Development of Cube Swarm for Search and Rescue Applications

A Major Qualifying Project submitted to the Faculty of

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the degree of Bachelor of Science by

William Albert (RBE), Phillip Brush (RBE/ME), Benjamin Dodge (ECE), Timothy Klein (RBE/CS), Andrew McCammon (ME), Jason Rockmael (RBE), Dang Tran (RBE)

Project Advisors:

Professor Gregory Lewin (RBE/ME), Professor Shubbhi Taneja (CS), Professor Reinhold Ludwig (ECE)

Worcester Polytechnic Institute

Abstract

Current search and rescue robots suffer from being either too large to fit into tight spaces or too small to traverse terrain. To overcome this challenge, we developed a swarm of interlocking, cube-shaped robots that allow the robots to explore individually and traverse obstacles while connected. We demonstrated our solution using a centralized Bluetooth communication network, AprilTag localization, and an efficient path-planning algorithm. Our robots are able to navigate an arena without collisions, and then assemble to bridge a gap.

Table of Contents

Table of Figures

Table of Tables

Chapter 1: Introduction

After a natural disaster, search and rescue (SAR) teams spend several days exploring survivable voids left in rubble (Arranz et al., 2023). To save as many survivors as possible, qualified personnel are responsible for balancing the tasks of searching for more survivors and rescuing the ones they have already found. A key challenge in search and rescue is locating these survivors within a limited timeframe. In addition, some disaster sites are too dangerous for people to go into. Stretching already limited resources thin necessitates the development of more advanced tools for locating survivors, enabling relevant personnel to concentrate on saving lives. Robots are capable of aiding or replacing human workers in these situations. A swarm of robots could traverse this type of environment and autonomously locate survivors, addressing the struggle of allocating resources in similar situations. This project intends to demonstrate the ability of swarm robotics to solve problems and overcome obstacles that traditional SAR robots could not. This project will serve as a foundation for further research on swarm robotics for search and rescue applications.

To demonstrate a successful system, several agents will explore their environment, connect to one another, and then cross a gap. Fundamentally, agents will demonstrate the ability to both work independently and work together as a system.

Chapter 2: Background

2.1 Search and Rescue Robots

The value of search and rescue robots is to traverse environments that humans cannot due to size and safety concerns. SAR robots can be autonomous or teleoperated and have the capability to identify humans, hazardous materials, and flaws in a building's structural integrity and establish lines of communication. Search and rescue scenarios inherently have a level of uncertainty until a thorough assessment of the situation has been conducted. Therefore, it is important that robots are adaptable to the environment and potential damage (Delmerico et al., 2019). However, current search and rescue robots may suffer from being either too large to fit into tight spaces or too small to traverse complex terrain.

2.2 Benefits of Swarms

A swarm of robots refers to a group of robots that can work collaboratively with one another in order to achieve a more complex goal than an individual robot is capable of completing (Murphy, 2004). A swarm has the potential to mitigate the shortcomings of a singular SAR robot by providing a more cost effective and adaptable system with smaller individual robots. Consider a scenario where a swarm is searching a site and one of the robots becomes damaged and unusable. Having multiple robots on site means if one is lost, another robot can replace it, exemplifying the adaptable nature of swarm robotic systems (Patil et al., 2013). Furthermore, by using a swarm, each robot can be allocated a sub-area within the wider area, allowing a greater total coverage of the area. Most swarm robots are typically

smaller than individual SAR robots, which allow for navigation in smaller spaces, but make obstacles harder to overcome.

A swarm of self-configurable, modular robots enable dynamic changes and adaptability to diverse tasks. The ATRON project illustrates the transformative capabilities of modular robots (Brandt et al., 2007). This system explores modular design principles, application versatility, and innovative control strategies, demonstrating the potential of self-reconfigurable robots in real-world scenarios. Traditional robots are limited in the variety of tasks they can perform by their fixed morphologies. The ATRON project removes this limitation by introducing a swarm of modular robots capable of autonomously changing configurations. Each ATRON module possesses basic capabilities such as connection, disconnection, communication, sensing, and rotational movement. ATRON robots exhibit cluster flow locomotion, a unique feature where modules move within the cluster, leading to collective movement. While cluster flow may be slower than fixed morphology locomotion, it offers adaptability to varying environments; however, the drawback to this functionality is that individual modules within the system are limited in their mobility. While limited in individual functionality, these modules form the building blocks for creating versatile robotic swarms.

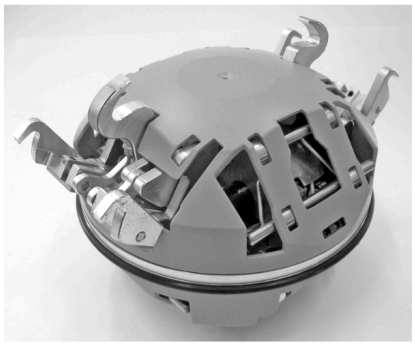

Figure 1: A single ATRON module. (Stoy et al., 2010)

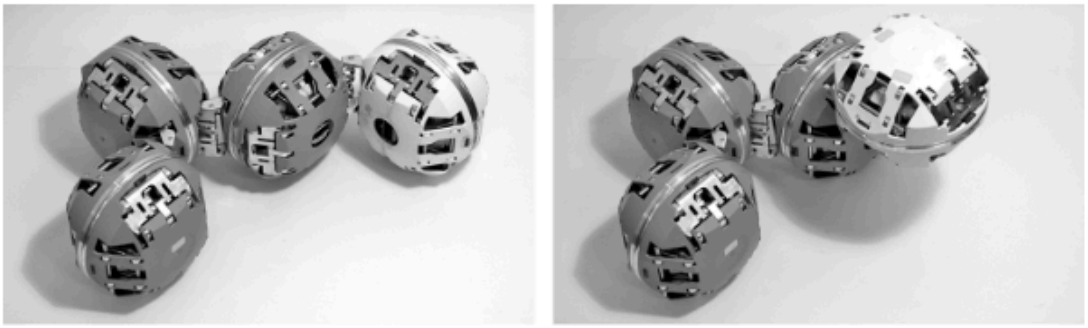

Figure 2: ATRON modules connecting to each other to form complex structures. (Stoy et al., 2010)

2.3 Limitations of Swarms

Despite its benefits, swarm robotics also comes with its respective set of limitations and challenges that need to be dealt with. The main challenge lies in their scalability and complexity management. For the swarm to be more cost-effective than small individual robots, each robot within the swarm needs to be low-cost while maintaining sufficient

functionality. Additionally, if the system is not properly designed, the possibility of failure will increase as the number of robots increases due to error propagation. Another challenge was ensuring robustness and fault tolerance, as the system must be designed to accomplish its task even if individual members are lost.

2.4 Project Objectives

In this project, the aim is to demonstrate the suitability of swarm robotics in SAR applications. There are five main design goals to be fulfilled in order to consider the swarm successful. Individual agents will be of minimal size and weight, as well as easily manufacturable. Agents will be able to communicate their current and desired positions with one another. Agents will be able to localize, or self locate, with respect to the environment and other robots. Agents will path-plan such that they can line up for autonomous connection to other agents without collisions. Finally, agents will autonomously cross a three robot wide gap by locking together to form a bridge.

Chapter 3: Design and Development

3.1 System Overview

The development of a swarm of robots that autonomously assemble demonstrated the value of using robot swarms in SAR missions. To consider this swarm a success, the proposed demonstration is shown in Figure 3. First, seven 6x6x6 inch robots will autonomously explore their environment independently until one or more robots find a global landmark. Next, the robots will then drive to a position in which they can line up to connect to one another. Then, the robots will connect and drive as a unified system across a three robot wide gap. Finally, the robots will disassemble and individually explore the environment on the other side of the gap.

Figure 3: Storyboard of Final Demonstration

In order to accomplish this task, the system's functionality can be broken down into five main systems: locking, communication, localization, locomotion, and path planning. A gripper was developed for locking that attaches to a receiver on a neighboring robot. Bluetooth was used to communicate between each robot’s Raspberry Pi Zero 2 W. For vision,

each robot was equipped with an Arducam. AprilTags were placed on the faces of each robot to allow other robots to get their relative distances. Two different size AprilTags were used on the robots. The larger 6x6 inch AprilTag was used for localization while the smaller 1x1 inch AprilTag was used for lining up the robots for locking. AprilTags were also used as global landmarks to allow robots to localize within the world. In order to navigate in the environment, each robot has a differential drive base and a multi-agent path planner to find and eliminate collisions between robots.

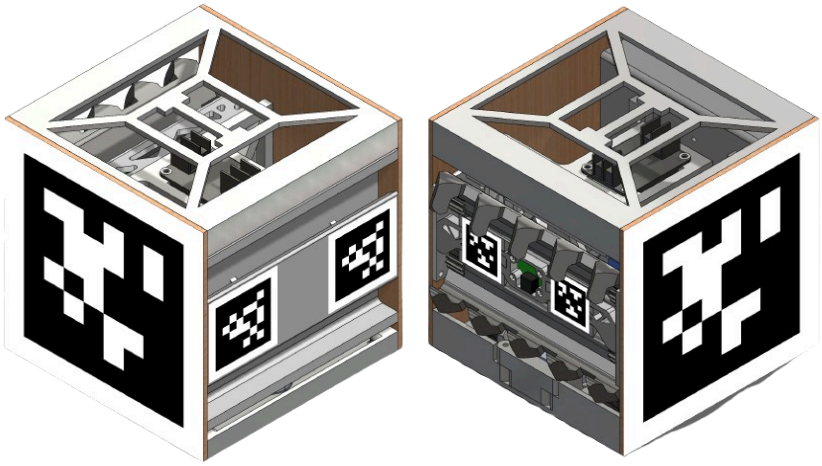

Figure 4: Cube Robot

3.2 Electronics

Firstly, each agent needed a platform to work off of. To this end, each agent was equipped with a Raspberry Pi Zero 2W, an Arducam OV5647, Gaoneng 850mAh 3S LiPo battery (3 cell series-configured Lithium-Polymer battery), a DRF 1025 buck converter, and a printed circuit board (PCB). Additionally, the robot uses a DRV 8835 motor controller, two MG90D servos, and two ROB 16413 hobby motors.

3.2.1 Raspberry Pi Zero 2W

Each agent needed an onboard method to control its motors and sensors. To accomplish this task, a typical solution would be a microcontroller. However, due to the heavy processing demand required by the Arducam and other processing needs, such as path-planning, each robot was equipped with a Raspberry Pi Zero 2W. This device is a system-in-package, which can be considered a micro-computer, rather than a microcontroller. Some other benefits of the Zero 2W were: the built-in Bluetooth and Wi-Fi chips, the quantity of general purpose input/output (GPIO) pins, and the ribbon cable connector for the Arducam.

These swarm robots needed greater processing power than those of other swarm robots because each agent needed the ability to do image processing calculations among other calculations. The onboard Wi-Fi and Bluetooth chips on the Zero 2W satisfied the need to communicate with other robots and the server. The Zero 2W needed to interface with the motors, servos, and sensors. The quantity of these sensors required a controller with sufficient

GPIO ports, which the Zero 2W accommodated. The built-in ribbon cable connector simplified camera connections as well.

3.2.2 Arducam

The Arducam OV5647 was a clear choice of camera because of its affordable price, widely available support, and compatibility with the Pi Zero 2W. Price was a major consideration to make each agent more manufacturable and Arducams are widely available for as little as $7. The OV5647 met the functionality requirements for this application as well. This camera also had a large amount of support, because it is a more widely used alternative to the Pi Camera (Raspberry Pi brand). Lastly, the OV5647 was proven to be compatible with Zero 2W because of the aforementioned ribbon cable connector. These considerations made the OV5647 a more suitable choice than other comparable options, like the ESP32-Cam.

3.2.3 Power Distribution

Two key challenges for this project were power supply and distribution. The choice for the power supply was made by considering the highest voltage required and finding a battery with suitable power capacity. The DRV8835 motor driver paired with the ROB-16413 could comfortably be operated at 11.1 volts. Therefore, a 3S LiPo with a nominal charge of 11.7 volts and a maximum charge of 12.6 volts, met this requirement along with energy density benefits. The Pi Zero 2W, servos, and motor encoders required five volts so the battery voltage had to be stepped down. To accomplish this, the robots have a DRF1025 selectable output DC-DC buck converter. In the future, this would be replaced with a tailor-made DC-DC converter.

3.2.4 Printed Circuit Board

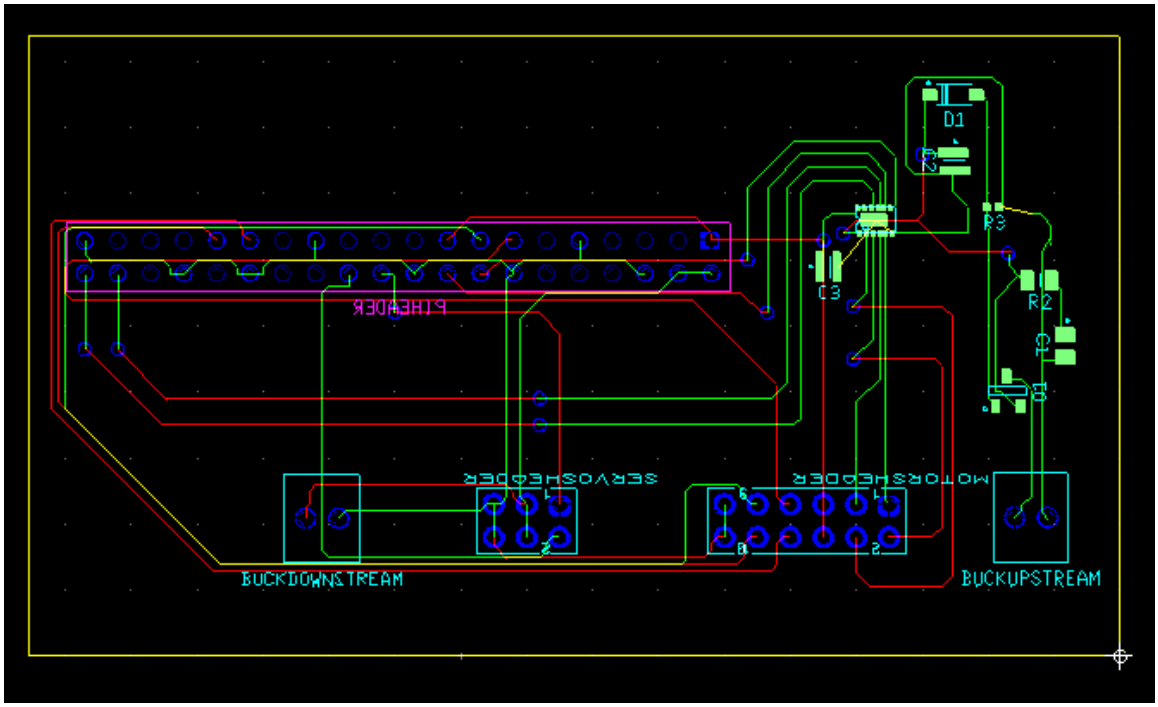

Figure 5: Ultiboard Schematic of PCB with top layer ground plane removed

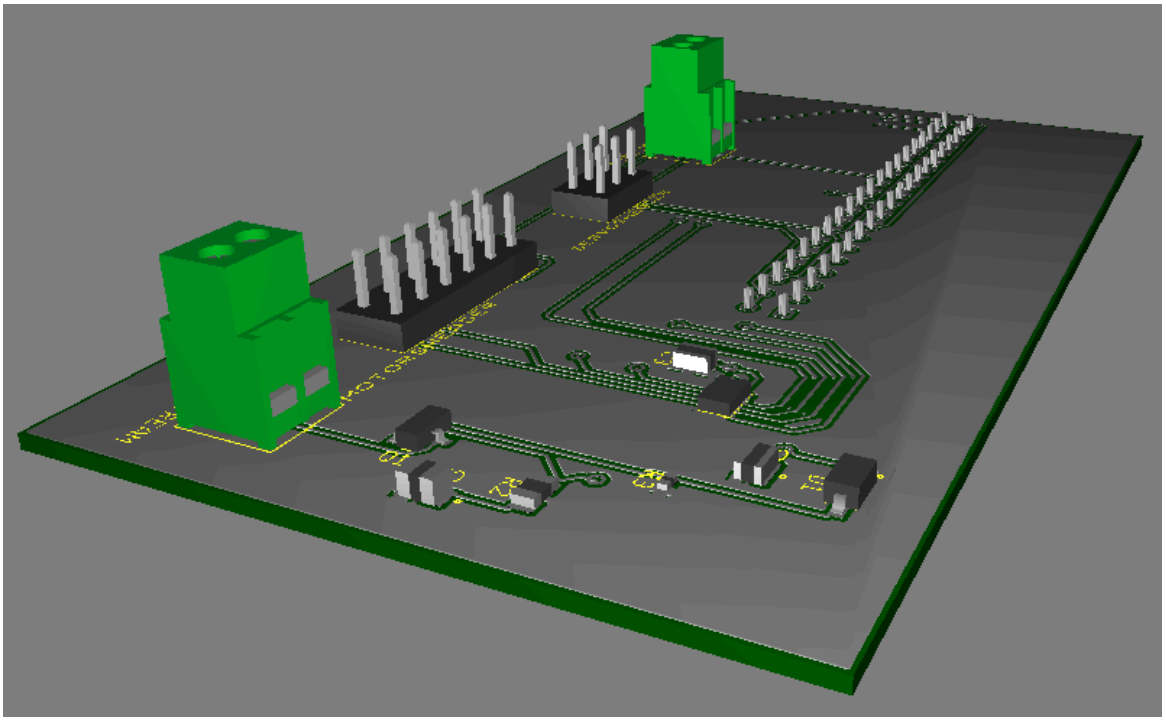

Figure 6: 3D rendering of PCB using Ultiboard

Each agent needed efficient electronics management in order to minimize the space taken by electronic components. The main benefit of the PCB was progress towards a well-built system; however, some added benefits were the simplification of the design by moving the motor driver and its reverse polarity protection circuit onto the PCB, a slight lowering of total cost and improving manufacturability, and improved heat dissipation for the motor driver.

a.) Protection Circuitry

The PCB shown in Figures 5 and 6 served as an alternate way to connect the components together. It accomplished this with pin headers for the servos and motors which allowed them to communicate with the Pi which was attached with another 40-pin header. The PCB was intentionally designed without mounting points so that it could be mounted as a Pi hat.

Caution had to be used when dealing with the 850mAh 3S LiPo as it had both a high voltage and extremely high continuous discharge rate of 120c: roughly 100A at 11.1V or 1.1 kW which is more than enough to cause serious damage to components. To deal with this there was a reverse polarity protection circuit integrated onto the PCB in conjunction with the on-board protection circuit on the DRV 8835 motor driver. This reverse polarity circuit was relatively simple with just a P-channel Mosfet, a reverse biased zener diode, and a resistor. It was configured such that the Mosfet's body diode was forward biased during normal operating conditions and had a high gate voltage which stopped current flow if wired in reverse. This was critical for protecting the load from improper wiring and also acts as a

backup protection circuit for induced current from the motors if the protection circuits on the DRV 8835 were to fail.

b.) Cost Saving and Manufacturability

A PCB was made during the initial prototyping phase in order to consolidate functional components onto one board. Integrating the DRV 8835 motor controller breakout board decreases the number of subcircuits needed to be manufactured individually and allows for some wired connections to be replaced with PCB traces.

In terms of cost, the main detriment of the breadboard approach is sourcing components from multiple suppliers: shipping small volume orders greatly increases overall cost. The PCB has an obvious advantage in this regard, as most individual electronic components can be sourced from one supplier (or even from the PCB manufacturer in the case of this project). This greatly decreases overall cost associated with shipping and manufacturing, and further decreases the cost when the PCBs are printed in bulk.

c.) Heat Dissipation

The PCB was designed with heat dissipation in mind; specifically, the DRV 8835's center pad was grounded to the ground plane to dissipate the heat of the component more efficiently. This was necessary because the DRV 8835 has thermal shutdown protection which shuts the integrated circuit off if it exceeds 150º C. In this regard, the PCB is an improvement over the breakout board as the larger ground plane offers improved heat dissipation.

3.3 Autonomous Connection

3.3.1 Prototyping

To cross from one side of a gap to the other, the robots will connect in a line using a gripping mechanism to attach to an adjacent robot's receiving end. A series of prototypes highlighted issues with initial designs and informed the design criteria for the final gripper. The locking mechanism must be able to lock together if perceived position and physical position are misaligned. Loads due to cantilevering shall be borne by the components and not require a significant torque from the servo motors when locked in place. The gripper and receiver must also be robust in that cubes or locking components can withstand the loads from bridging. The gripper must not block the camera's view, which is necessary for localization. The mechanism must sense when one cube has connected to another and communicate this information back and forth to confirm the cubes are locked together and ready to bridge.

The initial designs, seen in Figure 7, explored different methods of attachment, including a screw and hole, as well as gripper arms grabbing a bar, similar to but not as robust as the final design. The prototypes presented issues with accuracy of localization in order to function, the required torque output from actuators, and structural integrity. Two preliminary designs developed alternate methods of actuation to reduce the number of servo motors. One design uses gripper arms powered by one motor and a pair of slots to achieve grabbing motion seen on the right of Figure 7. The second design uses a gear train, left of Figure 7, to

actuate both arms allowing for controlled and smooth movement while minimizing the number of actuators. Upon 3D printing prototypes, the slots created too much friction, and tolerancing the slots to minimize play without increasing friction was infeasible. Preliminary 3D prints, seen in Figure 8, also gave insight into how the geometry between the arms and receiver worked for vertical tolerancing and potential for unintentional detachment.

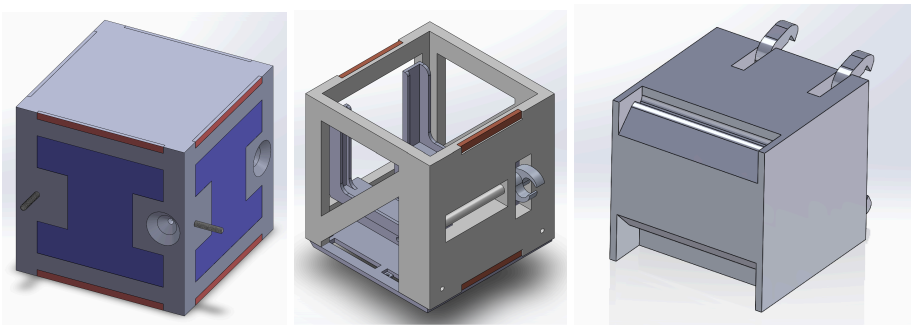

Figure 7: Connection Mechanism Prototype Designs

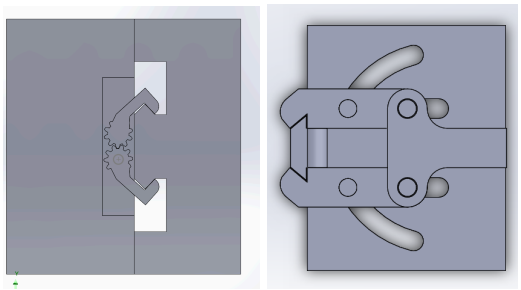

Figure 8: Flat Prototype Designs

Large and heavy actuators, which provide higher torque for the gripper, were not feasible to reduce weight, footprint in the cube, and cost. To avoid heavier actuators, the gripper and receiver geometry design ensured the force imparted on the receiver from the gripper was inline with the point of rotation of the gripper arm. Figure 9 highlights where the locking mechanism experiences external forces. While each design presented pros and cons, each one was susceptible to failing if localization was inaccurate and imprecise. Thus, having the gripper's success be independent of other subsystems became the most significant design constraint.

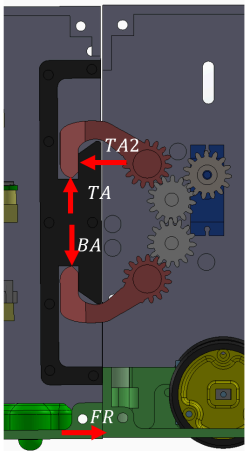

Figure 9: External Forces on the Locking Mechanism and Cube

One of the final locking and gripper designs seen in Figure 10, has similar action to grapples used on excavators. where one side has a gripper that grabs the receiver of an adjacent robot allowing more vertical and horizontal tolerance for localization. The arms of the gripper are actuated by two MG90D servo motors through a gear train. Two limit switches are embedded in the receiver communicating the robots have successfully attached to each other, creating a closed-loop system. The last component of the gripper assembly to meet design constraints is the camera mount. While not critical to the movement of the gripper, the locking design had to account for the camera to be able to see.

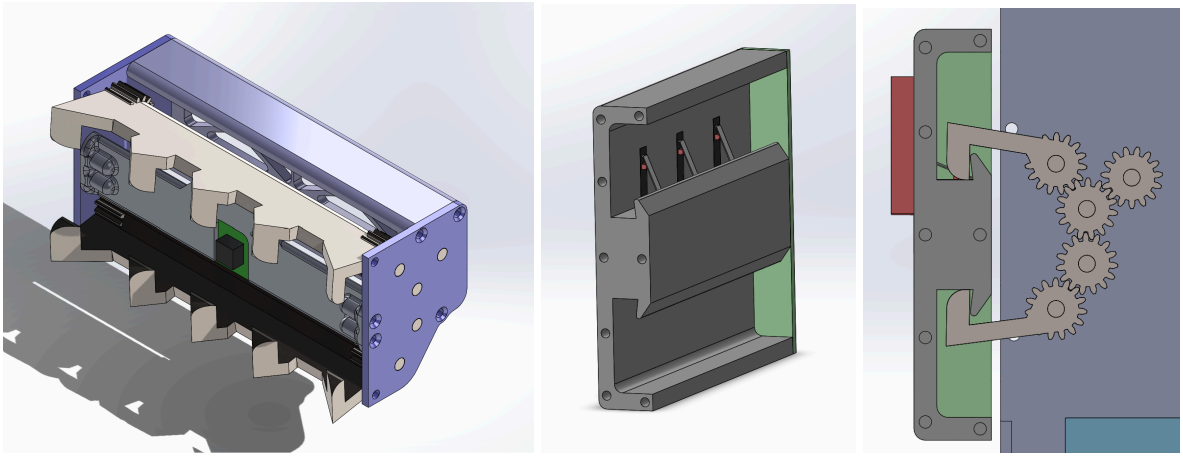

Figure 10: Prototype Autonomous Connection Mechanism

The final gear train uses parallel construction with 1:1 gear ratios and a servo to actuate the gripper arms from each side. To ensure the components held the load during bridging and not the actuators, the locking mechanism, by design, imparted minimal torque on the servo. Therefore having a high gear ratio or bigger, heavy actuators with a greater torque output were unnecessary. Additionally, using motors with a higher torque would have consequently required the gripper be larger Despite the locking system applying minimal torque on the servo, two servos were put in the gripper. Two servos better overcame friction from being assembled and was the most weight-efficient method to increase reliability. Each gripper arm, and all gears, were 3D printed out of PLA with two walls and 20% infill. Despite the gripper arms being a load bearing part, they did not plastically deform or fail during weight tests.

The gripper frame is one large piece of 3D printed PLA with two walls and 20% infill. The part has many weight reducing holes to decrease weight while maintaining rigidity. Each gripper frame has a mount for the MG-90Ds and pegs to hold the intermediary spur gears of the gear train. A side plate is used on each side of the gripper frame to hold the spur gears, gripper arms, and frame together. These components form a modular sub assembly that can be easily installed or removed from the cube. During strength testing the gripper deformed significantly under load. Each side plate is secured with only two screws, while ⅛” diameter bamboo rods were added to increase rigidity at the locations in Figure 11. After adding the bamboo rods, the gripper frame experienced no deformation when put under load.

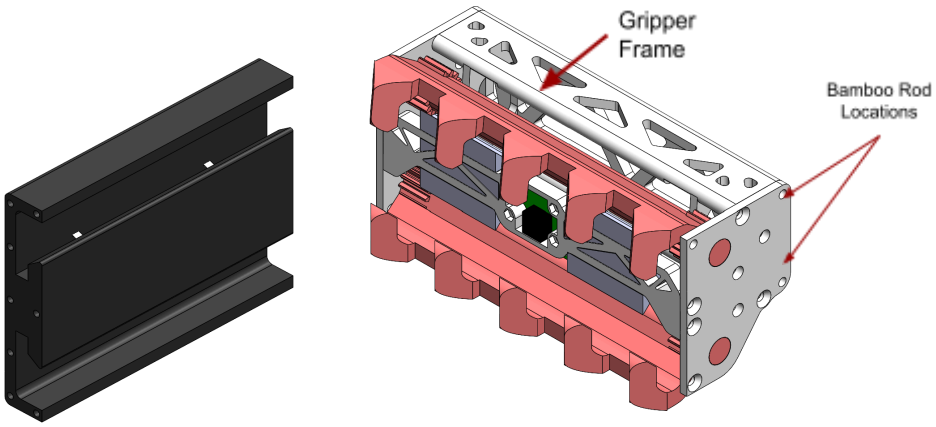

Figure 11a: Autonomous Connection Mechanism

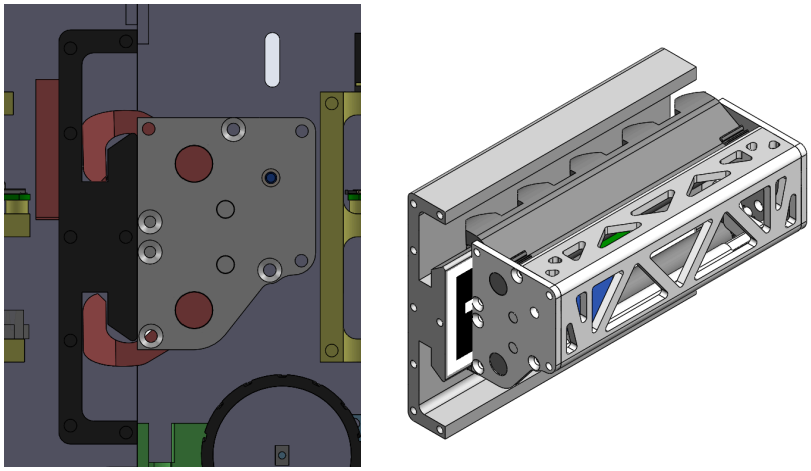

Figure 11b: Autonomous Connection Mechanism

To ensure gripping can occur with localization misalignment the gripper arms use a complex geometry to effectively lock onto other cubes. Each gripper arm has guided slots. If the robots are not aligned perfectly, the gripper teeth are designed such that they will self-align the robot into the optimal position on the receiver. The slots between teeth are the size of the side plates of the robot. The chamfers on the teeth will push the sideplates of a robot into the slots of the gripper teeth. Therefore, the slots allow two robots that are misaligned due to inaccurate localization to still lock together, as shown in Figure 12.

Figure 12: Misaligned Cubes

3.3.2 Testing

a.) Mathematical Analysis:

The mathematical analysis calculated the static forces that the locking mechanism gear train experiences. Additionally, the analysis looks at the static forces the gripper arm, receiver, and cube frame experience when holding one, two, and three cubes. The results from this analysis calculate the forces that different parts of the cube experience and are used in tandem with the digital simulation to quantify the magnitude of the forces on 3D printed and manufactured parts.

Summarizing the results of the static analysis the more cubes that one gripper assembly/cube try to hold up the larger the forces are on the gripper arms (TA, TA2, BA), cube face (FR), highlighted in Figure 13 below**,** and gripper assembly gears (fT1, fT2, fT3, fT4). With each additional cube added, the moment that the cubes generate on the locking mechanism holding them increases. The more cubes, the longer the moment arm, and in turn larger forces on the locking mechanism.

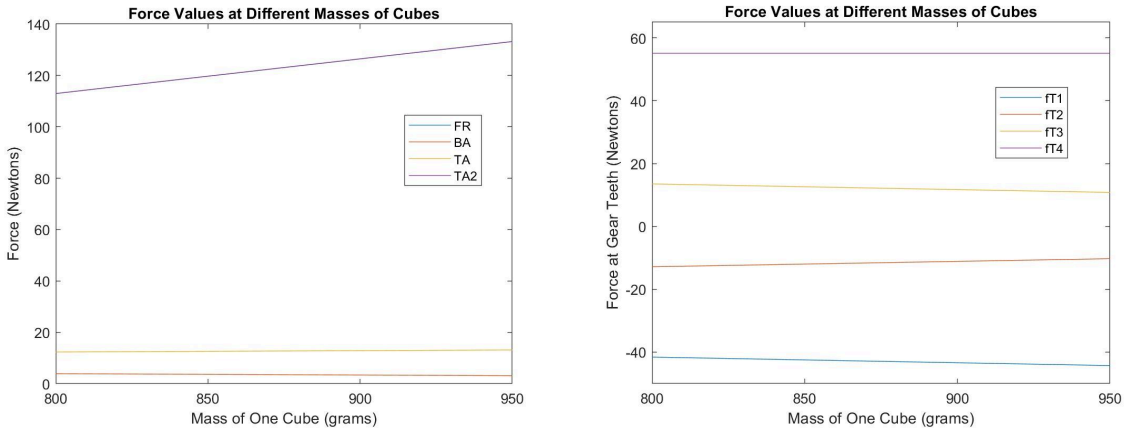

Figure 13: Force Values on Gripper/Cube (left) Force Values at Gear Train Teeth (right) with Three Cube Weight Connected. TA2 and FR overlap (left)

The forces seen in Figure 13 above highlight the maximum possible forces the gripper, cube, and gears would experience when bridging the three robot wide gap at various

cube weights. For example, at the robot's approximate weight of 850 grams, the maximum external forces on the gripper and the cube FR and TA2 are 120 newtons, BA is 3 newtons, and TA is 13 newtons. Other calculations in Appendix A highlight the maximum possible torques and forces the gripper, cube and individual components experience. For more information on static calculations, see Appendix A. The force values seen across the plots in Figure 13 and additional plots and tables from Appendix A were used in Solidworks FEA explained in the next section.

b.) Digital Simulation Stress Analysis:

Digital simulations validated the locking system through the use of Solidworks Finite Element Analysis (FEA). Using FEA informed the structural integrity of individual parts. The FEA simulates the maximum forces that parts can endure before reaching maximum yield stress, the max stress a part can undergo before the part can not return to its original state or fracture. The simulations used force values from static calculations of one, two, and three cube loads on the gripper, and additionally with the max force values it can endure before reaching its maximum yield stress. Simulating maximum forces and static calculation forces allows for comparison and how much additional force a part can endure before reaching maximum yield stress and potentially break. FEA for each part assumed it was made with PLA and had 35 MPa yield strength. PLA's actual yield strength varies from 60 MPa to 70 MPa, meaning a factor of safety of two was a safe choice for determining the maximum yield strength used for 3D printed parts. For example, stress analysis was conducted on one of the transfer gears in the gear train. Using the force values from the static analysis in Appendix A, at the locations in Figure 14 below the spur gear reached 43 MPa.

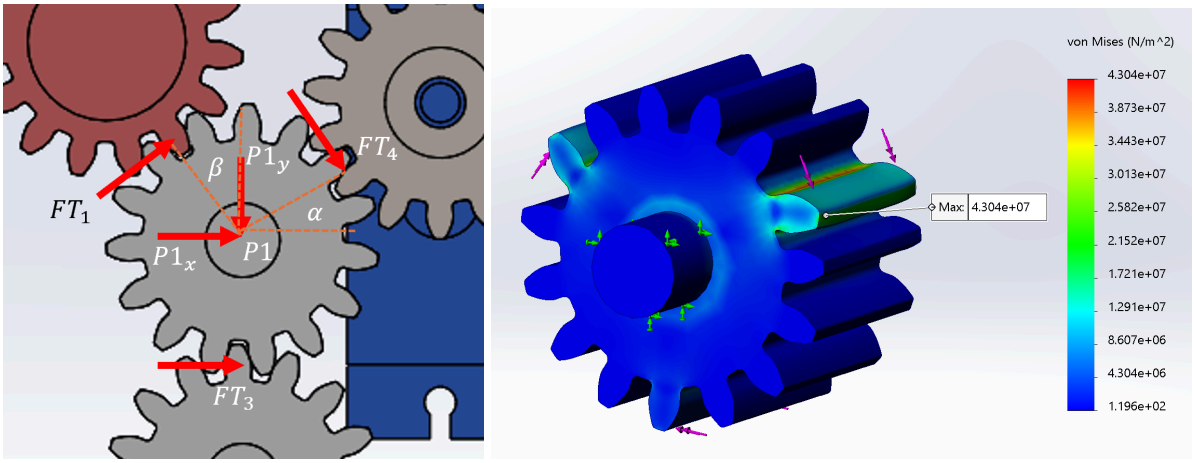

Figure 14: Spur Gear Static Calculation FBD (left) and FEA From Calculation Values (right)

However, Figure 15 highlights the spur reaching maximum yield stress (35MPa) with the largest possible force values (43.5N) applied at the three locations. Having all three forces from the static calculations applied to the gear in Solidworks FEA resulted in a max stress of 43 MPA, which is above the 35MPA max yield stress that was set. While noticing that the forces exceed the parts yield, there are many factors for this part that may make the max yield strength greater than 35 MPA. Factors affecting the yield stress results include infill density,

the forces acting on the pitch circle and not the end of the teeth, and various information on the maximum yield strength of PLA.

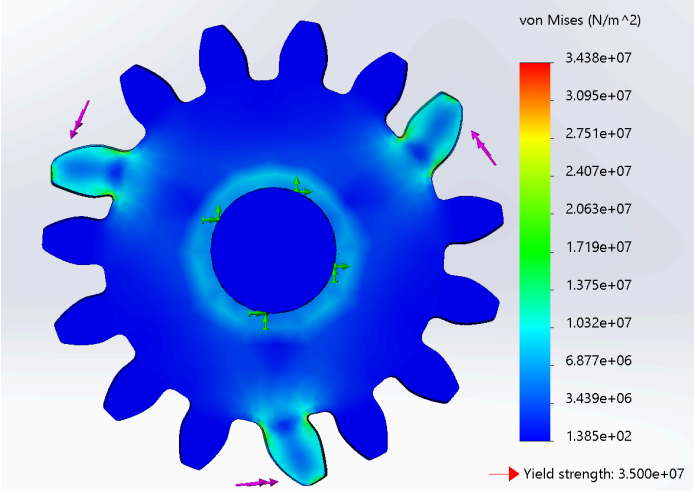

Figure 15: FEA on Spur Gear Reaching Maximum Yield with 43.5N forces Applied Circumferentially

The spur gear is one example of the FEA conducted on the locking mechanism components and more detailed information on the FEA is found in Appendix B. In summary of all the FEA, it was found that for a three cube load, three out of six locking mechanism parts analyzed exceeded the maximum yield strength of 35MPa. However, as mentioned previously, infill density, where loads were applied, and the factor applied to PLA's maximum yield strength may have caused parts to exceed the 35MPa limit.

c.) Physical Testing - Weight Test:

Physical tests conducted verified findings from static calculations and Solidworks FEA on the locking system. The first physical test was weight testing. A lever arm was mounted to the gripper, Figure 16, and weight was held at the end to better simulate the moment generated by the three overhanging cubes. However, the location where the weight was attached did not accurately simulate the center of mass locations when one, two, or three cubes were connected. Calculations to determine what equivalent weight values on the lever arm are to one, two, and three cubes can be found in Appendix C. The weight was measured using a digital scale and incrementally increased weight to simulate the weight of one, two, and three cubes held by one gripper assembly. When the weight was attached to the arm the cube on the table was held in place on a table to act as a rigid body. In addition, the deflection angle of the lever arm was recorded given that it would deflect as the load increased.

Figure 16: Physical Weight Testing Rig

The test resulted in the locking mechanism successfully holding the weight of one, two, and three cubes. The results table in Table 1 below shows details on the weight attached to the lever arm, how much deflection was recorded, and if the test passed. For a test to pass, no parts of the gripper or receiver assembly could fracture. The fourth test failed as a result of the lever arm fracturing at the base where it mounted to the gripper. The fracture is shown in Figure 17. During testing, it was noted that the lever arm when loaded twisted the gripper's frame. Further inspection led to the conclusion that the bending in the frame may have caused increased moment arm deflection angles.

Test #	Lever Arm Mass (kg)	Arm Deflection (degrees)	Number of Cube Equivalent	Test Passed	Notes
1	0.25	1	1	Yes	None
2	1	6	2	Yes	None
3	2.5	12	3	Yes	Lever arm bending gripper frame at base
4	3.5	*DNR*	~4	No	Lever arm fractured during test

Table 1: Physical Weight Testing Results Table

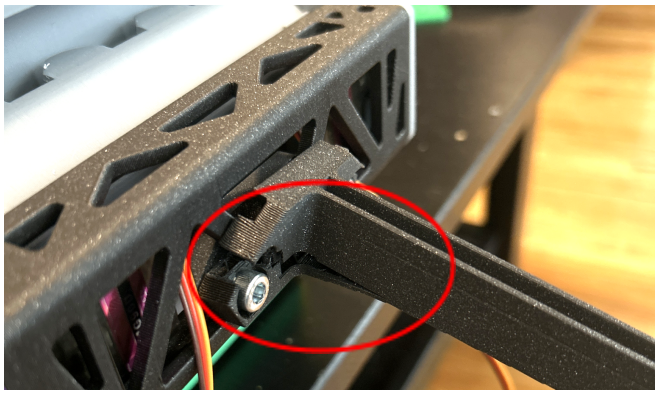

Figure 17: Physical Weight Testing Lever Arm Fracture

d.) Physical Testing - Sag Test:

The second physical test for the locking mechanism was a sag test. The purpose for this test was to measure the amount of sag the connected cubes experience when cantilevered over an edge and compare it with the Solidworks assembly prediction. Figure 18 shows an assembly of four cubes built and constrained to depict how the cubes were connected when trying to cross the gap. Measuring from the base of the first cube (far right) to the lowest point on the last cube (far left) the CAD showed that the cube had a predicted sag 0.25 inches seen in Figure 18 below. In addition, two cubes have approximately a 0.35° angle between them, and 1.05° from the first cube to the last cube in Figure 18 below when trying to bridge the three cube gap. The small angle seen is the root cause for the 0.25 inches of sag and if the angle reduced to zero, the sag amount would as well.

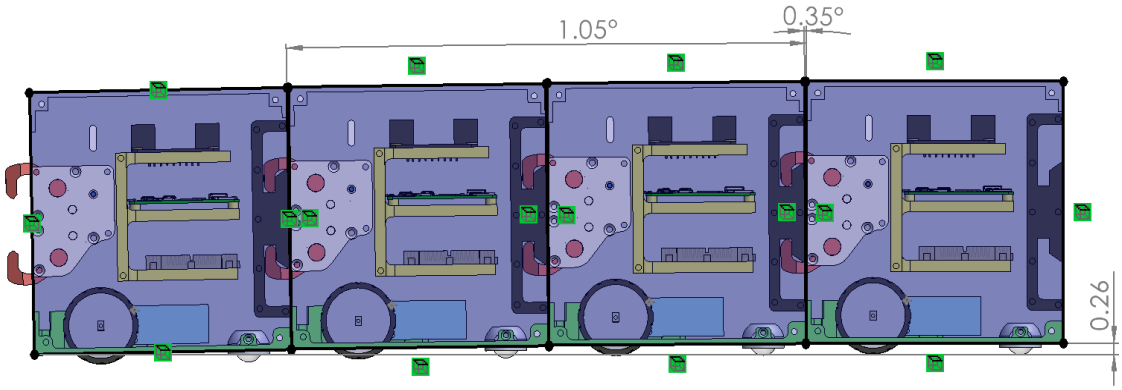

Figure 18: Solidworks Sag Test Assembly

The test conducted with physical cubes began by holding one cube down on a table and letting the other three cantilever out over the edge. A series of five tests produced measurements of the amount of sag of four robots. The tests analyzed how sag varied with respect to the alignment of the connection. Test one had the cubes connected in an ideal state where all the side plates were flush. Test two used the same configuration but force was applied by hand at the furthest cube. For tests three and four the same convention is used respectively, but instead the side plates were not perfectly flush. Test five had the cubes largely misaligned. Figure 19 highlights the three different methods the cubes were connected together.

Figure 19: Three Connection Methods Tested
From Left to Right: Side Plates Flush, Side Plates not Flush, Cubes Misaligned

By measuring the distance from the floor to the bottom of the cube on the table (Measurement 1) and to the lowest point of the most cantilevered cube (Measurement 2), the amount of sag the system experiences could be calculated. The measurement results for the three variations are below in Table 2.

Test	Measurement 1 (inches)	Measurement 2 (inches)	Total Sag (inches)
1	18.25	17.4375	0.8125
2	18.25	17	1.25
3	18.25	17.25	1
4	18.25	16.875	1.375
5	18.25	17.3125	0.9375
Average			1.075

Table 2: Sag Test Results

3.3.3 Analysis

The initial tests for the locking mechanism were positive overall given that it did not break under the desired load. However, the system experienced more sag over the ledge than expected. These results inspired a redesign of the gripper arms that decreased sag and introduced a self-aligning geometry. The new gripper design performed well during tests using the weights in Table 1. However, one test used three cubes as the weight instead of jugs of water, in which the system sagged up to an inch whereas only 0.25 inches was seen in the CAD model. The cause of sag in the final design is unknown, and either a new mechanism would need to be developed to decrease sag, or the material of each part would need to be replaced by something more rigid than plastic such as metal or composites. However, making the gripper and other parts from metal would significantly increase weight, requiring stronger

motors, causing a cascading effect that would necessitate a redesign of the majority of the cube.

3.4 Locomotion

3.4.1 Prototyping

Many SAR robots use legs, treads, or whegs (wheel legs) in order to appropriately adapt to their environments. However, this project uses a differential drive wheel design since locomotion is not the core focus of the project. All robots are equipped with two Sparkfun ROB 16413 hobby motors mounted towards the front and a caster mounted towards the back of the robots base plate. ROB 16413 motors used had low cost, accessibility, built in encoders, and acceptable stall torque. The initial drivetrain prototype used a 3D printed caster wheel with a bearing as the wheel to balance the robot. However, this design was replaced with a glass caster which was easier to manufacture and had a lower coefficient of friction on wood.

Upon printing base plates and assembling the drive train, experimentation determined that the small pegs, Figure 20 (left) that the motors mount to break under severe load from holding three robots off a ledge, Figure 20 (center). Before the pegs fracture, the wheels cant in at an angle negatively affect the robots ability to drive. The weight of the robot and load of robots overhanging imparts a moment on the motor mount, causing it to deflect toward the battery. Therefore, a support designed for the motor mount pegs around the battery ensured the robot's drivetrain was not affected. The battery cage provides support along the length of the motor mount and adds rigidity so the wheels do not bend in. Additionally, the support spans between the two motor mounts, meaning the moments push against each other, canceling each other out.

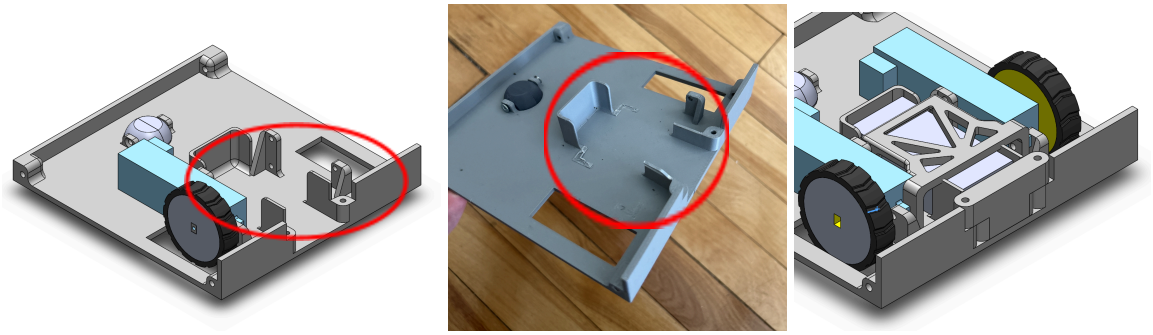

Figure 20: Motor Mount Pegs (left) Sheared Pegs (center) Motor Mount Support (right)

The initial wheel design was 3D printed out of 95A Shore hardness TPU. However, TPU was too hard to grip the surface. TPU's relatively high shore hardness led to using VytaFlex polyurethane rubber for the wheels. The polyurethane rubber wheels performed well with minimal slip given that the polyurethane has 30A shore hardness. Multiple 3D printed wheel hub prototypes were constructed to determine which hub provided enough structural integrity for the polyurethane rubber. However, during navigation testing the polyurethane broke off the hubs. The final design used 3D printed wheel hubs holding rubber O-rings.

Rubber O-rings as tires. Rubber O-rings did not break apart like cast polyurethane while still providing enough grip necessary to drive on a wood table.

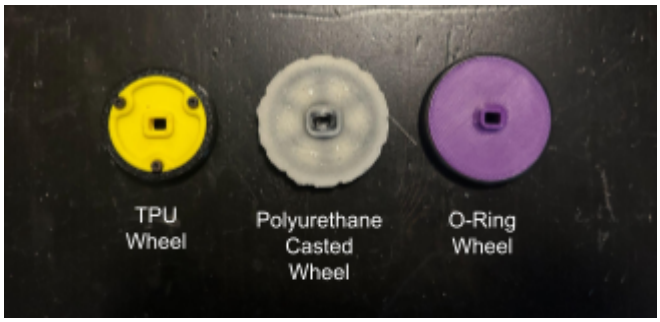

Figure 21: Prototypes of Wheels from B Term to D Term.

The wheels maintained adequate traction while experiencing minimal deformation due to the weight of the robot as predicted in the findings from the comparisons of materials through static calculations seen in Appendix A. The plot in Figure 22 graphs the friction force the wheels experience holding up the weight of one cube for different wheel material and driving surfaces are plotted. The weight of the cube varied to highlight the change in forces if individual cube weight was anywhere from 800 grams to 950 grams. The calculations show that a soft rubber wheel on dry wood, Fw1 in Figure 22**,** has the highest friction force allowing for higher torque output and grip while the robot drives around. These calculations were supported during testing with rubber O-ring wheels successfully driving the robot on a wood table without losing traction.

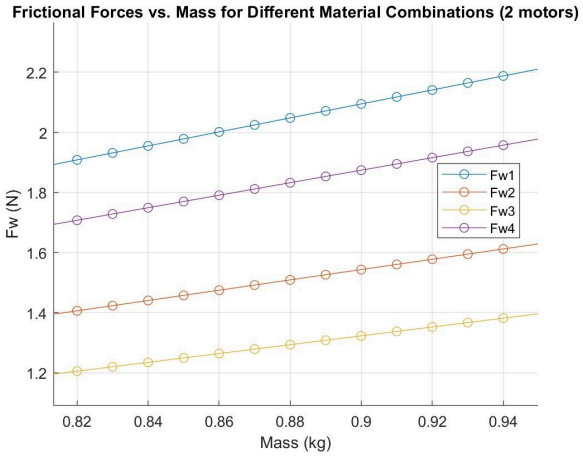

Figure 22: Friction Forces on the Wheels

With the ROB 16413 motors and the soft rubber wheels, one robot is able to drive around with a load up to the weight of another robot, which is approximately 850 grams based on calculations in Appendix A.

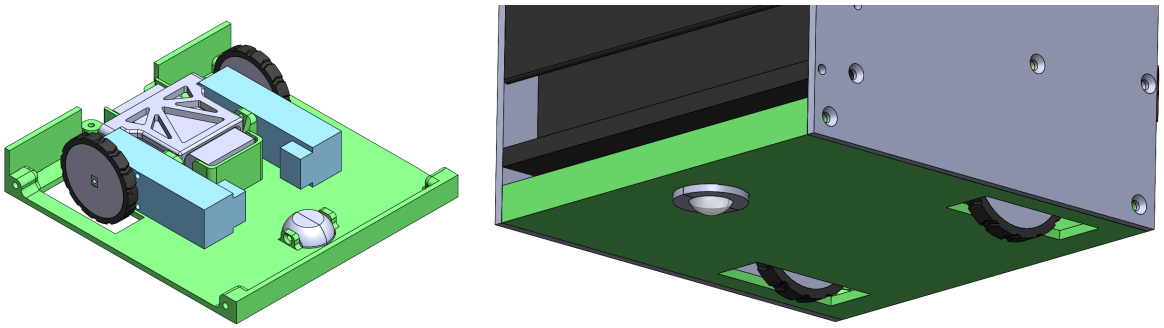

Figure 23: Final Cube Robot Drivetrain

For the swarm to localize and autonomously connect, the robots use a non-holonomic drivetrain system to traverse the demonstration field. As discussed previously in this section, research revealed that SAR swarms use a variety of drivetrain systems depending on their purpose. Locomotion is not the core focus of the project, therefore the final design uses a differential drivetrain to simplify path planning. Using a non-holonomic system also reduces the number of actuators and thus overall weight of the robot. The final design of the cube robots puts the axles more towards the front of the robot. Not having a centered axle for a two wheel robot makes spinning in place and driving kinematics harder, however, due the battery and motor cable locations, and restrictions to ensure electronics were able to be connected, centering the wheels was not an option.

As discussed previously, the drivetrain used 3D printed wheel hubs with O-ring's for tires because they provided adequate traction and experienced minimal deformation under the weight of the robots. Given that the wheels did not noticeably slip, the kinematics developed were very accurate. Additionally, the glass caster performed better than the 3D printed caster wheel with two bearings during testing.

3.4.2 Testing

The robots used a speed controller in order to control the wheel velocities given drive motor efforts. Testing the speed controller involved telling the robot to go to specific wheel velocities for a fixed amount of time and then telling the robot to stop. The measured distance that the robot traveled, the expected distance the robot traveled, and the robot's pose estimate were all compared. The accuracy of the robot in its position estimate was within ± 1 cm over distances up to 30 cm. The next test conducted used the heading of the robot. The accuracy at estimating the heading was within ± 0.1 radians in a span of 2π radians.

To test the robot's inverse kinematics, the robot was told to move to a specific pose relative to its starting location. Tests began with simple turning by giving the robot a heading of 90º, 180º, 270º, and 360º respectively. The final position was then compared to the desired position for an error measurement. These tests resulted in the same accuracy as that of the speed controller, which was about 0.1 radians or 5.7 degrees. The next set of tests conducted consisted of exclusively linear control by giving the robot a heading of 0º and only an x component of position. The robot was sent to positions of 10 cm, 20 cm, and 30 cm respectively and confirmed with a tape measure for error readings. These tests resulted in an accuracy of approximately 1 cm. An additional finding was that when there was wheel slip or an object that the robot got caught on, it properly corrected its heading to stay at 0º. Finally,

tests were conducted using both the x and y components of position as well as heading. The main test was sending the robot to 20 cm in the x direction, -20 cm in the y direction, and a heading of 36^{o}. This resulted in positional accuracy of 6 cm in the x, 4 cm in the y and heading accuracy of 5.7 degrees. One thing to note as well is that if the robot does several rotations before movement, the positional accuracy would decrease significantly due to positional drift.

3.4.3 Analysis

The locomotion tests performed were very successful and helped minimize the mechanical complexity of the system. The cast VytaFlex 30A polyurethane rubber wheels with 3D printed PLA hubs resulted in no wheel slip. The lack of slip eliminated the requirement of an Inertial Measurement Unit (IMU) to correct the robot's heading and position. Not only would an IMU decrease the amount of space for other electronics, but it would also require an additional ROS node to run and additional math for the Extended Kalman Filter. However, while the 30A shore hardness resulted in no slip, it resulted in the wheels compressing while other robots hung off the table, which contributed a significant amount to the sag in the overhang of three robots. A solution to this problem could be using Flex-Foam It 23FR polyurethane foam. With a shore hardness of 80A, it allows for a similar rigidity to TPU wheels, but with the form factor of foam, there is significantly more friction with the ground. An additional issue with the polyurethane wheels is that they wear quickly, and need to be replaced after minimal use. Another potential issue with the robot as a whole is that if the robot were to flip onto one of its sides, it would be rendered useless. To combat this, wheels on multiple sides would allow the robot to drive on its front, back, or top in ideal conditions.

3.5 Communication

3.5.1 Prototyping

A communication network allows robots to send information to each other such as their position or locking status, so that the system may better function as a swarm. Bluetooth is the primary communication protocol between the individual cube robots. The communication network is created by leveraging the Bluetooth module built into the Raspberry Pi Zero 2 W board that is equipped on each robot.

During the testing phase, various communication methods were considered including Bluetooth, ultra wideband, visible light, and IR light. Bluetooth was ultimately chosen due to its low energy consumption, range, library and integration support, and the wide use of Bluetooth chips. The bandwidth and speed of Bluetooth is more than enough for the planned system.

For the purposes of this project, a combined centralized and decentralized architecture reduced the required computing power for each robot. Each robot in the swarm calculates its position onboard and then sends that position data to a central controller which handles path planning. Offloading the computing required to plan the paths of each robot onto another

device on the network helps allocate power to the more intensive localization process with AprilTags detection. More optimized paths can also be calculated from a central controller that knows every robot's locations rather than from each robot that only knows its neighbors' locations.

The Bluetooth communication network worked as a centralized system, with each robot sending data directly to a server that disseminated messages to each of the other robots. These messages were used for a variety of purposes including localization, autonomous connection, and path planning. Robots would attempt to connect to the server three times before being considered unsuccessful, in which that robot would no longer be in use. The server was always running as long as there was at least one robot connected and would automatically shut down once all robots had disconnected.

3.5.2 Testing

To ensure that robots are able to receive messages from one another as well as the computer, the reliability of the Bluetooth communication system had to be tested. First, as shown in Figure 25, simple tests performed established a baseline for the maximum speed at which messages could be sent. These tests consisted of one robot sending singular messages to a particular target (either another robot or the computer) and verifying the accuracy of the recipient and message contents. Additionally, these tests proved that proper communication between all robots and the computer was established as no messages were dropped during the process. Standard Bluetooth chips allow for up to seven connections at once, meaning that when every cube was connected via Bluetooth, the computer's chip was at maximum capacity. Therefore, further stress tests on the communication network needed to be performed in order to ensure the stability of the network. These tests began by using four Raspberry Pi Zero 2Ws to simultaneously send messages to the server and each other at the maximum possible message rate. This test resulted in zero messages dropped at a sampling rate of greater than 100Hz, more than enough for the purposes of this project. The next test consisted of the same messaging protocol, but included six Raspberry Pis instead. As shown in Figure 25, this resulted in the same zero messages dropped at a sampling rate of greater than 100Hz. Overall, the communication network worked as intended with no major issues.

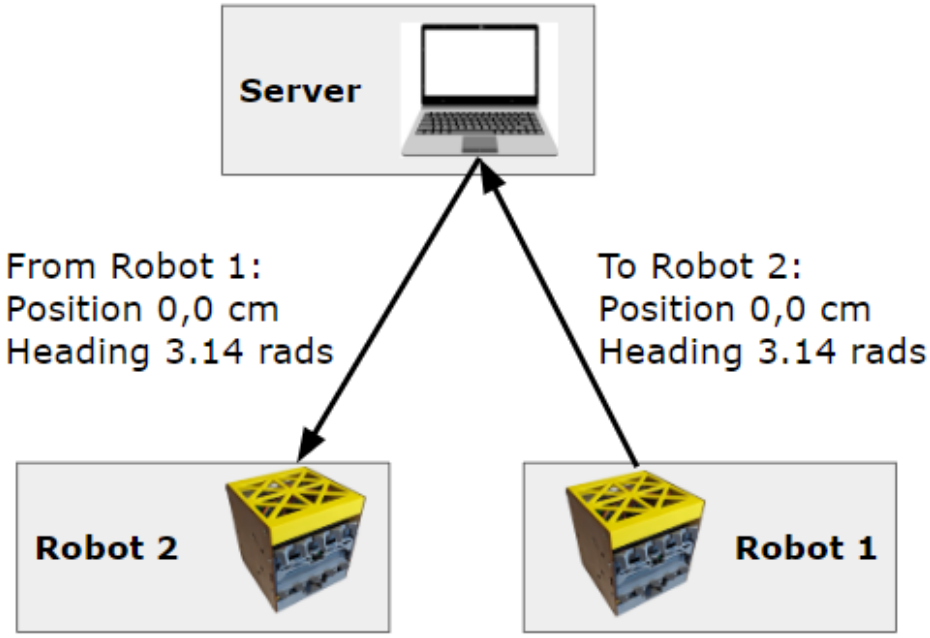

Figure 24: Simple communication diagram

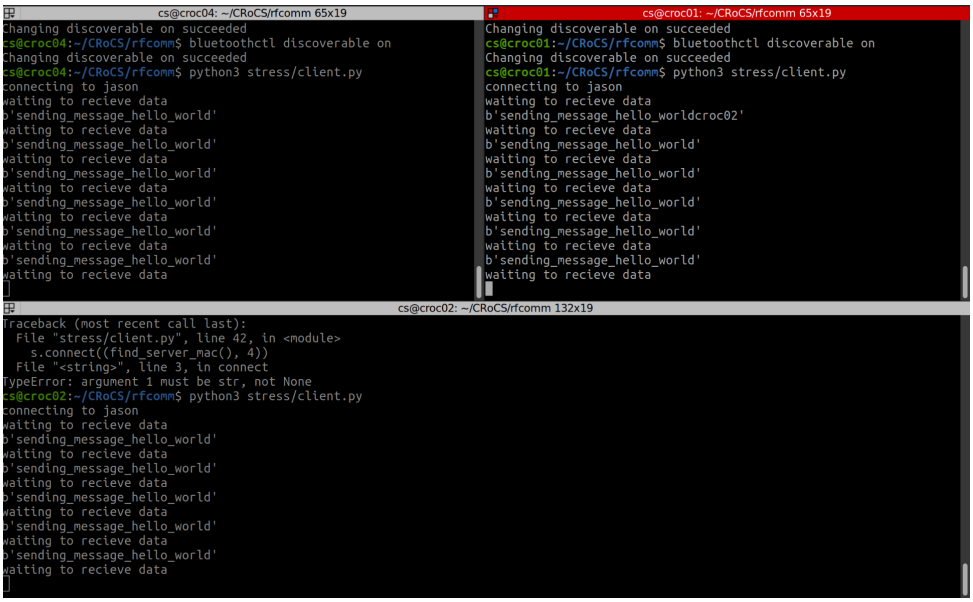

Figure 25: Communication Stress Test

3.5.3 Analysis

All communication tests resulted in positive results, with robots able to communicate with each other at a sampling rate greater than 100Hz with zero dropped messages. The communication system, therefore, is incredibly robust and the server can handle any load that the robots would be putting on it. However, further tests must be conducted with more robots in order to verify the limitation of the seven connections per Bluetooth chip. For future projects, two servers could be used to double the amount of possible connections, or one robot could have onboard communication, also effectively doubling the amount of possible connections. Overall, the tests performed proved that each robot was able to effectively send

information to one another in the swarm, and the system could be used for its intended purpose of navigation using target positions from other robots and the server.

3.6 Localization

3.6.1 Prototyping

In order for the cube robots to safely navigate, they need to localize themselves with respect to the environment. Knowing their positions and orientations in the world is crucial for collision free path planning and locking. The localization accuracy needs to be within the locking mechanism's tolerance of 0.635 cm to ensure that the cube robots can consistently lock together.

Some tested methods for localization were UWB triangulation, as well as color LED centroid detection. However, these methods performed poorly in their respective set of tests. UWB had an accuracy of 30 cm, which is too inaccurate for the system. LED centroid detection was not reliable as it is too sensitive to external lighting. Some LED colors were very difficult to distinguish from others, adding to the unreliability. Ultimately, the combination of the Arducam and AprilTags proved to be the most viable solution for the system. The AprilTag observations had an accuracy of 0.5 cm range and 0.1 radians bearing. In addition, the Arducam was able to detect different sizes of AprilTags even in different lighting environments.

Each robot uses an Extended Kalman filter (EKF) for calculating its global position and error correction. The Arducam and motor encoders have inherent noise that make them inaccurate at predicting the pose of the robot. Therefore, the system uses EKF as the sensor fusion algorithm due to its ability to handle noisy measurements from imperfect sensors.. Using EKF, both sensors are combined to improve the overall pose estimate of the robot.

The predicted location calculated from the built-in drive motor encoders allowed the velocity of each wheel to be measured. Forward kinematics equations for differential drive systems calculated the difference in heading and translation in both axes. Taking this difference and adding it to the previous pose estimate allowed the robot to predict its current pose. The EKF calculated the correction step using observations of AprilTags on other robots or on the landmarks. Appendix D goes into more detail about the process for implementing the Extended Kalman filter. The Arducam readings used for the correction step were consistent in predicting the pose of the robot.

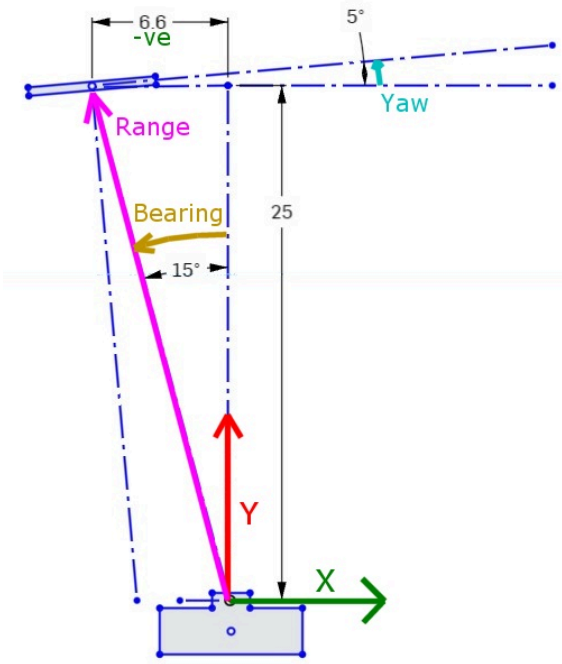

Figure 26: AprilTag reading gives the distance (range) and heading (bearing)

The overall system worked as follows: the prediction step using the drive motor-encoders was continuously predicting the pose of the robot. Once a robot got a reading from an AprilTag with the onboard Arducam, it used the communication network to get the global position of the AprilTag. The expected readings from the Arducam were calculated using the global position of the AprilTag and the current estimated global position calculated in the prediction step. Using the expected readings of the Arducam and the actual readings of the Arducam, the Extended Kalman Filter equations corrected the robots current pose estimate. This prevented the robots' global position estimates from accumulating a significant amount of error which could lead to failure to connect to one another and bridge the gap.

3.6.2 Testing

A simulation (discussed in section 4.3.1) of the system tested the localization and path-planning algorithms. The Gazebo and Rviz simulation included a simplified URDF model of the cube robot. The URDF can be seen in Figure 27 and Figure 28. The cube robot drove around in simulation using the built-in ROS diff_drive_controller package.

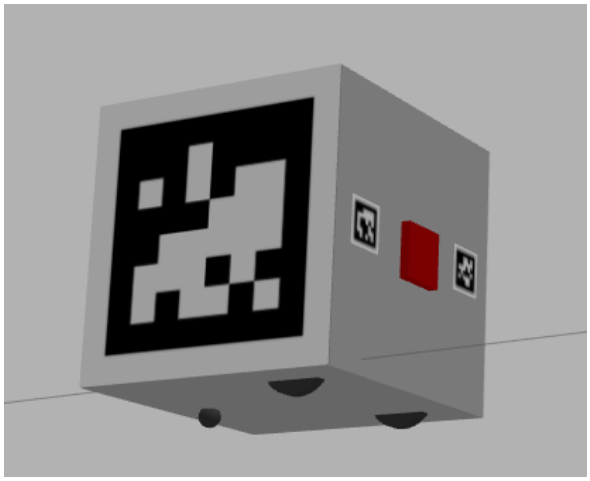

Figure 27: Simplified Cube Robot URDF

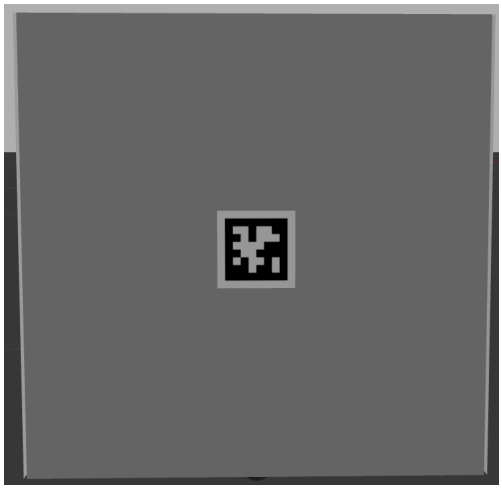

Figure 28: Back side of Robot with 1x1 inch AprilTag

To test the accuracy of the localization system, the simulated cube robot drove in a square using dead-reckoning and turning in place. At the end of each side of the square a AprilTag was set to allow the cube robot to continue running the update step of the EKF as it drives around. The Gazebo setup can be seen below in Figure 29:

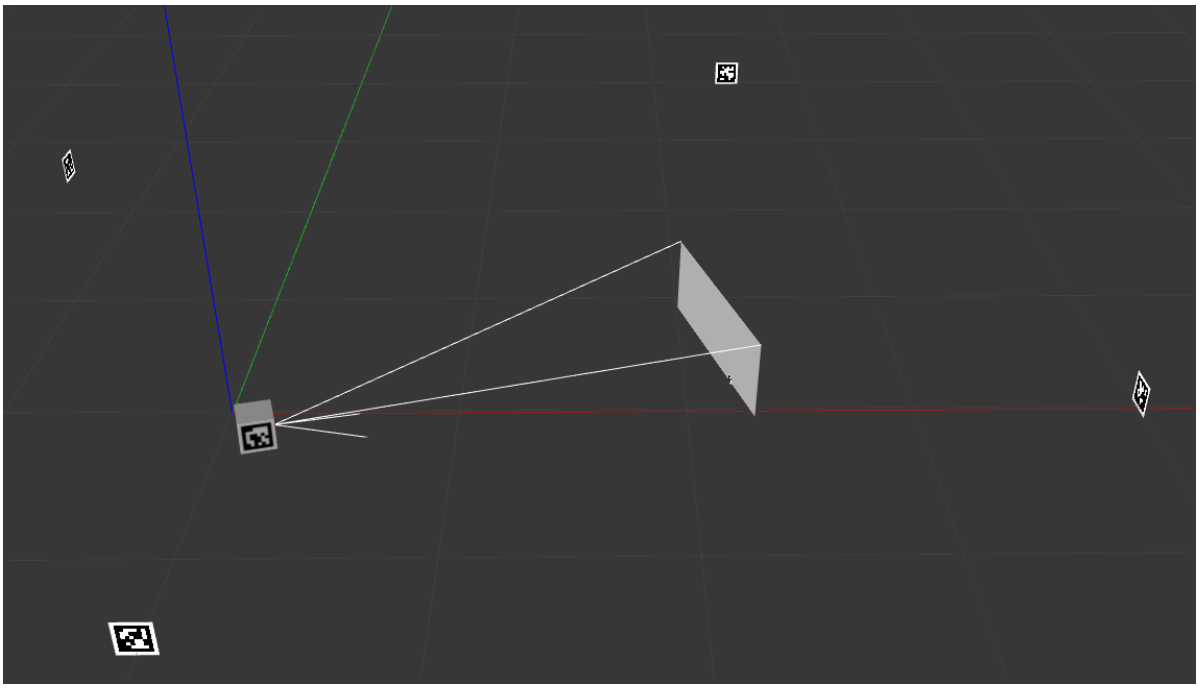

Figure 29: Localization Testing Setup in Gazebo

Rviz visualized the paths that the cube robot traveled. For testing and validation, Rviz displayed three paths: the true path, taken from Gazebo, the odometry path, taken from the differential drive package, and the calculated EKF path. The odom path is shown to visualize the location estimates using just odometry. These three paths can be seen in Figure 30. The red cube in Figure 30 is where the cube robot thinks it is from odometry. The red arrow shows where the cube robot thinks it is and its heading from the EKF. The yellow cone is the orientation covariance (how uncertain the cube robot is about its orientation).

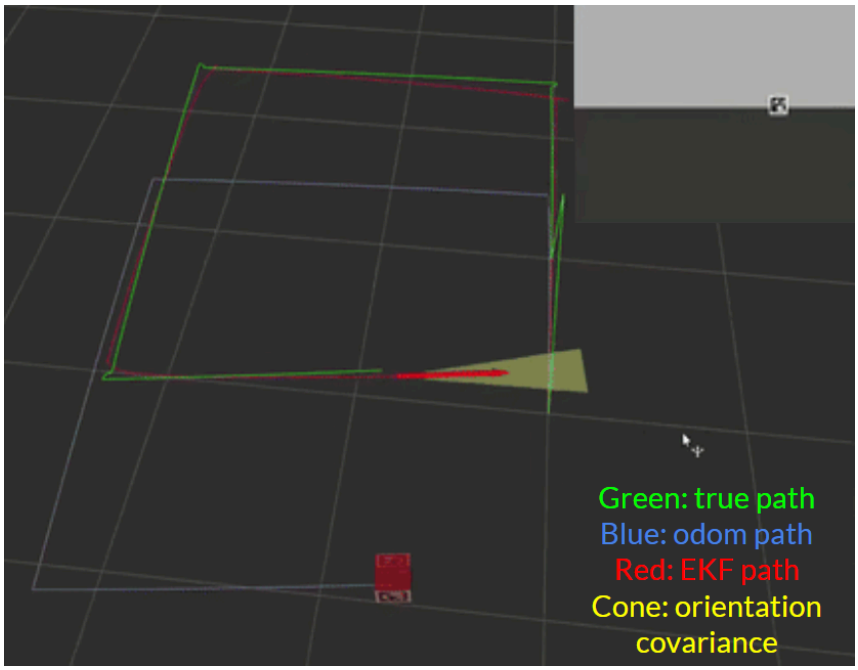

Figure 30: Rviz Simulation of Cube Robots' Traversed Paths

The accuracy result from the square test can be seen in Figure 30. To calculate error the EKF's predicted position was compared to the actual position from Gazebo at each timestep. Whenever the error exceeds a certain threshold, the EKF position is reset to the Odomoetry position, offset by its starting position to match the cube robot's actual position, to avoid error accumulation. This reset created a noticeable dip around the 40-second mark in Figure 31. After the run, the system resulted in an average of about 10 cm positional accuracy, which is about ⅔ of a cube's length.

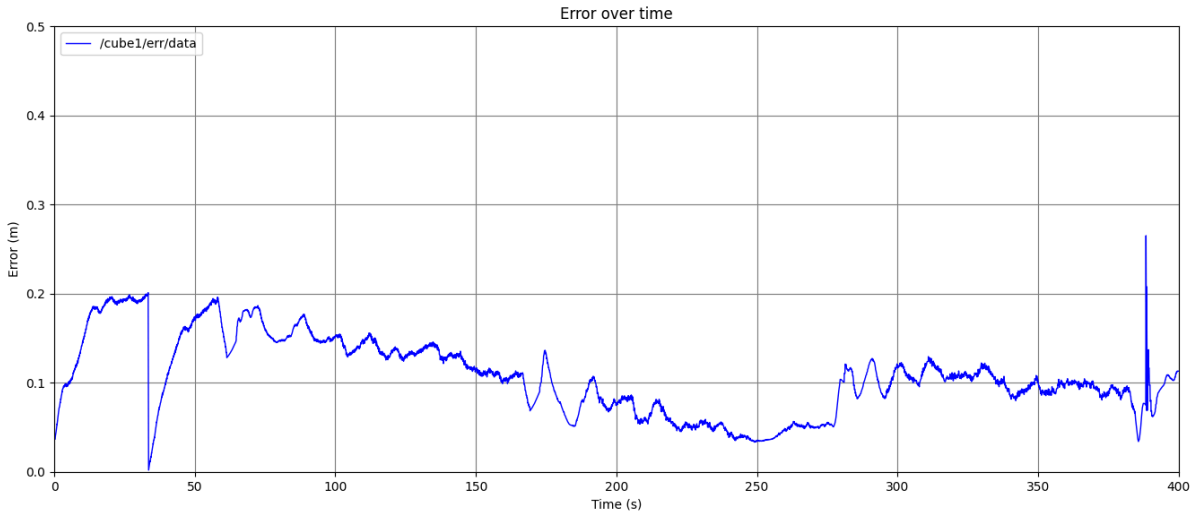

Figure 31: Position Error Over Time During Simulated Square Test

3.6.3 Analysis

The localization tests gave important insight as to the areas in which accuracy needed to be improved. The tests showed that the robots were able to successfully correct their heading when spotting an AprilTag, however, an individual robot and AprilTag had an average positional error of close to 10cm. While this is within the tolerance for simple navigation, locking requires a sub-centimeter level of accuracy in order to properly join with other robots. One way to solve this issue was to place one robot in front of the other. By using the smaller 1x1 inch AprilTags on the back of the robots (Figure 29), robots were able to correct their positional error by 80%.

3.7 Path Planning

3.7.1 Prototyping

For the swarm to lock together, it needed a solution to the multi-agent pathfinding problem. The solution to this problem, along with the main objective for a successful path-planning algorithm, focused on ensuring that the agents reached their destinations without collisions or deadlocks. Time was not an important factor for success as the project focused more on demonstrating the feasibility of the system rather than efficiency.

Initially, the swarm used a custom path-planning algorithm, modeled after a greedy-first search algorithm. The system would have cube robots plan straight paths to their desired locations. When conflicts happened between the paths, the algorithm would have one robot continue on its path while the others wait. However, due to time constraints, the swarm used an existing algorithm referencing an open-source GitHub repo rather than developing a new path-planning algorithm from scratch.

As a result, the swarm used Conflict-based search algorithm (CBS) as the multi-agent path-planning algorithm. CBS solved the multi-agent pathfinding problem by decomposing it into a set of constrained, single-agent pathfinding problems. The key idea of CBS is to grow a set of constraints and find paths that are consistent with these constraints. A constraint here is a tuple (cube, timestep, x, y) where a cube robot is prohibited from occupying position (x, y) at time step t. If a path has conflict (when 2 or more cube robots are occupying the same positions at the same time step), this conflict is resolved by adding new constraints. So overall, CBS worked on two levels. At the high level, conflicts are found and constraints are added. Meanwhile, the low-level found paths for each cube robot that are consistent with the constraints. The swarm used A* for the low-level path-finding algorithm due to its robustness and reliability. The path planning for all cube robots was calculated on the central computer to save computing power on the Pis.

In terms of design, CBS worked as an offline planning method. For context, online path planning is performed in real-time whereas offline path planning is performed before motion begins. Offline paths are calculated based on a complete prior knowledge of the environment. This worked well with CBS, as the algorithm is always aware of the robots' positions and goals as it simulated potential paths and conflicts. Offline planning ensured a global optimum path for a multi-agent system as the paths could be calculated without the need for real-time synchronization between agents.

3.7.2 Testing

The swarm implemented CBS, referencing code from a multi-agent path-planning GitHub repo (Bose, 2021). Initial tests were done using Matplotlib and then it was later moved to a more realistic setting with Gazebo (Section 4.2.1), similar to the simulation test for localization.

To test the path-planning algorithm, a simplified scenario of the cube robots lining up in preparation for locking was simulated. Each cube robot was assigned a number from one to seven as part of their name. Their goal was to line up in ascending order based on their assigned number. Each of them were initialized at random locations and are given a goal corresponding to their positions in the line.

The cube robots' name, initial positions, and goals were fed into the CBS algorithm as a yaml file. The code output a list of dictionaries of each cube robots' positions at each timestep. In Figure 32, the initial and final positions of the cube robots after following the CBS paths are shown. The cube here does not represent the actual size of the cube robots, rather it is the size of the cube robots plus an additional 4.5in configuration space for safety. The visualized cubes measured 15x15in. The path planning calculation took approximately 2-3 seconds.

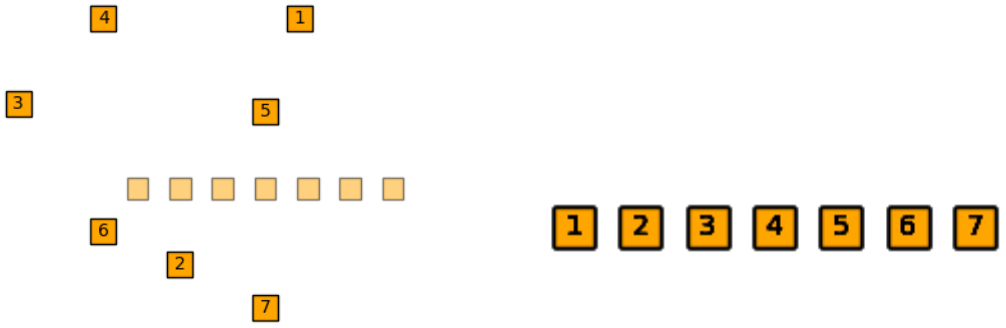

Figure 32: Before and After of CBS path-planning. Visualized with Matplotlib

3.7.3 Analysis

The path planning tests conducted have helped greatly reduce the complexity of the path planning algorithm and have yielded promising results. The CBS algorithm was able to calculate paths for robots without any collisions consistently. Furthermore, these tests achieved synchronous movement of the robots that matched the planned paths consistently. One issue with this path planning algorithm, however, is the complex calculations associated with the transformation matrices required to convert pixel coordinates to world coordinates. Currently, one pixel represents one coordinate in the path planning simulation, meaning that the distance between cubes lining up is dependent on this value. This pixel coordinate value is known as the configuration-space, or cspace, and with a larger cspace, it could result in greater positional accuracy error for the robots, but with smaller cspace, calculations will take longer as it is more likely that there will be a conflict between paths.

Chapter 4: System Testing and Verification

4.1 System Design

4.1.1 Introduction

This project used an external computer outside of the swarm as a centralized controller and database. While a decentralized controller would be more realistic for a search and rescue scenario, the controller structure is not the core focus of this project. This controller sent and received data from the robots in order to store and give desired positions for path planning.

4.1.2 Communication Architecture

Figure 33 shows the distribution of work between each robot and the central server to create a mixed centralized/decentralized system. On board, the Raspberry Pi Zero 2W handled the math for localization. This includes the odometry data, camera reading, and fusion using the Extended Kalman Filter. Additionally, the Zero 2W was running the Bluetooth communication network, which sent the estimates of its own pose as well as those

of robots within its sight to the central computer. Once the computer received these pose readings, it used the CBS path planning algorithm to generate desired poses for each of the cube robots. The computer then sent these desired poses back to the respective robots, which drove to the pose using the inverse kinematics calculations also being run on the Zero 2W. Once every robot was at its desired pose it informed the central computer that it had successfully reached the point. The computer then told the robots to start autonomous assembly, in which they drove forward and locked together. The communication network was able to properly send all of this data in individual steps, but was never tested back to back for the final demonstration.

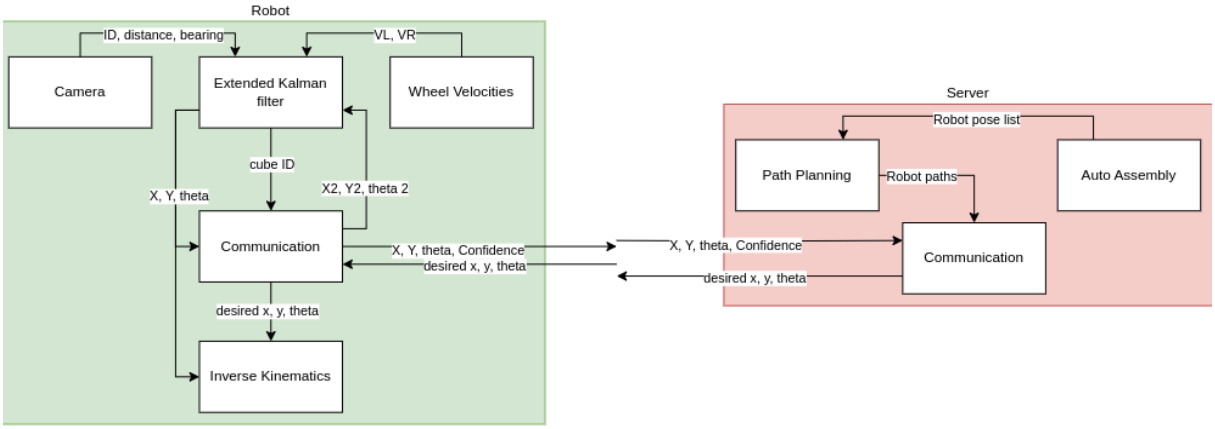

Figure 33: Diagram Showing Communication Between Computer and one Robot

4.1.3 State Machine

While the full state machine shown in Figure 34 was never fully implemented, it shows the complete set of robot states and state transitions required to complete the demonstration described in section 3.1. After initial setup and communication is established, the robots will explore the area looking for the AprilTag identifying the gap. Once the landmark is found, the robots will localize with respect to each other and the landmark to initialize their global positions. After each robot's position is established with a covariance below a desired threshold, the robots will begin planning. If the covariance of their position exceeds the threshold while the robot is following its respective path, the robot and the closest neighbor will stop to relocalize. To relocalize, the lost robot turns in place to find the neighboring robot's AprilTag and then recalculates its position. Once its covariance is within tolerance again, it will continue down the path. Once all robots reach their destination they attempt to autonomously connect, confirmed by their limit switches. If a robot does not sense a connection, it sends a message that the robot failed and attempts to reconnect. Once all of the robots have confirmed connections with each other, they cross the gap and disassemble, reaching the end of the state machine.

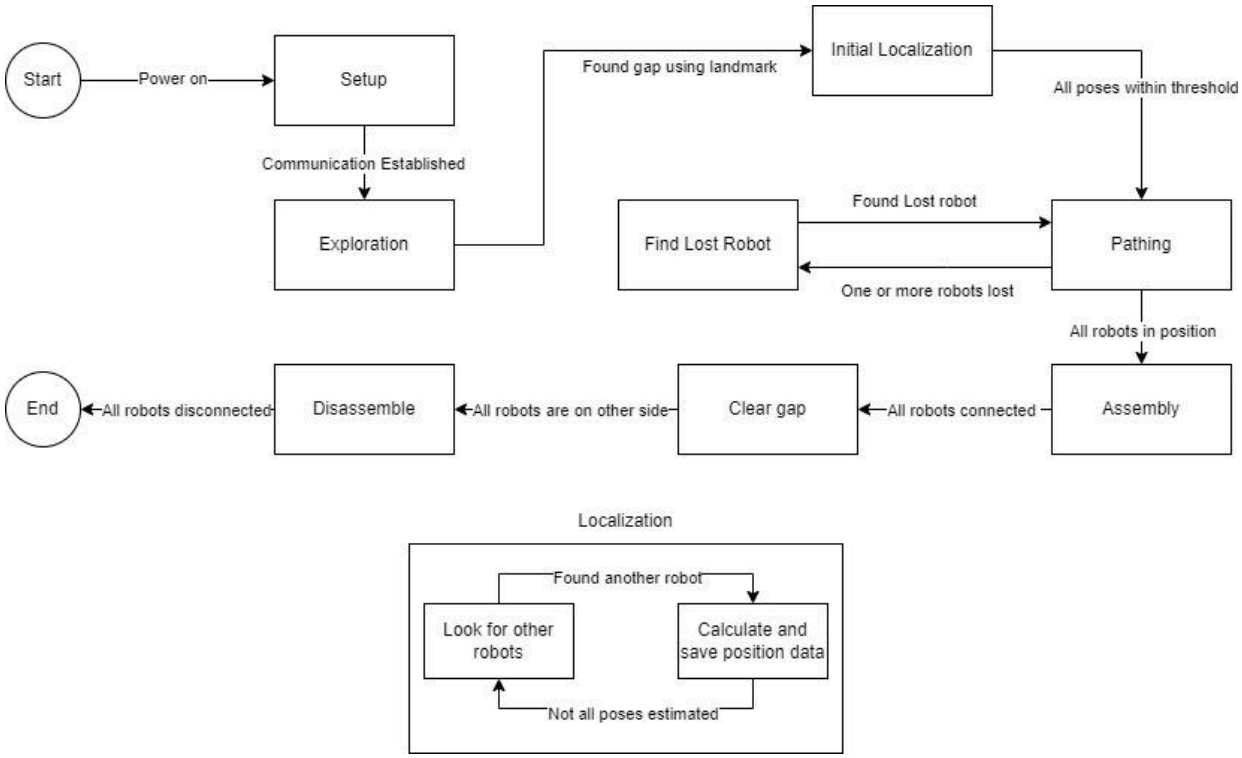

Figure 34: State Machine

4.2 System Testing

4.2.1 Simulation

While physical subsystems were in the integration process, a Gazebo simulation tested the software subsystems communication, localization, and path planning in parallel. In the aforementioned Gazebo simulation, the cube robots were initialized at random locations, navigated to line up, came together for locking, crossed a gap, and detached to explore independently. Key points of this simulation can be seen in Figure 35.

The simulated gap had two tables placed in the environment with a fixed distance equal to the length of three cube robots between them. The initial position and goals of each cube robot were fed into the CBS algorithm that generated collision-free paths for each agent in the swarm. The cube robots navigated along these paths using the gazebo differential drive plugin and their true gazebo locations. The simulated cube robots locked together using a separate plugin for attaching Gazebo models with a virtual joint.

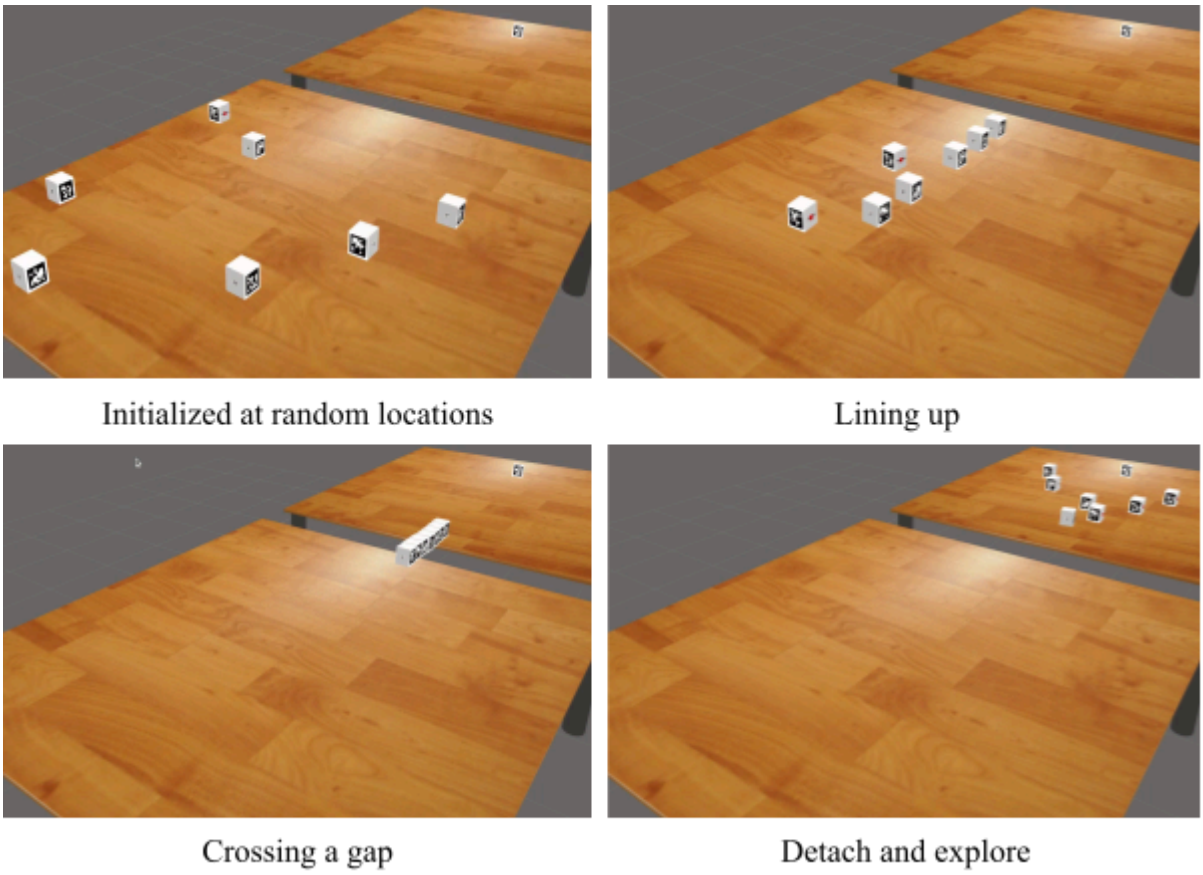

Figure 35: Gazebo Simulation Demonstration

4.2.2 Autonomous Locking

The first physical integration test performed was autonomous connection, as this is the final step required before bridging the gap. As shown in Figure 36, two robots were placed 12 inches (two robot lengths) apart and the rear robot attempted to use its gripper to attach the front robot. The rear robot used its Arducam, reading data from the front robot's AprilTag, to determine the correct pose it needed to reach before closing its gripper. For this test, the front robot remained stationary and did not have limit switches on its receiver to confirm a successful connection. Therefore, visual inspection confirmed the successful connection of the two cube robots. After 10 tests, the robots were able to successfully lock together 80% of the time. One of these successes is shown in Figure 36.

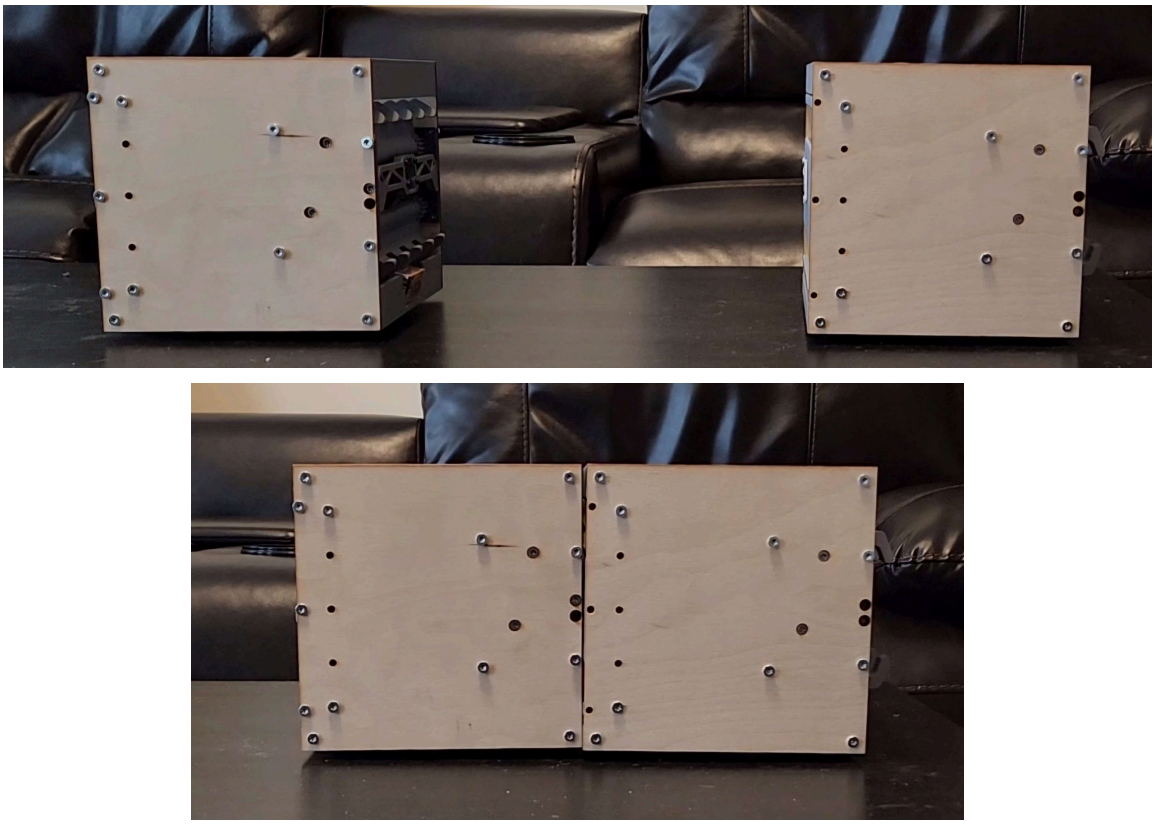

Figure 36: Two Cubes Locking

4.2.3 Bridging

The final integration test performed to confirm the functionality of the system was the bridging test. Three robots were placed close enough together to lock together without moving with their grippers open. The robots then closed their grippers to connect with one another and drove as a unit across a 7.5 inch gap, slightly larger than one robot. This test did not use the landmark for localization and robots stopped a fixed distance from the end of the chair so as to not fall off the other side. This test was ran five times with a 100% success rate at this distance. Additional tests with a gap of six inches (one robot) and eight inches respectively were performed as well. The robots successfully bridged the six inch gap all three times this test was performed. As for the eight inch gap, the front robot in the chain got stuck on the opposite side of the gap due to the low ground clearance of the cubes. Figure 37 shows intermediary steps of the cube robots bridging the gap.

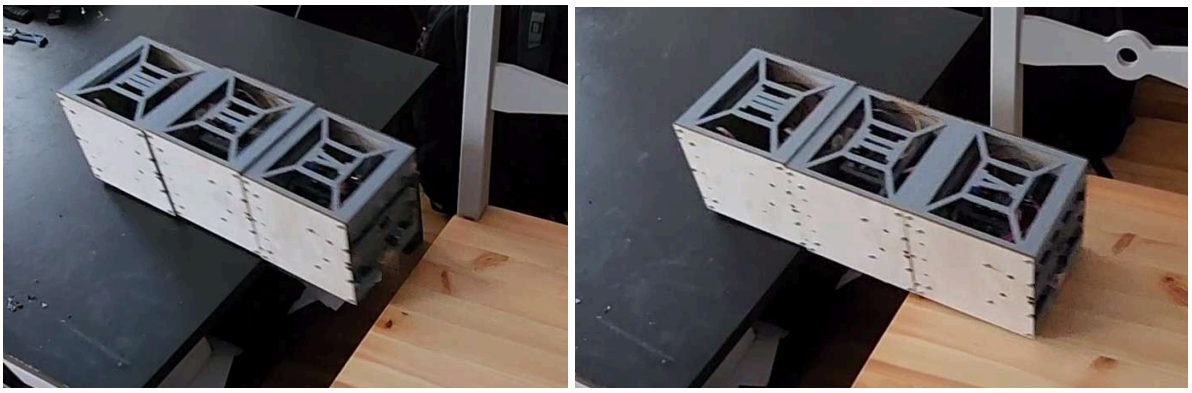

Figure 37: Three Cubes Bridging a One Robot Wide Gap

Chapter 5: Analysis and Discussion

The swarm was successful in accomplishing the majority of the objectives as stated in section 2.4. Individual agents were of minimal size and weight, as well as easily manufacturable. Agents were able to communicate their current and desired positions with one another. Agents were able to localize, or self locate, with respect to the environment and other robots. Agents planned their paths such that they can line up for autonomous connection to other agents without collisions in simulation. Finally, agents autonomously crossed a one robot wide gap by locking together to form a bridge.

Each robot was of 6x6x6 inch size, 850 gram weight, and was manufactured in about 18 hours. Robots were able to send messages to one another without any being dropped at greater than 100 Hz, well within the tolerance for the system. Robots calculated their local and global poses within 5 cm position and 0.1 radians heading, within the tolerance for the system as well. While path planning worked in simulation, the robots should be able to path plan in physical space as well. Robots drove to a specified point, however, improper collision avoidance resulted in inability for the subsystem to work as a whole. Finally, while a one robot gap demonstrated success of the overall system, a three robot gap as a proof of concept for SAR use would be a better test.

Chapter 6: Conclusions and Recommendations

Throughout the year of developing the Cube Swarm project, the team has experienced many early setbacks and has learned a lot about completing a project of the scale of to a senior capstone project. These learnings range from big picture to individual details about our specific project. First and foremost the team agrees that we all learned a lot about effectively scoping projects, and that determining a problem and effective solution could easily take more than a year to effectively implement. Having a goal and project objectives established before the year began would have been ideal and made implementing our solution throughout the year more feasible. Another key takeaway was the need to communicate clearly and effectively. Additionally, a theme present throughout the entire year was that tasks often took longer to complete than anticipated. The team ultimately ran out of time for full system integration, and had to complete this process in small steps given the amount of subsystems

the project required. This taught us the importance of proper systems engineering practices for a project of this scale, in which many tasks can be done in parallel for more effective integration.

This project proves that swarms of small robots can achieve the same lateral mobility as larger robots. This project also serves as a starting point for future development of swarm technology. Within the swarm, individual agents can act as traditional robot systems; perception, locomotion, localization, communication, and locking; as well as swarm behavior and a small form factor. The key advantage the swarm has over larger robots is redundancy.

This project should serve as a foundation for further research and development of swarm robotics. Some potential areas for improvement are: the swarm path planning algorithm, agent functionality, PCB development, and communication decentralization.

References

Arranz, A., Scarr, S., & Chowdhury, J. (2023, September 11). *How Search and Rescue Teams Pull Survivors from Rubble*. Reuters. https://www.reuters.com/graphics/EARTHQUAKE-RESCUE/mopajqojmva/

Beatty, B., & Ulasewicz, C. (2006). *Faculty perspectives on moving from Blackboard to the Moodle learning management system*. TechTrends, 50(4), 36-45.

Bello, Samuel, et al. Beach Swarm - Phase III. Major Qualifying Project, E-project-042822-111709, Worcester Polytechnic Institute, 28 Apr. 2022.

Blais, M.-A., & Akhloufi, M. A. (2023). Reinforcement learning for swarm robotics: An overview of applications, algorithms and simulators. *Cognitive Robotics*, *3*, 226–256. https://doi.org/10.1016/j.cogr.2023.07.004

Bogue, R. (2019). Disaster relief, and search and rescue robots: the way forward. *Industrial Robot: The International Journal of Robotics Research and Application*, *46*(2), 181–187. https://doi.org/10.1108/IR-11-2018-0227

Bose, A. (2021). *Multi-Agent path planning in Python*. https://github.com/atb033/multi_agent_path_planning

Brown, S., & Caste, V. (2004, May). *Integrated obstacle detection framework*. Paper presented at the IEEE Intelligent Vehicles Symposium, Detroit, MI.

Cao, Y. U., Fukunaga, A. S., & Kahng, A. (1997). Cooperative Mobile Robotics: Antecedents and Directions. Autonomous Robots, 4(1), 7–27. https://doi.org/10.1023/A:1008855018923

Cao, Z., Chen, P., Ma, Z., Li, S., Gao, X., Wu, R., Pan, L., & Shi, Y. (2019). Near-Field Communication Sensors. Sensors (Basel, Switzerland), 19(18), 3947. https://doi.org/10.3390/s19183947

Chen, L., Kuusniemi, H., Chen, Y., Liu, J., Pei, L., Ruotsalainen, L., & Chen, R. (2015). Constraint Kalman filter for indoor Bluetooth localization. https://ieeexplore.ieee.org/document/7362717

Chen, S., Yin, D., & Niu, Y. (2022). A Survey of Robot Swarms' Relative Localization Method. Sensors, 22(12), 4424. https://doi.org/10.3390/s22124424

Chitikena, H., Sanfilippo, F., & Ma, S. (2023). Robotics in Search and Rescue (SAR) Operations: An Ethical and Design Perspective Framework for Response Phase. *Applied Sciences*, *13*(3), 1800. https://doi.org/10.3390/app13031800

Delmerico, J., Mintchev, S., Giusti, A., Gromov, B., Melo, K., Horvat, T., Cadena, C., Hutter, M., Ijspeert, A., Floreano, D., Gambardella, L. M., Siegwart, R., & Scaramuzza, D. (2019). The current state and future outlook of rescue robotics. *Journal of Field Robotics*, *36*(7), 1171–1191. https://doi.org/10.1002/rob.21887

Dorigo, M., Theraulaz, G., & Trianni, V. (2021). Swarm Robotics: Past, Present, and Future [Point of View]. *Proceedings of the IEEE*, *109*(7), 1152–1165. https://doi.org/10.1109/JPROC.2021.3072740

Foster-Miller unveils TALON robot that detects chemicals, gases, radiation and heat. (2005). *Industrial Robot: An International Journal*, *32*(2). https://doi.org/10.1108/ir.2005.04932bab.003

Frey, J. P., Kendrick, R., Lowell, J. M., & Rothermel, J. M. (2006). *Energy Profiling for Off-Grid Energization Solutions in Namibia* (Undergraduate Major Qualifying Project No. E-project-050307-071942). Retrieved from Worcester Polytechnic Institute Electronic Projects Collection: http://www.wpi.edu/Pubs/E-project/Available/E-project-050307-071942

Gregg, Spencer O., et al. Beach Swarm - Phase II. Major Qualifying Project, Worcester Polytechnic Institute, 5 May 2021.

Gordon, Rachel | MIT CSAIL. (n.d.). Self-transforming robot blocks jump, spin, Flip, and identify each other. MIT News | Massachusetts Institute of Technology. https://news.mit.edu/2019/self-transforming-robot-blocks-jump-spin-flip-identify-each-other-1030

Gro, Roderich, Michael Bonani, Francesco Mondada, and Marco Dorigo. “Autonomous Self-Assembly in Swarm-Bots.” *IEEE Transactions on Robotics* 22, no. 6 (December 2006): 1115–30. doi.org/10.1109/TRO.2006.882919

Hirose, S. (1991). Three basic types of locomotion in mobile robots. *Fifth International Conference on Advanced Robotics ’Robots in Unstructured Environments*, 12–17 vol.1. https://doi.org/10.1109/ICAR.1991.240483

Hutter, M., Gehring, C., Lauber, A., Gunther, F., Bellicoso, C. D., Tsounis, V., Fankhauser, P., Diethelm, R., Bachmann, S., Bloesch, M., Kolvenbach, H., Bjelonic, M., Isler, L., & Meyer, K. (2017). ANYmal - toward legged robots for harsh environments. *Advanced Robotics*, *31*(17), 918–931. https://doi.org/10.1080/01691864.2017.1378591

Ifenthaler, D. (2011). *Multiple perspectives on problem solving and learning in the digital age*. New York: Springer.

Jahanian, O., & Karimi, G. (2006). Locomotion Systems in Robotic Application. *2006 IEEE International Conference on Robotics and Biomimetics*, 689–696. https://doi.org/10.1109/ROBIO.2006.340290

Kelly, G. (2008). *A collaborative process for evaluating new educational technologies*. Campus-Wide Information Systems, 25(2), 105-113. doi:10.1108/10650740810866594

Leinonen, A., Orjala, M., & Finland. (2008). *Feasibility study on electricity and pyrolysis oil production from wood chips in Namibia*. Helsinki: Ministry for Foreign Affairs of Finland.

Liu, J., Wang, Y., Li, B., & Ma, S. (2007). Current research, key performances and future development of search and rescue robots. *Frontiers of Mechanical Engineering in China*, *2*(4), 404–416. https://doi.org/10.1007/s11465-007-0070-2

Lu, F., & Milios, E. (1997). Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans. Journal of Intelligent and Robotic Systems, 18(3), 249–275. https://doi.org/10.1023/A:1007957421070

Marr, B. (2017). *The 4 Ds Of Robotization: Dull, Dirty, Dangerous And Dear*. Forbes. Retrieved October 2, 2023, from https://www.forbes.com/sites/bernardmarr/2017/10/16/the-4-ds-of-robotization-dull-dirty-dangerous-and-dear/

Murphy, R. R. (2004). Trial by fire [rescue robots]. *IEEE Robotics & Automation Magazine*, *11*(3), 50–61. https://doi.org/10.1109/MRA.2004.1337826

Murphy, R. R. (2014). *Disaster Robotics*. The MIT Press. https://doi.org/10.7551/mitpress/9407.001.0001

Murphy, R. R., Tadokoro, S., Nardi, D., Jacoff, A., Fiorini, P., Choset, H., & Erkmen, A. M. (2008). Search and Rescue Robotics. In B. Siciliano & O. Khatib (Eds.), *Springer Handbook of Robotics* (pp. 1151–1173). Springer. https://doi.org/10.1007/978-3-540-30301-5_51

NamWater. (Date Unknown). *NamWater Namibia Water Corporation Ltd.* Retrieved January 21, 2010, from http://www.namwater.com.na/

OV, S. S., & Jayaraj, D. (2012). Path planning in swarm robots using particle swarm optimisation on potential fields. International Journal of Computer Applications, 60(13).

Paillat, J.-L., Lucidarme, P., & Hardouin, L. (2011). Evolutionary Autonomous VGSTV Staircase Climbing. In J. A. Cetto, J. Filipe, & J.-L. Ferrier (Eds.), *Informatics in Control Automation and Robotics* (pp. 173–186). Springer. https://doi.org/10.1007/978-3-642-19730-7_12

Pathak, P. H., Feng, X., Hu, P., & Mohapatra, P. (2015). Visible light communication, networking, and sensing: A survey, potential and challenges. IEEE Communications Surveys & Tutorials, 17(4), 2047–2077. https://doi.org/10.1109/comst.2015.2476474

Patil, M., Abukhalil, T., & Sobh, T. (2013). Hardware Architecture Review of Swarm Robotics System: Self-Reconfigurability, Self-Reassembly, and Self-Replication. *International Scholarly Research Notices*, *2013*, e849606. https://doi.org/10.5402/2013/849606

Park, S., Oh, Y., & Hong, D. (2017). Disaster response and recovery from the perspective of robotics. *International Journal of Precision Engineering and Manufacturing*, *18*(10), 1475–1482. https://doi.org/10.1007/s12541-017-0175-4

Safar, P. (1986). Resuscitation Potentials in Mass Disasters. *Prehospital and Disaster Medicine*, *2*(1–4), 34–47. https://doi.org/10.1017/S1049023X00030314

Sanchez, Felix A., and Albert Jozsef Enyedy. Swarm Scaffolding MQP. Major Qualifying Project, E-project-042619-123913, Worcester Polytechnic Institute, 26 Apr. 2019.

Sathiyanarayanan, M., Azharuddin, S., Kumar, S., & Khan, G. (2014). Command Controlled Robot For Military Purpose. International Journal For Technological Research In Engineering, 1(9), 1029-1031.

Seo, S.-W., Yang, H.-C., & Sim, K.-B. (2009). Behavior Learning of Swarm Robot System using Bluetooth Network. International Journal of Fuzzy Logic and Intelligent Systems, 9(1), 10–15. https://doi.org/10.5391/IJFIS.2009.9.1.010

Stormont, D., & Kutiyanawala, A. (2007). Localization Using Triangulation in Swarms of Autonomous Rescue Robots. https://ieeexplore.ieee.org/document/4381290

Swarmrobot | Open-source micro-robotic project. (n.d.). Retrieved October 4, 2023, from http://www.swarmrobot.org/Communication.html#:~:text=Directional%20communication%20is%20extremely%20important,communicating%20neighbor%20and%20so%20on)

Takayama, L., Ju, W., & Nass, C. (2008). Beyond dirty, dangerous and dull: what everyday people think robots should do. *Proceedings of the 3rd ACM/IEEE International Conference on Human Robot Interaction*, 25–32. https://doi.org/10.1145/1349822.1349827

van Den Berg, J., Snoeyink, J., Lin, M. C., & Manocha, D. (2009, June). Centralized path planning for multiple robots: Optimal decoupling into sequential plans. In Robotics: Science and systems (Vol. 2, No. 2.5, pp. 2-3).

Venter, G., & Sobieszczanski-Sobieski, J. (2003). Particle Swarm Optimization. AIAA Journal, 41(8), 1583–1589. https://doi.org/10.2514/2.2111

Wei, H.-X., Mao, Q., Guan, Y., & Li, Y.-D. (2017). A centroidal Voronoi tessellation based intelligent control algorithm for the self-assembly path planning of swarm robots. Expert Systems with Applications, 85, 261–269. https://doi.org/10.1016/j.eswa.2017.05.048

Welsh, E. T., Wanberg, C. R., Brown, K. G., & Simmering, M. J. (2003). *E-learning: emerging uses, empirical results and future directions*. International Journal of Training and Development, 7(4), 245-258. doi:10.1046/j.1360-3736.2003.00184.x

Wood, S. (2023). *The 4 Ds Of Robotisation in Construction*. Construction Technology. Retrieved October 2, 2023, from https://www.constructiontechnology.media/news/the-4-ds-of-robotisation-in-construction/8025617.article#:~:text=Likewise%2C%20commentators%20have%20given%20much,been%20thrown%20into%20the%20mix.

Appendix A: Static Calculations

https://www.overleaf.com/read/mwbvqxqhddjh#06de55

Appendix B: Solidworks FEA

https://www.overleaf.com/read/mprnjjzdgxns#d36259

Appendix C: Physical Weight Testing Calculations

https://www.overleaf.com/read/drvbqpdyhgyv#fba341

Appendix D: Extended Kalman Filter Calculations

https://www.overleaf.com/project/65be79af92a24001aaaafd8b

Appendix E: ROB 16413 Data Sheet

https://cdn.sparkfun.com/assets/8/3/b/e/4/DS-16413-DG01D-E_Motor_with_Encoder.pdf

Appendix F: GitHub Repository

https://www.github.com/jdrockmael/CRoCS