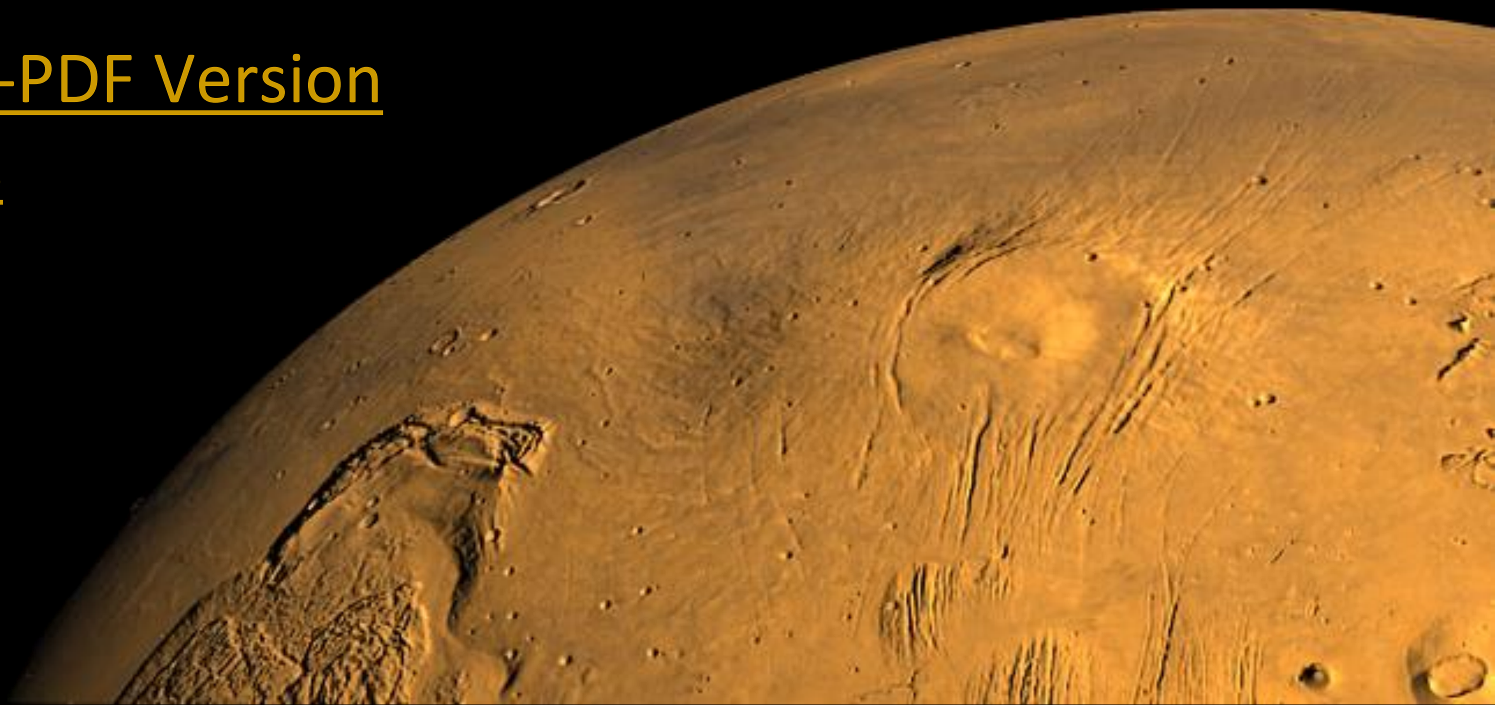
Stephen Mock Work Portfolio

[Link to Non-PDF Version with Videos]



About Me

- Graduated with Bachelor’s and Masters in Mechanical Engineering at Georgia Tech
  - Concentration in Robotics
- Worked at NASA JPL, iRobot, SharkNinja
- Interested in space, consumer products, IoT
  - Enjoy machine design, mechatronics, coding, robotics
- Outside of work
  - Sports, Music, IoT Projects
- Contact Info
  - sjmock99@gmail.com
  - Personal Website

Skills

Hardware:

- CAD Design
  - SolidWorks, Creo
    - EPDM
- Prototyping
  - 3D Printing, Laser Cutting
  - Mill, Lathe, etc
  - Soldering
- Controllers / Controls
  - Arduino, ESP32, Teensy
  - Raspberry Pi, Intel NUC
  - PID
  - Robot Kinematics
- Sensors
  - Force Torque (FTS), Thermocouples, IMU, Encoder, Infrared, Ultrasonic, Hall, Humidity/Temperature, Laser Displacement
- Components
  - Stepper Motors, Brushed Motors, Motor Drivers, 8020, Servos, Relays, Power Supplies, Thermal Controllers, various electronics

Software:

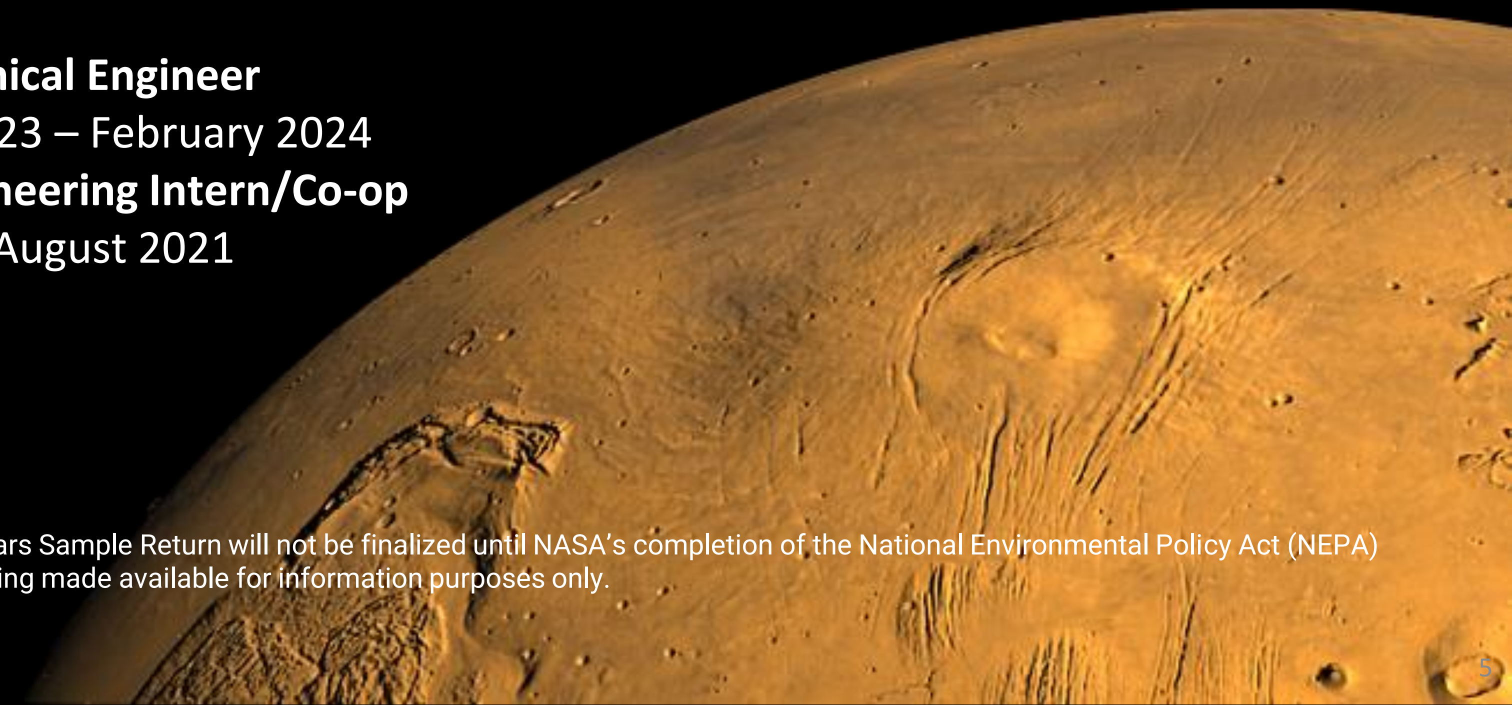
- Programming Languages / Frameworks / OS
  - Python, C/C++ [for Microcontrollers], MATLAB, LabView
    - Basic HTML, CSS, JS, Java
  - ROS1, ROS2, MicroROS
  - Linux (Ubuntu)
- Networking / Protocols
  - Serial Protocols: I2C, SPI, UART
  - MODBUS TCP, TCP/IP, SSH, VISA, SCPI
  - MQTT
- Applications
  - MATLAB, Git, SciKit Learn (Machine Learning), OpenCV, Linux, Jupyter Notebook, Notion, LabView, PlatformIO, VSCode

Portfolio Table of Contents

NASA Jet Propulsion Lab
Summary of Work

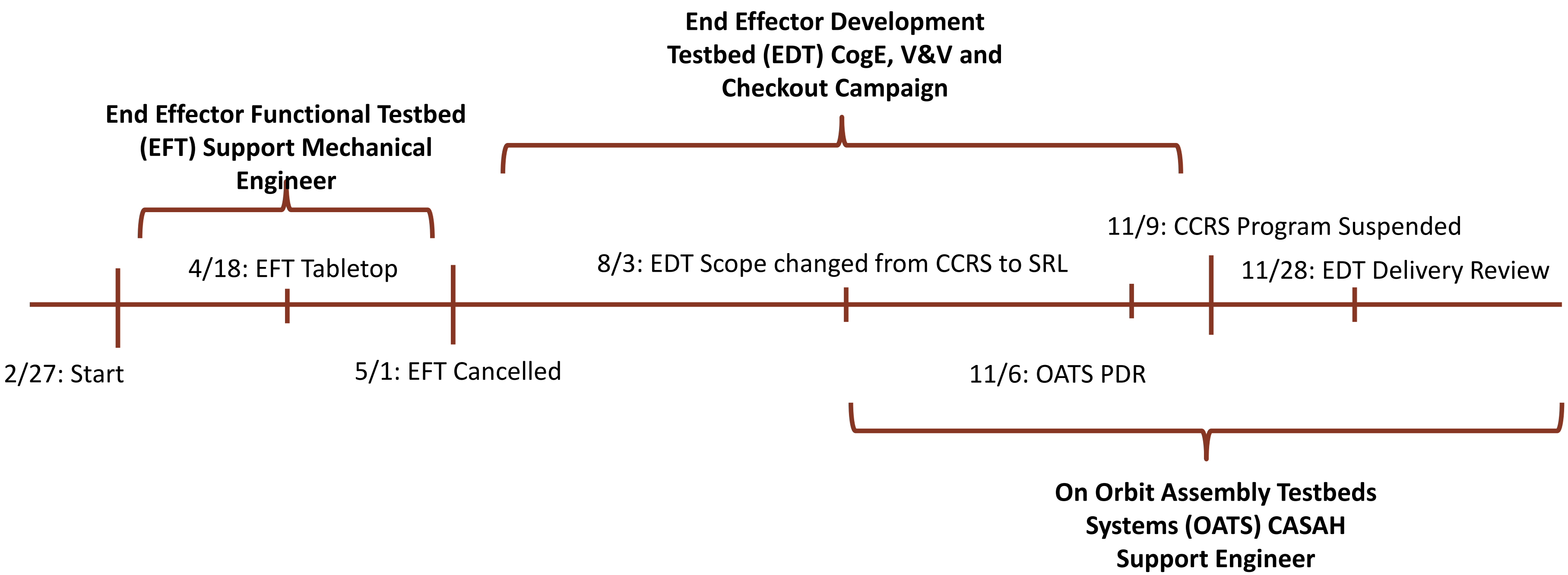
Robotics Mechanical Engineer
- February 2023 – February 2024

Mechanical Engineering Intern/Co-op
- May 2020 - August 2021



The decision to implement Mars Sample Return will not be finalized until NASA's completion of the National Environmental Policy Act (NEPA) process. This document is being made available for information purposes only.

Overview – CCRS Testbeds roles held by Stephen Mock



Org chart on next slide

On Orbit Assembly Testbeds Systems (OATS) Org Chart

Testbed Management (A01)

OATS Lead/CAM – Amila Cooray (352A)
Lead SE – Vladimir Arutyunov (347R)

Systems Engineering (A01)

RSE – Vladimir Arutyunov (347R)
MSE – Christine Gebara (355L)
Controls Engineer – Teo Wilkening (347A)
Operations Engineer – Anas Delane (347R) (0.5)

Testbed Mechanical (B01)

PIT Structure Engineer– Christine Gebara (355L)
EDT Cog-E – Stephen Mock (347C)
PIT Testbed CogE – Taylor Sun (355A)
Testbed Engineer – Denise Garcia (355A)
Testbed Engineer – Kelsy Coston (355A)
Testbed Engineer– Stephen Mock (347C)
Testbed Engineer – Stephen Gerdts (347C)
APX Student – Matt DiMarzio (355H)
APX Student – Joy Liu (347R)
Designer – Leon Huntsman (0.5) (355D)
Designer – Anasheh Tooroosian (0.5) (355D)
Structural Analyst – Mike Beale (0.25) (355H)
Technician – James Bailey (0.5) (357C)
Technician – John Phu (0.5) (357L)
Manufacturing Engineer – Ryan Scherich (0.5) (357A)
Thermal Support – Juan Villalvazo (0.25) (353F)

Testbed Electrical (C01)

EGSE Lead – Emilio Vazquez (1.0) (337C)
EGSE Support – Jose Fernandez (0.25) (337C)
Support Engineer – Heidy Kelman (0.5)(347A)
Harness Engineer – *TBD (337C)*

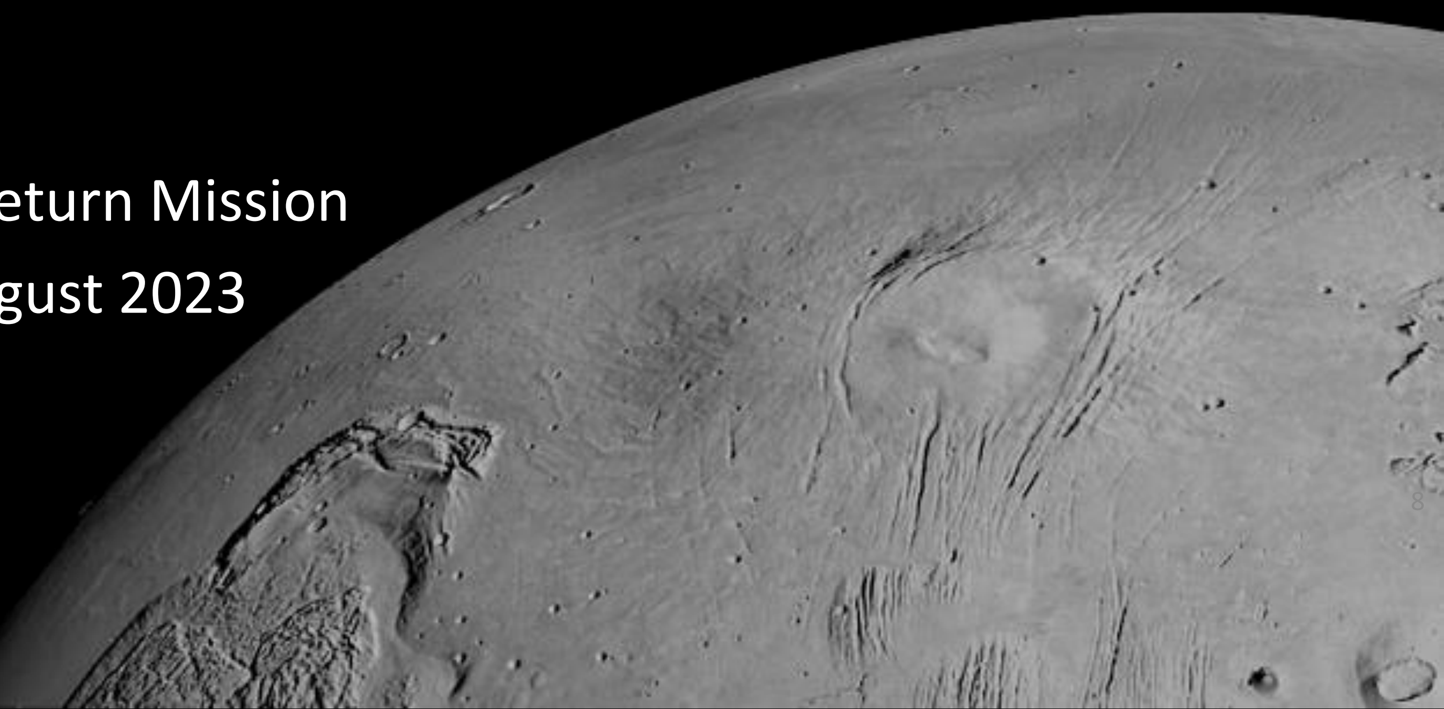
Testbed Software (D01)

CASAH Engineer – David Kim (0.5) (347G)
CASAH Support Engineer *–TBD (1.0)(347G)*
CASAH Support Engineer – Stephen Mock (347C)
Labview Engineer – Michael Errico (0.5) (3468)

End Effector Development Testbed (EDT) V&V

Mars Sample Return Mission

May 2023 – August 2023



Preface

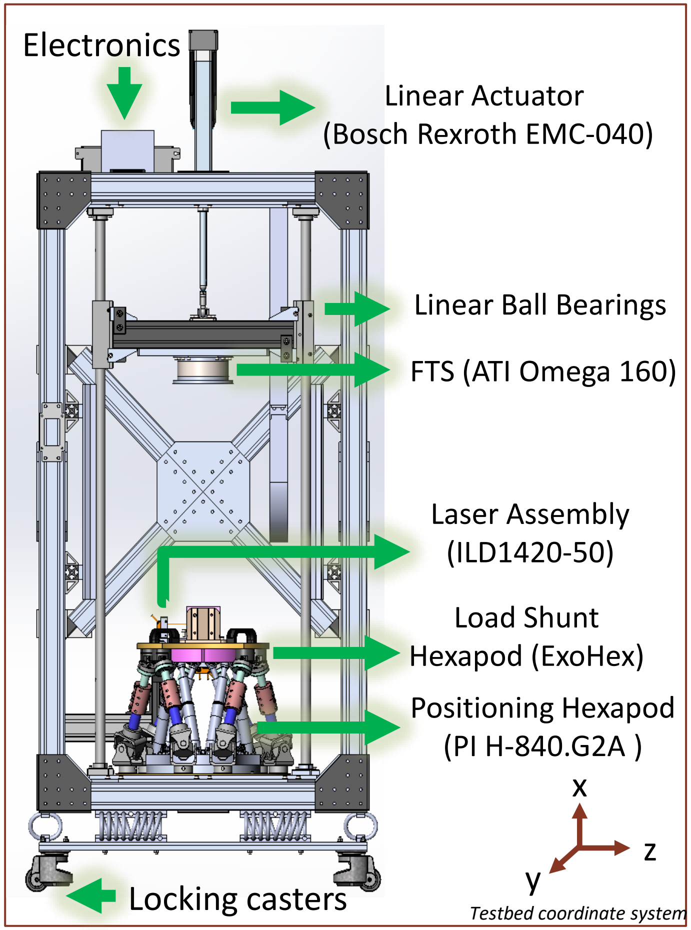
History

The End Effector Testbed (EDT) was created as part of the Capture Contain Return System (CCRS) Testbeds team to provide a venue to test prototype CCRS end effectors starting in Summer 2022. The objective of the testbed was to measure force and torque data during insertion for misaligned interfaces. It was a successor to a previous testbed for which the inner hexapod was originally purchased. With increases in load requirements, a larger 8020 structure and linear actuator were implemented for high axial loading and clocking moments. An ExoHex was designed to enable the inner hexapod to still be used for precise positioning, without having to survive high loads. EDT V&V started in May 2023, but was later rescoped in August 2023 to be delivered to the Sample Retrieval Lander (SRL) team in November 2023. As a result, the purpose of the testbed and checkout tests were focused on general functionality rather than CCRS specific implementation. This package highlights the capabilities of EDT, as well as the performed checkouts, and reference information. The checkouts relate to validating the basic functionality of the testbed, particularly for safety purposes.

JPL Team

- **EDT V&V**: Stephen Mock (347C)
- **EDT Software**: Michael Errico (3468)
- **EDT Design / Build / History:**
  - Vladimir Arutyunov (347R)
  - Stephen Gerdts (347C)
  - Jake Chesin (347B)
  - Heidy Kelman (347A)
- **EDT CAD:** Heidy Kelman (347A)

EDT Overview



Design Intent: Simulate misalignments in 6DoF such that forces/torques can be measured during end effector functions

CONOPS

Loosen ExoHex Legs → Move Inner Hexapod to Position → Lock down ExoHex Legs → Demate Inner and ExoHex → Linear Actuator moves to contact* → Linear Actuator moves to preload* → End Effector Functions → Linear Actuator backs off and resets to free space

**Forces/Torques measuring during whole operation*

Main Component Capabilities

Bosch Rexroth EMC-040
- 305 mm Stroke
- 3.4 kN rated peak axial load
- Absolute encoder
- Optional limit switches (not installed)

ATI Omega 160
- Dual Calibration
- High Load: SI-2500-400
- Low Load: SI-1000-120

ExoHex
- 31kN axial load capacity at zero position
  - based off single leg proof test to 5500N
  - full assembly not proofed
- ±10mm lateral
- ±1.15° rotation (tip/tilt)
- ±0.57° rotation (clocking)
  - *Tested values*

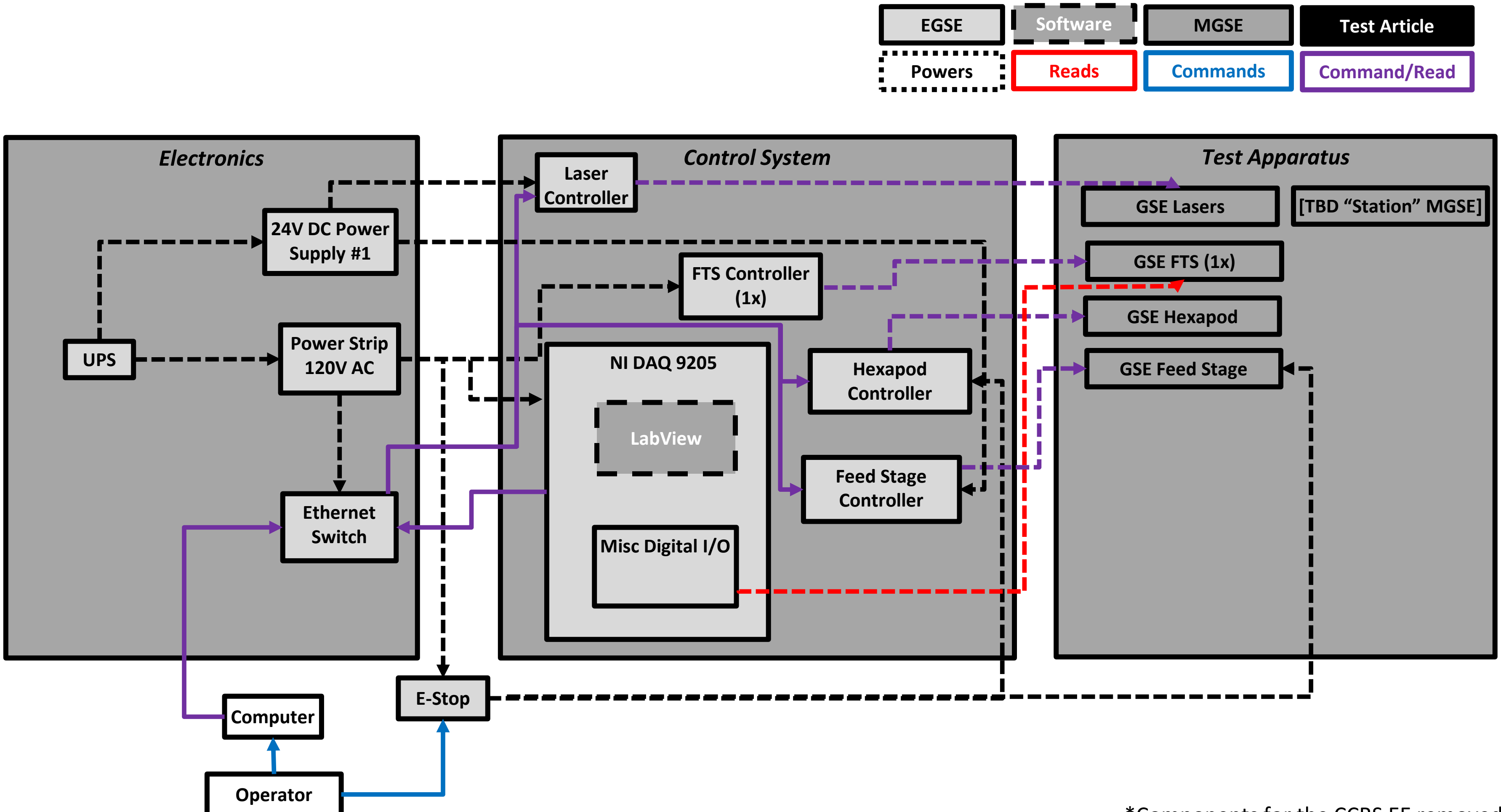
PI H-840.G2A
- 392N Fx load capacity (normal orientation)
- ±50mm lateral
- ±25mm vertical
- ±15° rotation (tip/tilt)
- ±30° rotation (clocking)
- Absolute encoder

Functional Capabilities

- ExoHex for high loads
  - Inner hexapod for precise positioning
  - ExoHex can always be detached
- LabView software for Operator GUI and Control
- FTS recording during test and for force limiting
  - NI DAQ 9205
- Interlock-based E-stop
- Feed Motion Functionality
  - Freespace Move
  - Move to Contact
  - Move to Preload / Move to No Load
- Hexapod coordinate frame changes
- Laser distance measuring capability

Not Fully Checked Out:
- Stiffness Characterization via. laser assembly
- ExoHex assembly proof loading
- Linear actuator proof loading
- Hexapod coord. frame changes in LabView

EDT Functional Block Diagram



*Components for the CCRS EE removed

Main Program Software Design

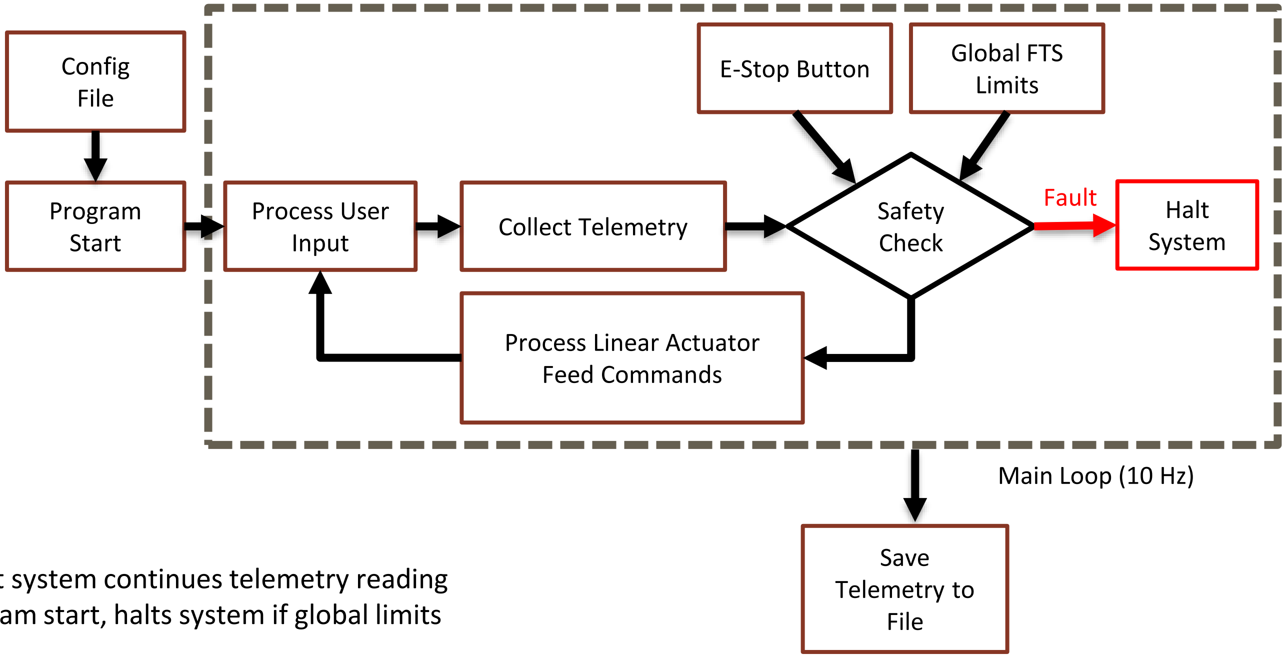
Software Written by **Michael Errico**

Configuration File:
- Linear Actuator Feed Command Parameters (position/force limits, speeds)

User Inputs:
- Linear Actuator Feed Commands
  - Free-space move
  - Move to contact
  - Move to pre-load
  - Move to no-load
- Hexapod Free-space Position move
  - Hexapod speed

Telemetry:
- Hexapod
  - Absolute Position (X, Y, Z)
  - Absolute Rotation (XRot, YRot, ZRot)
- FTS
  - Force (Fx, Fy, Fz)
  - Torque (Tx, Ty, Tz)
- Linear Actuator
  - Absolute Position
  - Speed

E-Stop: Full system halt with physical E-stop but system continues telemetry reading
Global FTS Limits: Hard-coded and set on program start, halts system if global limits are exceeded during any operation



Operator GUI Screen

Testbed Status

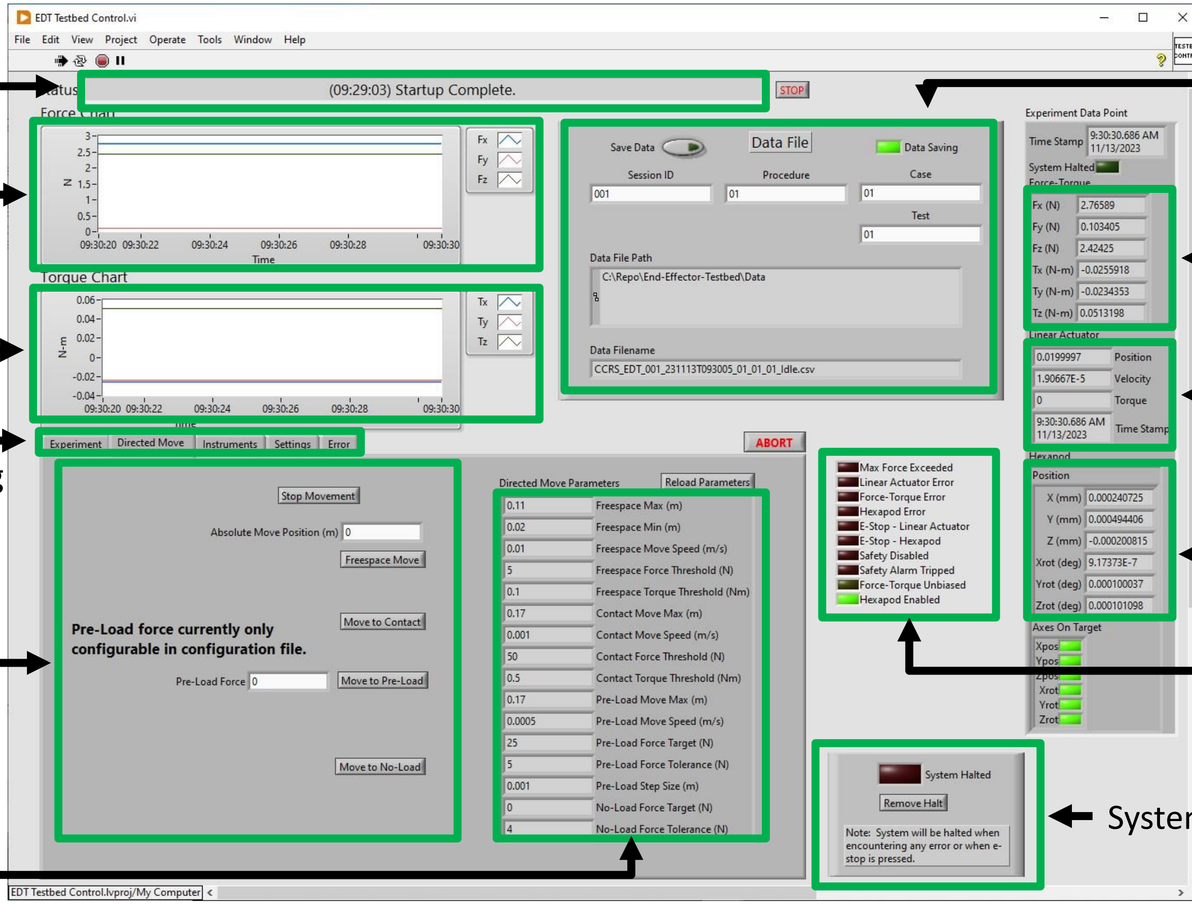
FTS: Fx, Fy, Fz

FTS: Tx, Ty, Tz

Other testbed functionality (moving hexapod, etc)

Linear Actuator Move commands

Linear Actuator Feed Command Parameters



Data Filename Settings

FTS Telemetry

Linear Actuator Telemetry

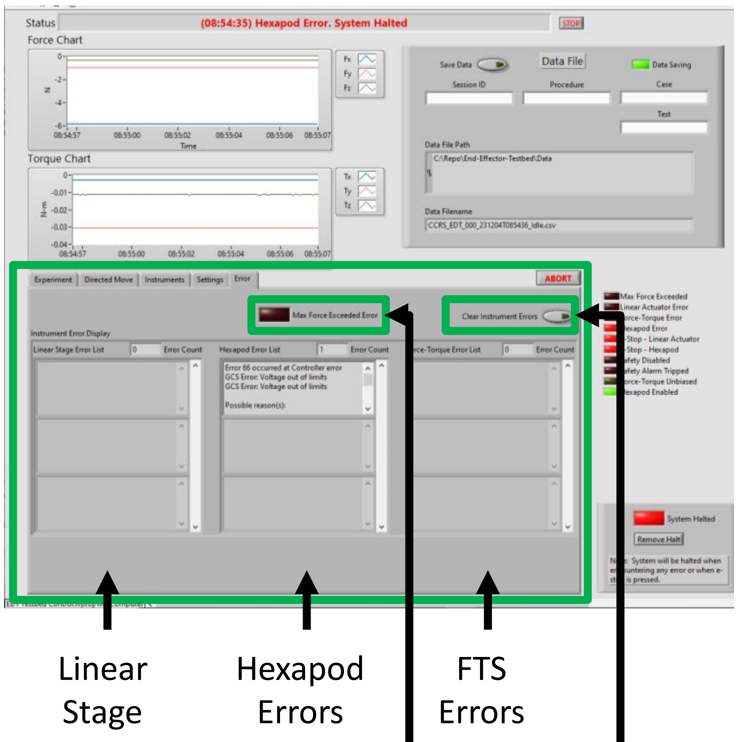
Hexapod Telemetry

Error Status Indicators

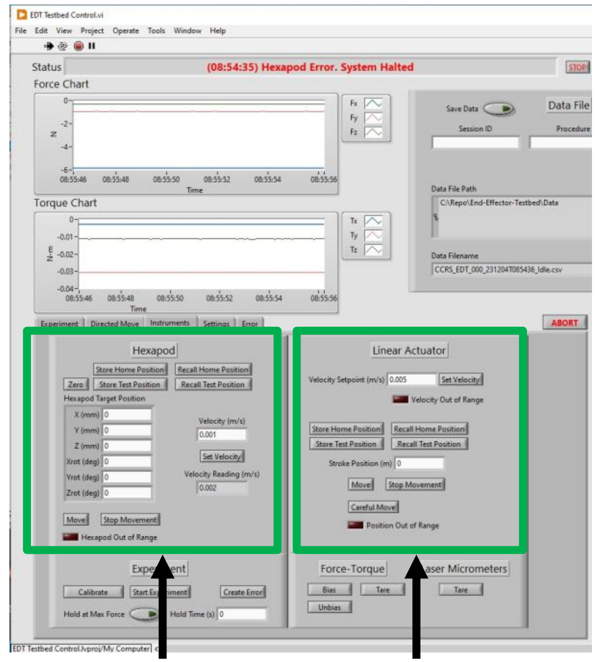
System Halt and Clear

Operator GUI Screen cont.

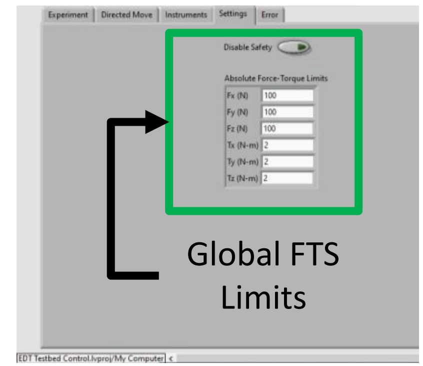
Error Tab



Instrument Tab



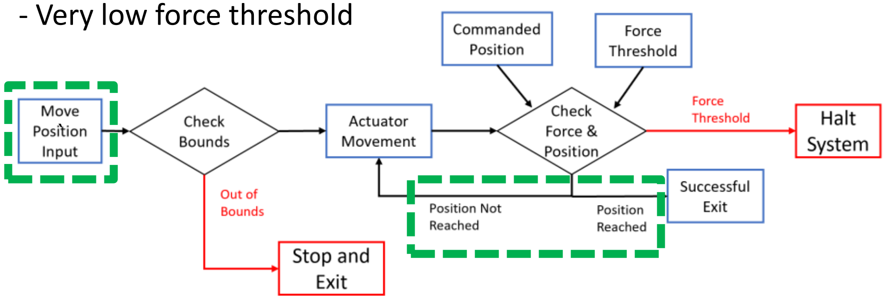
Settings Tab



Linear Actuator Movement Block Diagrams

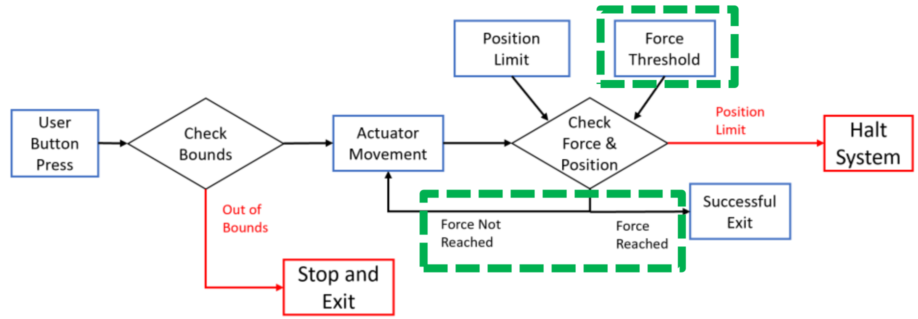
Linear Actuator Free-space Move:

- Higher speed position move with pre-defined limits
- Very low force threshold



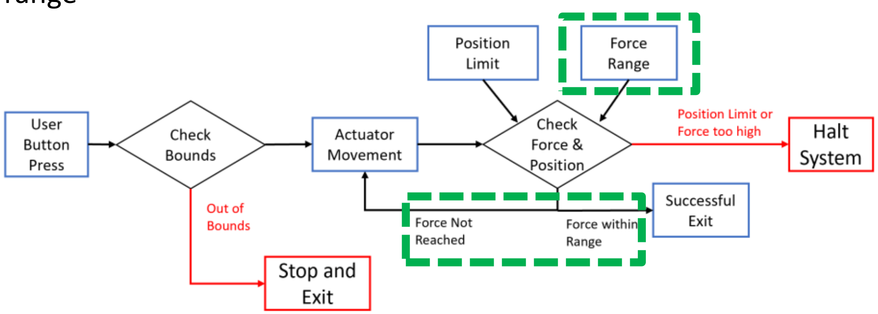
Linear Actuator Move to Contact:

- Slow movement to contact until force threshold is passed



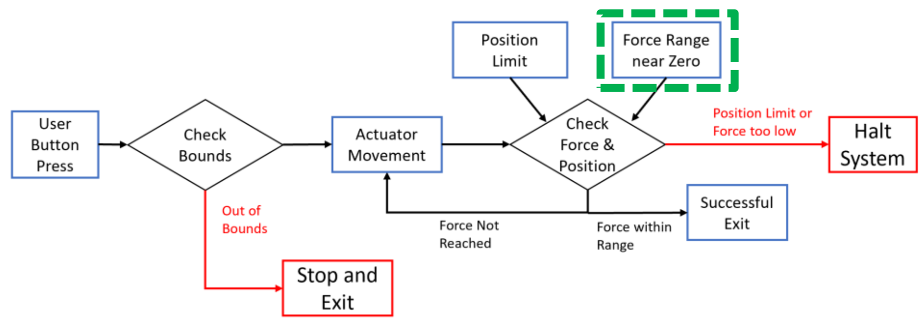
Move to Preload:

- Slow forward movement until target force is within force range



Linear Actuator No-Load Movement:

- Reverse movement to reduce force to near-zero



Checkout Summary

Basic E-Stop Checkout [10] ✅

Objective: Verify E-stop capability to stop motion of both the hexapod and linear actuator during operation, yet still maintain connection and FTS recording. Test how system halts are handled and cleared. Additional testing to see how MicroMove responds to an E-stop being pressed.

Result: E-stop halts motion, maintains connection and continues to record FTS data. System halt can be cleared when E-stop is removed. MicroMove will error when E-stop is pressed, and hexapod can be restarted after E-stop is unpressed.

Basic Hexapod Movement [06] ✅

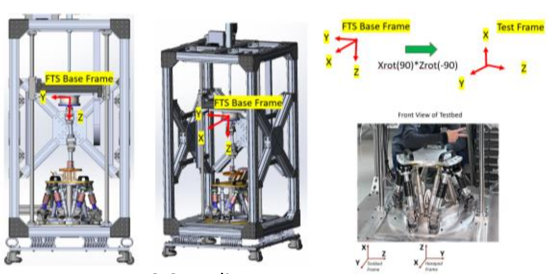
Objective: Move to the maximum 1DoF travel ranges of the hexapod using the testbed coordinate frame. Additionally, test the behavior of the hexapod to stop under global force overload error.

Result: Hexapod moves in accordance with testbed coordinate frame (using vendor provided software) and responds to force overload error in LabView.

Basic FTS Checkout [02] ✅

Objective: Confirm the coordinate system of the FTS for future coordinate transformations.

Result: FTS frame tracks as expected, and coordinate frame change to testbed frame maps correctly

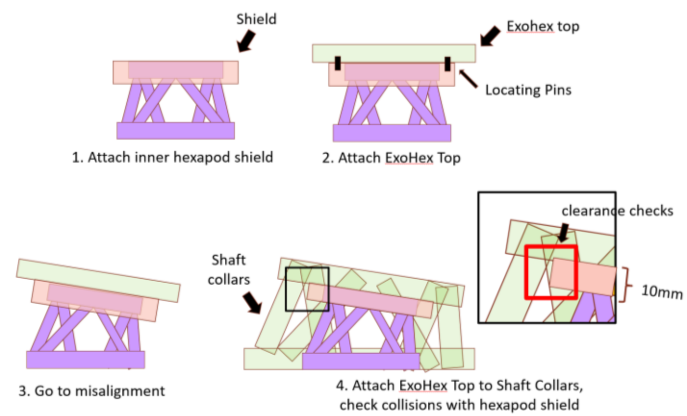


FTS Coordinate Frame

ExoHex Misalignment [09] ✅

Objective: Check for potential collisions between ExoHex strut legs and Inner Hexapod top plate during misalignment, and during potential demate motions (simulated by a hexapod shield).

Result: With the tested subset of misalignments (27 tests), no ExoHex struts were close to collisions. This however is only done for a smaller subset of misalignment and should be performed with actual test misalignments.



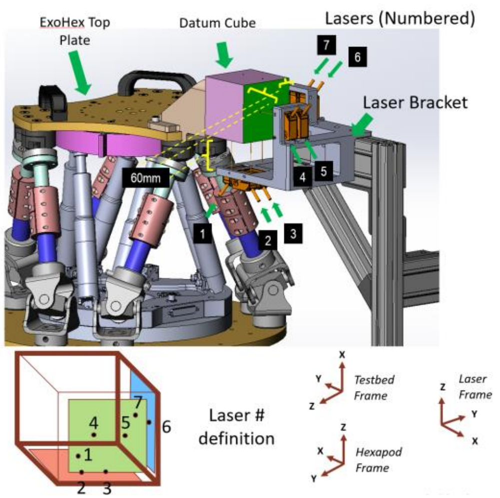
ExoHex Misalignment Test CONOPs

Checkout Summary (continued)

Basic Laser Checkout [03] ✅

Objective: Understand the capabilities of the laser nest assembly when measuring a static cube moving to different positions. Could potentially be used for future stiffness characterization

Result: Lasers measure relative movement accurately, but small errors exist which are likely due to overall misalignment of laser assembly to hexapod.



EDT Laser Nest and Coordinate Frames

Basic Feed Motion Checkout [05] ✅

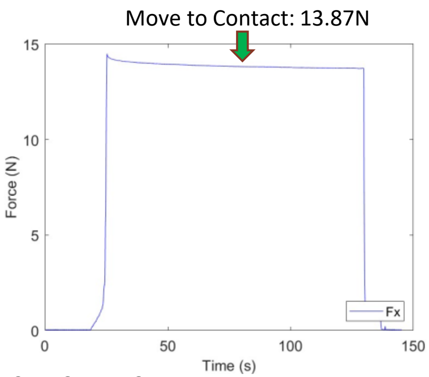
Objective: Test main linear actuator movements (freespace move, move to contact, and move to preload, move to no load).

- *Freespace Move*: higher speed movement to bring the linear actuator to a specific position, with very low force threshold.
- *Move to Contact*: Lower speed movement which moves to a force threshold and stops when it is exceeded (no tolerance).
- *Move to Preload*: Lower speed movement which moves to a specific force value with a given +- tolerance. Meant to reach the desired preload given by the test requirements.
- *Move to No Load*: Reverse "Move to Contact", in which the actuator moves away from contact so that Free-space Moves can be commanded.

Result:

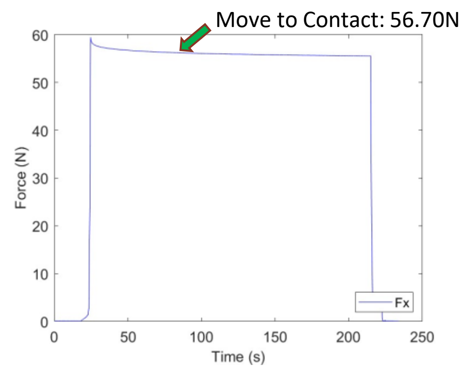
- Linear actuator program demonstrated its intended use for all four different types of movements through applying a specific preload to an aluminum can.
- Linear actuator triggers halt when overall testbed force thresholds are exceeded, preventing users from inputting new commands.

Basic Feed Motion Checkout Results



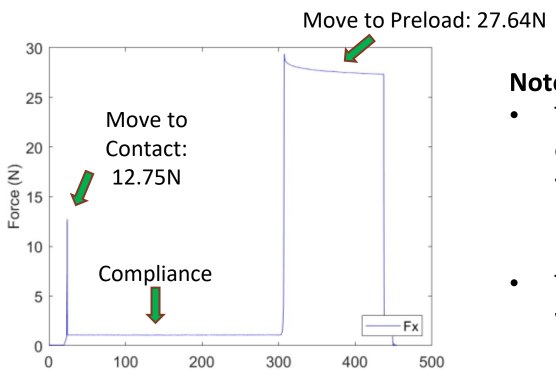
Case 2, Test 2

- Move to Contact to 10N, system stopped at 13.87N



Case 3, Test 3

- Move to Contact to 50N, system stopped at 56.70N

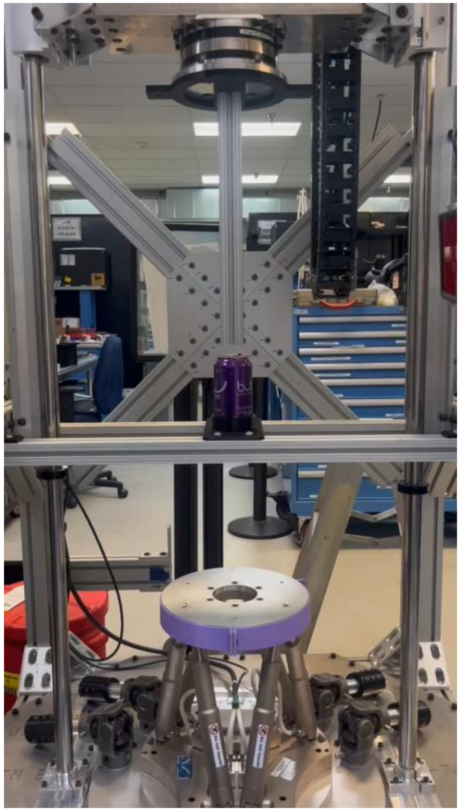


Case 3, Test 2

- Move to Contact to 10N, system stopped at 12.75N
- Move to Preload to 25N, system stopped at 27.64N

Notes:

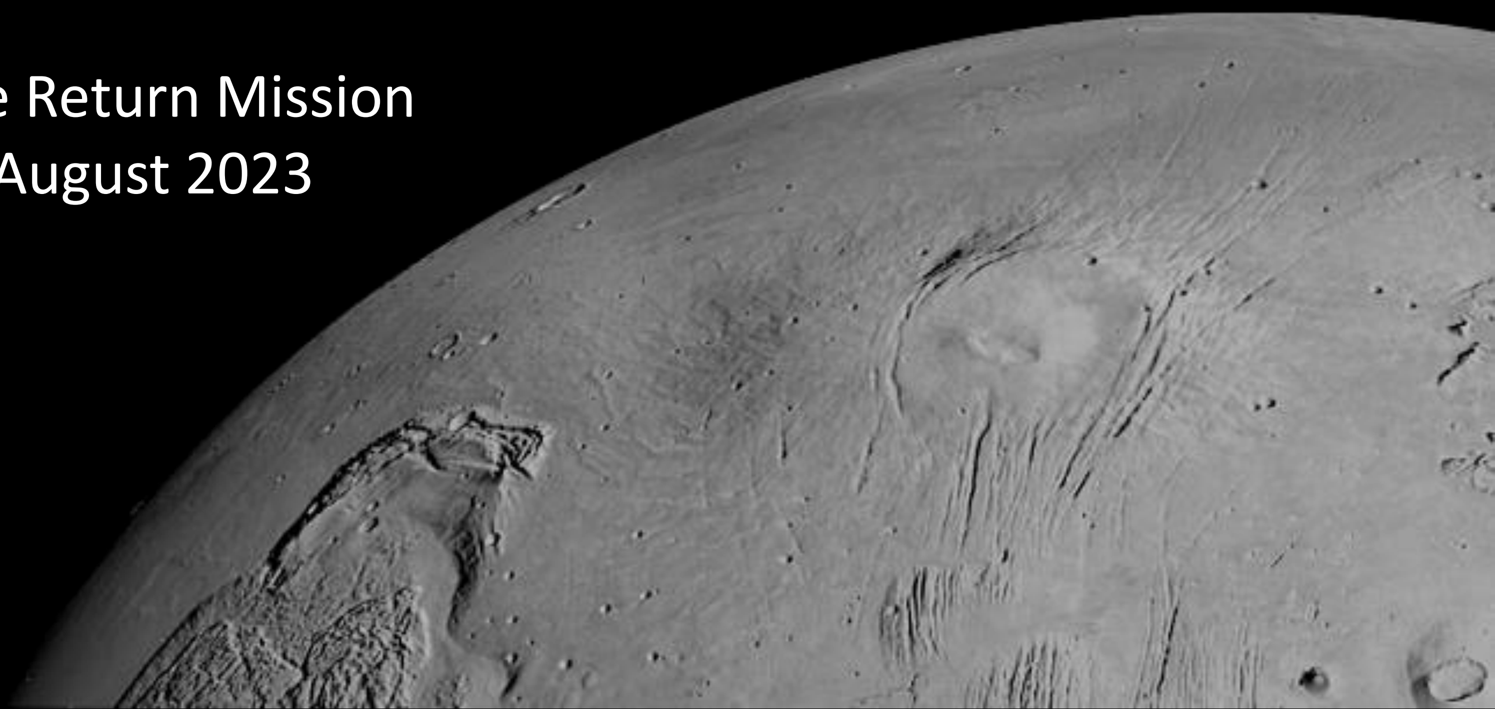
- The system does not halt perfectly as the force build-up occurs quickly; thus, the system does not stop exactly when the force threshold is crossed.
  - Move to contact speed: 1mm/s
  - Move to preload speeds 0.5mm/s
- There is some compliance in the aluminum can such that when the linear actuator stops, the force decreases



Move to Contact Demonstration (video)

Laser Transform Module for End Effector Development (EDT) Testbed

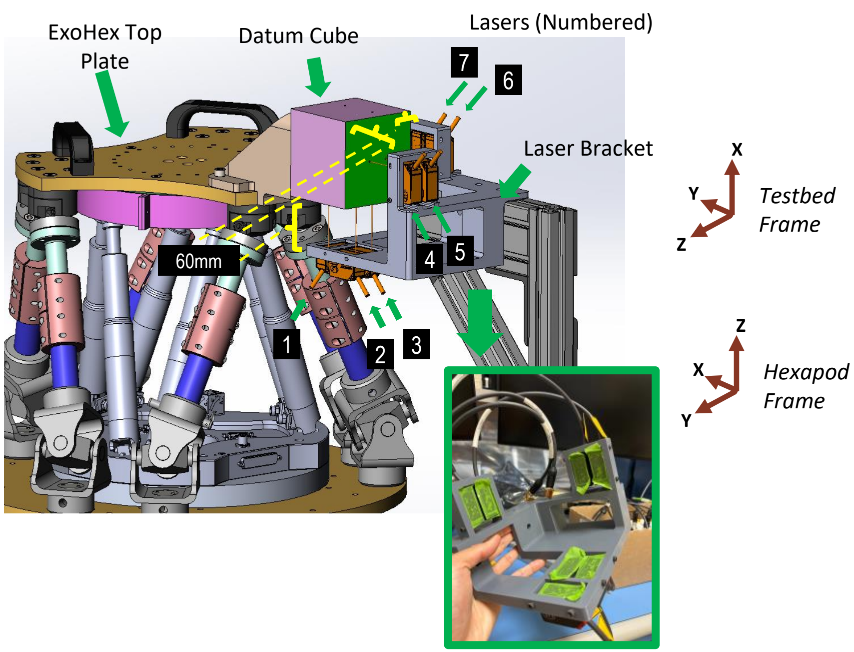
Mars Sample Return Mission
May 2023 – August 2023

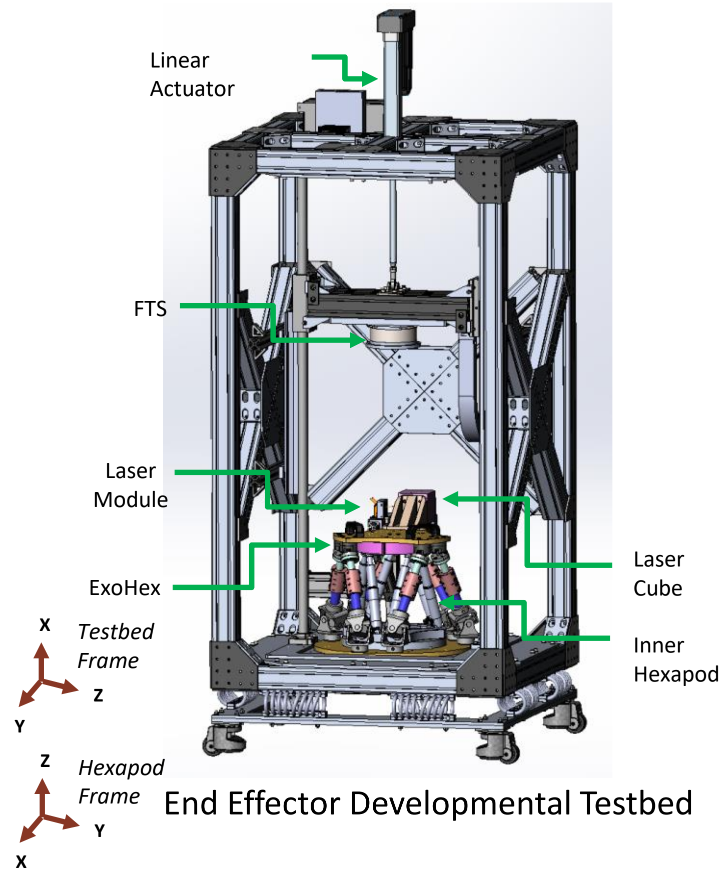


Testbed Background + Objective

Objective: Characterize stiffness of ExoHex top plate under proof load

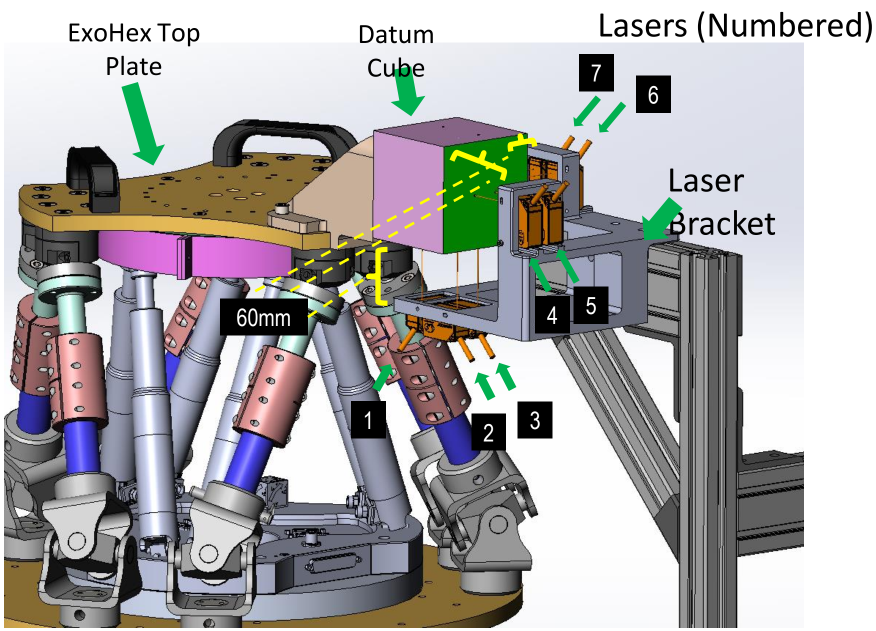
1. Datum (cube) mounted to top plate is considered rigid with hexapod top plate which will deform under external load
2. Lasers points to cube and measure changes in position in free space due to distortions
3. Using 7 lasers, perform transform calculation to define full homogenous transform of cube

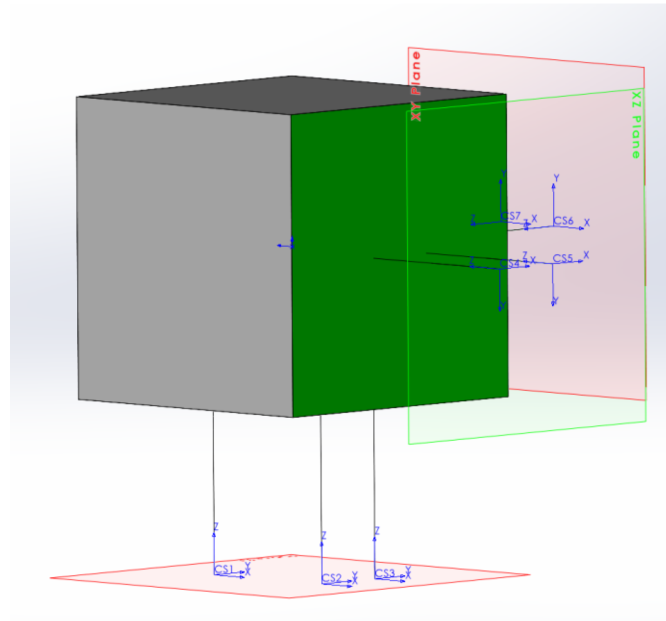


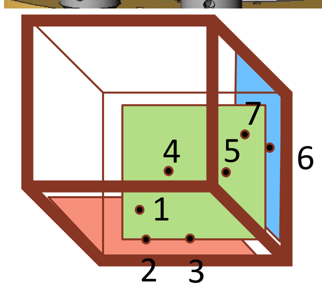


End Effector Developmental Testbed

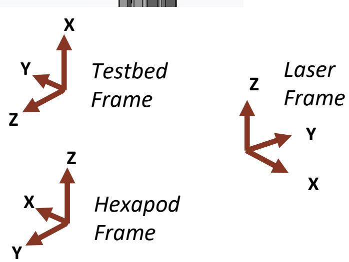
Coordinate Frame Definition







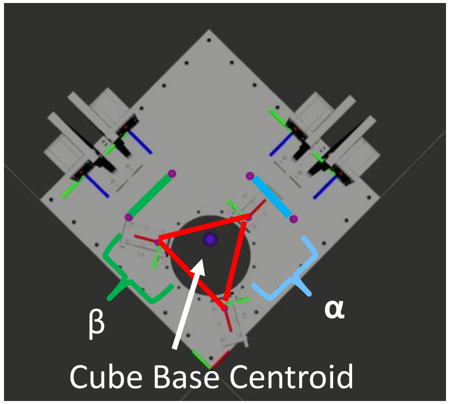
Laser # definitio n



- Separate CAD model to simulate rotations / translations
- Lasers represented as (very small diameter) extrusions up to surface for ground truth generation from nominal laser positions → SolidWorks sensors
- Laser Frame → Testbed Frame is Ry(-90)*Rx(90)

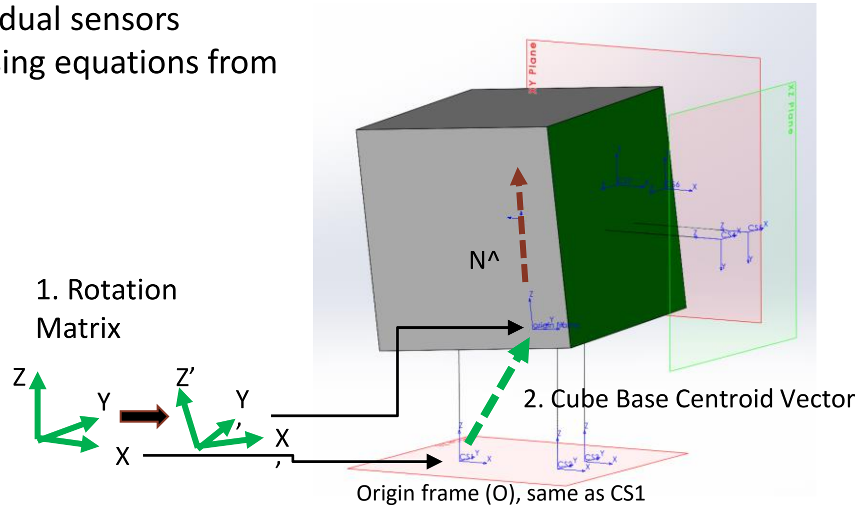
Implementation Approach

- Define frames of our 7 lasers such that we can perform the IK (inverse kinematics) to define the full transform of the cube
  - Coordinate system is defined on bottom of 3 planes, where n^ is defined
  - α and β are constants defined to surface of where lasers hit cube face
    - In our case alpha and beta are cube side lengths (4in / 2)
    - Reference

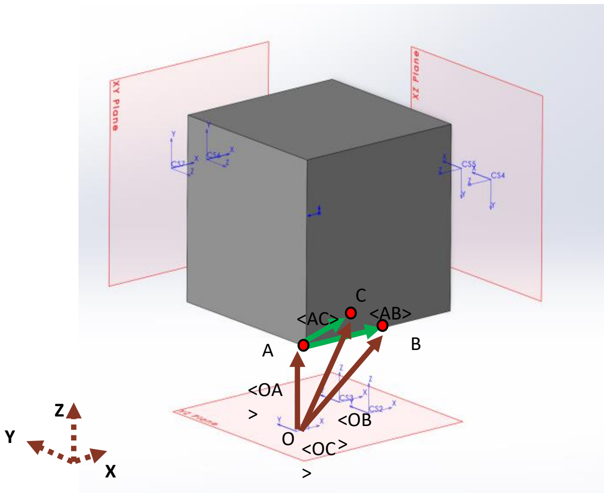


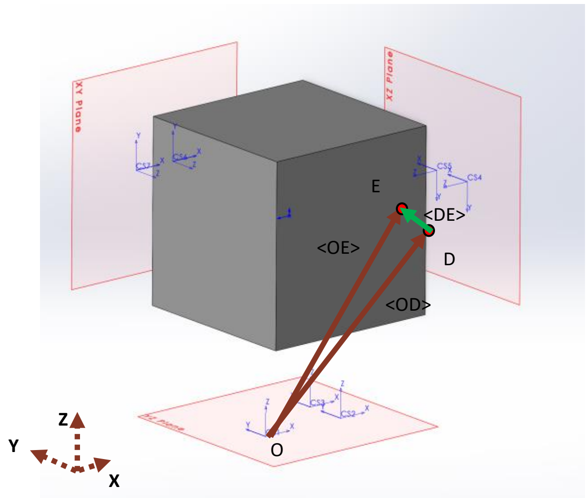
Setup

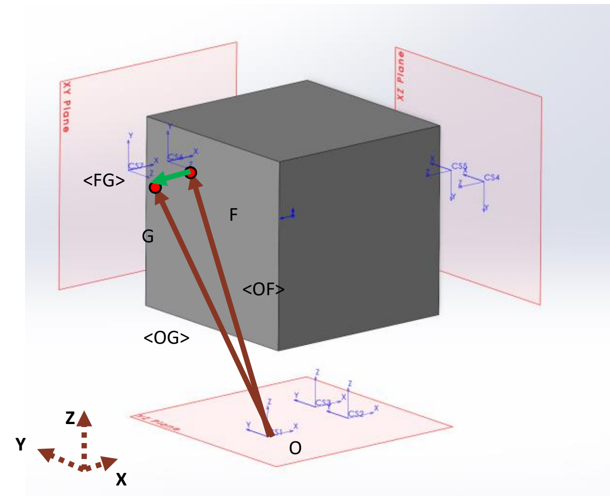
1) Choose a world coordinate frame (origin frame) for lasers
   1) Origin frame set at first laser frame (CS1)
2) Create transformations to each of the frames on each of the lasers
   1) Z-axis always the pointing towards the cube
   2) H_O1, H_O2 (ETC)
   3) Pure Z-translation in that coordinate frame from each individual sensors
3) Solve for transform from laser readings in laser frame on cube using equations from paper for
   1) Rotation Matrix
   2) Cube Base Centroid Vector
4) Perform transformation from world frame to testbed frame



Main Vector Definition







Origin Frame (coincident with CS1)

Defined in Origin Frame
Vector of Interest: <AC> = <OC> - <OA>, <AB> = <OB> - <OA>

Where <OC> = H03*P3, <OB> = H02 * P2, <OA> = H01*P1 where P1, P2, P3 are the magnitude of the laser (in Z axis)

$$\hat{n} = \frac{\vec{AB} \times \vec{AC}}{\|\vec{AB} \times \vec{AC}\|} \tag{3}$$

```
Eigen::Vector3d ab = b - a;
Eigen::Vector3d ac = c - a;

Eigen::Vector3d n = ab.cross(ac).normalized();
```

Defined in Origin Frame
Vector of Interest: <DE> = <OE> - <OD>

Where <OE> = H05*P5, <OD> = H04*P4 and P4, P5 are the magnitude of the laser (in Z axis)

$$\frac{\hat{n} \times \vec{DE}}{\|\hat{n} \times \vec{DE}\|} \cdot \vec{\mathcal{O}} = \frac{\hat{n} \times \vec{DE}}{\|\hat{n} \times \vec{DE}\|} \cdot D - \alpha \tag{9}$$

```
Eigen::Vector3d de         = e - d;
Eigen::Vector3d n_cross_de = n.cross(de).normalized();
```

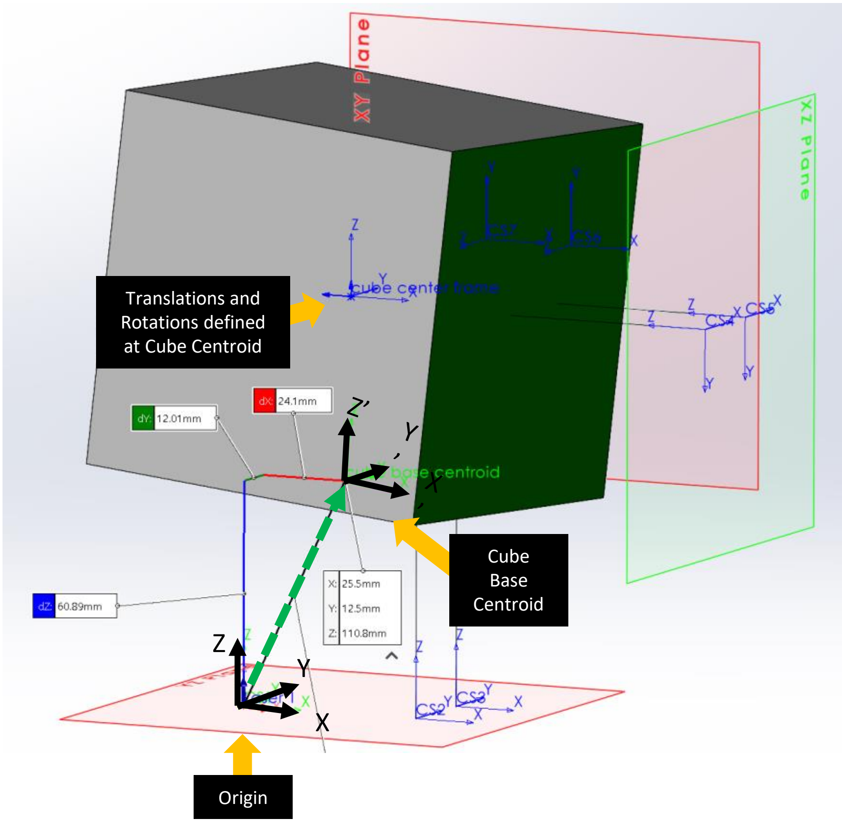
Defined in Origin Frame
Vector of Interest: <FG> = <OG> - <OF>

Where <OG> = H07*P7, <OF> = H06*P6 and P6, P7 are the magnitude of the laser (in Z axis)

$$\frac{\hat{n} \times \vec{FG}}{\|\hat{n} \times \vec{FG}\|} \cdot \vec{\mathcal{O}} = \frac{\hat{n} \times \vec{FG}}{\|\hat{n} \times \vec{FG}\|} \cdot F - \beta \tag{12}$$

```
Eigen::Vector3d fg         = g - f;
Eigen::Vector3d n_cross_fg = n.cross(fg).normalized();
```

Ground Truth Test Approach



Perform 3 types of ground truth tests

1. Rotations
2. Translations
3. Combined

- Using CAD, rotate the cube in a specific order
  - Laser output from CAD informs inverse kinematics and creates ground truths
  - Check rotations by outputting Euler Angles in (ZYX) format
- For tests cases, Δ translations and rotations defined at **CUBE centroid**
  - Cube Base Centroid (centroid of 3 lasers on cube face) may have some "parasitic" translations due to rotation about a different pivot

Inputs

1. Laser values from CAD
2. Rotations (Euler Angles) for comparison
3. Position vector between Cube Base Centroid and Origin Frame for comparison

Outputs

1. Rotations (Euler Angles)
2. Position vector between Cube Base Centroid and Origin Frame

Compare

1. Ground Truth Rotation vs. Inverse Kinematic Rotation solution
2. Ground Truth Position vector vs. Inverse Kinematic Position solution

Rotation (Orientation) Ground Truth Tests

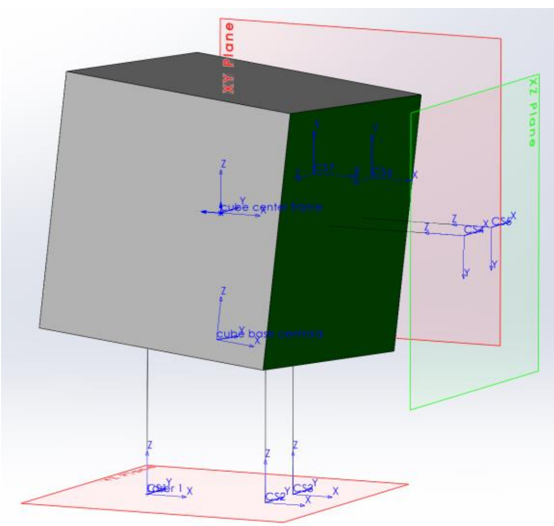
or in rotation matrix form:

$$R = \begin{bmatrix} \left(\frac{\hat{n}\times\vec{DE}}{\|\hat{n}\times\vec{DE}\|}\right)^T \\ \left(\frac{\hat{n}\times\vec{FG}}{\|\hat{n}\times\vec{FG}\|}\right)^T \\ -\hat{n}^T \end{bmatrix} \tag{17}$$

```
Eigen::Matrix3d R_ltm;
R_ltm << -n_cross_de(0), -n_cross_fg(0), n(0), -n_cross_de(1), -n_cross_fg(1)
    n(1), -n_cross_de(2), -n_cross_fg(2), n(2);
```

Ground Truth Test Cases

Test #	Δ X (mm)	Δ Y (mm)	Δ Z (mm)	Zrot (deg)	Yrot (deg)	Xrot (deg)	IK Rot Match?	IK Pos. Match?
1	0	0	0	0	0	0		
2	0	0	0	1	0	0		
3	0	0	0	0	1	0		
4	0	0	0	0	0	1		
5	0	0	0	1	2	0		
6	0	0	0	1	2	3		
7	0	0	0	-2	-5	3		



Rotation Test Cases

- Test individual rotations, as well as rotations in sequence, as well as with either +/- signage
- Correctly tracked Euler Angles (for both positive and negative), as well as sequences of Euler Angles
    - Rounded (to the first decimal place) → might be due to sig-figs on laser output from CAD
- When rotating about centroid of cube, the cube base centroid also translates, which was captured in the positional inverse kinematics

Translation Ground Truth Tests

$$\begin{bmatrix} \hat{n}^T \\ \left(\frac{\hat{n}\times\vec{DE}}{\|\hat{n}\times\vec{DE}\|}\right)^T \\ \left(\frac{\hat{n}\times\vec{FG}}{\|\hat{n}\times\vec{FG}\|}\right)^T \end{bmatrix}_{(3\times3)} \{\vec{O}\}_{(3\times1)} = \begin{Bmatrix} \hat{n}\cdot A \\ \frac{\hat{n}\times\vec{DE}}{\|\hat{n}\times\vec{DE}\|}\cdot D-\alpha \\ \frac{\hat{n}\times\vec{FG}}{\|\hat{n}\times\vec{FG}\|}\cdot F-\beta \end{Bmatrix}_{(3\times1)} \qquad (13)$$

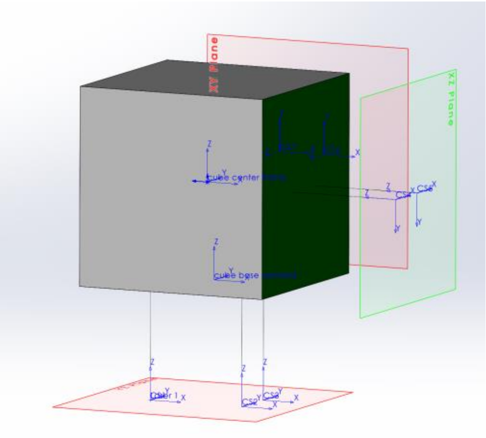
```
Eigen::Matrix3d A;
A << n(0), n(1), n(2), n_cross_de(0), n_cross_de(1), n_cross_de(2),
    n_cross_fg(0), n_cross_fg(1), n_cross_fg(2);

Eigen::Vector3d rhs;
rhs << n.transpose() * a,
    n_cross_de.transpose() * d + laser_target_tool_axis_offset_,
    n_cross_fg.transpose() * f + laser_target_tool_axis_offset_;

Eigen::FullPivLU<Eigen::Matrix3d> lu(A);
Eigen::Vector3d                  origin = lu.solve(rhs);
```

Ground Truth Test Cases

Test #	Δ X (mm)	Δ Y (mm)	Δ Z (mm)	Zrot (deg)	Yrot (deg)	Xrot (deg)	IK Rot. Match ?	IK Pos. Match?
1	0	0	0	0	0	0		
2	+2.5	0	0	0	0	0		
3	-2.5	0	0	0	0	0		
4	0	+2.5	0	0	0	0		
5	0	0	+2.5	0	0	0		
6	+2.5	+1	0	0	0	0		
7	+2.5	0	+1	0	0	0		
8	0	+2.5	+1	0	0	0		
9	+2.5	+1	+5	0	0	0		



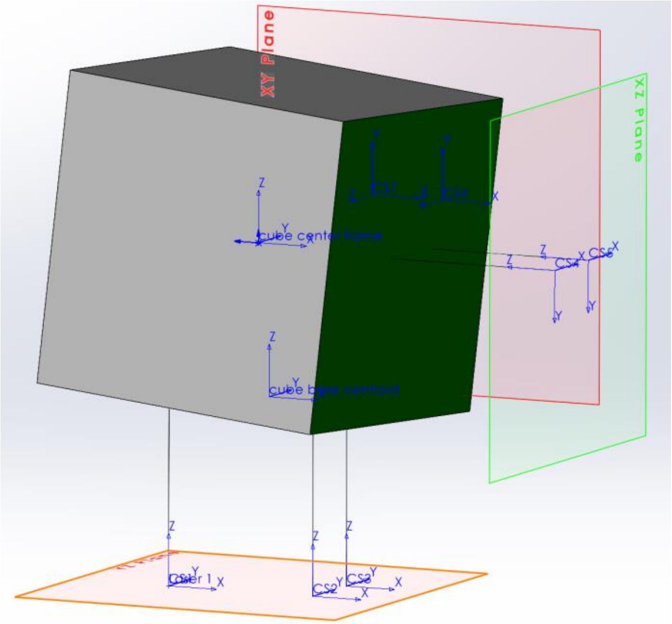
Translation Test Cases

- Test individual translations, as well as multiple translations, as well as with either +/- signage
- Tests worked in accordance with translations (and had no rotations)
- Rounded (to the first decimal place) → might be due to sig-figs on laser output from CAD

Combined Ground Truth Tests

Ground Truth Test Cases

Test #	Δ X (mm)	Δ Y (mm)	Δ Z (mm)	Zrot (deg)	Yrot (deg)	Xrot (deg)	IK Rot. Match ?	IK Pos. Match?
1	5	7.5	10	-2	-5	3		
2	-5	-7.5	-10	-2	-5	3		
3	-5	-7.5	-10	2	5	-3		
4	1	3	5	2	0	0		
5	5	3	1	0	3	0		
6	3	2	1	-5	0	0		



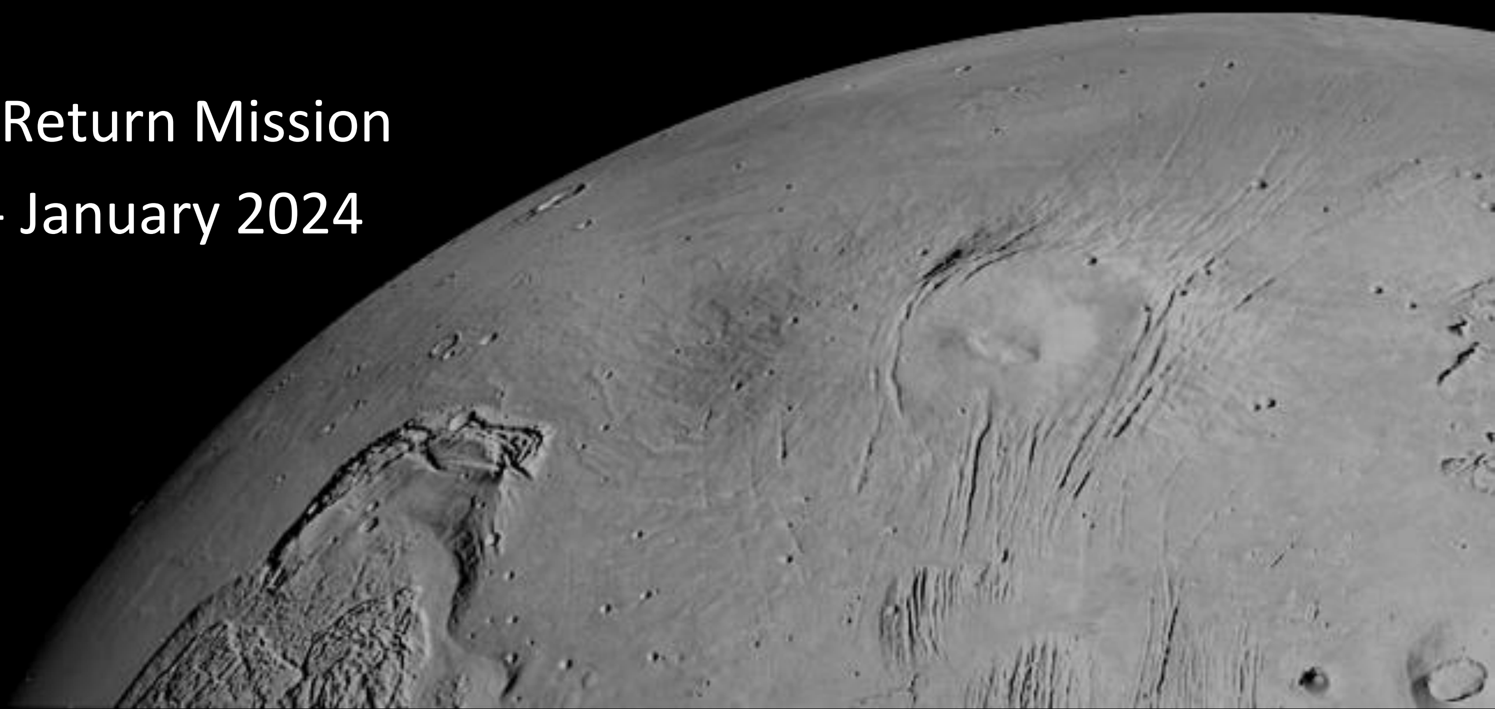
Combined Test Cases

- Test both translations and rotations at centroid of cube
  - Every case works for the inverse kinematics!
- Rounded (to the first decimal place) → might be due to sig-figs on laser output from CAD

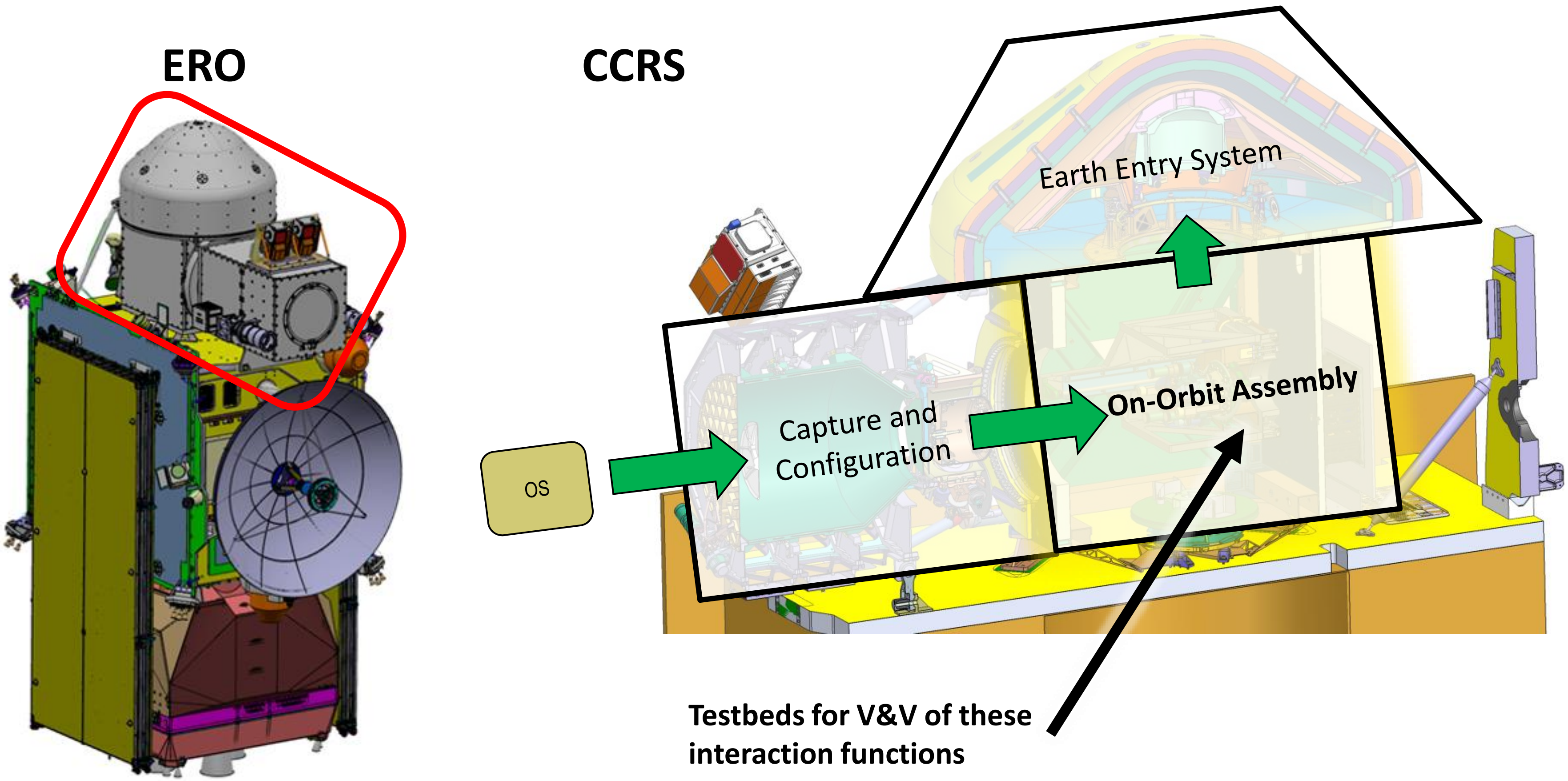
Software Development Summary of Work

Mars Sample Return Mission

August 2023 - January 2024



Capture, Containment, and Return System (CCRS) in ERO Context



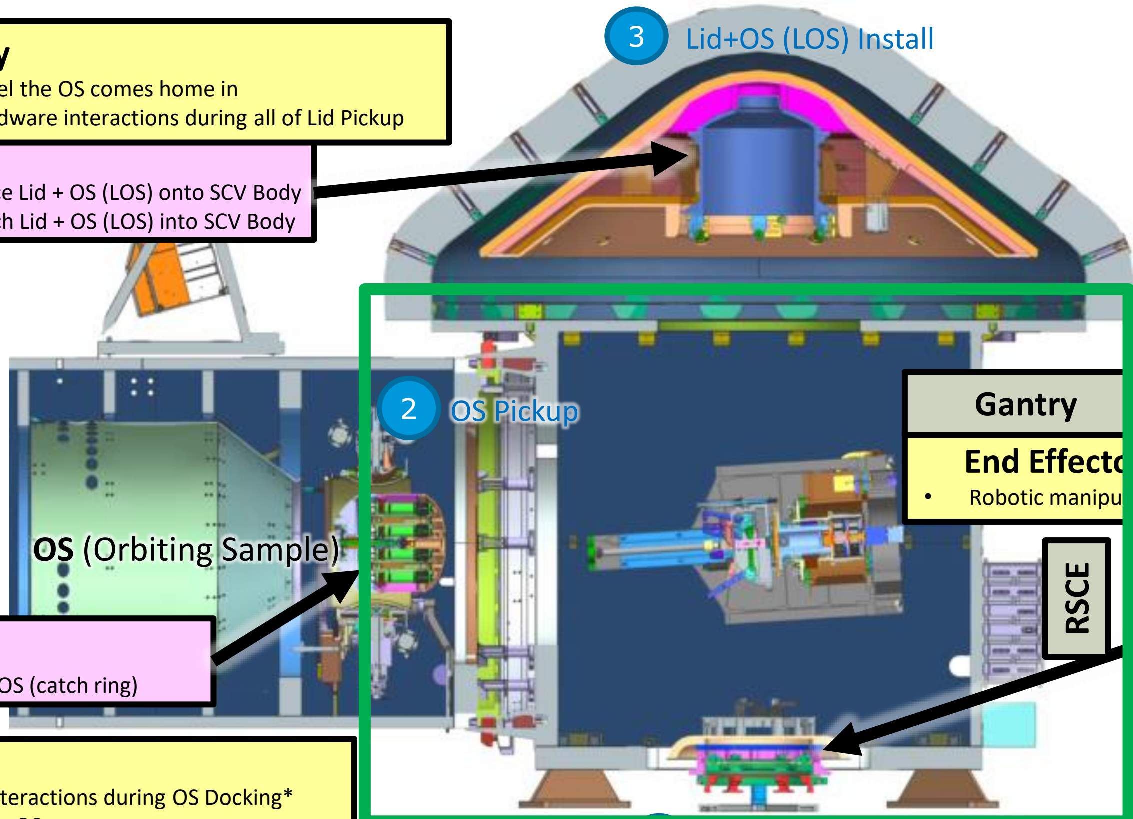
CCRS Overview for Testbeds context

PIE Station Interaction Functions

PIE Hardware

RTAS Hardware

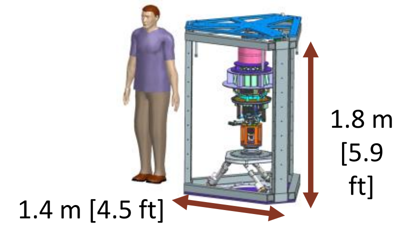
Acronyms:
- Robot Interface to Payload (RIP)
- Secondary Containment Vessel (SCV)
- Aerothermal Closeout (ATC)
- SCV-OS Latch, Align, and Restrain (SOLAR)
- Integrated Lid Assembly (ILA)
- Lid Release Mechanism (LRM)
- Orbiting Sample (OS)
- End Effector (EE)

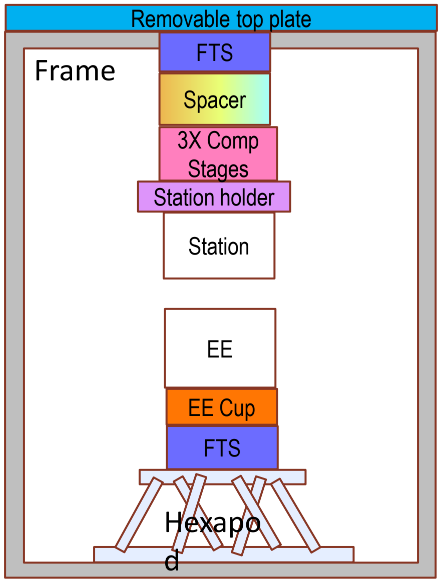


PIT Overview

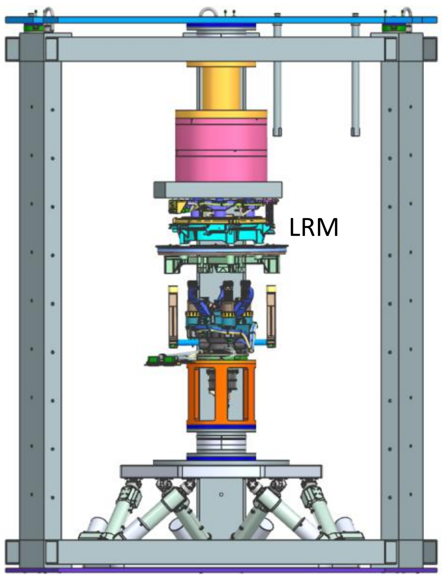
PIT: Pickup and Installation Testbed

Goal: Test and measure station and tool interactions between CCRS end effector and various interfaces given a specific misalignment in **TVAC** environment

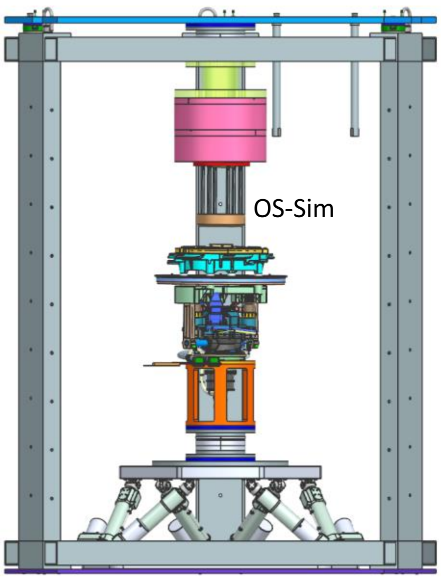




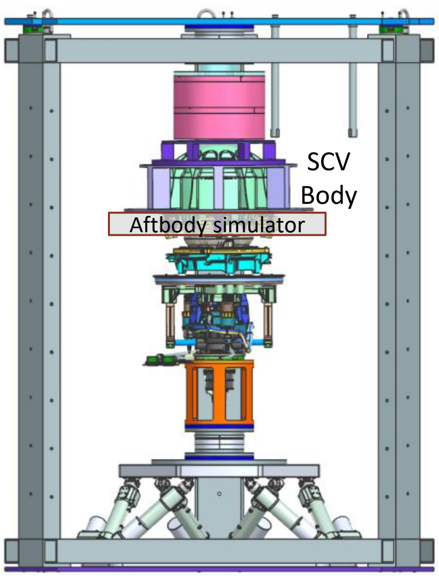
Generalized Configuration



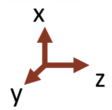
Lid Pickup



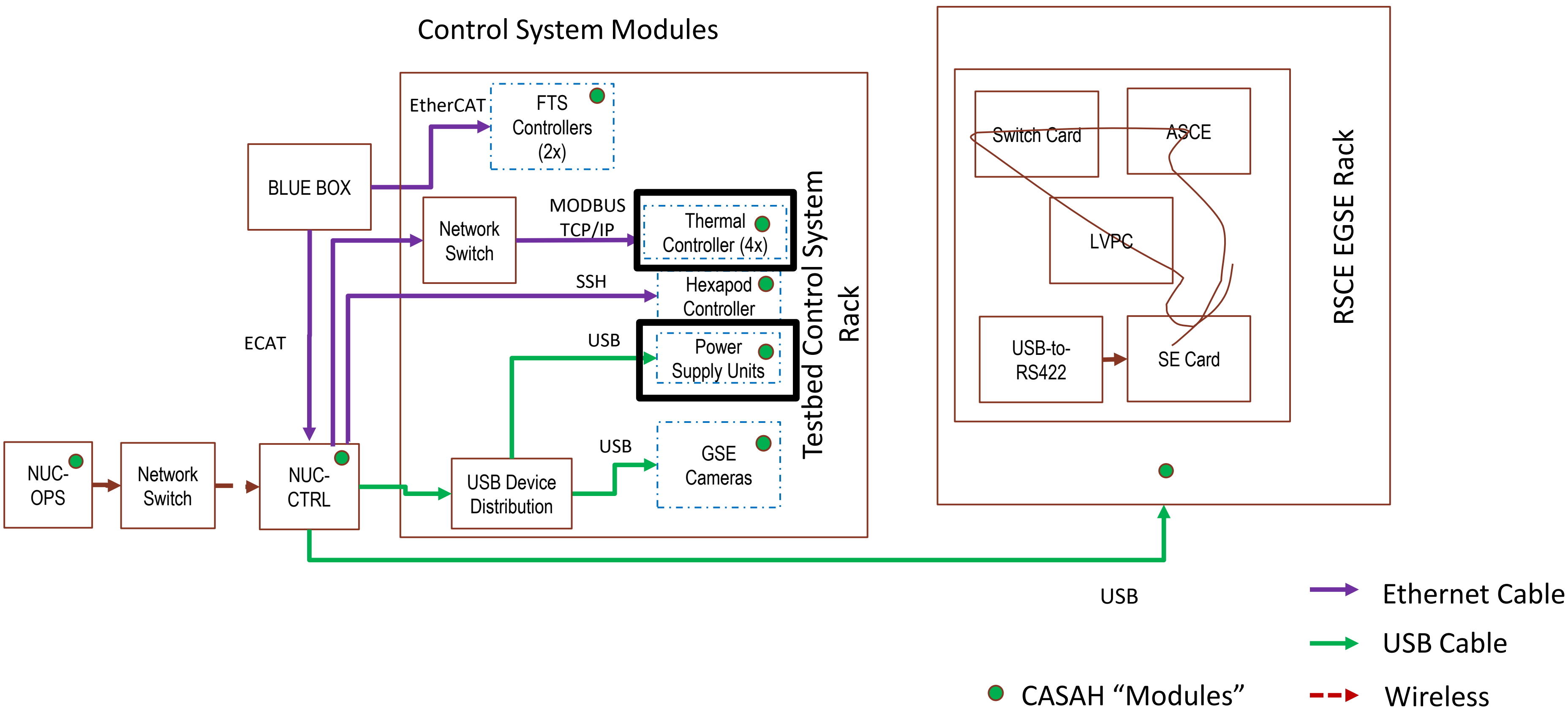
OS Pickup



LOS Install

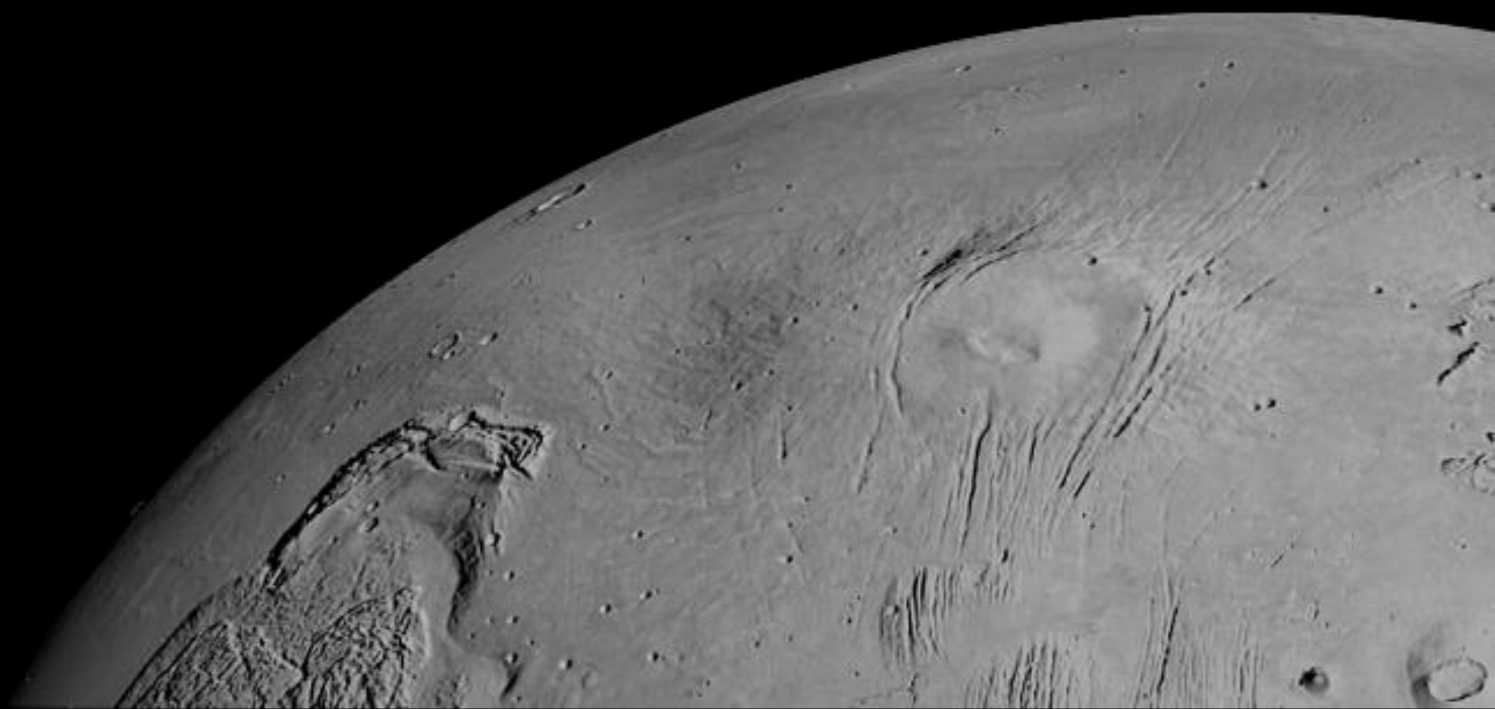


PIT Software Functional Block Diagram



PSU_MGR:

ROS2 Power Supply Package

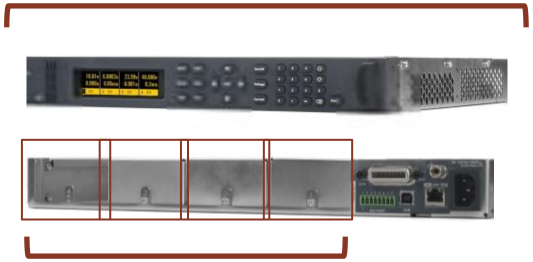


Keysight Power Supply

- Objective: Create a CASAH Module that can manage multiple Keysight Power Supplies (PSU) with the core functionality of
  1. Initialize PSUs
  2. Query and Publish Telemetry
  3. Allow operators / other modules to
     1. Clear errors
     2. Change channel outputs (ON/OFF)
- Intended use was to turn ON/OFF motors for RSCE Rack Sequencing
- Previous scope included error management -> later moved to FP_MGR

Keysight Series N6700

Mainframe (1x)



Channels (4x)

Nomenclature

- **Module** (CASAH Module): represented by **psu_mgr**, manages multiple instances of a Power Supply Class
- **Mainframe:** Refers to an instance of the Power Supply Class and represents a single mainframe which houses multiple channels
- **Channel:** One of the four smaller power supplies present in a mainframe which have their own voltage, current, power requirements
  - *Assumes each frame has four channels – some frames have two channels combined - have not tested this behavior*

PSU Output Channels

Main Functionality of UPS

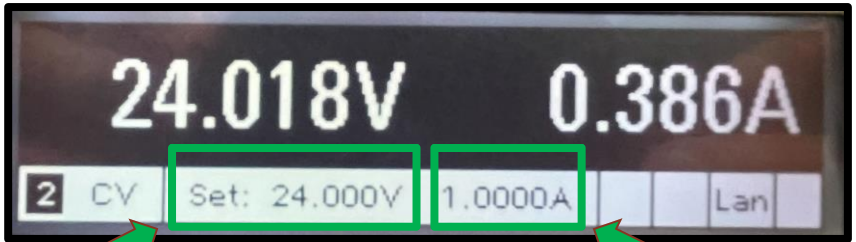
Hardware Pre-existing Functionality:

- Over Protection settings to Protect Hardware
    - Channel will stop outputting during Overvoltage (OV) or Overcurrent (OC) event

Needed Software Functionality:

- Turn on/off Power Supply Channels
- Read/Set Voltage Set Points, Current Limit
- Read/Set Overvoltage Set Points
- Read/Set Overcurrent Set Points
- Read Overvoltage Errors
- Read Overcurrent Errors
- Read Voltage Output
- Read Current Output

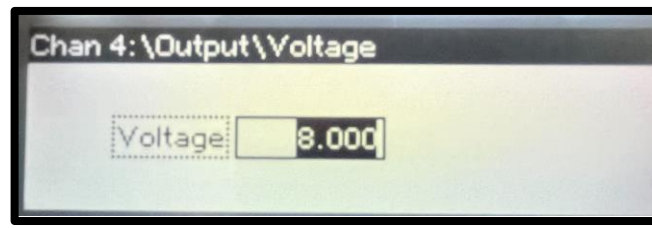
PSU can be separately programmed via. screen



Voltage Set Point

PSU Output Screen

Current Limit



PSU Voltage Settings

PSU Overvoltage Protection Settings

Keysight Power Supply Wrapper

KeysightPowerSupply.py

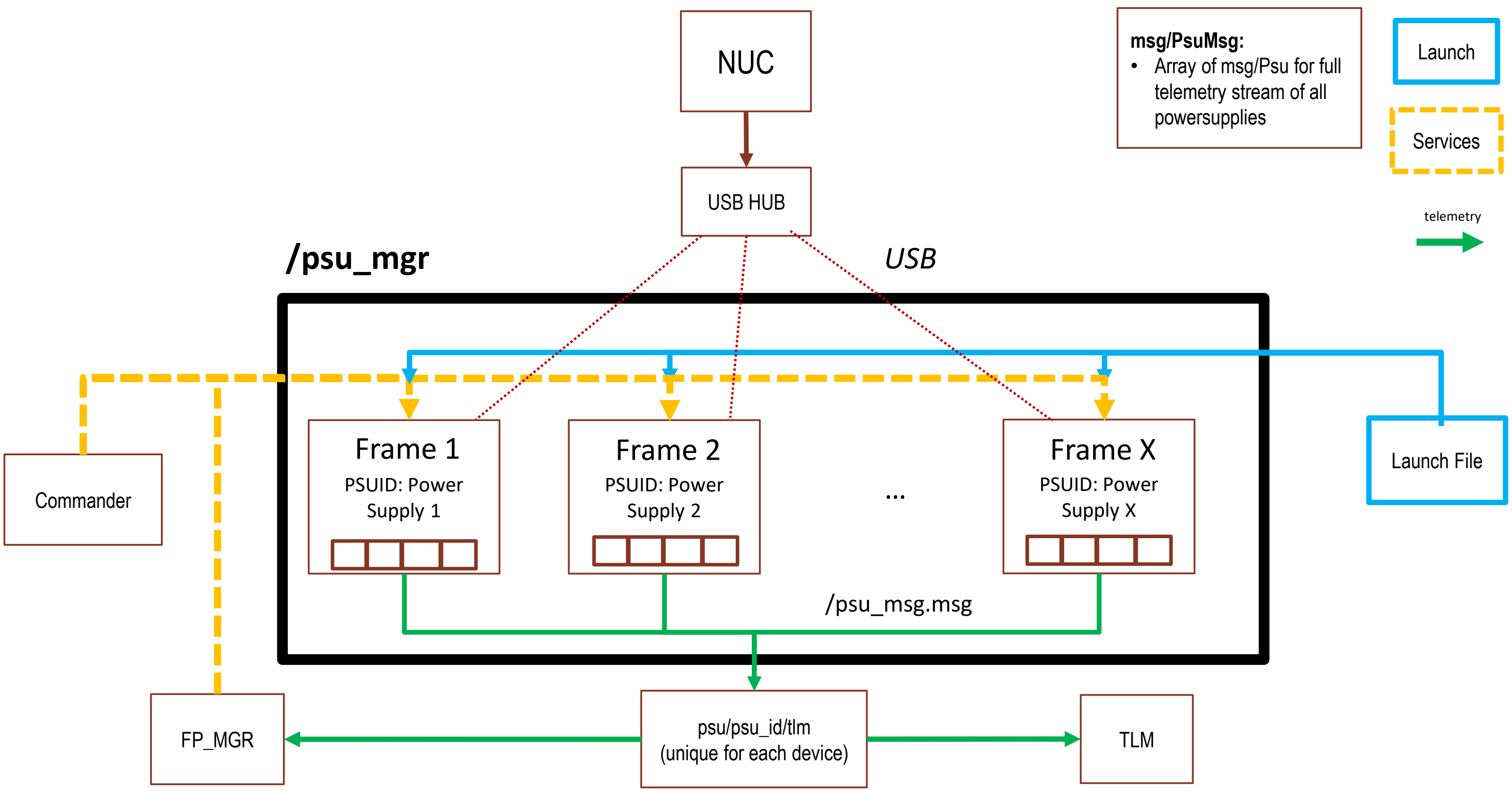
Instance Data:
- ID
- usb_addr
- output_states []
- currents []
- current_limits []
- voltages []
- voltage_setpoints []
- error []
- error_enum[]
- overvoltage_setpoints []
- overcurrent_enables []
- fault

Functions:
- Set/Read Output States
- Set/Read Current Limit Set Points
- Set/Read Voltage Set Points
- Set/Read Overvoltage Set Points
- Set/Read Overcurrent Enable State
- Read Error
- Read Error Enum
- Read Current Measurement
- Read Voltage Measurement
- Read Fault

- Fault defined as any error on any channel for a single frame

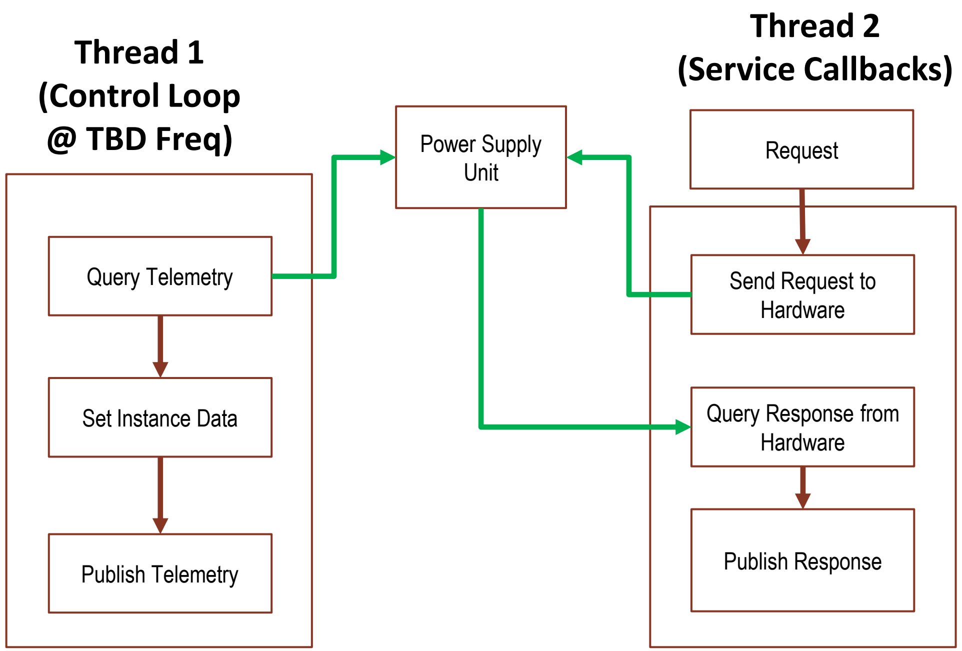
Communication & API:
- Uses Virtual instrument Software Architecture (VISA) API
  - Can communicate using TCP / USB
    - VISA Layer gives specific “VISA” address to hardware
    - NI-VISA Library
- Commands are sent through Standard Commands for Programmable Instruments (SCPI)
  - EX: OUTP ON, (@2)
    - Set channel 2 OUTPUT to ON
- Used **PyVISA** library -> module written in Python
  - Library supports multithreading
- Index of the array corresponds to channel of mainframe

PSU_MGR Setup



Architecture #1: Asynchronous Control, Multithread

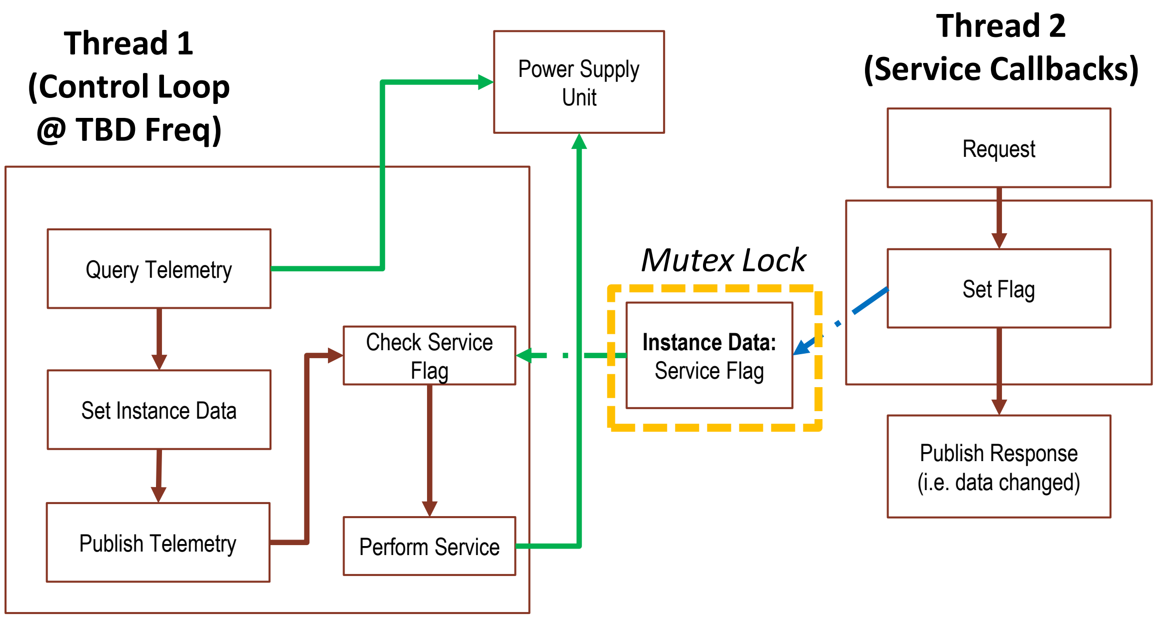
Assumed Requirements:
1. Query / Publish telemetry at specific frequency every time
2. Timing of service call timing is not strict; can be performed whenever possible



Notes:
- Asynchronous control scheme
- Requires multiple communication interfaces with different threads
  - i.e. multiple TCP Clients to the same server (hardware)
- **Keysight Power Supply** does not support multiple interfaces
  - Tested multiple clients and multithreading
    - Did not work (I/O errors)
  - Vendor claims that the PSU cannot query multiple requests at the same time

Architecture #2: Synchronous Control, Single Loop

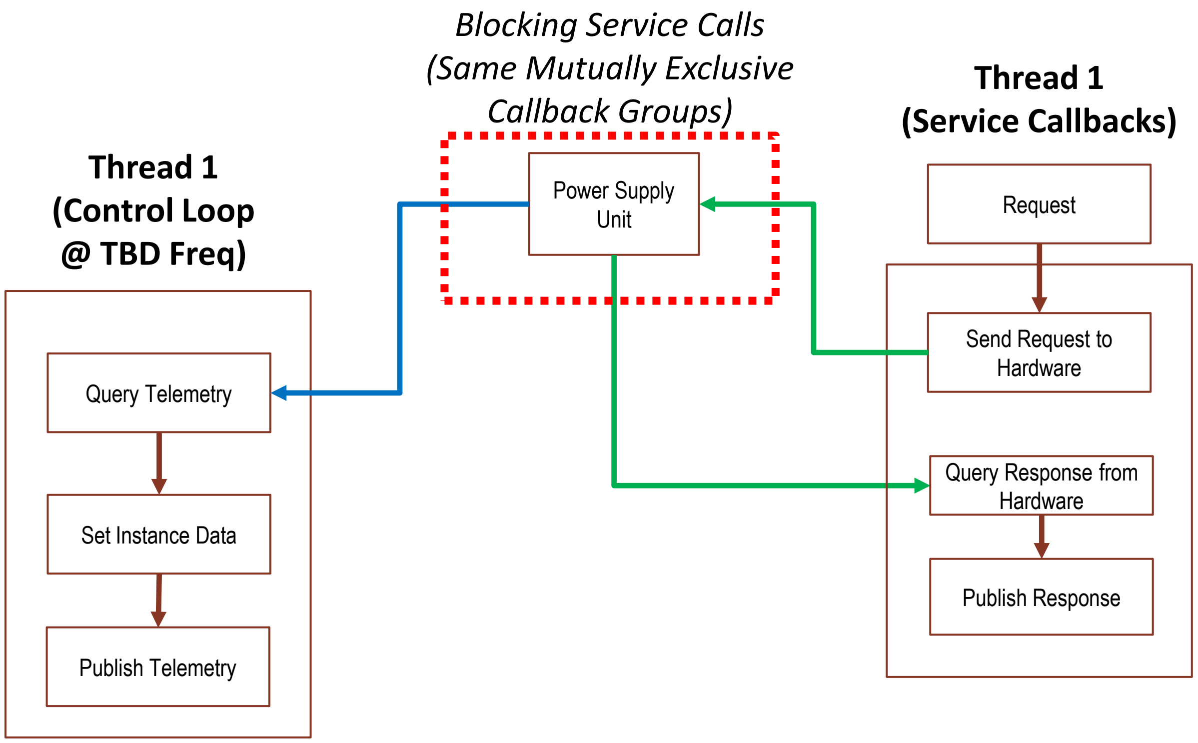
Assumed Requirements:
1. Query / Publish telemetry at specific frequency every time
2. Timing of service call timing is not strict; can be performed whenever possible



Notes:
- Synchronous Control Loop
- Only one thread can access hardware at a time
  - risks "overrun" in a single cycle if performing service takes a long time compared to required control loop frequency

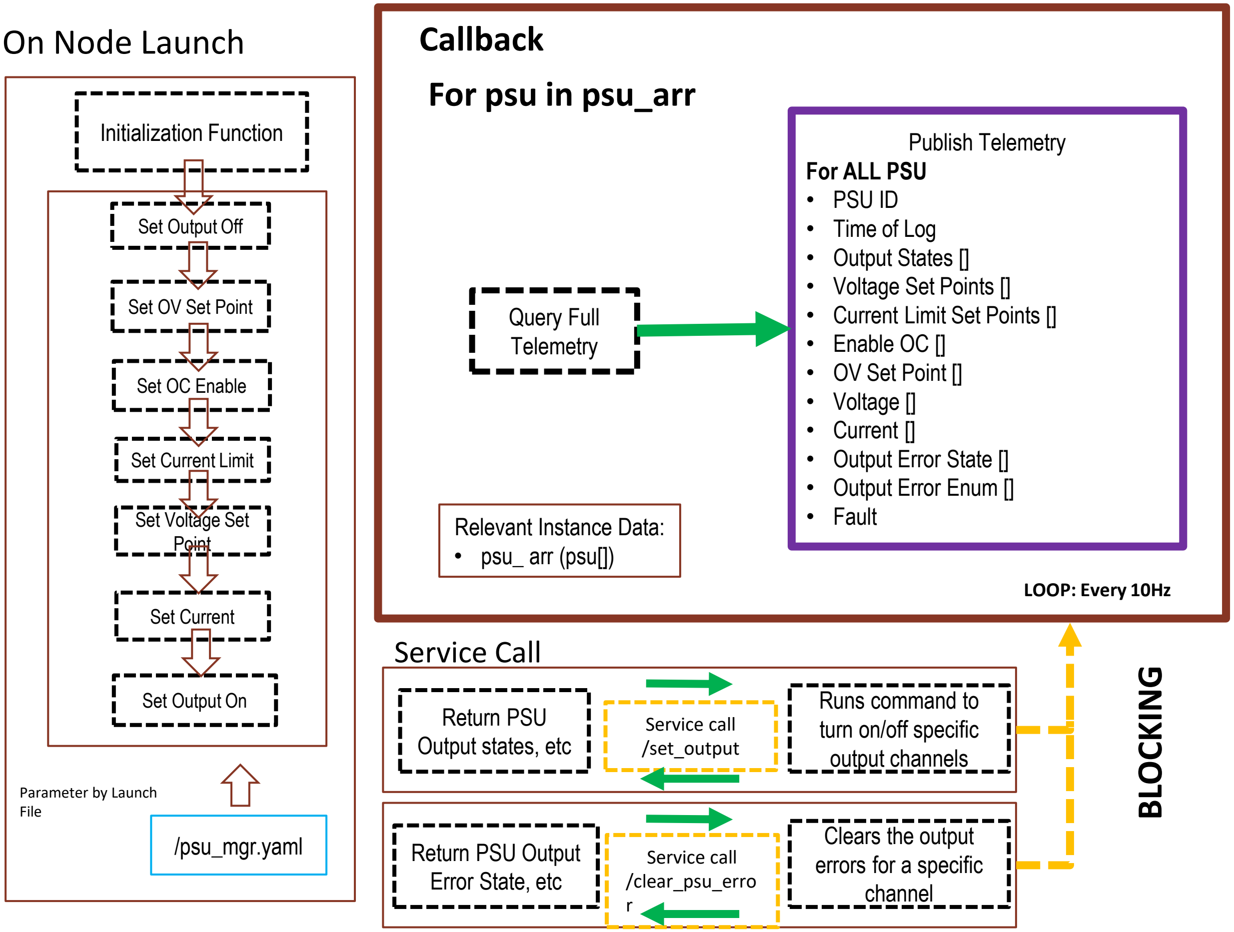
Architecture #3: Asynchronous Control, Single Thread

Assumed Requirements:
1. Query / Publish telemetry at specific frequency is not critical if **delayed**
2. Service calls should be performed when available (even if blocking)



Notes:
- Asynchronous Control Scheme
  - Only one callback runs and interfaces with hardware at a time
- Assumes that telemetry output can be delayed if service call takes too long
  - Need requirements on control loop frequency
- **Chosen architecture since hardware does not support multithreading**

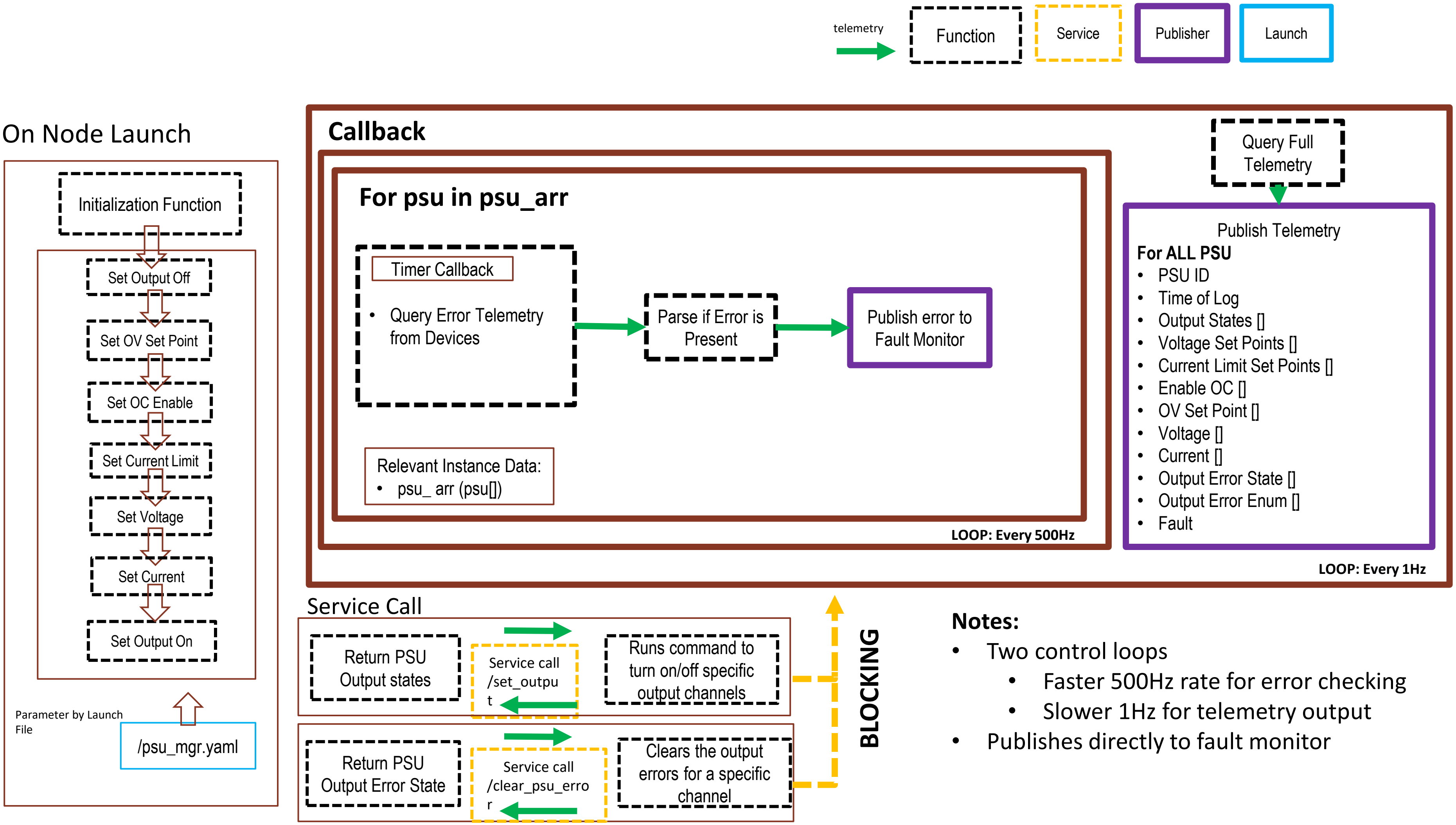
State Machine for Single Power Supply Node



Notes:

- Single control loops through every PSU and generates telemetry messages
- Service calls are **blocking** (mutually exclusive callbacks)
- Output can be changed during a fault via. service call
    - Handled by FP_MGR
- Stress testing with 1 NUC, 2 PSU, USB:
    - **10Hz control loop frequency**

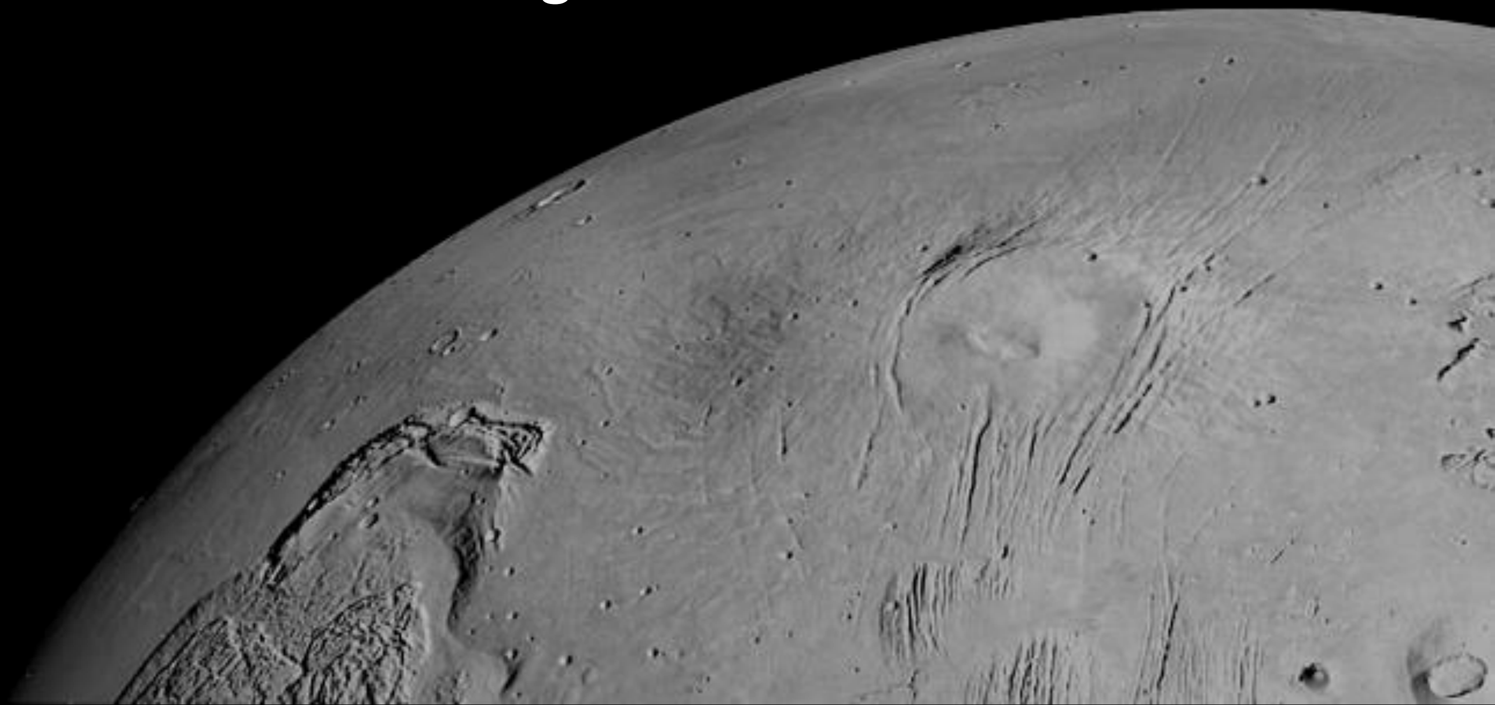
Alternative Architecture



Notes:

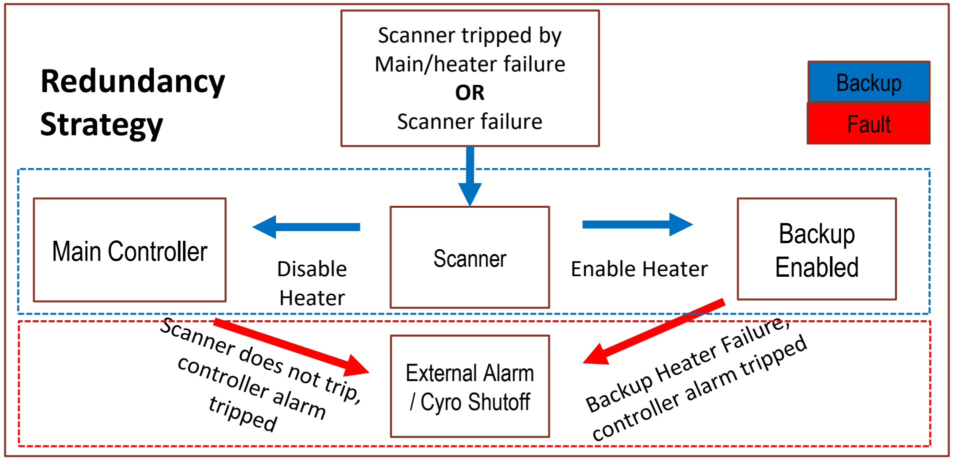
- Two control loops
  - Faster 500Hz rate for error checking
  - Slower 1Hz for telemetry output
- Publishes directly to fault monitor

THM_MGR:
ROS2 Thermal Controller Package



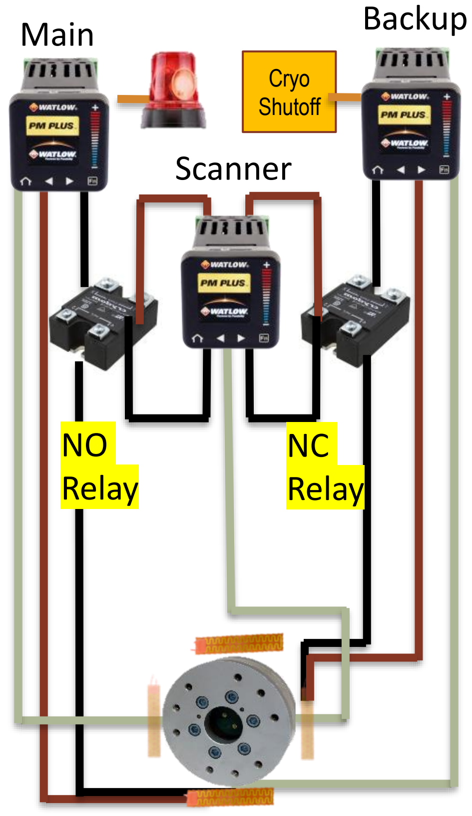
Thermal Manager Objective

- Objective: Create a CASAH Module that can manage multiple Thermal Controllers the core functionality of:
  1. Initializing Controllers
  2. Query and Publish Telemetry
  3. Allow operators / other modules to
     1. Clear errors
     2. Change Alarm Set Points
     3. Change Heating Control Set Point
- Controllers originally to be used for TVAC Testing [-50C to 70C]
  - Heaters and Thermocouples for closed loop control to set point
  - "Thermal Zones" for Single Redundancy
  - Watlow PM PLUS PID & Integrated Limit Controller, Omega Heaters, Crydom DC Relays



Single Failure - Redundancy

Double Failure - Hardware at Risk



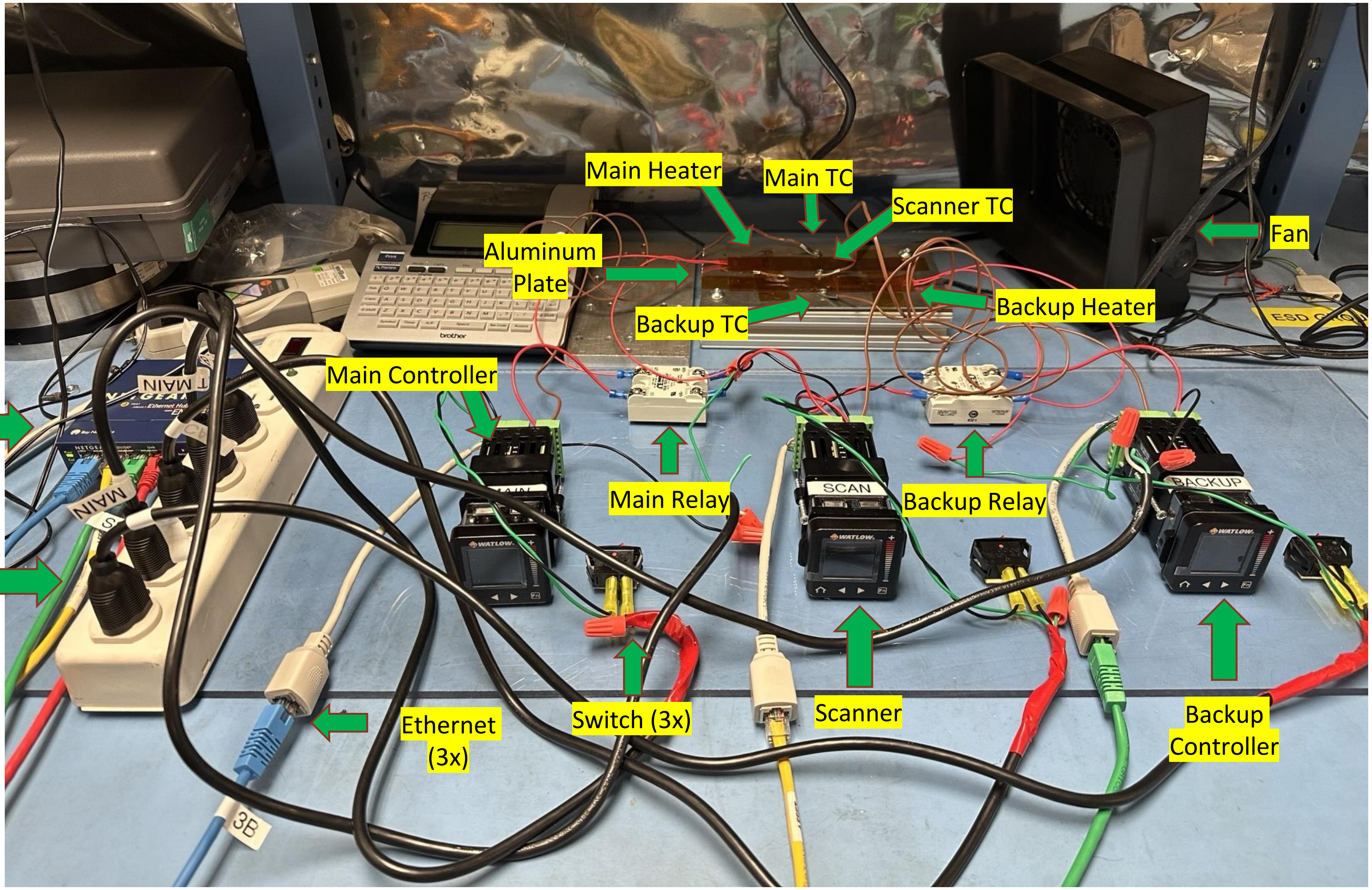
Thermal Zone Concept

FlatSat Physical Setup

* 120VAC or 24V DC

Network Switch

Power Supply*



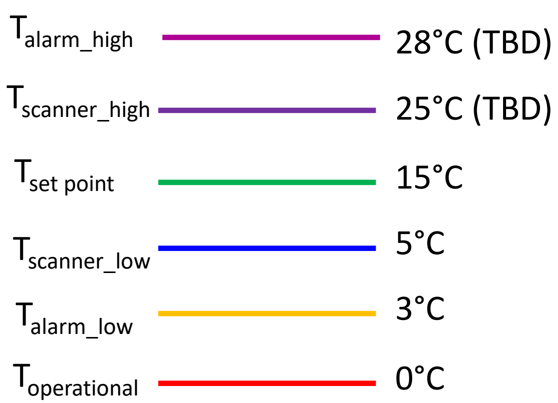
Thermal Controller Required Functions (Manual & Commanded)

Main Controller
1. Set Control Set Point to **T_Set_Point**
2. Set Alarm 1 Set Points to **T_Alarm_Low, T_Alarm_High**
3. Heat until **T_Set_Point** [Output 1]

Scanner
1. Set Alarm 1,2 Set Points to **T_Scanner_Low, T_Scanner_High**
2. Disable main heaters if **T_Scanner_Trip** is reached [Output 2]
3. Engage Relay for Backup Heaters **if T_Scanner_Trip** is reached [Output 1]

Backup Controller
1. Set Control Set Point to **T_Set_Point**
2. Set Alarm 1 Set Points to **T_Alarm_Low, T_Alarm_High**
3. Heat until **T_Set_Point** [Output 1]

Hardware Details
- All heating / alarm control is done by controllers onboard processing
- Controller sends PWM signals to heaters to reach temperature based on TC readings
    - Closed Loop (PID)
- Each controller has **2x** outputs, alarms
- If alarm is triggered, relays enable / disable outputs with latching
- Alarm set points must be set during operation
    - Otherwise, alarm will trip when trying to reach control loop set point



Main Functionality of THM_MGR

Hardware Pre-existing Functionality:

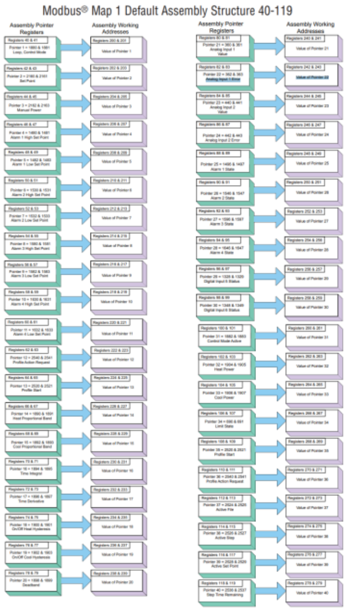
- PID Heating
- Alarm Output Behavior

Needed Software Functionality:

- Set Control Set Point
- Set Alarm (High / Low)
- Read Temperature
- Read TC Error
- Read Alarm 1,2 State
- Read Heat Power
- Read Alarm 1,2 Set Points (high / low)
- Read Control Set Points

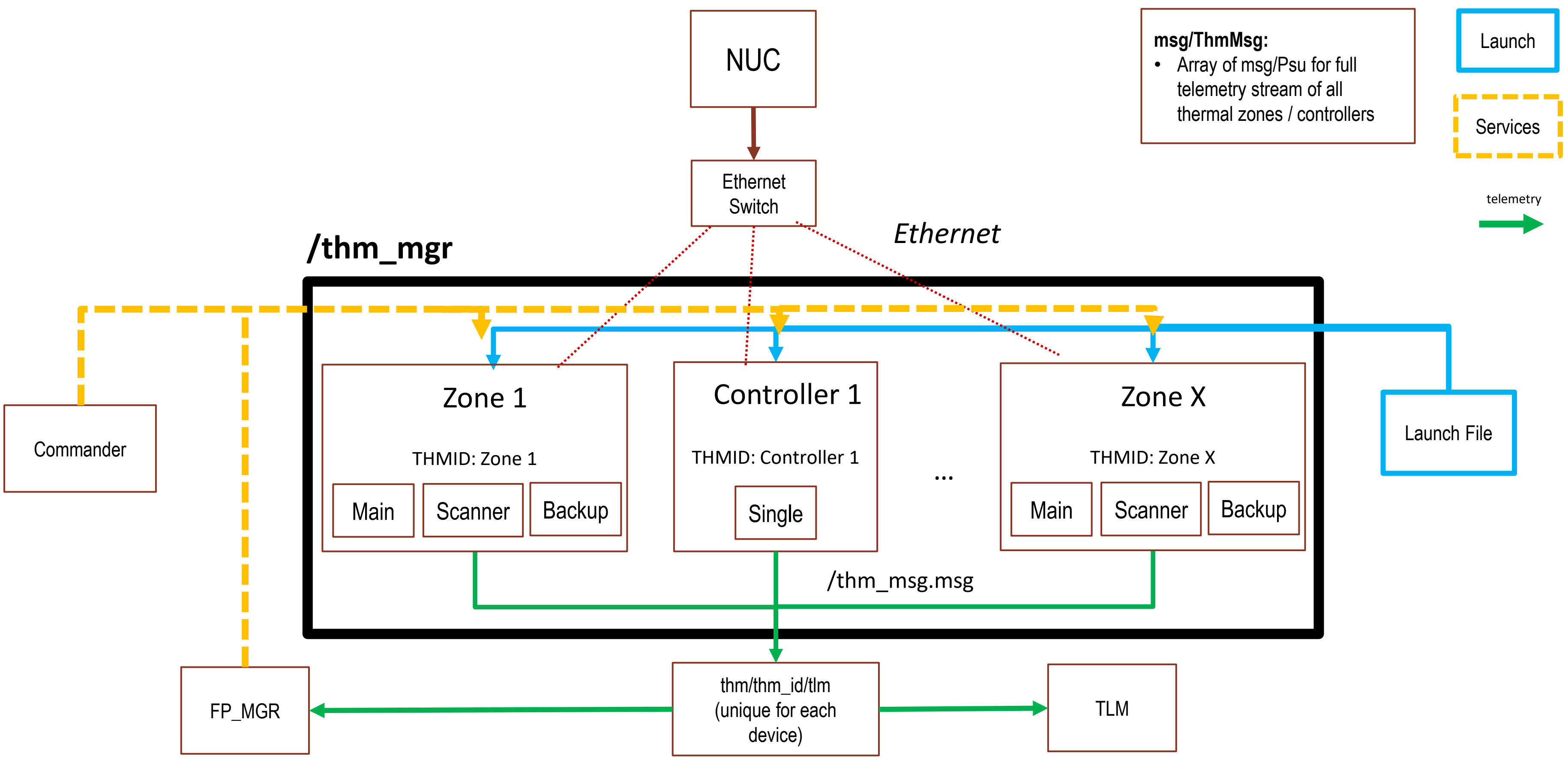
Thermal Controller can be separately programmed via. screen

- Watlow Controllers use MODBUS TCP
  - Write/Read from specific registers (32bit)
  - Using PyModbus library – **does not support multithreading**



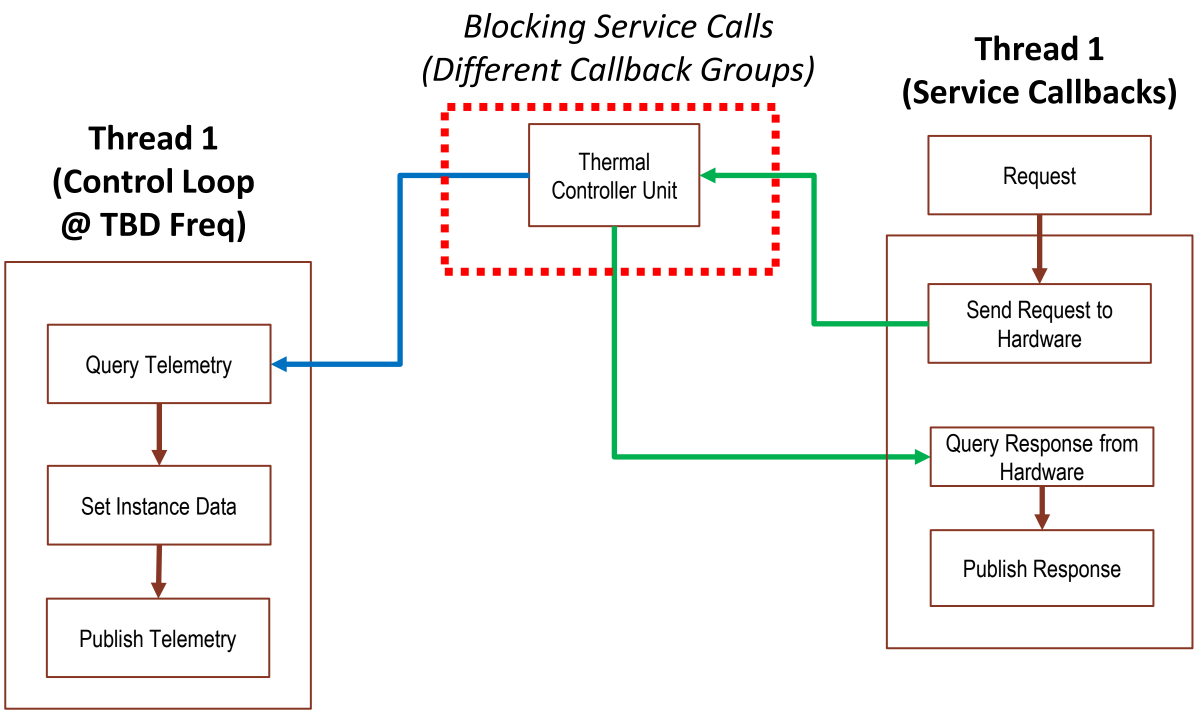
MODBUS Register List

THM_MGR Setup



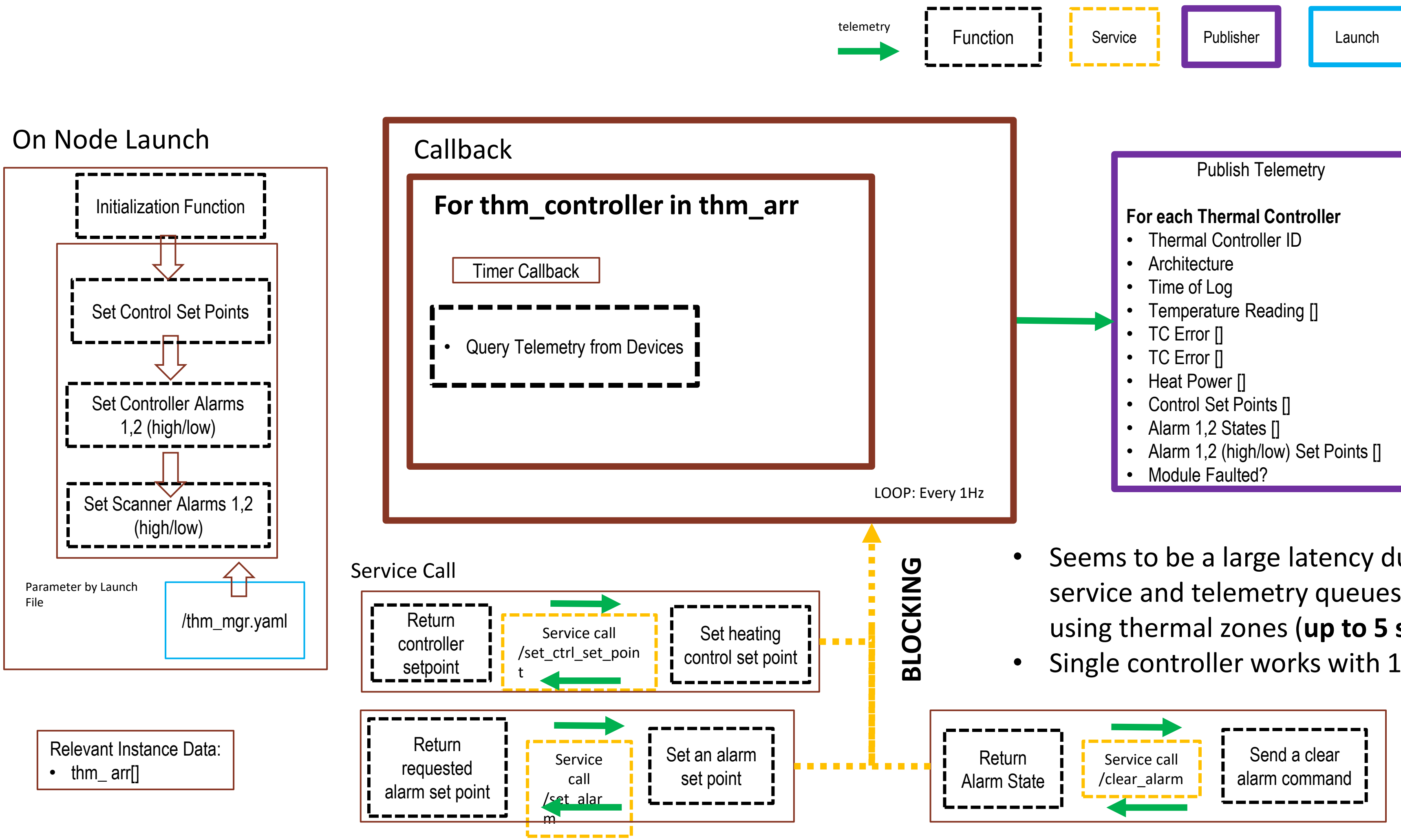
Architecture #3: Asynchronous Control, Single Thread

Assumed Requirements:
1. Query / Publish telemetry at specific frequency is not critical
2. Service calls should be performed when available (even if blocking)



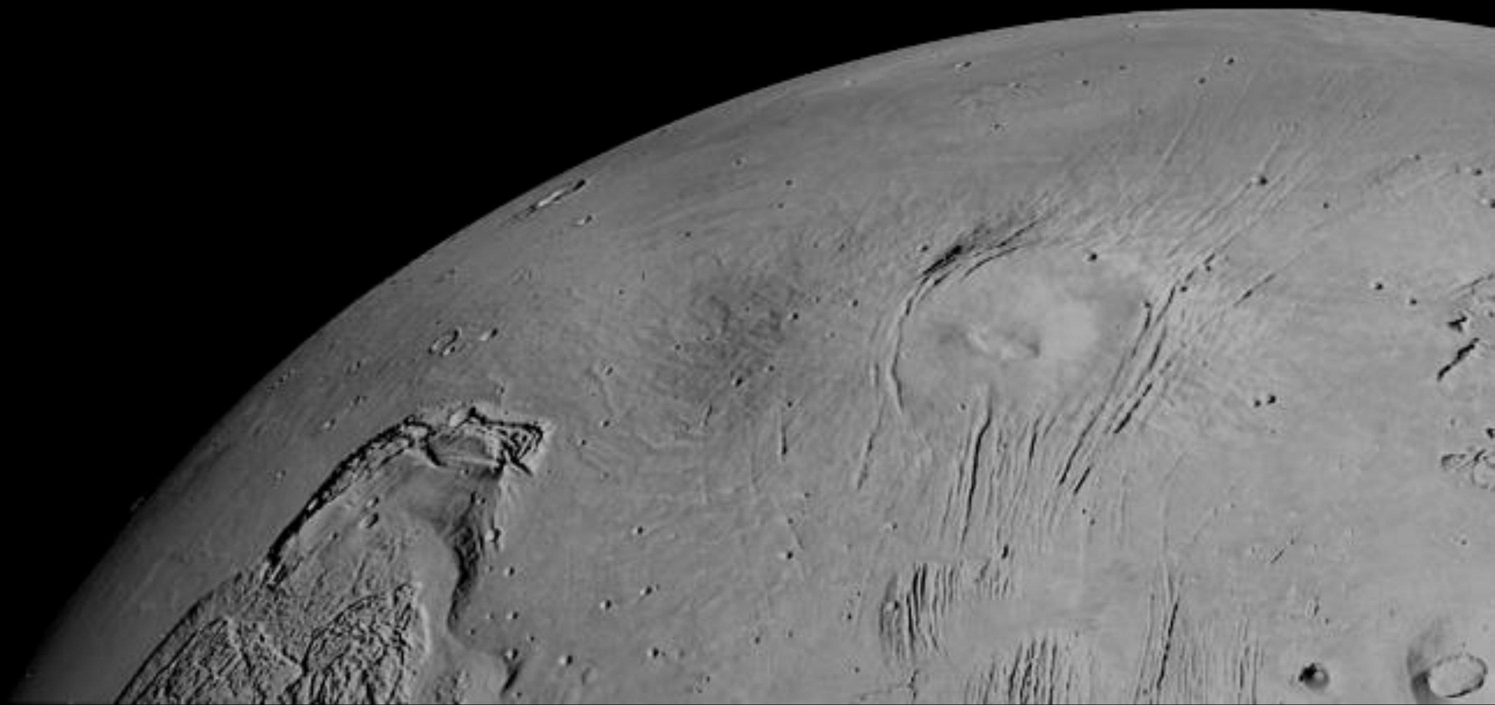
Notes:
- Asynchronous Control Scheme
  - Only one callback interfaces with hardware at a time
- Assumes that telemetry output can be delayed if service call takes too long
  - Need requirements on control loop frequency
- **Chosen architecture since PyModbus does not support multithreading**

State Machine for Thermal Manager



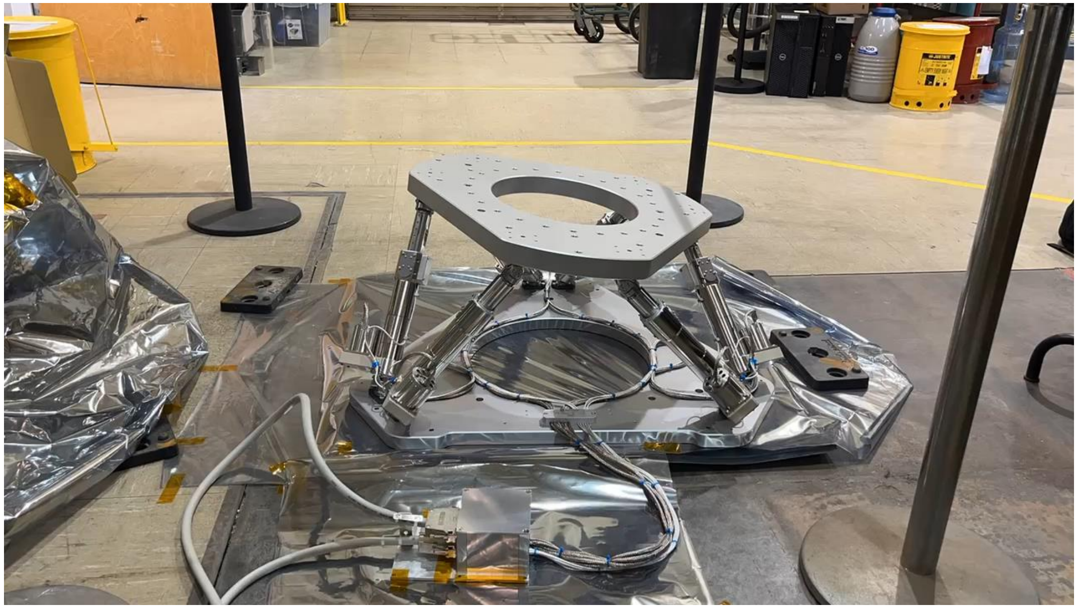
- Seems to be a large latency during service and telemetry queues when using thermal zones (**up to 5 seconds**)
- Single controller works with 1Hz

V&V



hxpd_mgr

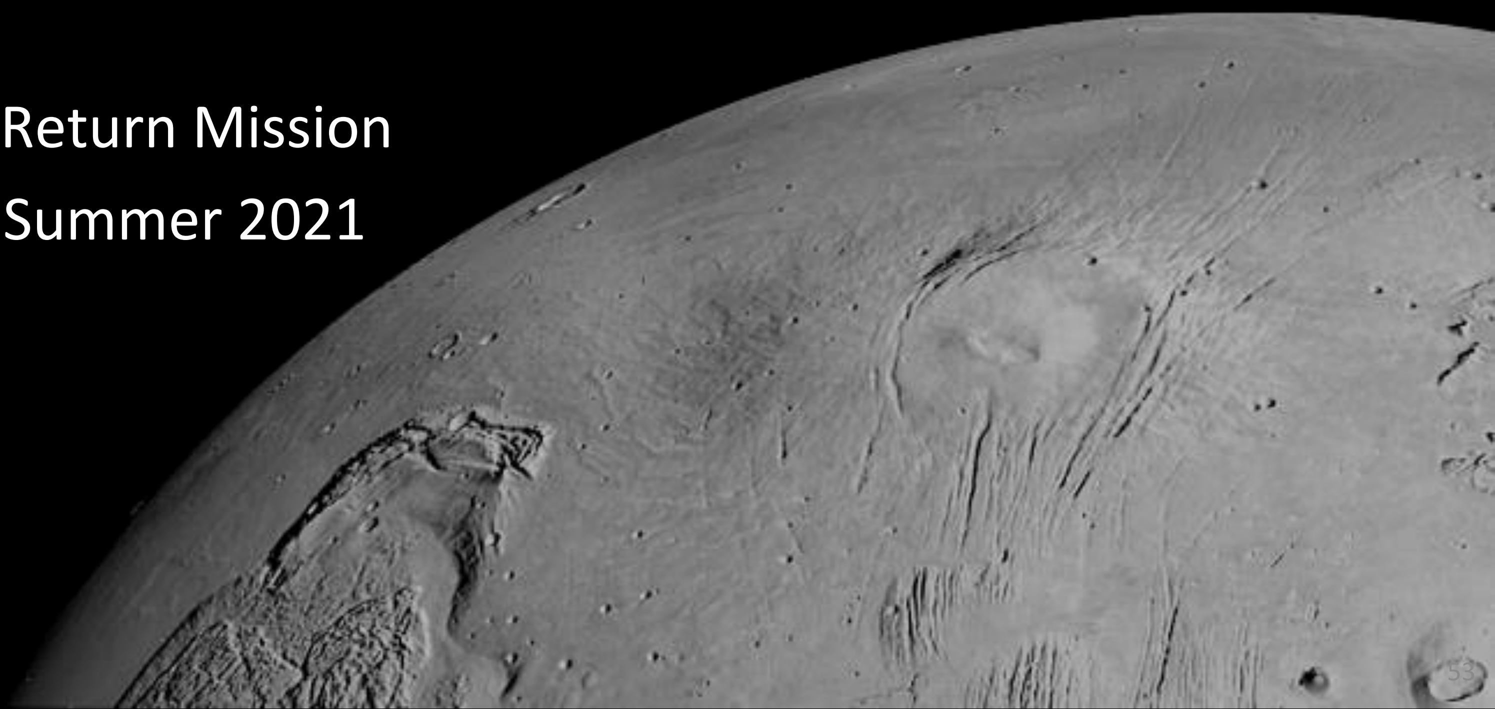
- Performed basic checkouts for **Symétrie** Hexapod
  - Range of Motion
  - Stopping with FTS
- Helped with test procedure / actually interfacing with CASAH operator tools, testbed deployment



End Effector Initial Developmental Testbed

Mars Sample Return Mission

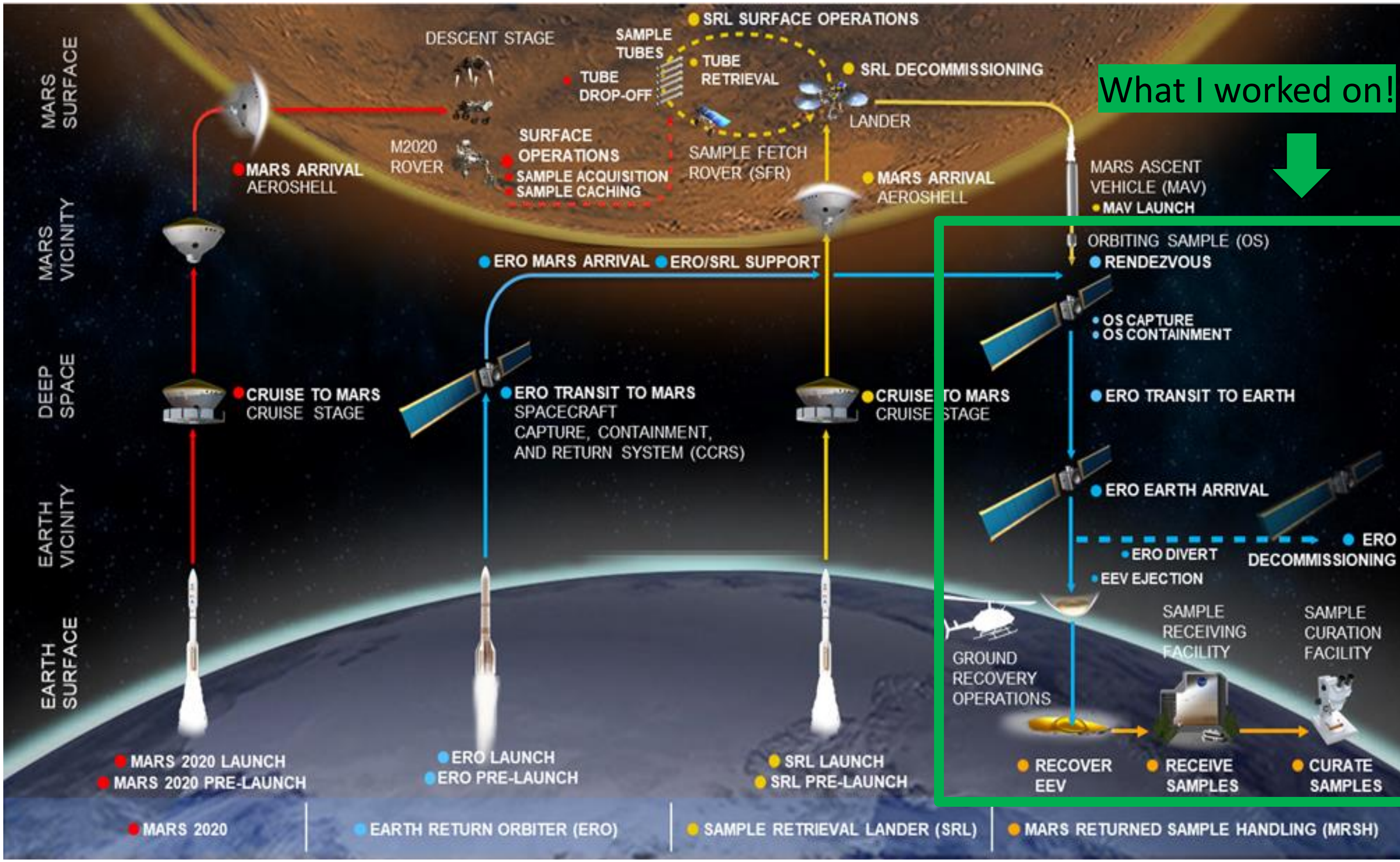
Spring 2020 - Summer 2021



Disclaimer

- The decision to implement Mars Sample Return will not be finalized until NASA's completion of the National Environmental Policy Act (NEPA) process
- This document is being made available for information purposes only
- The information presented has been approved through export control and has been released to be shown to the public

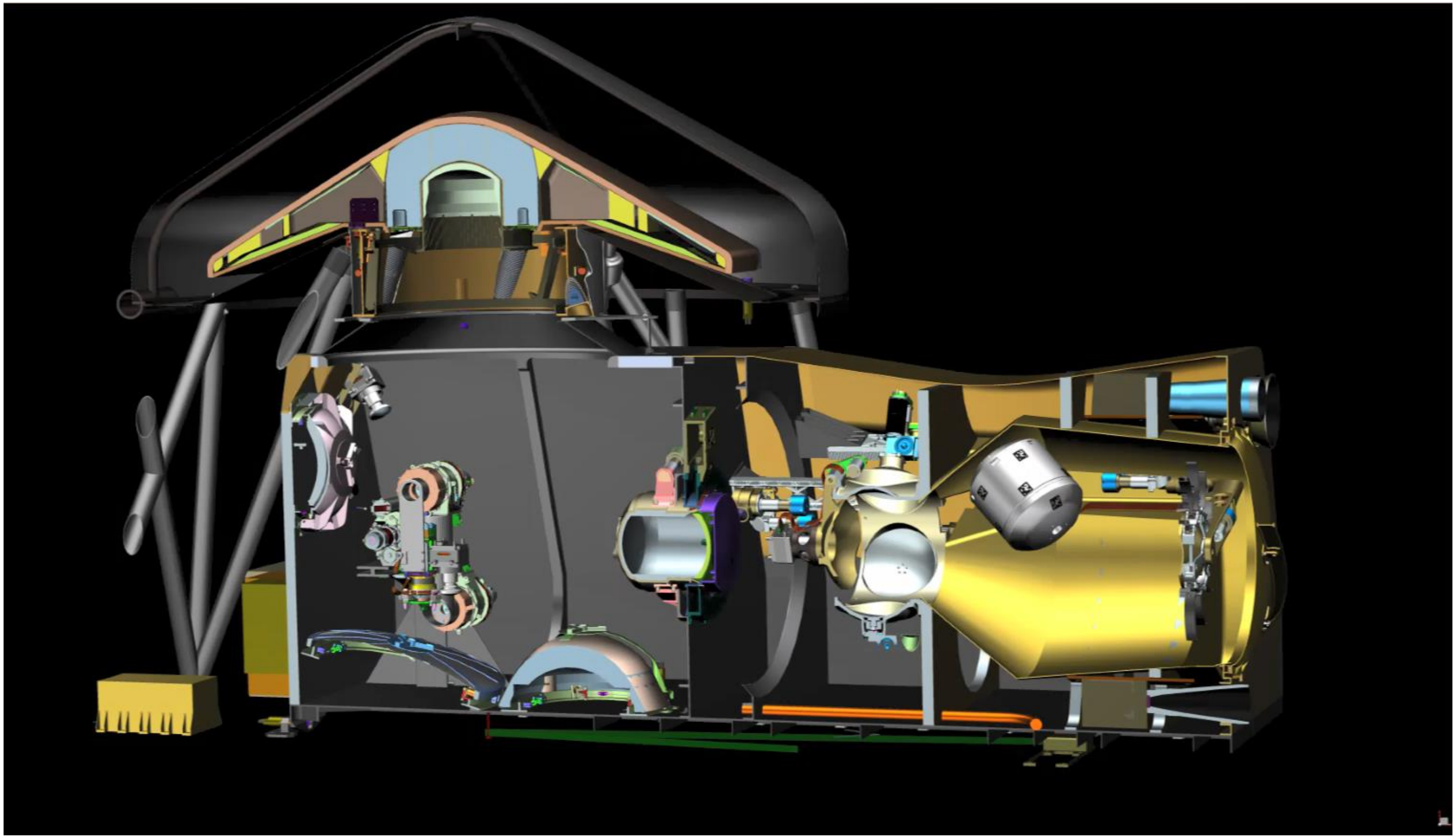
Mars Sample Return Planning Overview



Pre-Decisional Information – For Planning and Discussion Purposes Only

CCRS Containment Operations

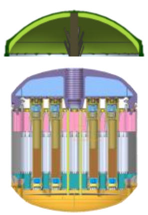
The CCRS Capture and Containment Module uses an end effector on the Robotic Transfer Arm to perform a series of assembly tasks to contain the OS, assemble the OS into the EEV, and maintain the Earth clean zone



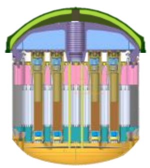
Artist's Concept

Assembly Operations

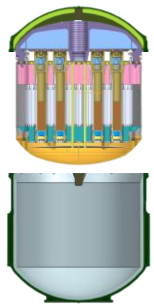
OS = Orbiting Sample
PCV = Primary Containment Vessel
SCV = Secondary Containment Vessel
CAM = Containment Assurance Module
EEV = Earth Entry Vehicle



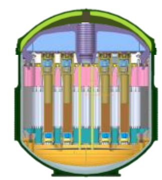
1. PCV Lid aligned with OS



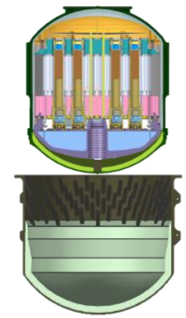
2. PCV Lid mated with OS



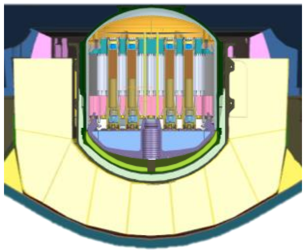
3. OS aligned with PCV Base



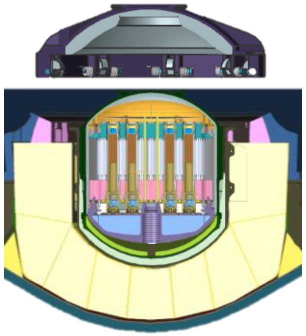
4. PCV Base and Lid mated and brazed



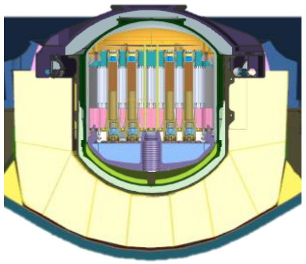
5. PCV aligned with the SCV Base



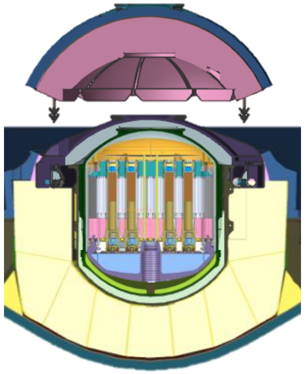
6. PCV inserted into SCV Base/EEV



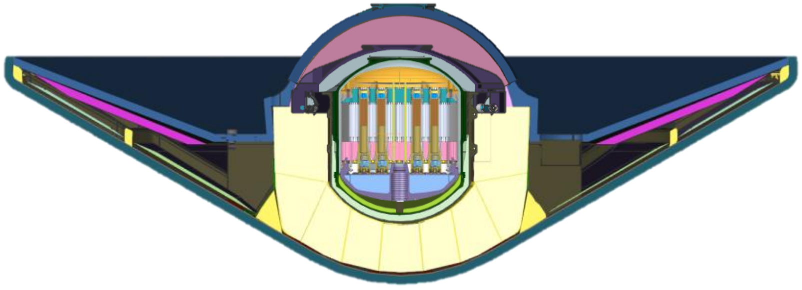
7. SCV Lid aligned with the SCV Base/EEV



8. SCV Lid mated with the SCV Base/EEV



9. CAM Lid aligned with the EEV

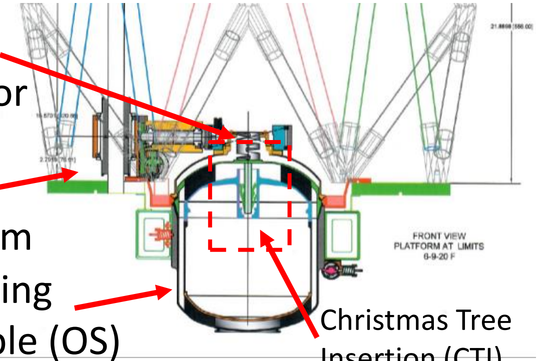


10. CAM Lid mated with the EEV

The Problem: End Effector Misalignment

End Effector Misalignment can be due to a variety of issues such as:

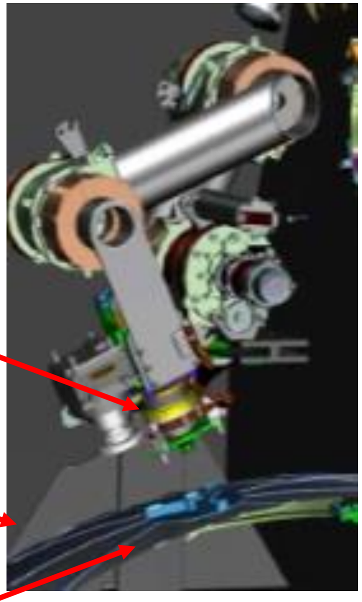
- **Joint errors** (encoder inaccuracy/sensitivity)
- **Kinematic Errors** (model is not 100% true to real geometry)
- **Non-kinematic errors** (backlash, stiffness, temperature effect)



PCV Lid Insertion into OS

Objectives:

1. Create a platform to test behavior of Robotic Transfer Arm (RTA) end effectors when misalignment is present
   - **Not testing static preloading of seals due to higher load requirements**
2. Test the capability of mechanical alignment strategies (e.g., Christmas Tree Insertion, end effector posts)
3. Measure the forces and torques present during docking/insertion procedures



Misalignment during docking

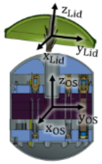
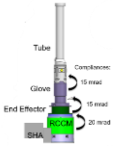
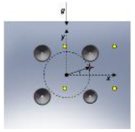
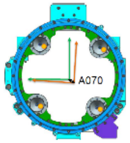
Testbed Requirements

Load Requirements

Function	Force Required (N)	75% Margined Load Required (N)*	Rationale
CTI Insertion	80	140	Load Estimates calculated and provided from previous work. Magnitudes of values quite similar to those presented in MSL Bit Box and M2020 SHA insertion tests.  MSL Bit Box  M2020 SHA
Braze Insertion	80	117	
PCV Grip	200	350	
PCV Place	200	350	
SCV Lid Grip	200	350	
SCV Lid Place	200	350	
CAM Lid Grip	200	350	
CAM Lid Place	100	175	
ERM Lid Grip	200	350	
ERM Lid Place	200	350	

* 75% added margin accounts for uncertainty in force required

Displacement Requirements

Offset	Required	Rationale
Position (along x-axis)	+/- 10 mm	All of the requirements are based off the SHA EE Misalignments. Each of the following requirements have ≈200% Margin. Additionally, the magnitude of the errors are similar to those present in the MSL Drill and Bit Box Tests  EE Misalignment  OS Pin Insertion  SHA EE RCCM (M2020)  Drill and Bit Box (MSL)  SCS Bit Exchange (M2020)
Position (along y-axis)	+/- 10 mm	
Angular (about x-axis)	+/- 2.86 deg	
Angular (about y-axis)	+/- 2.86 deg	
Clocking (about z-axis)	+/- 2.86 deg	

Testbed Requirements

Stroke Requirements

Function	Stroke Required (mm)	50% Margined Stroke Required (mm)*	Rationale
CTI Insertion	65.8	98.7	Stroke Estimates calculated and provided from current CAD models of CCRS Architecture.
Braze Insertion	184.6	276.9	
PCV Grip	11.3	17.0	
PCV Place	201.6	302.4	
SCV Lid Grip	10.4	15.6	
SCV Lid Place	88.6	132.9	
CAM Lid Grip	19.5	29.3	
CAM Lid Place	133.5	200.3	
ERM Lid Grip	10.4	15.6	
ERM Lid Place	17.6	26.4	

* 50% added margin accounts for uncertainty in stroke required

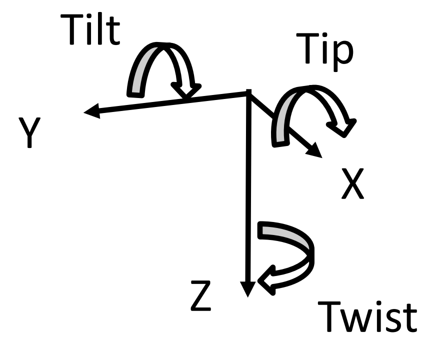
Testbed Requirements

Functional requirements the testbed:

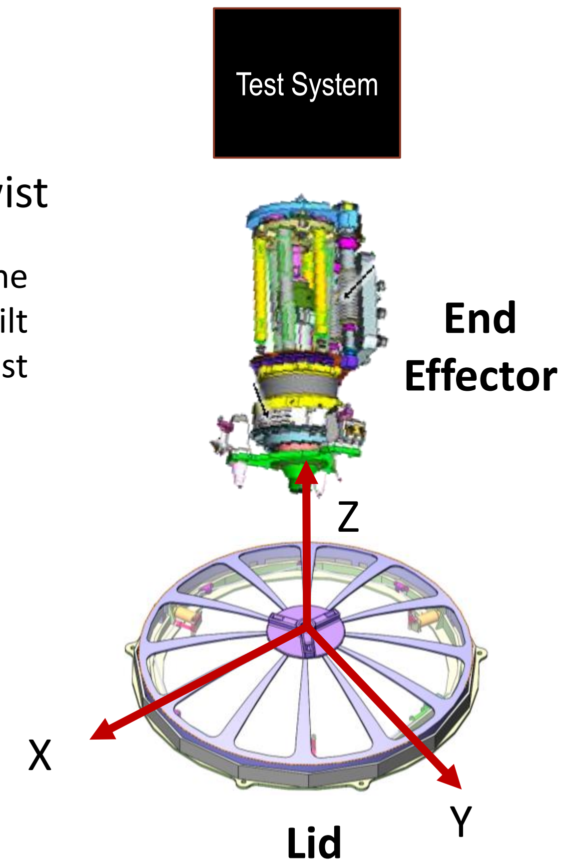
- Apply Force
- Provide Motion (6-DoF)
  - Provide Initial Alignment Error
    - Lateral
    - Angular
    - Clocking
- Measure Forces (6-DoF)

These requirements can be met using:

- Stewart Platform
- Linear Actuator
- 6-Axis Force Torque Sensor

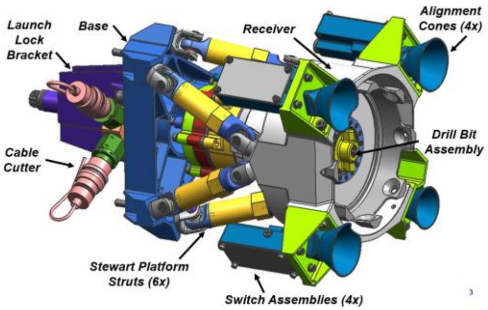


Lateral – XY plane
Angular – XY Tip/Tilt
Clocking – Z Twist

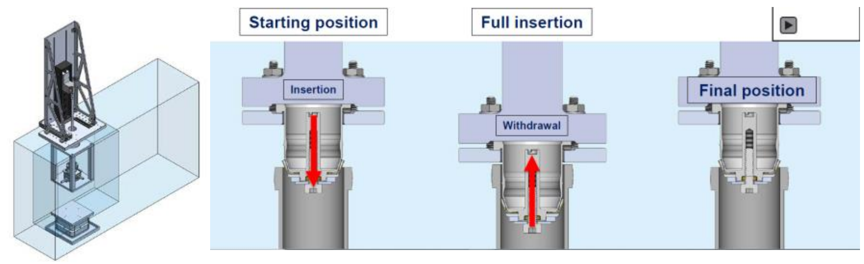


Previous Flight Project Testbeds

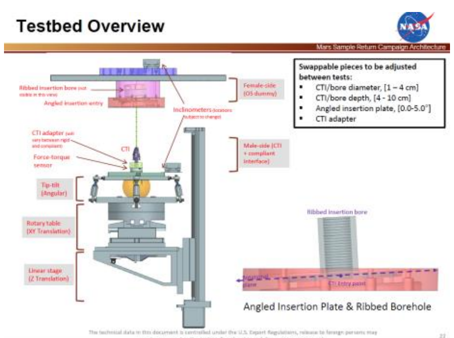
MSL Bit Exchange Development Test (2008)

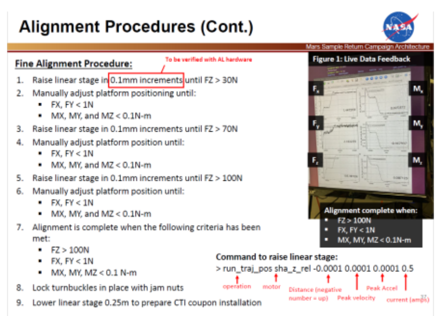


M2020 Tube Retainer Performance Characterization Testing (2020)



IFACT: Insertion Force & Alignment Characterization Testbed





Alignment Procedures (Cont.)

Mars Sample Return Campaign Architecture

Fine Alignment Procedure:

To be verified with AL hardware

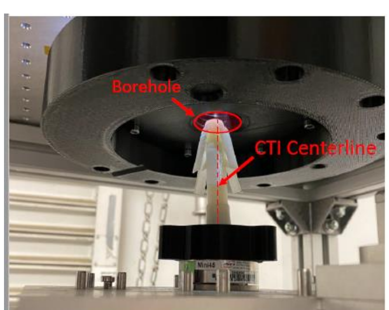
1. Raise linear stage in 0.1mm increments until FZ > 30N
2. Manually adjust platform positioning until:
   - FX, FY < 1N
   - MX, MY, and MZ < 0.1N-m
3. Raise linear stage in 0.1mm increments until FZ > 70N
4. Manually adjust platform position until:
   - FX, FY < 1N
   - MX, MY, and MZ < 0.1N-m
5. Raise linear stage in 0.1mm increments until FZ > 100N
6. Manually adjust platform position until:
   - FX, FY < 1N
   - MX, MY, and MZ < 0.1N-m
7. Alignment is complete when the following criteria has been met:
   - FZ > 100N
   - FX, FY < 1N
   - MX, MY, MZ < 0.1 N-m
8. Lock turnbuckles in place with jam nuts
9. Lower linear stage 0.25m to prepare CTI coupon installation

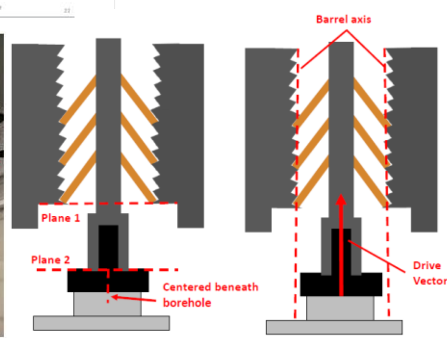
Figure 1: Live Data Feedback

Alignment complete when:
- FZ > 100N
- FX, FY < 1N
- MX, MY, MZ < 0.1N-m

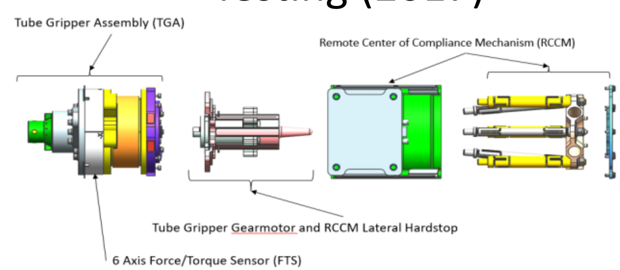
Command to raise linear stage:
> run_traj_pos sha_z_rel -0.0001 0.0001 0.0001 0.5

operation motor Distance (negative number = up) Peak velocity Peak Accel current (amps)





M2020 SHA Insertion/Misalignment Testing (2017)



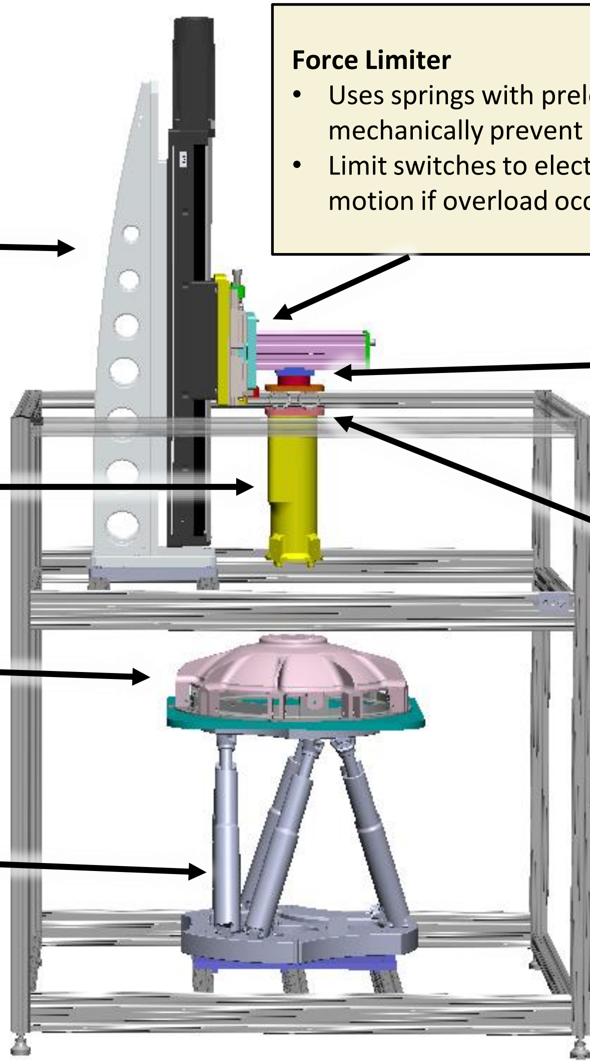
Testbed Configuration

Linear Actuator
- Attaches to post assembly
- Provides force and travel in Z-direction

End Effector (EE)*
- Grip/Place various containment lids/OS
- Test alignment strategies (cones/CTI)

Containment Lid*

Hexapod
- Movement in 6 DoF
- Mounts SCV (or various) Lids
- Provides misalignment for EE/Lid interface
- Supports EE/Lid/OS interface load



Force Limiter
- Uses springs with preload in order to mechanically prevent force overload
- Limit switches to electrically stop all motion if overload occurs

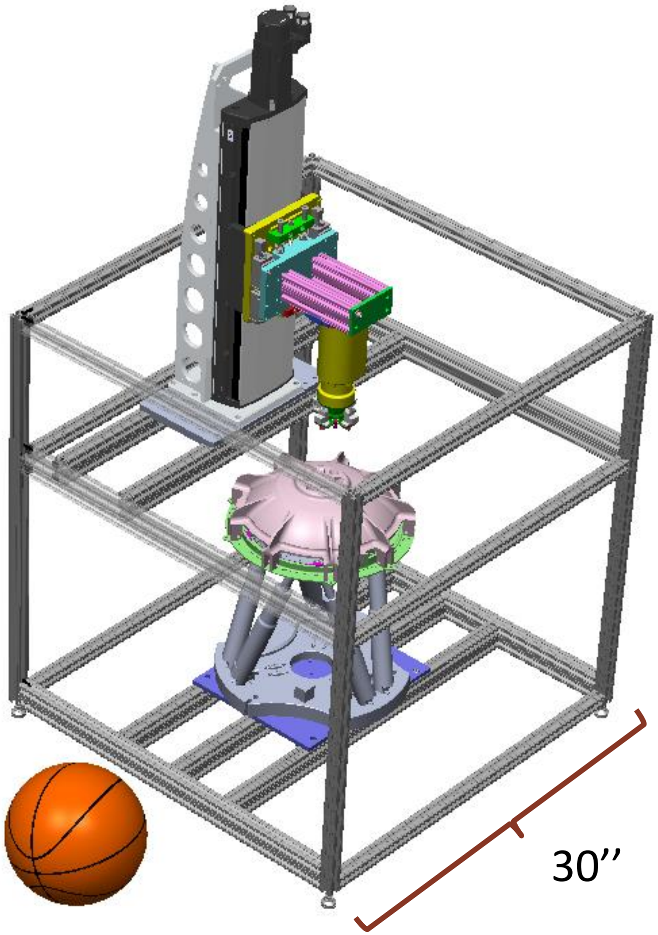
Force Torque Sensor (FTS)
- Records docking Force/Torques between EE and Lid/OS
- Feedback control if overloading

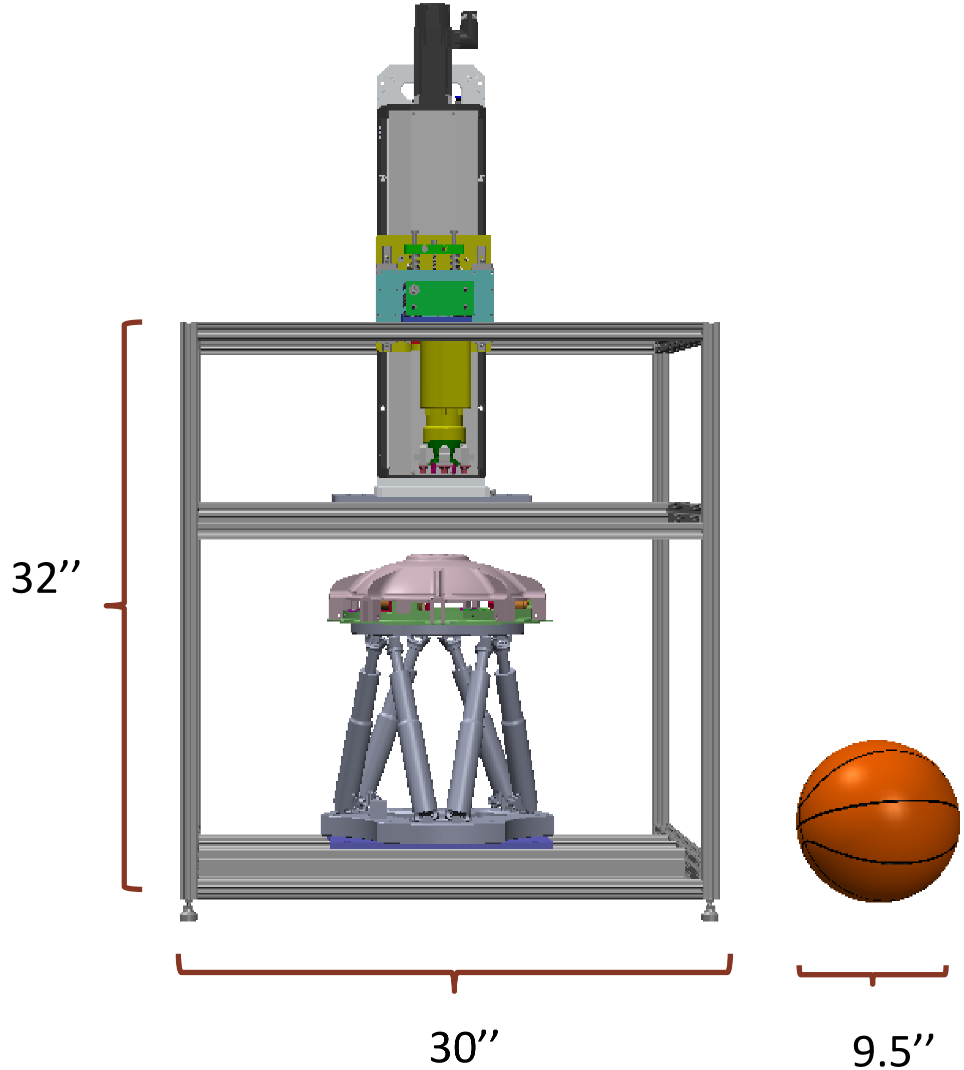
Remote Center of Compliance (RCCM)
- Helps to correct for misalignment during insertion
- Final design being produced (currently wire rope isolator stand in)

- Conceptualized and designed testbed
- Hardware trade studies
- Obtained quotes from vendors
- Procured parts (BoM), assembly instructions

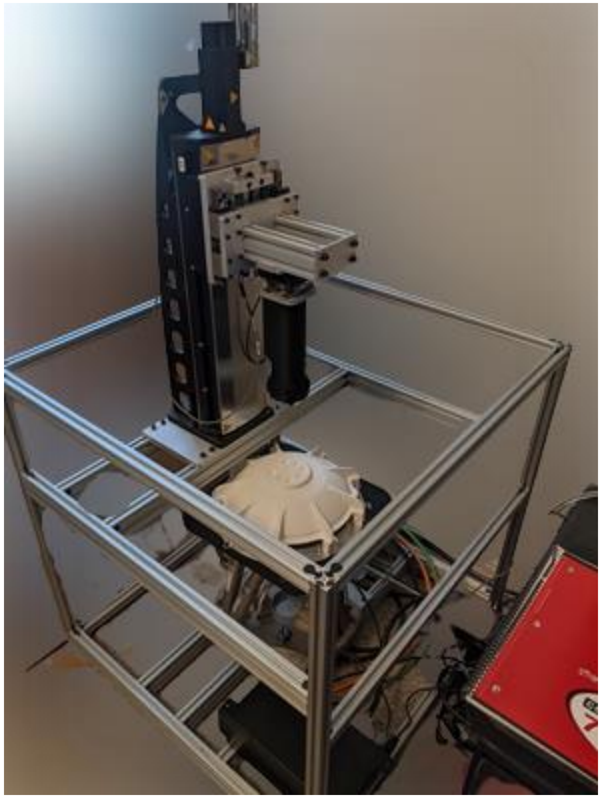
*These pieces are still in development, not final models

Test Bed Dimensions

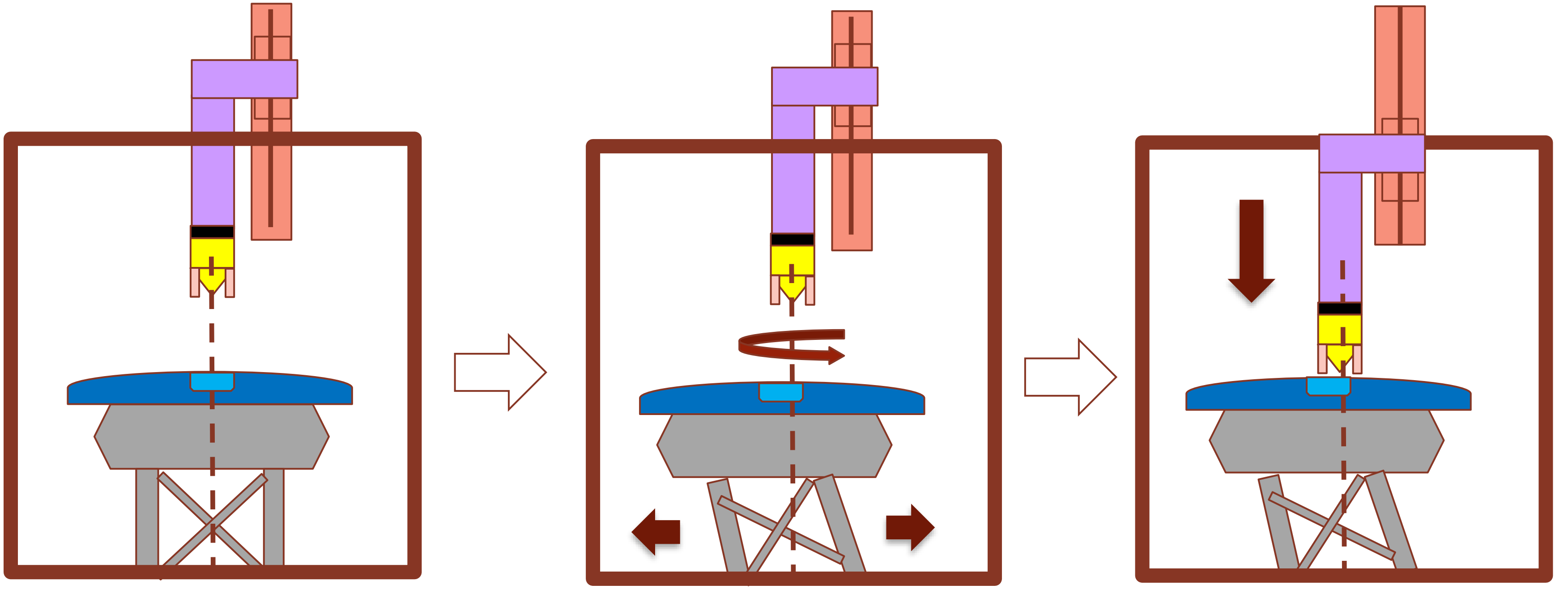




Additional Photos



Testbed Operational Concept



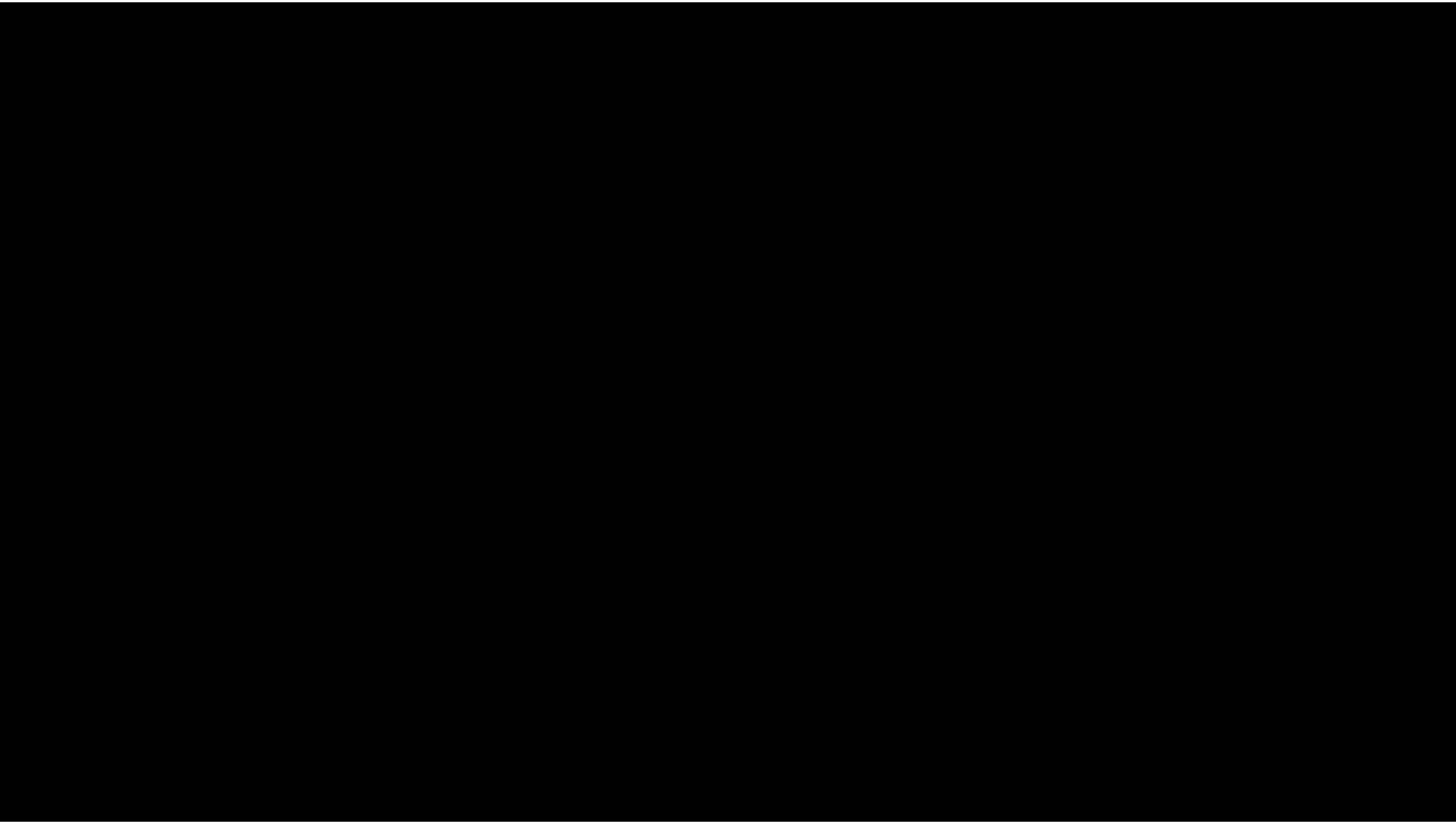
1. Calibrate and align hexapod to end effector

2. Move the hexapod in 5 DoF (lateral, angular, clocking) to create misalignment

3. Move Linear Actuator down vertically, begin docking until
- Load reached (350N)
- Switches Triggered
- Timeout

Record FTS Data and reset alignment

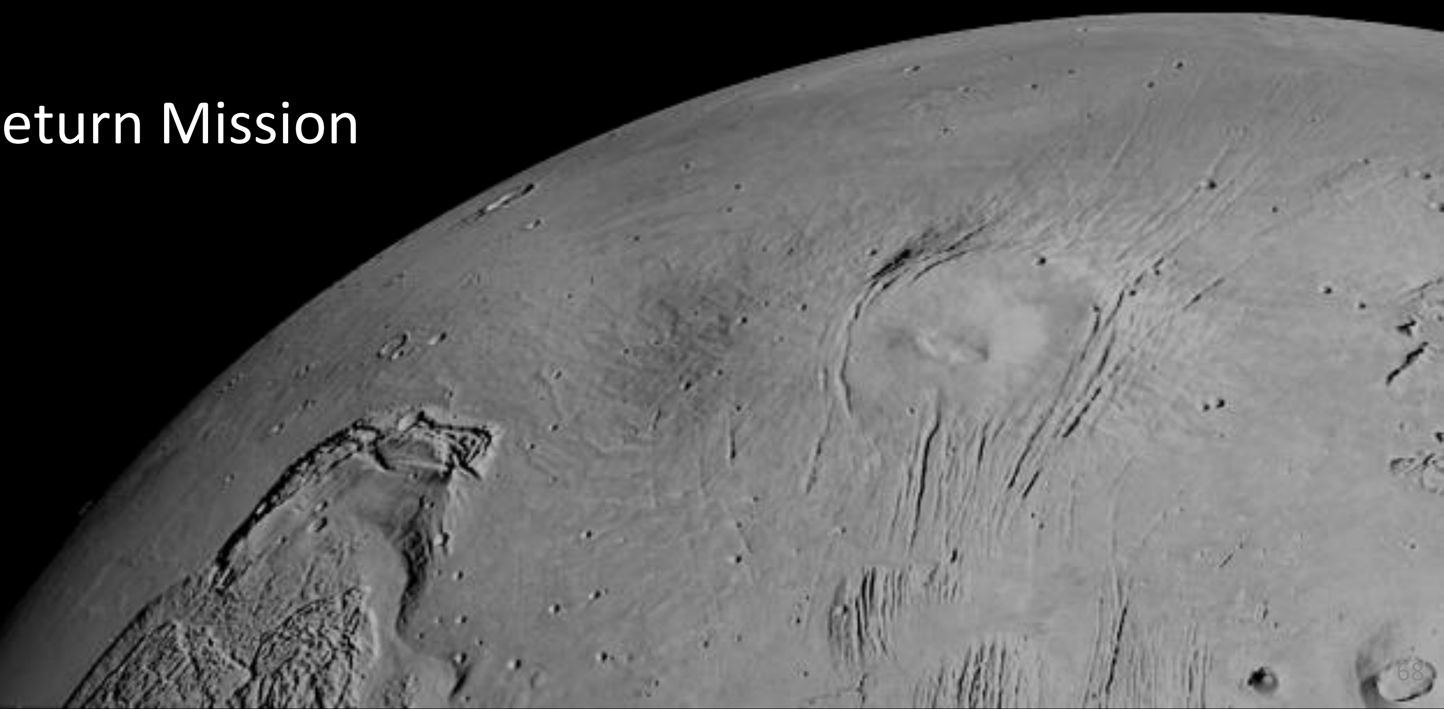
Real Life Photos/Demonstration



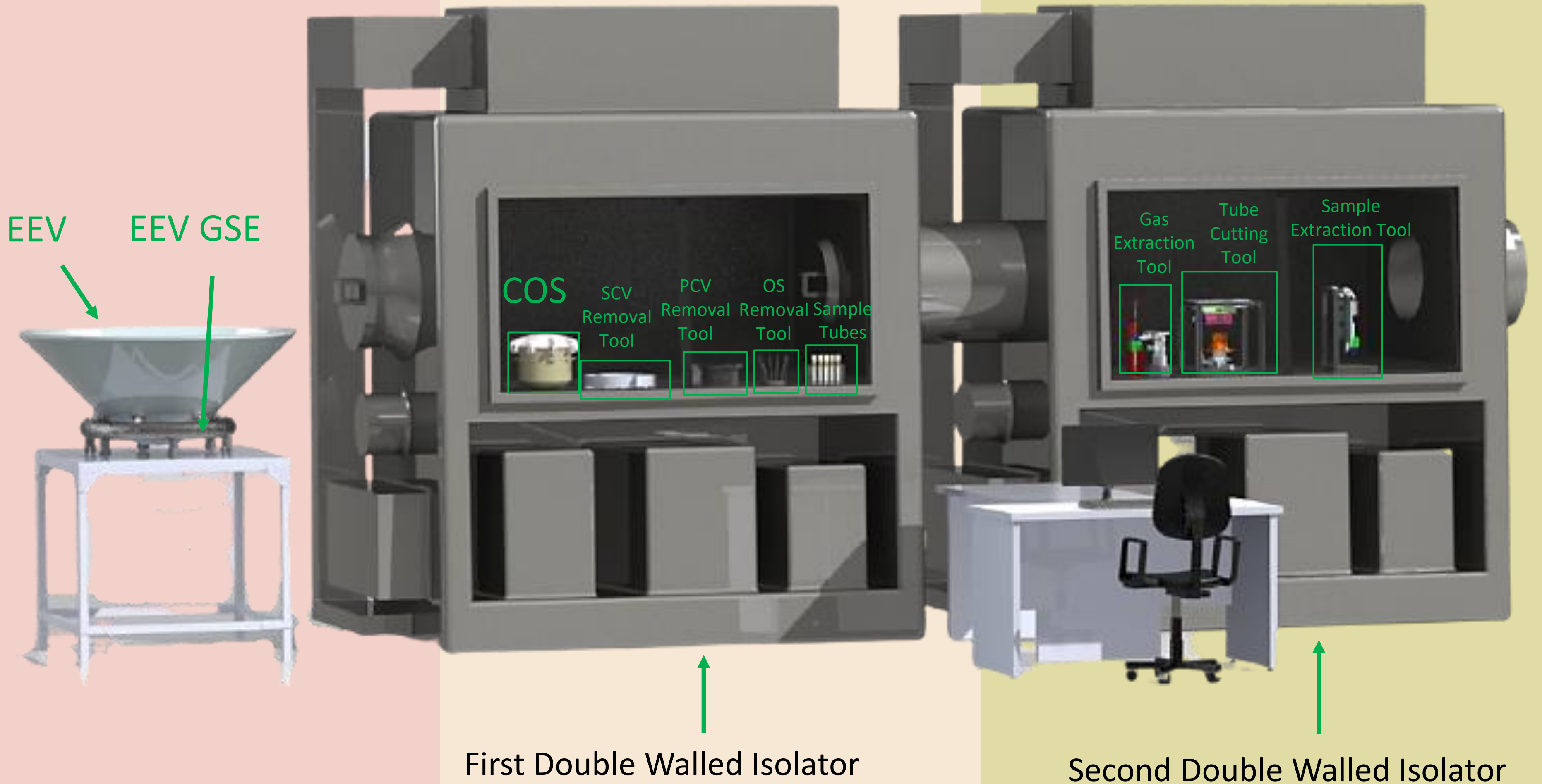
Mars Sample Return Handling Concept of Operations

Mars Sample Return Mission

Winter 2021



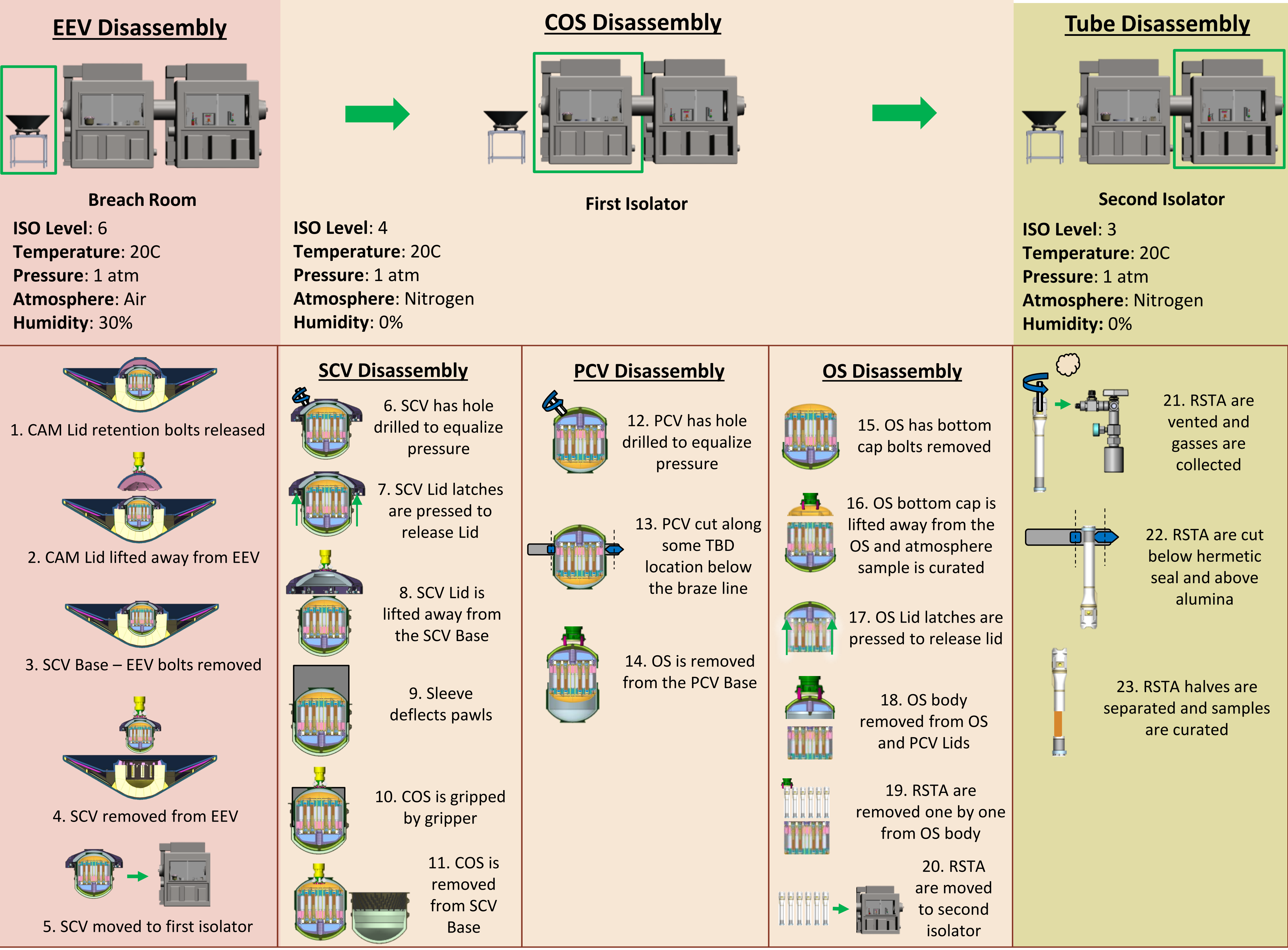
Disassembly Components



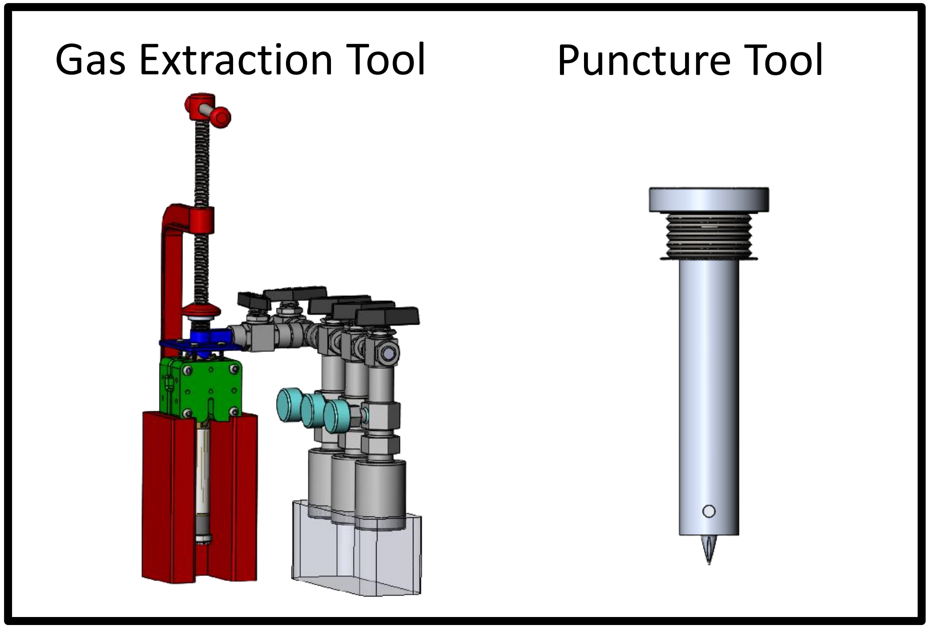
EEV Disassembly

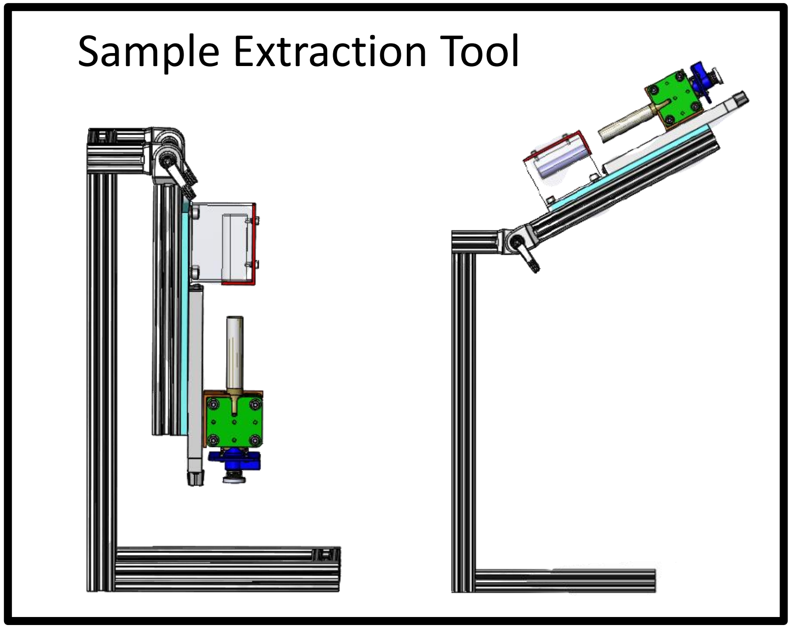
COS Disassembly

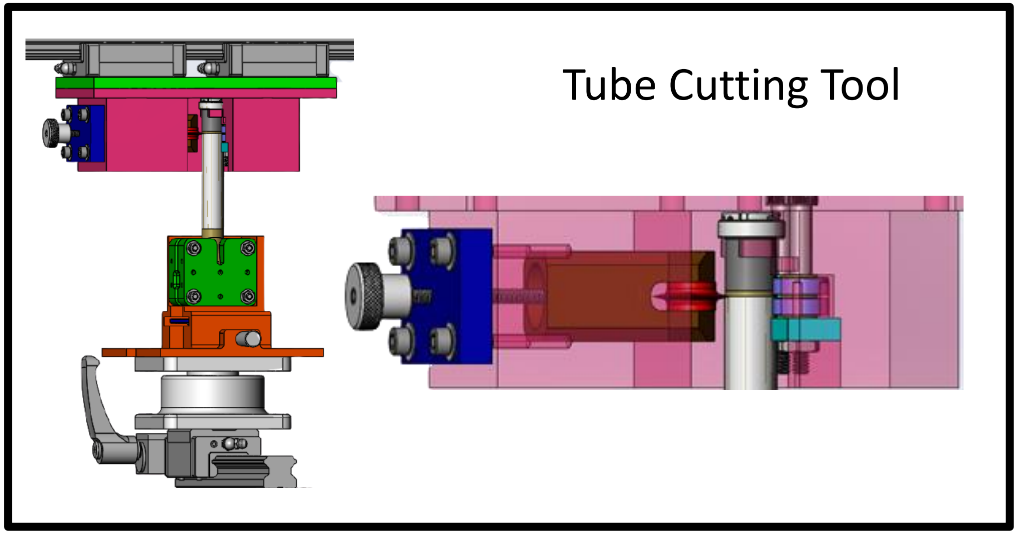
Tube Disassembly


EEV Disassembly
Breach Room
ISO Level: 6
Temperature: 20C
Pressure: 1 atm
Atmosphere: Air
Humidity: 30%
COS Disassembly
First Isolator
ISO Level: 4
Temperature: 20C
Pressure: 1 atm
Atmosphere: Nitrogen
Humidity: 0%
Tube Disassembly
Second Isolator
ISO Level: 3
Temperature: 20C
Pressure: 1 atm
Atmosphere: Nitrogen
Humidity: 0%
1. CAM Lid retention bolts released
2. CAM Lid lifted away from EEV
3. SCV Base – EEV bolts removed
4. SCV removed from EEV
5. SCV moved to first isolator
SCV Disassembly
6. SCV has hole drilled to equalize pressure
7. SCV Lid latches are pressed to release Lid
8. SCV Lid is lifted away from the SCV Base
9. Sleeve deflects pawls
10. COS is gripped by gripper
11. COS is removed from SCV Base
PCV Disassembly
12. PCV has hole drilled to equalize pressure
13. PCV cut along some TBD location below the braze line
14. OS is removed from the PCV Base
OS Disassembly
15. OS has bottom cap bolts removed
16. OS bottom cap is lifted away from the OS and atmosphere sample is curated
17. OS Lid latches are pressed to release lid
18. OS body removed from OS and PCV Lids
19. RSTA are removed one by one from OS body
20. RSTA are moved to second isolator
21. RSTA are vented and gasses are collected
22. RSTA are cut below hermetic seal and above alumina
23. RSTA halves are separated and samples are curated

Tube Disassembly Tools







- Three conceptual tools
  - Demonstrate the various tube opening processes
- Published IEEE Paper!
- High level; no detailed design

Gas Puncture Trade Space

Process	Vibration	Chip/Dust Production	Potential Loss of Gas	Tube Deformation	Sample Composition Affected	Introducing Contaminant	Complexity	Overall Risk
Center Punch with Arbor Press	Medium	Low	Low	Low	No	Low	Low	Medium
Center Punch with Jackscrew	Low	Low	Low	Low	No	Low	Low	Low
Standard Drill Bit	High	High	Medium	Medium	Low	Low	Low	High
Step Bit	Medium	Medium	Medium	Medium	Low	Low	Low	Medium
Rotary Cutting Disc	High	High	Medium	Low	Medium	Low	Low	High
Slide Hammer	High	Medium	Low	Medium	Medium	Low	Low	Medium
Laser Cutter	Low	Low	High	Low	High	High	High	High
Melting and Inserting Tool	No	No	Low	High	Very High	No	High	High

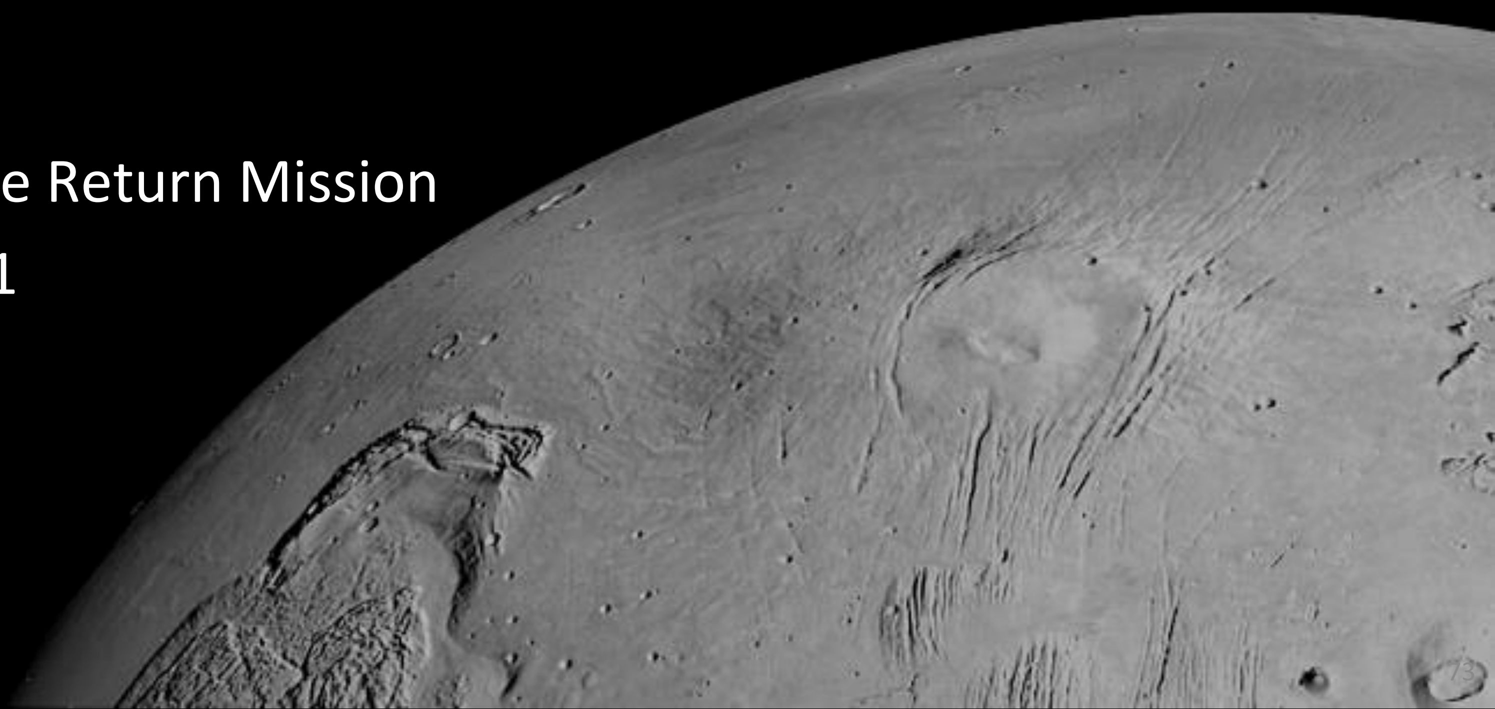
Suggestion: Using a center punch with a jackscrew to create a slow and continuous pressing motion. Small hole the size of tool tip will be created with little chip production for gas extraction

Testing has proven the capability of this tool with arbor press but jackscrew design has not been tested

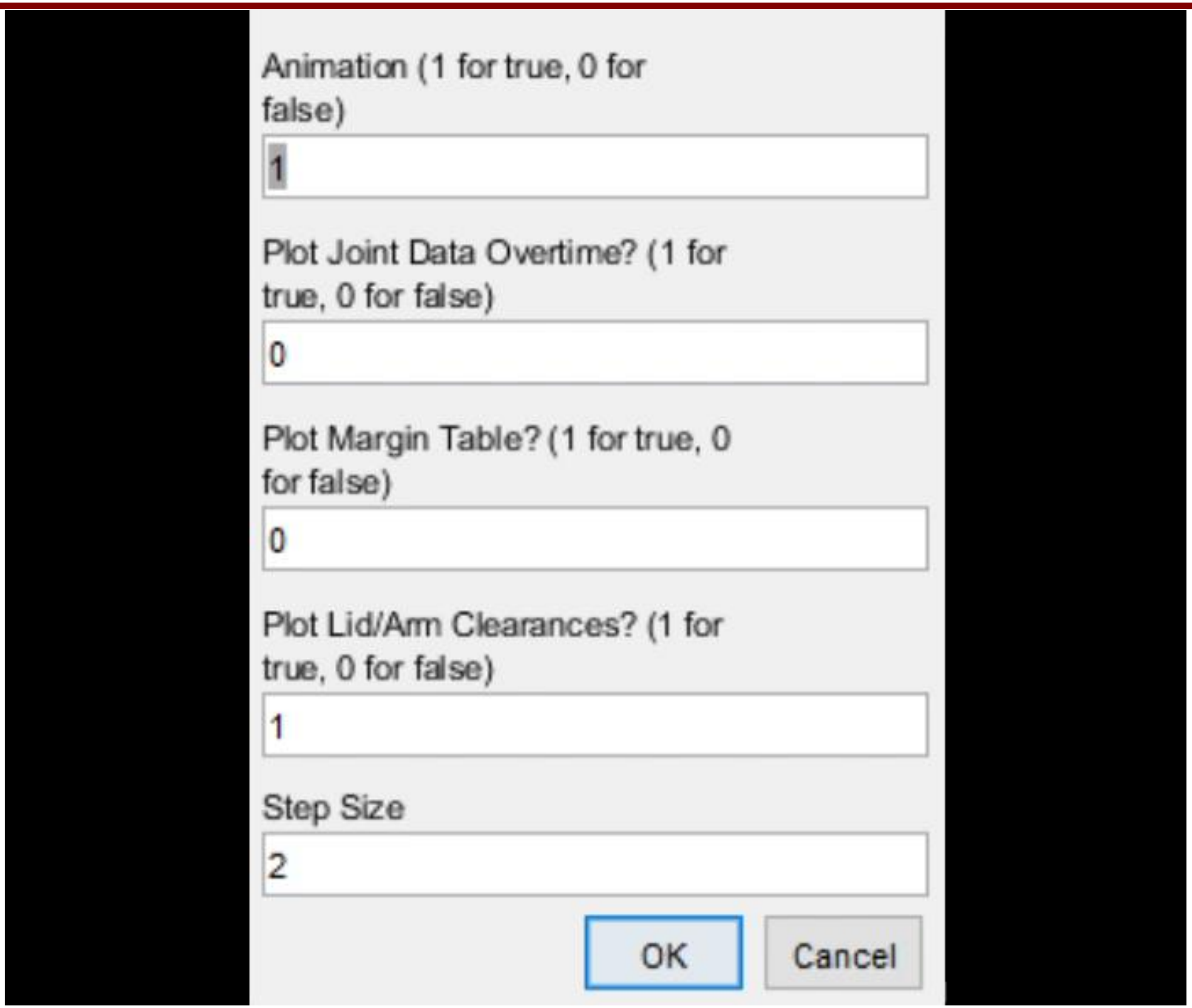
Robotic Transfer Arm (RTA) Kinematics

Mars Sample Return Mission

Winter 2021

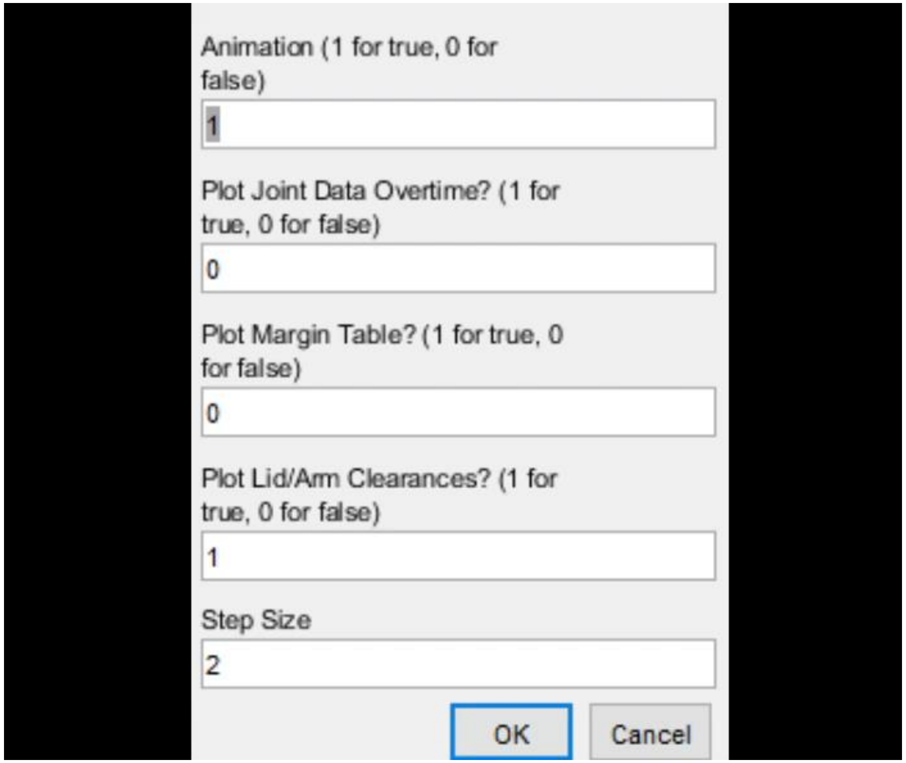


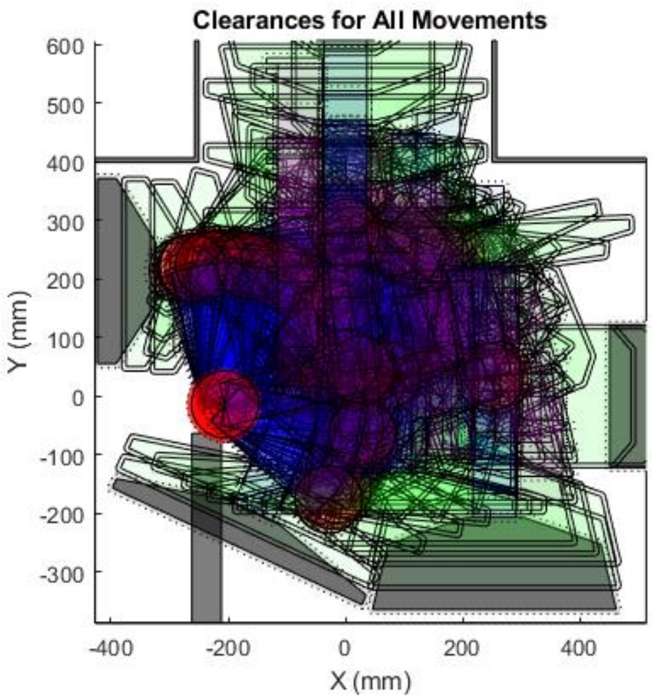
RTA Kinematics Simulation



Program Features

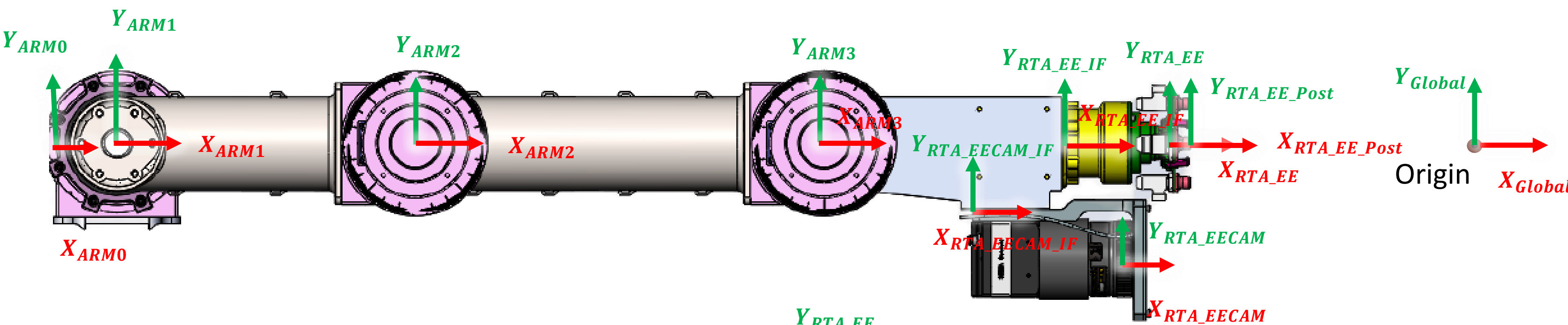
- Forward and Inverse Kinematic simulation of 3DoF Planar RTA
- Specific poses, linear path planning, step size, animation, elbow transitions
- Calculates joint torques
  - Assuming arm moves slowly; static analysis
- Clearance checks with NTE Volumes
- Optimizing script to decrease link length and decrease joint torque
  - Parameter Search



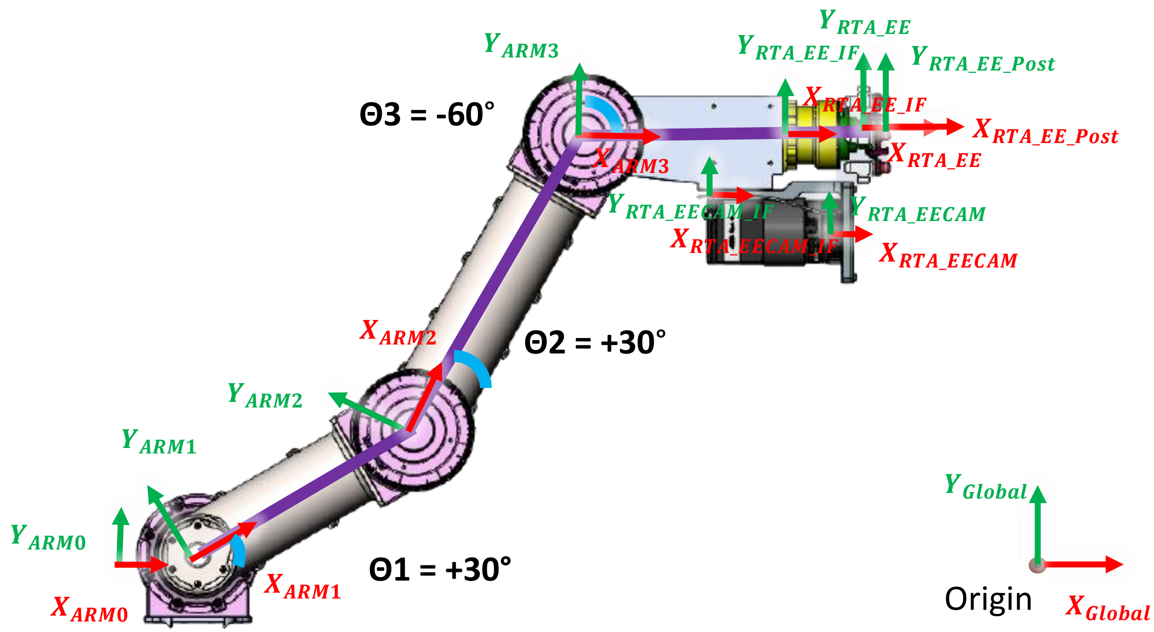


Arm Frames

ZERO POSE



RANDOM POSE



Ψ = Θ1 + Θ2 + Θ3 = 0° = EE rotation

Θ represents local rotation
ψ represents global rotation

MATLAB Joint Optimization

- Created a cost function that seeks to minimize total link length, and ultimately find the lowest torque generated
- Brute force, optimization

INPUTS: Length of Link 1, Link 2, Link 3 as well as Joint 1 X,Y position
OUTPUTS: Configuration with lowest total torque

PSUEDO CODE

```
For Link1 Length Bounds:
    For Link2 Length Bounds:
        For Link3 Length Bounds:
            For J1 X position Bounds:
                For J1 Y Position Bounds:
            if: iterations L1, L2, L3, X1, Y1 can meet the main kinematic frames, store this combination and the sum of link lengths
            else: record the combination and move onto next iteration
```

→ For solutions that close, choose lowest torque out of the available results

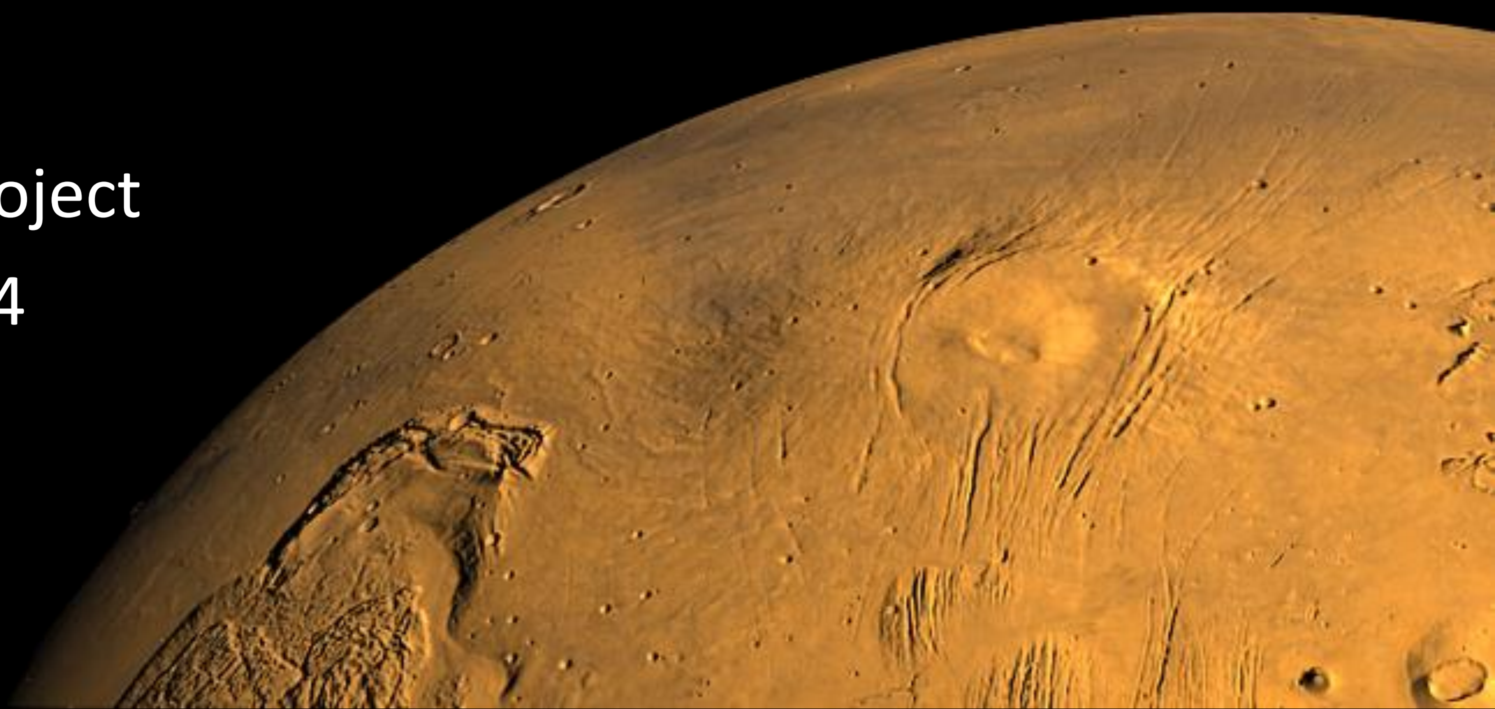
Results:

- Took a day to run (using multithreading) but ultimately worked!
- Optimized link lengths informed 3D printed RTA design

Chat Controlled Twitch Robot

Personal Project

Winter 2024



Objective

Background:

- In 2014, Twitch Plays Pokemon was a popular streaming channel where users completed the game through Twitch Chat commands
- My personal objective was to create a robot that streams itself and is remotely controllable through Twitch Chat

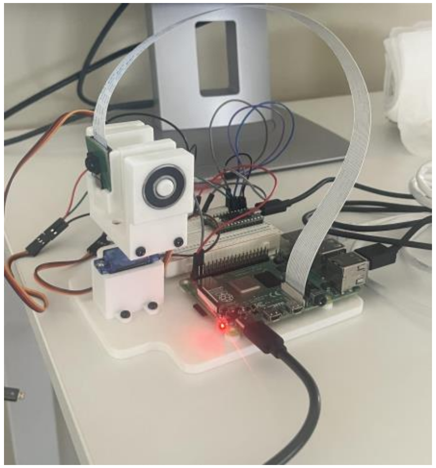
Twitch Plays Pokemon

Requirements:

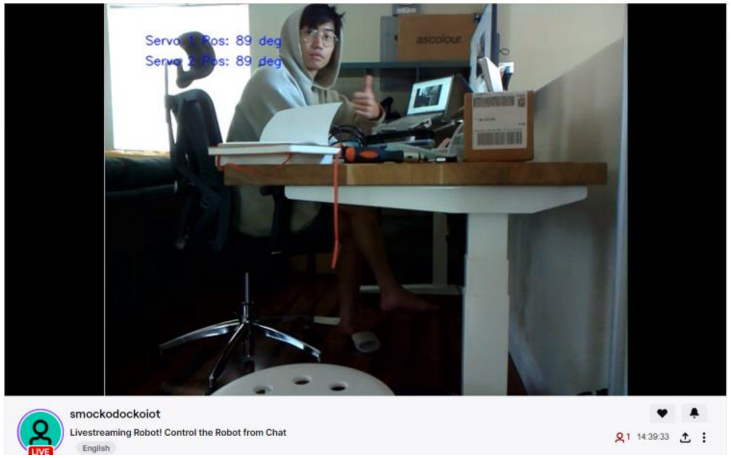
- Fully autonomous (stand-alone system)
- 2DoF camera control (pan/tilt)
- Chat integration
- Permanently streaming

Technologies Used:

- ROS2
- MicroROS
- Teensy / Raspberry Pi / Servos
- Arduino C/C++
- Networking (UART, SSH)
- OpenCV, Video4Linux (V4L)
- Linux (Ubuntu)



Twitch Robot Setup



Live Twitch Stream

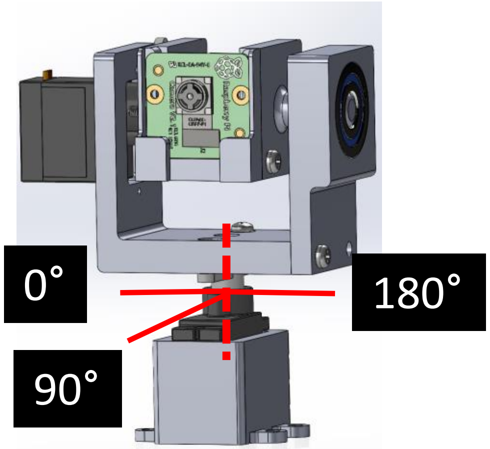
[Twitch Stream Link](#)

Demonstration Video

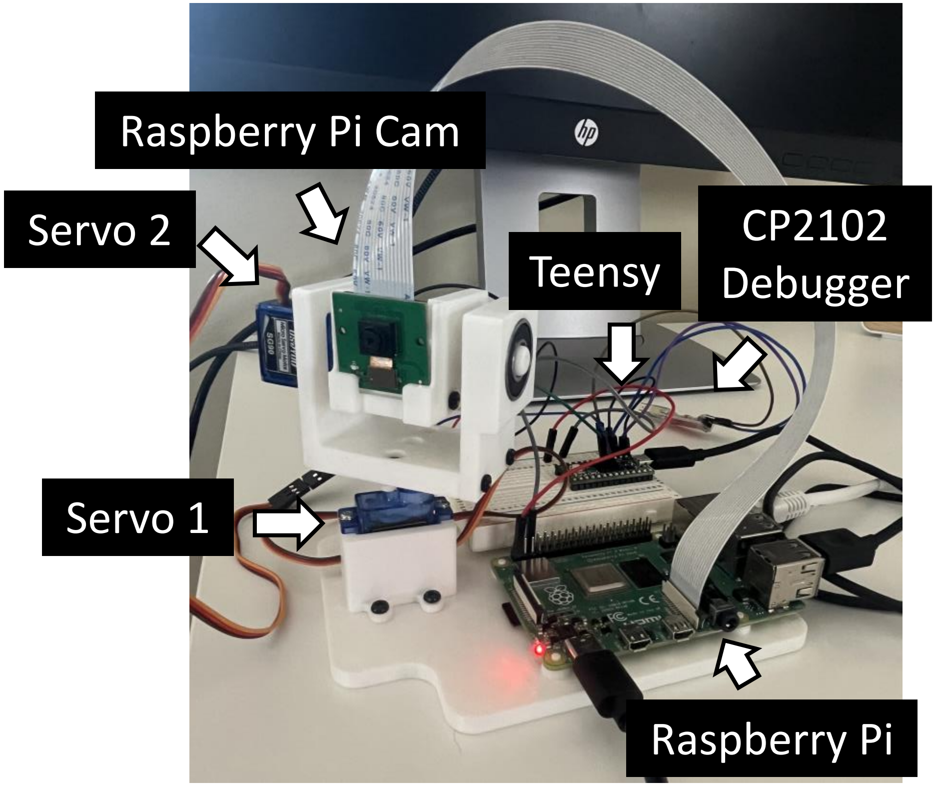
Twitch Robot Demonstration

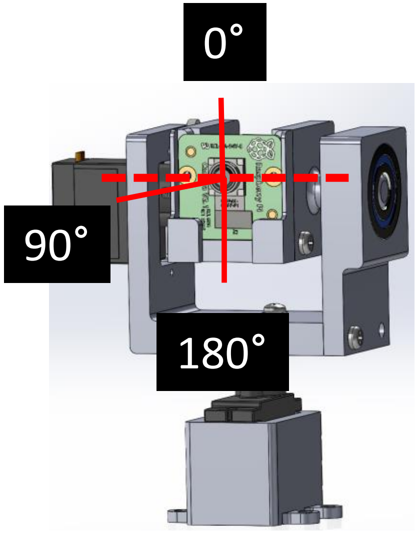
Mechanical Design

- Simple Pan-Tilt Camera Design
    - SG90 Servos use 5V from Raspberry Pi (convenience)
- 2 Degrees of Freedom (Pan, Tilt) that go from 0-180deg
- Camera works with Raspberry Pi by using Video4Linux (V4L) drivers / ROS2 package sourced from online



Servo 1 Axis

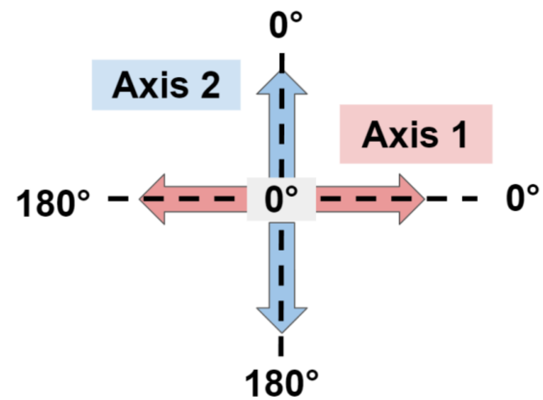




Servo 2 Axis

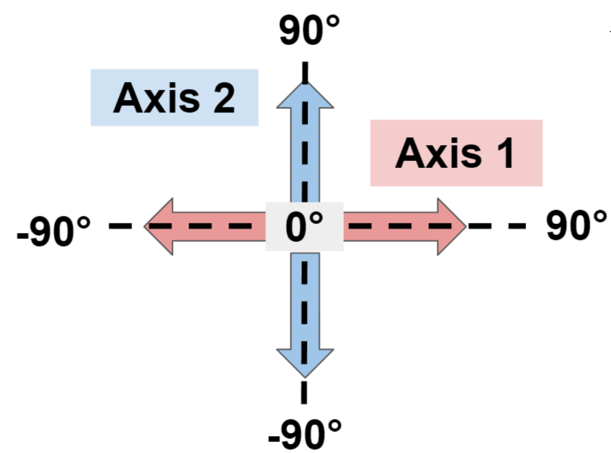
Axis Orientation

- Objective is to map the coordinates that the viewer sees, to the actual coordinates of the servo motors
- Makes for intuitive user experience as a centered position is [0,0]



Servo-based Definition

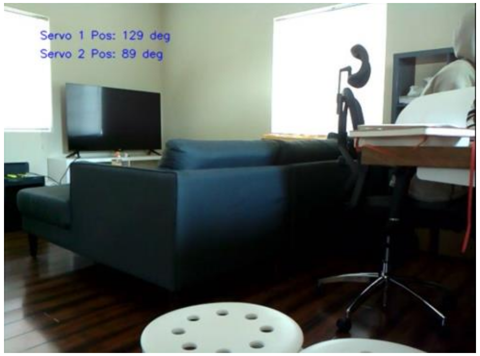
90-X



Camera Frame Definition

High Level CONOPS

1. Live Stream is Started

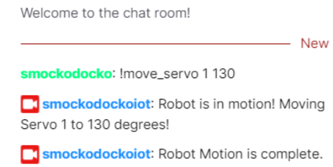


2. User Inputs Chat Command

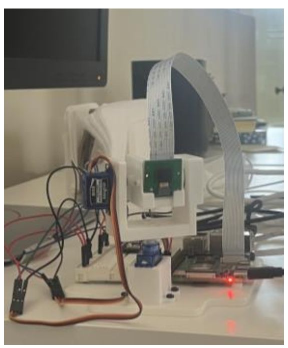
Command: !move_servo <servo_num> <servo_pos>

3. Message is parsed by Chat Bot for:

1. Servo Number (servo_num, [1 or 2])
2. Servo Position (servo_pos, [0 to 180°])

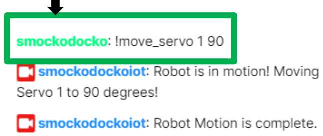


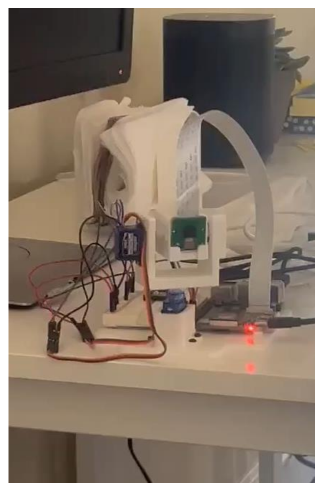
Chat message



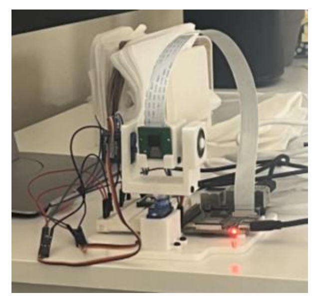
Start: Servo 1 Pos = 130°

Chat message sent



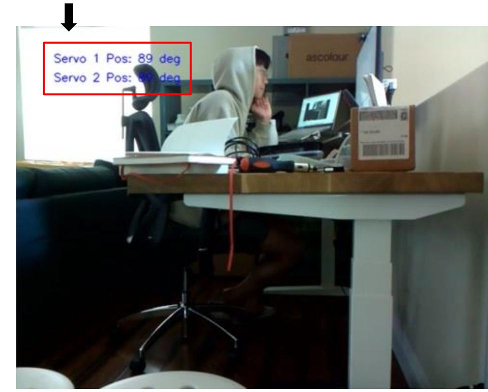


Movement from 130° to 90°



End: Servo 1 Pos = 90°

Position Updated



4. Command is sent to servo via. Teensy

5. Servo moves to new position

6. Teensy reports motion is complete

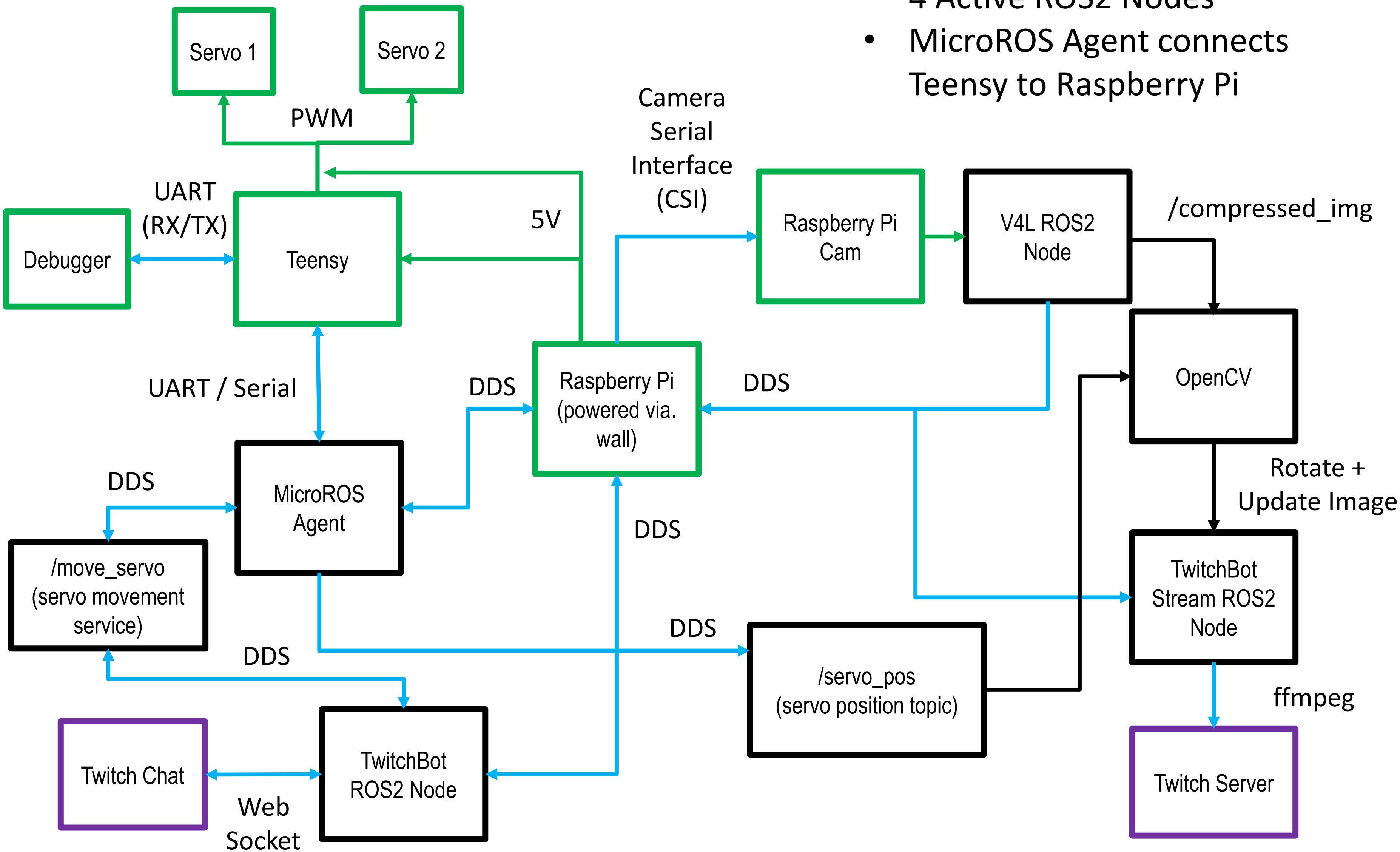
7. Chat bot is open to new commands

8. Camera image to stream is updated (delay exists)

System Diagram

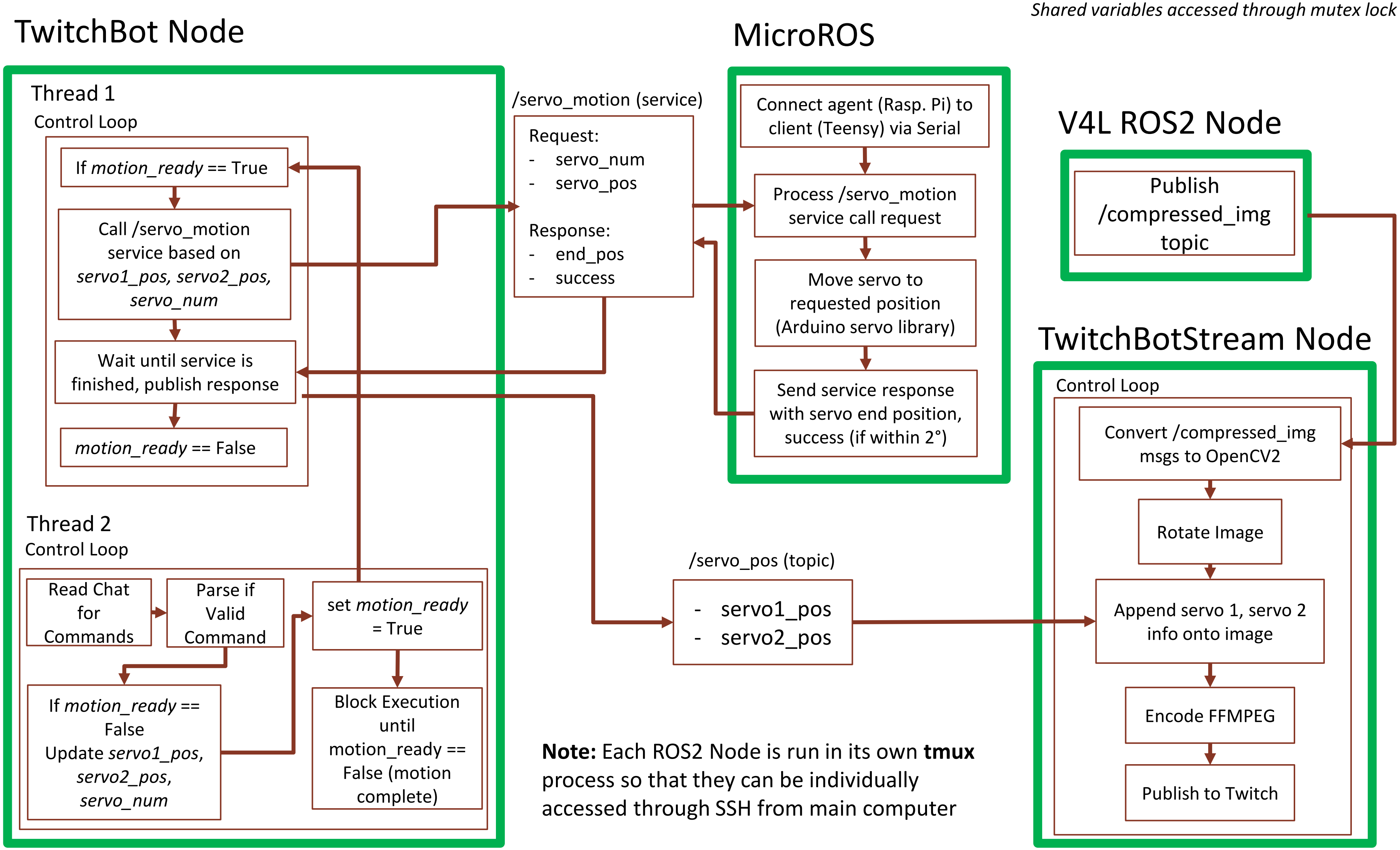
Hardware | Networking / Interface | ROS2 | Twitch

- 4 Active ROS2 Nodes
- MicroROS Agent connects Teensy to Raspberry Pi



ROS Architecture

Shared variables accessed through mutex lock



Note: Each ROS2 Node is run in its own **tmux** process so that they can be individually accessed through SSH from main computer

Project Summary

Summary

- Twitch Bot is currently working as expected and is accessible on Twitch!
- Has been tested to run for a week straight without any disruptions or network drops
- Latency mainly depends on user's internet speed (Twitch App works best)

Potential Improvements:

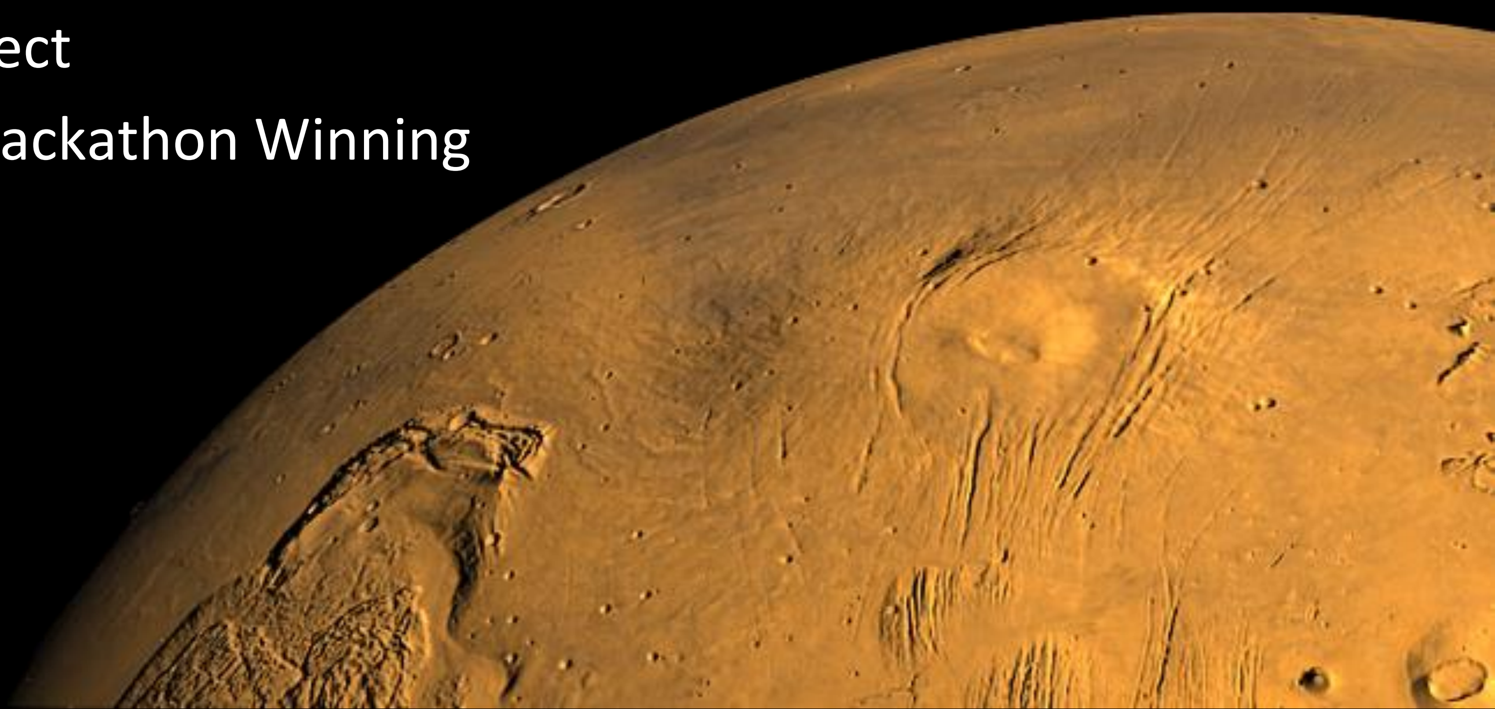
- Improve internet connection for higher quality upload speeds on Raspberry Pi
- Add additional commands to move to specific waypoints (i.e. !kitchen, !couch)
- Improve clarity of command arguments (add diagrams to the stream)
- Improve mechanical setup (higher quality servos)

MedMate

Personal Project

Flowers IoT Hackathon Winning Submission

Fall 2020

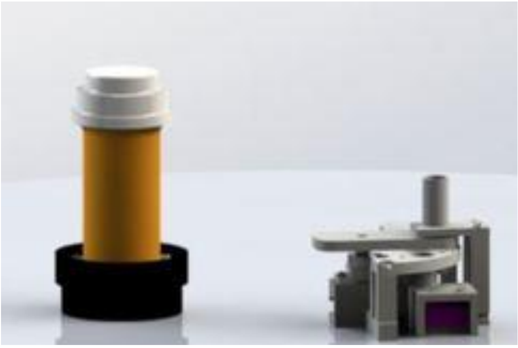


MedMate Description

Flower Invention Studio IoT Hackathon Objective: Design a custom IoT Prototype within two weeks to solve a home automation task

Solution:

- MedMate, a device that helps patients and caretakers monitor medicine intake
  Two Forms: pill bottle monitor and a pill dispenser
- MedMate tracks:
  1. when a user should take their prescription
  2. senses when a user has taken their medicine, and then
  3. logs this information in a database that is presented on a webpage
- Winning submission for Flowers Invention Studio IoT Hackathon
  - Worked with one other partner who focused primarily on web development
  - I designed the product, state machines, device code
  - Completed remotely!



Pill Bottle Monitor **Pill Dispenser**

References

- Hackathon Submission Link (with video)
- GitHub Link

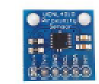
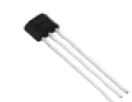
MedMate demonstration Video

Hardware/Software Used

Tools Used:
- Ender3 Printer (personal printer)
- Solder

Software Used:
- Arduino C (C/C++)
- Eclipse Paho MQTT Python client (Python)
- Cloud Firestore (Google Firebase)
- React (JS)

Protocols:
- MQTT (Message Queuing Telemetry Transport)
- I2C (Inter-Integrated Circuit)

Hardware	Images
ESP32 IoT Microcontroller	
VCNL4010 Proximity Sensor	
DRV5053 Analog-Bipolar Hall Effect Sensor	
Tower Pro SG90	
5V Power Supply	
Breadboard, 220 Ohm Resistors, Multicolor LED	

Project Motivation

Motivation: Help my Dad and caregiver track medications by:

- reminding patients to take medicine
- allow caregiver to monitor intake history

Design Philosophy:

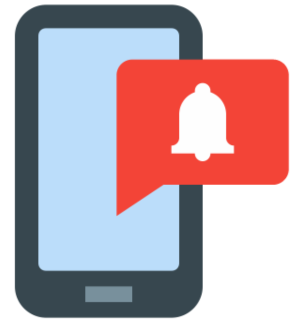
1. Low profile
2. Cheap
3. Simple hardware
4. Simple user interface
5. Interfaces with any standard pill bottle

ESP32 was our chosen controller because it is configurable with the Arduino client and has full IoT functionality. The chosen network protocol was MQTT, as it is designed for simple communication between a controller and host server.

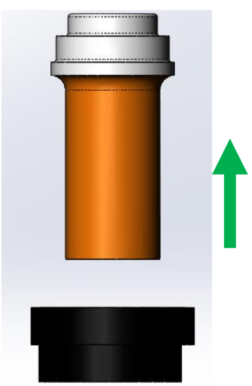
MedMate Pill Bottle Monitor Storyboard

1. User enters their prescription information

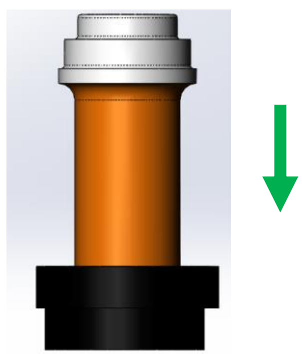
2. Server keeps track of when user should take medicine



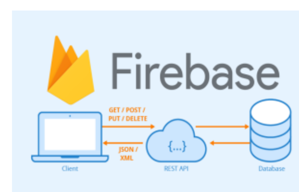
3. User is notified to take medicine



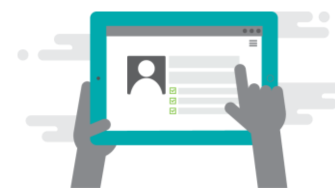
4. Pill bottle is taken off MedMate



5. Pills are consumed and bottle is put back on MedMate



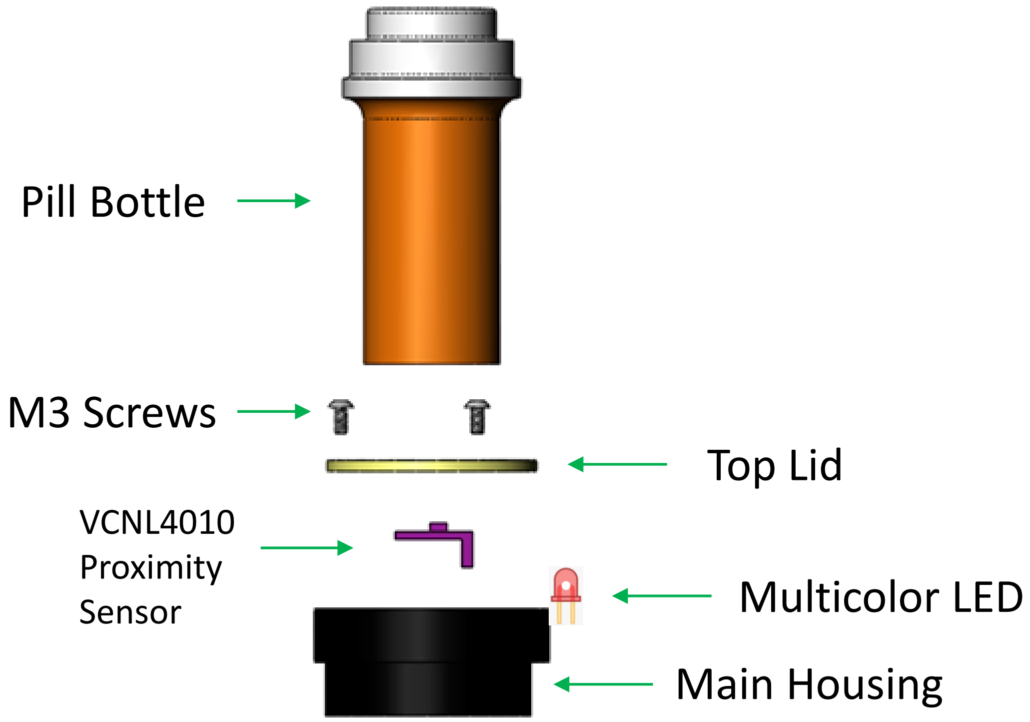
6. Time of Intake is uploaded to Database

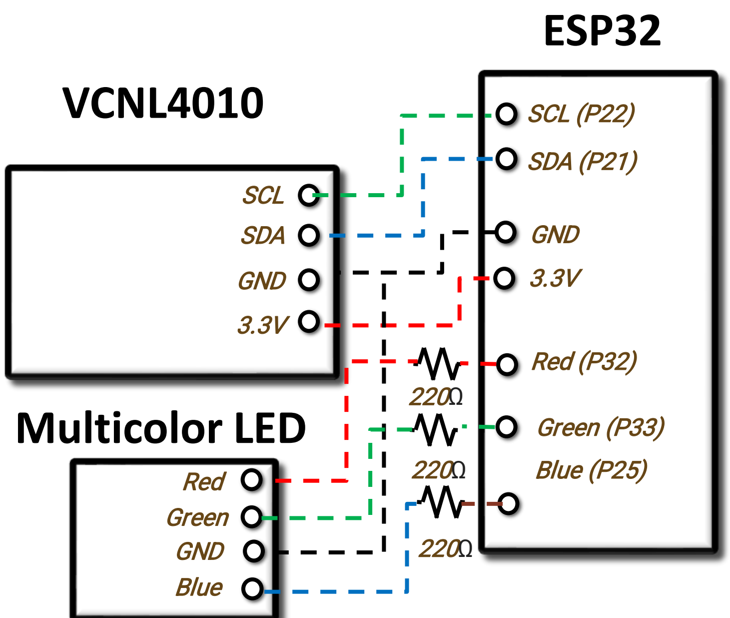


7. Patient History shown on Webpage

Final Prototype (Pill Bottle Monitor)

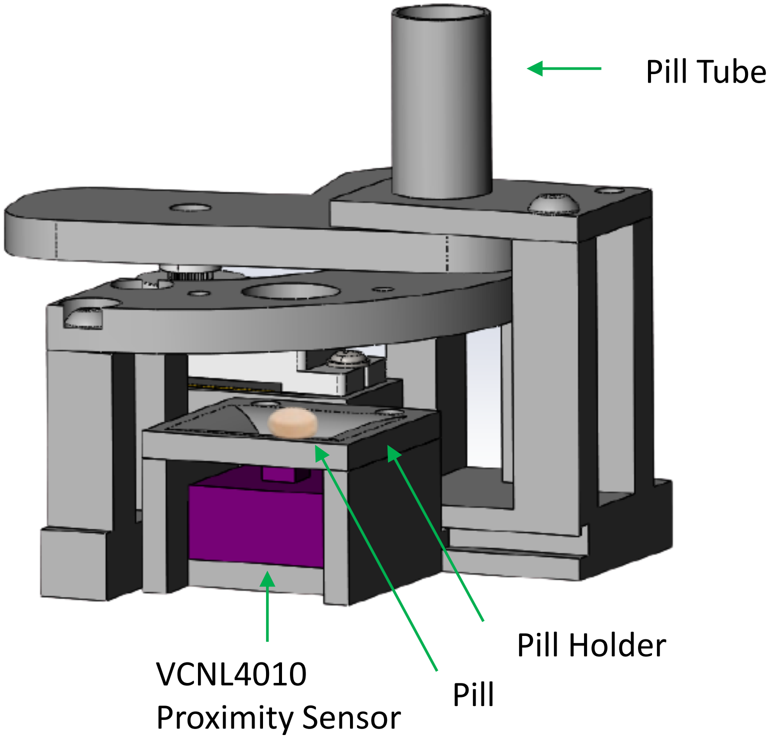
- Uses Proximity Sensor to sense pill bottle presence
- Added LED for debugging/clarity
- Fits common pill bottle diameters

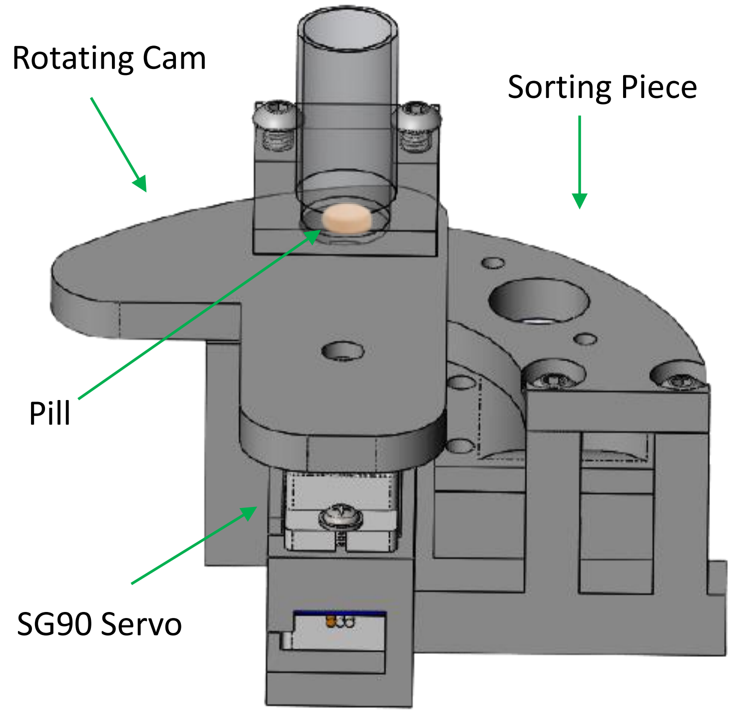




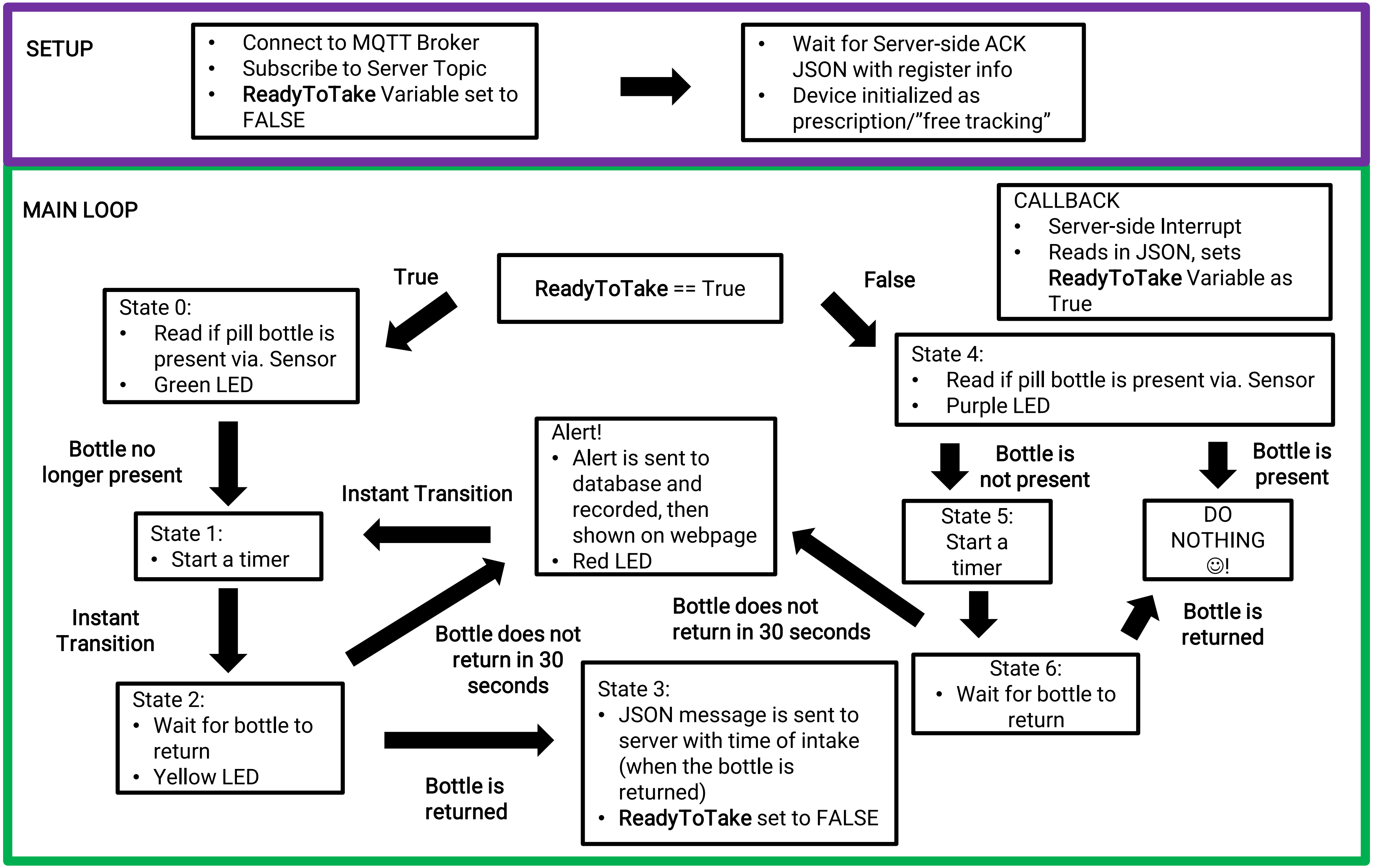
Final Prototype (Pill Dispenser)

- Actuates using servo and rotating cam to release pills at medication time
  - Controls how much medicine the user can take.
- Still uses proximity sensor to detect if pill has been taken
- Can be part of a larger assembly – designed for demonstration purposes





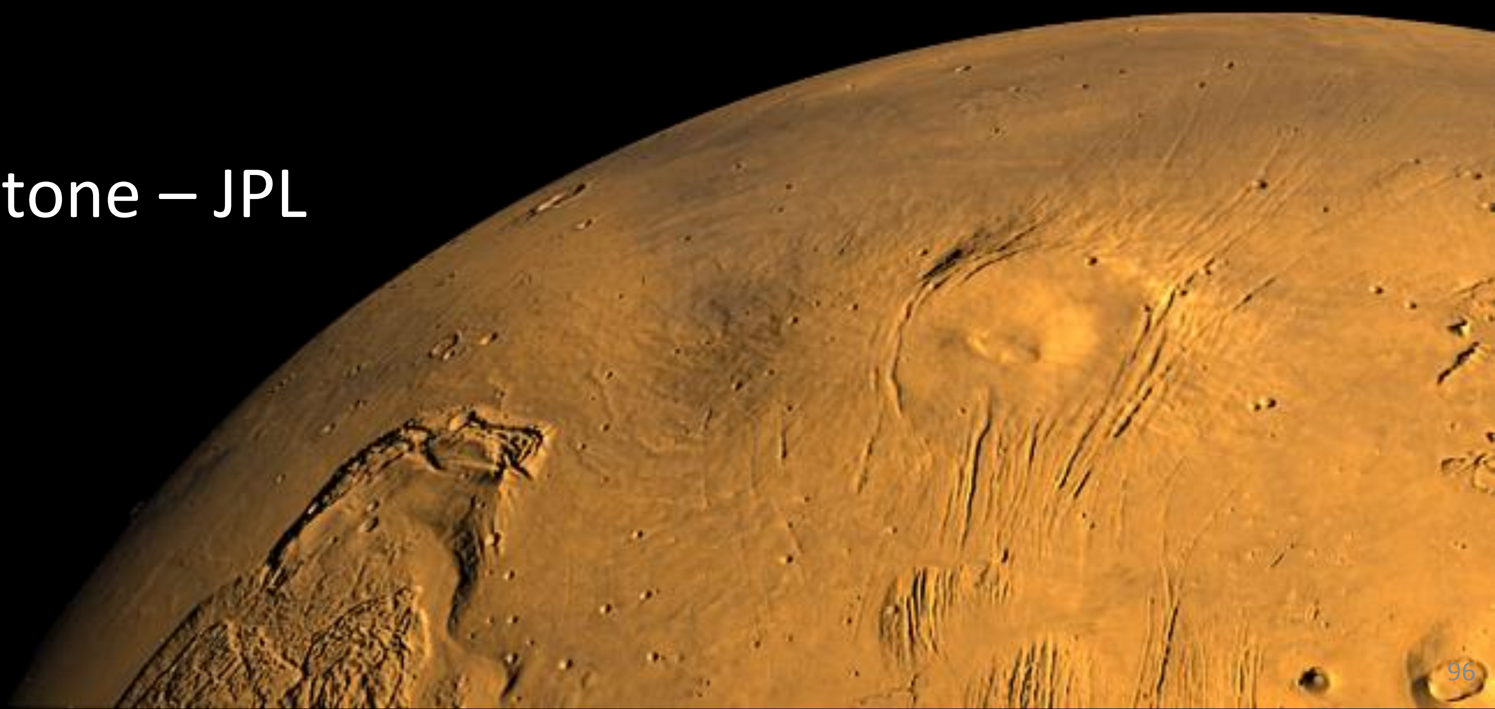
MedMate Pill Bottle Monitor State Machine



Exobiology Extant Life Surveyor Robot Sampling System

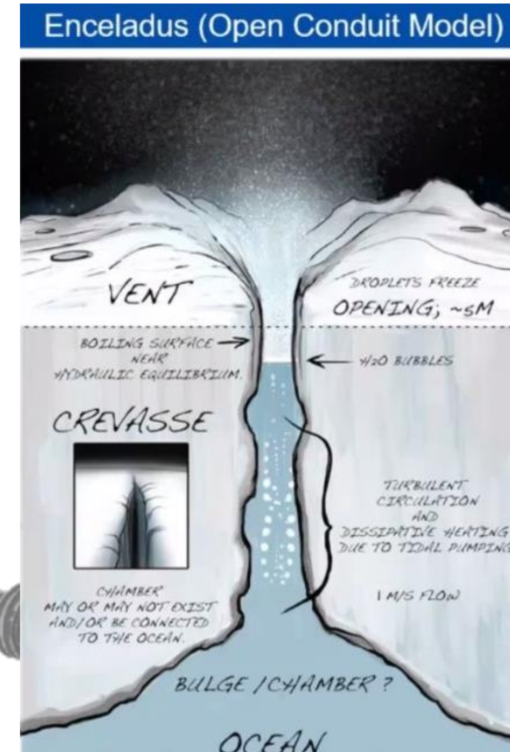
Senior Capstone – JPL

Fall 2021



Background & Problem Statement

- Exobiology Extant Life Surveyor (EELS)
  - Bio-inspired snake-like robot
  - Traverse Enceladus’ icy ocean-world terrain to **search for life**
- Front nose segment must be designed to:
  - Collect **sub-glacial liquid samples**
  - Gather **environmental data**
  - Travel over icy terrain and underwater
- Impact
  - Understand factors contributing to **melting glacial icecaps**
  - Explore glacial moulins and crevices traditionally **inaccessible to humans**
  - Determine conditions required to **sustain life**



Personal Contributions: Project Management, Mechatronics / Code Design, Literature Research

System Requirements

EELS System Integration

- Fits within Ø12cm X 50cm cylinder
- Withstands icy environmental conditions
  - -20 to 20°C
  - 0 to 150psi
  - 0 to 2m/s flow
- Withstands traversal through environment

Liquid Sampling System

- Acquires 2 separate 1L samples of liquid
- Sterilizable collection system
- Easily removable liquid samples
- Fill at rate of 0.5L/min

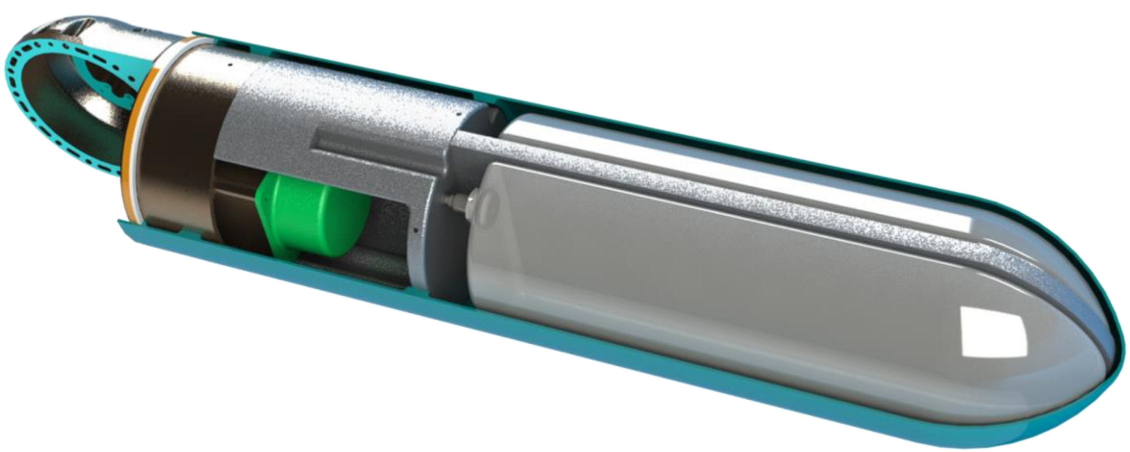
Sensor Data Collection

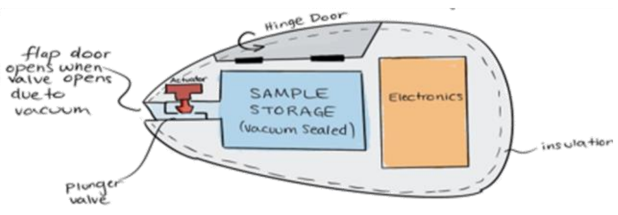
- Gather pressure, temperature, aquatic chemistry, and imagery data
- 0°C minimum sensor operating temperature
- -20°C minimum sensor storage temperature

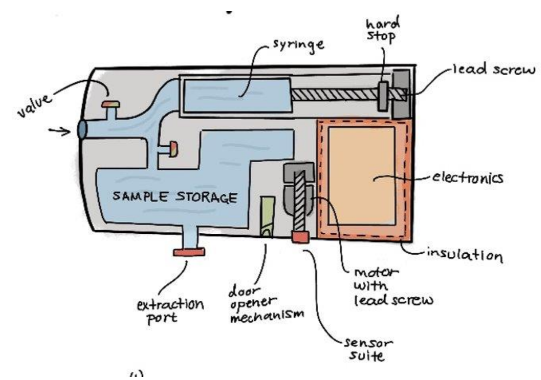
Preliminary Sketches & CAD

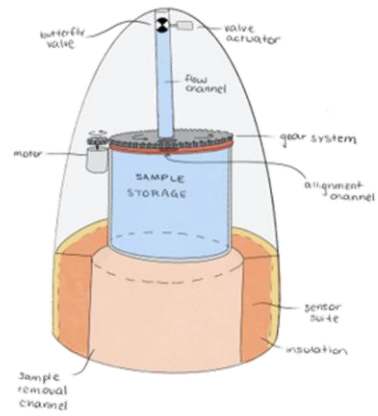
- Initial ideation created an outline of the system shape and major mechanical components

- Both **vacuum** and **pump-based** designs were considered

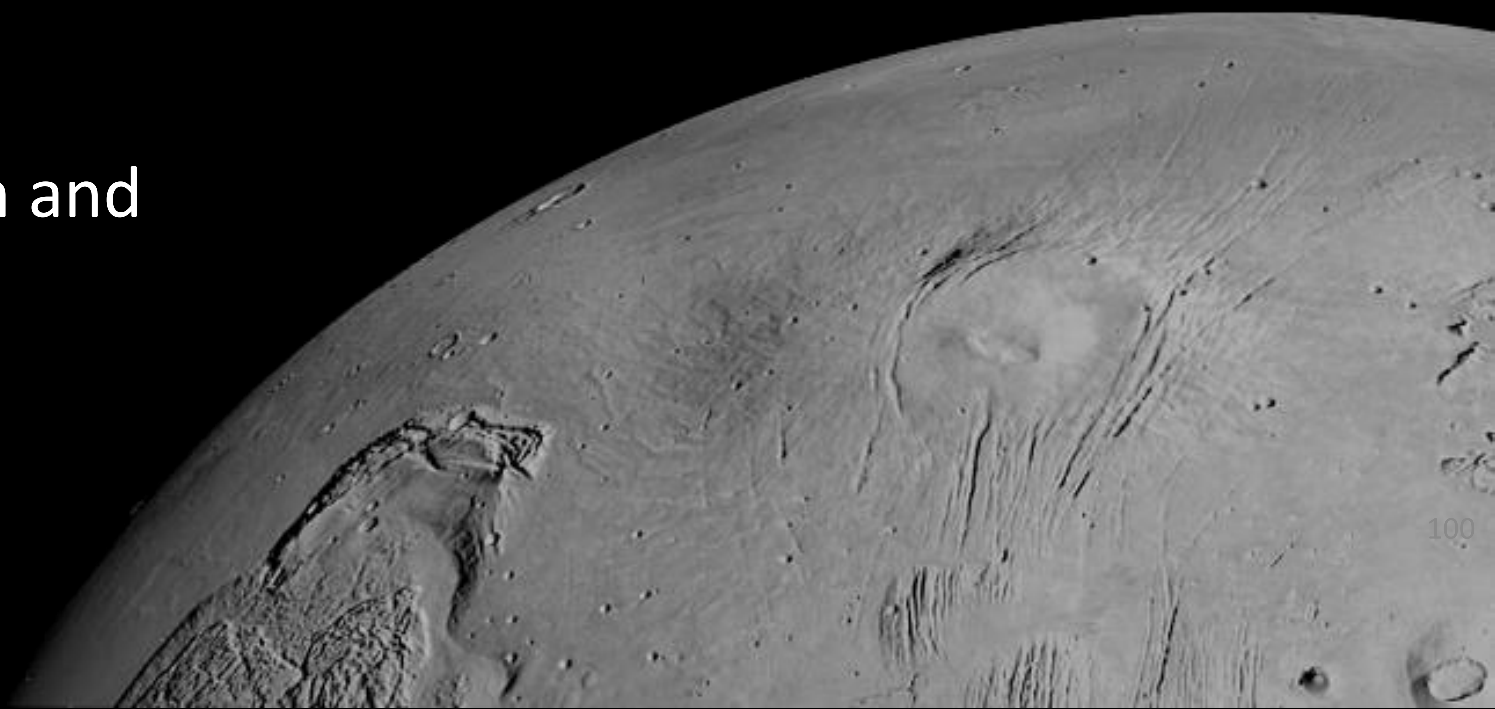




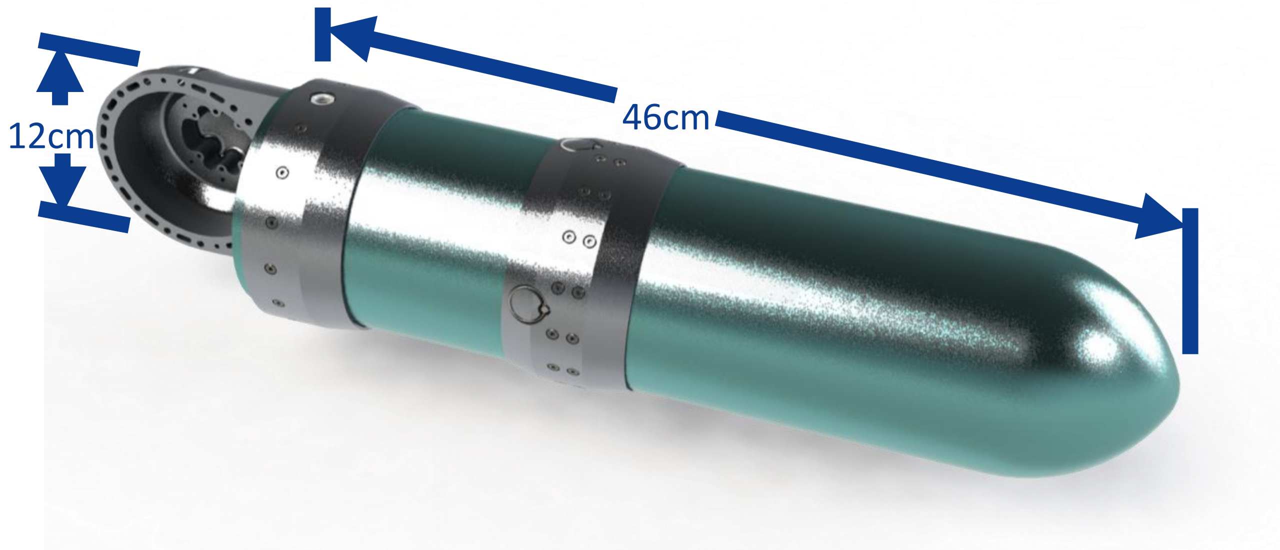


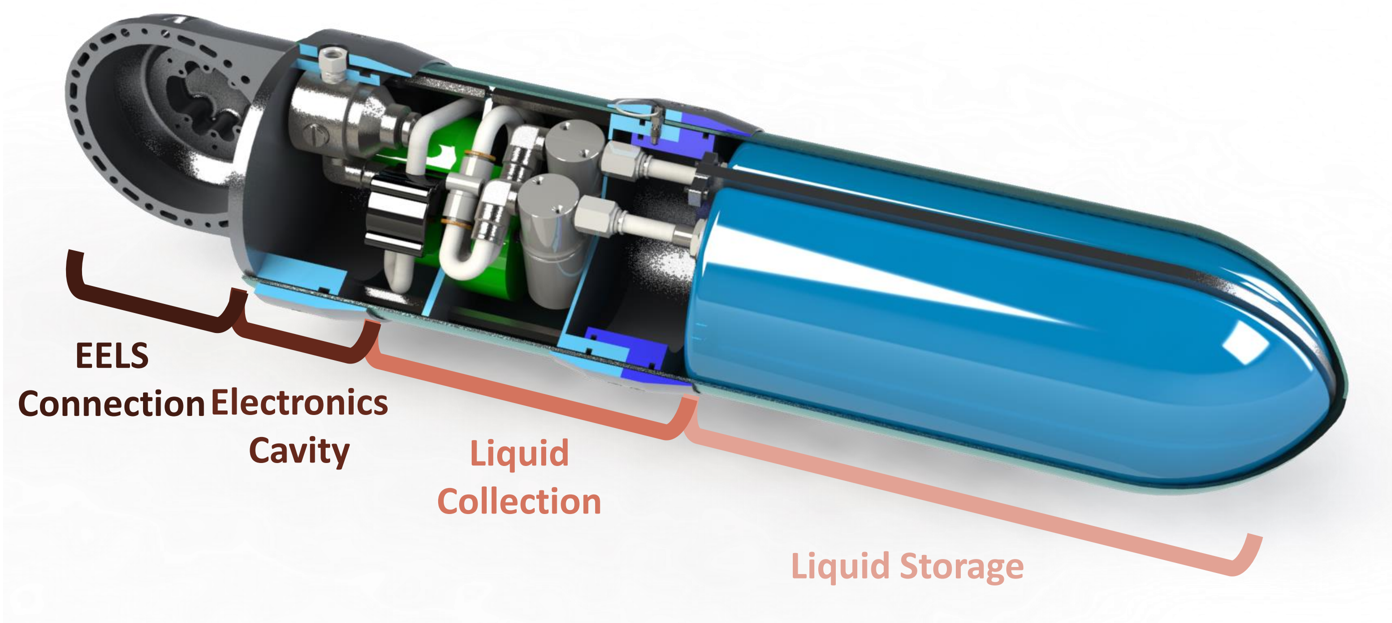


Final Design and Prototype

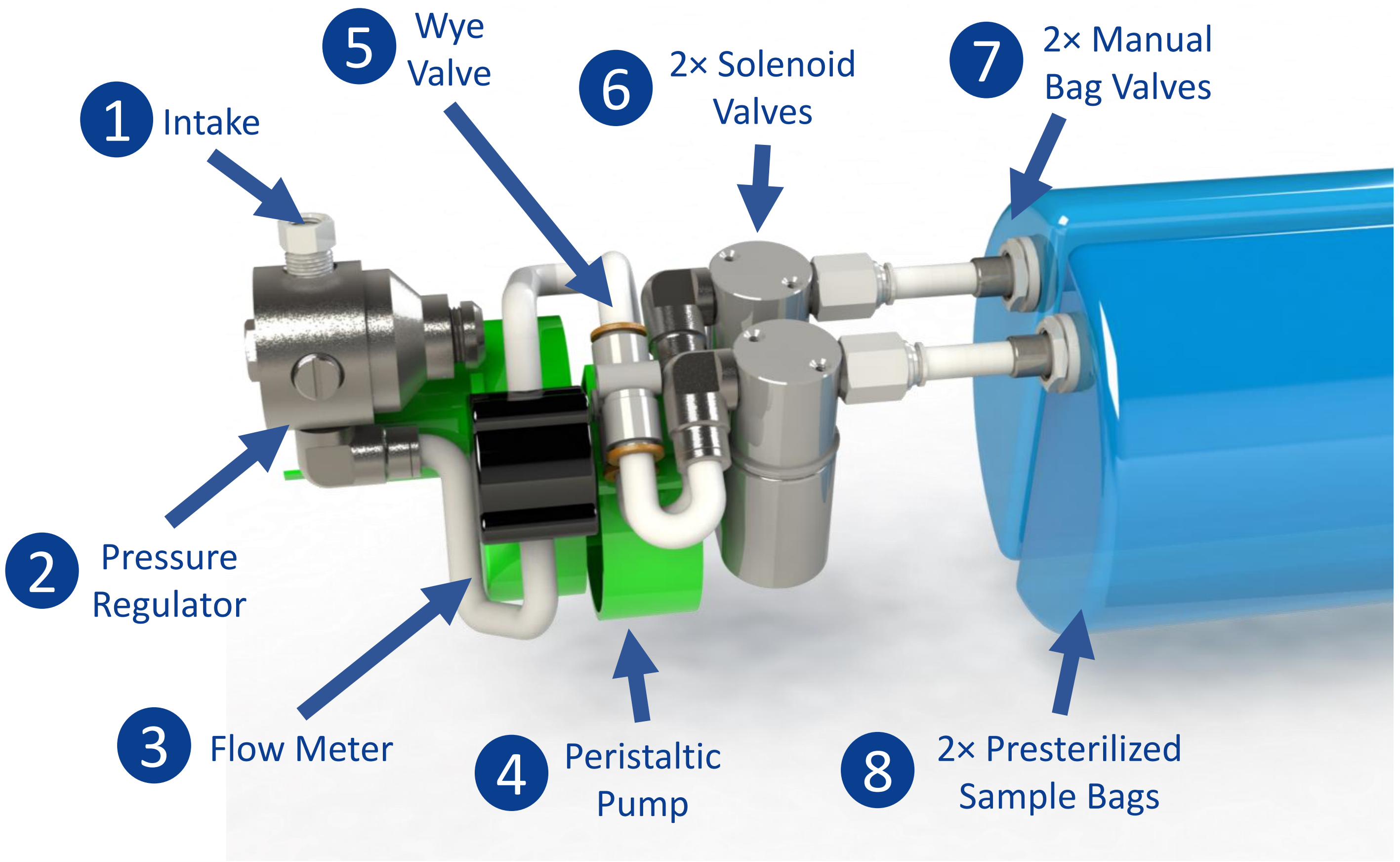


Mechanical: Overview

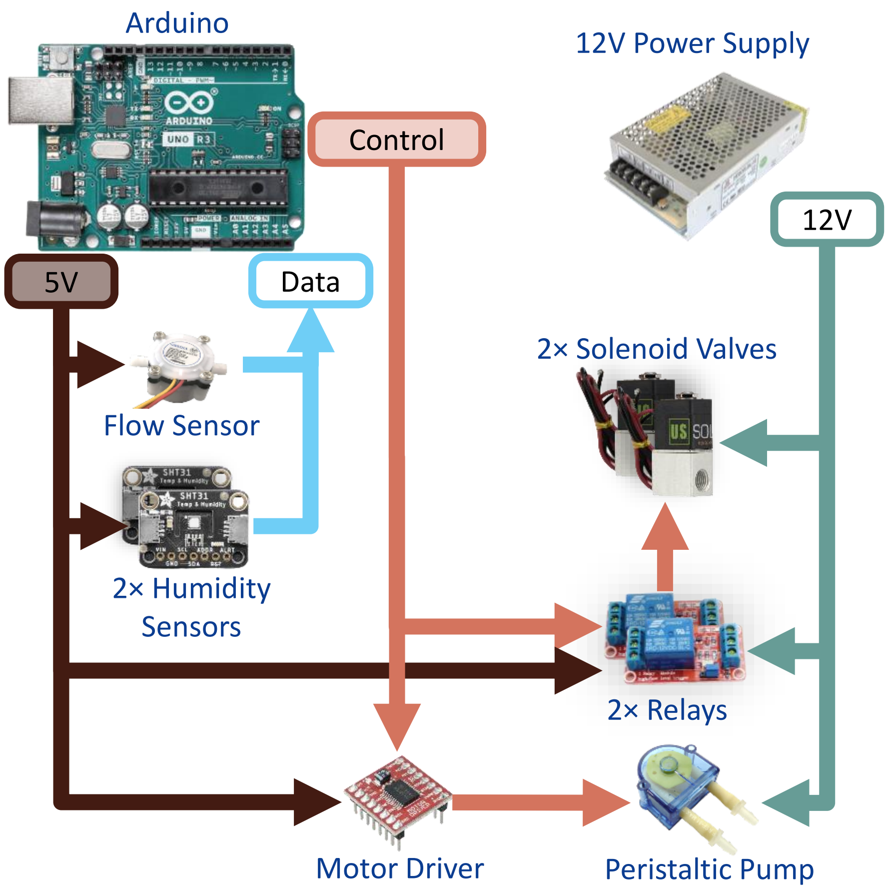




Mechanical: Liquid Collection

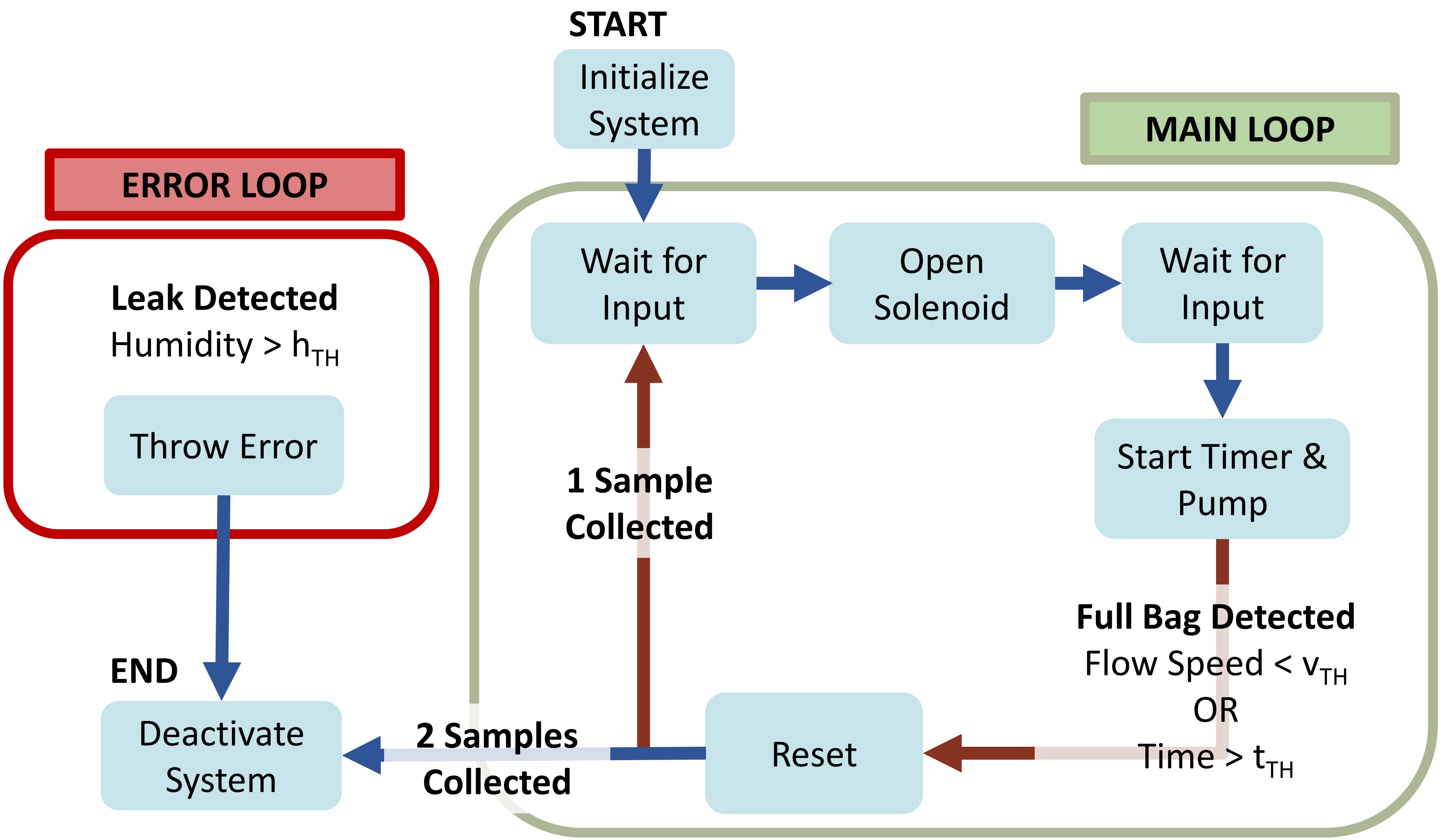


Electrical Overview

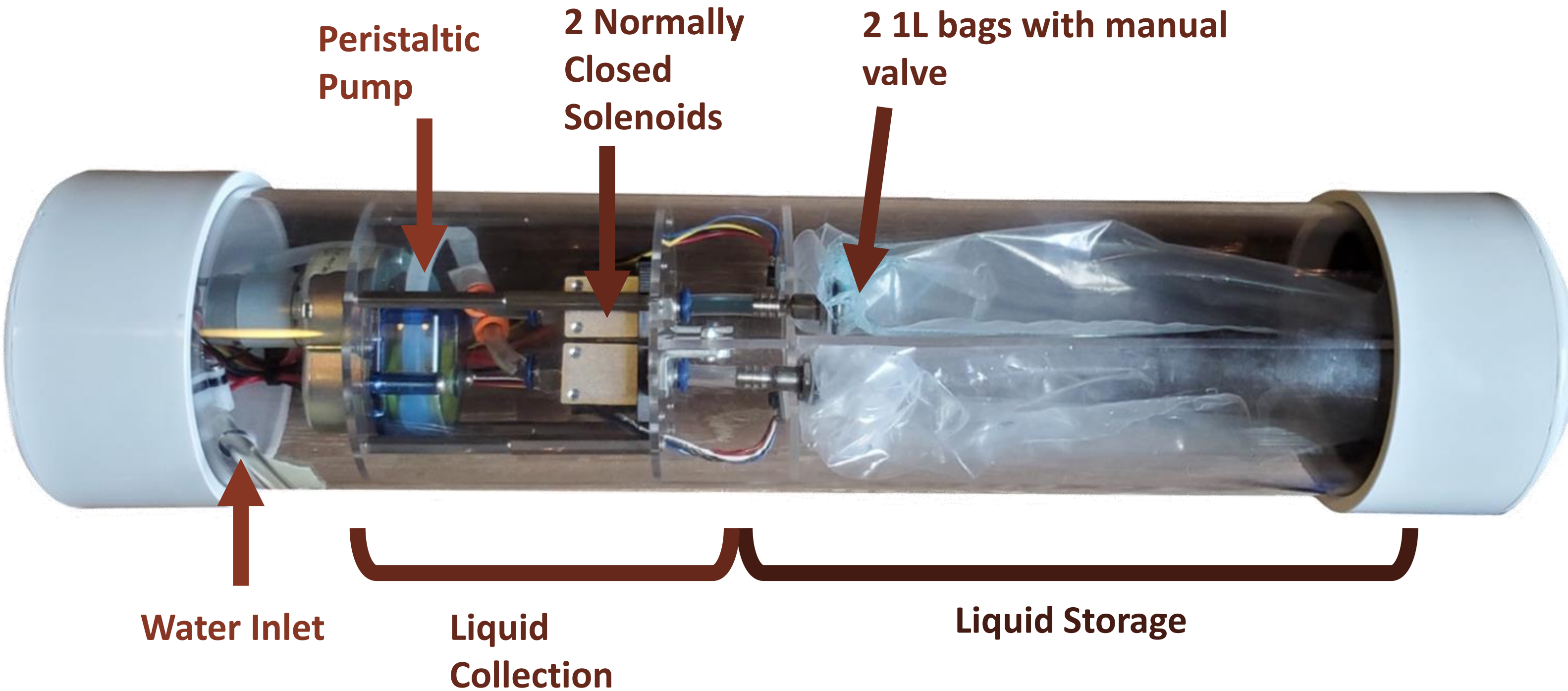


- Flow sensor to detect when bag is full
- Humidity sensor to detect leaks as small as 5mL
- Two solenoid valves to control the filling of bags separately

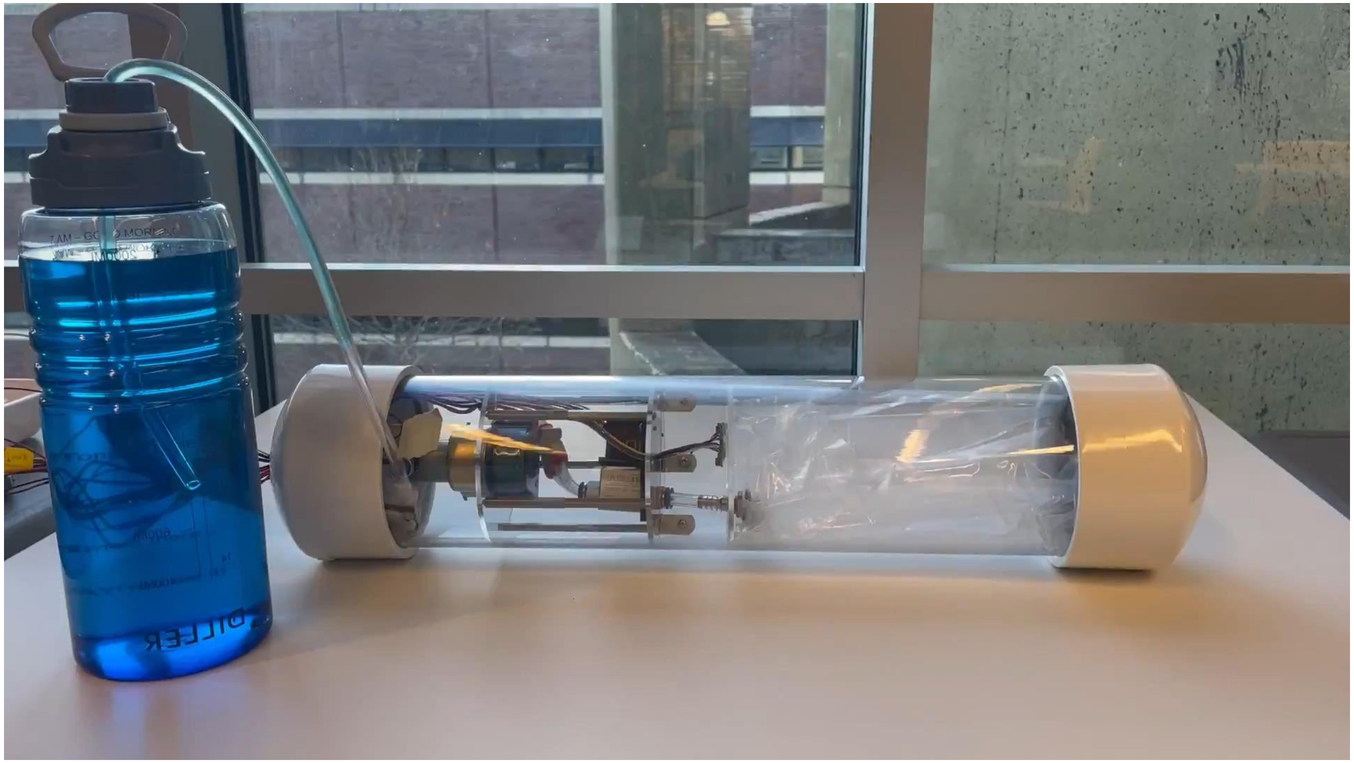
Software Overview



Final Prototype



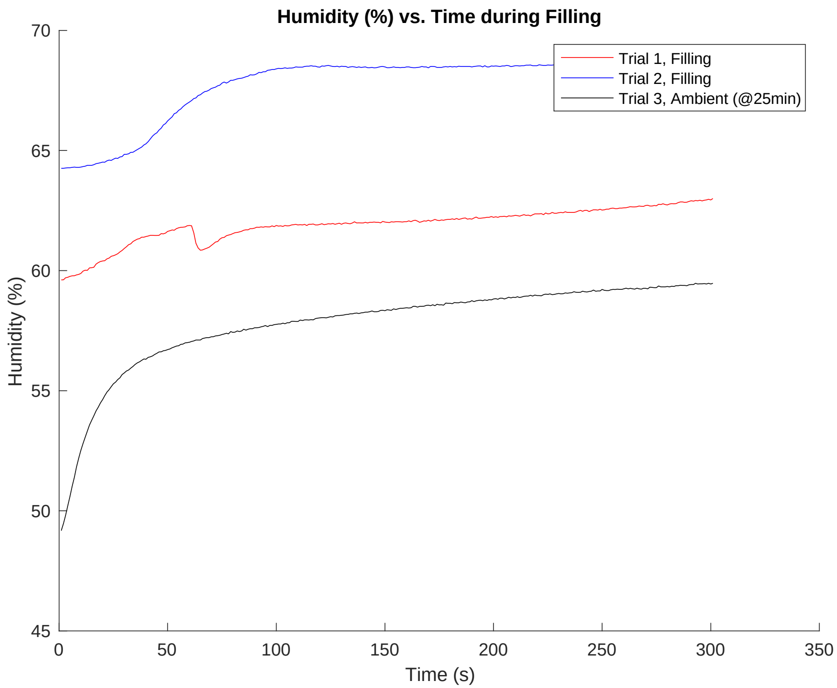
Final Prototype Video

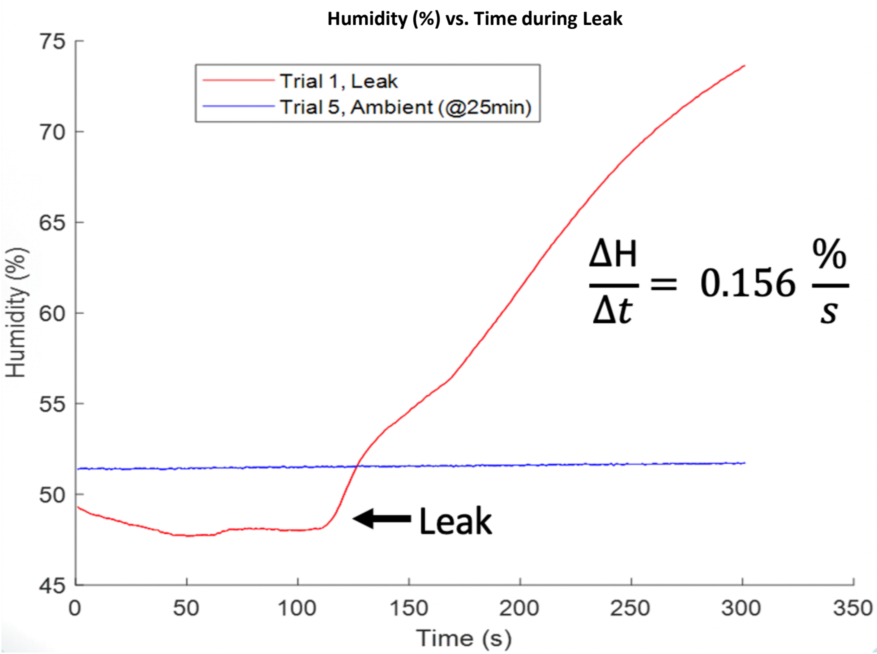


Fill Time: 2min 15s (135s)

Humidity Sensor Testing

- Conducted test trials with previous prototype to determine humidity sensor effectiveness in determining leaks
- Performed ambient, filling, and leak tests
- Rate of change of humidity can be used to determine if leaks occur





Conclusion

Future Work

- More refined prototype rated for Antarctic conditions
- Test while submerged underwater
  - Fit electronics into sampling system
  - Create custom PCB and choose a lower profile microcontroller
- Implement full system control using flow meter and humidity sensors
- Internal air pressurization

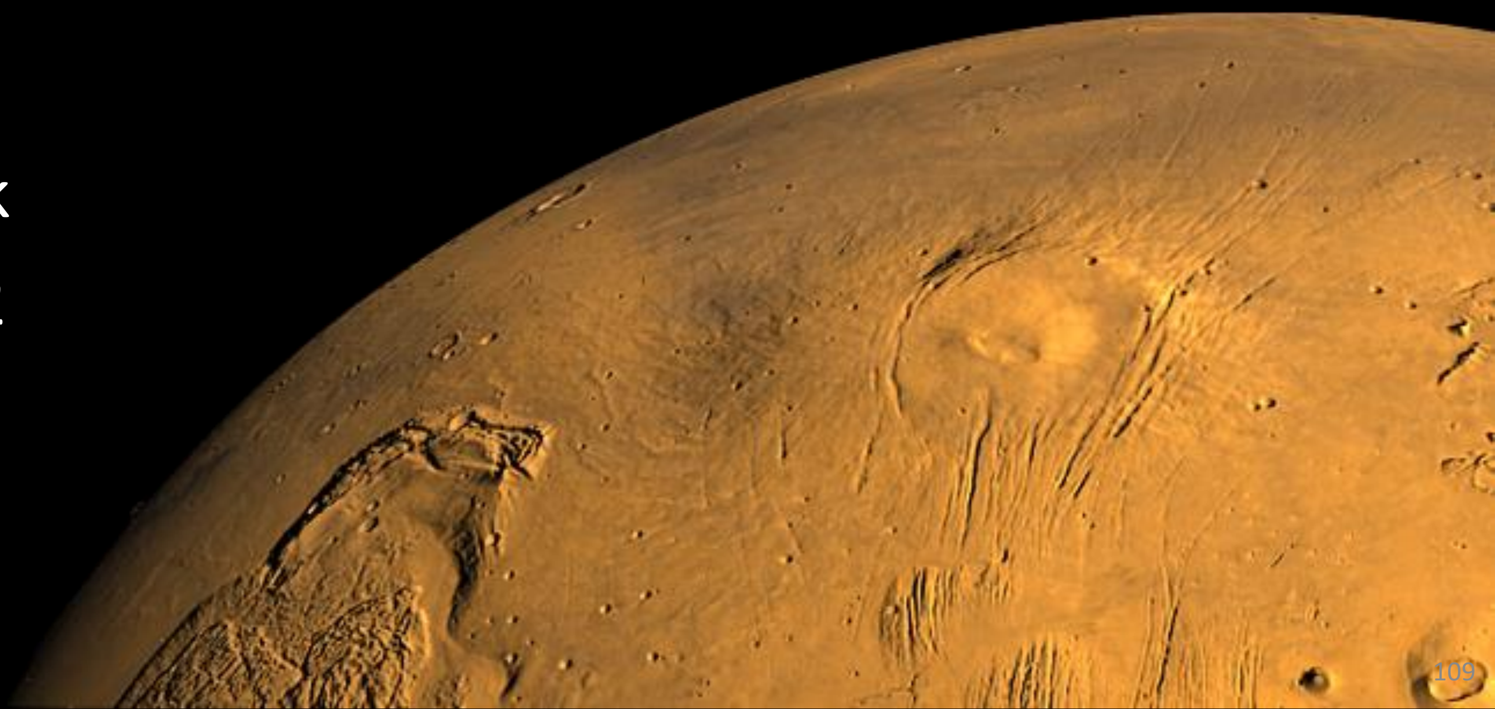
Impact

- Aid in ongoing efforts to understand **melting glacial icecaps** and **inaccessible** glacier moulins
- Provide data of the subglacial environment to **determine conditions suitable to sustain life**

TurtleBot ROS Demonstrations

Coursework

Spring 2022

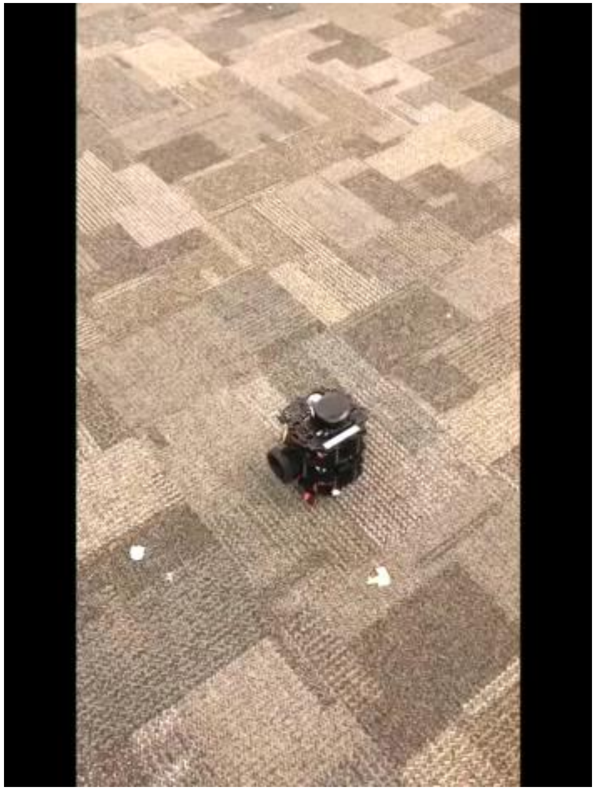


OpenCV Object Following with PID

- Robot to follow a specific object in space while maintaining object in center and at correct distance
  - Used LIDAR to determine distance to object, PID to maintain relative distance/angle
  - Used OpenCV to track object and maintain a specific orientation

Robot Navigation with Odometry/Lidar

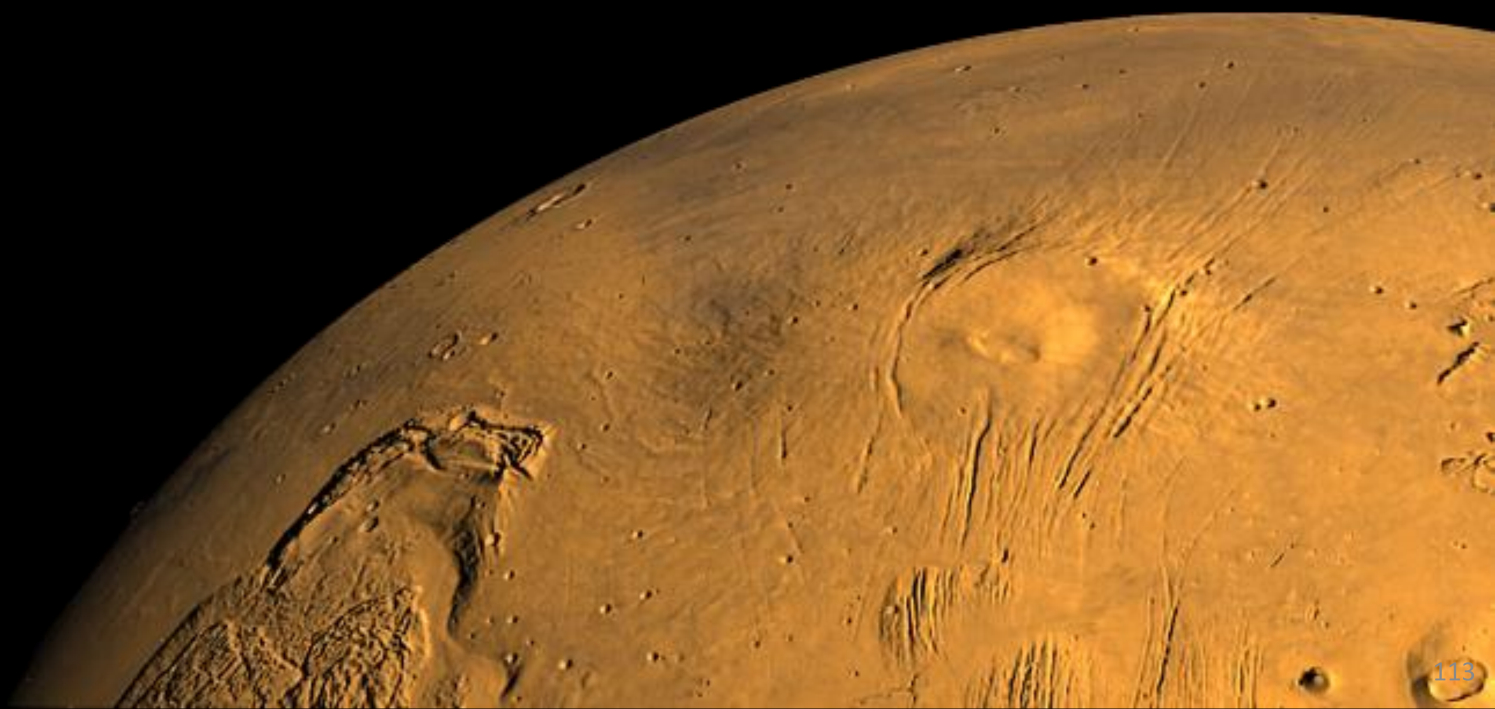
- Using robots onboard odometry and dead reckoning to navigate to various waypoints while avoiding random obstacles
  - Writing and subscribing to odometry nodes, robot kinematics
  - Filtering noisy lidar data
  - Obstacle detection with avoidance routine, while maintaining knowledge of position



Final Project: Sign Classification and Navigation to Goal

- Robot classifies 6 different signs to navigate towards an end goal
  - Used state machine to control robot behavior
    - Implemented behaviors if sign is not correctly classified or if robot is stuck
  - Trained and used KNN ML model to classify signs with image processing
  - LiDAR, filtering, odometry, PID, ROS, image processing, classification

Additional 3D Printed Projects



3D Printed Gearbox

- 9:1 Gear Ratio with 3D Printed Gears and COTS Bearings
- Demonstration of gear feasibility and design with 3D Printed parts
  - Skeleton Modeling