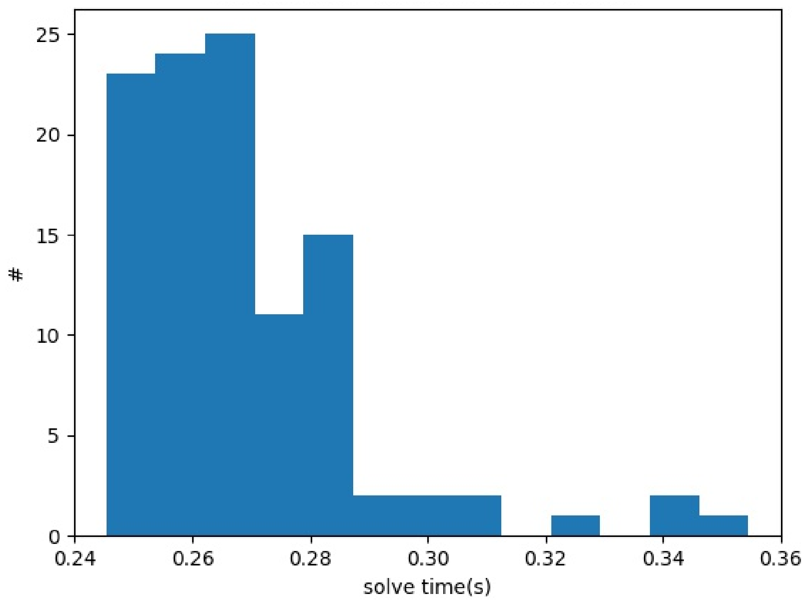


Pi3 ISS imagery total error (deg)

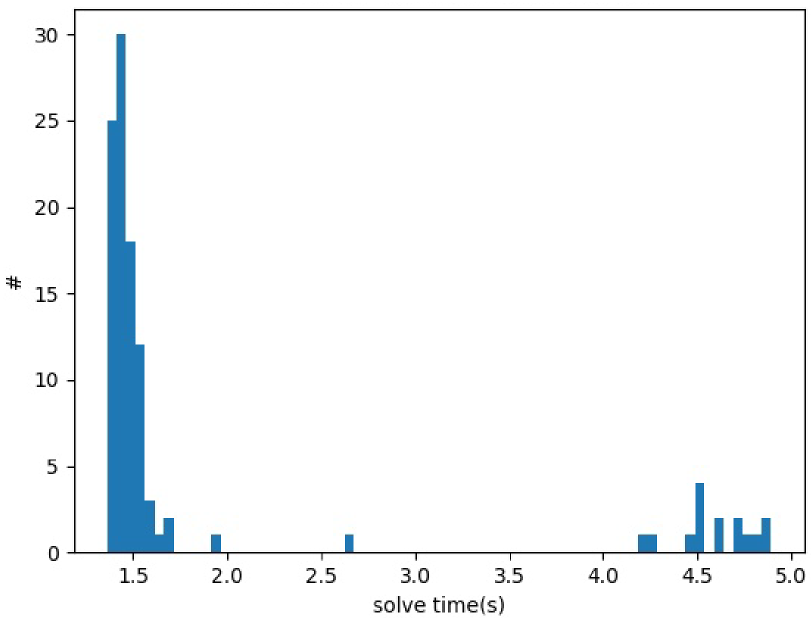


ISS DTO data solve time

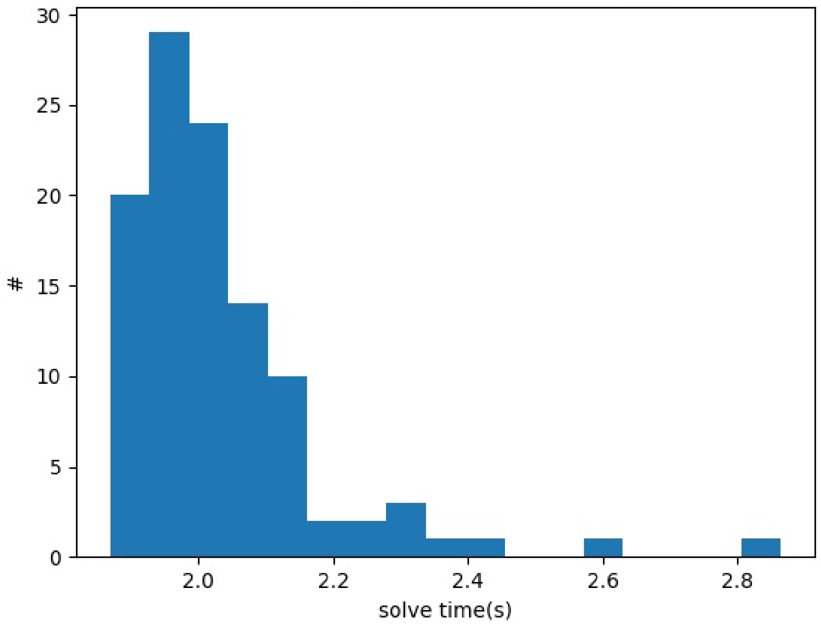
Windows machine ISS imagery solve time (s)



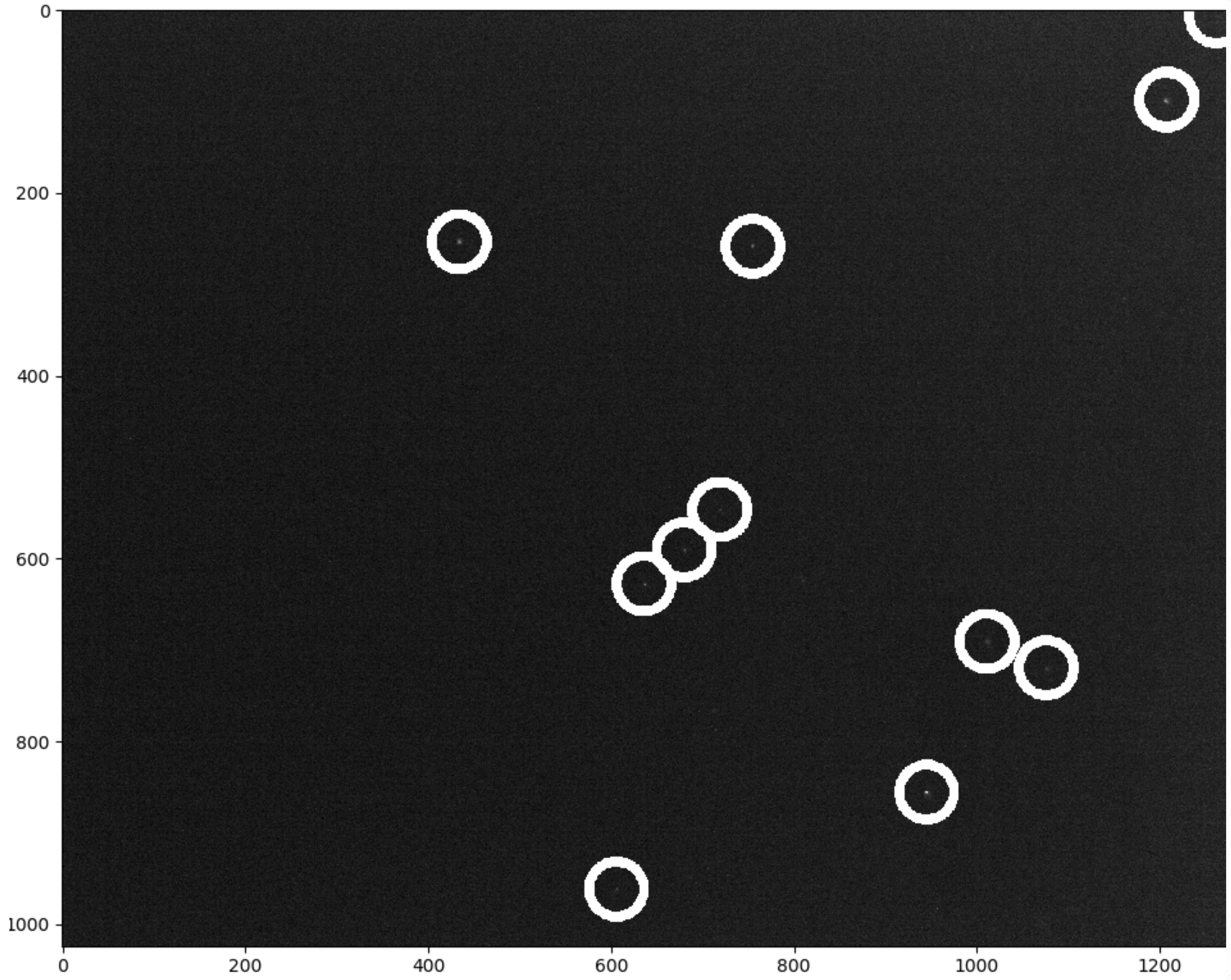
Odroid ISS imagery solve time (s)

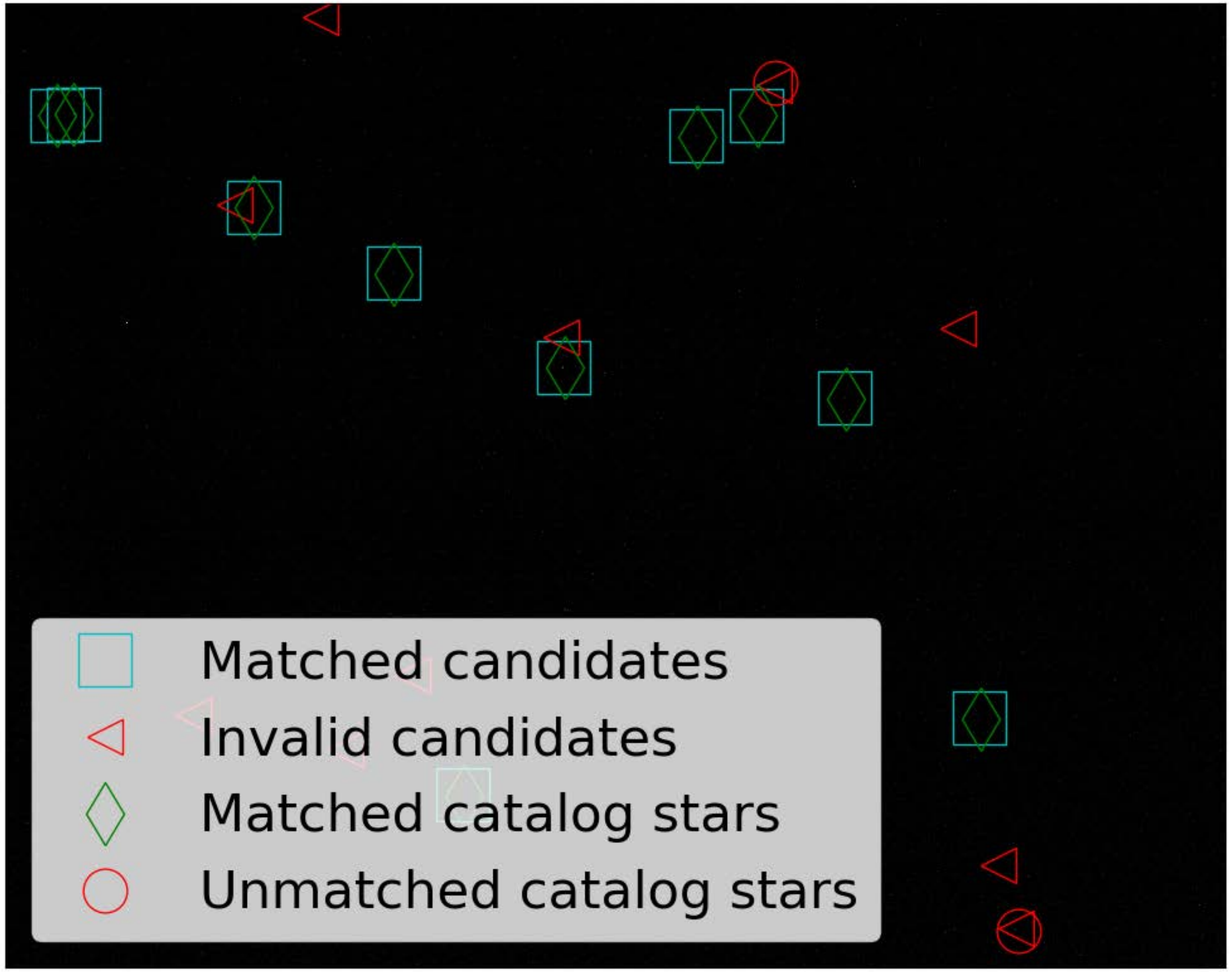


Pi3 ISS imagery solve time (s)



Graphical Output





Challenges

- ❑ **Camera calibration**
- ❑ **Getting pictures of stars**
  - Camera drivers (UVC is insufficient)
  - Houston weather and light pollution
- ❑ **Python package variability**
  - Allows for great flexibility of hardware/OS, but leads to variation in performance and additional code
- ❑ **Parameter selection**
  - Catalog brightness threshold
  - Number of matched stars
  - Star match pixel tolerance
  - Low-pixel threshold
  - Minimum star size threshold

Forward Plan

- ❑ **Q2 CY 2020: More automation**
  - Better installation
  - Automated camera cal
  - Automated parameter selection
- ❑ **Q3 CY 2020: Code cleanup**
- ❑ **Q3 CY 2020: More HWIL demonstration**
- ❑ **Q4 CY 2020: More SBC/camera integration/demonstration**
- ❑ **Q4 CY 2020: Open-source release**
- ❑ **Q2 CY 2021?: Flight demonstration**

Acknowledgments

Special thanks to:

- ❑ **The JSC Technology Working Group**
- ❑ **Steve Lockhart**
- ❑ **And more!**

Backup

Computer Specifications

Computer	Operating System	CPU	RAM	Disk	Python version	OpenCV version
Odroid XU4Q	Ubuntu 18.04	Samsung Exynos5422 (Cortex-A15 and Cortex-A7)	2GB LPDDR3	64GB SanDisk Extreme U3 card	2.7.17	3.2.0
Raspberry Pi 3B+	Raspbian Stretch	Broadcom BCM2837B0, Cortex-A53	1GB LPDDR2	64GB SanDisk Extreme U3 card	2.7.13	2.4.9.1
2018 Dell Precision 7720	Windows 10	Xeon E3-1535M	32 GB DDR4	512 GB NVMe PCIe SSD	3.6.4	4.0.0

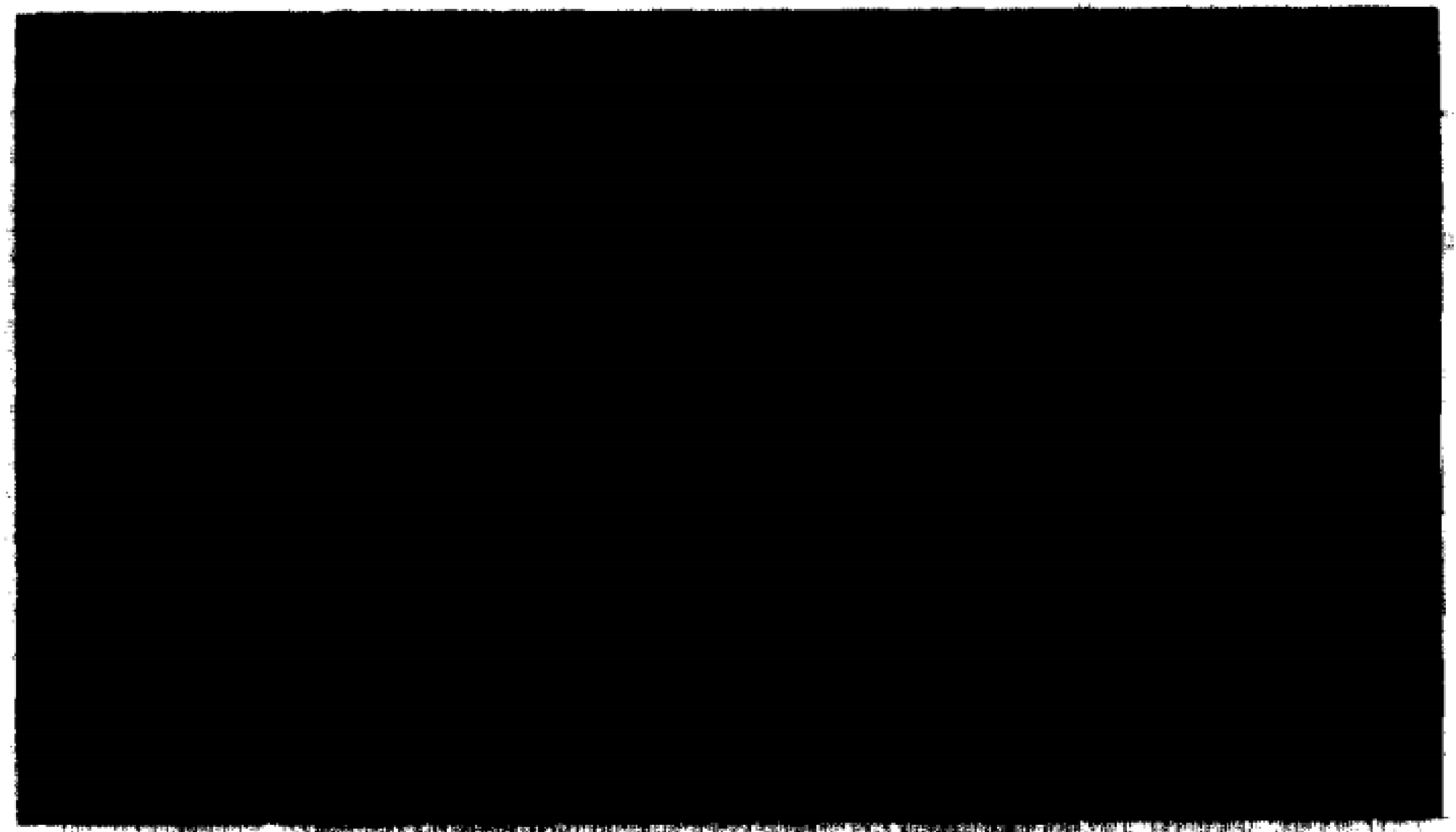


Fig. 2. A Typical Image from a Star Tracker

Figure 2 pictures a typical image acquired from a CCD