RockSat-X 2022 Final Report

MARSHA

MARSHA was designed to prove that Deep Reinforcement Learning is a viable method for adaptive learning.

Aaron Borger, Jonathan Herberger, Juliette Haggith, Riley Mark, Samuel Mark, Jacob Sutherland, Jakob Waltman, Cole McCall, Nathan Appleby

Dr. Dan Lawrence, Dr. Dale Hamilton, Autumn Pratt

Northwest Nazarene University

20 September 2022

1.0 Mission Statement

Current spacefaring robotic systems struggle to adapt to varying tasks and environments. Intelligent robotic systems have the potential to assist astronauts and solve challenging problems such as space debris removal, collision avoidance, and in-orbit spacecraft servicing and assembly. Current robotic systems used in space require manual control, which will prove difficult for distant uncrewed missions where communication times deem remote control impractical. Remote control will also hinder the creation of large robotic workforces, as the necessity for human involvement would be too great. It has been demonstrated that Deep Reinforcement Learning (Deep RL) can be used to solve complex problems such as solving a Rubix Cube with a dexterous robotic hand. This technology could be revolutionary in the aerospace industry as an intelligent, dexterous system could learn to perform tasks in space and would decrease the need for human intervention. This work aims to develop adaptive robotic arms that train in simulation and then complete a complex task in the real-world space environment. For this project, the complex task involves two robotic arms that throw and catch a ball in space. The main contribution of this work is to demonstrate that deep meta reinforcement learning can bridge the gap between simulation and the harsh environment of space and could become a significant asset in the mechanization of the aerospace industry.

2.0 Mission Requirements and Description

Background of MARSHA

Recently, deep reinforcement learning (deep RL) has become a standard method to enable robots to adapt to changes in the environment. Deep RL is a branch of machine learning that combines learning through trial and error (reinforcement learning) with deep neural networks (deep learning). Although deep RL has shown promising results, reinforcement learning simulates learning through trial and error which requires attempting a task thousands of times before the agent is successful. Training for deep RL is impractical for robots as it is too cumbersome for the mechanical assembly to perform the training. In addition, creating an environment that a robot can train in without any human involvement is impractical, if not impossible. Artificial intelligence researchers have begun to train intelligent robots in simulations and then deploy that training into real-world robots to solve this issue. Training an agent in simulation is a challenging task as it is incredibly difficult to simulate the physical world perfectly. However, recent advancements have shown that it is possible to overcome the imperfections in simulators. This new project, MARSHA, was designed to test the accuracy of simulations for space applications where there will be significantly less gravity than on Earth, as well as to show the viability of AI controlled robotic arms to perform simple tasks in microgravity.

Overall MARSHA Requirements

The MARSHA payload had many requirements both self-imposed and from COSGC. COSGC had many requirements, but the most important were the requirements to have a max electrical charge of 1 Ah, a max voltage of 28V, remain within a compact usable payload volume and maintain a deployment speed limit of one inch per second. These requirements were taken into account when choosing electrical components, creating the software, and designing the mechanical system. The self-imposed requirements included having both arms successfully deploy, a ball be thrown and caught, video be recorded and saved, and have the arms fully retract and the grippers latch on for reentry.

Background of ISRAEL

Soft robotic development has opened up a new world of possibilities for robotics with its development in recent years. Integrated Soft Robotics Above Earth Limits (ISRAEL) has integrated soft robotic technology onto a barnacle experiment within the MARSHA payload to test the viability of using similar technology in future projects. The project is independent from the primary experiment and aims to test whether silicone based soft robotic digits actuated using hydraulics will be a viable technology to use for space applications.

Overall ISRAEL Requirements

ISRAEL aimed to actuate a silicone digit using DOT5 brake fluid. The minimum success criteria was to expand the finger using the fluid. The maximum success criteria was to measure fluid pressure and applied pressure data from the system.

3.0 Payload Design

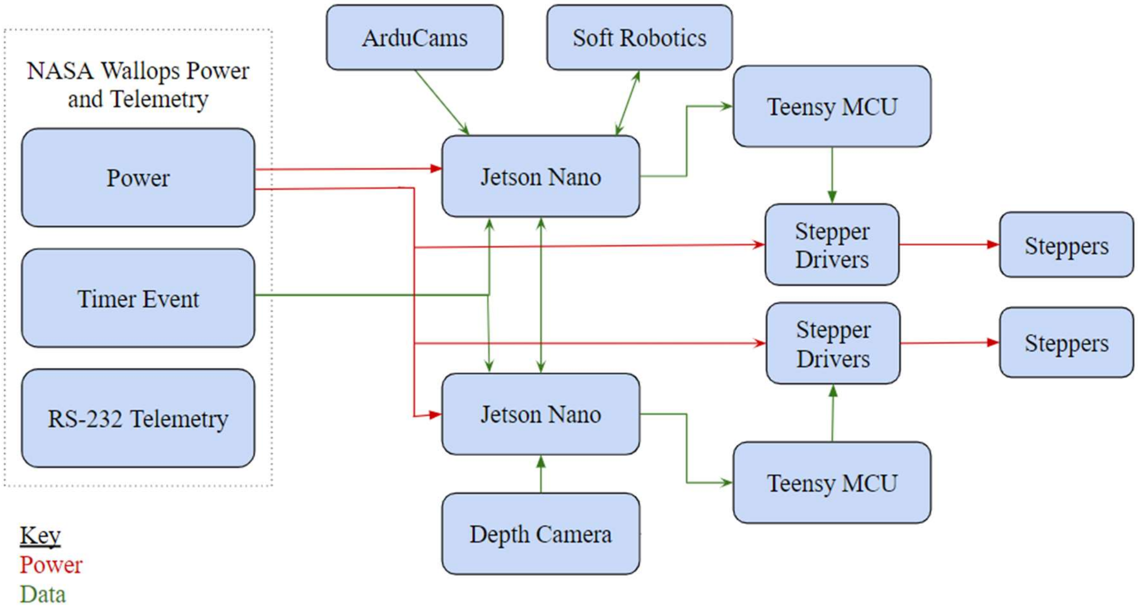


Figure 1 - Functional Block Diagram

MARSHA/ISRAEL Design Overview

This payload was designed to incorporate two robotic manipulators, four balls, and all necessary electrical components for the MARSHA project. Project ISRAEL was an independent "barnacle" experiment that was placed in between the two robotic arms. Figure 2 shows a rendering of the final design of the payload as it would look while operating in space.

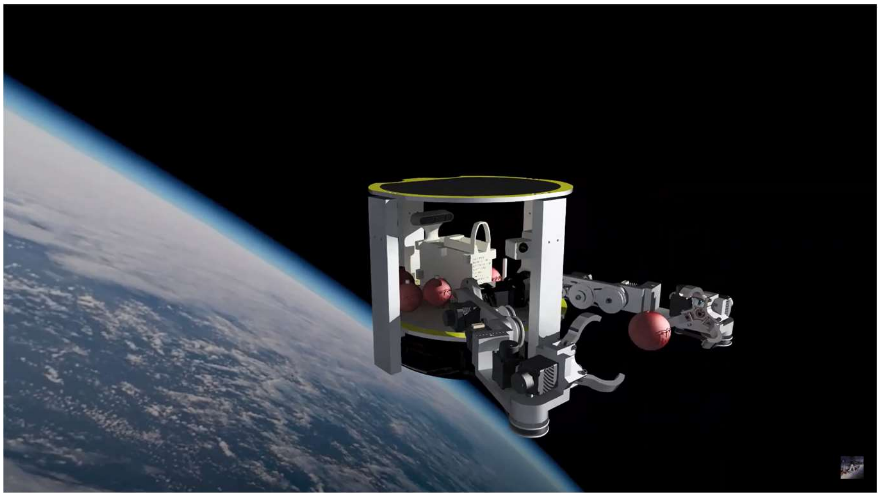


Figure 2: MARSHA Deployment CAD Rendering

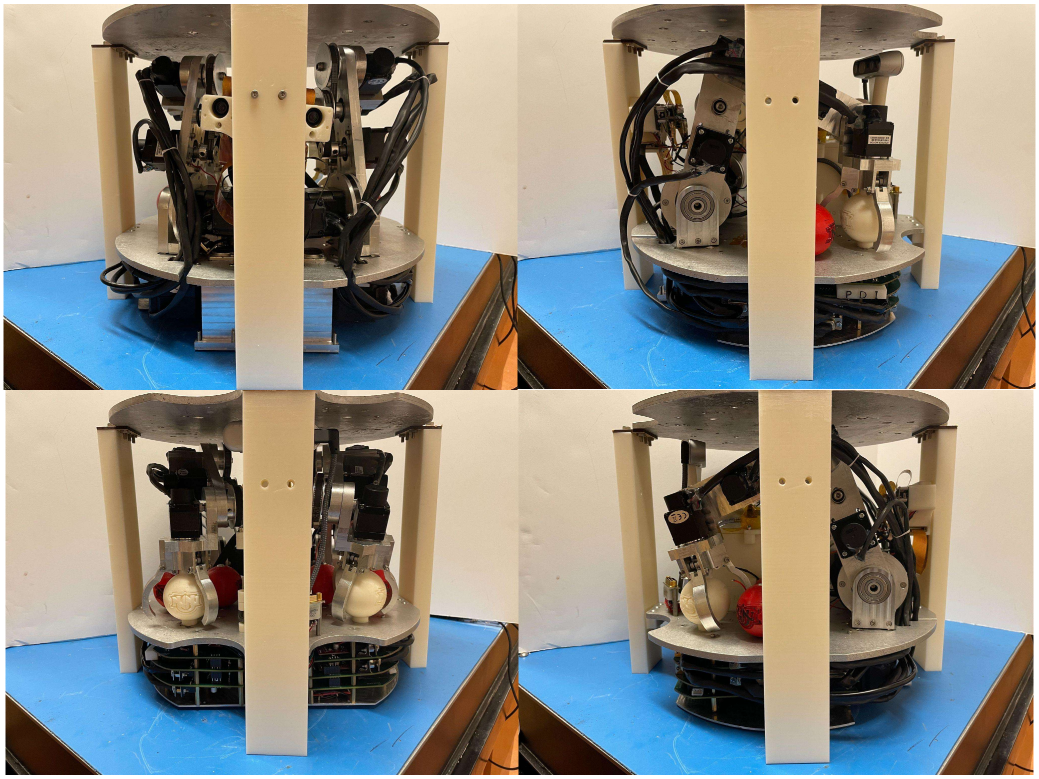


Figure 3: MARSHA/ISRAEL Pre-Flight Orthogonal Views

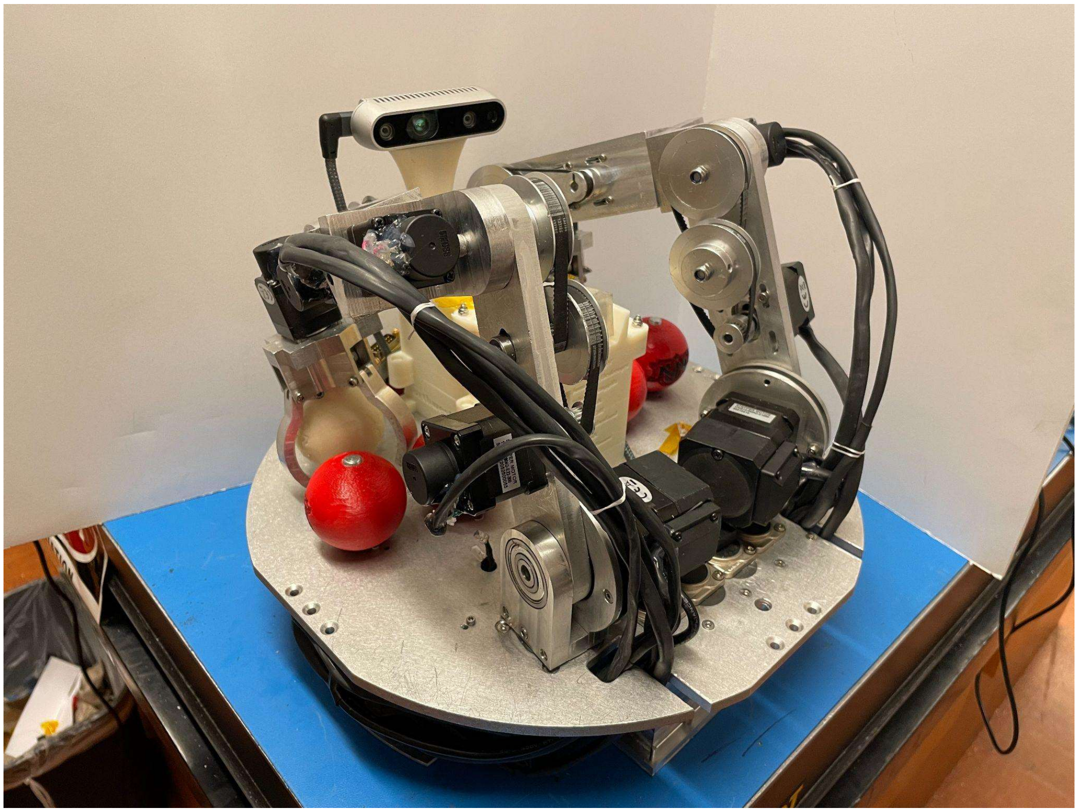


Figure 4: MARSHA/ISRAEL Pre-Flight Isometric Views

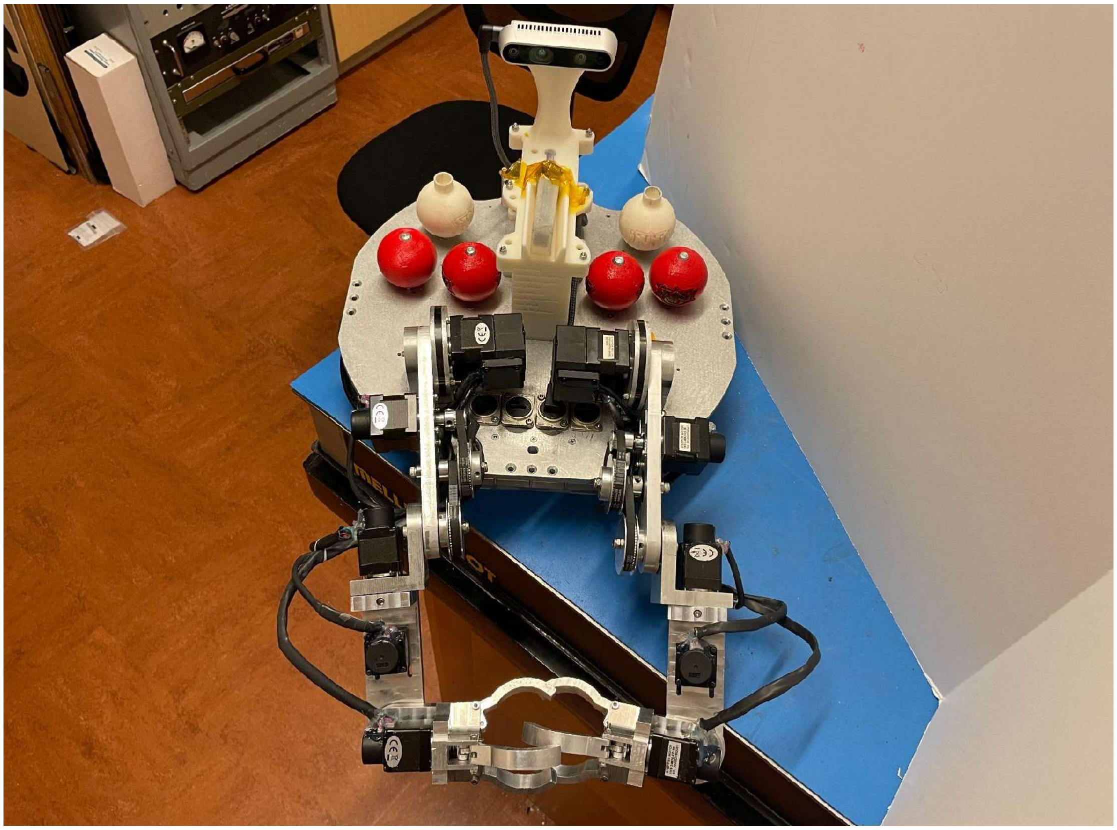


Figure 5: MARSHA/ISRAEL Pre-Flight Deployment View

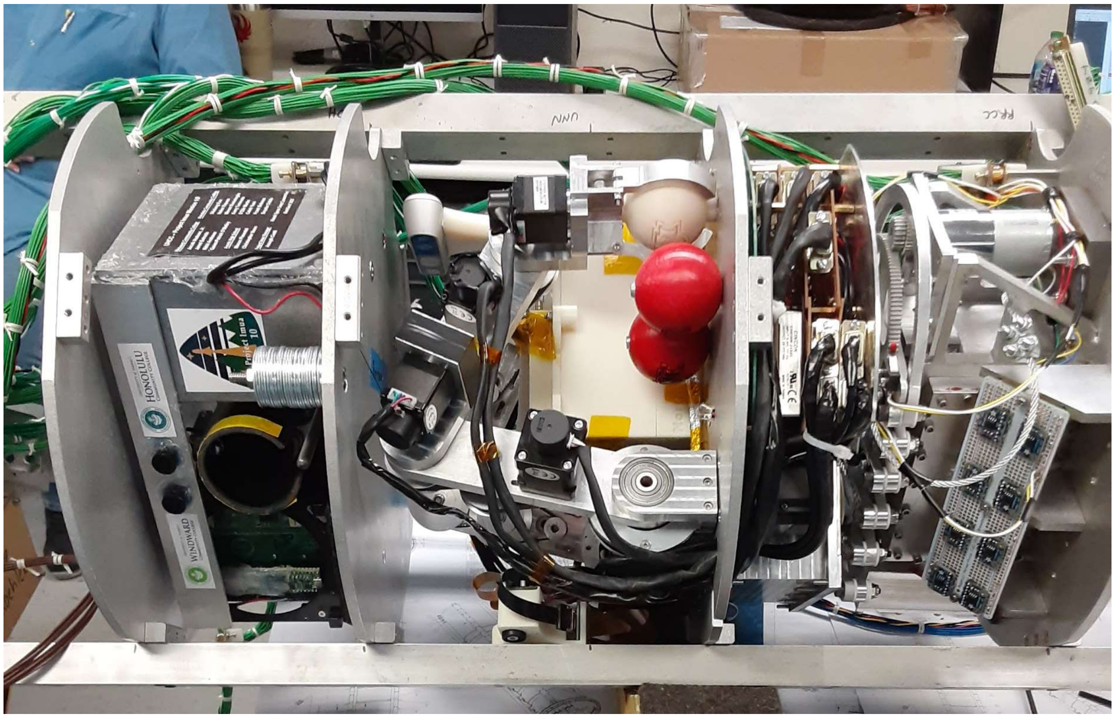


Figure 6: MARSHA/ISRAEL Fully Integrated at Wallops

Robotic Arm Design

In order to successfully complete our mission objectives, we needed to have a set of robotic arms that were both compact and capable enough to perform complex maneuvers. After researching possible off the shelf options, it was apparent that there was nothing on the market that would meet all mission requirements, so a robotic system was designed from the ground up using a series of stepper motors. The robotic arm design went through more than twenty iterations, including test articles that were 3d printed, before the design was finalized and manufactured out of aluminum.

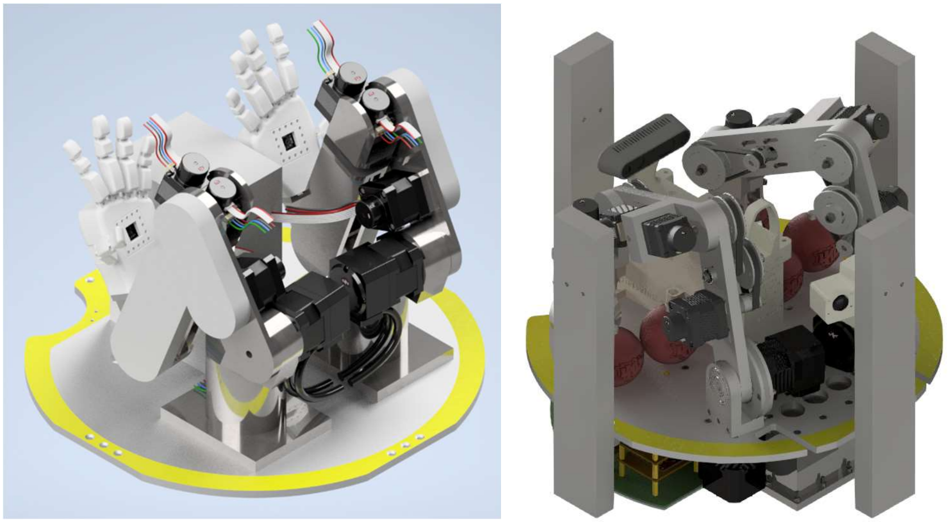


Figure 7: MARSHA/ISRAEL Concept and Final CAD Models

It was decided that the payload deck would be mounted at a the ¾ position to allow for the robotic arm to be mounted above the dry box and electrical components.

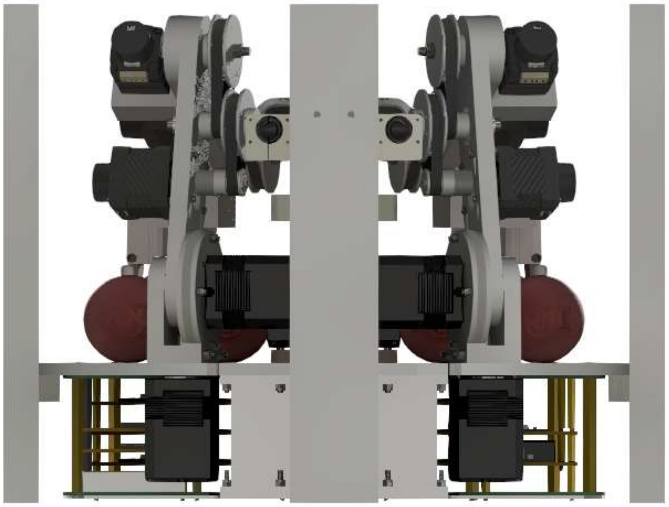


Figure 8: ¾ payload deck position CAD Model

As tests were conducted, and torque calculations were completed, it was evident that the arm would not have enough torque to operate in one G. Itwas decided that gearing the motors was the best way to provide more torque while remaining in a compact form factor.

The bottom link required the highest amount of torque with a necessary gearing ratio of more than 5:1. Although there were several off the shelf options for planetary gearboxes, none would fit within the size constraints. The solution was to design a custom compact planetary gearbox to fit within a 10mm footprint.

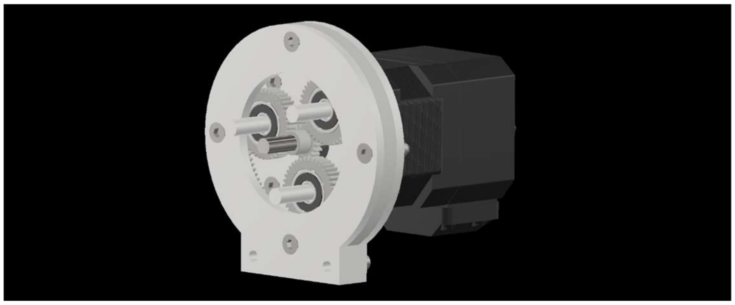


Figure 9: Planetary Gearbox CAD Model

A gear and belt system was employed on two of the links to simultaneously increase the torque output and decrease the necessary torque by moving the motor closer to the fulcrum of the arm.

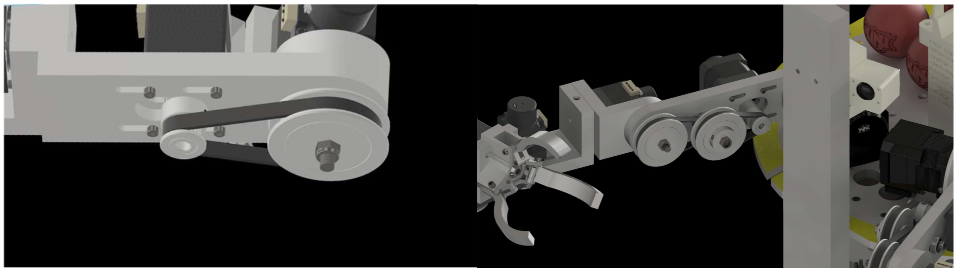


Figure 10: Gear/Belt System CAD Model

The robotic arms featured an independently-actuated robotic gripper that used a single stepper motor and threaded rod to animate three gripper links. The gripper was designed to interact with a ball of known dimensions and also featured a locking mechanism that used the threaded rod to secure itself to the payload deck for launch and re-entry.

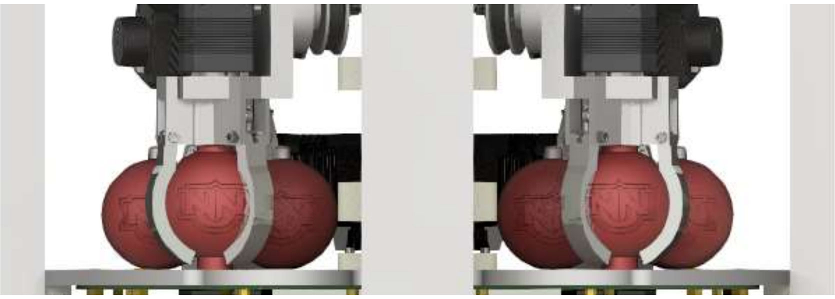


Figure 11: Gripper/Gripper Mount CAD Model

Electrical System

Due to the size requirements of the mechanical system, the base plate was arranged in the lower mid position. This allows for the entire electrical system to be integrated below the base plate which allotts 2.82 in. of height for the system. In this space, there is the dry box, with two Jetson Nanos, and the board stacks with the rest of the electrical components mounted around the dry box.

The dry box was designed to fit two Jetson Nanos, three HDMI high vacuum connectors, one USB high vacuum connector, and a 24-pin high vacuum connector. The dry box design allows for stiff cables on the USB and HDMI ports to occupy the empty space and contains the three USB - Ribbon Cable converters. This design can be seen below in Figure 12. The dry box has been made in house by machinists at NNU. As per the machinists recommendation, the o-ring was bought as 75 durometer cord stock. The o-ring was then cut and glued to be the correct size and fit for the dry box.

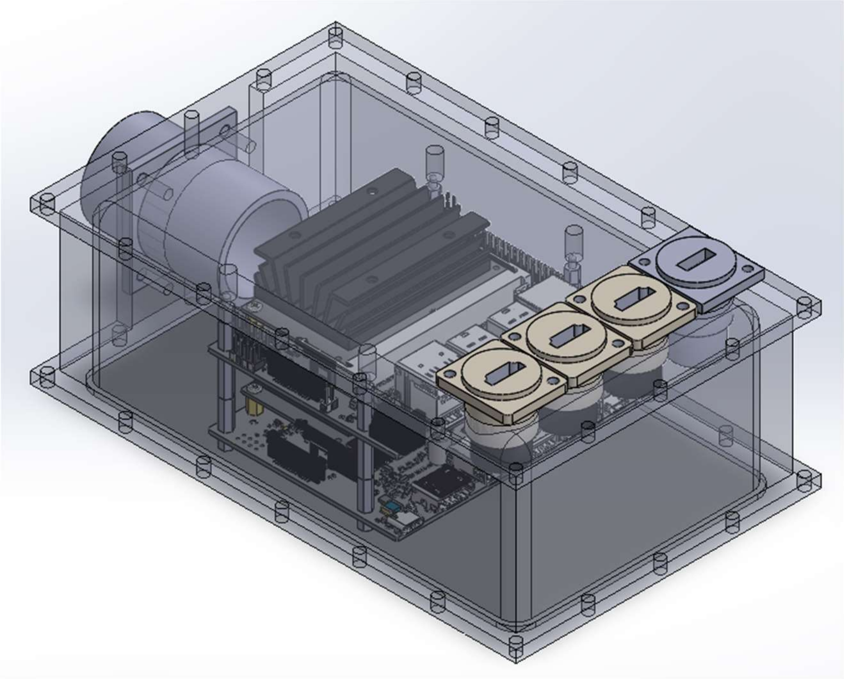


Figure 12 - Dry Box Design

The board stack went through three primary design iterations. The initial prototype of the electrical system consisted of a breadboard with the necessary components for one arm. The first iteration of the electrical system was also the first electrical system to use a professionally manufactured printed circuit board (PCB). This system can be seen in Figure 13. Many wires were required to complete this circuit board, and wire solders regularly broke during testing and debugging of the system. An Othermill Pro CNC and PCB Milling Machine was used to create the copper boards shown. These boards were problematic as the copper would break off of the fiberglass with little force. Due to time constraints, this system had to be finished and installed into the payload in preparation for testing at the Wallops Flight Facility in June 2022.

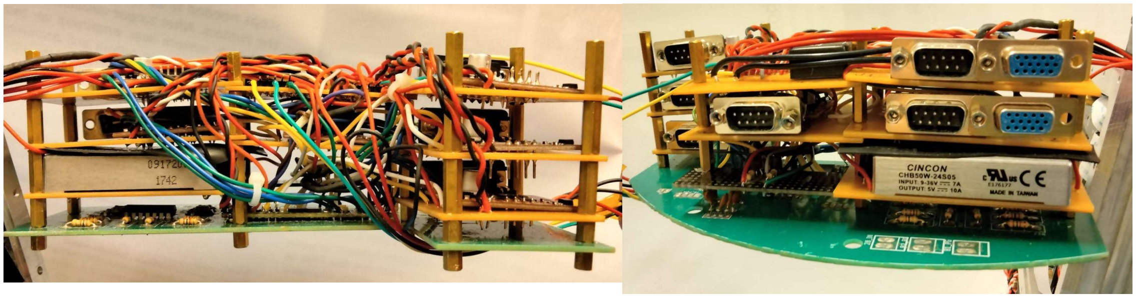


Figure 13 - First Electrical System

During the end of the month of June, a new electrical system was designed and sent to be professionally manufactured. This system uses in-board traces and header pins were used where the first system would require wires. This means that each board can be easily removed without desoldering or cutting wires. Each board is also equipped with test pads, where the voltage of power traces can be measured, and capacitor mounts, where a capacitor can be soldered to reduce noise and interference in the power lines. There are two mirrored board stacks on the payload, each one dedicated to its own arm.

The bottom board in the stack houses the Teensy and can be seen in Figure 14. The Teensy mounts to the bottom of the board where there is an op amp circuit that compares the values from four of the motor's redundant encoder lines. Each board stack is equipped with two LEDs (which solder in above the op amp circuit), one red and one green, to show the status of that arm's Jetson. The top portion of the board houses the 28V - 3.3V converter and the plug-ins for the Partial Deployment Inhibit or Full Deployment Inhibit depending on which side the board is on. The converter takes the 28V TE-R line and converts it to 3.3V. This signal is then sent to the Jetson Nanos to confirm that the rocket has entered space. The six header pin pads can be seen with 4 or 6 connections; those will send signals to the motor controllers on the following boards. The larger header pin pans (two of 5 connections) connect the Teensy to the Jetsons.

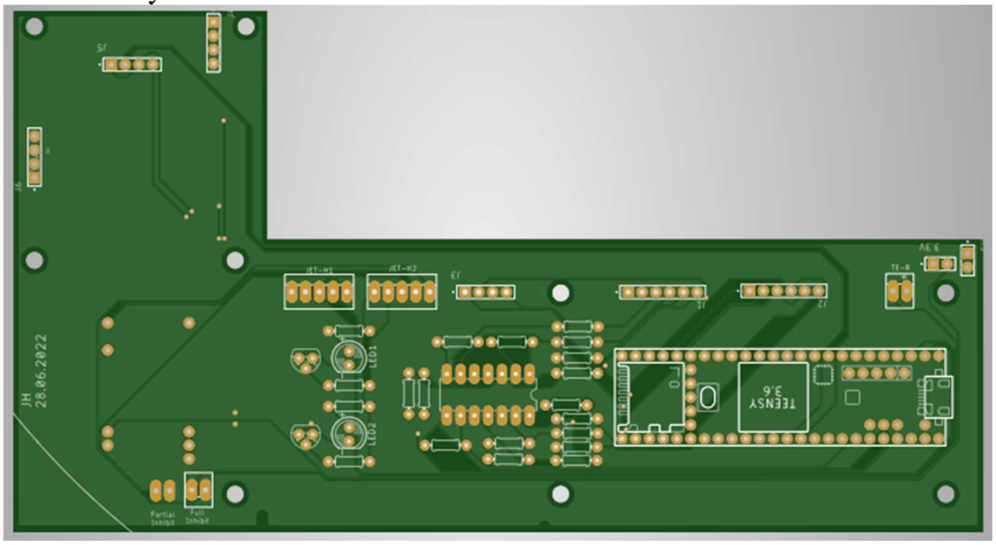


Figure 14 - Teensy PCB

The second board on the stack holds four of the six motor controllers. This board can be seen in Figure 15. This board carries the signals from the Teensy Board to the dedicated motor controller and then to the 9 or 15 position D-sub plugs. From there, the motor can plug in and be run.

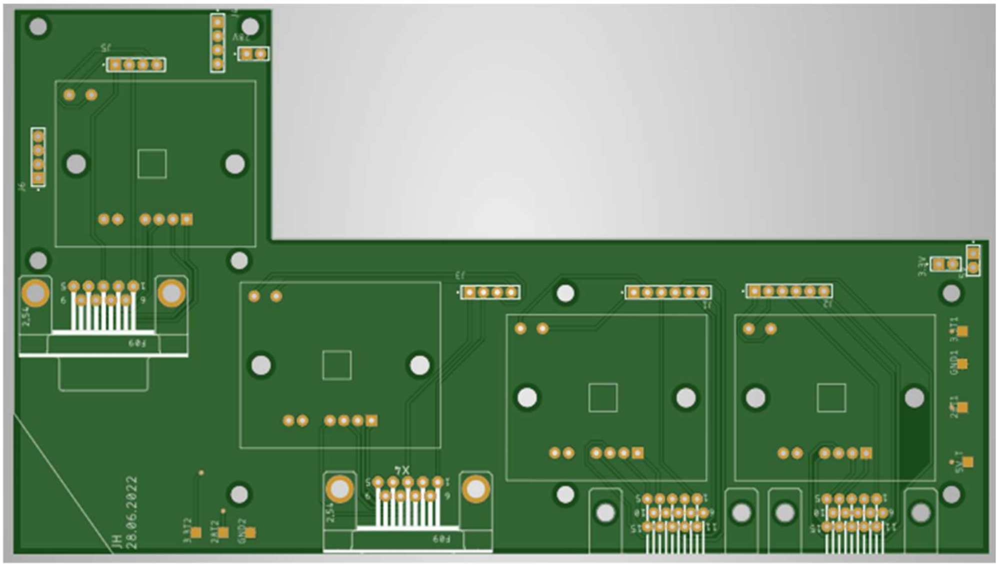


Figure 15 - Motor Controller Board

The top board holds the last two motor controllers and the 28V - 5V converter. This board can be seen in Figure 16. The board completes the same function as the motor controller board for the two motor controllers. The 28V - 5V converter provides the Jetson Nanos with the 5V power they need from the 28V GSE lines.

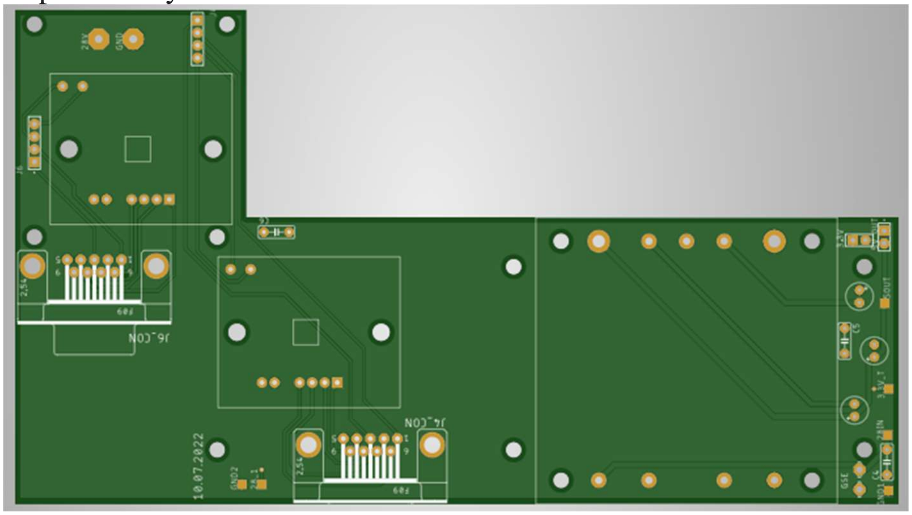


Figure 16 - 28V-5V Board

The three boards stack on top of one another and are then screwed into the baseplate from the top and into the dry box lid on the bottom. This ensures that there is minimal movement during vibration testing and flight.

Payload Command System Software

The payload requires an overhead system to control each of the subsystems. This system is called the payload command system (PCS) which is controlled collaboratively by the dual Nvidia Jetson Nanos that are the payload's central On Board Computing (OBC). The PCS is a ROS service in which each payload subsystem queries to indicate its current state and to receive commands such as activation, deactivation, and shutdown.

When GSE is activated as shown in Figure 17, the Jetson Nanos receive power and begin to boot up. During the boot up process, the PCS begins as a system service. At the beginning of the PCS, connection is tested between the two Jetson Nanos to ensure that they can begin the control system.

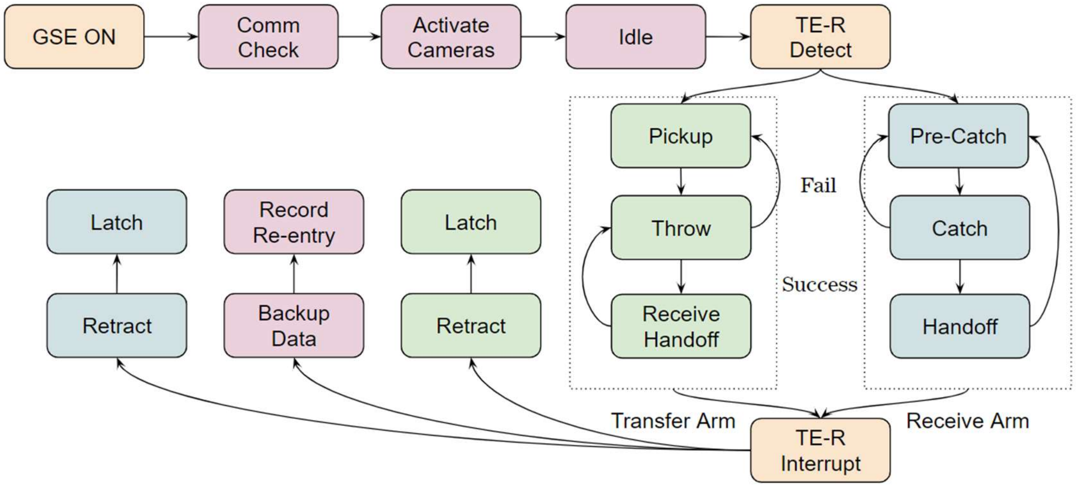


Figure 17 - Master Payload State Machine

The PCS continues by testing for communication between each of the Jetson Nanos and their respective Teensy MicroController Unit (MCU) that are responsible for the real time control of the robotic arms. After successfully connecting, both Jetson Nanos will remain idle, only recording footage from the longeron cameras until the timer event (TE-R) detection system is triggered. The timer event is a redundant signal provided by NASA that is scheduled to activate T+82 seconds after launch, which is when the rocket has reached a nominal altitude of 87 km above the Earth and the skirt has deployed, exposing the payload to space. This signal is provided to the Jetsons from the 28V-3.3V converters through the GPIO pins. Both Jetson Nanos receive the signal and confirm the other Jetson received it as well. If one of the Jetson Nanos does not receive the signal, the TE Detection Failure procedure is conducted If the redundant timer event signal (TE-R) is received by both Jetson Nanos the master payload state machine continues as indicated in Figure 17. A block diagram for the TE Detection system is shown in Figure 1 which gives an overview of how each subsystem in the payload is connected electrically.

After the TE-R is detected, the experiment begins with the robotic arms unmounting from their secure positions. The transfer arm begins by conducting the pickup procedure and then the unfold procedure. The receiving arm begins by conducting the unfold procedure and then waits in the pre-catch position. At T+120 seconds after launch when the rocket has passed the Kármán line and is 122 km above the Earth, the receiving arm begins the object detection and Kalman Filter systems. The transfer arm then conducts the throwing procedure. Once a sufficiently precise measurement of the ball's position and velocity has been achieved, the position and velocity is used as the observation for the TD3 reinforcement learning algorithm which serves as the grasp evaluator network. This neural network determines the optimal grasp configuration to catch the ball. The trajectory prediction system is used to determine the time the ball is at the desired catch position. The receiving arm then performs the agent's determined catch maneuver. As the payload is symmetrical, either arm can be assigned the transfer or receiving role. Therefore, if the catch is successful the roles can be switched and the receiving arm that just caught the ball can become the transfer arm and throw the ball. However, if the catch is unsuccessful, the next roles will be determined based on the number of balls left. Each arm has access to two balls which as shown in Figure 18.

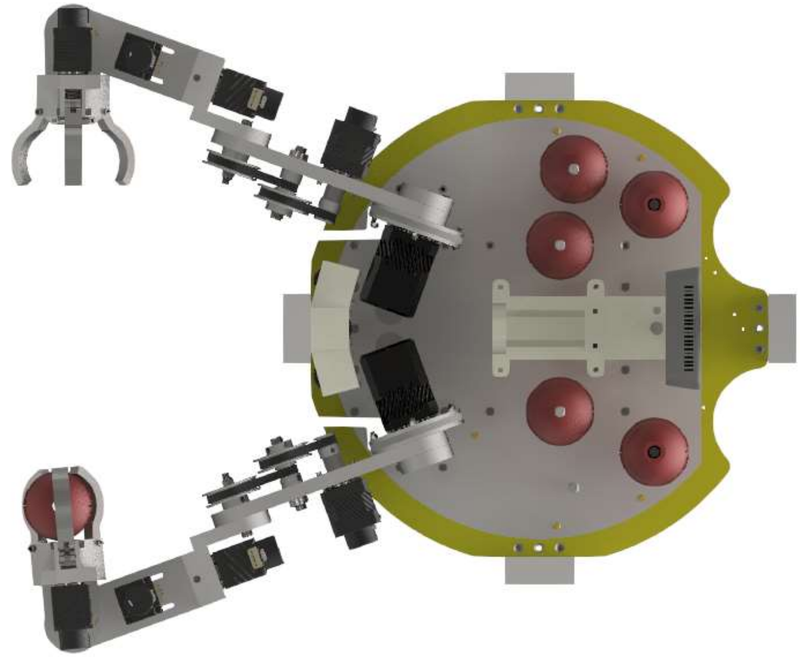


Figure 18 - Placement of Balls on the Payload

Longeron Cameras

The payload features three additional cameras that are solely for viewing the payload. The three cameras are mounted on the longerons that each of the payloads are mounted to. One of the cameras faces the inside of the payload to capture a view of the soft robotics experiment and the arms picking up balls. Two of the cameras face the outside of the payload where the throw and catch will occur. OpenCV is used to combine this footage into one image so the throw and catch can be viewed in the same video. Figure 19 shows an example of this view.

Figure 19 - View from the Longeron Cameras

Artificial Intelligence Software

The artificial intelligence subsystem is responsible for determining the position of the ball, predicting the ball's position at a set time in the future, determining the ideal grasp configuration to catch the ball using a deep reinforcement learning algorithm and then learning how to perform the catch better at the level of gravity currently being experienced. Figure 20 shows a block diagram of the subsystem.

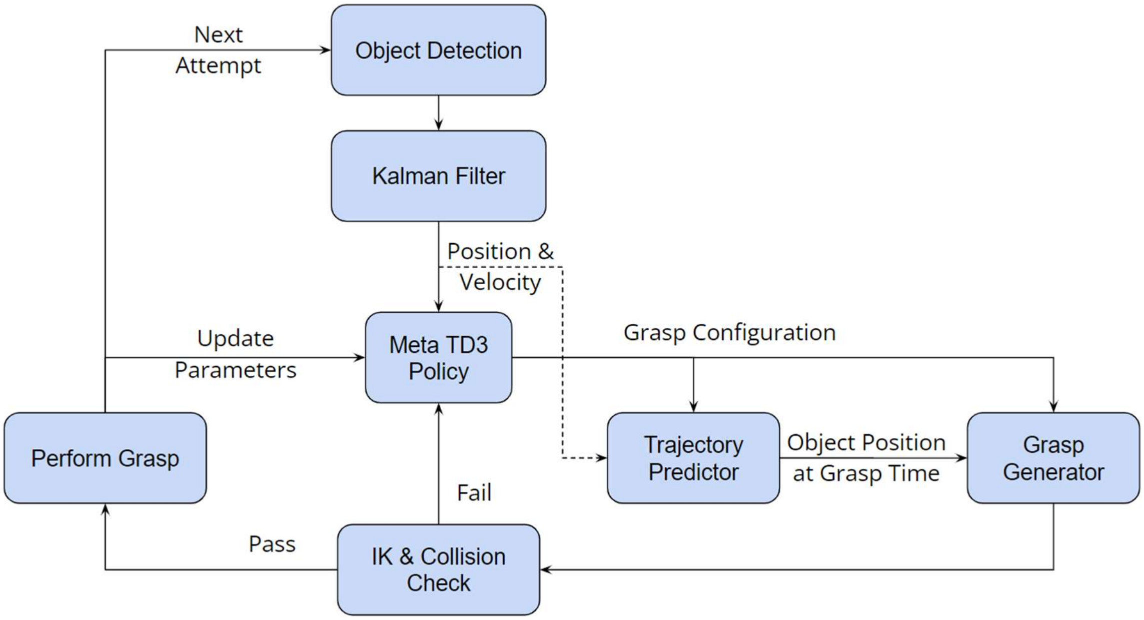


Figure 20 - AI Subsystem Flow Diagram

The loop begins by detecting the position of the object using computer vision. After the position of the object has been determined, a Kalman filter determines the position and velocity of the object with noisy measurements from the object detection system. The position and velocity are used as the observation for the TD3 deep reinforcement learning algorithm. The TD3 algorithm outputs the grasp configuration as the action. The grasp configuration includes the grasp approach vector, trajectory slice, and grasp time offset.

The grasp approach vector is composed of spherical coordinates relative to the center of the ball. The gripper will move along this vector to catch the ball. The trajectory slice corresponds to a point along the section of the object's trajectory that intersects with the robotic arm's workspace. Finally, the grasp time offset is an offset from the predicted time the object will arrive at the trajectory slice. This offset allows for the grasp vector to originate at a point slightly ahead or behind the location of the object. The trajectory predictor converts the normalized trajectory slice into the position and time that the ball is at the agent's chosen position. Once the future position of the ball is determined, the grasp generator converts the spherical coordinates relative to the object into positions relative to the base of the catching arm. Next, inverse kinematics and motion planning are performed by the ROS MoveIt library to determine the trajectory of each joint required to attempt the catch. A collision check is also performed to ensure the robotic arm does not collide with any part of the rocket structure. If the motion planning or collision checking fail, the TD3 policy is queried with an updated object position so it will generate a new grasp configuration. Once a successful motion plan has been determined, the grasp is attempted, and the performance is recorded.

Reinforcement learning uses a reward function to give the agent feedback on which actions performed well and should be repeated for the percepts that it received. The reward function that produced the best results for Marsha is given below:

$$r = \begin{cases} -4, & \textit{if pre grasp planning fails} \\ -2, & \textit{if grasp planning fails} \\ -1 * d_{grasp}, & \textit{if catch unsuccessful} \\ +10, & \textit{if catch successful} \end{cases}$$

The reward is determined by the state of the environment after the catch is attempted. In the first state, a reward of -4 is produced if pre-grasp planning fails. This negative reward punishes the agent for attempting a pre-grasp position that is not kinematically possible. In the second state, a reward of -2 is produced if the grasp planning fails. This is produced if the MoveIT rapidly-exploring random tree (RRT) motion planning algorithm can not find a path from the pre-grasp position to the grasp position. This could be because there is an unavoidable obstacle in the way, or the grasp position is kinematically impossible. A negative reward is given so the agent learns to avoid positions that result in failed motion planning. If motion planning fails, the agent is queried again with an updated position in an attempt to find an action that results in successful motion planning.

Robotic Control Software

Figure 21 shows the flow of data through the robotic control system. The different control methods (AI, traditional, and manual) interact with the move interface which serves as the robotic arm's API. The move interface converts the different control methods into a list of positions and quaternion orientations for the end effector. The move group uses the current angle of each joint from either the joint states in simulation or the encoders in hardware to determine the current state of the arm. The move group then performs motion planning and inverse kinematics to output a list of angles each joint must reach to start at its current state and reach each of the given positions and orientations. This list of angles called the joint trajectories are given to the joint trajectory controller which sends the trajectories to the Teensy microcontroller with UART. The Teensy uses a PI controller to smoothly move each joint to its correct trajectory.

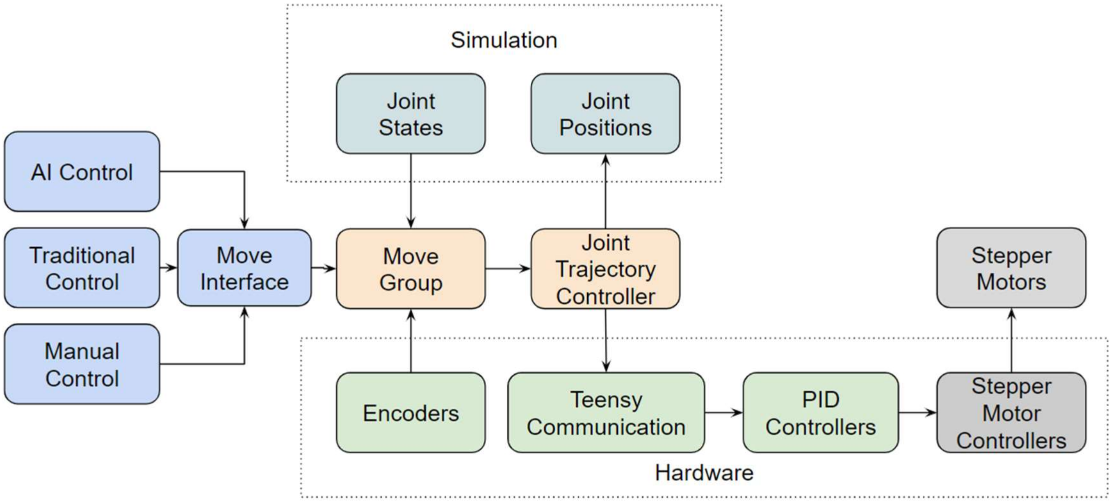


Figure 21 - Data Flow through Robotic Control System

Figure 22 displays the closed loop control method used to smoothly accelerate and decelerate each joint. The integral part of the PI controller is designed specifically to reduce the speed if the number of steps the stepper was supposed to move is more than the steps the encoder has actually moved.

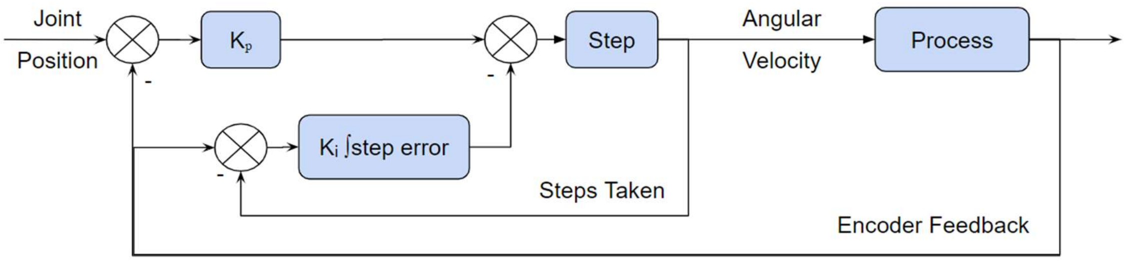


Figure 22 - Stepper PI Controller Diagram

4.0 Student Involvement

Project MARSHA was a student lead mission that consisted of seniors working to fulfill their design project requirements, paid research students, and volunteers. Members of the team were assigned different tasks to complete with the intent to create NNU's most ambitious project yet.

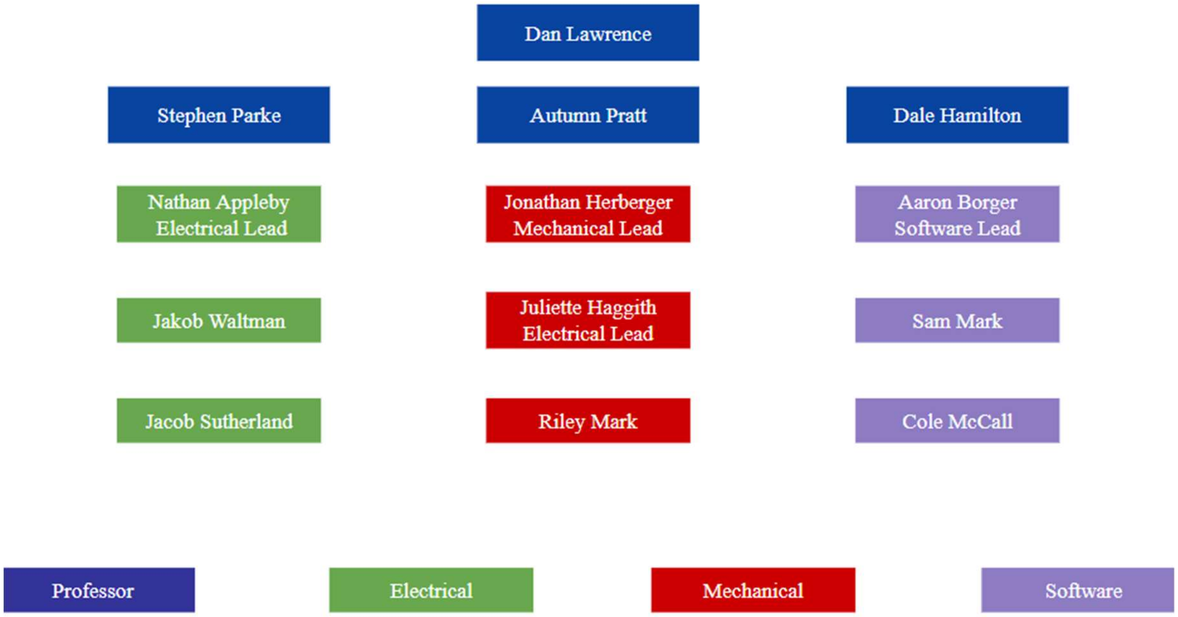


Figure 23 - MARSHA Organizational Chart

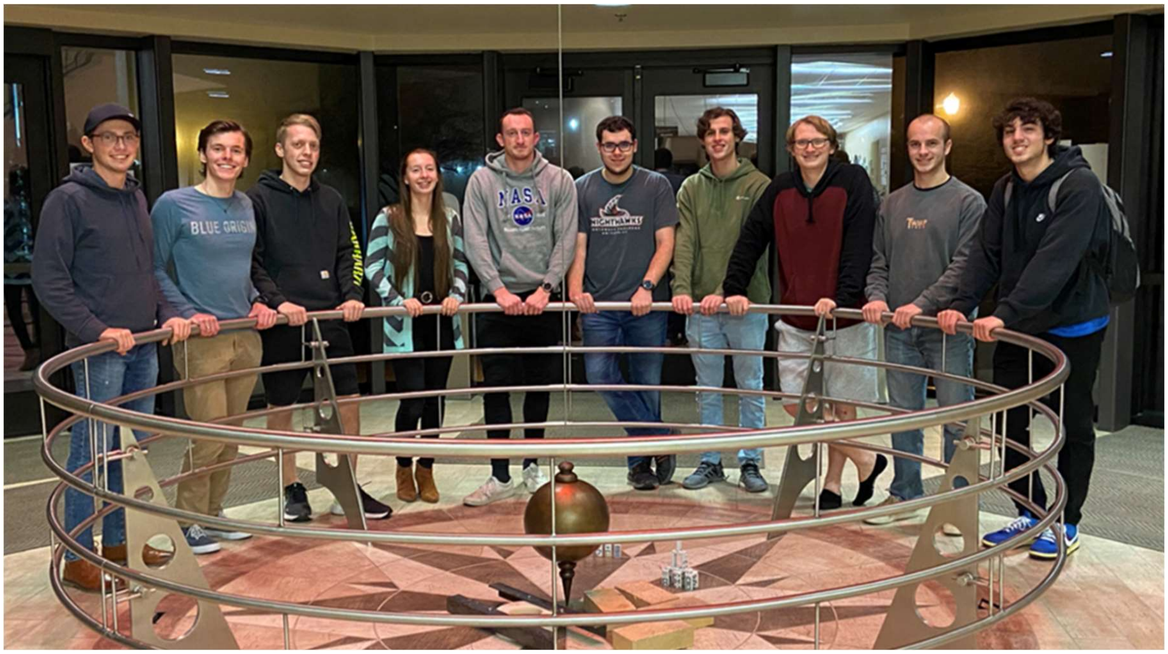


Figure 24 - MARSHA Team Photo

Aaron Borger: MARSHA Lead Software Engineer

Aaron Borger studied Engineering with a Computer Engineering concentration and computer science at NNU and was both the software lead and the team manager for MARSHA. Aaron developed the entirety of the software system, which included designing and training the AI algorithm to detect a ball and learn the ideal grasp configuration to catch a ball. He also worked on designing, implementing, testing, and troubleshooting the electrical system over the summer. After launch in August, Aaron began working at Blue Origin as an Aerospace Software Engineer programming the controller for the BE-7 lunar lander rocket engine. Aaron will continue working on AI controlled robotic arms through multiple projects. One of which will be mentoring the NNU Rocksat team next year and developing the next iteration of the object catching AI algorithm.

Jonathan Herberger: MARSHA Lead Mechanical Engineer

Jonathan is studying Engineering with concentrations in both Mechanical and Physics Engineering at NNU, and has worked as a member of NNU's Rocksat-X Research Team since Fall 2019. As the lead mechanical engineer for the project, Jonathan's primary job was to design, develop, and test the robotic arm system along with creating CAD drawings and renderings for the project. Over the summer, Jonathan helped extensively on the electrical system, wiring motors, and soldering components along with rigorously testing the mechanical systems to ensure that the robotic arms worked effectively. Jonathan will continue working as the lead mechanical engineer and chief engineer on next year's project.

Juliette Haggith: MARSHA Lead Electrical Engineer 2

Juliette is studying Physics and Engineering with a concentration in Mechanical Engineering at NNU and has been a part of MARSHA since its inception, but she took over as the lead electrical engineer after Nathan graduated. During the school year, she worked on mechanical projects to help out Jonathan. Juliette's responsibilities included the dry box design and electrical subsystem design, implementation, testing, and troubleshooting. Juliette will continue working as the lead electrical engineer and team manager for next year's project.

Nathan Appleby: MARSHA Lead Electrical Engineer 1

Nate studied Engineering with a concentration in Electrical Engineering at NNU and was the electrical lead of MARSHA until his graduation. He designed the first iteration of the electrical subsystem, but was unable to complete the system before he left. Nate started working at Micron as a dynamic random-access memory test engineer.

Riley Mark: ISRAEL Project Lead and MARSHA Mechanical Engineer

Riley Mark is studying Engineering with concentrations in Mechanical and Electrical Engineering at NNU. Riley was in charge of ISRAEL, the soft robotics experiment. He designed and tested the soft robotics system, and helped on the mechanical system. Over the summer, he started working on a separate project, and he and Juliette didn't have enough time to fully integrate the soft robotics system into the electrical system. Riley will continue working on NNU's Rocksat project as a software engineer.

Sam Mark: MARSHA Electrical Engineer

Sam Mark is studying Engineering with a concentration in Electrical Engineering at NNU. He developed several circuits that were required to control the robotic arms in a closed loop feedback control system. He will continue working on NNU's Rocksat team next year.

Cole McCall: MARSHA Software Developer

Cole is studying Computer Science at NNU. He completed the initial work of researching depth cameras and object detection algorithms.

Jakob Waltman: MARSHA Electrical Engineer

Jakob Walman is studying Engineering with a concentration in Electrical Engineering at NNU. He worked on the camera system and helped with the electrical system. Jakob will be continuing to volunteer on NNU's Rocksat program next year.

Jacob Sutherland: MARSHA Electrical Engineer

Jacob is studying Engineering with a concentration in Electrical Engineering at NNU. He worked on creating the test box so that each line could be turned on or off individually just as is done on the rocket.

5.0 Testing Results

MARSHA Testing

The robotic arm system underwent significant testing throughout the design process to ensure that it was capable of surviving the stresses of launch and performing the complex maneuvers necessary to perform the throw and grasp maneuvers. Early tests of the payload revealed that the motors did not have the necessary torque required to function properly in one G, so after precisely calculating the necessary torque for each joint a gearing system was employed. Figure 25 shows the original mechanical prototype on the right that was quickly replaced in favor of a more heavily geared system, as shown on the left.

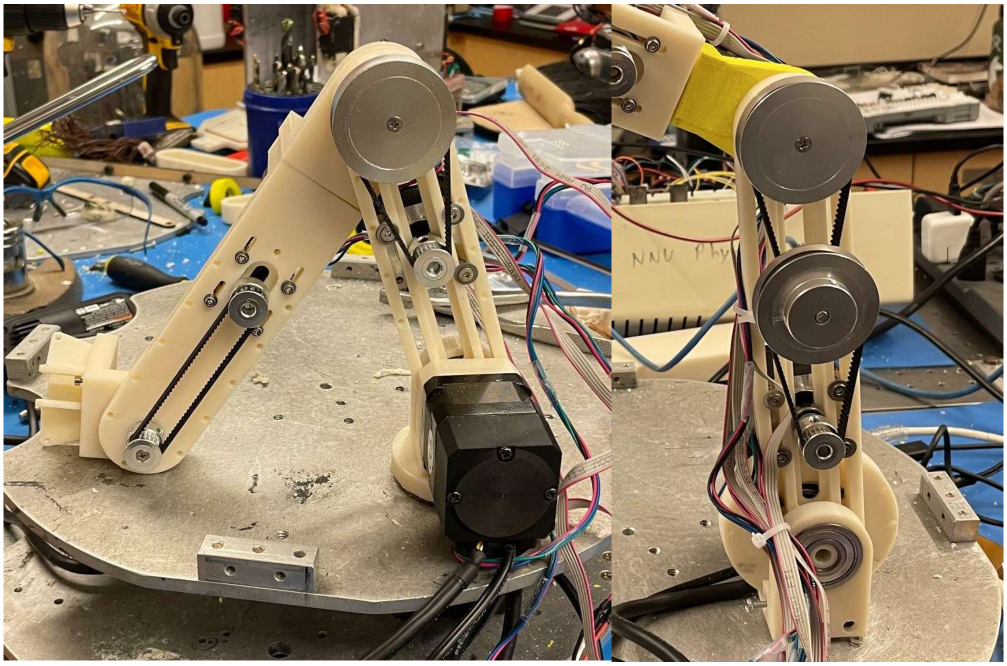


Figure 25 - Robotic Arm Side-By-Side Iteration

Additional testing was performed with a payload deck placed at the correct height above our experiment to ensure that the arms would not make contact with any other plates during the manual deployment and grasp maneuver. This test required the team to make minor adjustments to the position of the stepper motors and change the code to reflect those adjustments. The tests were complete once it was able to reliably deploy and grasp a ball without interference.

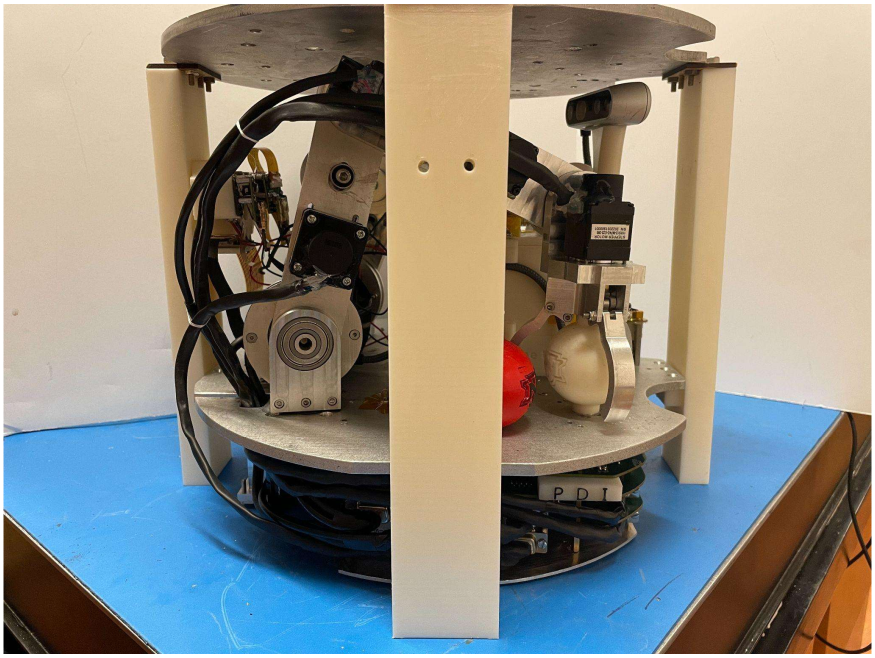


Figure 26 - Robotic Arm With Upper Plate

Software Testing

The AI was tested repeatedly in simulations. The results can be found below in Section 9.0. The mission state machine also went through extensive testing in simulation and was found to perform correctly. Due to time and physical constraints, the AI and mission state machine were not able to be tested outside of simulation.

6.0 Mission Results

Following recovery of the payload, both of the SD Cards were analyzed for mission results and the payload was physically inspected to determine the condition of the mechanical and electrical components.

Post Flight Mechanical Condition

Following the flight and recovery, the payload was in poor condition due to the stresses of reentry. The robotic arms were designed to secure themselves to the mount before the power turned off, but due to a software problem, the arms remained deployed for reentry. This caused the robotic arms to be ripped off at the drive shafts and not be recovered. This was not unexpected, however, as it was known that there would be a low probability that the robotic arms would survive reentry if they were unable to latch. Additionally the majority of insulation on wires was burnt off, and the ISRAEL experiment lost its top plate with the finger attached.

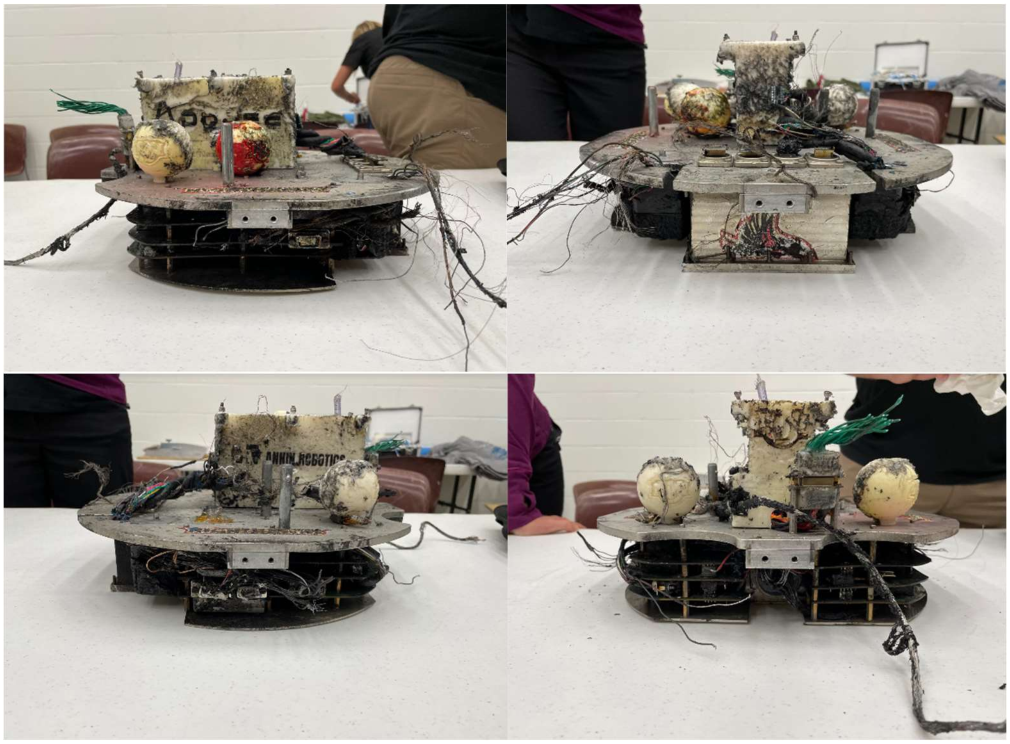


Figure 27 - MARSHA/ISRAEL Post-Flight Orthogonal Views

A part of one of the robotic arms, however, was recovered tangled in the parachute after landing in the ocean.

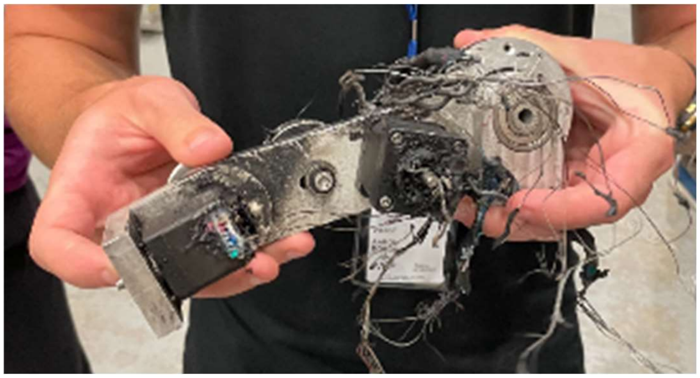


Figure 28 - Robotic Arm Recovered From Parachute

One aspect that was surprising was that the two balls that were not picked up by the robotic arm and were mounted on a pin survived reentry and recovery.

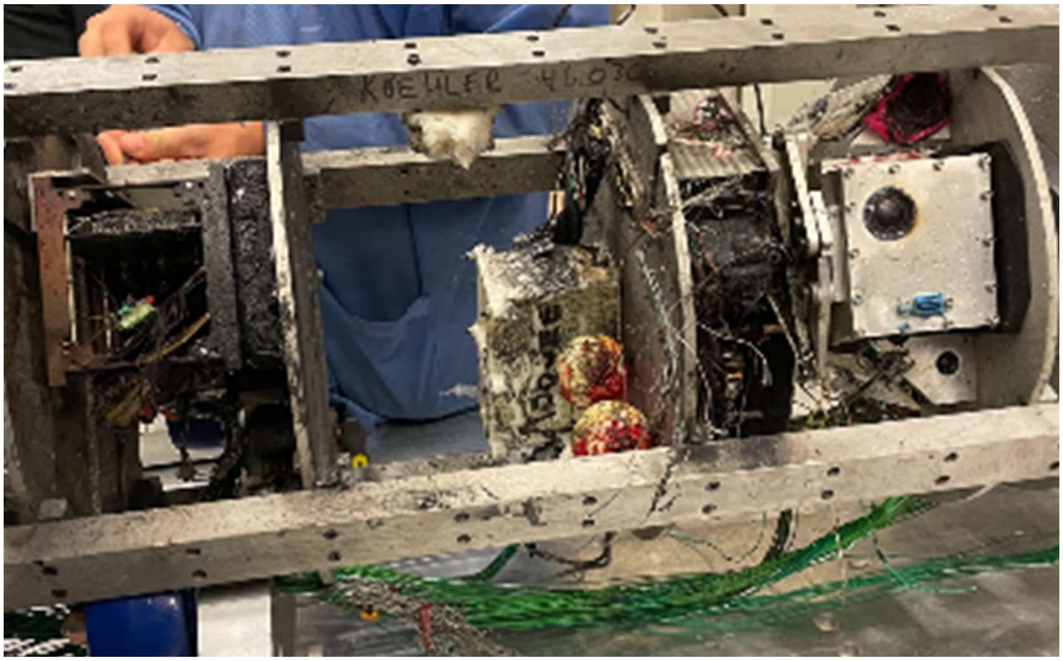


Figure 29 - Balls Still Attached To Payload After Reentry

Post Flight Electrical Component Condition

The electrical system also sustained severe damage. As can be seen in Figures 27, 28, and 30, all the motor wires had been burned or torn off. The damage can also be seen in the motors and board stack.

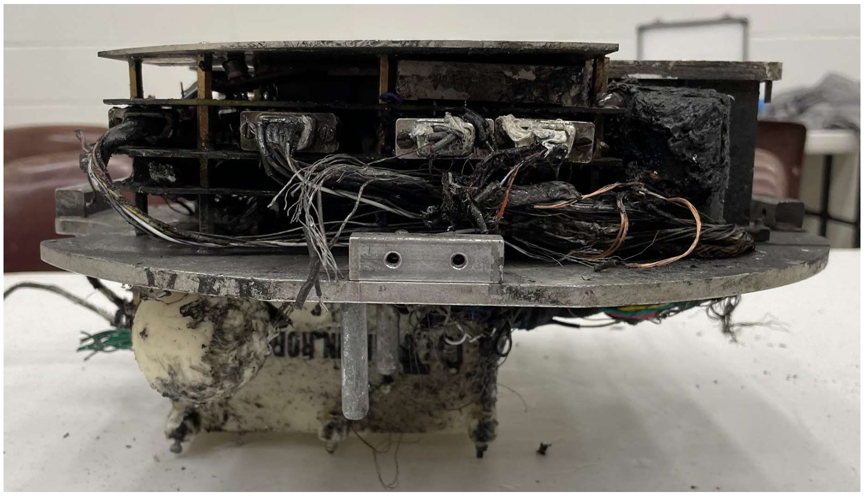


Figure 30 - Electrical system after recovery, side view

After removing the dry box lid, more damage is viewable on the board stacks and in the dry box. This can be seen in Figure 31. As can be seen on the top left, one motor controller fell off during reentry, most likely due to the extreme heat. There was also lots of water in the dry box. This was due to the o-ring not gluing together properly and therefore creating a significant point of failure.The Jetson Nano's will not be able to be reused, but the SD Cards were recovered safely and were still functioning for the data to be collected.

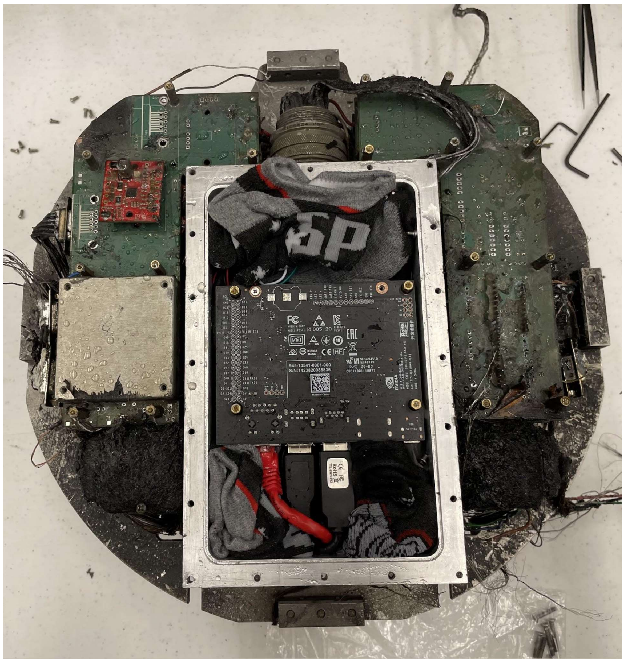


Figure 31 - Electrical system after recovery, bottom view

Mission Results

After analyzing the SD Cards, it was determined that no video footage was captured. Due to labor shortage and time constraints, the longeron cameras were not completed and were not set up to capture footage during the mission. Based on the logs, the depth camera was running during the mission and was able to detect the ball's position, however, the video file was corrupted and could not be recovered. Based on prior experience, the footage will corrupt if it is not properly saved before power is lost. Originally, an interrupt was going to be used to save the footage when the TE-R line deactivated which would have triggered the saving command and allow the footage to finish saving before power off. However, this method was descoped and instead the footage was programmed to save after the arms latched. Through more testing it would have discovered that this was not a reliable way to trigger saving.

During the mission, the state machine transitions for both arms were logged which makes it possible to determine what occurred during the mission:

The TE-R line was detected successfully at the specified time. This triggered the right arm to unlatch and fully deploy and await a ball to be thrown. Meanwhile the left arm unlatched and successfully picked up the first ball. The left arm then successfully deployed and threw the first ball. The degree of success for the throw is unknown due to the failure of the cameras. However, we do know the ball was grasped, the arm deployed, and un-grasped the ball while moving towards the other arm. In addition, the University of Hawaii Community Colleges captured footage of the ball after it was released as seen in Figure 32.

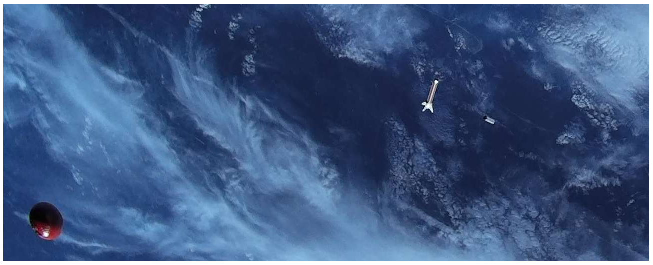


Figure 32 - Ball after being thrown.

The right arm did not catch the ball, but the degree of the catch failure is unknown due to the camera failure. After the attempted throw and catch, the left arm attempted to retract, but was unable to. This could be due to the arm not moving to the intended system because the manual retraction was not tested rigorously. Because the arm was unable to retract, the mission was effectively ended as the arm was programmed to give up if it was unable to move as planned. This was to protect the integrity of the rocket in-case it had collided with the longerons or one of the base plates, however, the motors would not have enough torque to cause any damage. With more testing it could have realized this was not the ideal way for the arms to react to this event, instead the left arm should have deployed again and then attempted to retract again. If the left arm was still unable to retract, then that ball should have been skipped and the arms should have transitioned to the state where the right arm grabs and throws a ball to the left arm.

There was a failsafe implemented wherein the arms would retract and latch no matter what state they were in. The failsafe was successfully triggered, and the right arm began to retract, however, the retraction time was miscalculated, so it did not finish latch before power off. The left arm was still unable to retract and was in the deployed position during re-entry.

7.0 Conclusions

While the results from the launch were not desirable, the arms were successful in deploying and throwing a ball. Also, the lessons learned throughout the development of the system were invaluable.

The longeron cameras did not work due to the complicated system. The data had to go through two ESD sensitive converters (to and from HDMI) and an HDMI High Vac connector. This brought many more points of failure into the system. Many cameras and convertors had to be bought due to the easy shorts in the ESD sensitive convertors or cameras. The cameras also required ribbon cables. These cables had very thin pads that required them to be lined up perfectly, and if they were not, it could short the camera. The ribbon cables were also extremely easy to crease which could cause the lines inside it to become disconnected and unable to transmit data. All these failure points made the camera system extremely hard to debug and the team over the summer didn't have the time to spend debugging.

The o-ring failed during the flight as well. The cord stock required being cut and glued to the correct length. This process ended up requiring a specific glue that the team did not have access to. The o-ring failed because it was unable to make a proper glue bond to seal in the dry box.

Our team was attempting to accomplish novel, graduate level work in one year with 90% of the work being completed by three team members. At the beginning of this project, no one on our team had experience with robotics, yet the team was still able to create a set of robotic arms that were able to function in microgravity. After the results from this project, we now have the knowledge that will allow us to develop and test multiple iterations before another launch next August.

8.0 Potential Follow-on Work

The lessons learned from this year will allow us to develop a more robust version in following years. In addition, it will be possible to develop it quicker which will allow for more iterations and testing. The team will continue by fixing the torque, cameras, mission state machine, and electrical development process issues that prevented a successful catch during the mission. While the arms were unable to catch a ball during the mission, it was shown the arms could catch the ball in simulation with a 70% success rate. This is sufficient to demonstrate it is possible to develop an AI that can control robotic arms to catch a ball. In addition to fixing the mentioned issues, the NNU Rocksat team will continue by developing a new AI algorithm and gripper capable of catching different shaped objects. The new algorithm will include detecting an object, learning the object's motion dynamics from a 3D point cloud, predicting the best time in the future to grasp it, and the best grasp configuration to grasp that specific object. The new method will feature new machine learning such as the transformer model introduced by Google Brain, the 6-DOF GraspNet developed by Nvidia, and the TD3 algorithm provided by OpenAI's Spinning Up educational resource.

A major goal for future work on this project involves following the requirements stated in airborne standards such as DO-178. While the team was not able to entirely follow DO-178, the NNU software engineers would like to develop the software in a way so the robotic arms may eventually be certified to fly on crewed missions. This adds an additional challenge as no machine learning algorithm has been successfully certified to be used on an airborne system, however we believe the TD3 algorithm will allow us to develop a deterministic system, and training and testing in simulation will allow us to sufficiently verify and validate that the system follows all safety requirements. Since the code already uses ROS to control the robotic arms, it is planned to eventually use Space ROS which is being developed to be used in mission-critical applications.

Accomplishing this goal of developing an AI controlled robotic arm system capable of catching different shaped objects in space would result in a technology that would be beneficial towards manufacturing and servicing spacecraft in orbit and removing space debris.

9.0 Benefits to the Scientific Community

Although not much data was received during the mission, the team was able to develop technologies that will be beneficial to the scientific community while designing and testing this year's payload. It was shown that computer vision can effectively be used to detect and track objects despite having a similar colored background as shown in Figure 32.

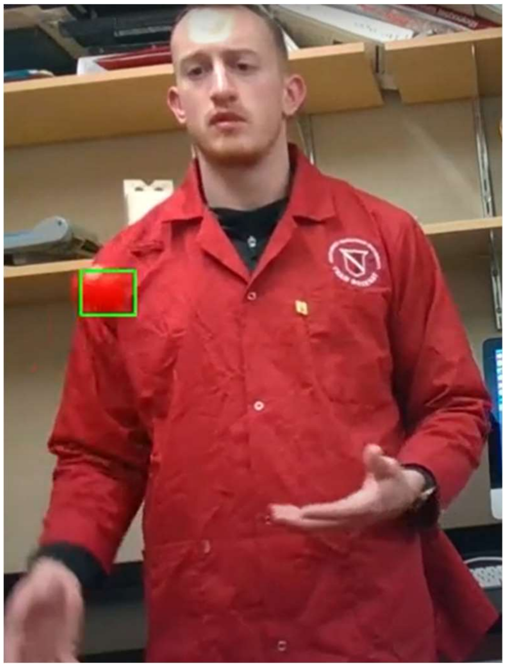
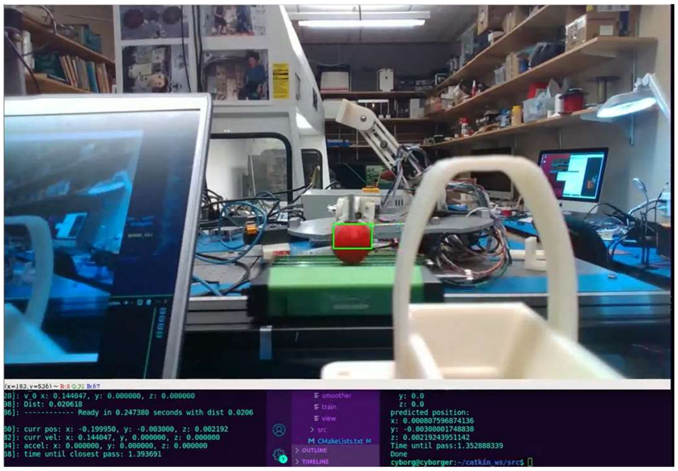


Figure 32 - Ball Detection with Same Colored Background

The team learned that a Kalman Filter could be used to estimate the current position and future position of an object from the inaccurate and uncertain measurements from the object detection system.



```
28]: v_0 x: 0.144047, y: 0.000000, z: 0.000000
98]: Dist: 0.020618
86]: ------------ Ready in 0.247380 seconds with dist 0.0206

50]: curr pos: x: -0.199950, y: -0.003000, z: 0.002192
82]: curr vel: x: 0.144047, y, 0.000000, z: 0.000000
94]: accel: x: 0.000000, y: 0.000000, z: 0.000000
58]: time until closest pass: 1.393691
```

```
  y: 0.0
  z: 0.0
predicted position:
x: 0.000807596874136
y: -0.00300001748838
z: 0.00219243951142
Time until pass:1.352888339
Done
cyborg@cyborger:~/catkin_ws/src$ 
```

Figure 33- Kalman Filter Predicts When the Cart Passes the Line

It was shown that a variant of the binary search algorithm could be used to find the time at which an object on a non-linear trajectory is closest to a fixed point. While this wasn't used in this mission because the ball was on a linear trajectory, this will be useful in any case where the robotic arm is attempting to catch an object in gravity or while attached to an accelerating object. Table 1 shows the trajectory of an object on a non-linear path. Table 2 shows the distance between the object and a fixed point at the origin. Assuming the base of the robotic arm is at the origin, it is possible to calculate if the object is within reach, and provide this information as an input to the machine learning algorithm.

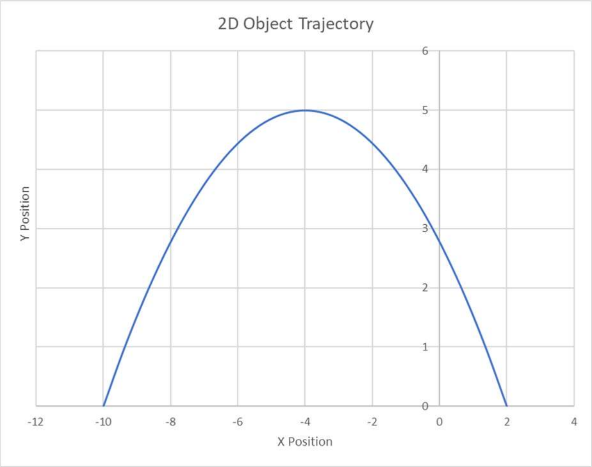


Table 1 - Path of an object experiencing acceleration

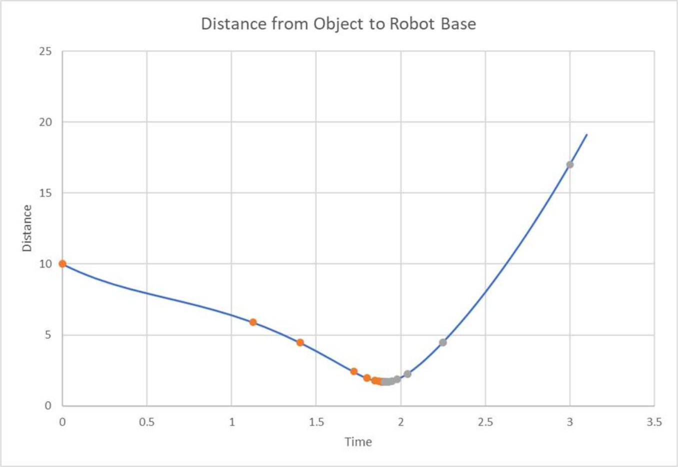


Table 2 - Binary Algorithm Converging on
the time the object is closest to a fixed point

Additionally, it was shown that the AI algorithm could successfully control a robotic arm to catch a ball while avoiding colliding with the environment in a simulated zero-gravity environment. The testing resulted in a successful catch occurring 70% of the time after 6,000 attempts. Table 1 shows one of the tests where this occurred. The light blue line shows the reward during each episode. While the reward is less than 0, the robotic arm was unable to move to the desired grasp because it was either kinematically impossible, or would have experienced a collision. Each time the reward spikes to a value of 10, a successful catch has occurred. The dark blue line shows a moving average of the reward to demonstrate that the catch rate is improving throughout training.

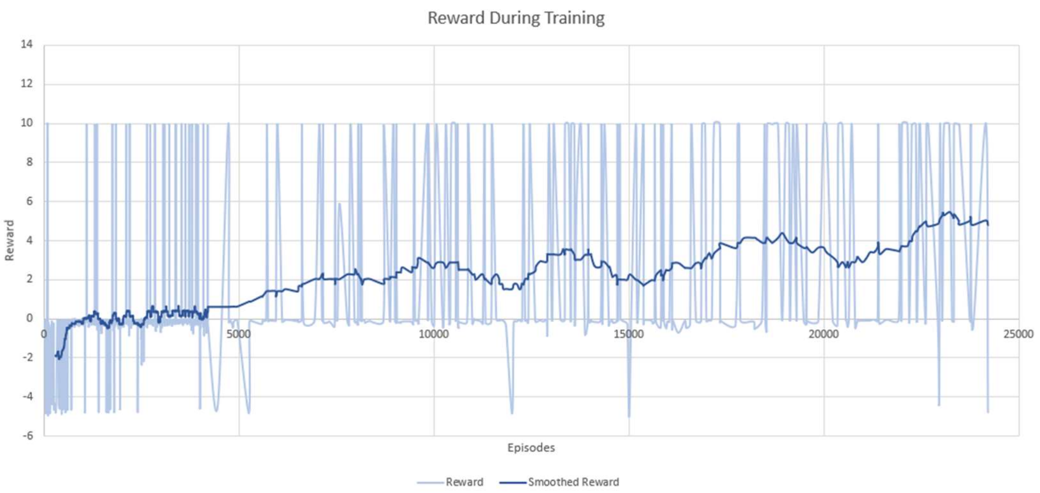


Table 3 - Reward Increases Throughout Training

Table 3 shows another test that demonstrated the AI algorithm was able to learn kinematic possible grasps. During the exploration phase, the AI algorithm uses random grasp configurations to attempt to catch the ball. Most of these configurations are not kinematically possible for the arm to move to. However, after the exploration phase, learning begins and the AI agent uses the reward it has been given during the exploration phase to determine which grasp configurations are kinematically possible. Therefore, instead of taking multiple motion planning attempts, the agent has learned to select grasp configurations that are kinematically possible on the first try most of the time.

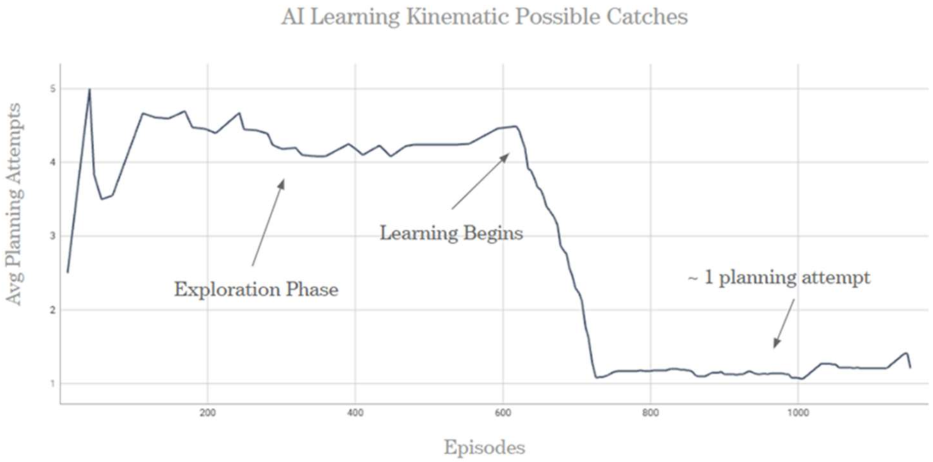


Table 4 - AI Learning Kinematic Possible Catches

Additionally, it was proven that a set of highly-capable robotic arms could be made compact enough to fit into a small payload, and robust enough to survive the forces of launch. This is the first of its kind in the industry and will pave the way for the advancement of space robotics in the future.

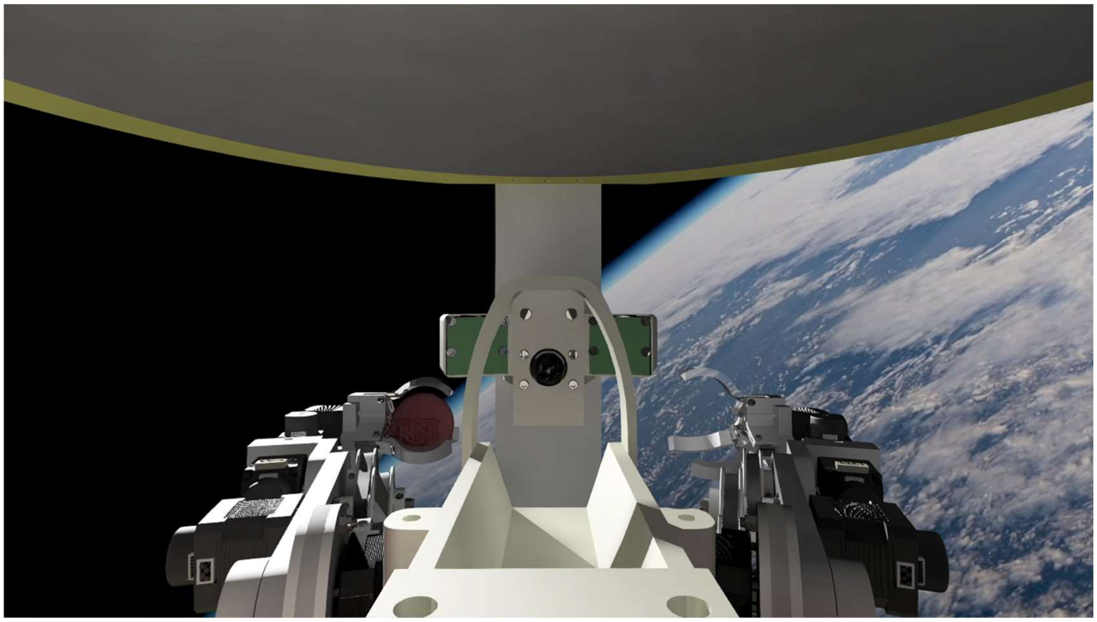


Figure 34 - Rendering of Robotic Arms Operating in Space

10.0 Lessons Learned

Throughout the development of the payload and after analysis of the flight data, many lessons have been learned. These lessons will be applied in future years to ensure that the team has the maximum chance for success.

The most important lesson learned is to have the electrical system prototyped and tested as early on in the design progress as possible. The electrical system is integral to testing the motors and the mechanical systems and to finishing the code. To do this, the lead electrical engineer needs to take initiative to spend the time needed on the project to have the system ready well in advance. In future years, the first electrical iteration, made with professionally manufactured PCBs, should be finished before the end of January. This gives a semester to prototype on breadboards. With the first iteration done early, there is still plenty of time to make changes, refine the system, and get another iteration or two manufactured before the end of the year. Getting the first iteration done early also will allow for more testing of both the mechanical and computer subsystems.

A design improvement to be made in the future is to use a prefabricated o-ring instead of cord stock. The o-ring did not glue together properly which allowed water into the dry box. In future years, cord stock should not be used due to complications caused by the process of gluing the o-ring.

The camera system was unable to be completed due to the complexity of the system. To simplify this process in the future, cameras with wires should be used instead of those with ribbon cables. As mentioned above, if the cables are plugged in crooked the cable isn't able to transfer data correctly and it can even short the camera. Using wires instead will take out the risk of it being plugged in incorrectly. It will also allow for the wires to go through the high vac connector instead of requiring 3 extra HDMI high vac connectors.

Due to the significant delays in the electrical system, the final mechanical assembly was unable to be rigorously tested. Although torque calculations were performed for each link of the arm, there were additional stresses on the arm and motors that had not been anticipated. This resulted in the arm not performing exactly the same way between tests and would often cause links to slip from not having enough torque. The torque was calculated without significant margins to account for the forces such as friction in the gears. In the future, this will be mitigated by increasing the margins for torque and having a more refined gearbox design that more efficiently transfers the energy from the drive shaft to the arm link. This will allow for an increase in the precision and reliability of the system for testing in one G.

11.0 Appendices

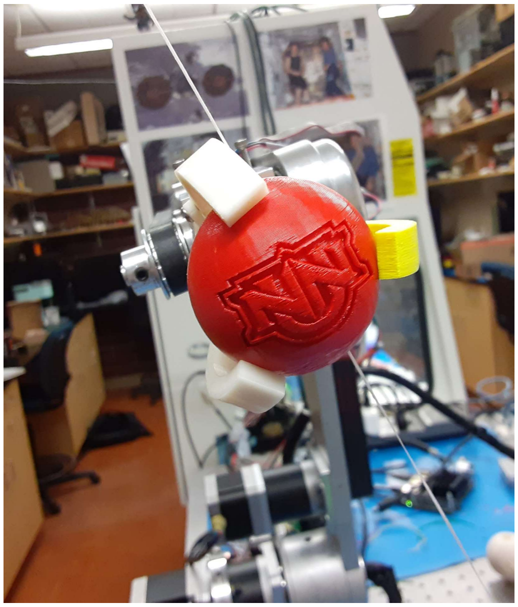


Figure 35 - Ball Caught With AR3 Test Robotic Manipulator

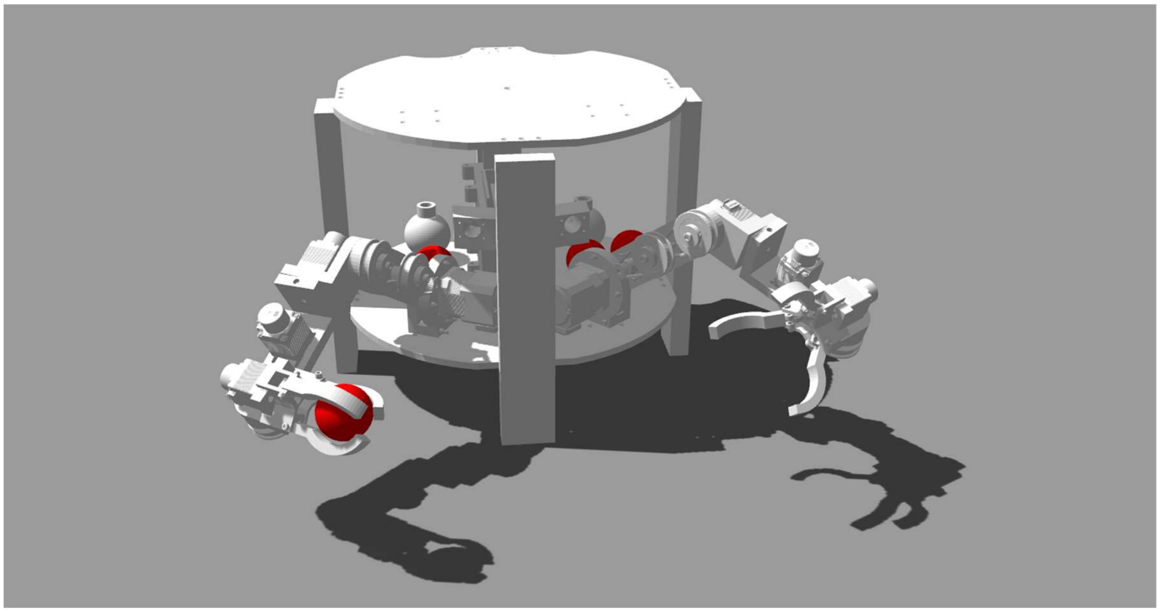


Figure 36 - Robotic Arms Training in Simulation

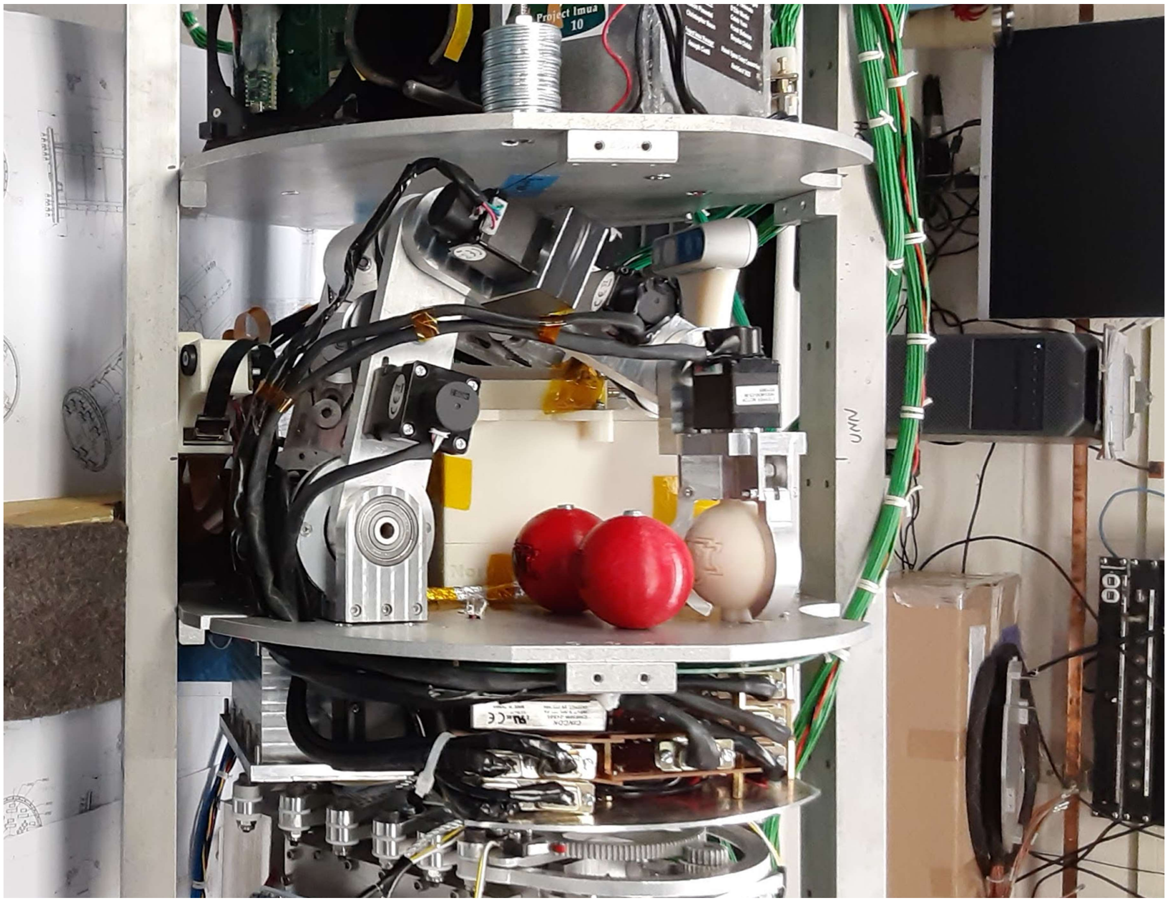


Figure 37 - Integrated Payload at NASA's Wallops Flight Facility

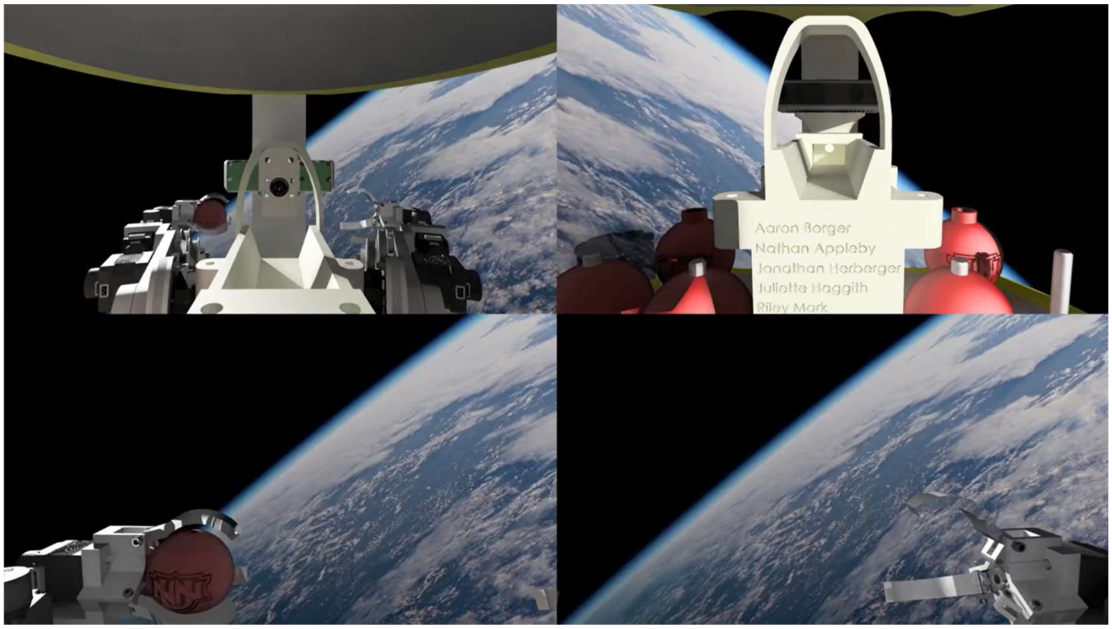


Figure 38 - Rendering of Onboard Camera Views During Flight