15/04/24

Demonstrating Visual-Inertial A&OD & On-Orbit Edge Computing

Progress summary

17 days before May 1st

Updates

- **Estimation**
  - Continued experimenting for the star tracker
    - Unsuccessful at capturing images of stars using the existing camera
      - Tried capturing in regions with darker skies
    - Benchmarking study
      - Looked at other Cubesat projects, camera + lens used
      - Hobbyist all-sky camera builds using raspberry pi
      - Available cameras in Rex lab
    - Identified few potential camera options, waiting to test them out under clear skies

Blockers

- Testing dependent highly on clear skies. Been difficult

Weekly Plan

- Estimation
  - Capture images of stars using the alternate camera options
  - Run the images through Openstartracker and the LOST star tracker pipeline

Interface dependencies

Some Camera Options

	 Rpi HQ Camera	 Arducam 64MP camera	 Rpi Camera Module 3
Sensor	IMX477R	OV64A40	IMX708
FOV	Dependent on lens used	84°(D)×68°(H)×56°(V)	75°(D)x66°(H)x41°(V)
Pixel Size	1.55µm × 1.55µm	1.008 µm x 1.008 µm	1.4µm × 1.4µm
Low Light operation	Yes	Yes	Yes
Used for astrophotography	Yes, used to build all-sky cameras	Yes	Limited

1/04/24

Demonstrating Visual-Inertial A&OD & On-Orbit Edge Computing

Progress summary

30 days before May 1st

Updates

- **FSW**
  - Functional camera interface that can:
    - store and retrieve time stamped images
    - check operational status and logs different errors
    - Adjust exposure time of camera
    - Check for blinding from the sun
  - Worked with avionics on low-level faults detection and handling for critical components
- **Estimation**
  - Continued experimenting with openstartracker
    - Explored tools to generate simulated night sky images to be used for testing
    - Used Arducam to capture images of the night sky on top of Flagstaff hill, couldn't capture images of stars
    - Continued testing of openstartracker with synthetic night sky images generated from different tools
  - Initial testing of MEKF with satellite playground.jl, working

Blockers

- Unable to capture images of stars with Arducam
  - Testing dependent highly on clear skies. Been difficult
  - Might need access to really dark skies to be able to capture usable images

Weekly Plan

- FSW development
  - Continue development of camera interface and integrate with other subsystems
- Estimation
  - Try again to capture images with arducam, use different method to change exposure time, analog gain of the camera
  - More rigorous testing of openstartracker using simulated night sky images

Interface dependencies

Testing with Simulated Night Sky Images

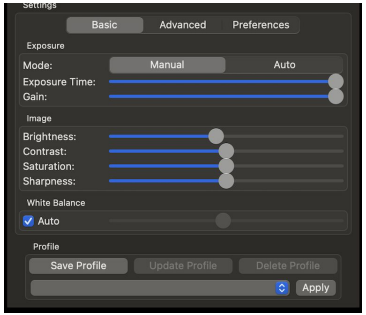
Tool	Notes	Calibration data using openstartracker	Inference using openstartracker
star_simulator	Allows adjustment of focal length, camera pixel size	Yes	No
lost	Lots of customization, noise, camera specifics, more realistic images	Yes	Initial testing yielding accurate results.
Stellarium	No imaging system only customization option, tied to telescope	-	-
Kstars	Preliminary configuration didn't generate results, needs more troubleshooting	-	-

Next steps:

- Generating simulated images through lost most promising
- Continue generating more images and testing with it

Actual Night Sky Images captures with Arducam

1. Sky was clearest on Friday, captured images on Flagstaff fill. Nothing visible in arducam images, even after post processing. Phone could capture images of stars.
2. Played around with exposure time, analog gain, brightness, hue, saturation, white balance values of the camera.
    a. Used a Camera controller app in Mac to do that
    b. Provides scales to adjust from min to max values



Next steps:

- Use v4l-utils to modify camera parameters. Set specific values, more control over camera
- Try again on a clear night

25/03/24

Demonstrating Visual-Inertial A&OD & On-Orbit Edge Computing

Progress summary

38 days before May 1st

Updates

- **FSW**
  - Incorporated error logging capturing various errors with predefined error code messages in camera interface.
  - Camera interface executes command, checks operational status, real time video feed and stores images.
  - Camera interface can zoom, focus and change exposure time
  - Updated detumbling control task to handle sensor, magnetorquer diagnostic status, system states, and battery status.
  - Tested packetization of bytestream and basic protocol over UART between jetson and argus.
- **Estimation**
  - Worked with the mechanical team to create a mount for taking appropriate camera images for the star tracker

Blockers

- Bidirectional broken unknown reason- main suspect bad solder- only able to send data from argus to jetson
- Hit with bad weather for capturing calibration images for the star tracker. Hoping to get some good images in this week and complete testing of the tool

Weekly Plan

- FSW development
  - Continue development of camera interface and test with camera
  - Refine communication protocol with looser guarantees as automatic flow control and parity check is performing well
- Estimation
  - Complete taking images with camera, testing the star tracker tool
  - Complete MEKF testing in Sim

Interface dependencies

18/03/24

Demonstrating Visual-Inertial A&OD & On-Orbit Edge Computing

Progress summary

44 days before May 1st

Updates

- **Estimation**:
  - Familiarized with openstartracker
  - Identified and resolved issues encountered during the setup process
  - Achieved functionality with sample images provided in the tool
  - Captured night sky images on phone camera for calibration
  - Created a pre-processing script that detects stars, makes the background dark
  - Ran calibration and inference tests for captured images
- **FSW**
  - Refined camera interface requirements and restructured code
  - Tested packetization and comm protocol for PyCubed and Jetson

Blockers

- Waiting for UART adapter for Jetson for testing on UART
- C firmware module for interrupts on TX and RX pins

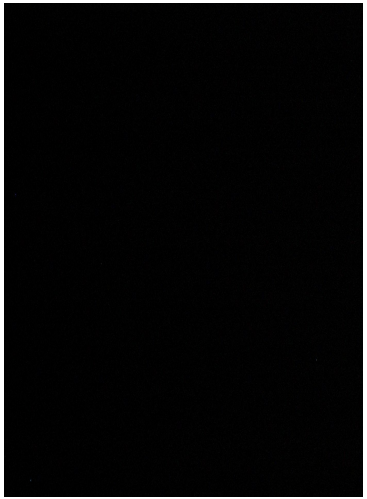
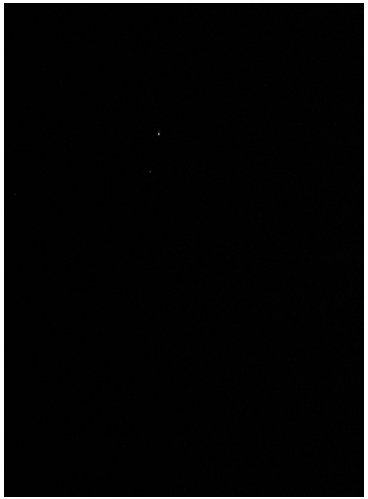
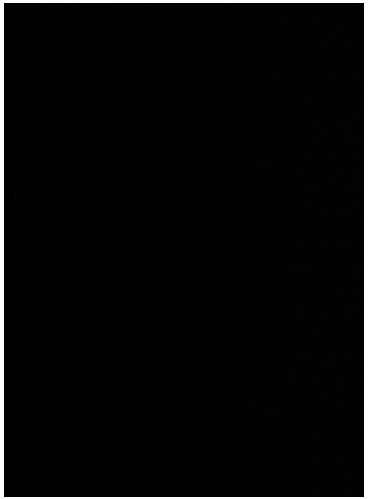
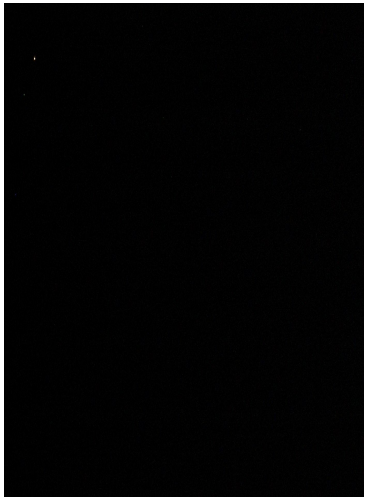
Weekly Plan

- Estimation
  - Capture appropriate images for calibration through actual camera for openstartracker
  - Run inference using the star tracker tool
  - Sun sensor calibration
- FSW development
  - Continue development of camera interface
  - Add logger interface for PyCubed

Interface dependencies

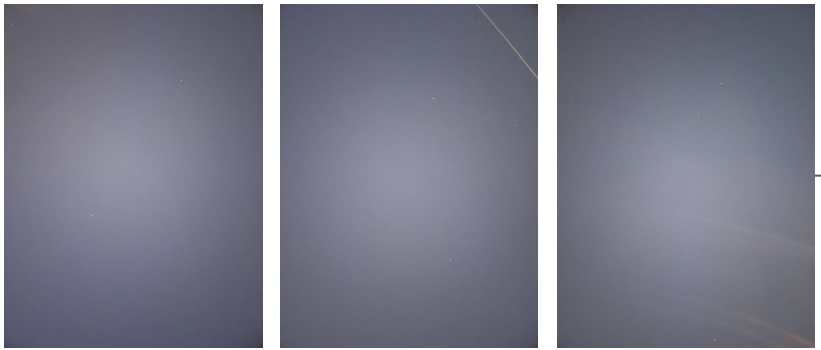
- Mechanical:
  Inertia measurements for the updated CAD model

Experiment 1 - Testing the tool's functionality using pre-supplied sample images within the tool's environment

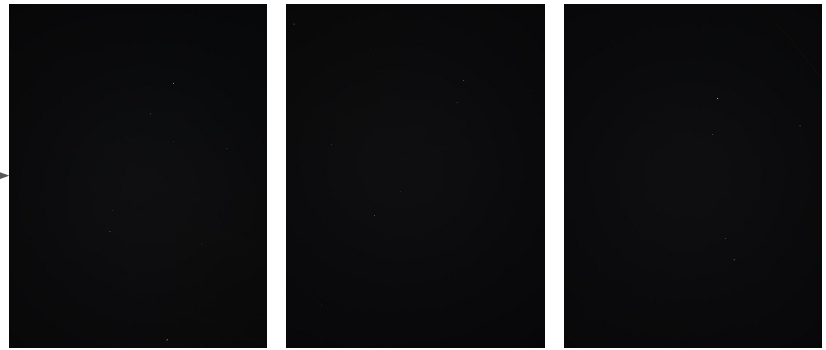
- Performed calibration test using the provided 9 sample images
- Calibration ran successfully
- Inference script generated RA (Right Ascension), Dec (Declination) and orientation results for the images

Experiment 2- Testing using images captured on phone



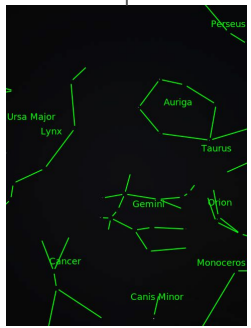
3 Images captured on phone camera

Pre - processing



Detected stars, made background darker

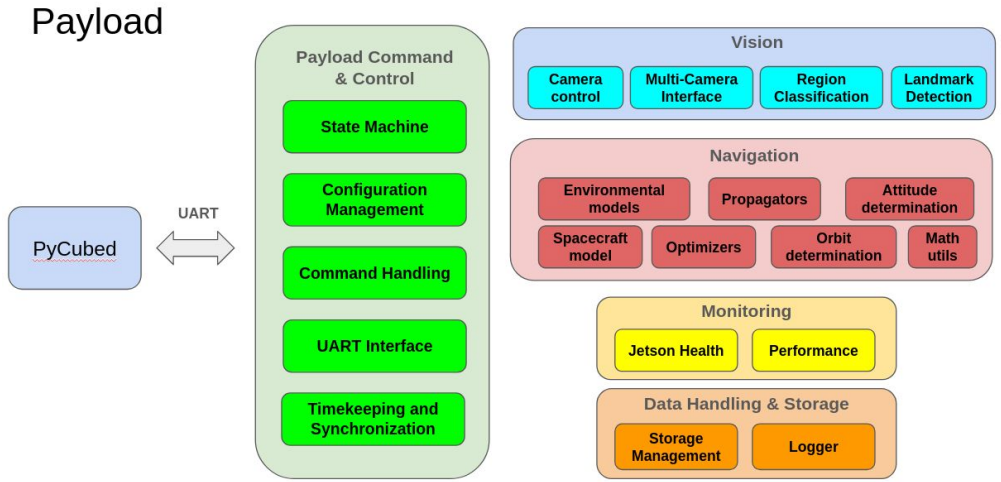
- Pre-processed the captured images
- Performed calibration test using the three images
- Only one image provided calibration data in both Astrometry.net and the OpenStarTracker tool
- Inference script did not generate RA (Right Ascension), Dec (Declination), or orientation results for the images



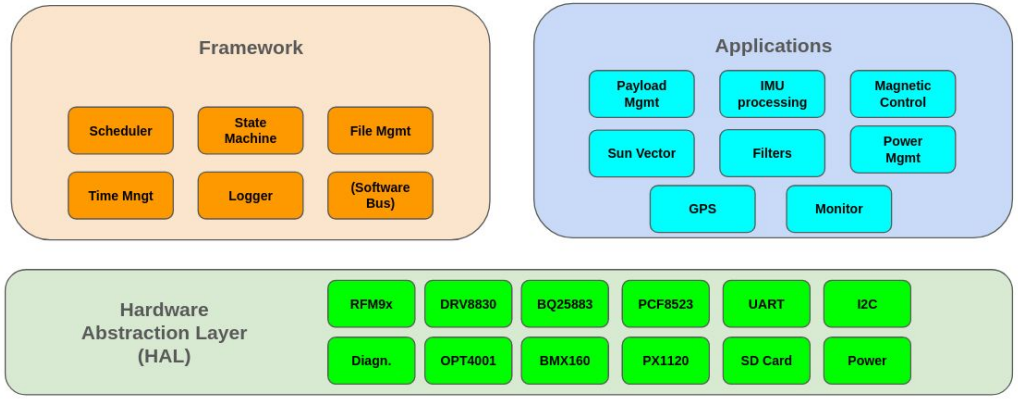
Next steps:

- Capture better images for calibration- More images of different parts of the sky
- Ensure that all of them consistently solve and yield calibration data on astrometry.net
- Run unit tests again

Flight Software



PyCubed



10/03/24

Demonstrating Visual-Inertial A&OD & On-Orbit Edge Computing

Progress summary

51 days before May 1st

Updates

- **Estimation**:
  - Studied MEKF resources and papers
  - MEKF for estimating attitude using IMU and sun sensor data
    - First-pass implementation completed
    - Sim integration and testing in progress
    - Recorded time stamped sensor data to use in MEKF implementation
- **FSW**
  - Architecture separation between the hardware interface layer (HAL) and the application layer
  - Configuration and state machine early architecture
  - Preliminary camera payload requirements

Blockers

Weekly Plan

- FSW development
  - PyCubed Time Distribution and Configuration
  - Jetson-PyCubed board communication
  - Sun Vector module (processing, calibration)
  - Camera interface improvement
  - Trained vision models ⇒ FSW implementation (vision)
- Estimation
  - Complete MEKF testing in Sim
  - Sun sensor calibration

Interface dependencies

- Mechanical:
  Inertia measurements for the updated CAD model
  - Need it now to update the sim

26/02/24

Demonstrating Visual-Inertial A&OD & On-Orbit Edge Computing

Progress summary

65 days before May 1st

Updates

- **FSW** development:
  - **Onboard File Storage**
    - Single interface to SD Card
    - File Management Services for every logging tasks
    - Interface for telemetry downlink
  - **Camera interface**
    - Read and configure all 6 cameras
    - Access to camera's status, latest image and live feed
    - Storage of time stamped images of each camera is created
  - **IMU interface**
    - Sample sensor at pre-set frequency
    - Support Moving Averaging Filter for smoothing the data
  - **Jetson-PyCubed Inter-communication**:
    - Complete protocol spec
    - Helper library built for packet parsing and creation
    - Initial implementation to read and send messages done
  - **State Machine Manager**
    - Preliminary design, implementation in progress

Blockers

Weekly Plan

- FSW development
  - PyCubed Configuration and Time Management & State Machines
  - Jetson-PyCubed command and control
  - Sun Vector module (processing, calibration)
  - Continue development on existing modules
- Estimation
  - MEKF for attitude estimation using IMU, magnetometer and sun sensor
  - Record time stamped sensor data to use in MEKF implementation.

Interface dependencies

- Mechanical:
  Inertia measurements for the updated CAD model, updated CAD of the CubeSat for test bed design

19/02/24

Demonstrating Visual-Inertial A&OD & On-Orbit Edge Computing

Progress summary

72 days before May 1st

Updates

- Vision:
  - Dataset download from 23 most salient regions
  - Landmark pruning for to identify ideal landmark size
  - Started hyperparameter tuning for LD
  - Looking into custom loss function focusing on pixel error
- Estimation integrated testing:
  - Generated camera vectors using landmark and satellite ground-truth through vector transformations
  - Tested batch optimizer to validate attitude estimation
  - Created small test setup to validate pixel to camera vector transformation
- Validation:
  - SIL environment setup between PyCubed and Simulation
  - First SIL test for detumbling control
  - Gyroscope noise analysis

Blockers

- Computing resources for LD training
  - ECE Community Compute Clusters
  - Pittsburgh Supercomputing Center
  - ROBO Cluster

Weekly Plan

- Vision
  - Continue training experiments with pruning
  - Continue dataset download
  - Tune hyperparameters and look into custom loss function
  - Working LD detector release by end of week
- Estimation
  - Continue development on batch optimiser
- FSW development
  - Work on PyCubed-Jetson communication
  - Finished PyCubed tasks for Alpha version

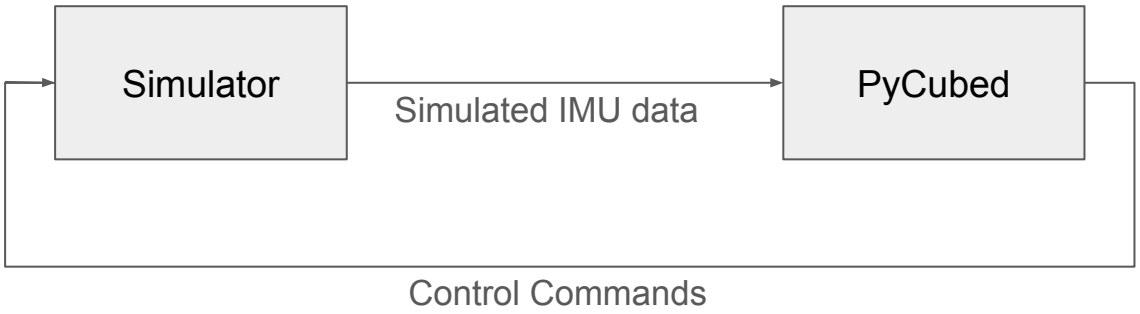
Interface dependencies

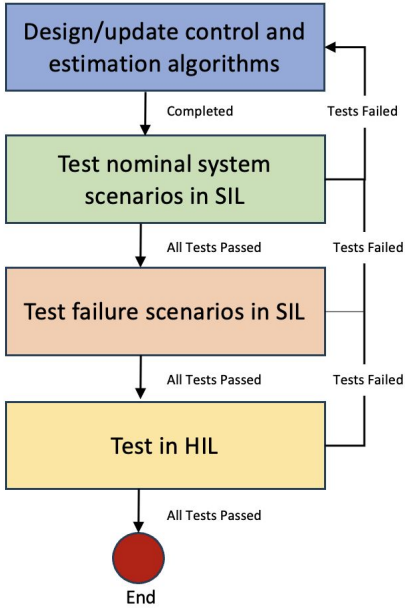
- Final CAD of the CubeSat for test bed design

Integrated testing

- Created small setup to test camera vector generation using image coordinates
- Verified transformation equations using actual x,y and depth information to find corresponding pixel coordinates and vice versa.
- Tested batch optimisation using groundtruth satellite ECEF and landmarks detected.
- Process
    - Get landmark lat long and convert to ECEF, get satellite groundtruth ECEF
    - Subtract the two vectors to get vector pointing from Landmark to satellite, invert to get vector from satellite to landmark
    - Convert this vector into camera frame to get camera vectors
    - Use ECI coordinates and these vectors as inputs to estimator

GNC Software Validation



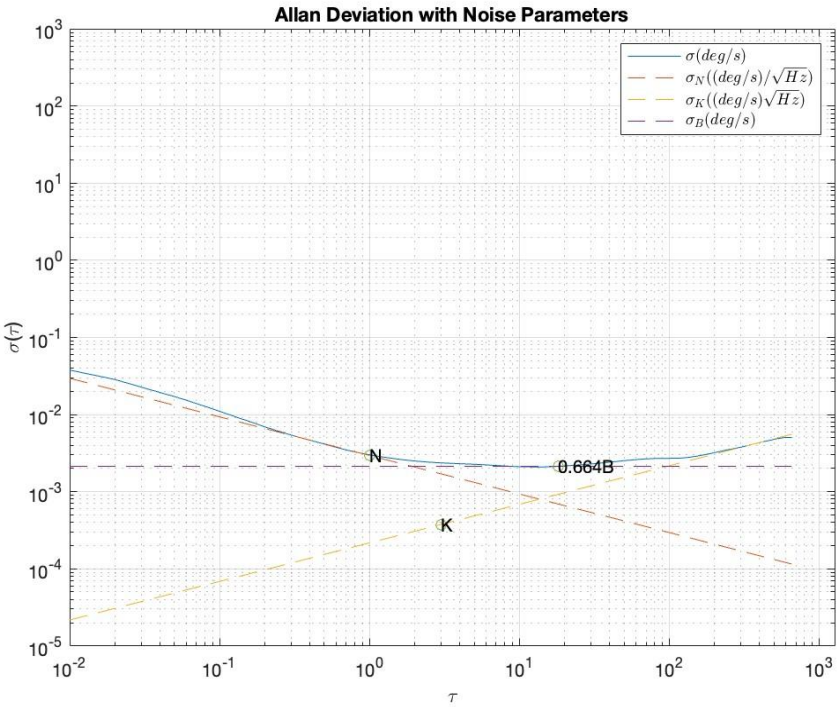


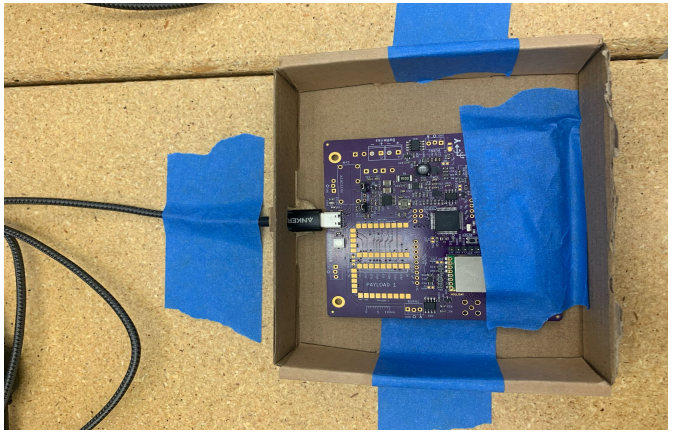
Software in Loop environment setup completed.

- Sensor readings generated in real-time on Simulator
- Readings are sent to PyCubed with Magnetic Control software running via Serial Communication
- Magnetic Control software processes the sensor readings and generates the control commands
- Simulator computes the next state for the satellite using the received control commands

Preliminary testing for Detumbling Control completed.

Gyroscope Noise Analysis





6 hours of datalogging of stationary IMU

- Allan variance for gyroscope noise parameter analysis, Parameters: N (angle random walk), K (rate random walk), B (bias instability)
- Helps us identify noise sources in stationary gyroscope data clusters
- Will be used for modelling the gyroscope accurately