HuskySat-2
Payload

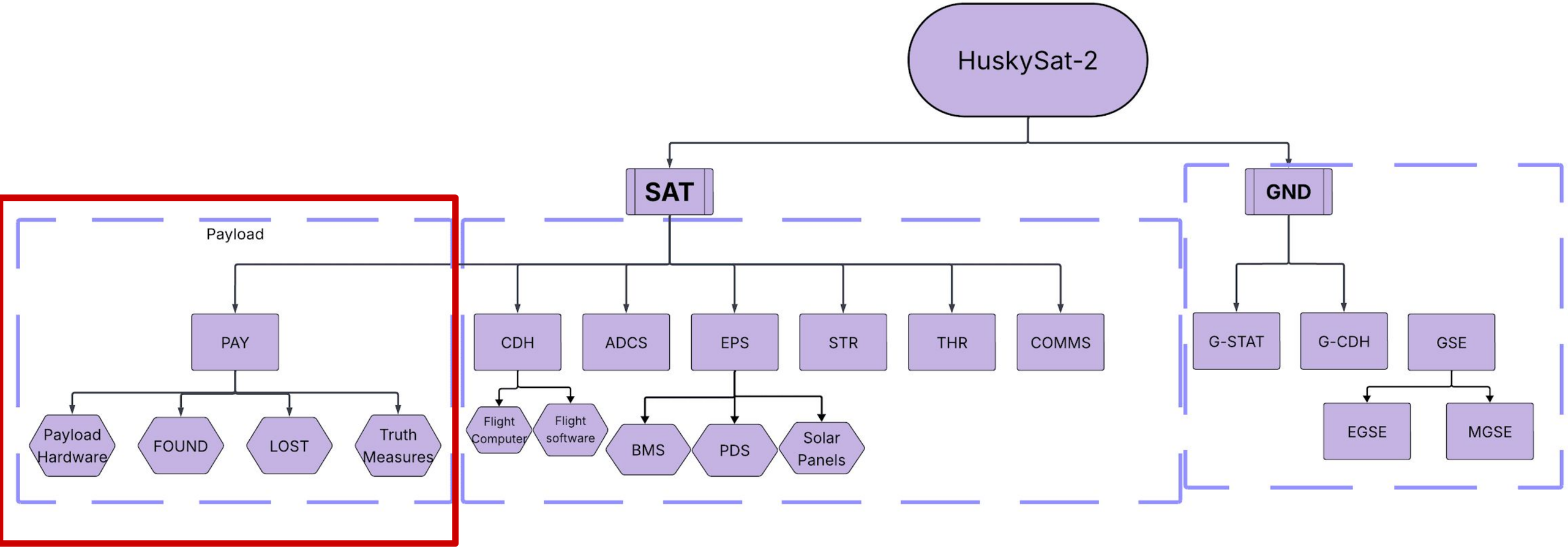
University Nanosatellite Program NS-12
Preliminary Design Review

Presented by: X, X, X

Introductions & Agenda

LOST	**Manasi Ganti** LOST Lead
FOUND	**Josh Lando** Payload & FOUND Lead
CONOPS	
Software	**Prince Gill**
Hardware	**Josh Lando**
Avionics	**Adam Vengosh**

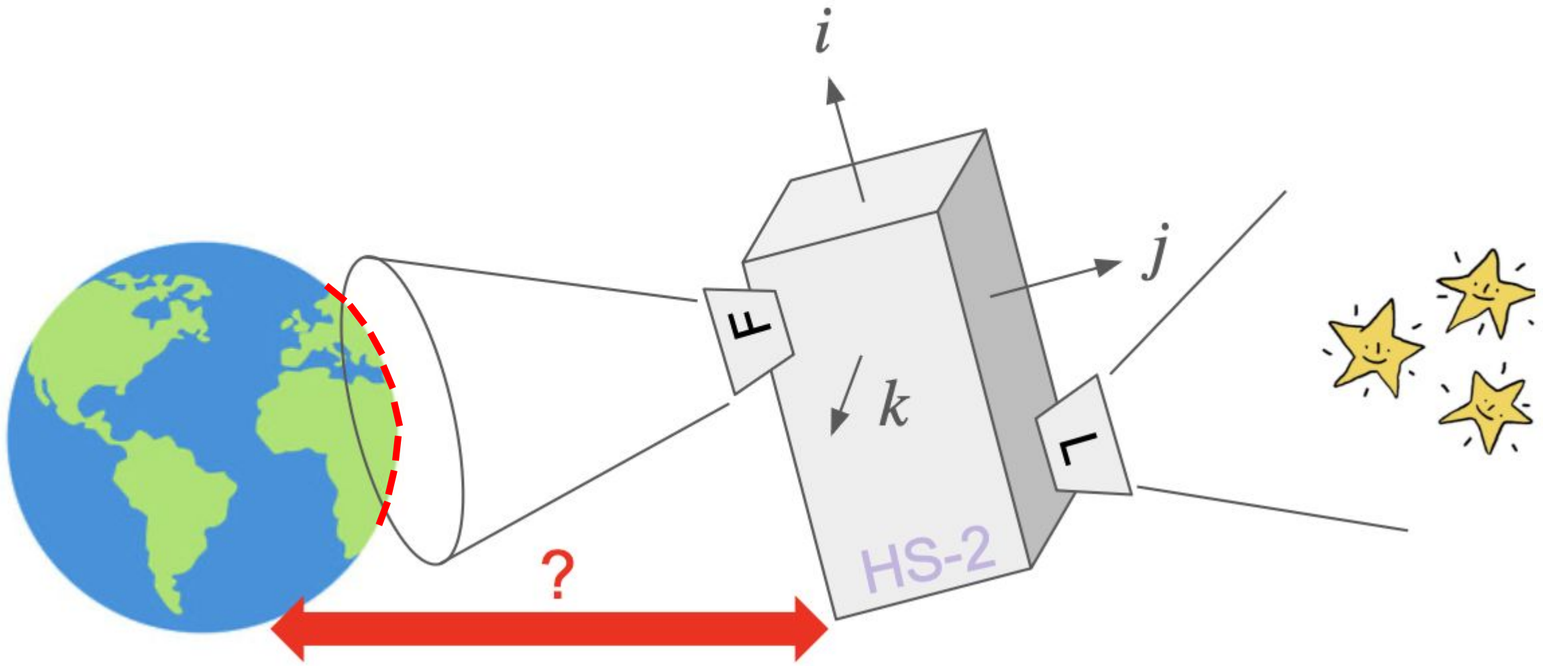
System Architecture



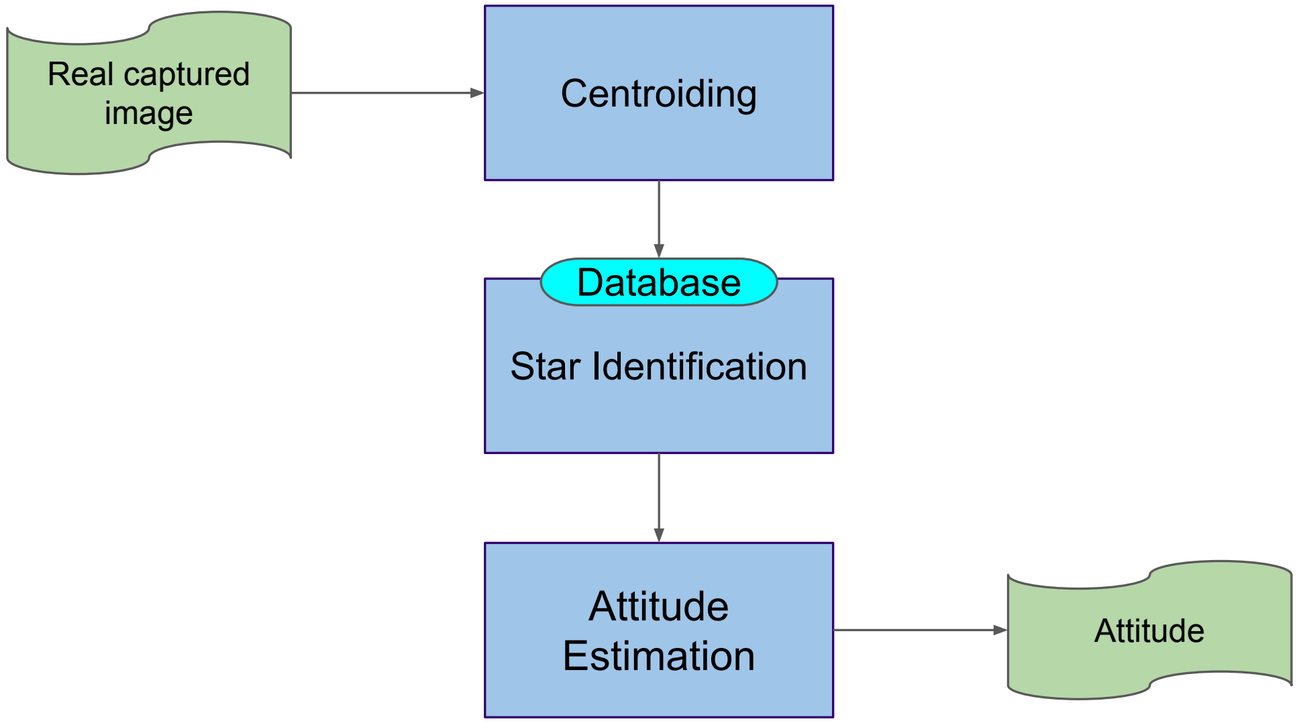
HuskySat-2 Science

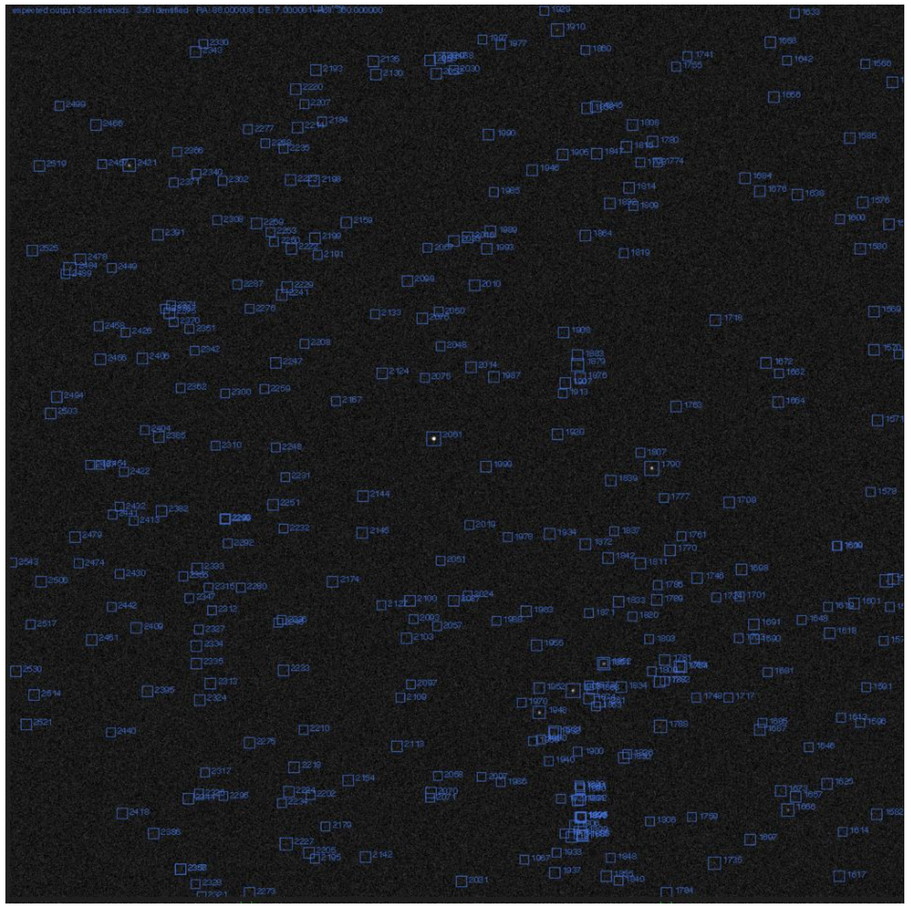
LOST: Open-source Star Tracker (LOST)

FOUND: Open-source Universal Navigation Determiner (FOUND)



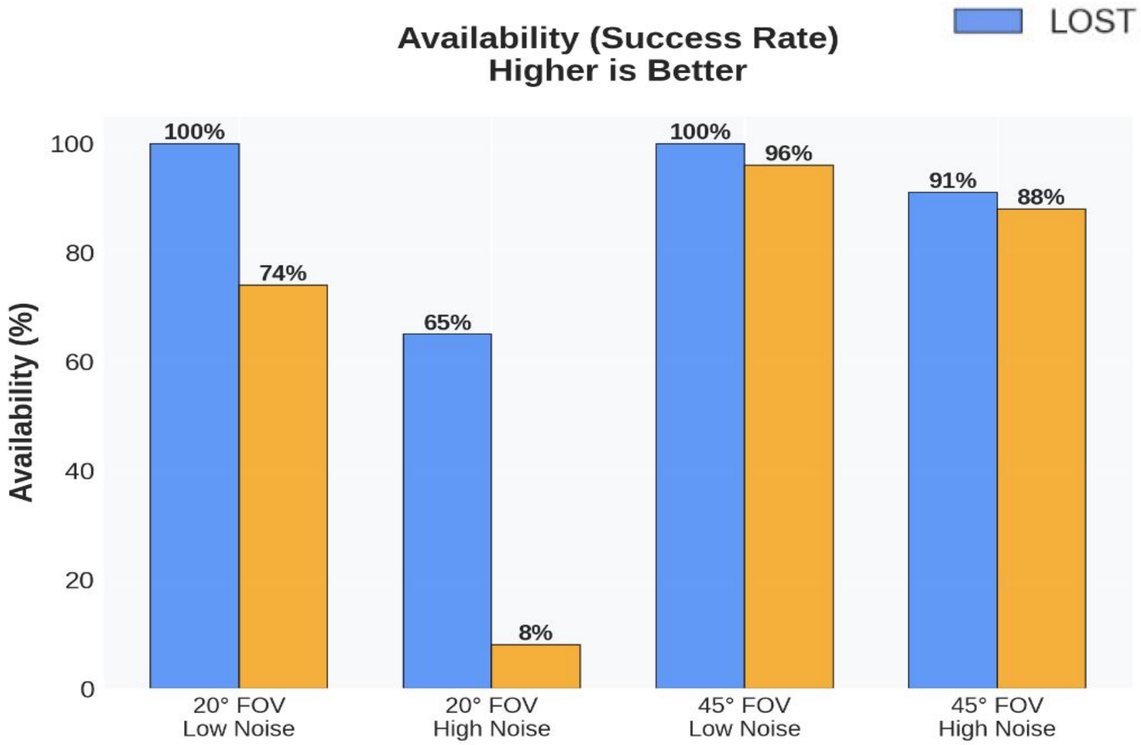
LOST - Open-source Star Tracker (LOST)

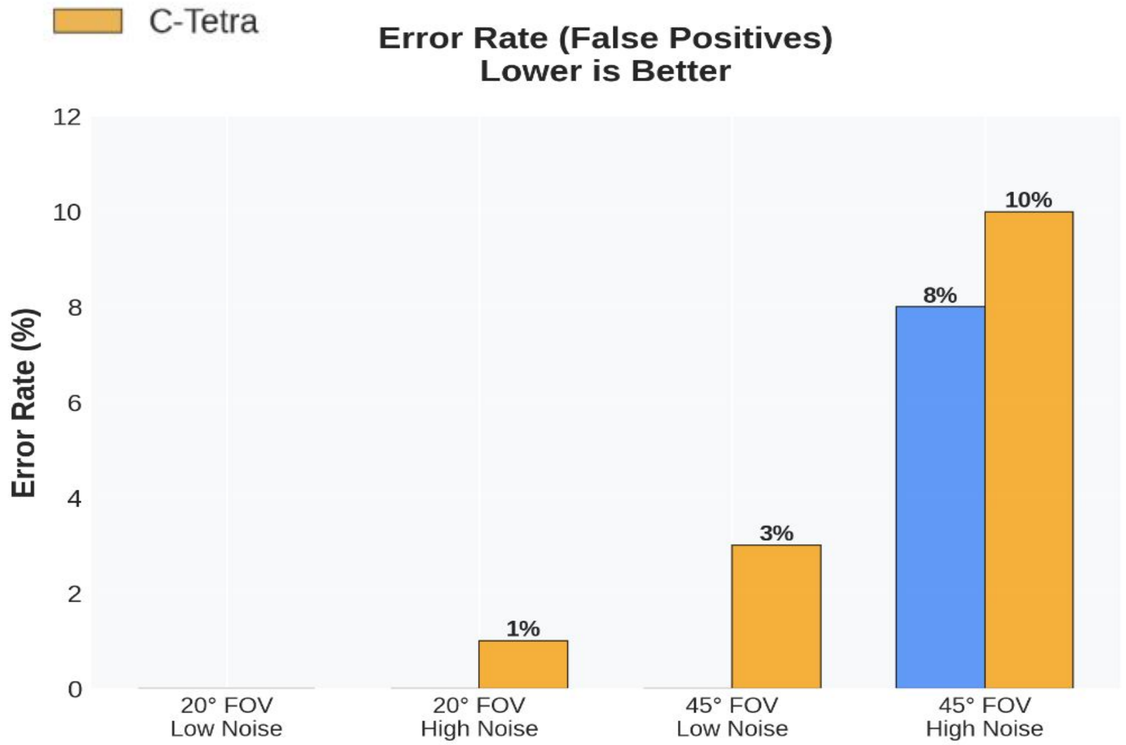




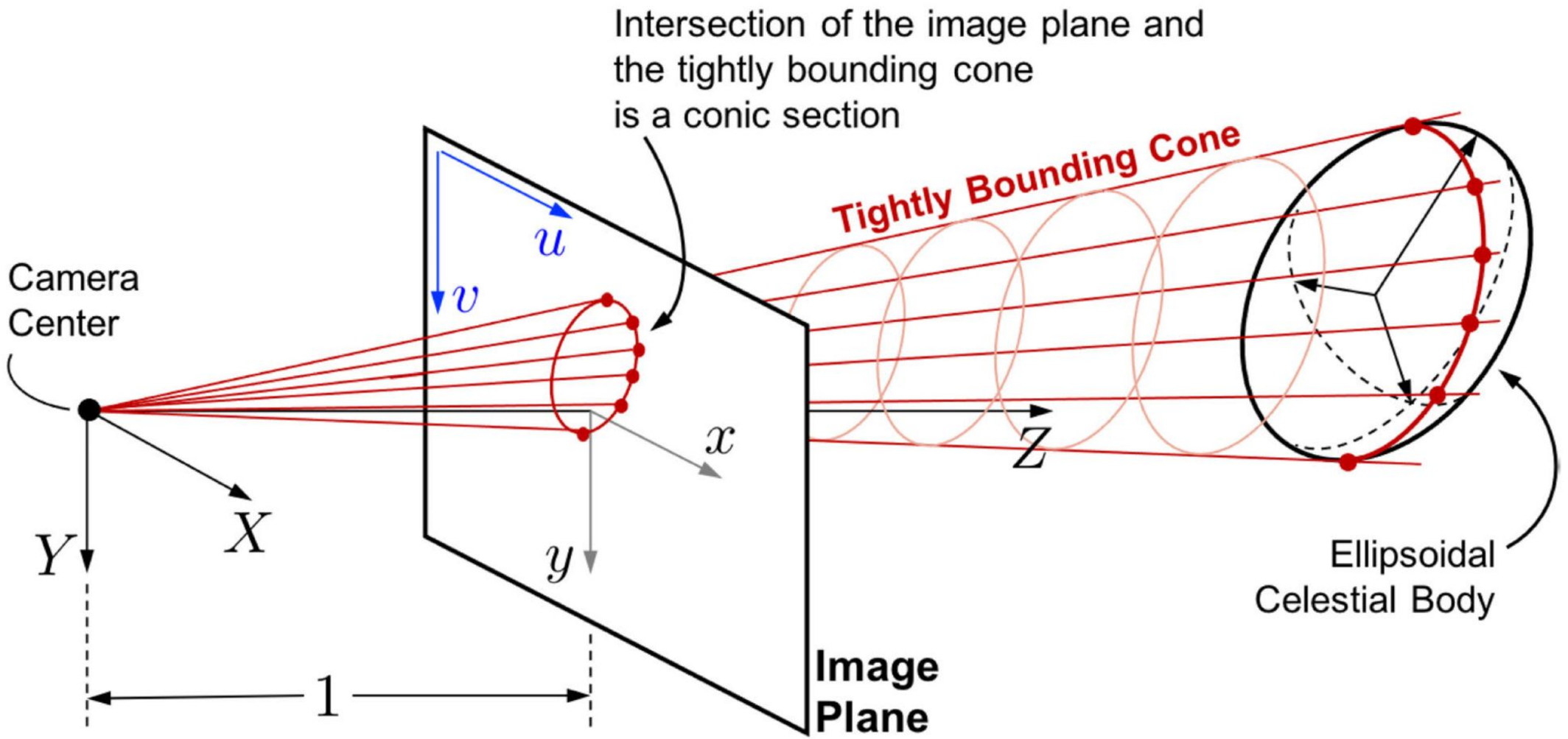
Why LOST?

LOST	Tetra-C	OpenStarTracker	SOST
- written in C++ , optimized for memory access	- requires a large database (hence, more storage)	- uses heavyweight libraries like OpenCV	- written in Python, which has high memory consumption and performance limitations

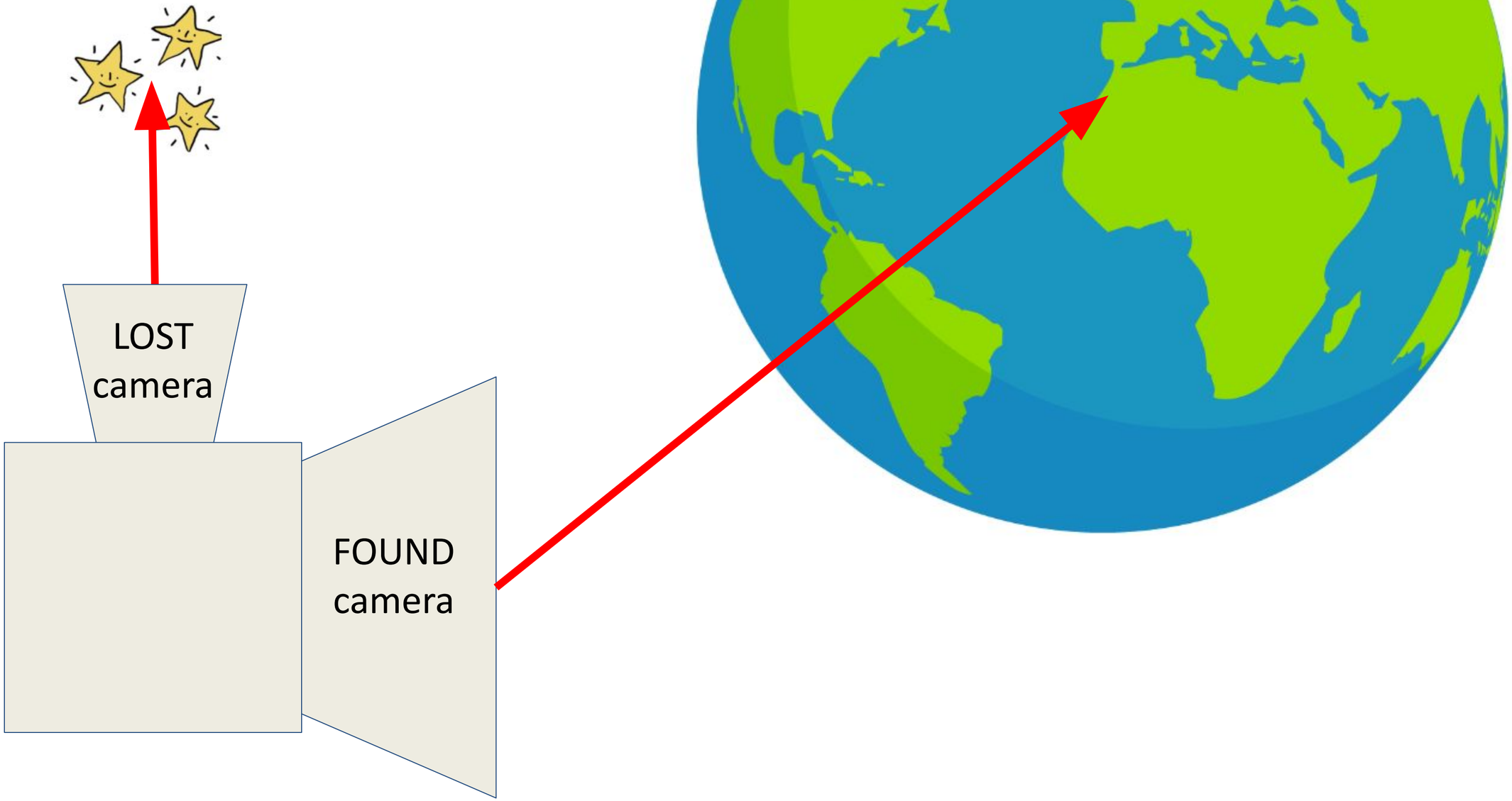




FOUND, distance algorithm in one picture

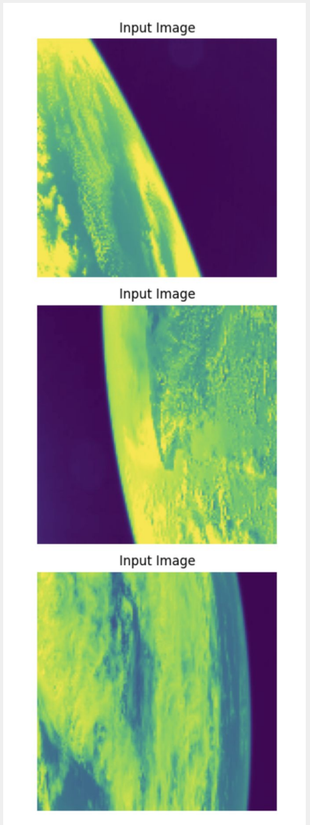


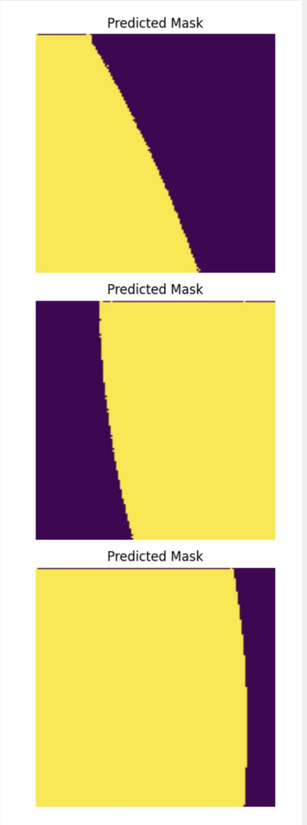
How LOST attitude informs FOUND position



FOUND pipeline, overview of our algorithm

Repeat N times





Attitude (LOST)

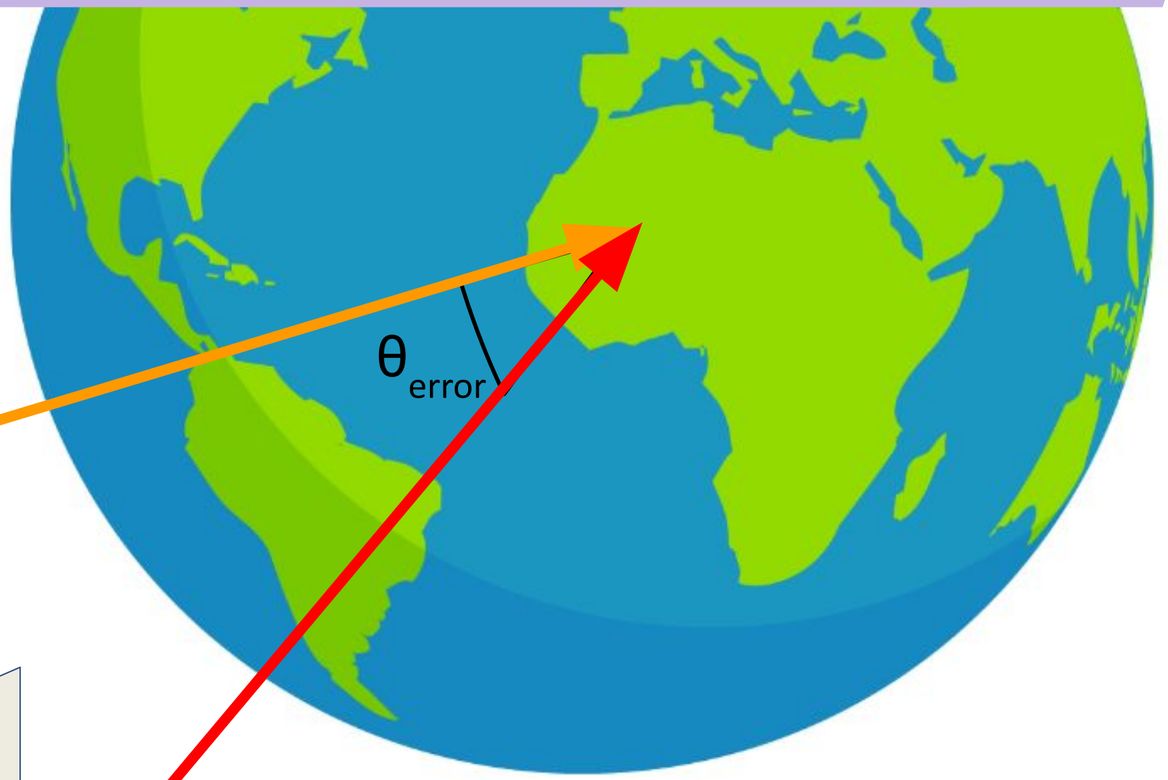
Correct Distortion

Distance Determination

Position ECI

Extended Kalman Filter

How attitude error contributes to FOUND error



Estimated Position

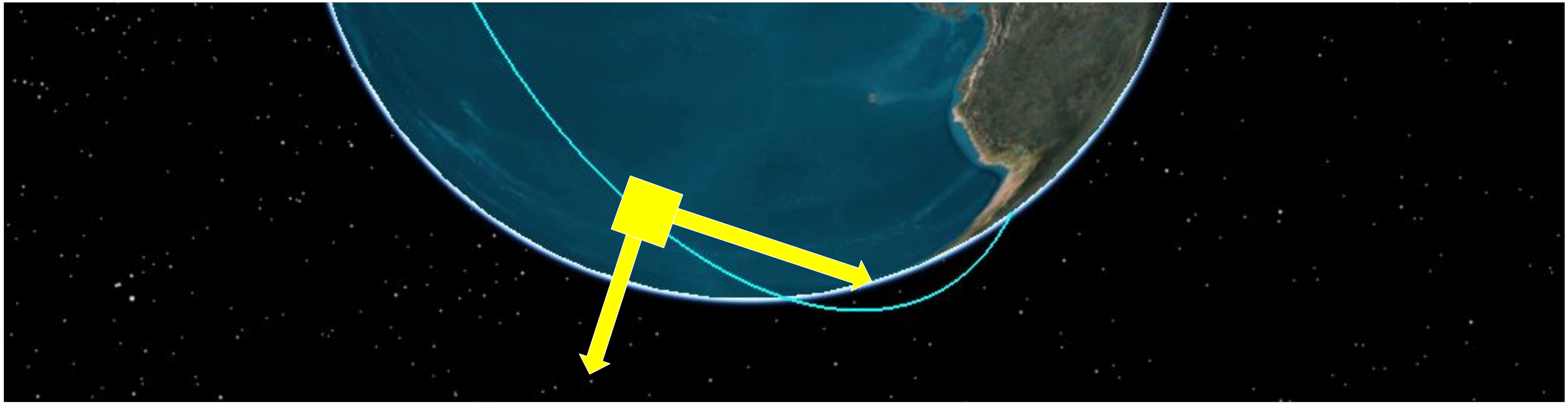
FOUND camera

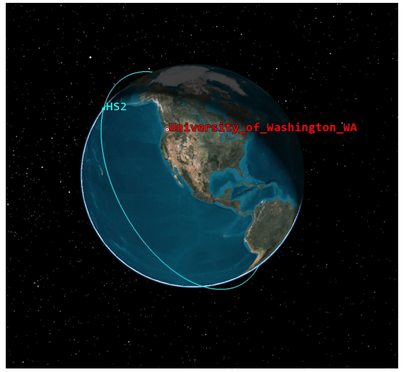
True Position

FOUND camera

$$\hat{p} = r\theta$$

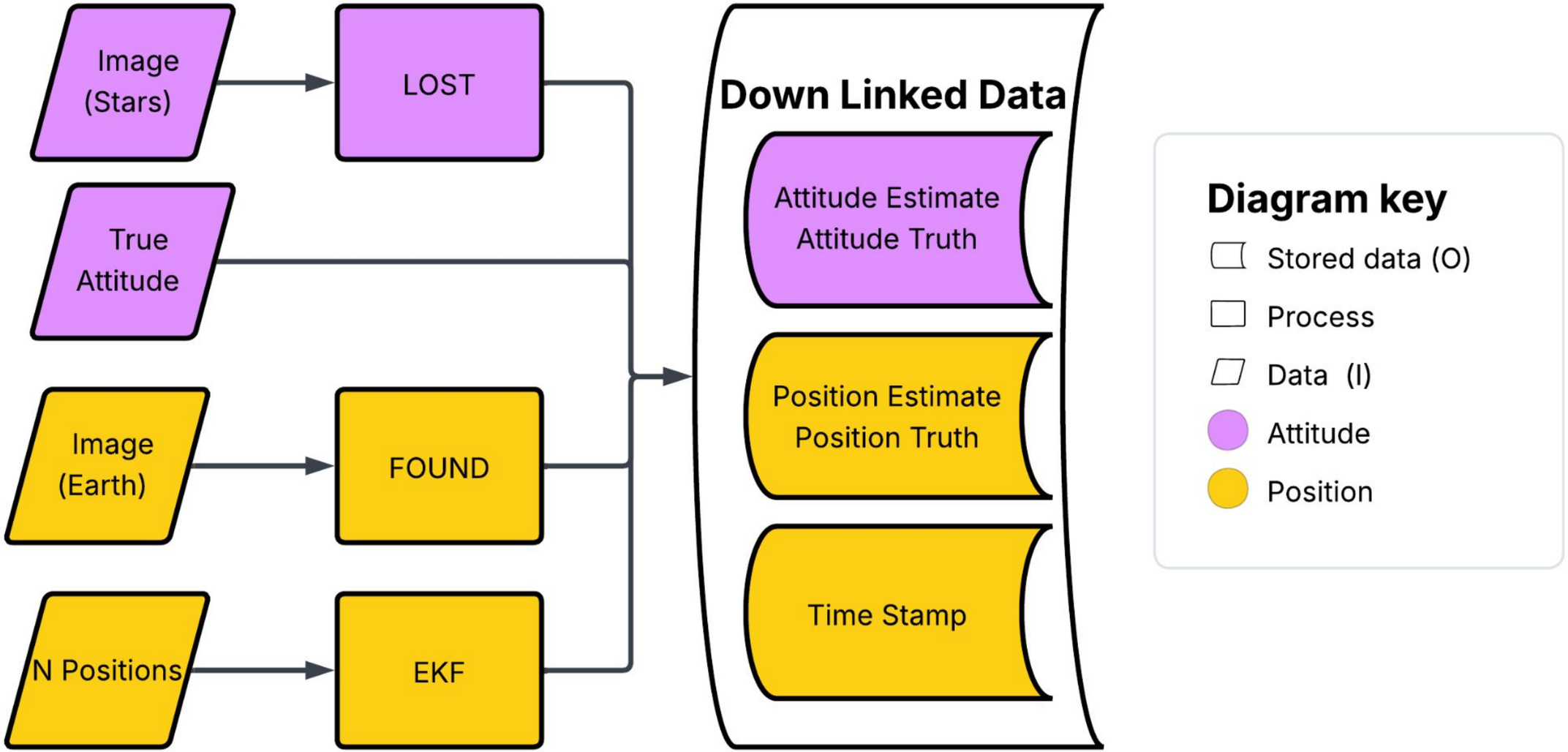
HS2 payload operates in ISS orbit



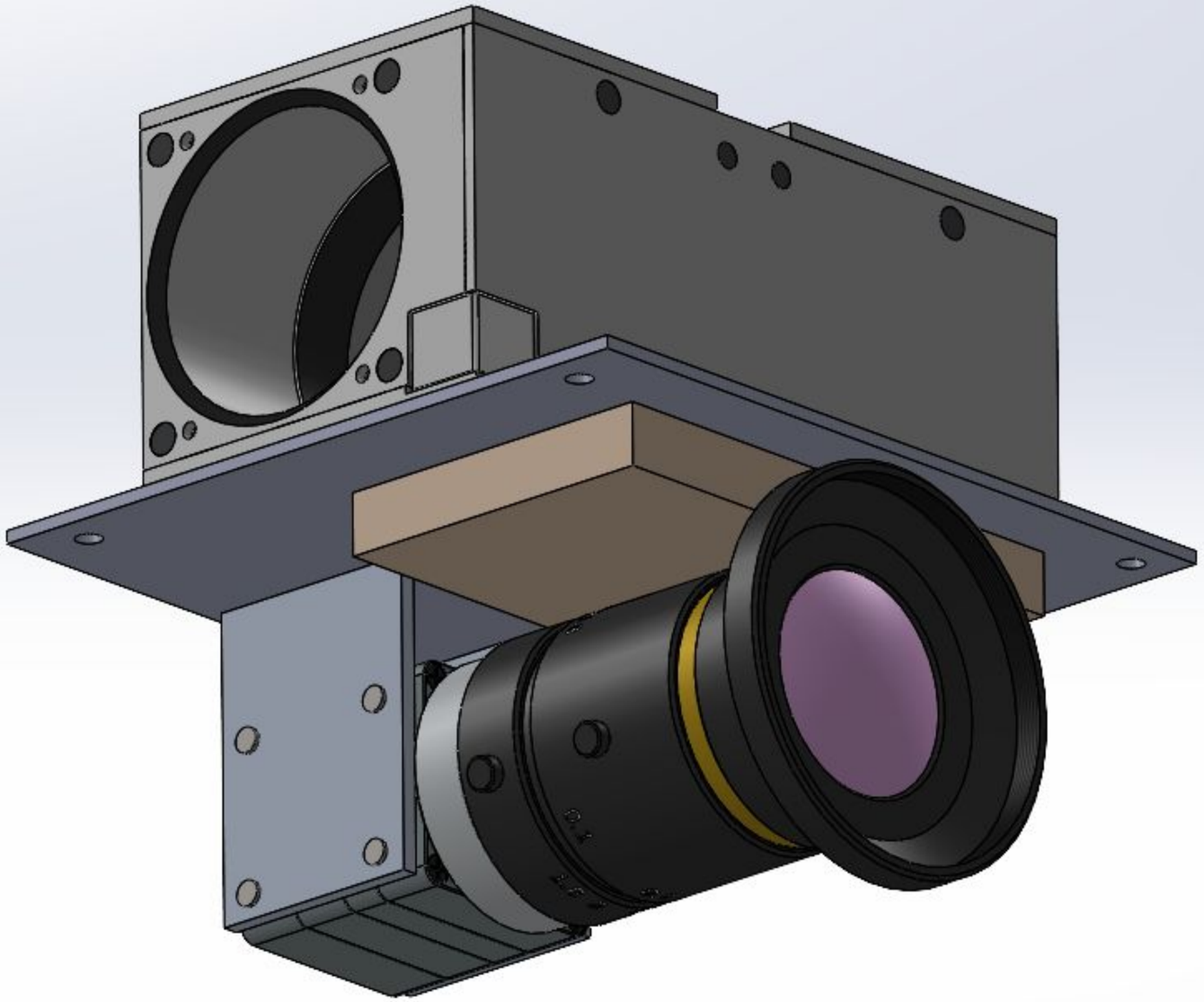


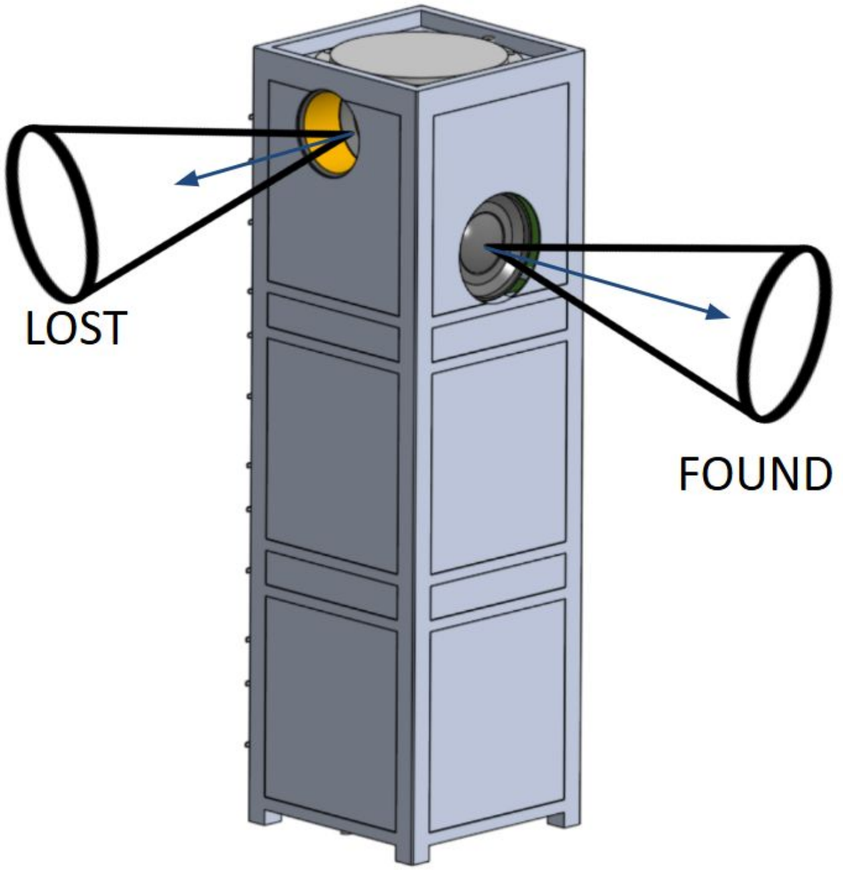
- FOUND points at limb of Earth
- LOST points at stars
- Neither point at sun

Science data pipeline (Space), 64 bytes per experiment



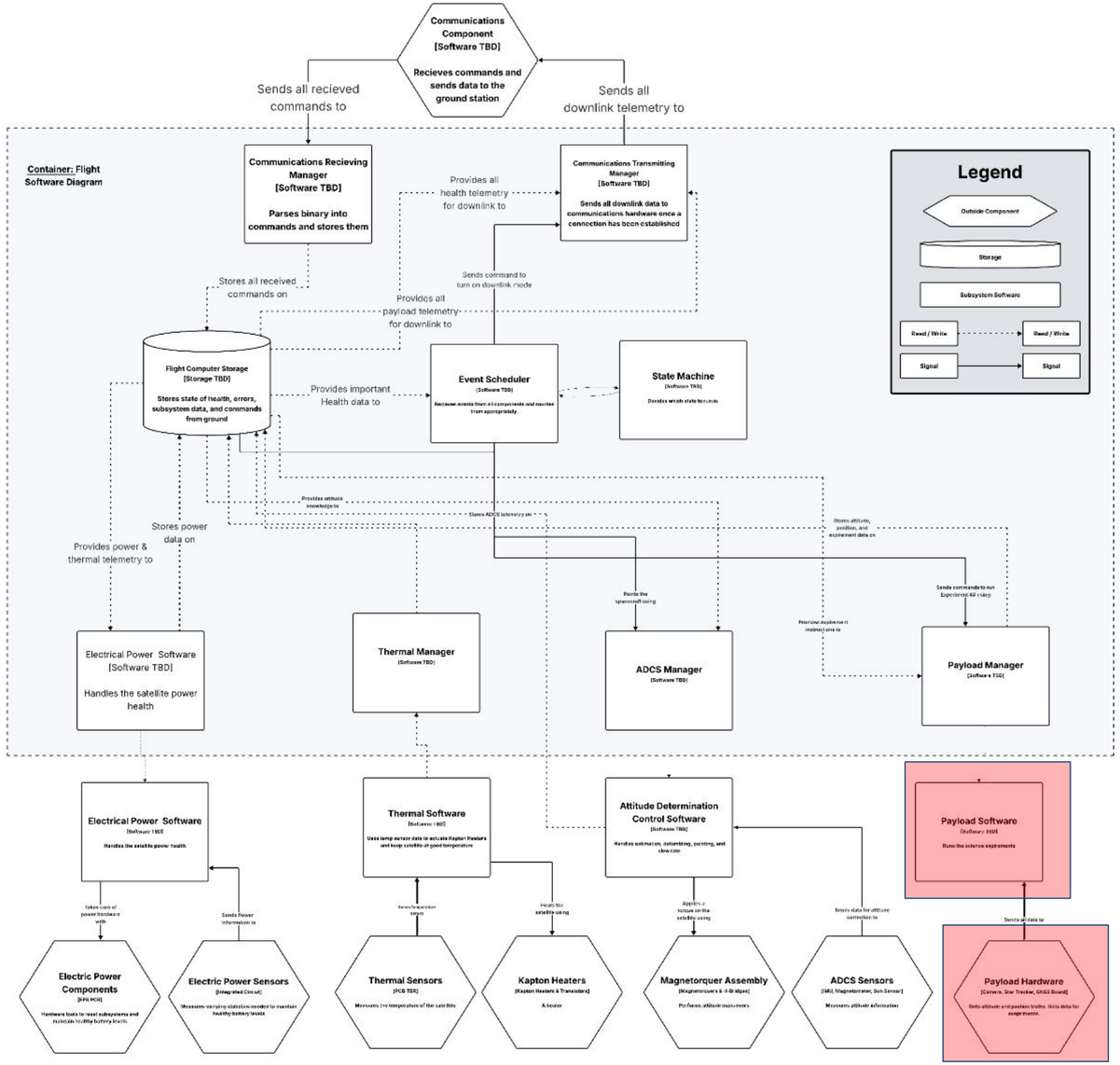
Payload hardware assembly





Visual of External CubeSat Layout

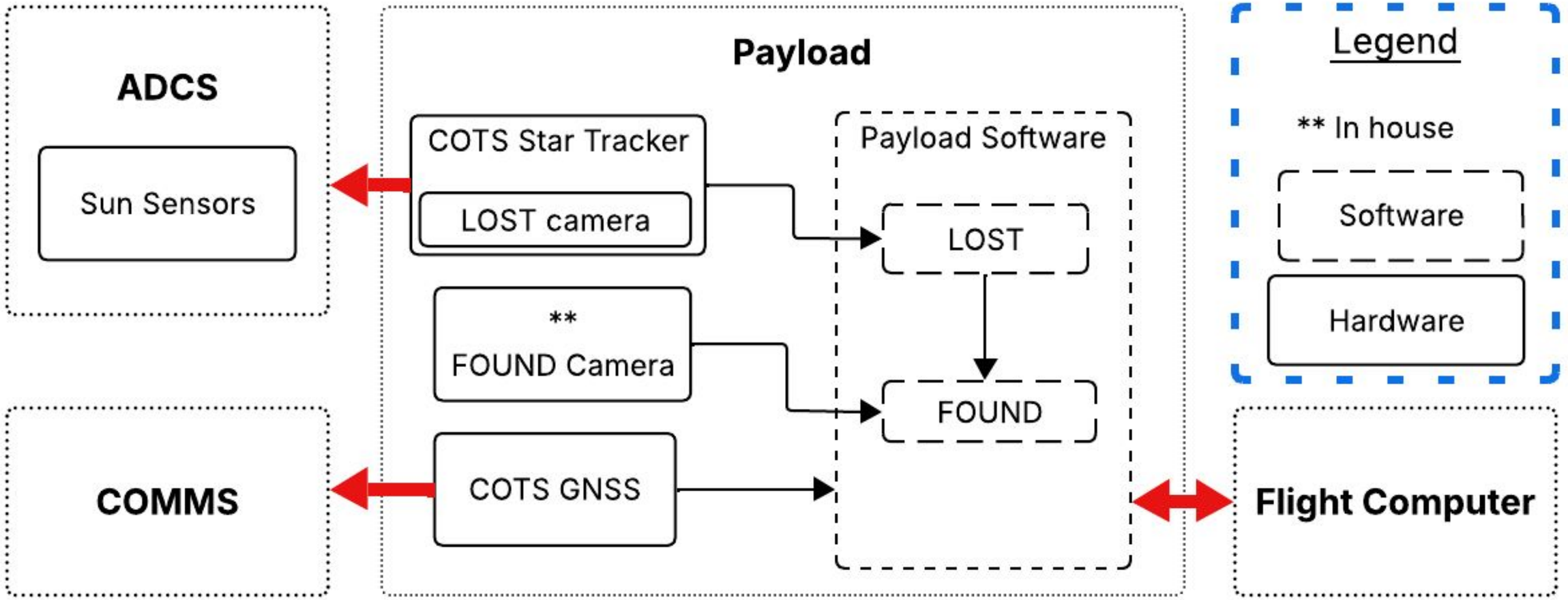
Software Requirements



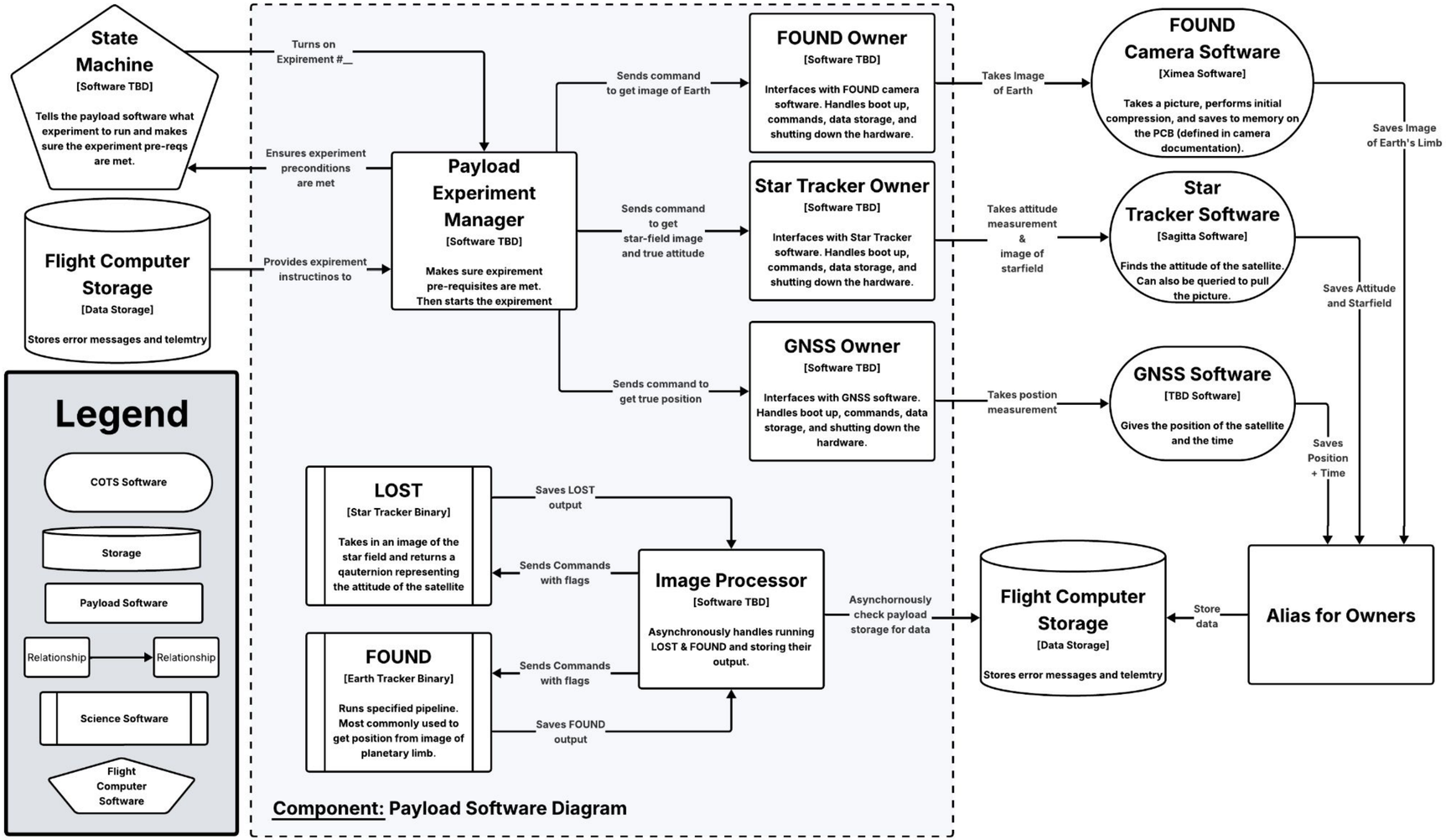
PAY-18 (Avionics)

The payload software shall **carry out experiments** as specified in the Mission Design Document

Important Payload Interfaces

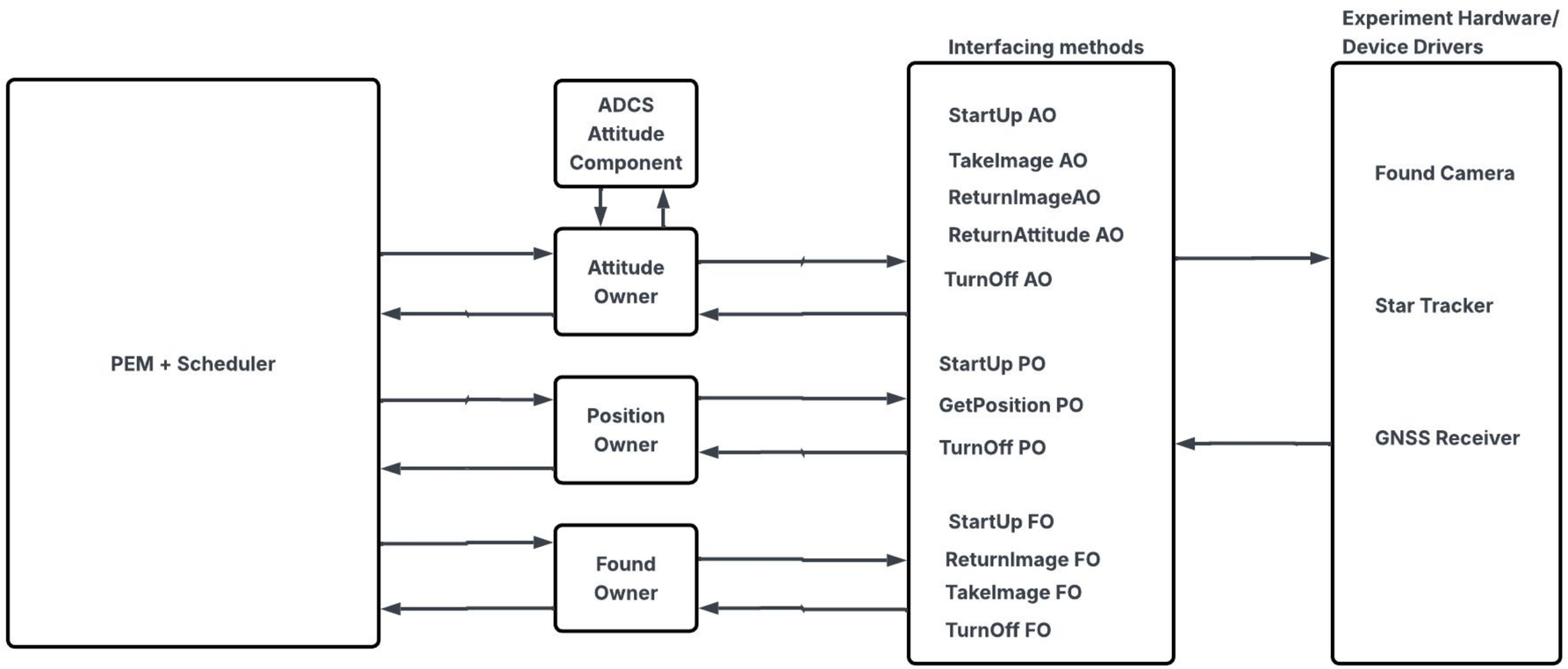


Payload Software Block Diagram



The goal of payload software is to <u>complete experiments</u> and allow access of **mission critical hardware** to **ADCS** and **Communications**.

Software Interface to Payload Hardware



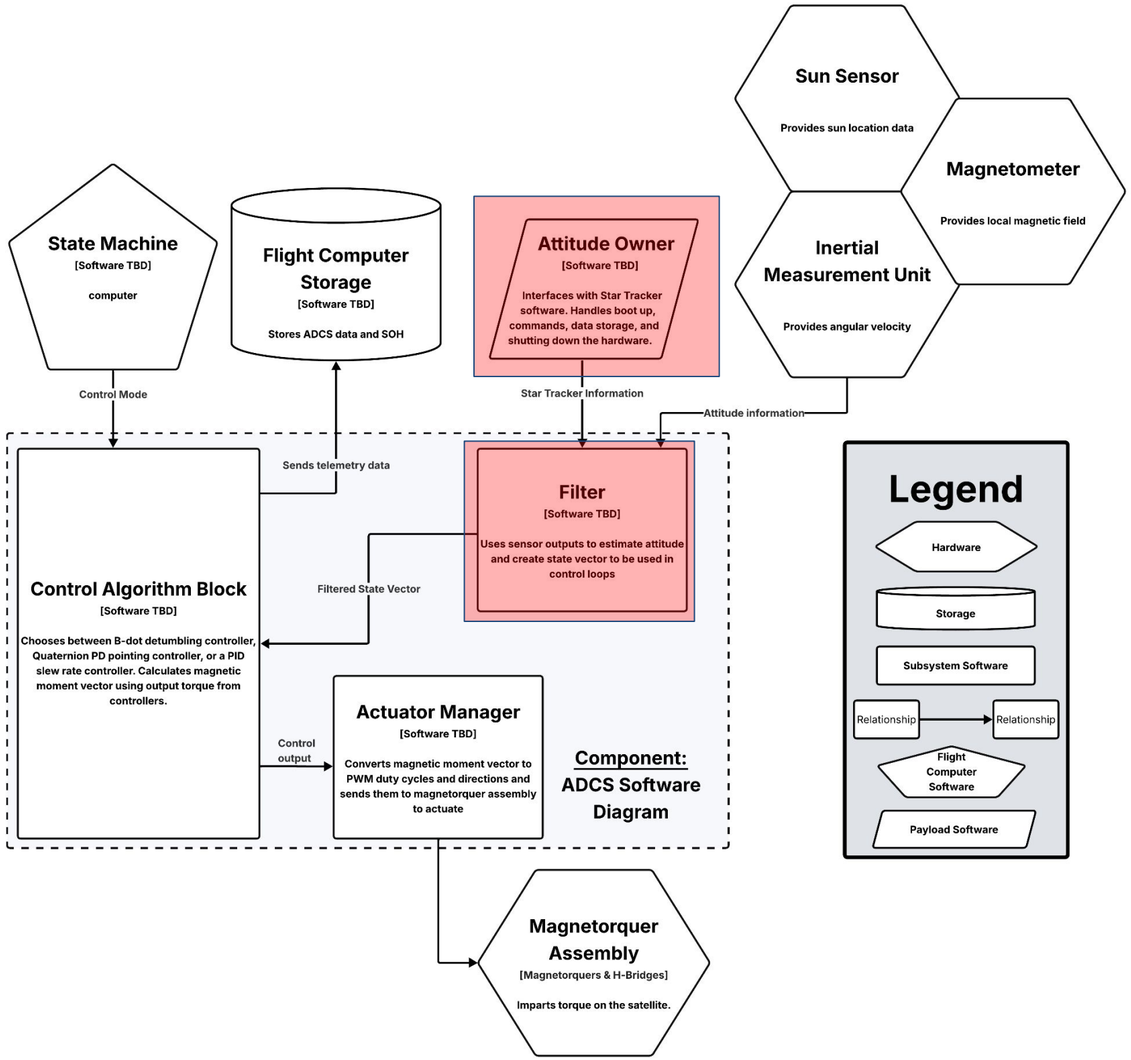
AO -> Attitude Owner

FO -> Found Owner

PO -> Position Owner

PEM -> Payload Experiment Manager

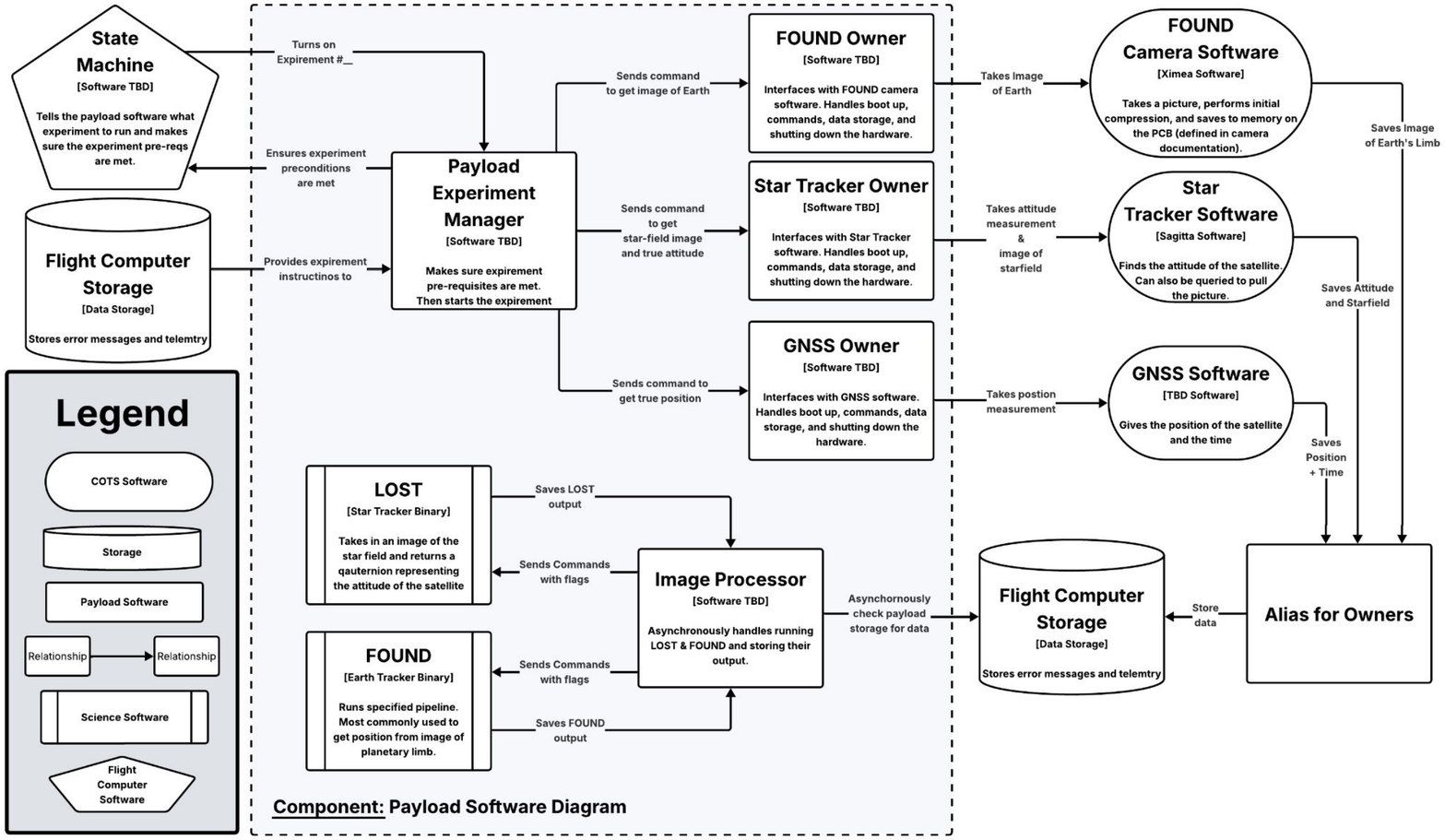
How ADCS accesses attitude in F Prime Architecture



Major Risks

- Attitude Owner does not meet real time constraints of ADCS

Software Next Steps



Continuous Integration

100% Code Coverage

Linting, Memory Tests, Google Stylecheck, Comments

Autonomous Code Checks & Code Reviews

Integration Test & HIL Test

FOUND camera requirements

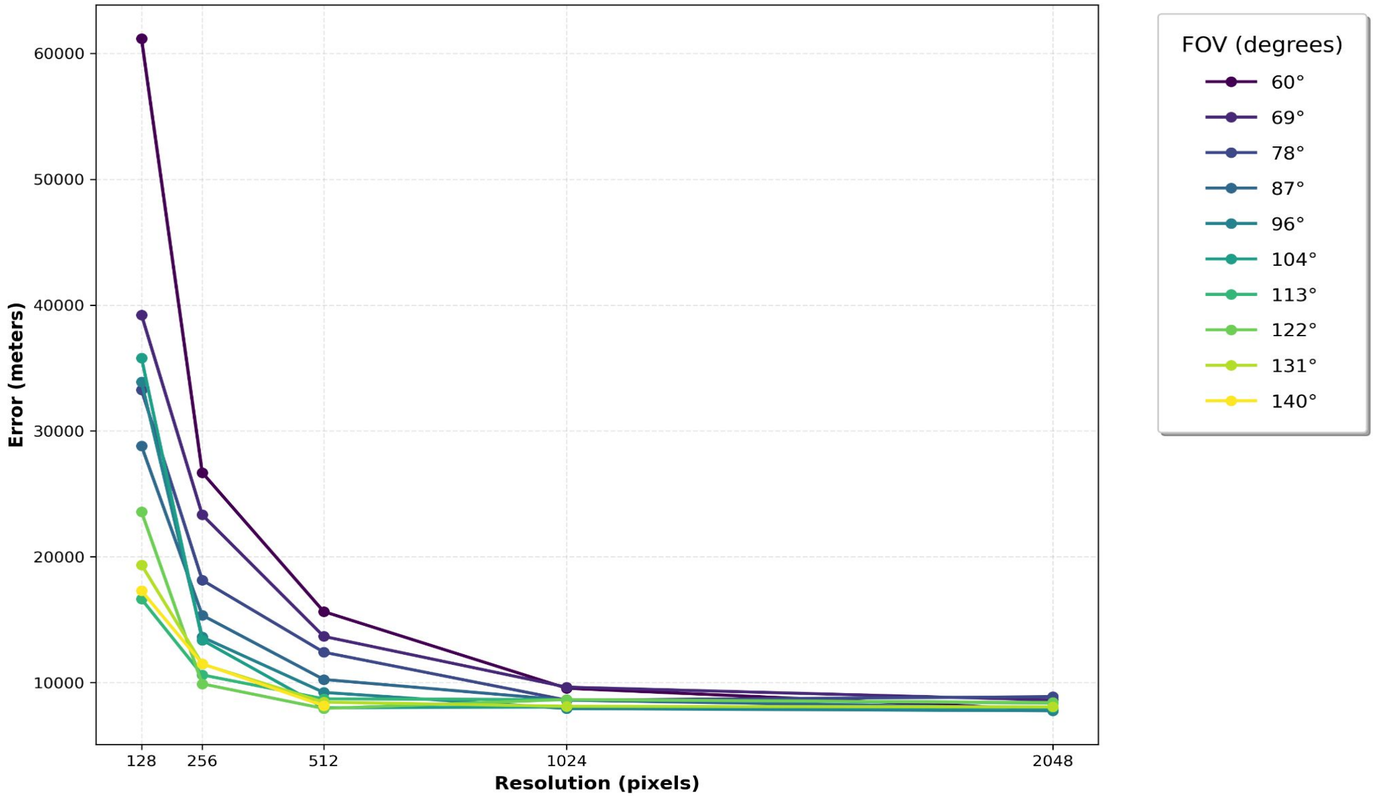
Category	Requirement
PAY-1 (FOUND)	The **FOUND** camera shall have a **1024x1024 pixel** resolution.
PAY-2 (FOUND)	The **FOUND** camera shall have an **FOV** between **78 [TBR]** and **90 [TBR]** degrees.
PAY-3 (FOUND)	The **FOUND** camera shall blur the horizon across **≤ 2 pixels [TBR]**.
PAY-4 (FOUND)	The **FOUND** camera shall image Earth **≥ 20dB [TBR] above** the **noise floor**

The FOUND optical system

THOR LABS MVL5TM23 - 5 mm EFL, f/1.8, for 2/3" C-Mount

XIMEA MQ013RG-E2-S7 - 1.3 Mpix, NIR e2v, Global shutter

FOUND shall have FOV between 78-90°

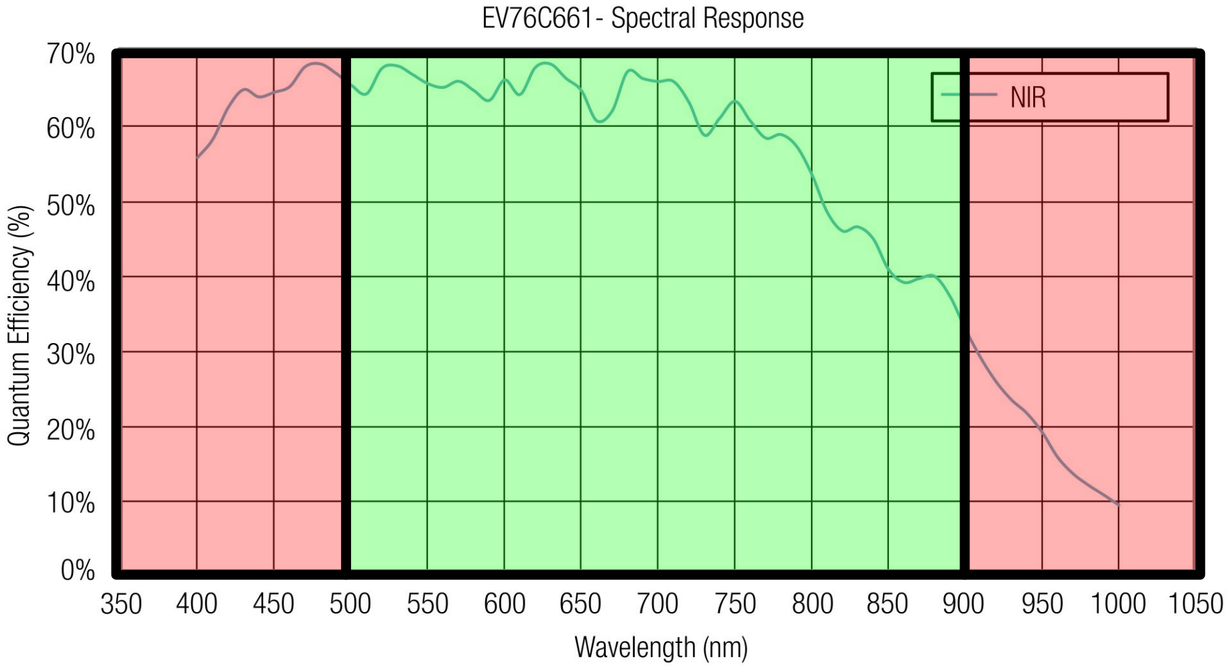


- performance increases with **larger FOVs**
- **uncharacterized** for **non-rectilinear** camera

THOR LABS MVL5TM23 -

5 mm EFL, f/1.8, for 2/3"

FOUND optical system meets requirements



Rayleigh Scattering

Measured Spectrum

Chromatic Aberation

Optical System Specs	
Resolution	1280 x 1024
Pixel size	5.3 µm
Saturation Capacity	7892.5 e-
Read out noise	23.7 e-
Exposure minium	100 µs
Exposure maximum	1s
Aperture	f/1.8
Focal Length	5mm
Optical Throughput	TBD
FOV (1/1.8)	84°

electrons-per-pixel(radiance, optical throughput, quantum efficiency, exposure time, spectrum) = **328**

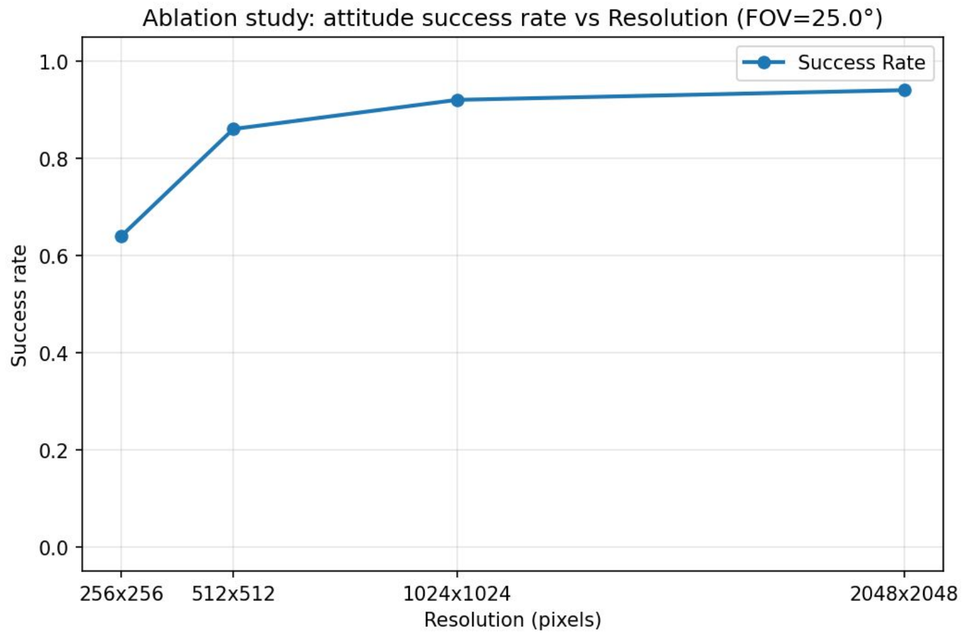
FOUND Camera Risks

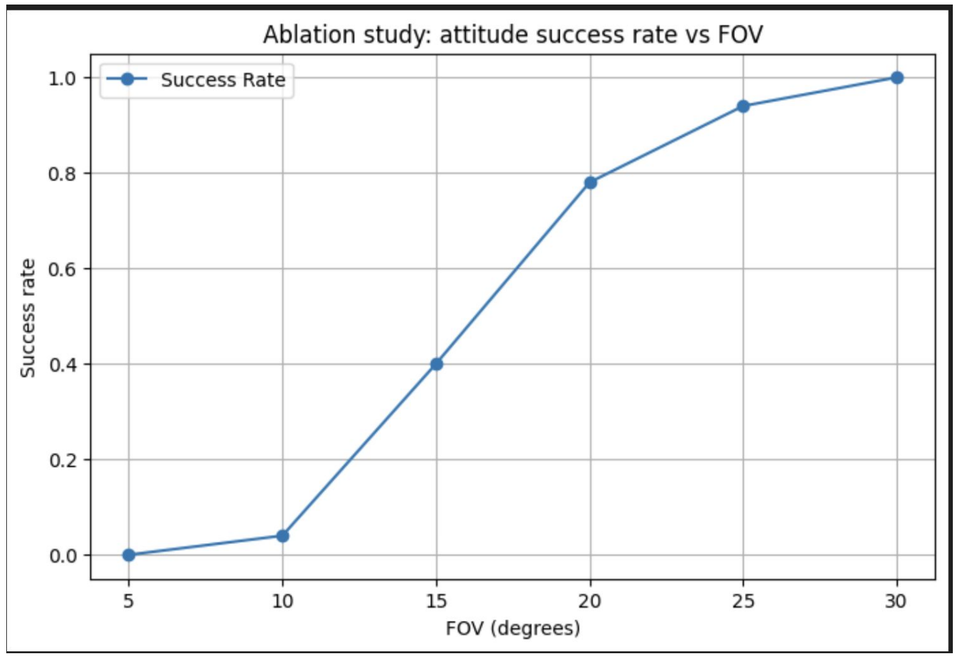
Risk	Consequence	Likelihood	**Blocker** / Strategy
Outgassing release pressure & limit particles	unknown	5	**Lens selection**
orbit and baffle design	5	2	**Time** / recruiting member
chromatic aberration, optical low pass filters, radiation shielding, defocusing, & other **high order camera effects**	2	5	**Time**
distortion calibration	3	2	**Monetary** / Reach out to labs around UW

LOST camera & Attitude truth requirements

Category	Requirement
PAY-5 (LOST)	The **LOST** camera shall have a **1024x1024 pixel** resolution.
PAY-6 (LOST)	The **LOST** camera shall have an **FOV** between **20 [TBR]** and **25 [TBR]** degrees.
PAY-7 (Attitude)	The payload shall determine the **true attitude** to **4 [TBR]** arcseconds cross-boresight
PAY-8 (Attitude)	The payload shall determine the **true attitude** to **26 [TBR]** arcseconds around-boresight

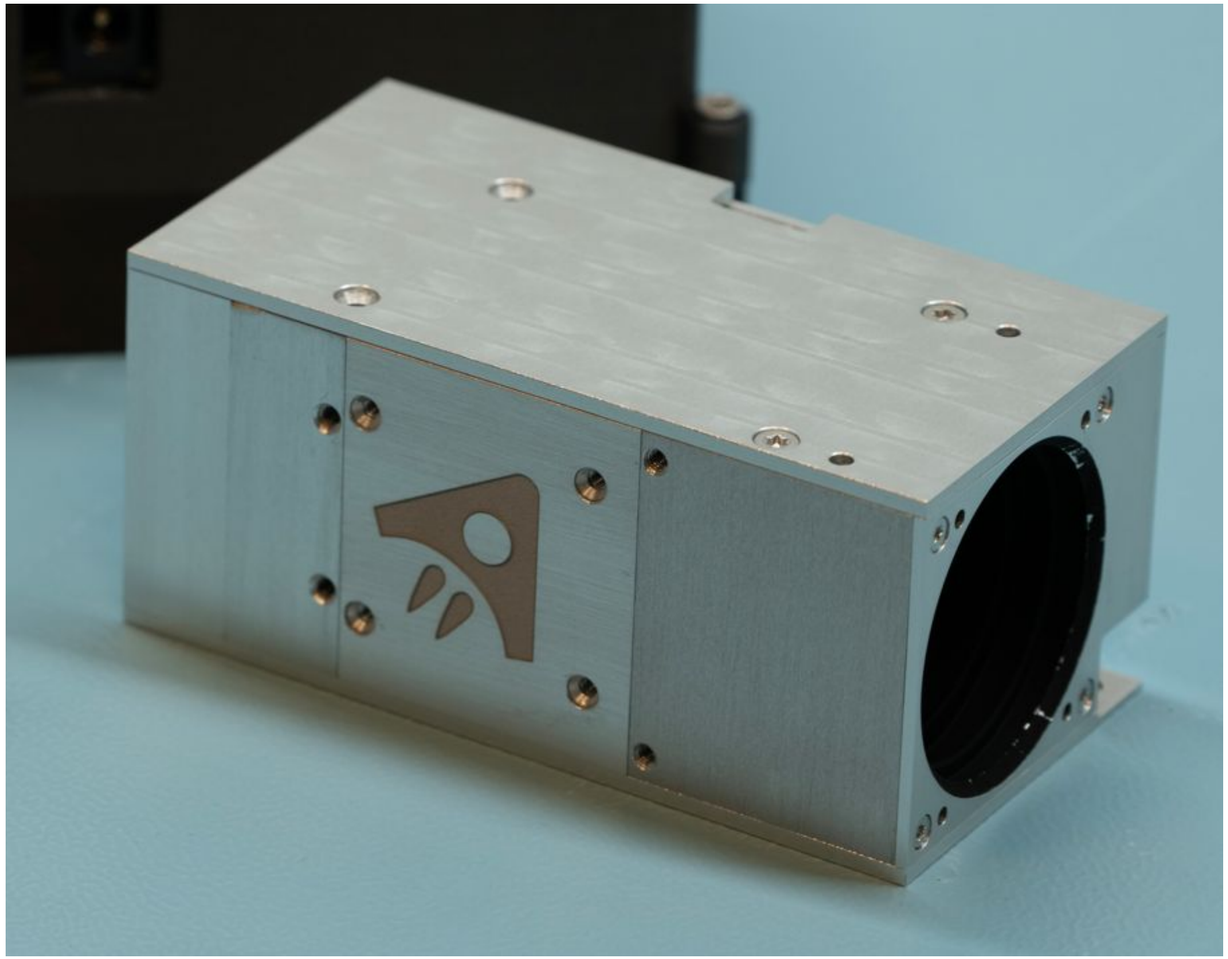
LOST Camera & Attitude Truth Requirements





Cross-boresight accuracy: **4.37 arcseconds**
Around-boresight (roll) accuracy: **26.18 arcseconds**
(1024 x 1024 resolution, 25° FOV, averaged across 100 random trials)

Sagitta Star Tracker from ArcSec



Performance

- Cross-boresight accuracy: 2” (1σ)
- Around boresight accuracy: 10” (1σ)

Power Consumption

- Nominal: 1.5 W

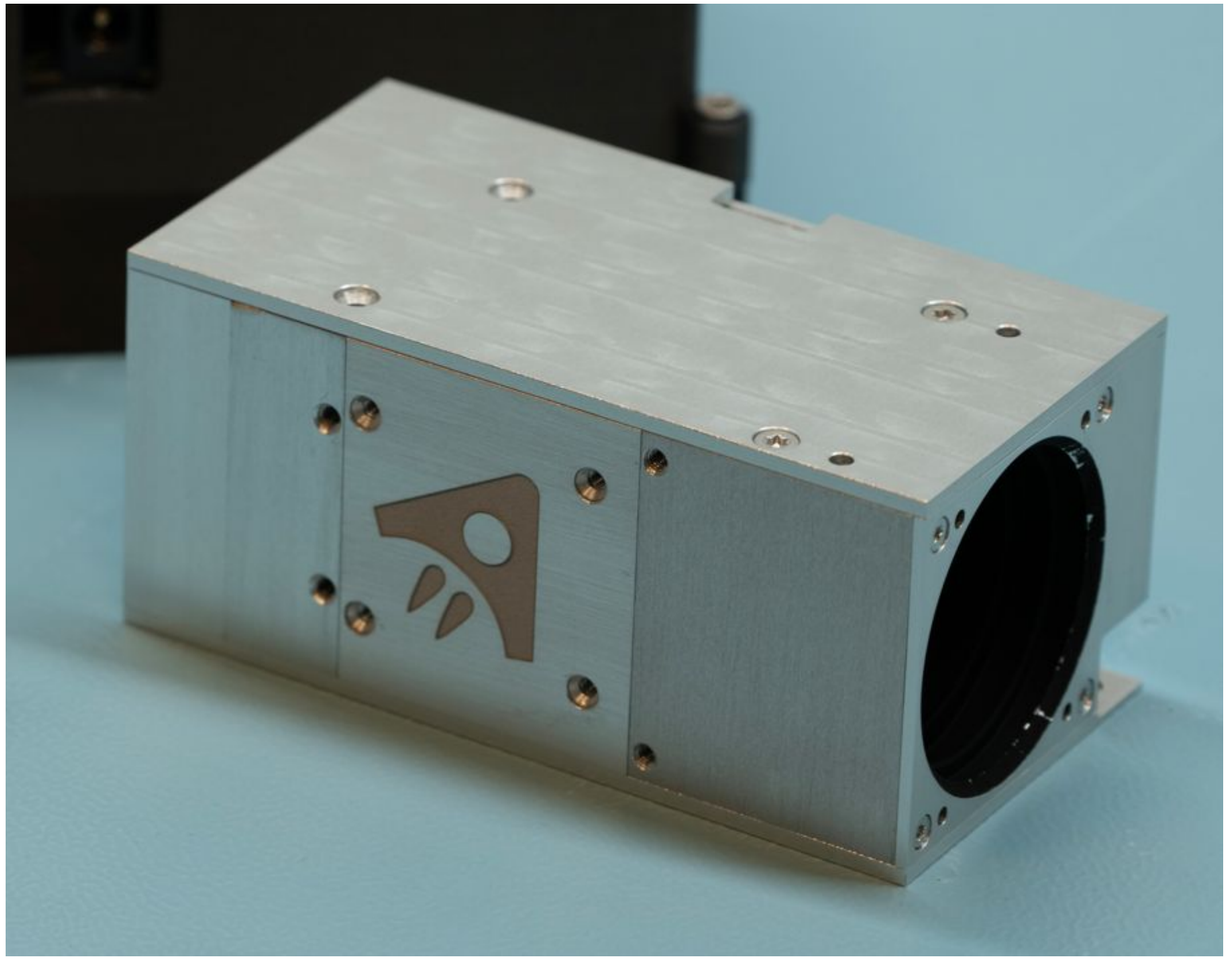
Interface

- 4.5-5.5 V
- Mass 270g

Camera Specs

- meets requirements

LOST Camera / Truth Measure Next Steps

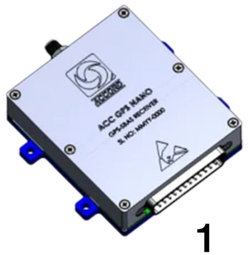


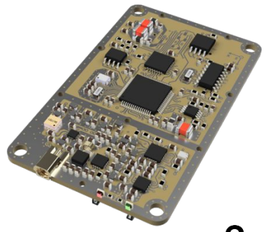
1. Clean Room
2. Validation & Testing
3. Procedures

We are terrified of breaking the ~$40,000 engineering model!!!

Position Truth Requirements

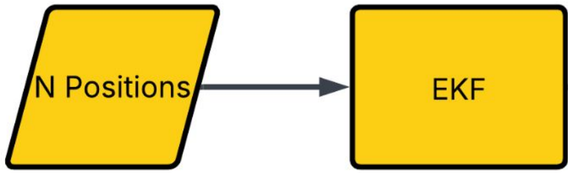
Category	Requirement
PAY-9 (Position)	The **ECI coordinates** shall be known to within **500m** accuracy at **time of FOUND image** .


1


2


3

1) Accord GPS SBAS Receiver ACC-GPS-NANO
2) Spacemanic Celeste
3) SkyFox Labs piNAV-NG



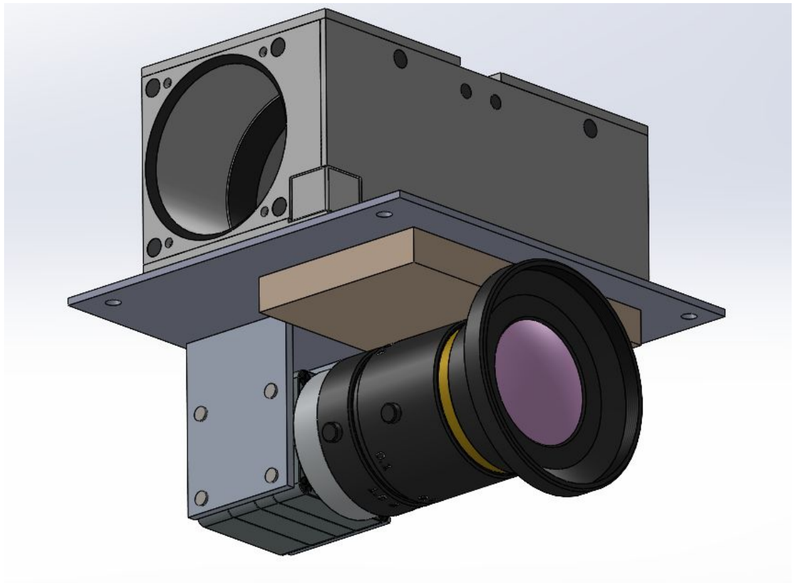
Next Steps

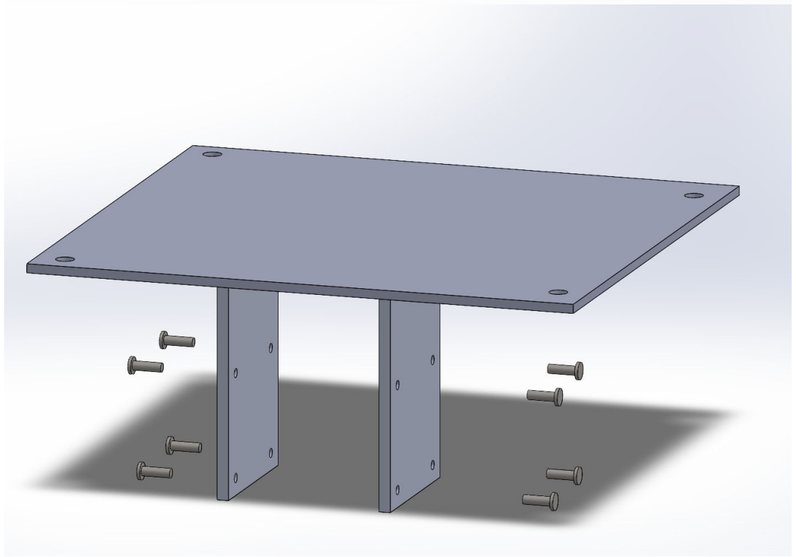
- Select Receiver
- Select Antenna
- Antenna Placement
- Ground Test against RTK

Payload Structures Requirements

Category	Requirement
PAY-10 (Structure)	The **LOST** and **FOUND** camera shall be **oriented** at an angle of **90 ±[TBD]** degrees
PAY-11 (Structure)	The **angle between** the **LOST** and **FOUND** camera shall be **known** to within **± .10** degrees
PAY-12 (Structure)	The payload shall have a mass of **1100 g [TBR]**
PAY-13 (Structure)	The payload shall have a volume of **10cm x 10cm x 11cm [TBR]**.

Payload Structures Assembly

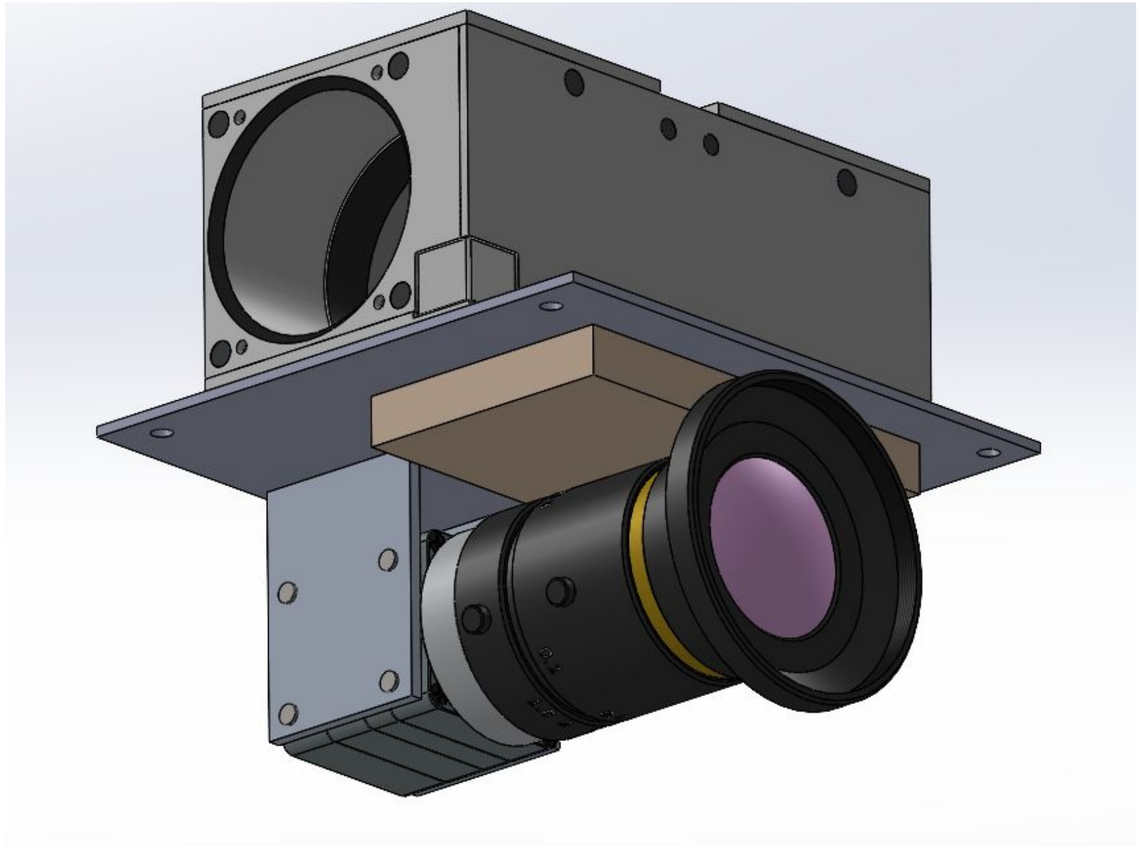




Next Steps

- Thermals
- Deformation
- Structural & Electronic Interfaces
- Machine
- Real Life Testing

Payload Structures Requirements Progress



Requirement

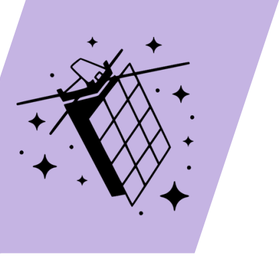
The **LOST** and **FOUND** camera shall be **oriented** at an angle of **90 ±[TBD]** degrees

The **angle between** the **LOST** and **FOUND** camera shall be **known** to within **± .10** degrees

The payload shall have a mass of **1100 g [TBR]**

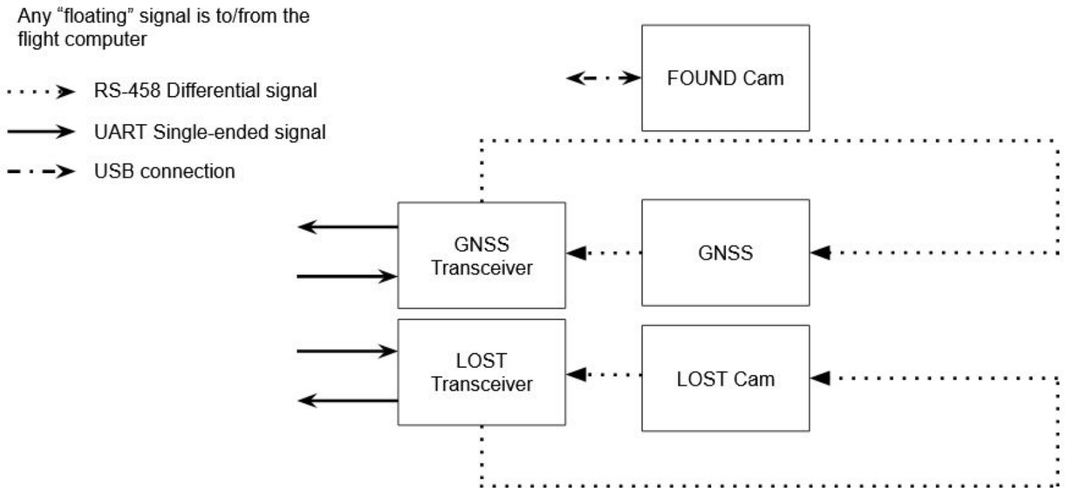
The payload shall have a volume of **10cm x 10cm x 11cm [TBR]**.

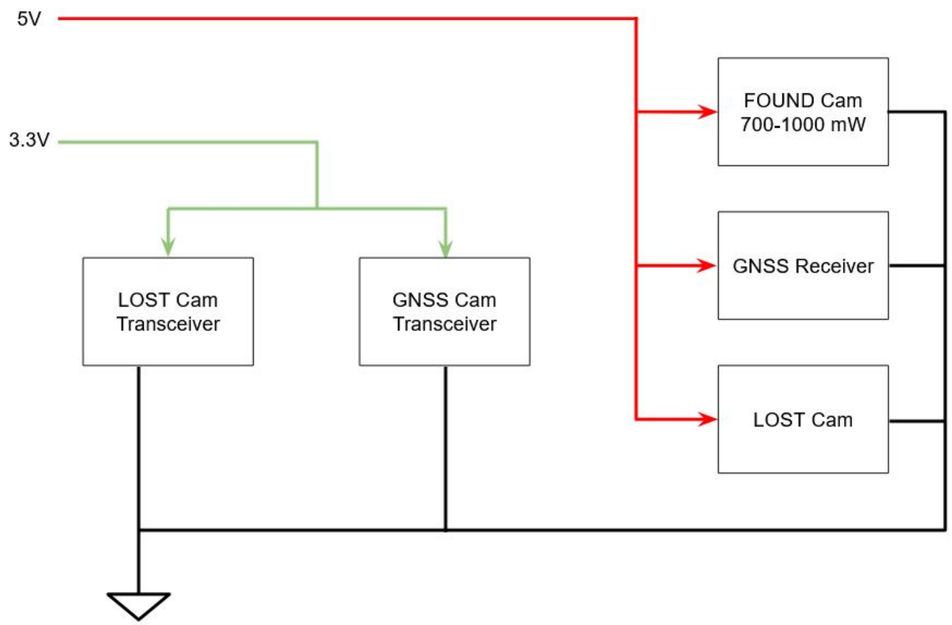
Avionics Requirements



Category	Requirement
PAY-14 (Avionics)	The **FOUND camera** and **GNSS receiver** measurements shall be less than **±65 milliseconds** apart
PAY-15 (Avionics)	The **FOUND camera** and **Star Tracker** measurements shall be less than **±10 milliseconds** apart
PAY-16 (Avionics)	The payload shall use under **3.2 W [TBR]** during Experiment Mode.

Interface





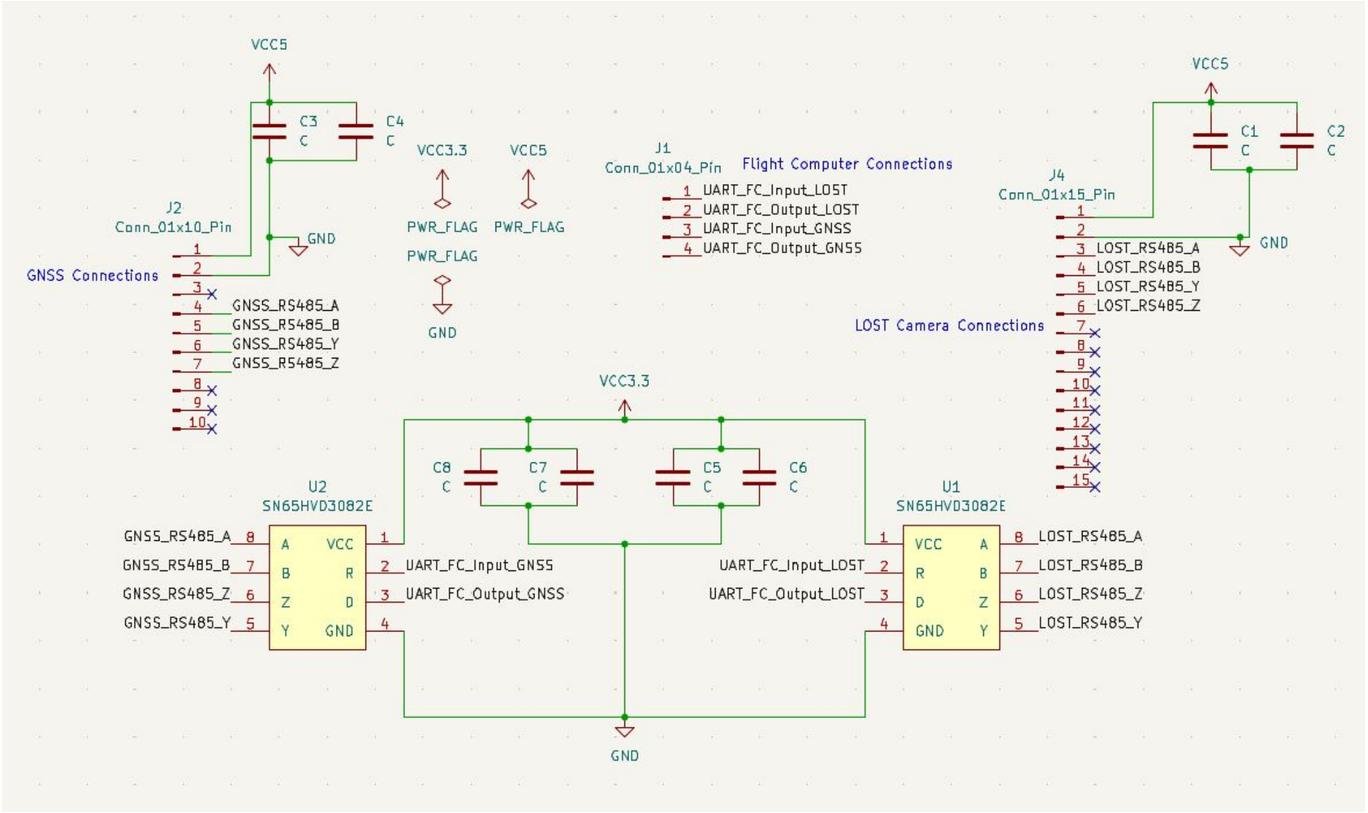
Schematic

- LOST camera (Sagitta) and GNSS Receiver (TBD)
  - RS-485 communication
- Full-Duplex Transceivers ⟶ Flight computer
  - RS-485 to UART “translation”

Interface Next Steps

Considerations
Schematic
PCB Layout
Timing

Key
Started
Not Started



System Level Risks

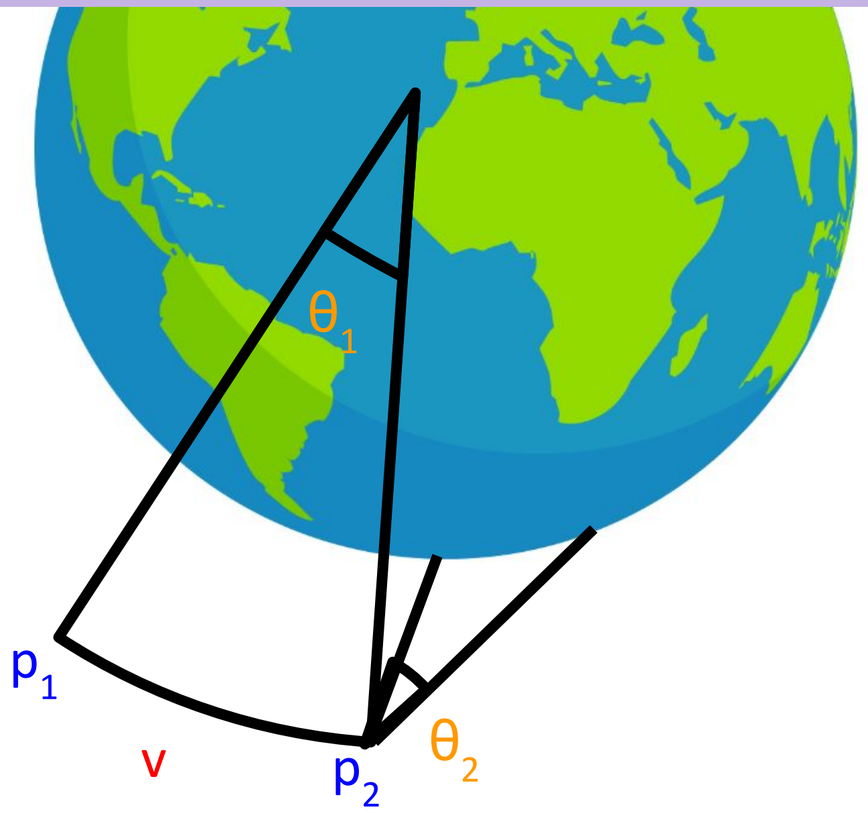
Risk	Consequence	Likelihood	**Blocker** / Strategy
Outgassing	unknown	5	**Lens selection**
The angle between the LOST and FOUND camera shall be known to within **± 0.10 degrees**	5	2	Vibration & Thermal Simulations
Machine Structures	4	2	**Personel**
Single Computer	3	2	**implement software** / F-Prime
orbit, conops, orientation, and baffle design	1	3	**miscommunication issue**

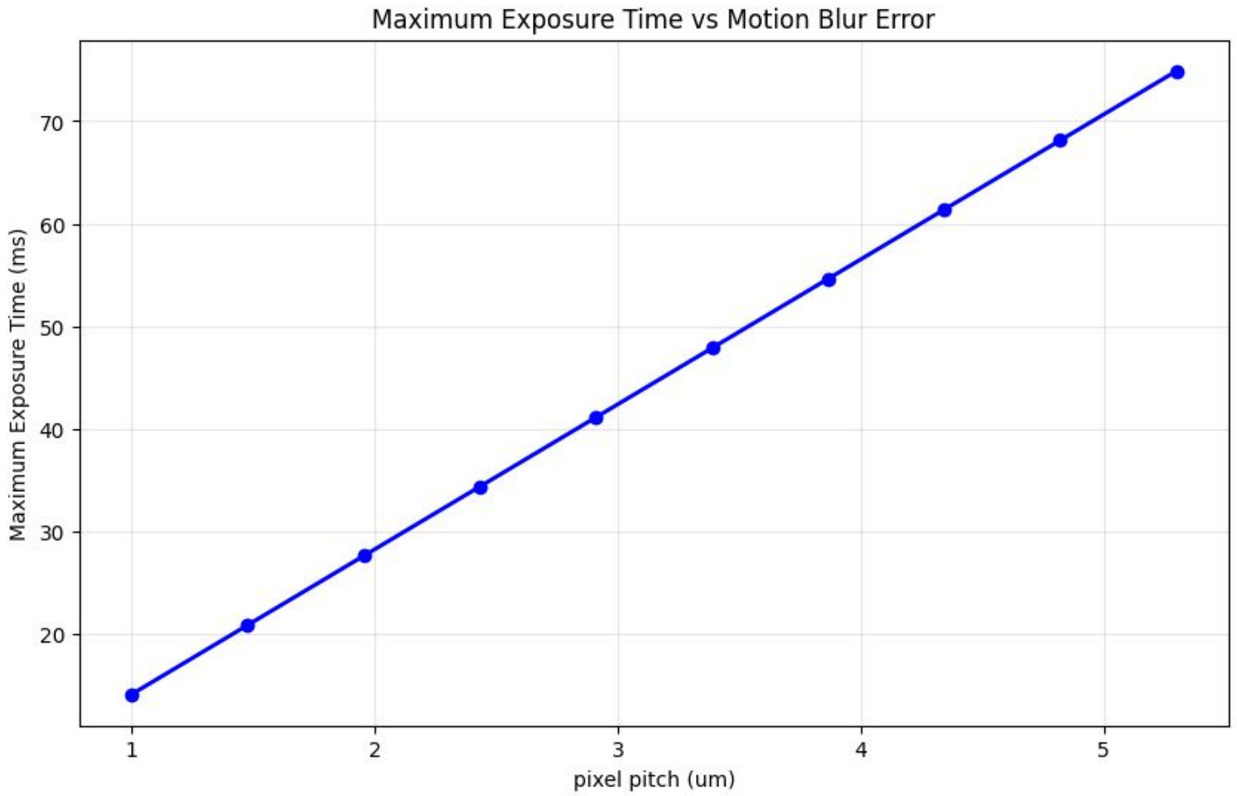
Backup Slides

Position Knowledge Error Budget back-up-slide

Error source	Estimate [meters]	Estimate [arcseconds]	Estimate [miliseconds]	Allocated [meters]	Allocated [arcseconds]	Allocated [milliseconds]
FOUND and attitude measure alignment knowledge	[TBD]	[TBD]		12004.37	360	
Attitude knowledge	333.45	10		1500.55	45	
Timing with attitude measure	[TBD]	[TBD]	[TBD]	600.22	18	10
Position knowledge	10	0.3		100	3	
Position truth measure timing	[TBD]	[TBD]	[TBD]	500	15	65.68
Error Margin				294.86	8.8	
Total	—	—		**15000**	—	—

FOUND camera shall blur the horizon ≤ 2 pixels back-up-slide

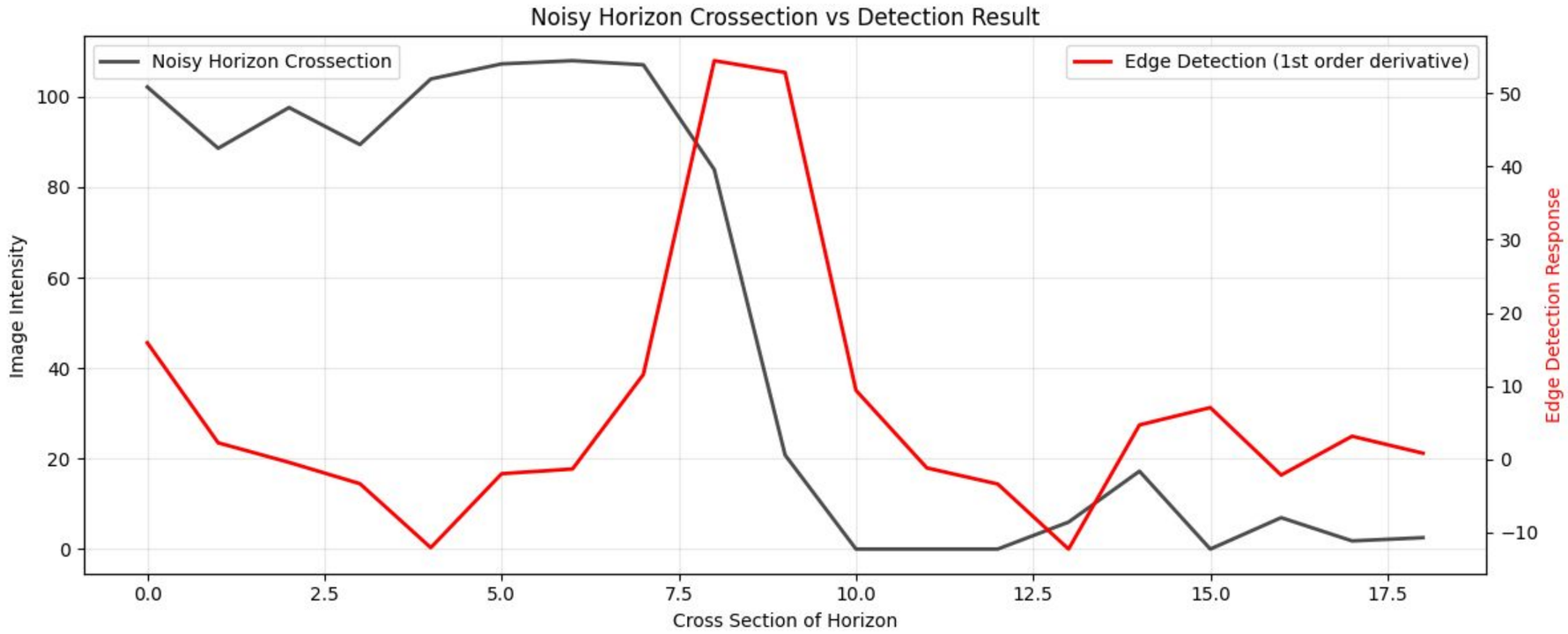




$$\frac{\text{pixel pitch} * \text{altitude}}{\text{focal length}} \approx (\text{altitude} + \text{R}_{\text{Earth}}) * \tan\left(\frac{v_{\text{translational}}}{\text{altitude} + \text{R}_{\text{Earth}}}(t)\right) + \text{altitude} * \tan(\theta_{\text{slew}}(t))$$

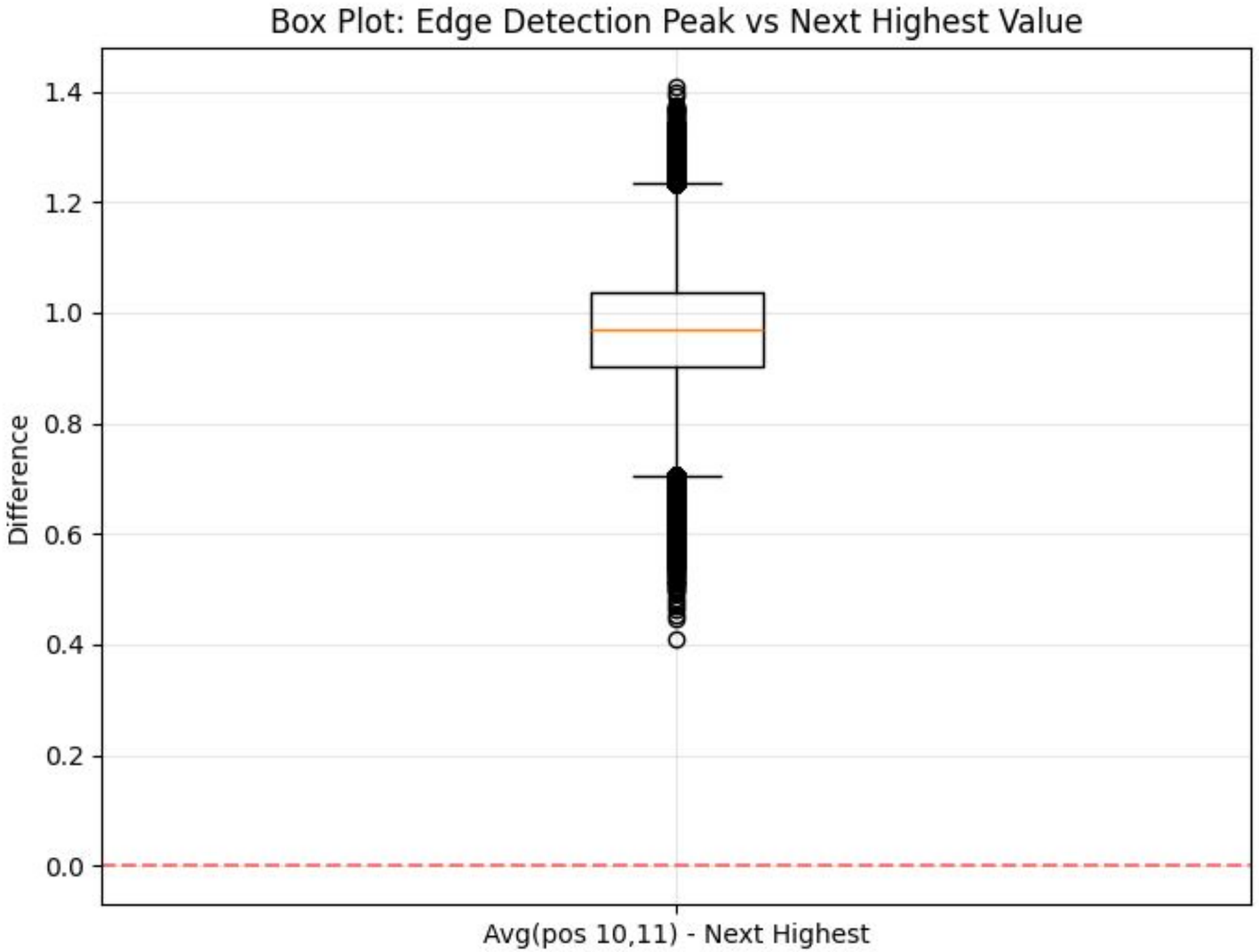
θ_1 θ_2

Earth shall ≥ 20dB above the noise floor back-up-slide



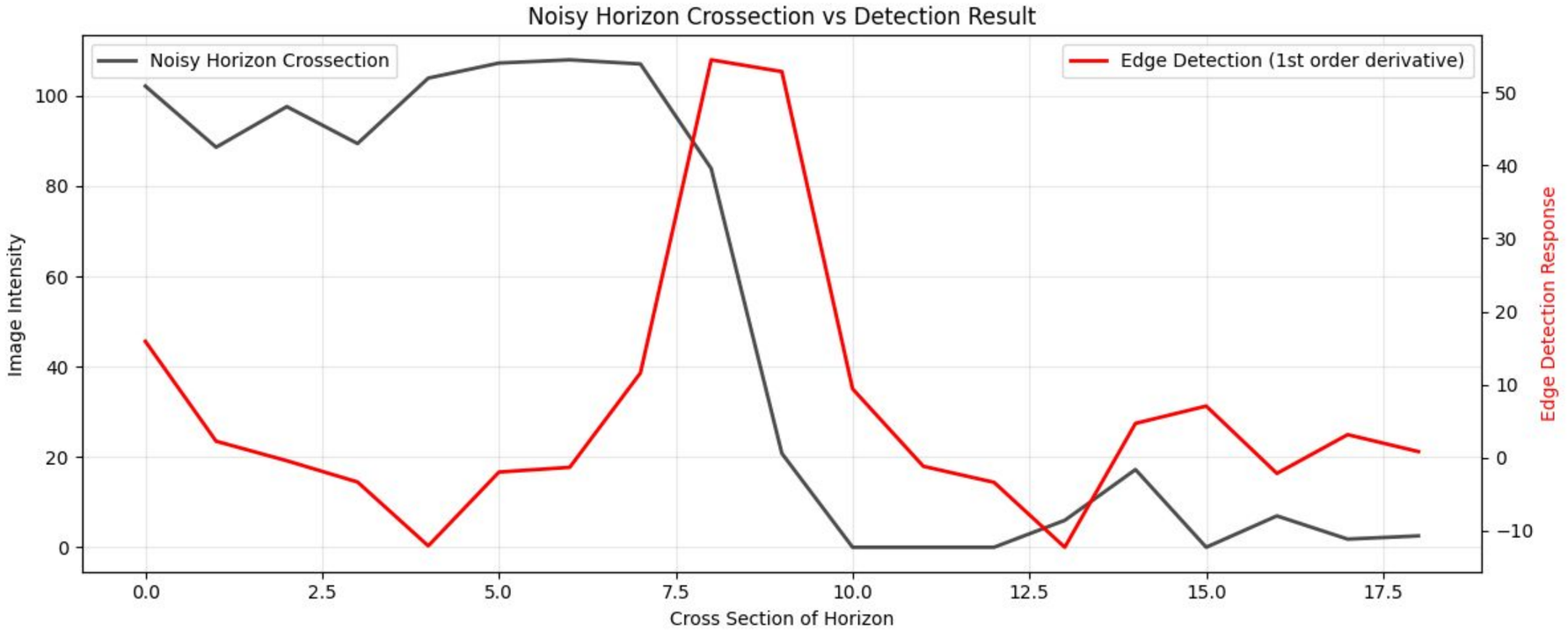
$$\text{dynamic range} = 20 \log\left(\frac{\text{signal}}{\text{noise}}\right)$$

Read Noise’s effect on Image Gradient back-up-slide




```
Mean difference: 0.9680
Std deviation: 0.0987
percent below fifty: 0.00%
```

Gradient backup-slide



$$\text{dynamic range} = 20\log\left(\frac{\text{signal}}{\text{noise}}\right)$$

Four point central difference

$$g'_r(r,c) = \frac{8[I(r+1,c) - I(r-1,c)] - [I(r+2,c) - I(r-2,c)]}{12\,h}$$

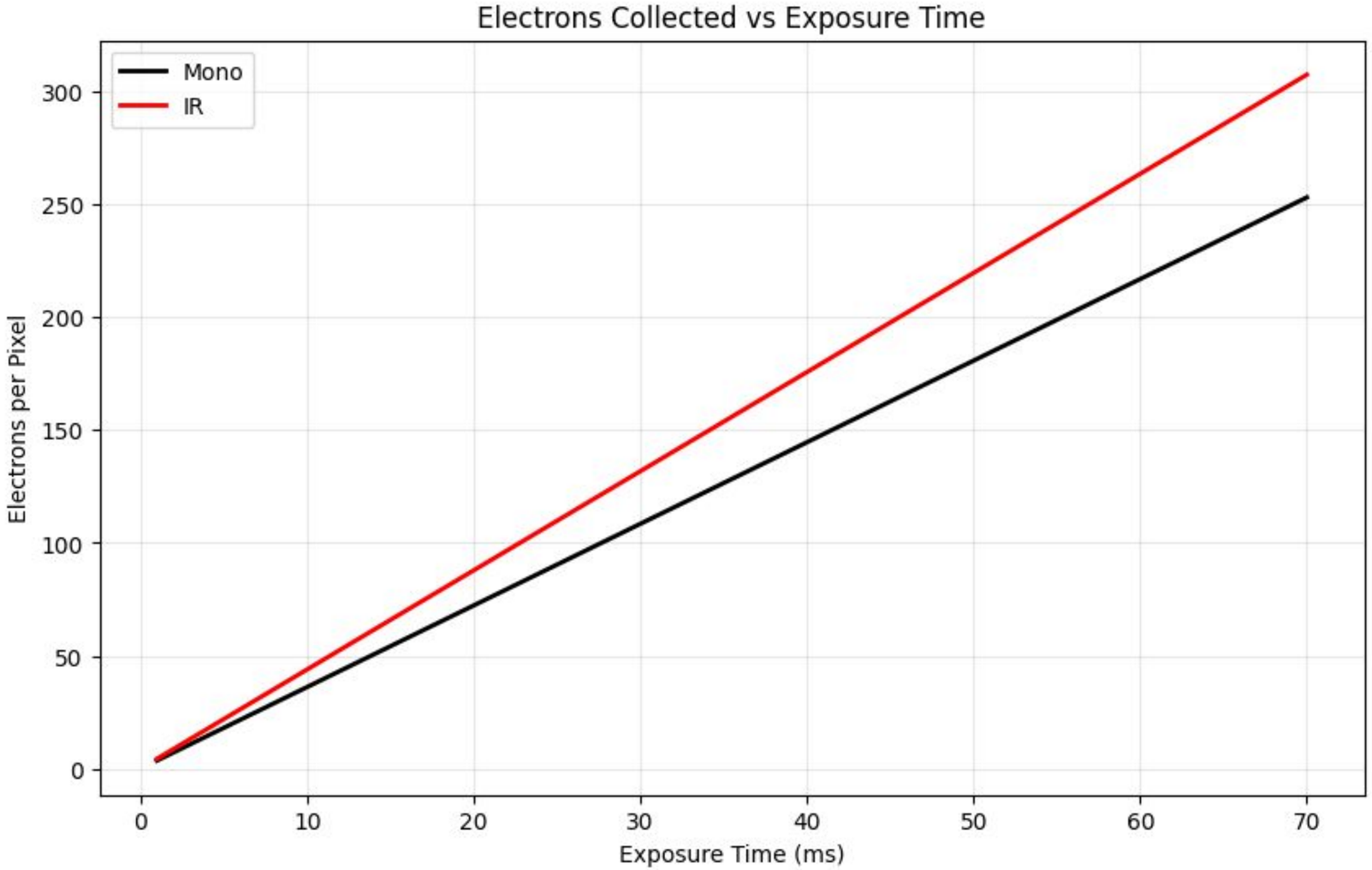
Richardson Extrapolation

$$\boxed{g'|_{true} = g'_h + \frac{1}{2^4 - 1}\left(g'_h - g'_{2h}\right)}$$

Electrons-per-pixel backup-slide

$$\text{electrons-per-pixel} \approx \left(\pi \tan\left(\frac{\text{FOV}}{2}\right)^2\right)\left(\frac{\pi\left(\frac{\text{f}}{2*\text{f\#}}\right)^2}{\text{altitude}^2}\right)\left(\text{A}_{\text{pixel}}\right)\ \text{exposure-time} * \int_{\lambda_{\text{min}}}^{\lambda_{\text{max}}} \frac{\text{Raidance}(\lambda)}{\text{E}_{\text{ph}}(\lambda)}\ \text{optical-throughput}(\lambda)\ \text{quantum-efficiency}(\lambda) d\lambda$$

Electrons-per-pixel backup-slide



Radiance from MODTRAN backup-slide

Mode
- ○ Transmittance
- ◉ Radiance

Atmosphere Model: Mid-Latitude Summer

Water Column (atm-cm)	3635.9
Ozone Column (atm-cm)	0.33176
CO_2 (ppmv)	400
CO (ppmv)	0.15
CH_4 (ppmv)	1.8
Ground Temperature (K)	294.2
Ground Albedo	0.05

Aerosol Model: Rural

Visibility (km)	23.0

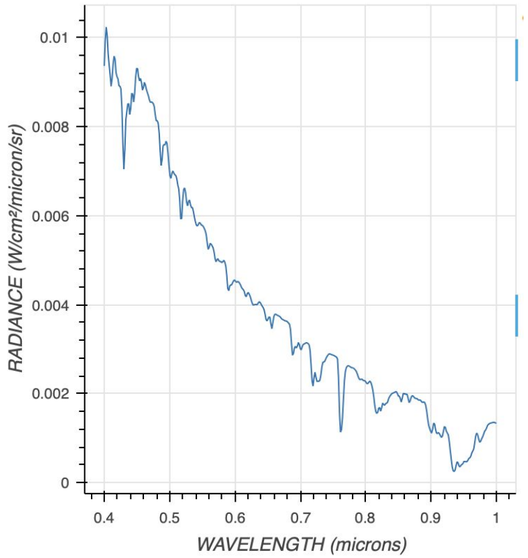
Sensor Altitude	99 km
Sensor Zenith	180 deg

Spectral Units
- ○ Wavenumbers (cm^{-1})
- ◉ Microns (μm)

Spectral Range

0.4 μm — 1.0 μm

Radiance



Flux Totals

Quantity	Total Flux (W/cm^2)
Upward Diffuse (0 km)	0.00357899
Downward Diffuse (0 km)	0.017399
Direct Solar (0 km)	0.0541974
Upward Diffuse (100 km)	0.00808776
Direct Solar (100 km)	0.0845244

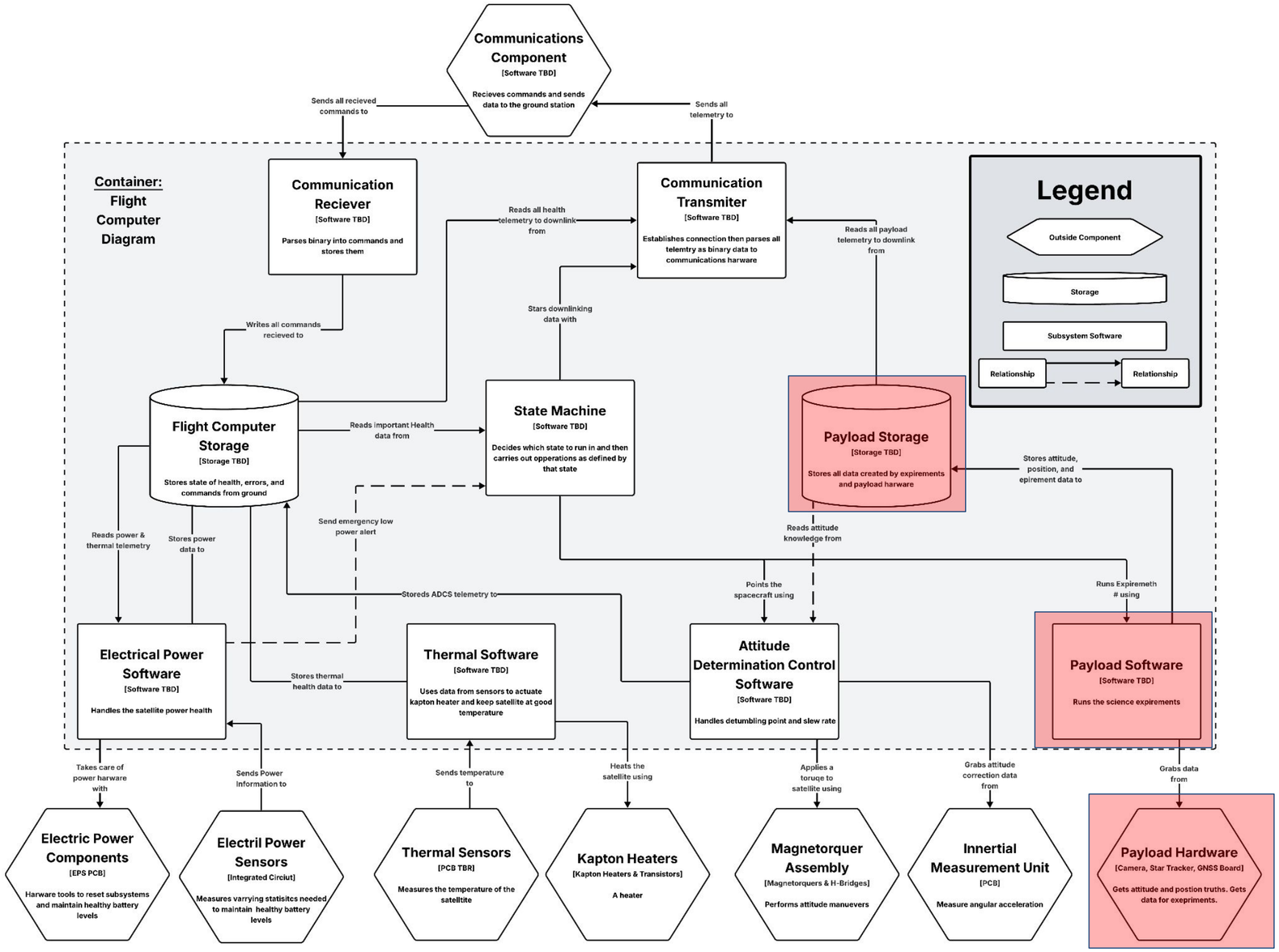
FOUND Camera Specs backup-slide

Sensor properties	
Resolution	1.3 MPix
Sensor resolution	1280 x 1024
Sensor vendor	e2v
Sensor model	EV76C661
Sensor type	NIR
Sensor technology	CMOS
Sensor size	1/1.8"
Sensor active area	6.8 x 5.4 mm
Sensor diagonal	8.7 mm
Pixel size	5.3 µm
Shutter type	Global shutter

Camera properties	
Frame rate	61 FPS
Bits	10 bits
On-chip downsampling	1x1, 2x2
Interface	USB3
Data interface speed	5 Gbit/s
Data connector type	USB3 Micro-B
GPIO count	Inputs: 1, Outputs: 1
Exposure minimum	100 µs
Exposure maximum	1 s
Power consumption	0.98 Watt
Heat dissipation	Natural convection
Sensor cooling	No
Special features	Single board

EMVA 1288 results	
Dynamic range	50.2 dB
Full well Capacity	8400 e-
Saturation Capacity	7892.5 e-
Readout noise	23.7 e-
Signal to noise ratio	38.8 dB
Peak QE	68%
Gain analog max	18 dB
Spectral Range	350 - 1050 nm

Software Requirements

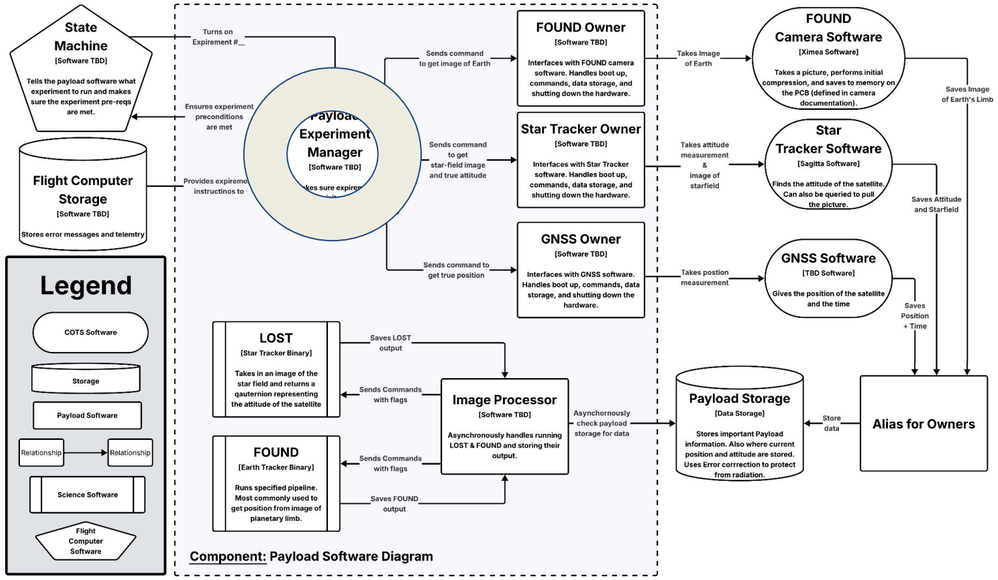


PAY-18 (Avionics)

The payload software shall **carry out experiments** as specified in the Mission Design Document

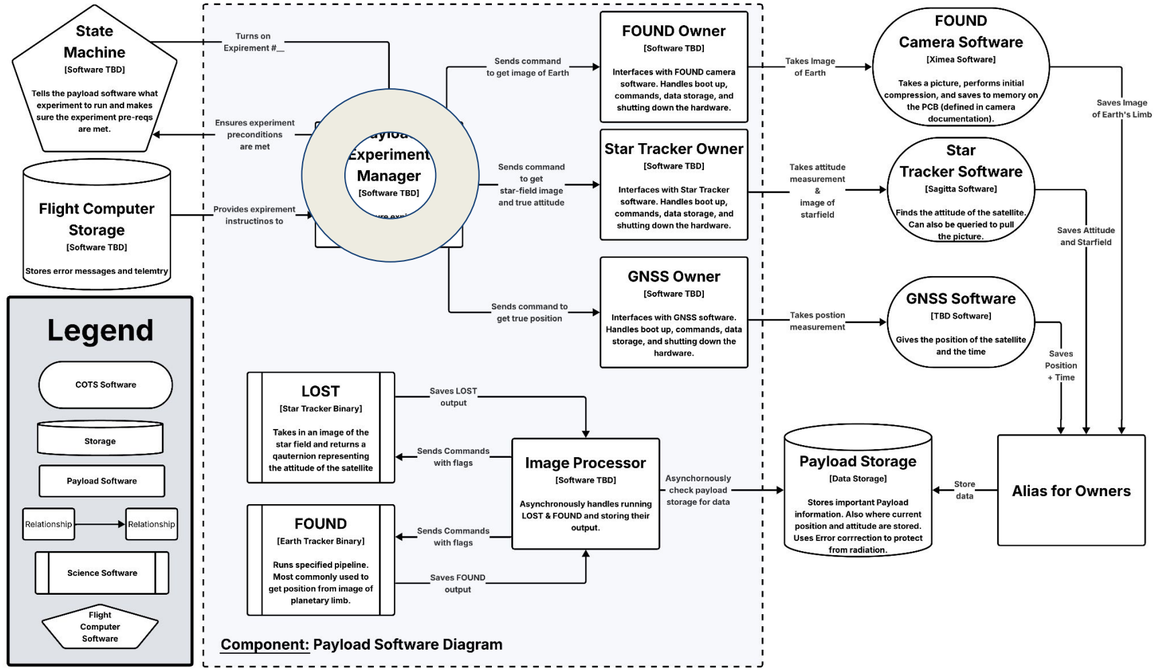
Payload Experiment Manager (PEM) Input backup-slide

```
1
2  /**
3   * This is the code passed to the FlightComputerStorage to obtain the actual
        experiment
4   */
5      struct ExperimentCode {
6      int ExperimentCode;
7      }
8
9  /**
10  * contains all information on how an experiment should be run
11  */
12     struct Experiment {
13     /**
14      * Defines the experiment mode there are four experiment modes:
15      *
16      * 0: Run the star tracker and only get attitude information
17      * 1: Run the star tracker and get attitude information as well as a
18      *     picture of the starfield then process starfield with LOST
19      * 2: Run the FOUND camera and get attitude information from the star
20      *     tracker.
21      * 3: Run the FOUND camera and get attitude infromation from the star
22      *     tracker and LOST.
23      */
24     int experiment-mode
25     // number of experiments to run
26     int number
27     // number of time inbetween each experiment in seconds
28     int rate
29     }
```



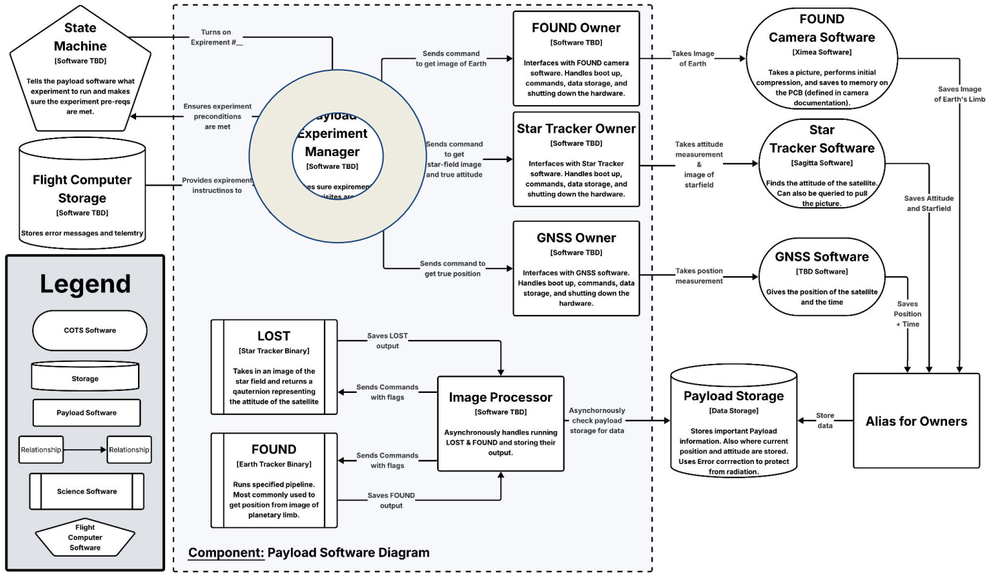
PEM Outputs backup-slide

```
1
2       PEMtoFlightComputer {
3       int Experiment Code;
4       }
5
6       PEMtoStateMachine {
7       bool arr[n]; //Confirms preconditions state, this is the attitude matrix
            required
8
9       }
```



PEM Port Interactions backup-slide

```
1
2  /**
3  *    Takes in Experiment values to be send to the Attitude Owner
4  *    and Position Owner
5  */
6
7      PEM -> StateMachine (PORT)
8      output port PEMoutput: PEM -> StateMachine (PORT) //PEM Side
9      //Input port is defined by CDH, currently TBD
10
11     PEM -> FlightComputerStorage (FCS) (PORT)
12     output port PEMoutput: PEM -> FlightComputerStorage
13     outport port FCSoutput: FCS -> PEM
14     synch input port PEMInput: FCS -> PEM
15     //Input of FCS is defined by CDH
16
17
18 }
```



PEM Pre-conditions + Implementation backup-slide

Preconditions, Post Conditions, Errors

For the FOUND/LOST along with our star tracker truth measure to function correctly the satellite must have the correct attitude. This is our Precondition. Currently there is no check to see if the satellite has the correct attitude and instead a command is sent through the flight computer (state machine) which then communicates with ADCS to adjust the attitude. The picture and adjustment occur asynchronously, as in simultaneously. The satellite should have attitude adjusted by the time of picture but will take the picture regardless of attitude. Before the command to ADCS is sent the PEM will check the current attitude.S

Implementation

Listing 3.5: Image Processor Implementation

```
1
2  active component PayloadExperimentManager {
3      void ExperimentIn_Handler (port num, context) {
4              Get Experiment
5              Split Experiment into 3 structs
6                  ->StarTracker
7                  ->FoundCam
8                  ->GNSS
9              Check utility softwares for active experiments
10                 if (false) {
11                     Send Experiments to Owners
12                 }
13                 else {
14                 Queue the experiments and wait for a rate group call to
                       recheck
15                     status of the utilities
16                 }
17     }
18
19 }
```

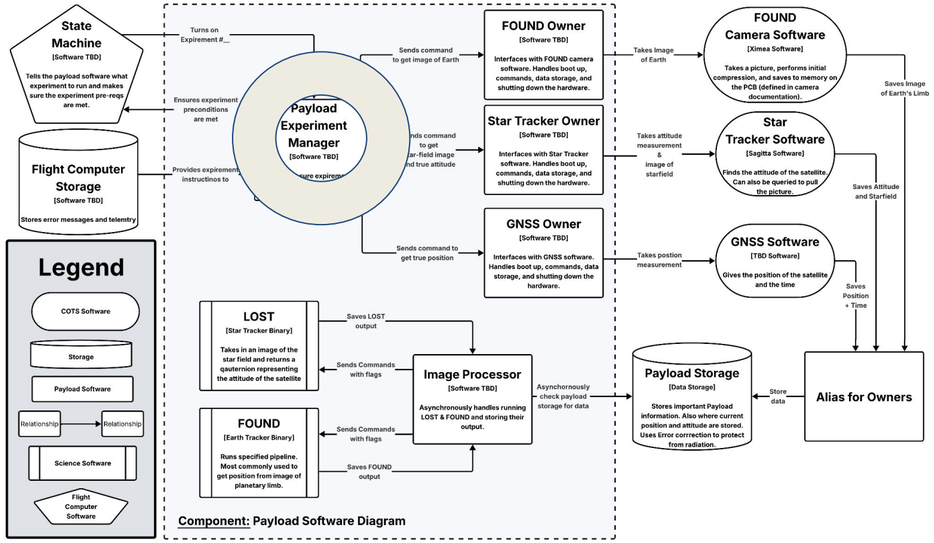


Image Scheduler Description + Input backup-slide

3.2 Image Scheduler (Within the PEM)

This is part of the PEM component as keeping a separate component is a bit redundant and introduces an unnecessary extra step. This takes advantage of the queued nature of an active component in FPrime. It will speak to the owners for what the frequency and amount of measurements to take

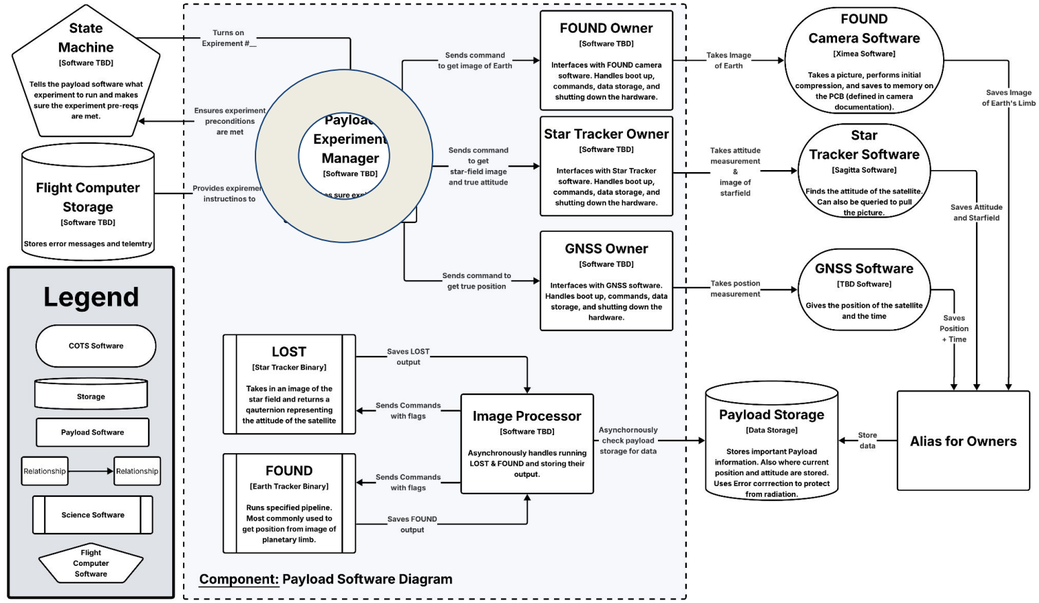


Image Scheduler Outputs backup-slide

Outputs

Listing 3.7: Image Scheduler OutPuts

```
/**
*    This will establish how many photos and at what rate the star
*    tracker and GNSS receiver will be pulling attitude/position
*    measurements, these are sent at the same time.
*/
     struct StarTracker {
          int photos;
          float rateStar;

   }
     struct FoundCam {
          int photos;
          float rateFou;
  }

     struct GNSS {
          int positions;
          float rateGNSS;

     }
```

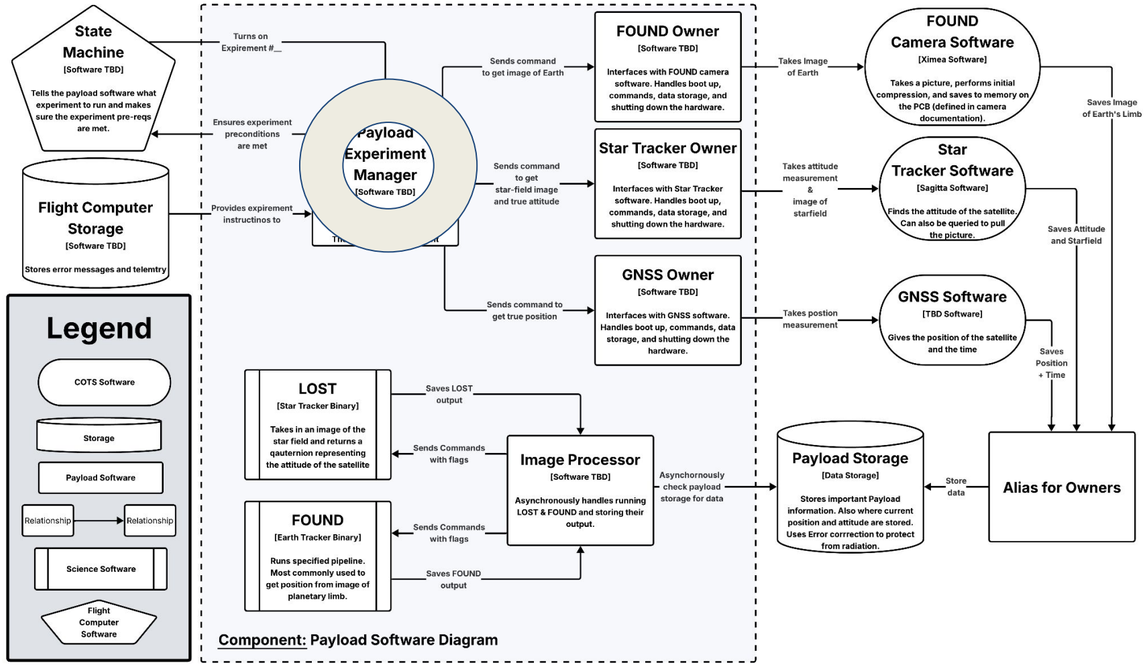


Image Scheduler Ports + Preconditions backup-slide

A collection of port connection and input Handlers than depend on other port connections Ports can be repeated so different components have the same port (different handler implementations) The image scheduler worries about the current state of the utility software and the owners are just compartmentalizing the functions to keep talking to the utilities simple

Listing 3.7: Image Scheduler Ports

```
1
2       //FOUND
3        output sendToOwner(port num, context )
4           -> synch input receiveCommand(context)
5       //StarTracker
6        output sendToOwner(port num, context )
7           -> synch input receiveCommand(context)
8       //GNSS Receiver
9        output sendToOwner(port num, context )
```



```
10          -> synch input receiveCommand(context)
11
12      //Follows FIFO still
13       synch input SchedIn_handler: rategroup -> PEM(image scheduler)
```

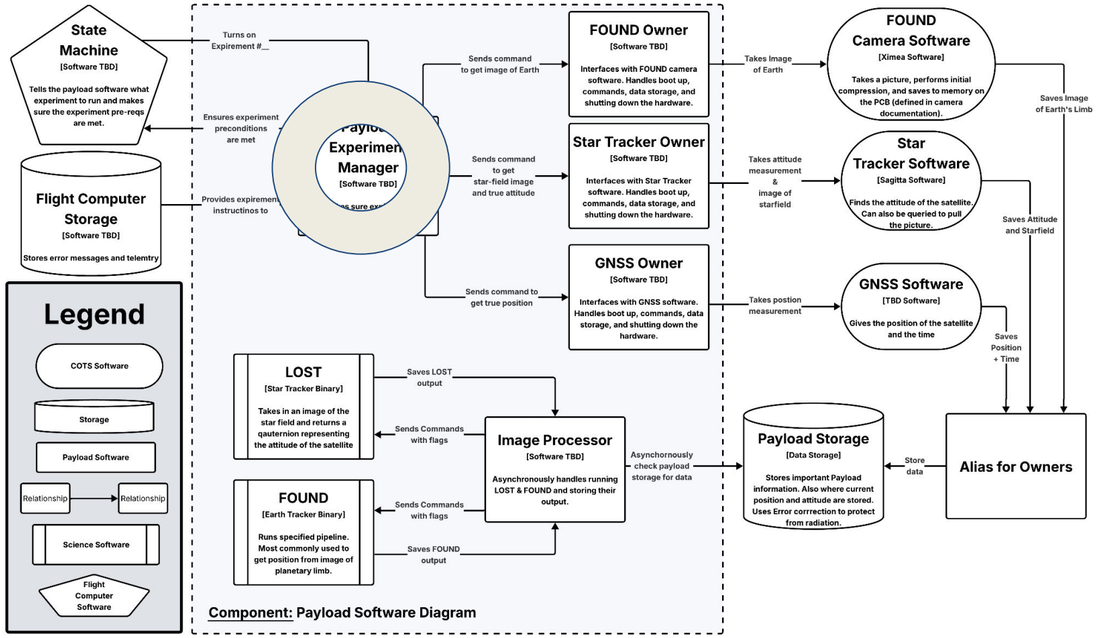


Image Scheduler Ports + Preconditions backup-slide

```
9
10  queued component Image Scheduler {
11      //
12
13      schedIn_handler(port num, context) {
14          getStatus of current experiment //an enum?
15              -> if(null) {
16                  check if camera is on
17                      -> if(false) {
18                          send command to turn on
19                      }
20                      -> if(true) {
21                          popoff top of the queue
22                          send to handlers
23                      }
24
25              }
26              -> if(ongoing) {
27                  return 0; //does this even work here?
28              }
29              -> if(paused) {
30                  check if cameras are on
31                      -> if(false) {
32                          send command to turn on
33                      }
34                      -> if(true) {
35                          popoff top of the queue
36                          send to handlers
37                      }
38              }
39
40
41      }
42
43  }
44
45  }
```

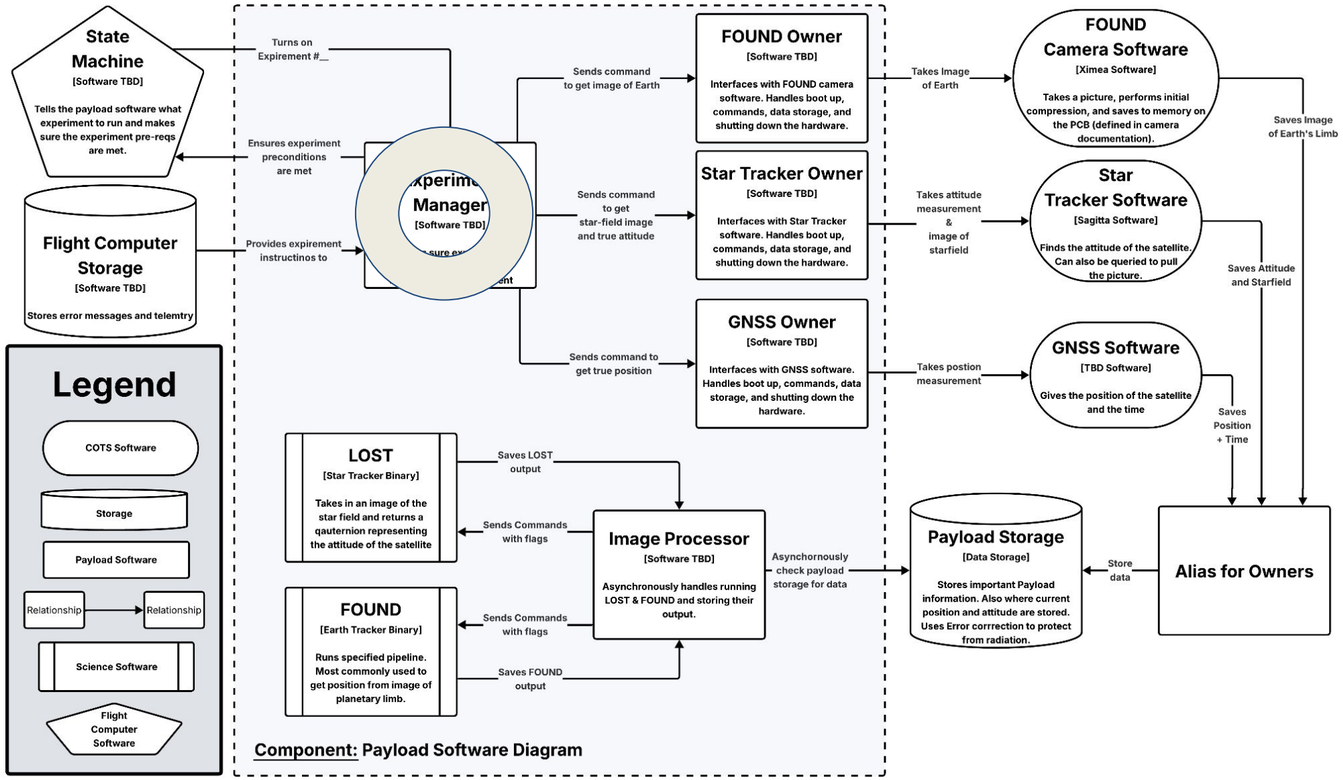


Image Scheduler Ports + Preconditions backup-slide

3.3 Image Processor

*Potentially add a component between PEM and image processor that stores commands and ir address together

*Completely get rid of the PEM to the IP port and just have the command bundled with the comr to take the photos and just follows it into the Payload storage. (Tentative)

Inputs

Listing 3.11: Image Scheduler OutPuts

```
1  /** A command that allows the image processor to asynchrnously process th
       image
2  *      that will be coming in decompress image for both FOUND and LOST +
       error  correction
3  */
4  /**
5      String flags: String representation of flags from PEM
6      String filepath: address of image to run Lost & found on from payload
           storage
7
8  */
9  struct ImageProcessorCommand {
10     bool processWithLost;
11     bool processWithFound;
12     std::string flags;
13     std::string filepath;
14 }
```

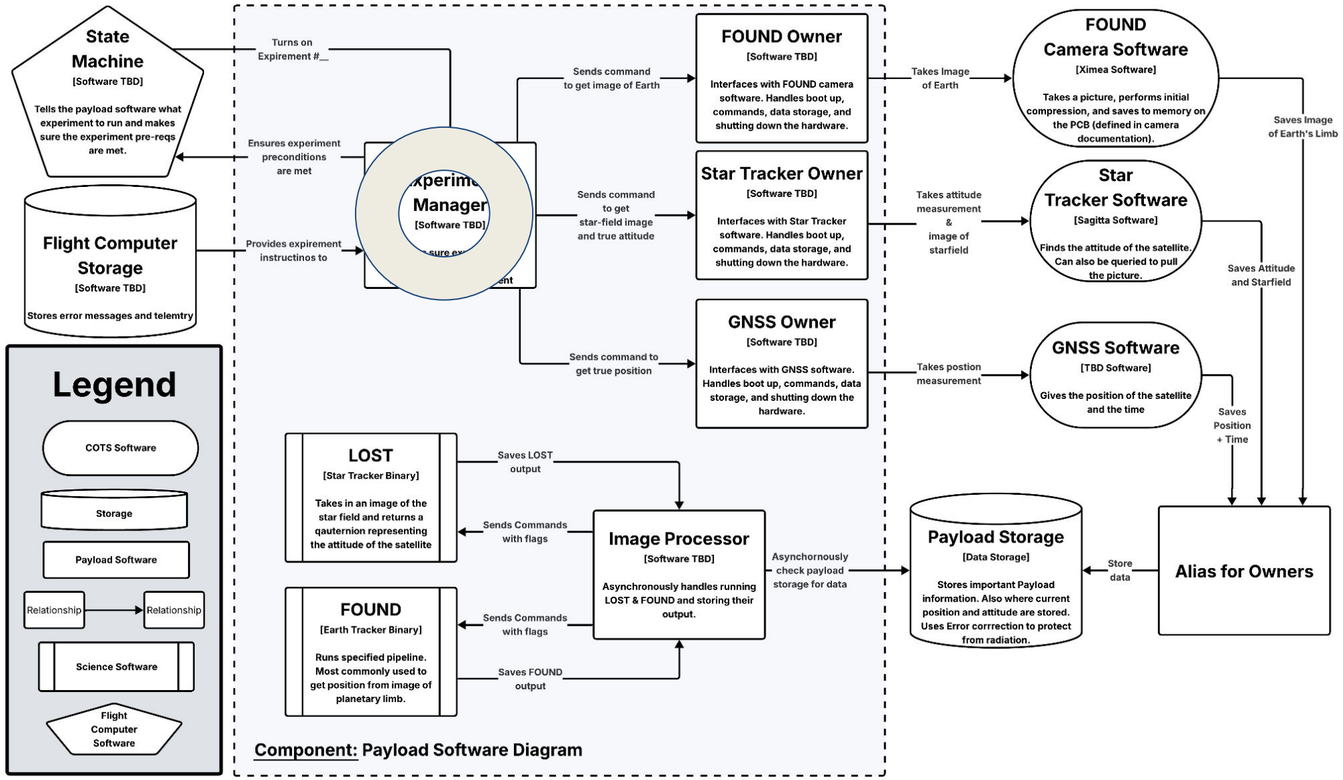


Image Processor backup-slide

Outputs

Listing 3.12: Image Scheduler OutPuts

```
/**
* ProcessedImageData is not actually the output itself, just part of the
    command send to respective binaries
* LOSTCommand and FOUNDCommand contains imgPointer and flags.
*/

struct ProcessedImageData {
    uint32_t width;
    uint32_t height;
    float* intensityMatrix;
    size_t matrixSize;
    int processingStatus;
}
```



```

struct LOSTCommand {
    ProcessedImageData* LOSTimg;
    std::string flags;
}
struct FOUNDCommand {
    ProcessedImageData* FOUNDimg;
    std::string flags;
}
int processingError;
/**
* Image file does not exist or is not accesseible. processingError = -1
* Image format is invalid or unsupported. processingError = -2
* Unable to correct image errors. processingError = -3
* Cannot communicate with LOST/FOUND binaries. processingError = -4
*/
```

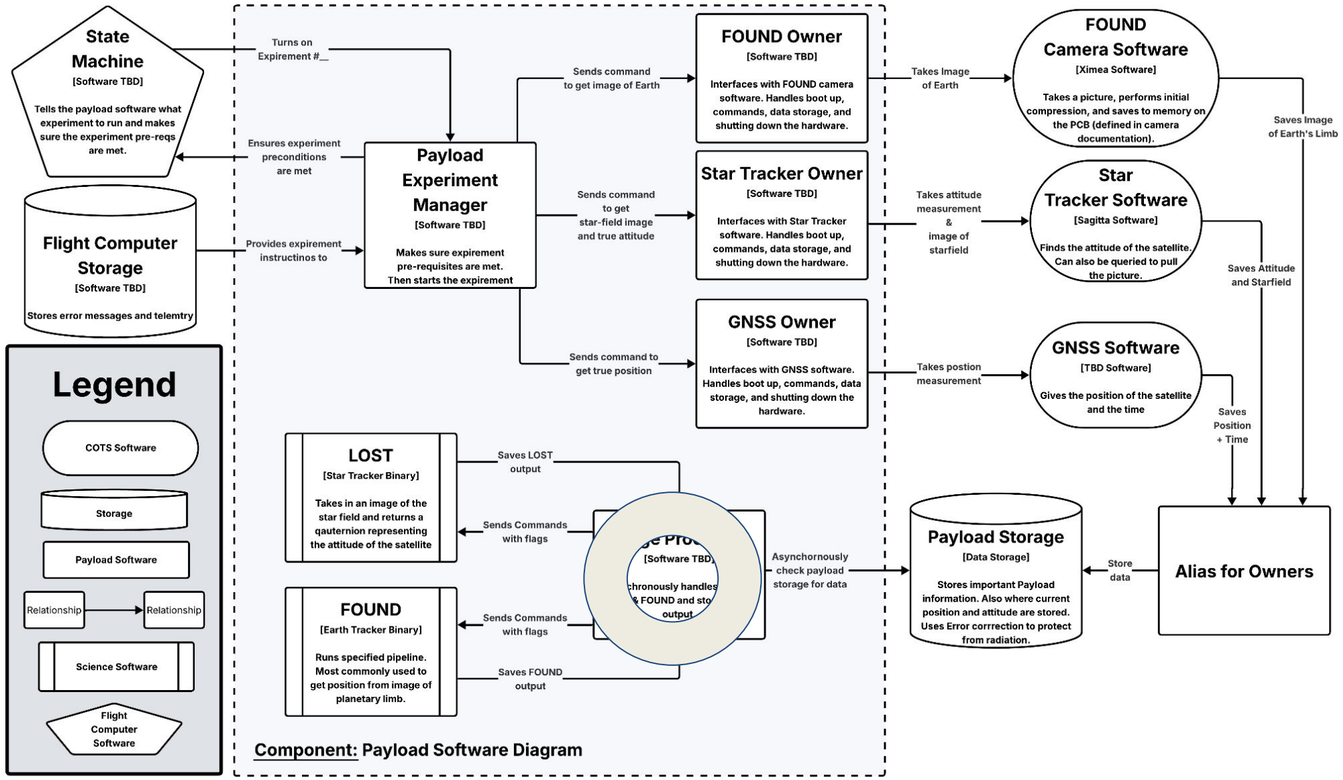


Image Processor backup-slide

Preconditions, Post Conditions, Errors

Preconditions: imageFilePath must point to a readable image file either processWithLost or process-WithFound should be true Sufficient memory should be available for decompressed image matrix.

Post Conditions: Image is decoded, error corrected, and decompressed to intensity matrix Processed data is sent to specified binaries (LOST/FOUND) Original image file remains unchanged Resources (file handles, memory) are properly released

Errors: Image file does not exist or is not accessible. processingError = -1
Image format is invalid or unsupported. processingError = -2
Unable to correct image errors. processingError = -3
Cannot communicate with LOST/FOUND binaries. processingError = -4

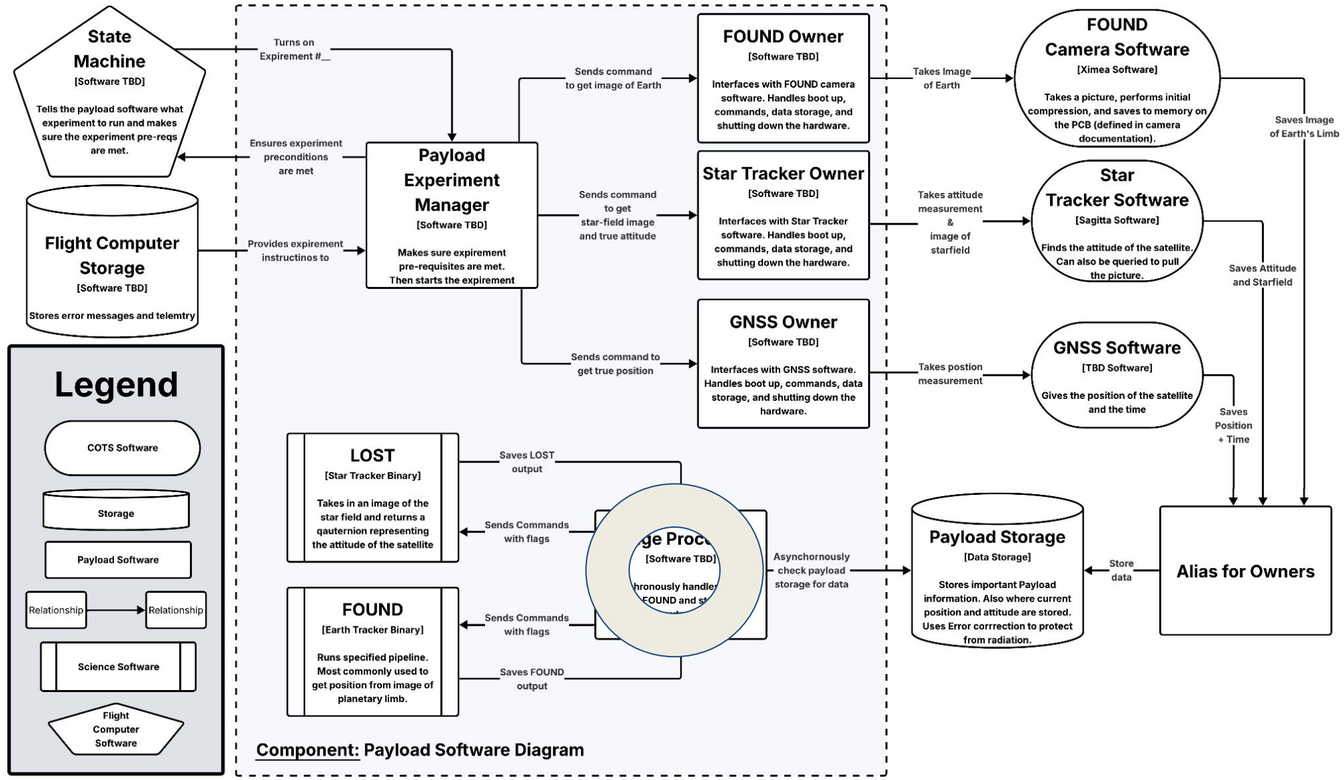
Implementation

Listing 3.12: Image Processor Implementation

```
/**
* Takes an ImageProcessorCommand
* gets the compressed image from payload storage
* does error correction and decompression
* then forwards to LOST/FOUND
*/

component ImageProcessor {
    // called by PEM
    void ProcessCommandIn_handler(const ImageProcessorCommand& cmd) {
        processingError = 0
        ProcessedImageData data;
        data.processing.status = -1

        // check that we're actually doing something
        if (!cmd.process.WithLost && !cmd.processWithFound) {
            processingError=-2;

    return;
}
```

Image Processor backup-slide

```
//This relies on the PEM having access to the storage, which needs
   to be added*
CompressedImage img = readCompressedImage(cmd.filepath);


if (!img.valid) {
    processingError = -1;
    return;
}

CorrectedImage corr = applyErrorCorrection(img); //We already spoke
   about this
if(!corr.valid) {
    processingError = -3
}

// Error correction. We're going to go with error correction instead
    of detection to err on the side of safety. Since we don't know
   what the camera will see in orbit, we cannot determine the
   thresholds for error detection.

 //img handling (meta)
 data.width=corr.width;
 data.height=corr.height;
 data.intensityMatrix=decompressToIntensity(corr);
 data.matrixSize=data.width*data.height;
 data.processingStatus=0; //success

 //LOST/FOUND
 if(cmd.processWithLost){
     LOSTCommand lc;
     lc.LOSTImg = &data;
     lc.flags=cmd.flags;
     send_LOSTCommandOut(lc)
 }
 if(cmd.processWithLost){
     FOUNDCommand fc;
     fc.FOUNDImg = &data;
     fc.flags=cmd.flags;
     send_FOUNDCommandOut(lc)
 }
```



Attitude Owner backup-slide

Inputs

The Attitude owner just takes in one part of an experiment as well as a potential ADCS request. It will then create a new struct at the time of receiving either struct or both to make one request.

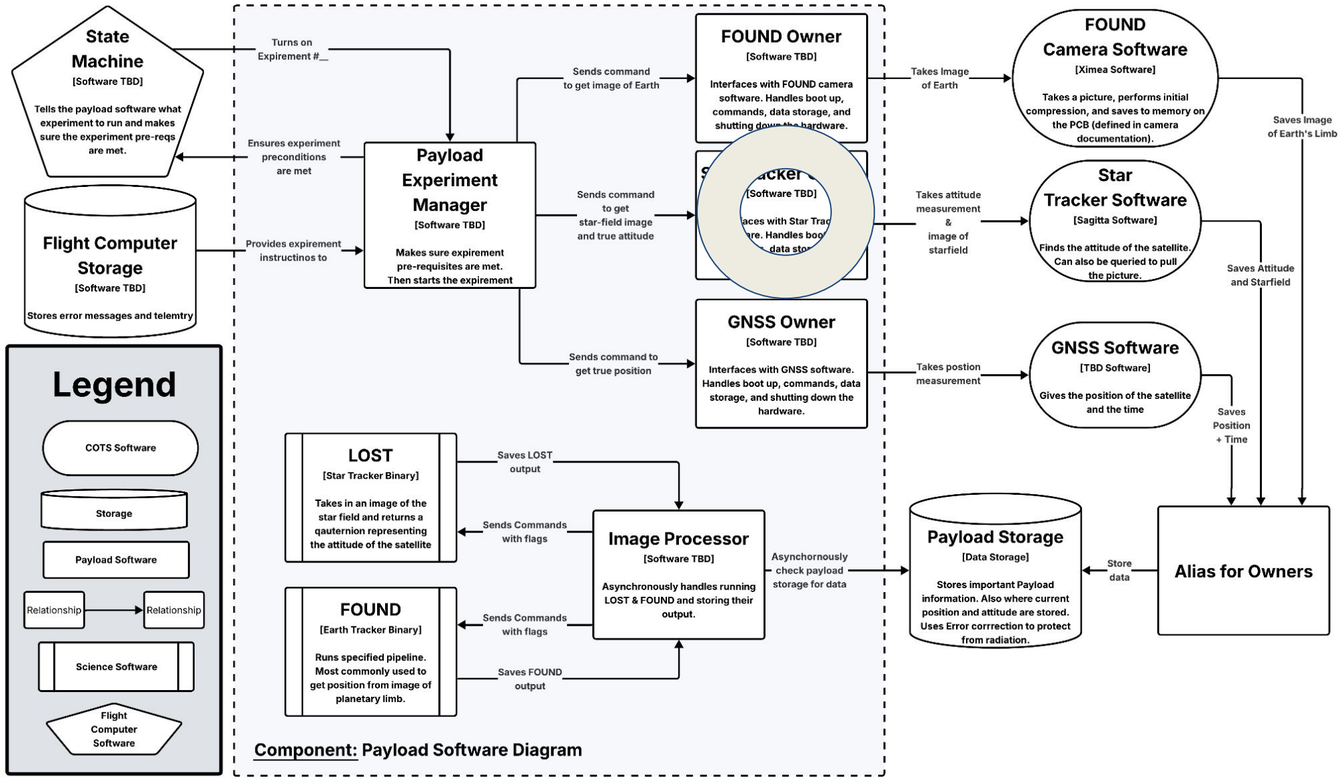
```
1   struct ExperimentStar {
2       int takeImages;
3       float rate;
4   }
5
```

3.4 Utility Interface | 25

```
6   struct ADCSRequest {
7       bool getAttitude
8   }
```

Outputs

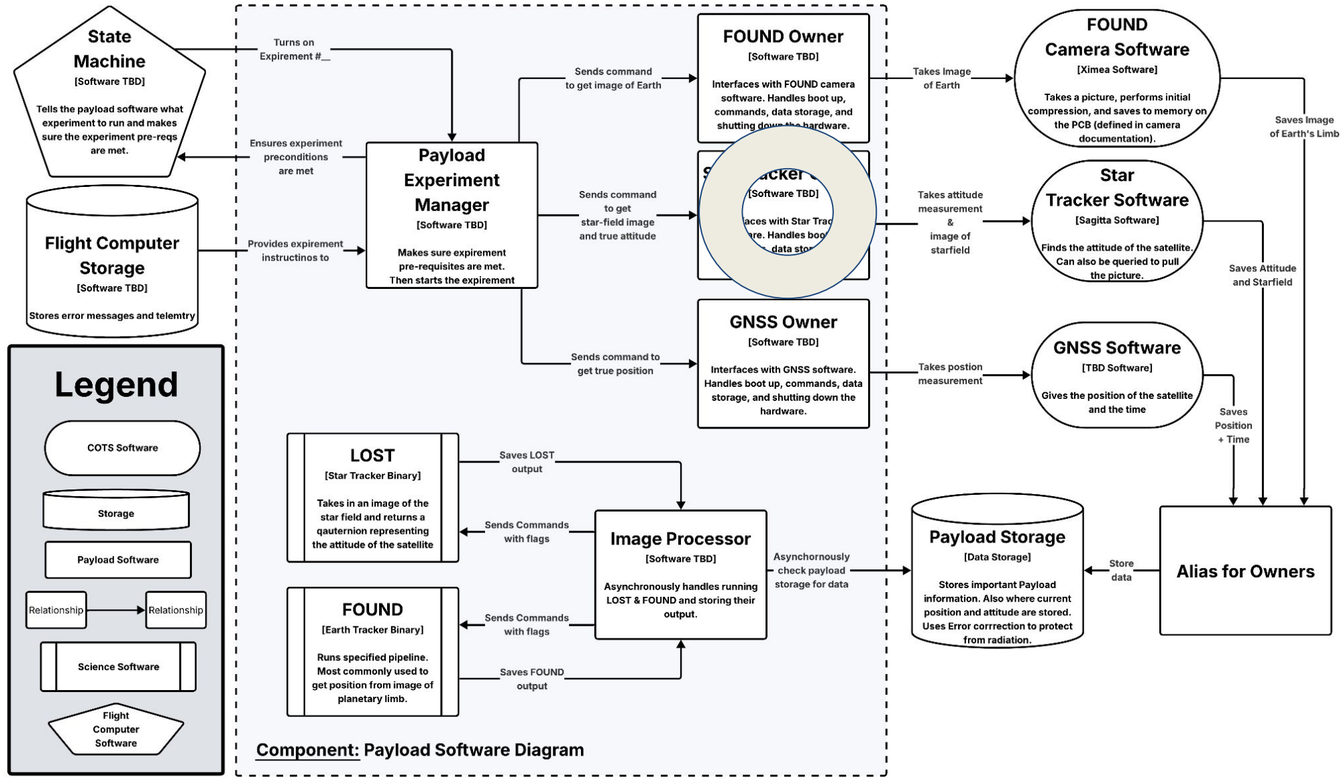
```
1   struct ExperimentOutput {
2       arr[n][m] attitude;
3   }
4
5   struct experimentOut {
6       image* StarImage;
7   }
8
9   struct Active {
10  bool cameraOn;
11  bool experimentActive;
12  }
13
14  struct Data:
```



Attitude Owner, Implementation backup-slide

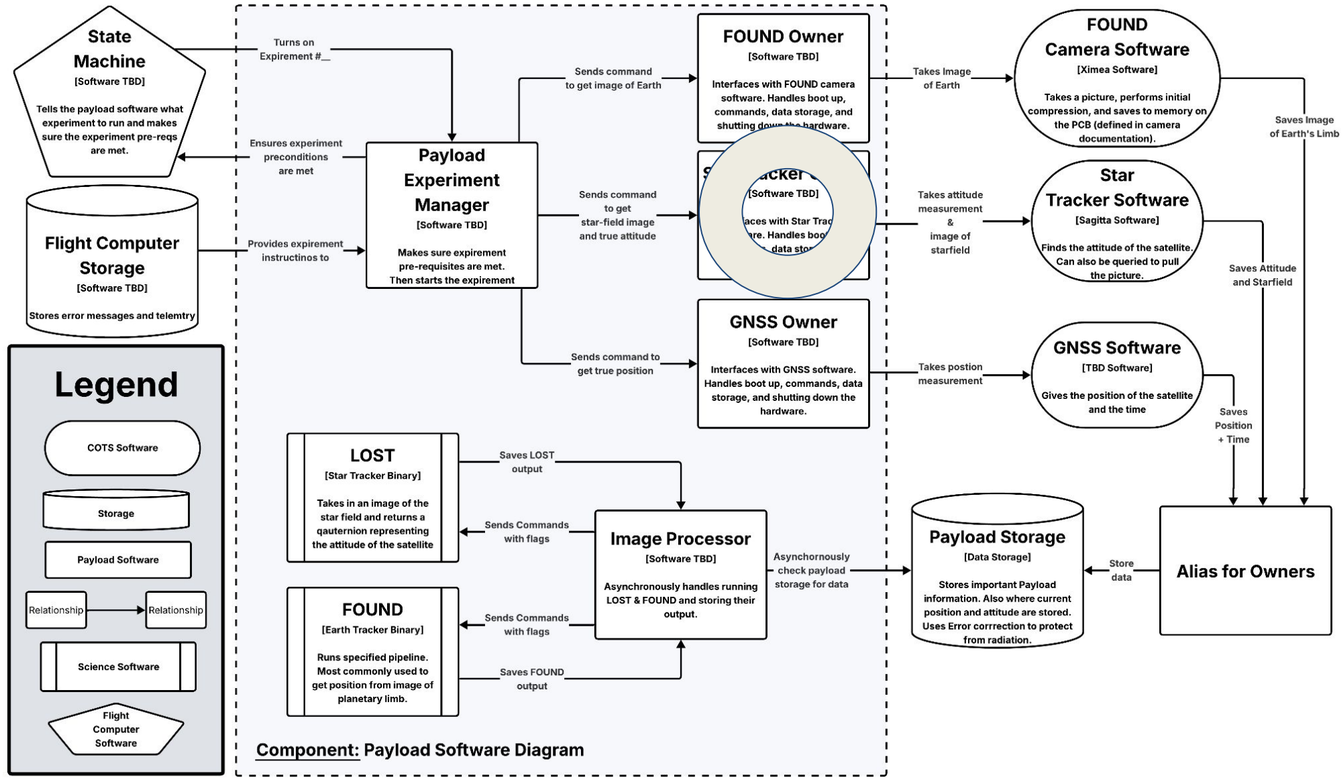
Type 1: The utility is turned off between measurements Type 2: The utility is kept on between measurements Attitude Owner Each owner has a frequency of measurements that decide the state of the utilities between measurements

```
    //Each type contains a way to turn on and
    active component AttitudeOwner {
        void ExperimentIn_handler(port num, Context) {
        if(Camera on && No active experiment) {
            if (Experiment.rate > frequency) {
                enable type 1//Takes images and turns off startracker
                    between images
            }
            else {
                enable type 2//Takes images without turning off star tracker
            }
        }
        }

    }

    Startup() {
        if(startracker off)
        send ON to startracker
    }


//Will command the camera software component to capture space
    takeImage() {
        if (Camera is on) {
            if(No active experiment) {
            Send the command to take the image
            }
        }
        else {
        LogEvent(-4);
        }
    }




    ReturnImage() {
        get Image from star tracker

        }
```

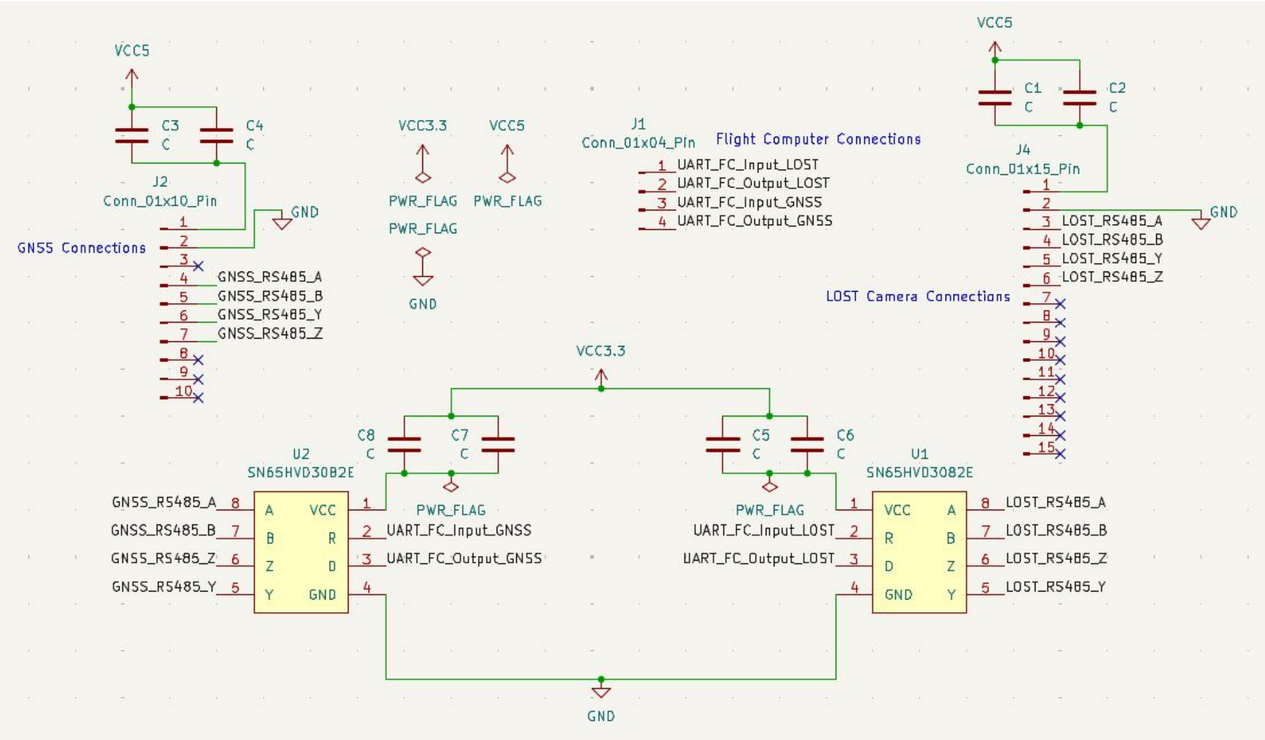


Attitude Owner, Implementation backup-slide

```
45      GetAttitude() {
46          take in attitude
47          check to see if attitude is valid
48              if(valid) {
49                  send attitude to ADCS unit
50                  }
51              else {
52              LogEvent(-1)
53              }
54
55
56      }
57
58  //Done by image scheduler when no experiments are to be run
59      turnOffStarTracker() {}
60      turn off the camera //some input port?
61      if(false) {
62      LogEvent(-3)
63      }
```



Interface back-up slide



Schematic

- LOST camera (Sagitta) and GNSS Receiver (TBD)
  - RS-485 communication
- Full-Duplex Transceivers ⟶ Flight computer
  - RS-485 to UART “translation”

Mission Objectives back-up-slide

MSC-1	Determine and deliver the attitude of a satellite using LOST and compare with truth measure, at 4(TBR) equally spaced +- 10 (TBR) degrees apart points in the orbit with error within 360 (TBR) arcsec **around-boresight**.
MSC-2	Determine and deliver the attitude of a satellite using LOST and compare with truth measure, at 4(TBR) equally spaced +- 10 (TBR) degrees apart points in the orbit with error within 180 arcsec **cross-boresight**.
MSC-3	Determine and deliver the position of a satellite using FOUND and compare with truth measure, at 4(TBR) equally spaced +- 10 (TBR) degrees apart points in the orbit with error within 60 km (TBR).