Design and Development of a CubeSat Robotic Arm for In-Orbit Inspection and Servicing

Alice Sukhostavskiy and Shealyn Miller
Stephen Kwok-Choon, William Albertini, Caleb Arbreton, AJ Presto, Cole Sterba
Cal Poly, San Luis Obispo

CAL POLY

Mission/Scope

- Vision:
  - To contribute to the sustainability and shared use of space
- Mission:
  - To produce a cost-effective method for in-orbit maintenance
- Values for the Mission:
  - Cost, usability, low complexity, ease of deployment and operation

Mission/Scope

- **Design and development of a four-degree-of-freedom CubeSat robotic arm featuring a series of revolute joints with a three-fingered end-effector contained within a 3U CubeSat for applications such as on-orbit inspection and servicing**
- **The robotic arm shall function with two sensors, a camera and a lidar module. Relevant sensor data is proposed to be streamed to the operator's location, with commands sent back to the robot controller**
- **A first iteration will require remote operation with future iteration aided by computer vision**

Concept of Operations

Complete Mission:

1. Deployment
2. Detumble
3. Orbital Corrections
4. Rendezvous with Satellite
5. Robotic Arm Deployment
6. Robotic Arm Performs Maintenance
7. Rendezvous with Home Satellite
8. CubeSat Charges at Home Satellite

Concept of Operations

Robotic Arm:

1. Deployment
2. Reposition
3. Maintenance
4. Reposition
5. Stow

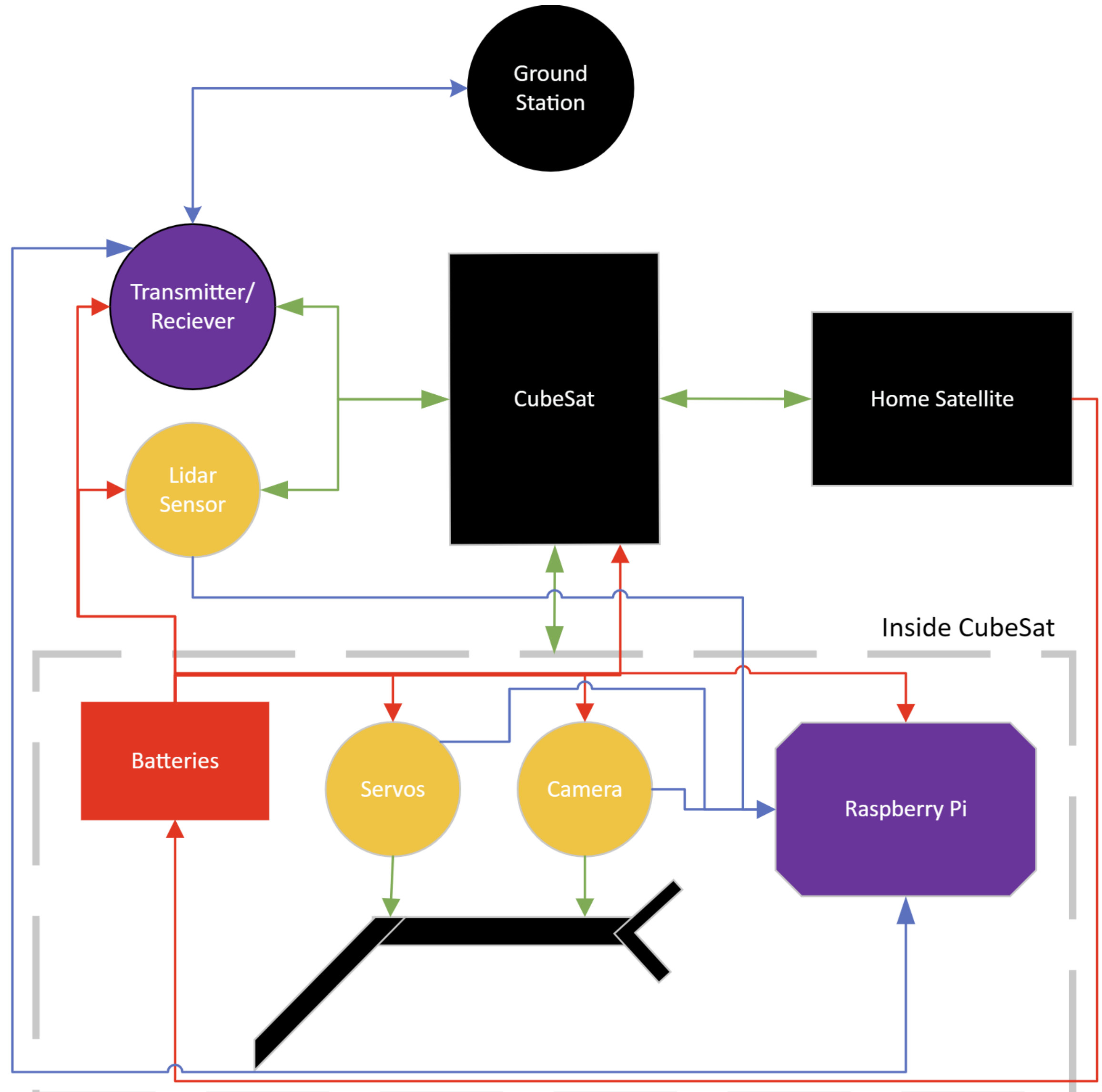
Robotic Arm Initial Design

- 4-5 degrees of freedom
  - Revolute shoulder and elbow joint for in plane translational motion (with possibility of third joint for out of plane motion)
  - Two revolute wrist joints for roll and yaw motion of end effector
- Lidar and camera aided remote operation
  - Lidar to help with coordinate position of object in workspace
  - Camera to help locate and grab object once close enough

System Architecture

Key:

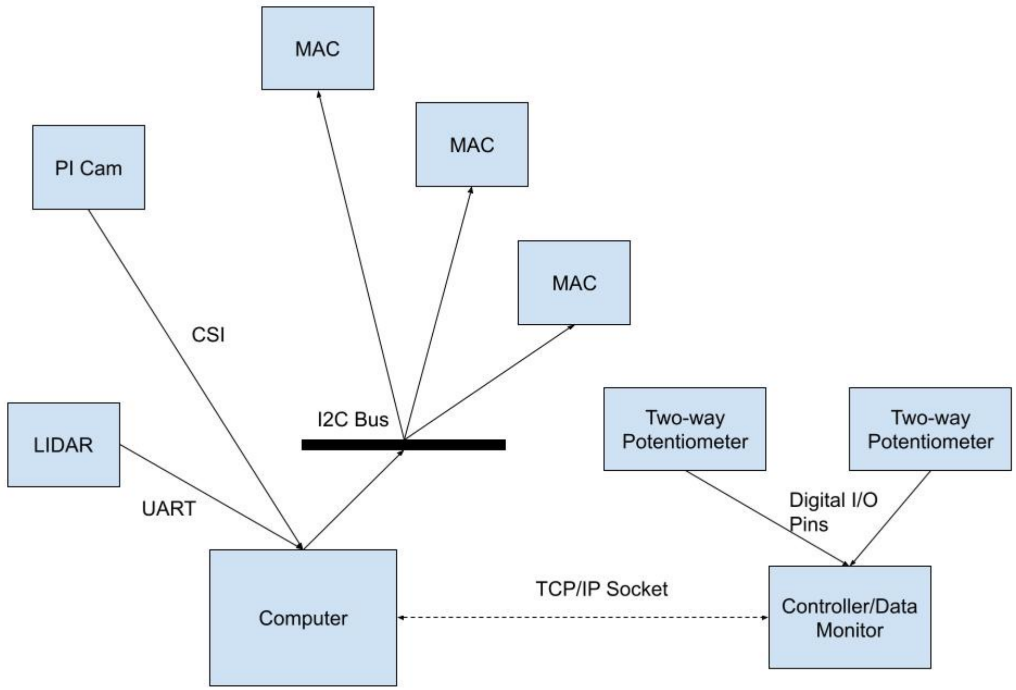
- Power Flow
- Data Flow
- Structural Connections
- Mechanisms Subsystem
- Controls Subsystem
- Power Subsystem
- System



CAL POLY

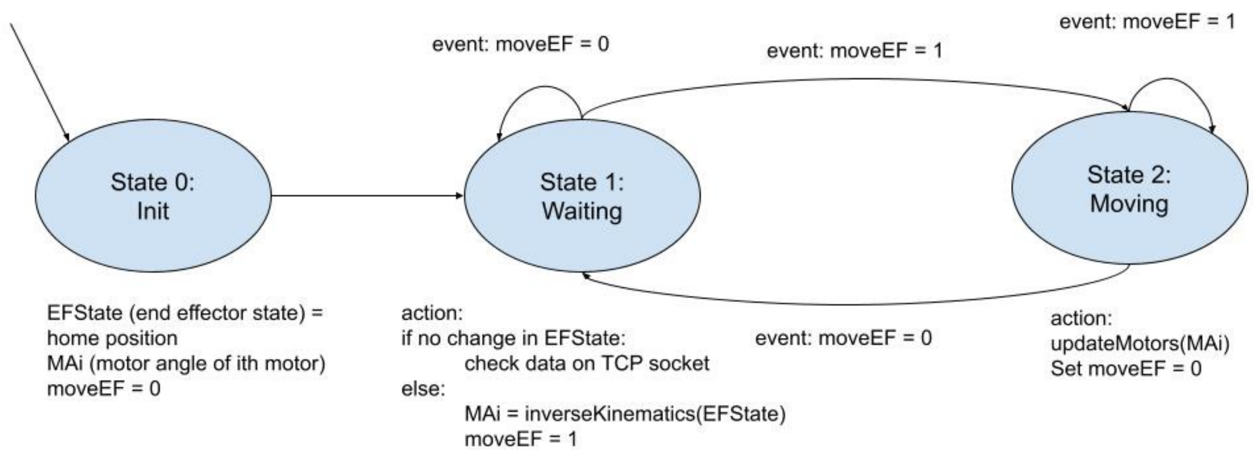
High Level Hardware Layout

- Positional commands are sent wirelessly to computer
- Inverse kinematics determine joint angular position
- Single I2C bus services all motor actuation and control (MAC) units
- PI cam and LIDAR continuously stream data to controller



Computer Finite State Machine

- Computer waits for data to be sent through TCP socket
- When data is received, inverse kinematics updates motor position
- Computer returns to waiting state
- Two separate process are running to stream optical and LIDAR data



Computer and LIDAR/Optical Sensors

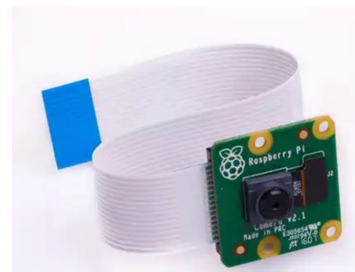
- Raspberry PI 4B provides processing power and interfaces for sensors/actuators
  - Used for arm logic, networking, sensors/actuation
  - 5V and 3-5 watt (average power)

- Marker LIDAR module is a lightweight and cheap LIDAR sensor with UART interface
  - Used for object detection
  - 3.7-5.2V and 70mA average current

- PI camera module for video stream
  - Used for camera-guided operation
  - Native PI connection (CSI)

Raspberry PI 4B

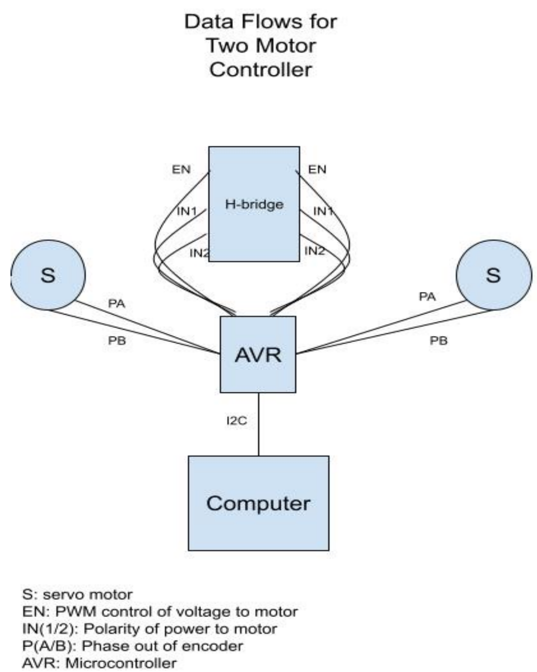
Marker Focus LIDAR Range Finder

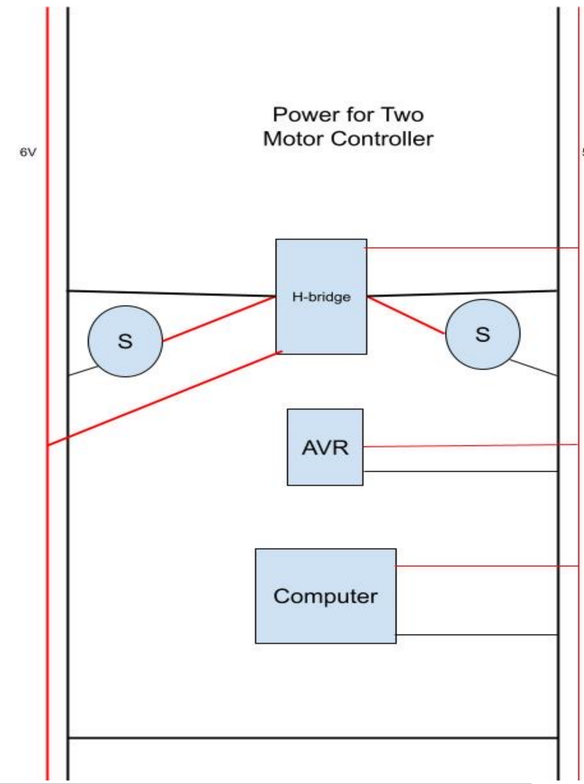


PI Camera Module

Motor Actuation and Control (MAC) Layout

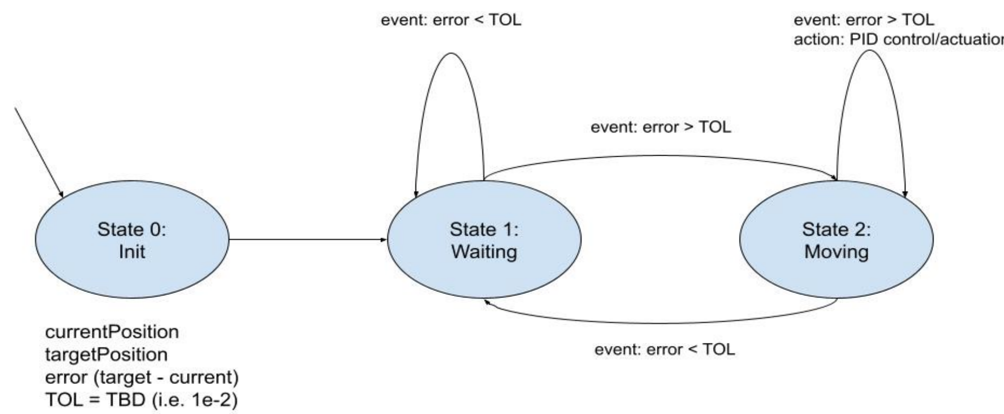
- Motor control will use an H-bridge and Atmega 328p microcontroller
- Allows for dual motor control
- Controller will listen on an I2C bus for position commands
- Closed loop motor control is the responsibility of controller (not computer)
- Separate power lines are used for motor power and microprocessor/computer power





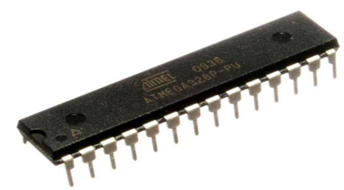
MAC Finite State Machine

- Motor Controller (Atmega 328p) will wait until address is sent data on I2C bus
- currentPos is changed by incoming data
- error is updated and state is changed to "Moving"
- PID is used to change motor angular position until error is within tolerence



Motor Control Hardware

- Atmega 328p is an 8-bit AVR microcontroller
  - Used for motor control logic
- L293D is an H-bridge capable of 4.5-36V and 1A bidirectional current
  - Used for motor speed modulation and direction
- 26RPM BDC motor, quadrature encoder
  - 3V-12V operation voltage, .21A no load current
  - Used for joint actuation and angular position determination



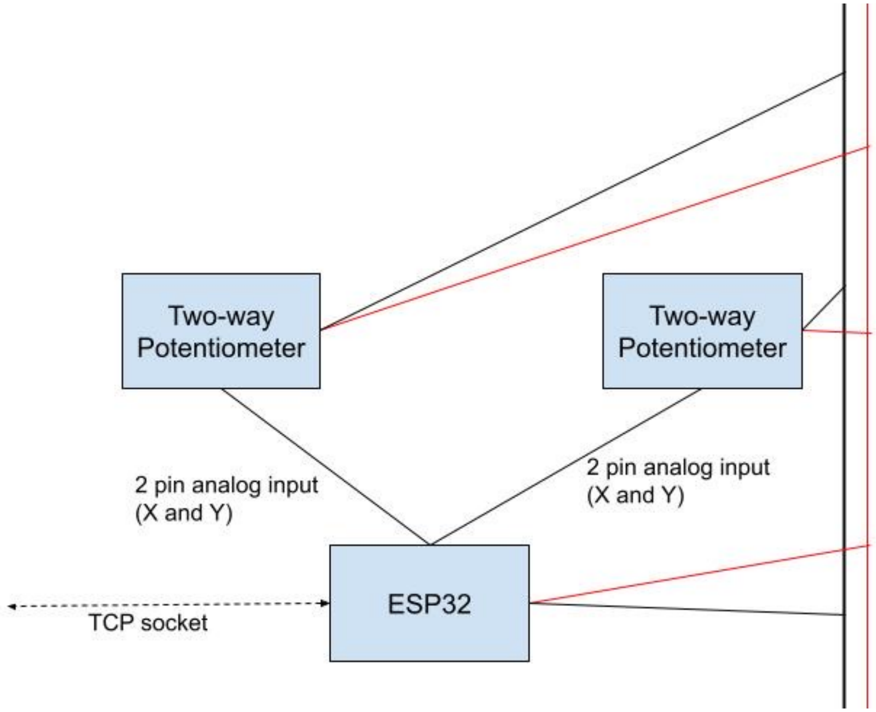
ATMEGA328p

L239

26RPM BDC Motor

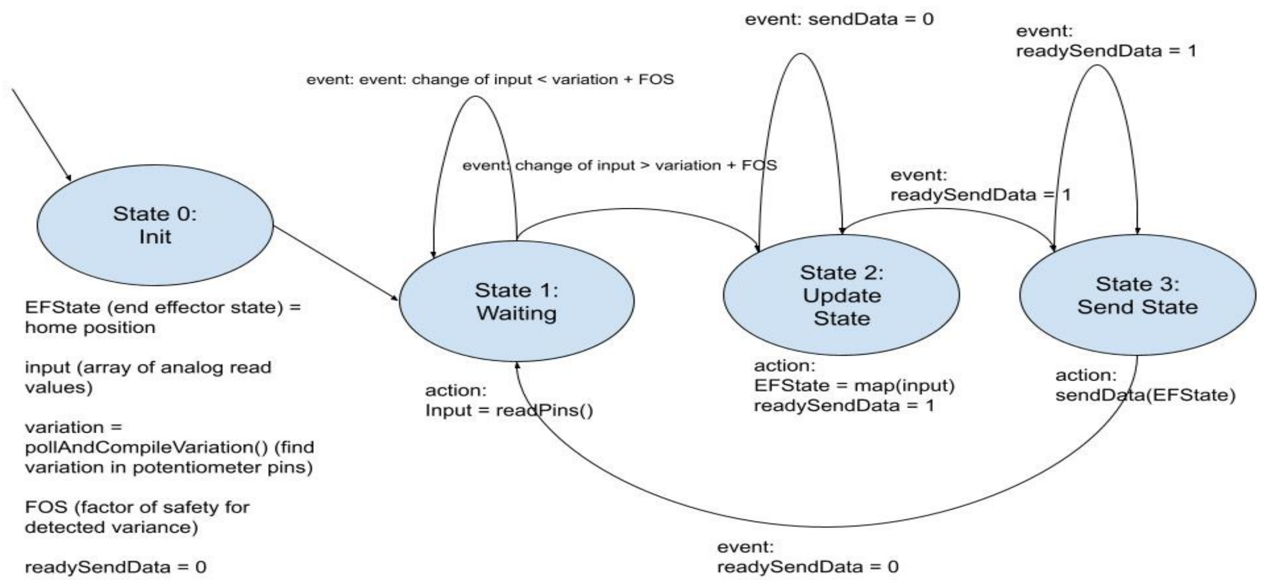
Controller Layout

- ESP32 is a microcontroller with IoT capabilities (wifi and bluetooth)
- Two two-way potentiometers monitored for changes
- When a change is detected, analog values are mapped to updated EF states
- New states are sent to computer and arm orientation is updated



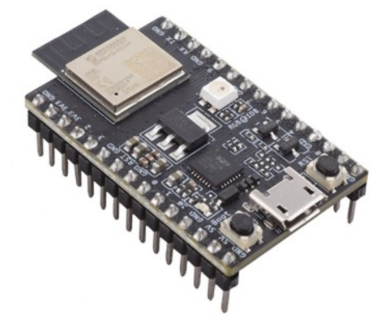
Controller State Machine

- Controller updates end effector position using input from two joysticks
- Each joystick is a two-way potentiometer allowing for 4 DOF control
- End effector position is updated using analog values (from input pin) and a mapping function
- These values are streamed to robot computer



Controller Hardware

- ESP32 provides a way to wirelessly connect to raspberry PI
  - Provides GPIO access for reading pin values
  - Provides processing power for mapping values
- Analog Joystick provides user interface
  - Analog input is mapped to changes in end effector position

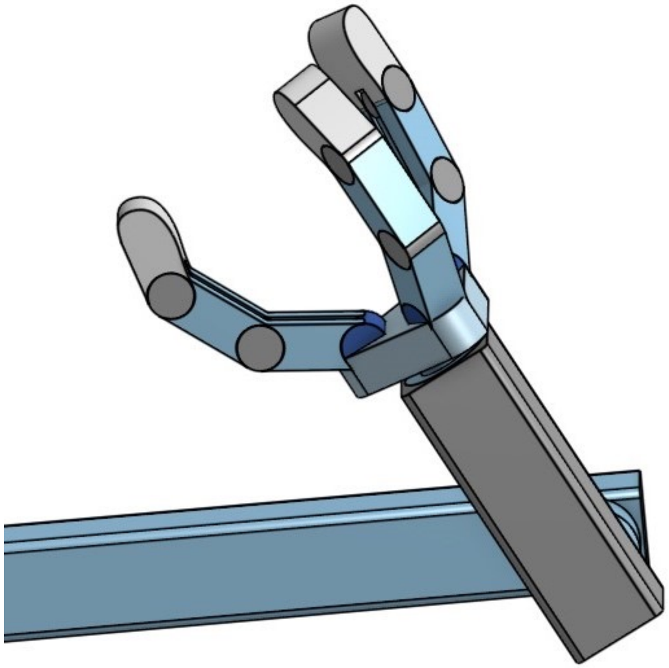


ESP32-C3

Adafruit Analog 2-axis Joystick

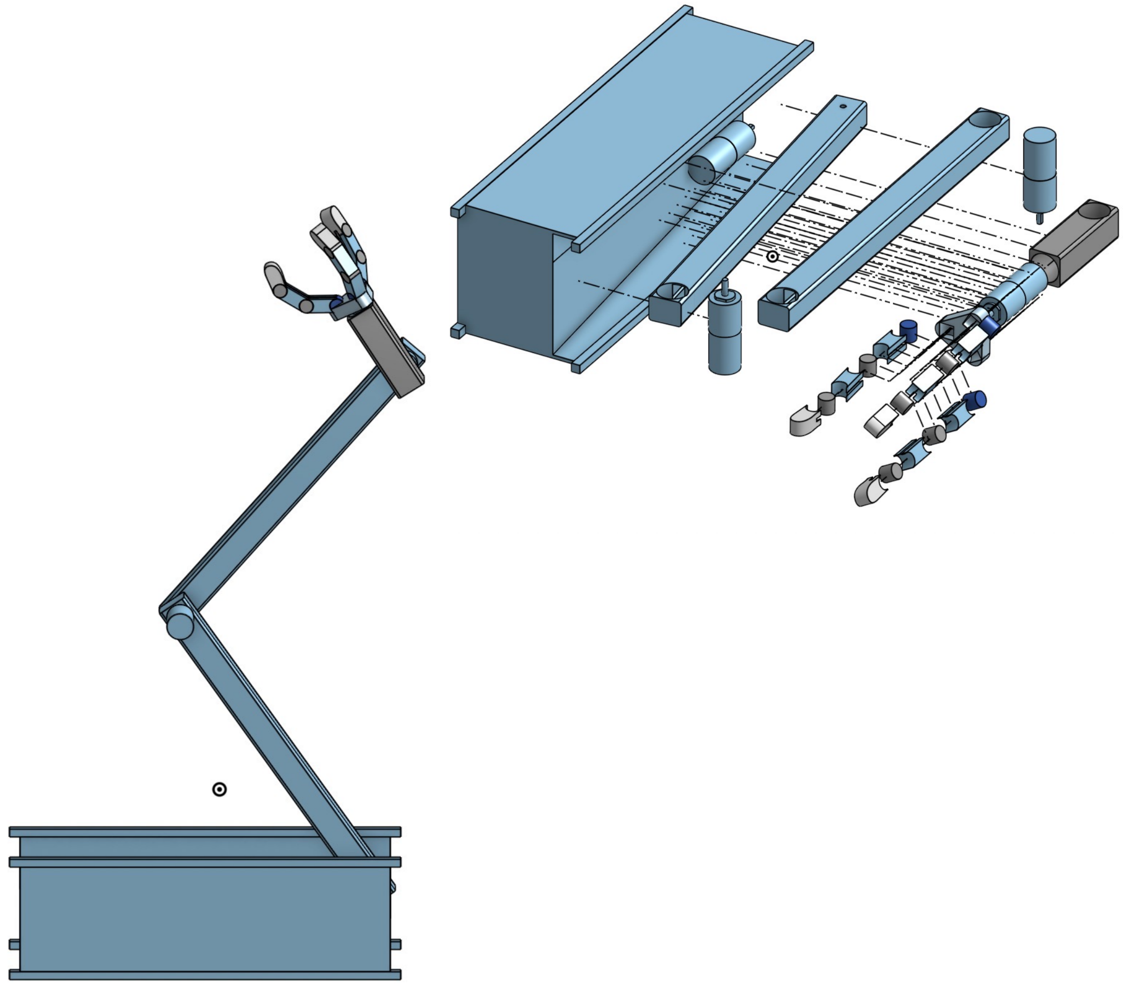
End Effector Design

- 3 prong clamp end effector
  - Each prong features a conformable surface for increased traction
- A Raspberry Pi system will be used for control and actuation of the arm and end effector
- Gecko tape will be used
  - Gecko tape is a non-chemical adhesive tape

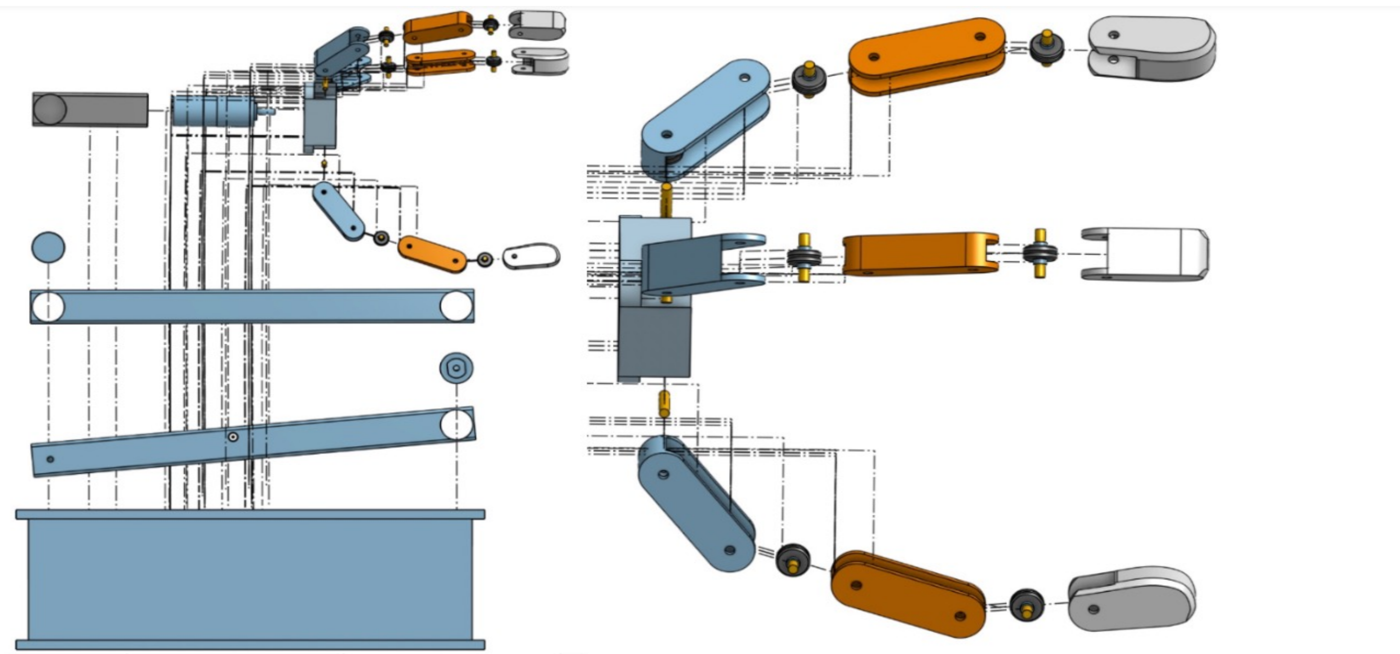


Design

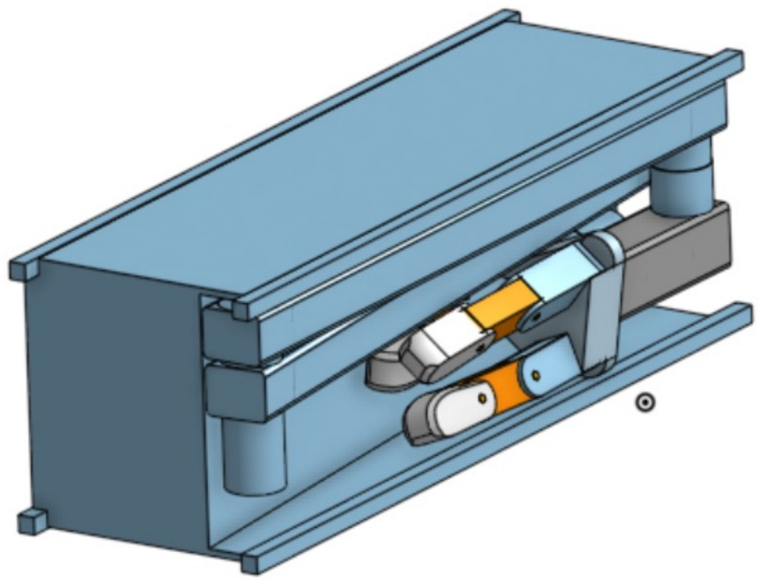
- The frame of a 3U CubeSat
- 3 servos
- 2 arms
- 3 prong end effector with 3 joints



Exploded View

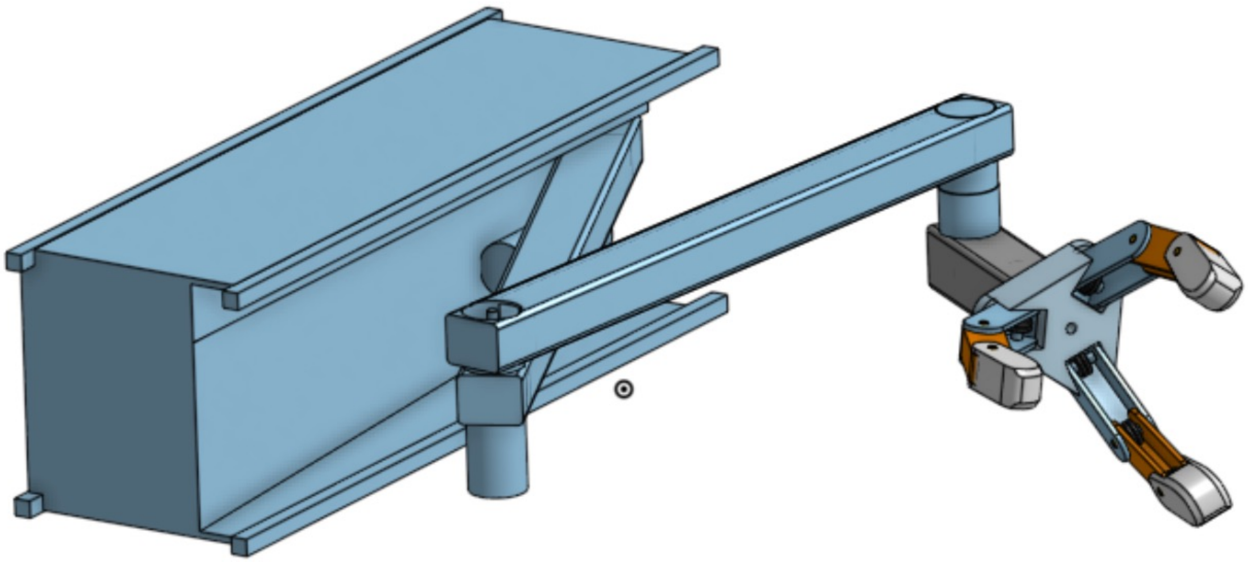


Stowed Arm Configuration



CAL POLY

Extended Arm Configuration



Thank you!