Campus Delivery Robot

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

Date: May 2022

Authors:

Lucas Buermeyer, Robotics Engineering and Mechanical Engineering

Tian Yu Fan, Robotics Engineering and Computer Science

Nicholas Hom, Robotics Engineering

Advisors:

Nicholas Bertozzi, Robotics Engineering Senior Instructor

Michael Engling, Computer Science Assistant Teaching Professor

Bradley Miller, Robotics Resources Center Associate Director

Table of Contents

Table of Contents 2

List of Figures 5

List of Tables 7

Abstract 8

Introduction 9

Background 11

Methodology 12

Funding 12

Makerbucks 12

Tinkerbox 12

Mechanical systems 13

Prototype Chassis 13

Drivetrain Design 14

Prototyping 22

Final Wheel Construction and Testing 30

Chassis Design 35

Payload Storage Design 39

Full System 41

Belt Box Testing Methodology 42

Electronics 43

Central Processor 43

GPS 44

Lidar 45

Camera 45

Encoders 46

VESC 47

Compass 47

Software Systems 48

Operating System 48

ROS 48

OpenCV Computer Vision 48

TensorFlow Computer Vision 49

Campus Map 49

User Interface 52

Navigation 52

Internet Connectivity 53

Collision Avoidance 53

Discussion & Results 54

Mechanical Evaluation 54

FEA 54

Physical Testing 58

Electronic Evaluation 61

Central Processor 61

Mapping 61

Lidar 64

VESC 65

Software Evaluation 66

User Interface 66

Navigation 66

Computer Vision 68

Internet Connectivity 70

Collision Avoidance 71

Conclusion 71

References 72

Appendices 74

Appendix A 74

Appendix B 74

Appendix C 74

Appendix D 74

Appendix E 74

Appendix F 74

List of Figures

Figure 1 Prototype platform with Mantella chassis 13
Figure 2 GT3 belt profile speed and torque sizing chart 16
Figure 3 Robot FBD up a 10 degree incline 16
Figure 4 Wheel FBD on inclined plane 17
Figure 5 Side view of inverse belt box wheel assembly sketch 17
Figure 6 Defining the belt profile of the wheel to reference dimensions in a future sketch... 18
Figure 7 Front view of the wheel assembly, defining individual components 18
Figure 8 Cross-section view of the wheel pod assembly 19
Figure 9 Location of dowel in holes radially constraining the ring pulley 20
Figure 10 Front cross-section view of the wheel hub assembly 21
Figure 11 Wheel encoder integration 22
Figure 12 First 3D printed prototype testing the accuracy of the belt path master sketch 23
Figure 13 Second prototype using a printed pinion and bearings 24
Figure 14 New Aluminum standoffs for bearing retention 25
Figure 15 Printed prototype of half the hub assembly with a motor attached 25
Figure 16 Testing the fit of a 10" pneumatic tire 26
Figure 17 Carefree tire in the lathe before boring out the center 27
Figure 18 Hub inserted into the Carefree tire 28
Figure 19 Attempted outer diameter reduction on a Carefree tire did not turn out concentric29
Figure 20 3D printed wheel hubs 30
Figure 21 Wire EDM cut aluminum ring pulley 31
Figure 22 First version of acetal sliders 32
Figure 23 Split bearing and acetal retention system 33
Figure 24 Manufacturing process to make the inverse timing belt box hub motor 34
Figure 25 Labeled assembly view of hub motor system 35
Figure 26 Fully assembled hub motors 35
Figure 27 Box tubing frame prototype CAD for the robot 37
Figure 28 Updated chassis base 38
Figure 29 Robot Chassis with top and bottom plates 39

Figure 30 Amazon polyethylene box used to store payloads 40
Figure 31 Aluminum structure used to retain the box 40
Figure 32 Full system CAD 41
Figure 33 Chassis and payload storage structure components..... 41
Figure 34 Full assembly of final robot..... 42
Figure 35 Startup torque testing of wheel assembly..... 43
Figure 36 Raw GPS coordinate data collection 50
Figure 37 Graph for campus map after data filtering 51
Figure 38 NEO-M9N coordinate data vs. C94-M8P coordinate data..... 51
Figure 39 User interface for choosing goal destination..... 52
Figure 40 Safety factor on frame with 500lb load evenly distributed 54
Figure 41 Safety factor on frame with 500lb load distributed over center of frame 55
Figure 42 Safety factor on frame with 500lb load distributed over center of panel 55
Figure 43 Safety factor on wheel assembly with 100lb load distributed over wheel side..... 56
Figure 44 Exaggerated radial safety factor on wheel assembly..... 56
Figure 45 Exaggerated tangential safety factor on wheel assembly 57
Figure 46 Raspberry Pi with heatsinks 61
Figure 47 C94-M8P coordinate datastream with GNSS fix 62
Figure 48 C94-M8P RTCM messages for RTK data..... 62
Figure 49 GPS coordinate reading while stationary in outdoor setting..... 63
Figure 50 Data summary for latitude and longitude readings from GPS 63
Figure 51 GPS data overlaid on campus map..... 64
Figure 52 Lidar scan data visualized in Rviz..... 65
Figure 53 Sample campus map for A* implementation 67
Figure 54 Shortest path between two points, returned by A*..... 68
Figure 55 Comparison between input image, true mask, and predicted mask 69
Figure 56 Training and Validation Loss over Time..... 69
Figure 57 Computer vision road detection..... 70

List of Tables

Table 1 Pros and cons list of different structure systems... 36
Table 2 Free spin speed measurements... 59
Table 3 Kinetic torque testing... 59
Table 4 Static torque testing 59

Abstract

The demand for robotic deliveries has increased exponentially under the mandates of social distancing due to the safety of contactless deliveries. This project develops an autonomous delivery robot for the WPI campus, capable of transporting packages, food, equipment, and other items from one location to another. To complete deliveries, the robot utilizes a novel inverse belt box transmission system and navigates using pathfinding, path following, and collision avoidance.

Introduction

Autonomous service robots perform repeatable tasks in various application domains to increase productivity, service quality, and convenience [1]. For example, cleaning robots sweep public places, delivery robots transport packages, fire-fighting robots combat fires, and surgery robots assist in medical procedures. Beyond their designated features, these robots must also engage in human interactions. Designing for successful human interactions encourages the integration of sensors, computer vision, speech recognition, machine learning, and artificial intelligence, making the robots more intelligent, mobile, and interactive in dynamic environments.

The market for autonomous service robots has been predicted to grow from USD 37 billion in 2020 to USD 102.5 billion in 2025, at a compound annual growth rate of 22.6% [2]. Furthermore, 52% of online users believe that robots will perform most human activities and 94% of those who have integrated robots into their businesses stated that they increase productivity [3]. Customers have also reported higher satisfaction with robot-staffed locations, such as hospitals and hotels because they decrease the exposure rate of the COVID-19 virus [4].

Delivery service robots are pedestrian-sized vehicles that autonomously deliver packages [5]. Over the past two decades, the annual sales rate of e-commerce has increased at an average of 16% and expected delivery times are decreasing. As a result, companies are investing more effort into the development of autonomous delivery solutions [6]. Furthermore, many individuals have advocated for cities to provide large-scale operation of autonomous delivery robots [7].

The objective of this project is to design and manufacture an autonomous delivery robot to operate on the WPI campus. This robot should transport packages, weighing up to 100lb and of size less than 24” x 24” x 36”, from one location on campus to another while avoiding obstacles, including objects and students. The integration of this robot on the WPI campus would improve the work efficiency and safety on campus. It would allow mail services to deliver packages directly to student dorms, eliminating the need for an employee to perform this tedious task. Another use would be to assist lab staff members in transporting heavy equipment from one building to another, alleviating the physical strain that they would have experienced. This robot could also transport food and other necessities to students quarantined due to COVID-19 exposure, minimizing the need for infected students to exit their quarantine space and reducing potential future exposures.

Background

Within the past several decades, the number of service robots in use has grown exponentially. Service robots improve their customer's quality of life by performing repetitive tasks, such as vacuuming (as done by the iRobot Roomba) and managing smart home systems (as done by Alexa from Amazon). A prominent subcategory of service robots is autonomous delivery robots that transport items from one object to another. These robots dramatically increase the efficiency of working environments and handle tasks that are tedious or unsafe.

DEKA Research and Development has been working on a delivery robot, named Roxo, which specializes in package delivery for FedEx. The system uses computer vision processing to safely operate on sidewalks, bike lanes, and roadsides, located within three to five miles of the retailer's location. While the robot is autonomous, a team of remote operators can override autonomous motion in the event of an emergency and communicate with surrounding people through a speaker and microphone. Furthermore, it uses turn signals, lights, and screens to alert nearby travelers of its navigation intentions [8].

Starship is another company that focuses on the development of autonomous delivery robots. They specialize in rapid food delivery within a 4-mile radius. Starship robots have recently been integrated on college campuses to minimize the risk of COVID-19 exposure. The robot's services can be requested through a mobile app by students, where its navigation journey can be tracked. Starship robots use multiple cameras, ultrasonic sensors, radars, and neural networks to detect nearby obstacles and ensure the safety of pedestrians [9].

Amazon is also implementing delivery robots into its business model to meet customer demands and reduce human-to-human contact throughout the COVID-19 pandemic. The Scout robot drives along sidewalks to deliver packages in nearby neighborhoods. The robot relies on cameras and ultrasonic sensors for navigation with the support of machine learning algorithms. The mechanical design is watertight to ensure that it can withstand harsh weather conditions. Another Amazon autonomous delivery solution is the incoming Prime Air service, which will utilize autonomous aerial vehicles (drones) to deliver packages to customers within 30 minutes [10].

Methodology

Funding

A significant challenge in this project was to obtain funding and stay within the financial limitations. Each participating student received $250 from the WPI Robotics Department, totaling to $750 for the project. As the project progressed and global shortages increased material costs, it became evident that more funding would be required to complete the robot. The first source of additional funding was the WPI Makerbucks program. After this funding was secured and more was required, the team applied to and received funding from the Tinkerbox program.

Makerbucks

Makerbucks is a program from the WPI Innovation Studio Makerspace that is intended to fund student teams to assist prototype developments. The grant is open to all WPI students and funding must be used exclusively for tools and materials to develop their prototype, with a preference for innovative and multi-disciplinary teams. After applying for this grant, the team was provided an additional $200 for further project development [11].

Tinkerbox

The Tinkerbox program fosters innovation and entrepreneurship within the WPI community by providing seed funding for student-initiated ideas, including MQPs. Select student teams are provided with access to mentors and entrepreneurial workshops. Funding amounts typically range from $500 to $3000, depending on the scope and innovation involved in the project [12]. The team applied for Tinkerbox funding in early Spring for the development of the inverse belt box transmission system. The Tinkerbox foundation generously provided $2,250 of funding that was donated by the Women’s Impact Network (WIN).

Mechanical systems

Prototype Chassis

To develop the system software while the robot chassis was being designed and manufactured, a temporary test platform was developed. The test platform consisted of the chassis of a former combat robot, Mantella, and a box holding the relevant electronics, as shown in *Figure 1* [13]. This chassis was chosen due to its high maneuverability (due to the four-wheel-drive system) and ease of electronics mounting (due to the flat robot top). Initial testing was done with this platform using BLHeli_32 brushless ESCs, which were later upgraded to VESC 6.7 brushless ESCs. These ESCs, in addition to the electronics, were secured to a cardboard box atop the chassis. Although much smaller than the final robot design, this test platform allowed for the effective testing of the autonomous driving code and sensor integration.

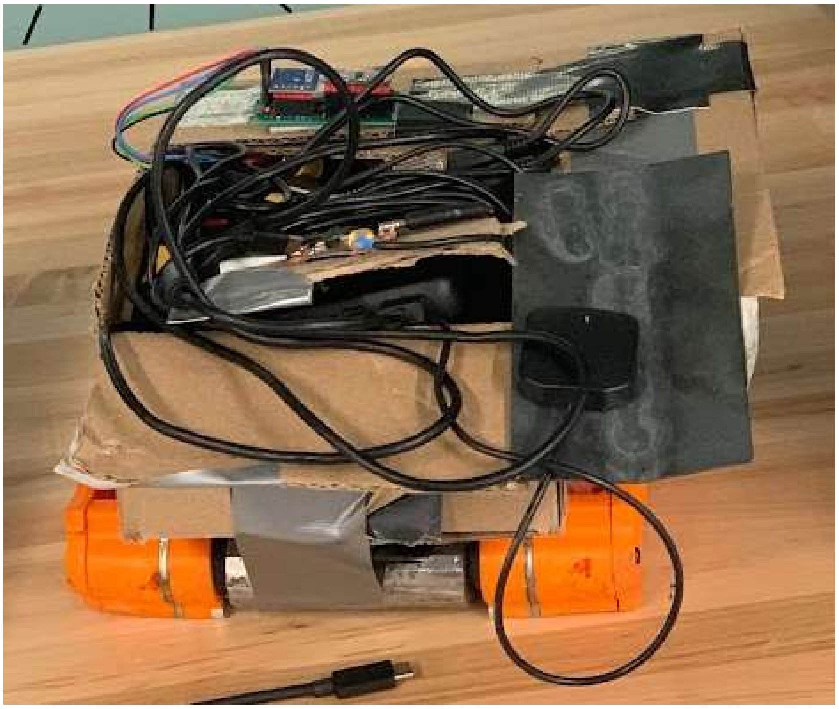


Figure 1 Prototype platform with Mantella chassis

One feature lacking from the test chassis was the motor encoders. While encoders were intended to be used on the final platform, they were not integrated into the Mantella design. The team decided to forego encoders on this prototype chassis due to the difficulty associated with mounting encoders and integrating them into the existing software architecture. Rather than spend time on this temporary solution that would have to be redeveloped for the final chassis, time was redirected toward GPS integration and computer vision development.

Drivetrain Design

The drivetrain design was driven by the following objectives: reliability, low sound volume, small form factor, and weather resistance.

In the development phase, robustness and reliability are important to prevent delays in software integration and testing. Full autonomous deployment of a delivery robot also necessitates durability to minimize maintenance costs and product down-times, as well as robustness against potential physical challenges resulting from interactions with college students and collisions with outdoor obstacles.

This robot is intended to provide a convenient service to the students and staff of WPI without disrupting the surrounding people. A potential source of campus disturbance from the robot would be loud noises. The primary noise concern of this robot is the drivetrain system, so a low noise level was crucial to the robot's deployment.

A compact drivetrain decreases the overall footprint of the robot. Because the robot was intended to drive on sidewalks, the width of the robot must be less than that of a sidewalk. A smaller wheel volume results in a larger carrying capacity for a given robot size, thus making more efficient use of space. The designed drivetrain system includes integrated sensors, speed controllers, and motors, contained within a single wheel assembly. This significantly reduces the volume within the robot reserved for the drivetrain components and allows for the overall footprint of the robot to be reduced.

The delivery robot was intended to complete deliveries in all normal weather conditions; the robot must operate in rain or snow. Due to the potential for the electronics to electrically short if wet, the wheels must be waterproof to prevent water from entering the robot and destroying electronics or rusting steel components.

To create a compact system, a hub motor-type assembly was ideal due to its disregard for any external reduction methods. Several types of reduction methods were considered including planetary, cycloidal, strain, and an inverse belt box. The team decided on the inverse belt box reduction because it was the most innovative design, allowing for additional research into a unique and interesting transmission system.

The inverse belt reduction utilizes a small pinion, ring pulley, and belt. The belt is driven by a pinion located in the center of the ring pulley and the belt drives the outer ring pulley. The belt is pressed against the outer ring pulley using multiple bearings and plastic sliders to ensure contact between the belt and ring to transmit torque. The ratio of the system is determined by the ratio of the teeth or pitch diameters of the center pinion and ring pulley. In this implementation, the torque is transmitted from the ring pulley through dowel pins to the wheel hub, which constrains the wheel around the ring pulley.

The wheel assembly design was developed in a series of parametric Solidworks sketches. These sketches utilized Solidworks parameters and equations to enable easy modifications to the design. The sketches were later used as a foundation to later create 3D geometry. Each wheel transmission system was designed to be fully contained within a tire, including the inverse belt reduction, motor, encoder, and ESC.

Of the wide variety of different belt profiles available on the market, the GT2mm belt profile was chosen for its circular profile characteristics. The circular profile found in GT2 belts allows for easy entry and exit meshing on gear teeth, a quality that is greatly desired in a design that runs opposing direction teeth. The small 2mm pitch enabled a greater final reduction by reducing the size of the ring pulley and wheel.

In order to properly size the belt for use in the wheel assemblies, the first calculation was the expected tension in the belt. Due to size and weight constraints, the most desirable belt was the smallest belt that would maintain sufficient strength. The model in *figure 2* was used to determine the proper belt size. At the peak motor speed of 10,000 rpm, the 2 mm profile is sufficient for up to a peak continuous motor load of 0.4 Nm. Given that the peak expected torque when carrying a full payload of 200 lb up the steepest incline on campus (10° from horizontal) is 0.377 Nm of motor torque, the belt should be sufficient under the full range of max speed conditions and reasonable torque conditions. The peak possible torque of the motor at full speed, however, does exceed the limits of the 2mm belt profile, but the motor should not stall continuously, and current limitations can be placed on the motor to prevent excessive force through the belt.

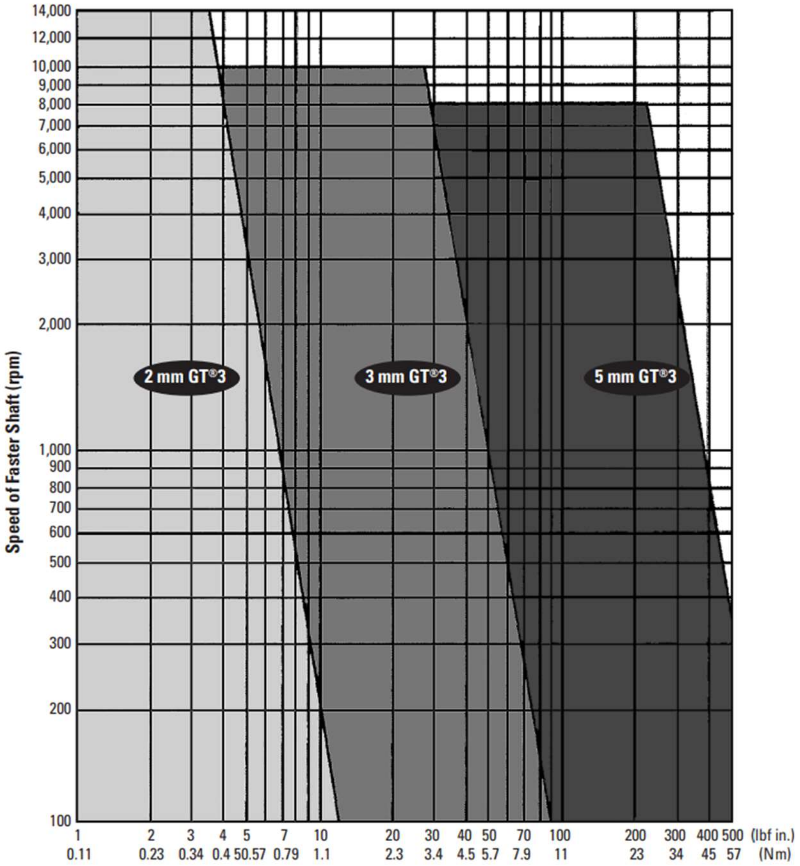
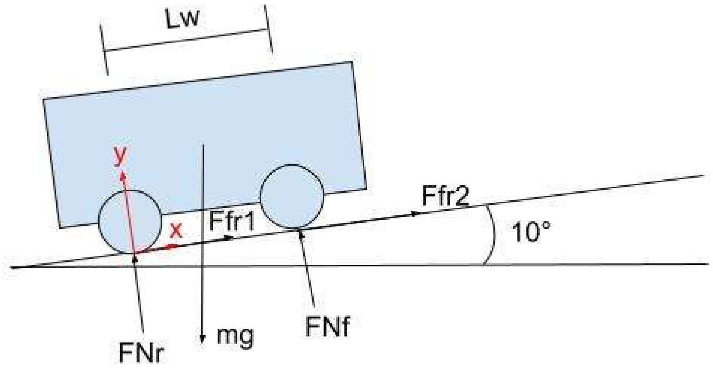


Figure 2 GT3 belt profile speed and torque sizing chart



Fx = Ffr1 + Ffr2 - mgsin(10)
Fy = FNr + FNf - mgcos(10)

Figure 3 Robot FBD up a 10 degree incline

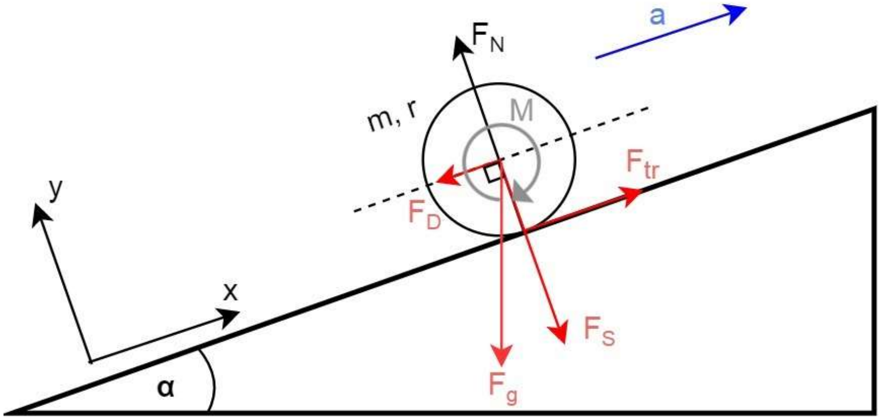


Figure 4 Wheel FBD on inclined plane

$$F_g = m * g$$

$$F_{tr} = F_g * \sin(\alpha)$$

$$M = F_{tr} * r = F_g * r * \sin(\alpha) = m * g * r * \sin\,(\alpha)$$

$$m = 20\;kg, g = 9.81\frac{m}{s^2}, r = \;2.75in = 0.07m, \alpha = 10°$$

$$M = \;2.4\;Nm$$

Equation 1 Free body diagram math

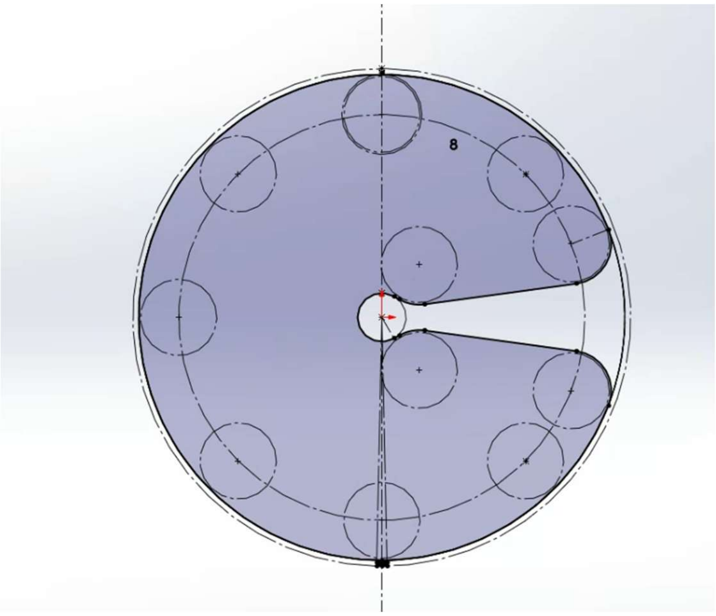


Figure 5 Side view of inverse belt box wheel assembly sketch

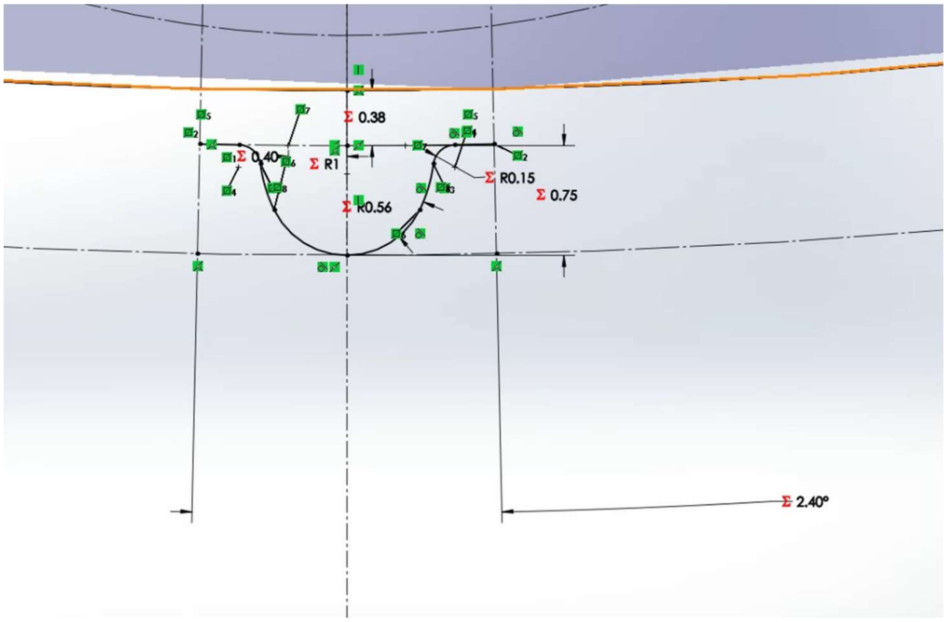


Figure 6 Defining the belt profile of the wheel to reference dimensions in a future sketch

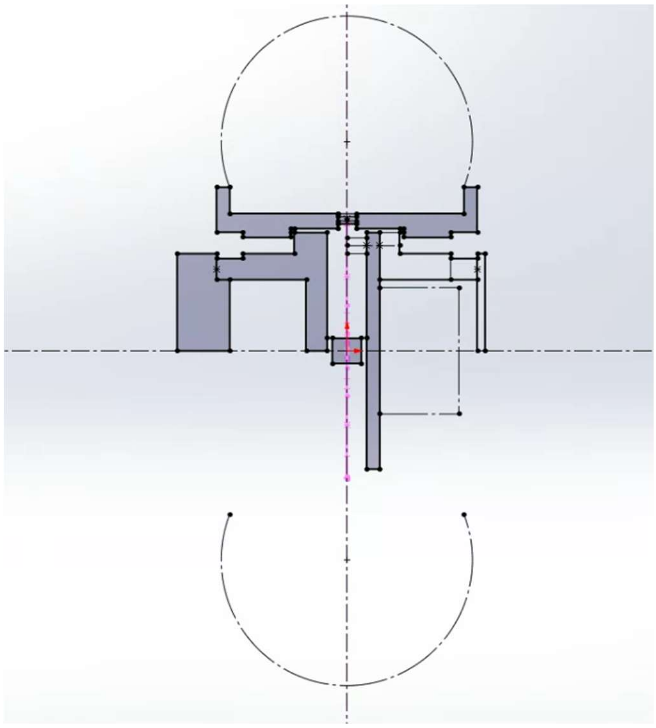


Figure 7 Front view of the wheel assembly, defining individual components

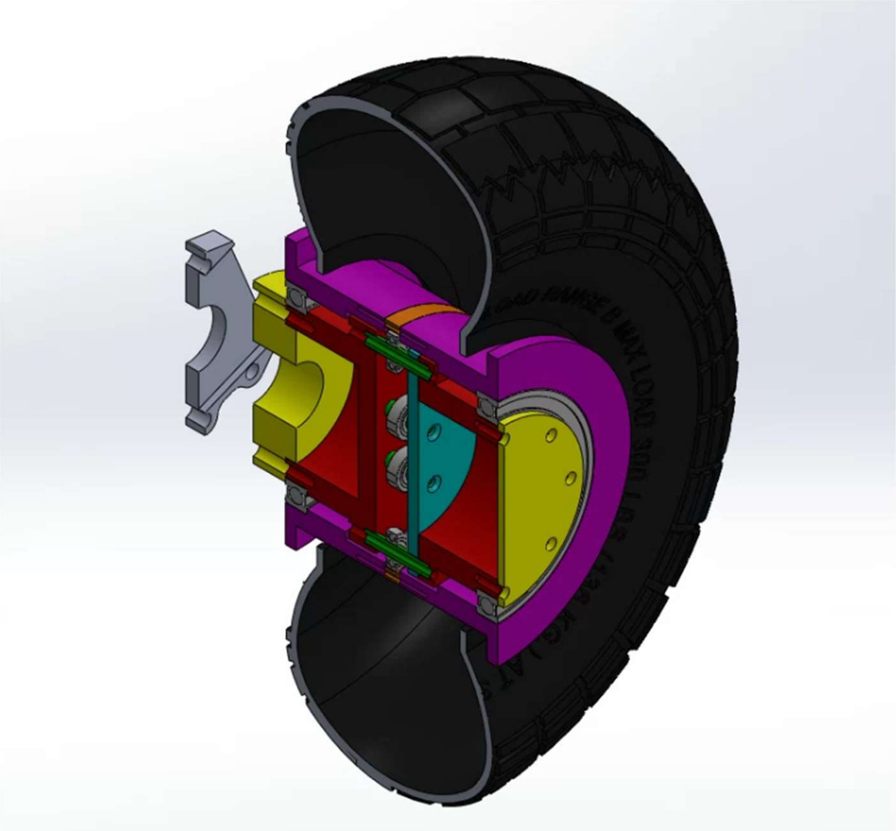


Figure 8 Cross-section view of the wheel pod assembly

After generating a master sketch (containing the key 2D design features and dimensions) for the drive pod, the design could be realized in 3D (*Figure 7*). The construction and assembly of the wheel pod utilize standoffs to compress two halves of the wheel hub together, clamping onto the tire, as well as retaining the bearings to maintain contact between the belt and ring pulley. Dowel pins are pressed through the ring pulley(shown in orange in *Figure 8*) to transmit torque and ensure assembly tolerances.

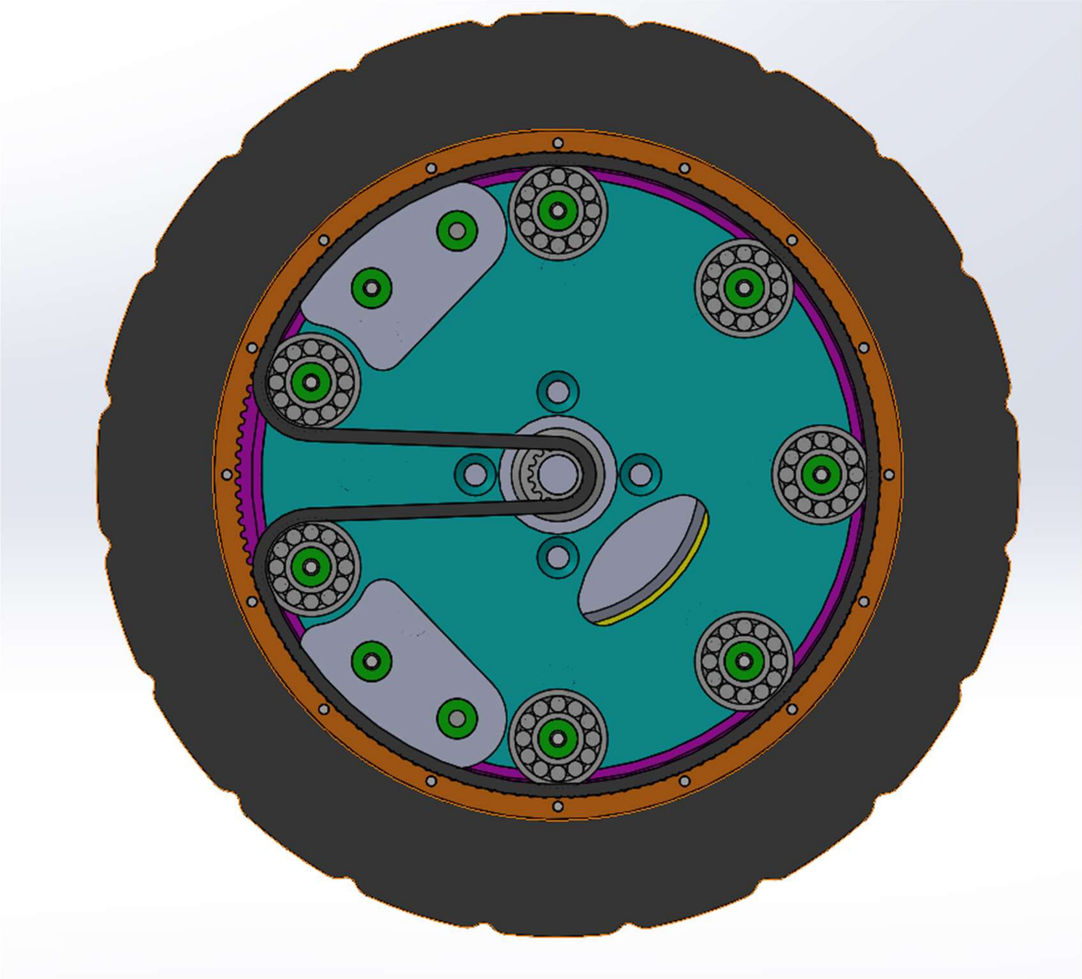


Figure 9 Location of dowel in holes radially constraining the ring pulley

The motor and electronics are housed in each hollow portion of the components shown in the red area of *Figure 9*. The yellow components form a weather-tight seal and provide mounting to the chassis. The bearings in *Figure 8* ride tangent to the back of the belt, radially constraining the belt to maintain consistent tooth engagement. Full co-radial support is unnecessary because the rotation of the system enables the bearings to flatten out undesirable bends in the belt, and full matching of the belt path was deemed unnecessarily complex for the scope of this project.

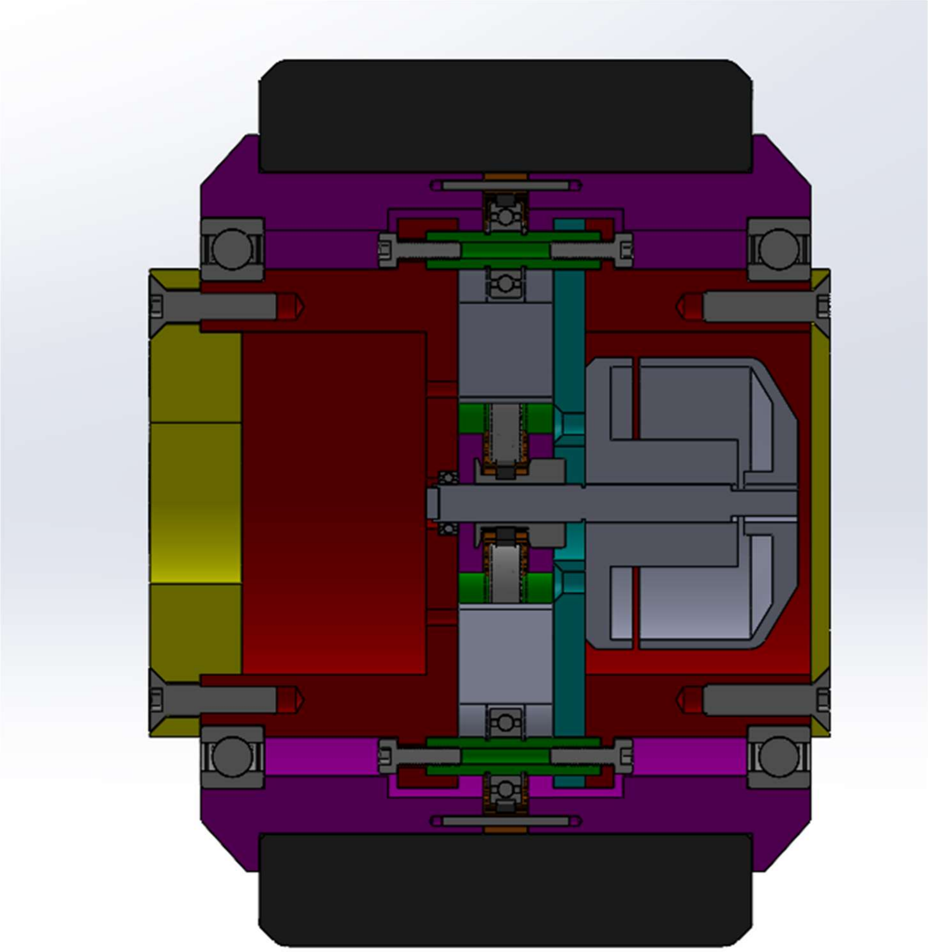


Figure 10 Front cross-section view of the wheel hub assembly

Interfacing geometry and indicating features ensured that components in shear were properly braced and impossible to assemble incorrectly. Ease of assembly, while not critical for the limited scope of this project, would bring this system closer to a mass-production-ready state.

Each motor was intended to have a magnetic rotary encoder to increase the efficiency and power output of the system. To do so, the stock motor shaft was replaced with a custom machined shaft, featuring a diametric cylindrical magnet on the end facing the encoder (*Figure 10*)**.**

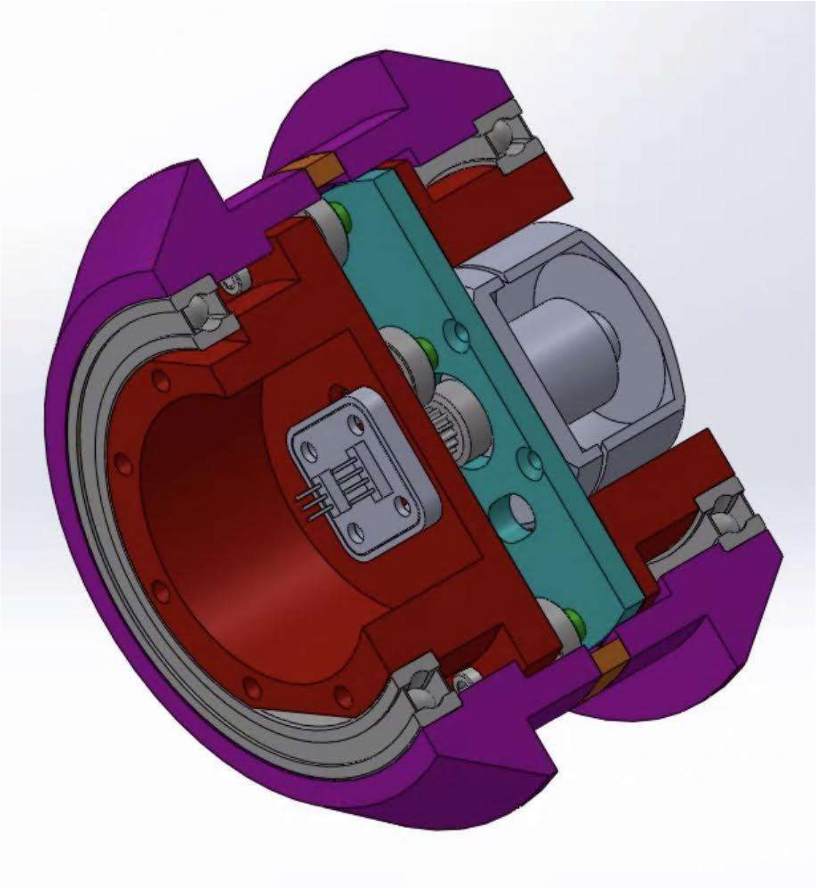


Figure 11 Wheel encoder integration

Prototyping

Due to the novelty of this drivetrain system, it was rapidly prototyped using 3D printed components before manufacturing a more expensive metal version. The focus of these tests was the accuracy of the belt profile and maintaining sufficient belt tension along the ring pulley.

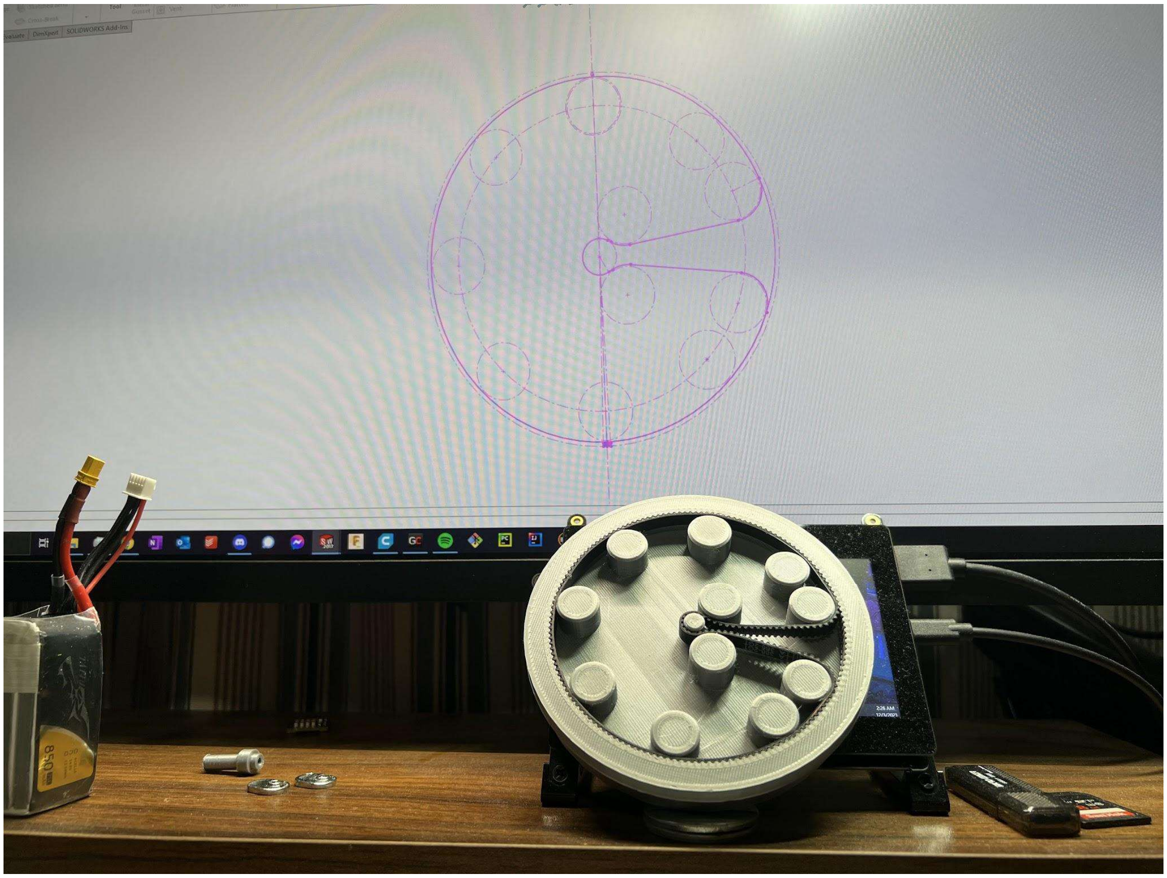


Figure 12 First 3D printed prototype testing the accuracy of the belt path master sketch

The first 3D printed prototype was designed to test belt tension. Solid 3D-printed bosses were in place of the pinion and bearings. After confirming the belt path was accurately modeled, the next step was to test a rotating assembly.

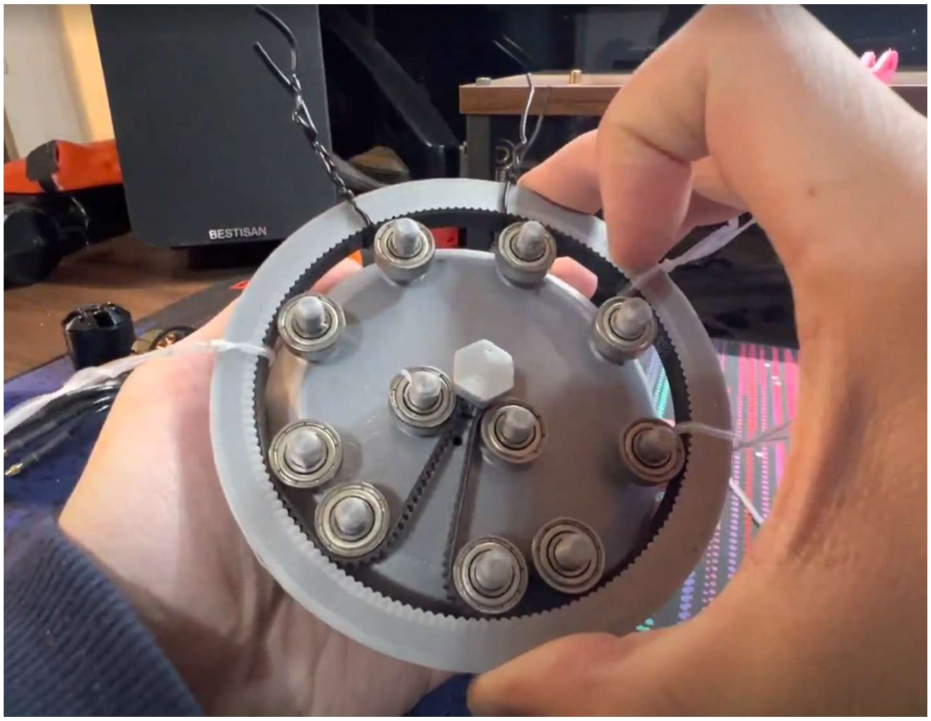


Figure 13 Second prototype using a printed pinion and bearings

The second prototype spun smoothly while the belt remained on the ring pulley. Due to the lack of retaining geometry for the bearings, belt, or the ring pulley in this limited test, the belt and bearings frequently fell off of their intended path, preventing a powered test. Additionally, several of the 3D printed standoffs broke along the 3D printed layer lines. Despite these shortcomings, this prototype demonstrated that the pinion was able to drive the ring pulley, even without being fully constrained. At this point of the prototyping phase, there was enough confidence in the system to manufacture a model capable of being driven by a brushless motor.

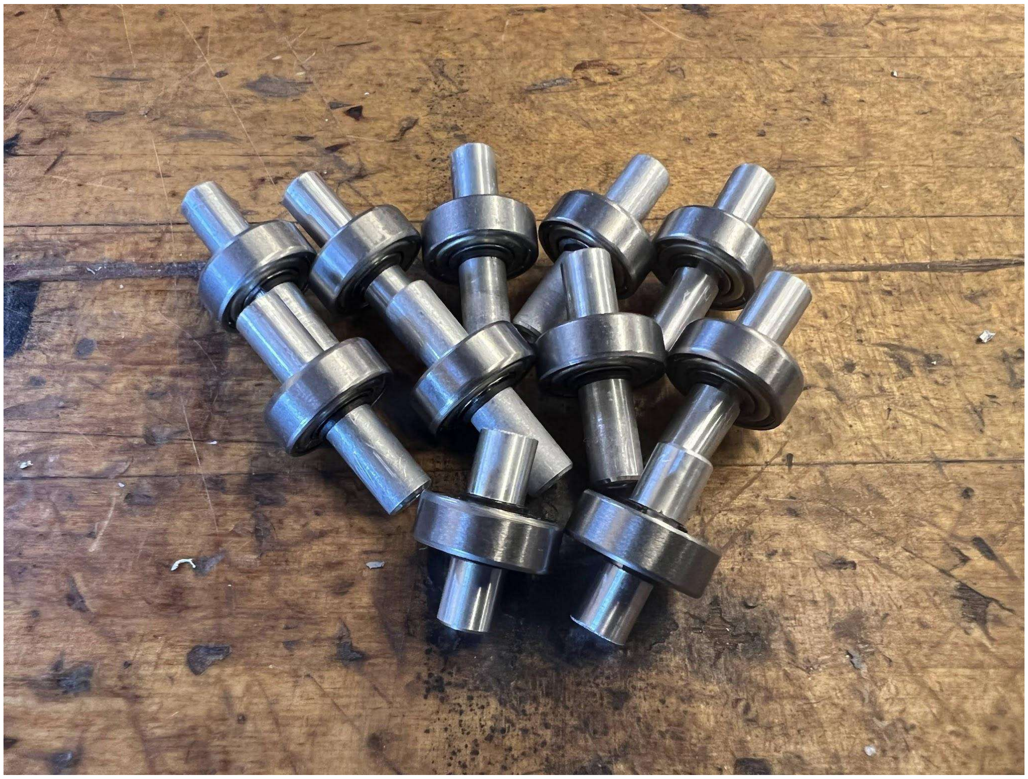


Figure 14 New Aluminum standoffs for bearing retention

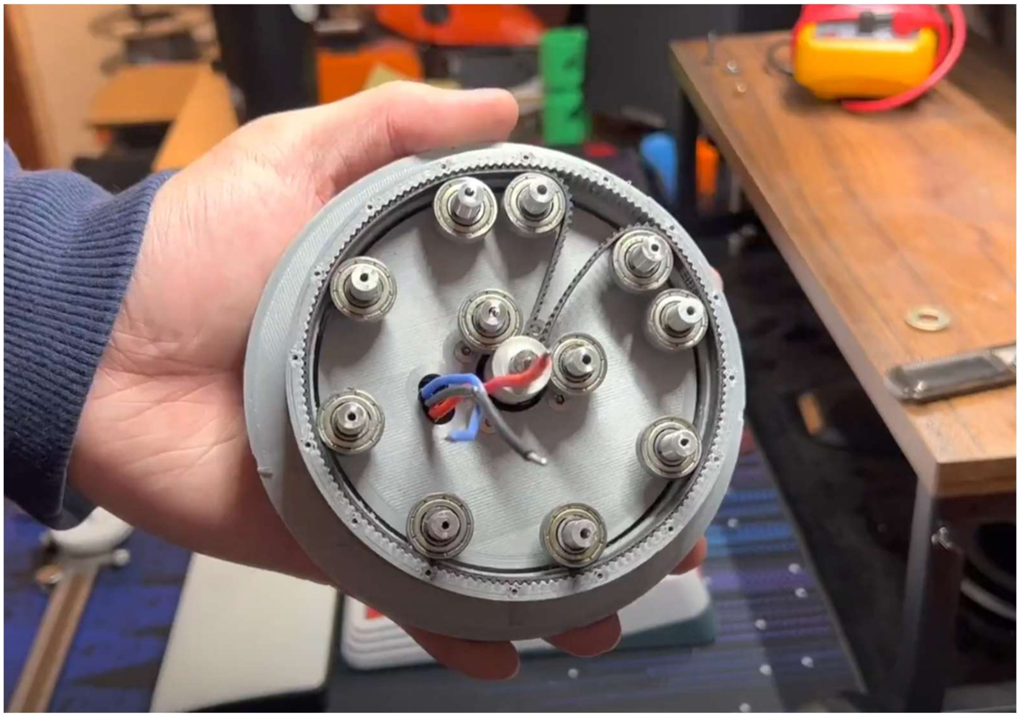


Figure 15 Printed prototype of half the hub assembly with a motor attached

The assembly shown in *Figure 14* was first spun up by hand, and this initial test can be found using this video link in Appendix C. After verifying that the system spun smoothly, the motor was wired into an external power supply and spun at full throttle, which can be seen in Appendix D. This 3D printed prototype performs well under light to moderate load but the plastic ring pulley flexes considerably when stalled, causing the belt to slip. This issue should be solved by manufacturing the ring from aluminum by compressing the ring with the other half of the hub assembly, and by indicating dowel pins. Another observation was that the bearings did not keep proper tension on the belt at high speeds. When the motor rotates the pinion in one direction, one side of the belt is tensioned while the other is compressed, and only the tensioned side can transmit torque from the center pinion to the ring pulley.

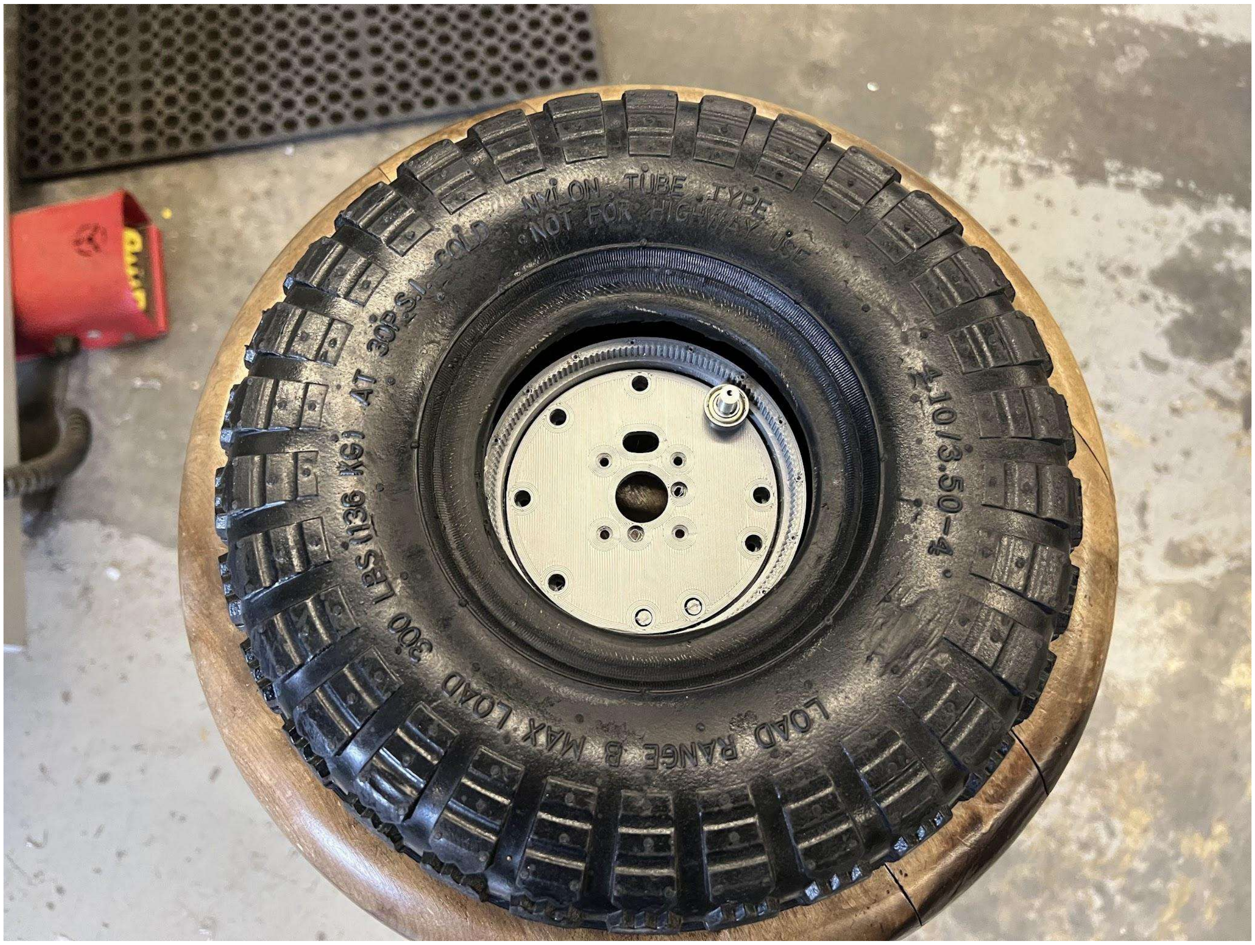


Figure 16 Testing the fit of a 10" pneumatic tire

Fitting the tire shown in *Figure 15* proved to be a difficult task due to the wire bead along the edge of the rim. These tires are intended to be installed onto stamped and tapered sheet metal rims which are then expanded using a pneumatic tube. For this system, however, there is insufficient space for a tapered hub or a pneumatic valve. The alternative solution proposed in this project was to fill the tire with an expanding foam mixture, making a more robust wheel and filling the tire without the need for pneumatics. Upon further testing, this transmission proved incompatible with the wire bead on the tire and thus an alternative tire was required. The first alternative tire is shown in *Figure 17* and is a solid foam wheel.

Unfortunately, the tire bore was still smaller than required by the belt box transmission, so the wheel bore had to be enlarged for the wheel to be useful. This was completed by turning the tire on a manual lathe and using a heated boring bar to open the bore by approximately 1/8“ (*Figure 17*). Because of the low density of the foam, the boring bar insert wasn’t able to effectively form chips out of the wheel. While turning, the foam elastically deformed around the boring bar insert, making it difficult to cut. A propane torch was used to apply a constant amount of heat to the cutting edge, allowing for easier material removal (Appendix E). The added heat expanded the foam, allowing it to be cut rather than compressed by the boring bar.

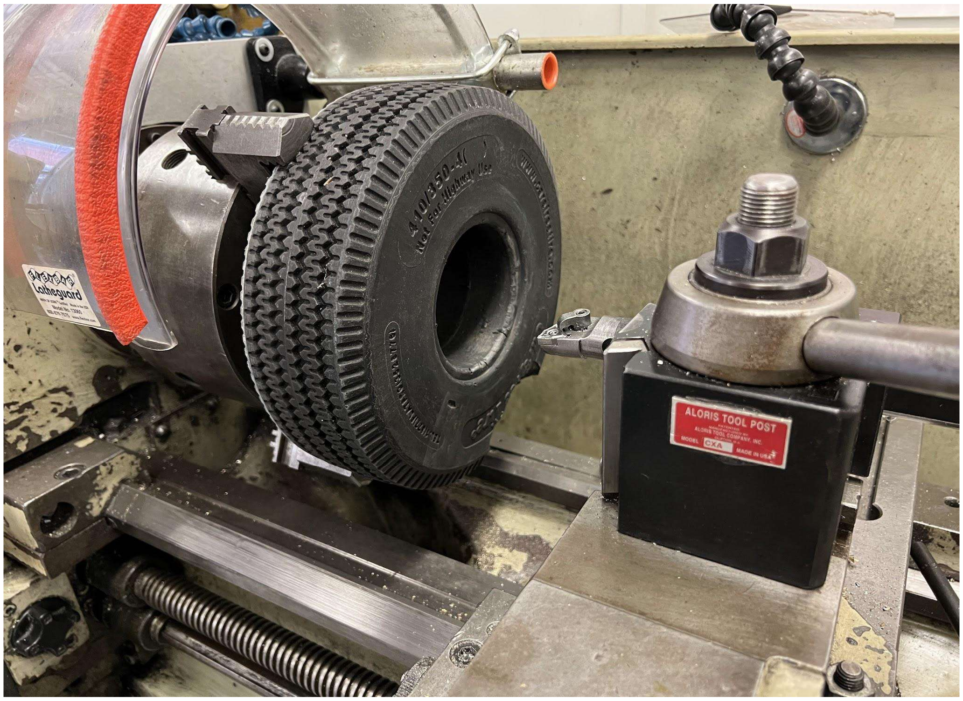


Figure 17 Carefree tire in the lathe before boring out the center

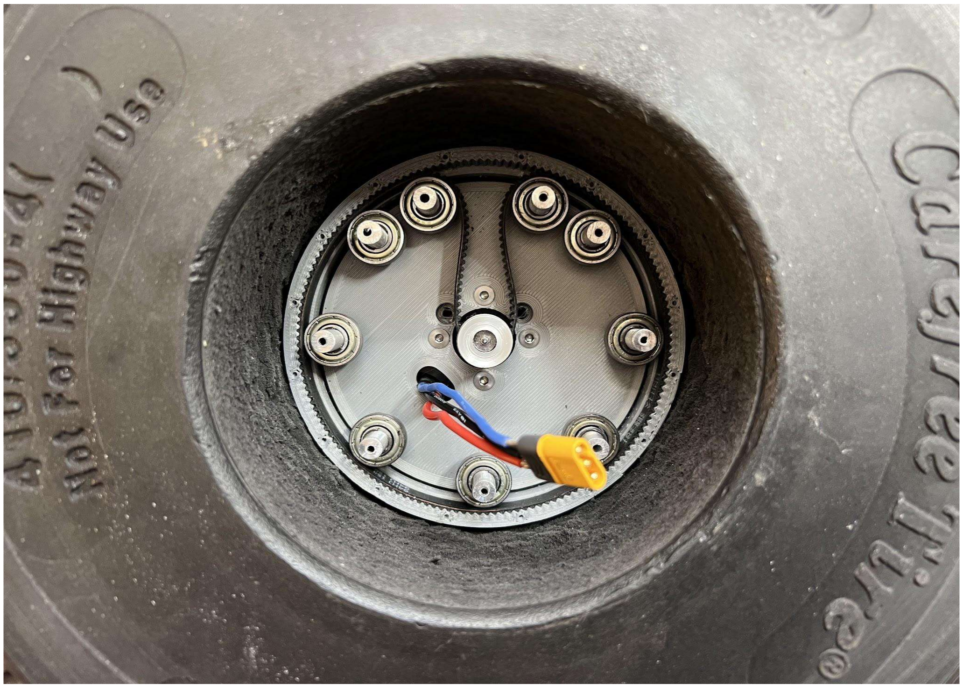


Figure 18 Hub inserted into the Carefree tire

After boring out the tire, the tire made a light press fit with the hub, increasing ease of assembly. Although the boring approach was successful, the outer diameter of the wheel needed to be decreased to achieve the target robot speed. Two attempts were made on the manual mill where the center bore was chucked up in a four-jaw and we tried conventionally turning the tire down. Even if we added heat, the outer diameter refused to turn down. We then tried to use a hole saw but were uncomfortable with how fast the four-jaw would need to be spinning in order for the teeth to engage correctly with the material. The third attempt to decrease the wheel diameter used a vertical bandsaw and rotary guide to cut a smaller outer diameter onto the wheel, but the outer diameter was not concentric with the inner diameter making such a process infeasible.

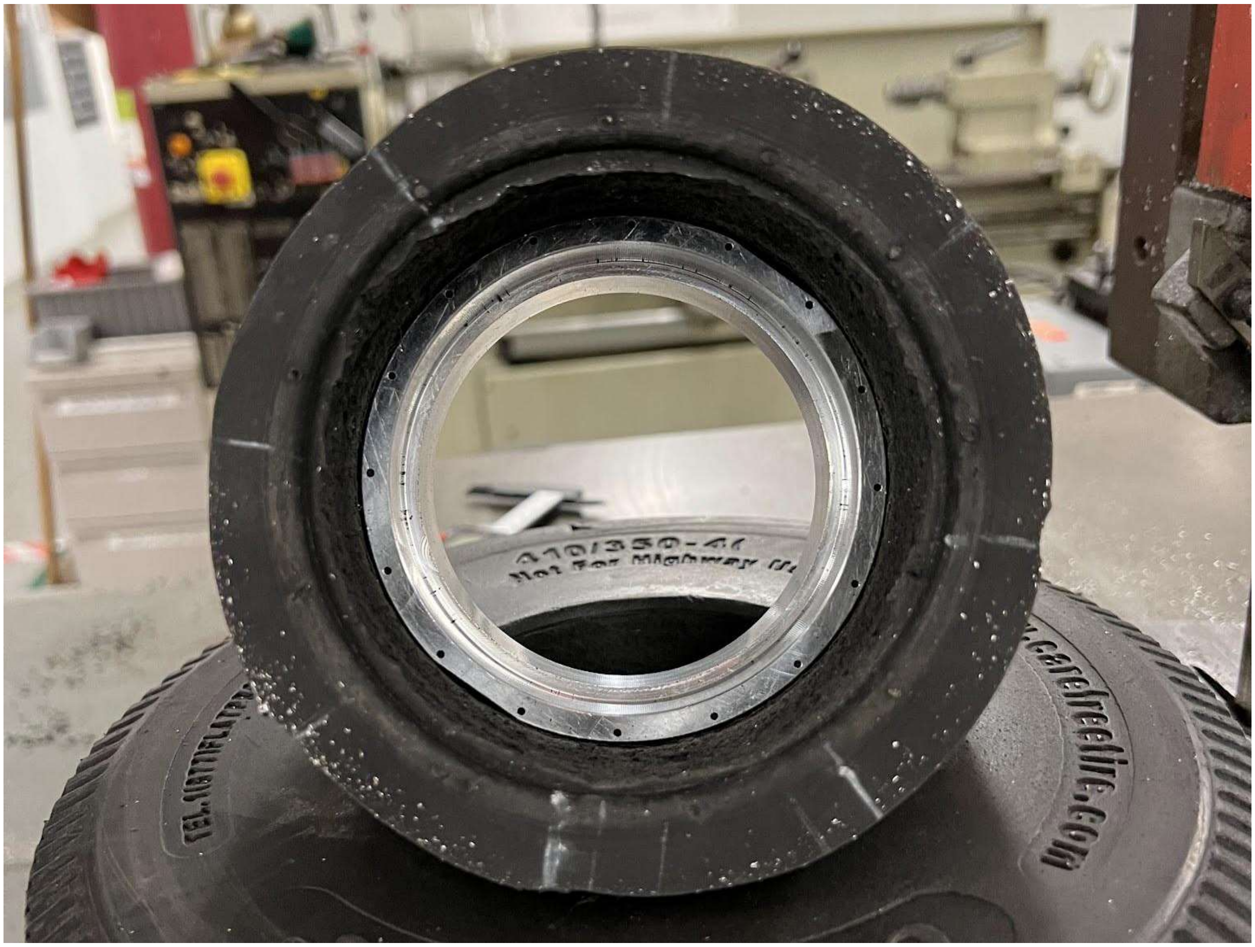


Figure 19 Attempted outer diameter reduction on a Carefree tire did not turn out concentric

The final tire was manufactured by 3D printing a flexible TPU filament. TPU is abrasion-resistant, flexible, durable, and able to be easily 3D printed. The flexibility of TPU loosened tolerancing restrictions and the tire bore diameter was undersized by 0.010" to form a tight fit onto the hub. The press-fit between the flexible TPU surface and aluminum hub, in addition to the clamping force between the wheel hub lips, provided sufficient friction to negate the need for interfacing geometry or adhesives. This allowed for easier assembly and disassembly.

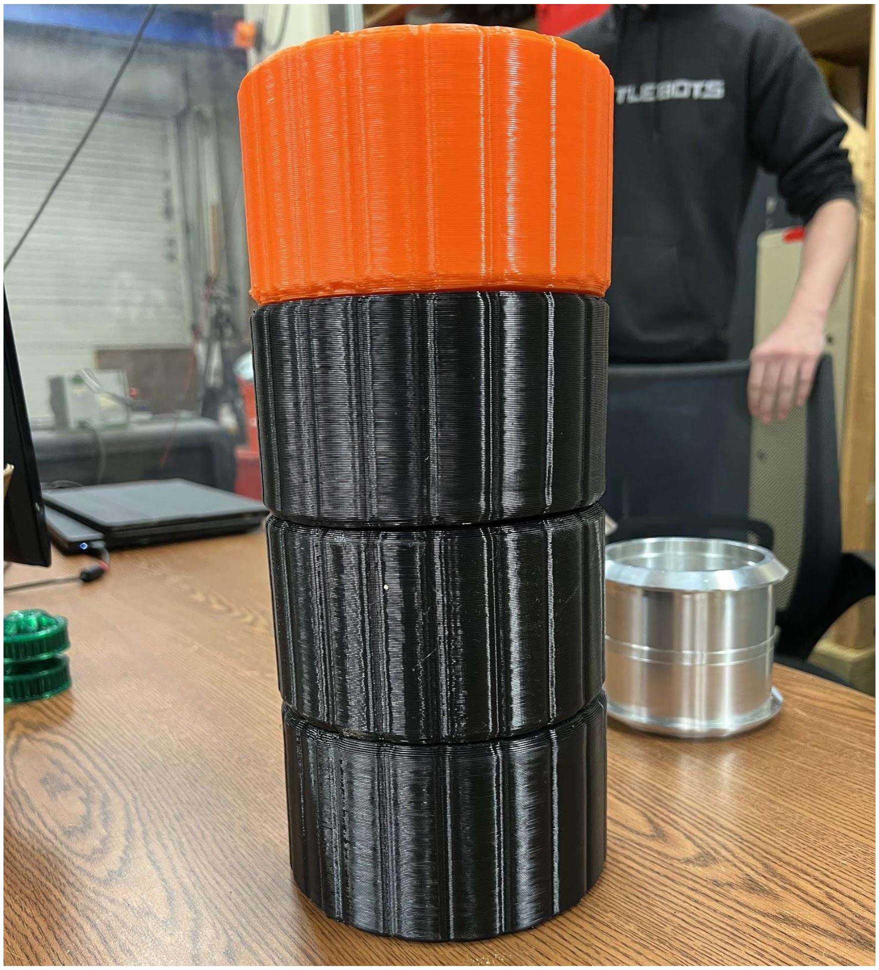


Figure 20 3D printed wheel hubs

Final Wheel Construction and Testing

Although 3D printing allows high-speed prototyping of complex geometries, standard plastic printing materials are weaker and more flexible than machined aluminum components and thus had limited usefulness. 3D printed pulleys were also subject to the limited resolution of 3D printers and this caused the tooth profile to be slightly incorrect. These errors in the printed tooth profile result in the belt slipping and not properly meshing with the printed ring pulley. For the final components, the tooth profile was custom manufacturing using a wire EDM in the Washburn shops (*Figure 21*).

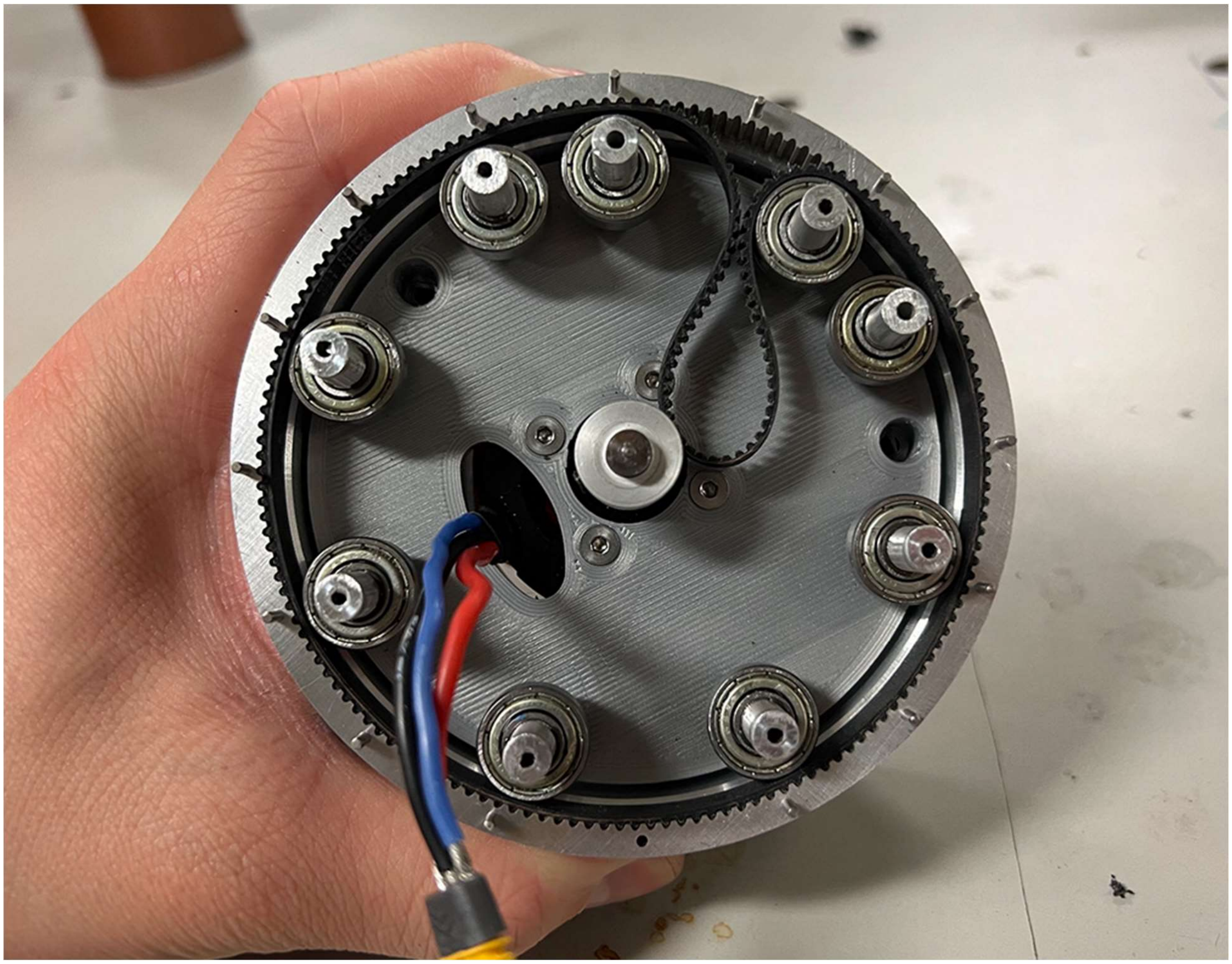


Figure 21 Wire EDM cut aluminum ring pulley

Manufacturing the ring pulley from aluminum revealed a new problem: the relatively small bearing outer diameter relative to the ring pulley diameter resulted in an insufficiently small contact area between the bearing and belt. This small contact area meant that only 1-2 teeth around each bearing were properly meshing with the ring pulley, allowing for the small teeth of the GT2 profile to skip.

The first attempt to increase the number of meshing teeth while keeping minimal friction was to replace the leading 3 bearings with acetal sliders (*Figure 22*), constraining approximately 30 teeth. This configuration was not backdrivable, revealing significant internal friction.

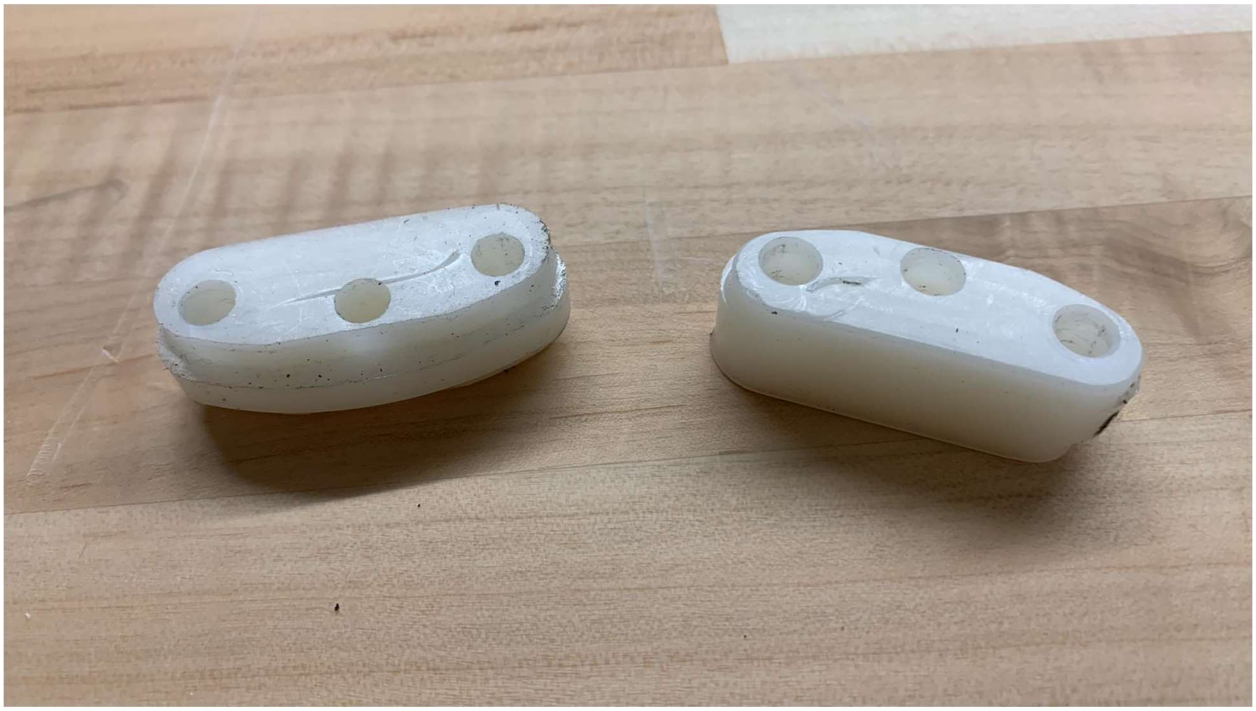


Figure 22 First version of acetal sliders

The primary location of friction along the acetal slider was the leading edge, due to the sharp belt radius leading to the pinion, so the previously leading bearing was reintroduced. The new acetal slider replaced the second and third bearings, rather than the previous first three bearings, while still constraining approximately 20 teeth (*Figure 23)*. This solution provided significantly less friction and allowed the motor to spin the hub assembly without belt slippage. This test provided sufficient confidence in the wheel assembly to manufacture three more.

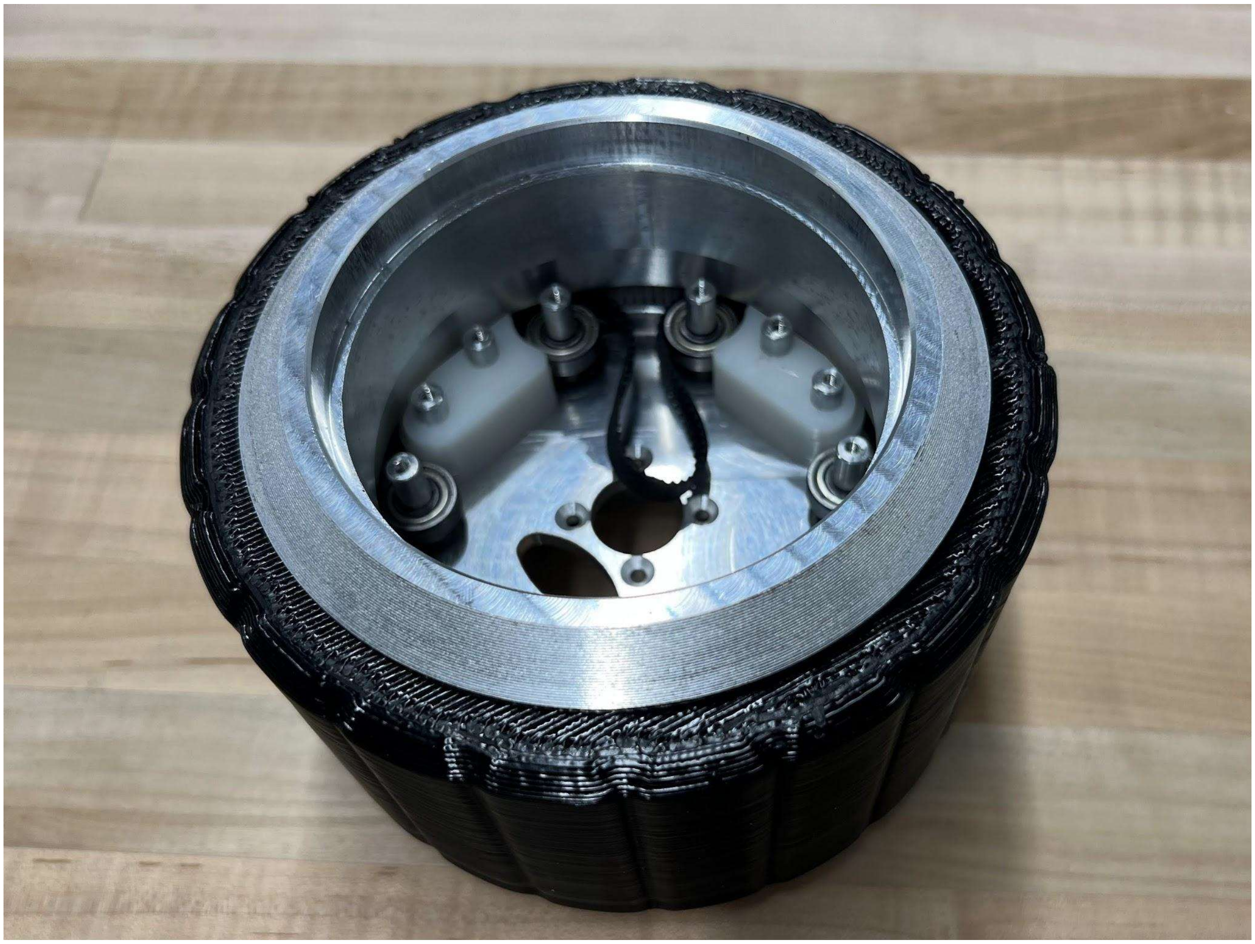


Figure 23 Split bearing and acetal retention system

Four custom wheel assemblies were manufactured in the WPI Washburn Shops. The primary on-campus resources used were CNC machines, CNC lathes, and a wire EDM. The CAM for each CNC part was created using Fusion360.

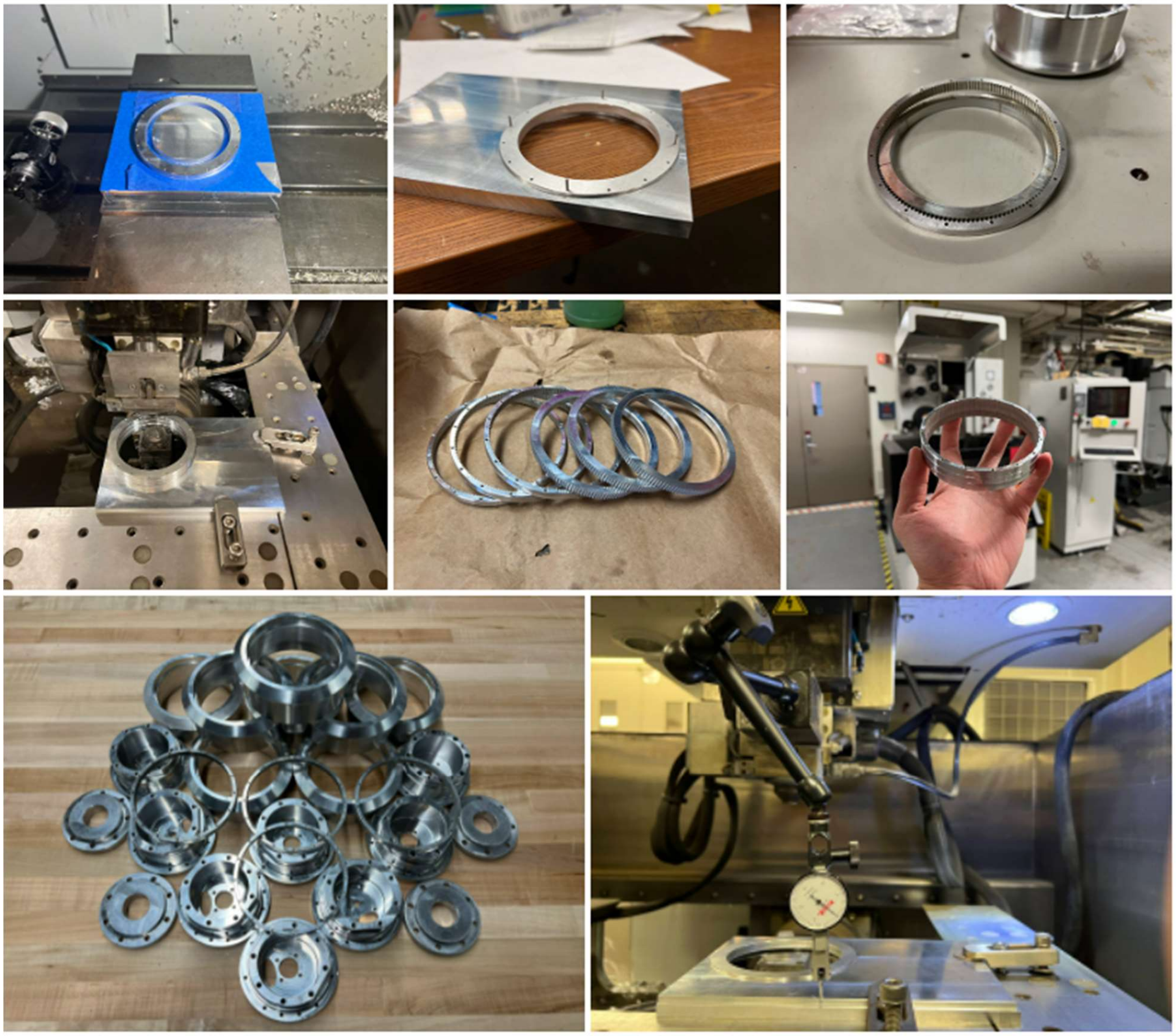


Figure 24 Manufacturing process to make the inverse timing belt box hub motor

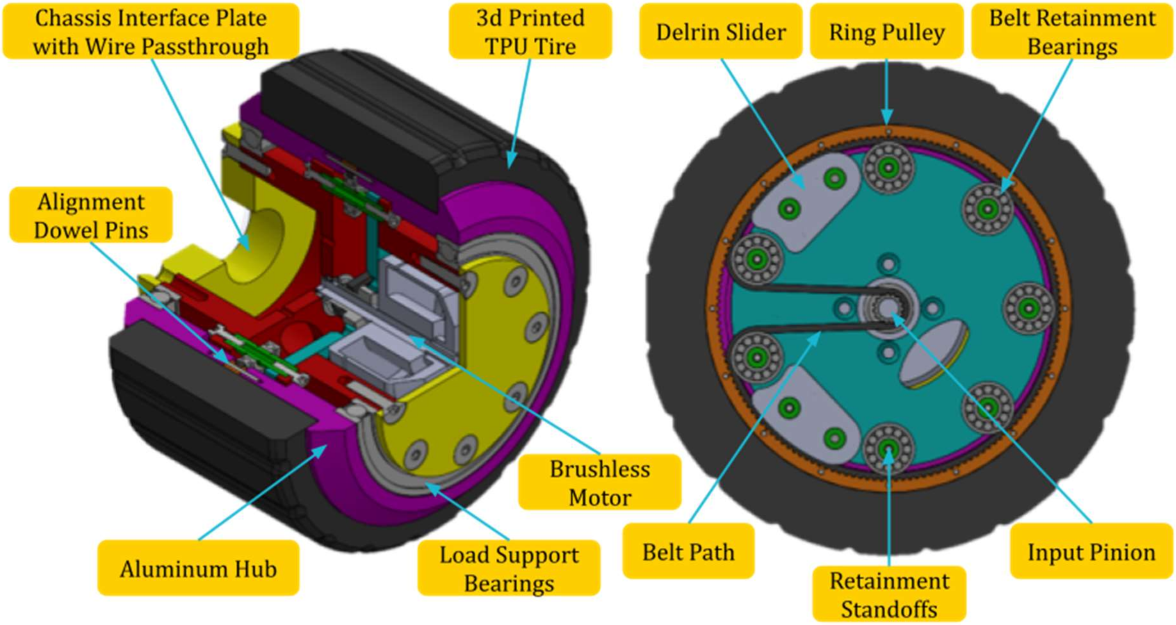


Figure 25 Labeled assembly view of hub motor system

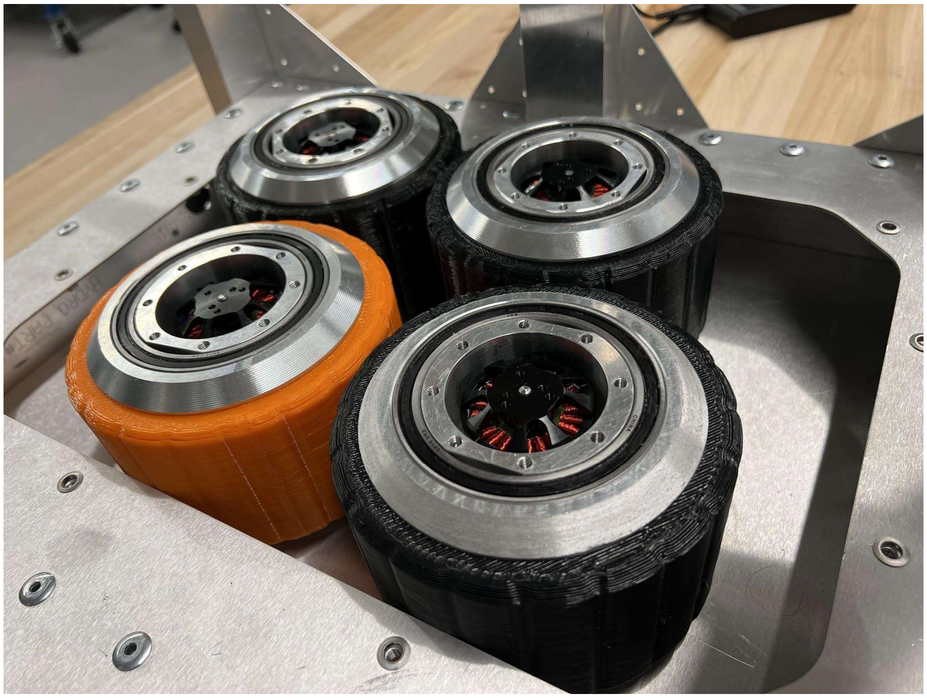


Figure 26 Fully assembled hub motors

Chassis Design

Due to the limited budget, a low-cost approach needed to take place while still allowing for maximum functionality of the robot. Several different materials and building structures were considered during the brainstorming phase. The following table details the several design structures considered:

Structure System	Pros	Cons
Billet Aluminum	- Solid construction - Team members are experienced with this kind of fabrication and design - Easy to waterproof	- Expensive - Heavy - Fabrication is time-consuming
80x20 Extrusion	- Go-to method for building fast structures - Rigid enough for our use case	- Expensive - Still requires machining - Joining techniques yield low tolerances - Difficult to weatherproof
Bent Sheet Aluminum	- Lightweight - Can achieve more complex shapes and assemblies	- Team is inexperienced with sheet metal design - Fabrication methods are lacking in Washburn shops
Wood	- Low cost - Lightweight - Materials are easy to obtain	- Not weatherproof - Lacking fabrication methods - Can easily splinter/material properties are undesirable
Aluminum Box Tubing	- Low cost - Lightweight - Easy to design	- Harder to create complex shapes - Will likely require some welding which the team is inexperienced with for waterproofing purposes

Table 1 Pros and cons list of different structure systems

After assessing the pros and cons of each structure, a box tubing chassis was selected. The ability to make robust structures quickly and at a low cost was the main selling point. The box tubing design was completed using a custom weldment profile and the weldment feature to construct the chassis frame. The chassis was held together using sheet metal gussets and rivets.

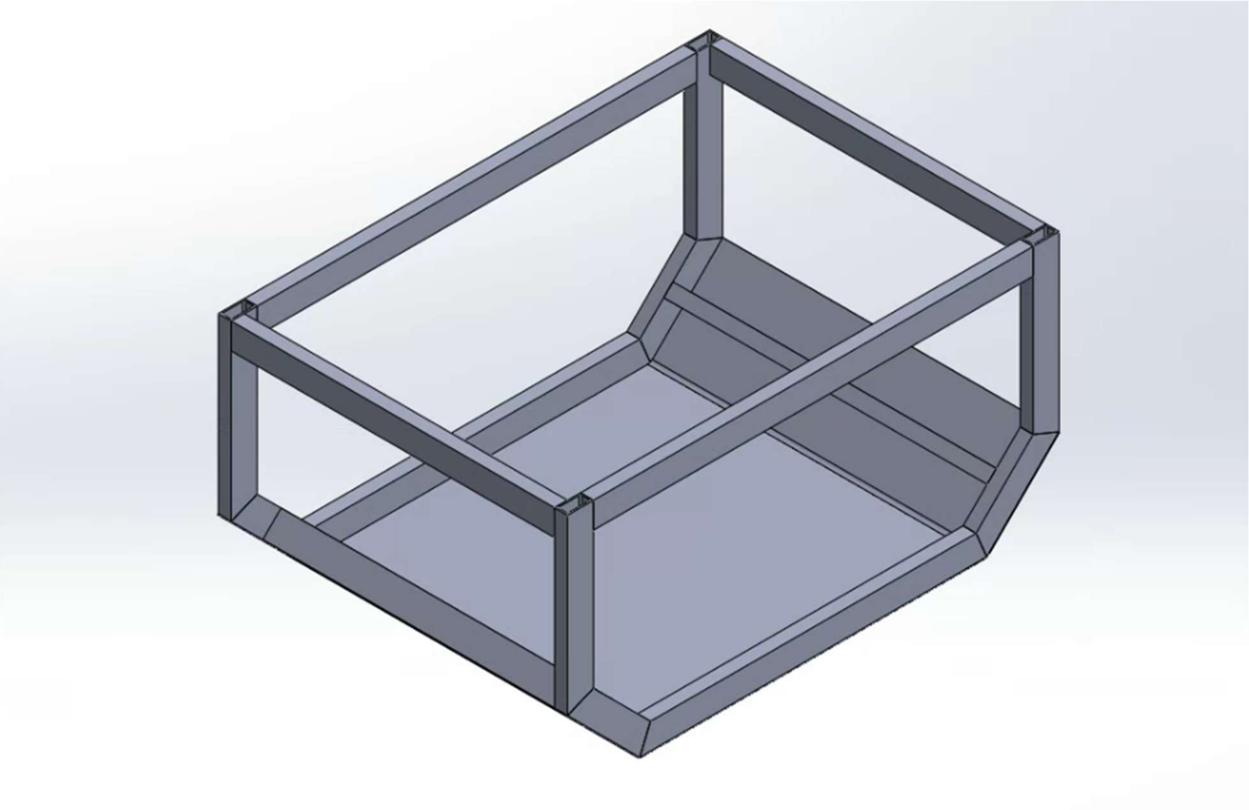


Figure 27 Box tubing frame prototype CAD for the robot

Figure 27 showcases an early chassis design, but the bends in the chassis wasted material and made the payload storage solution more complex than necessary. In addition, the belly pan utilized a bent sheet metal belly pan which goes against the decision matrix in (*Table 3*). Due to this unneeded complexity, the chassis was redesigned to optimize payload size for given material usage.

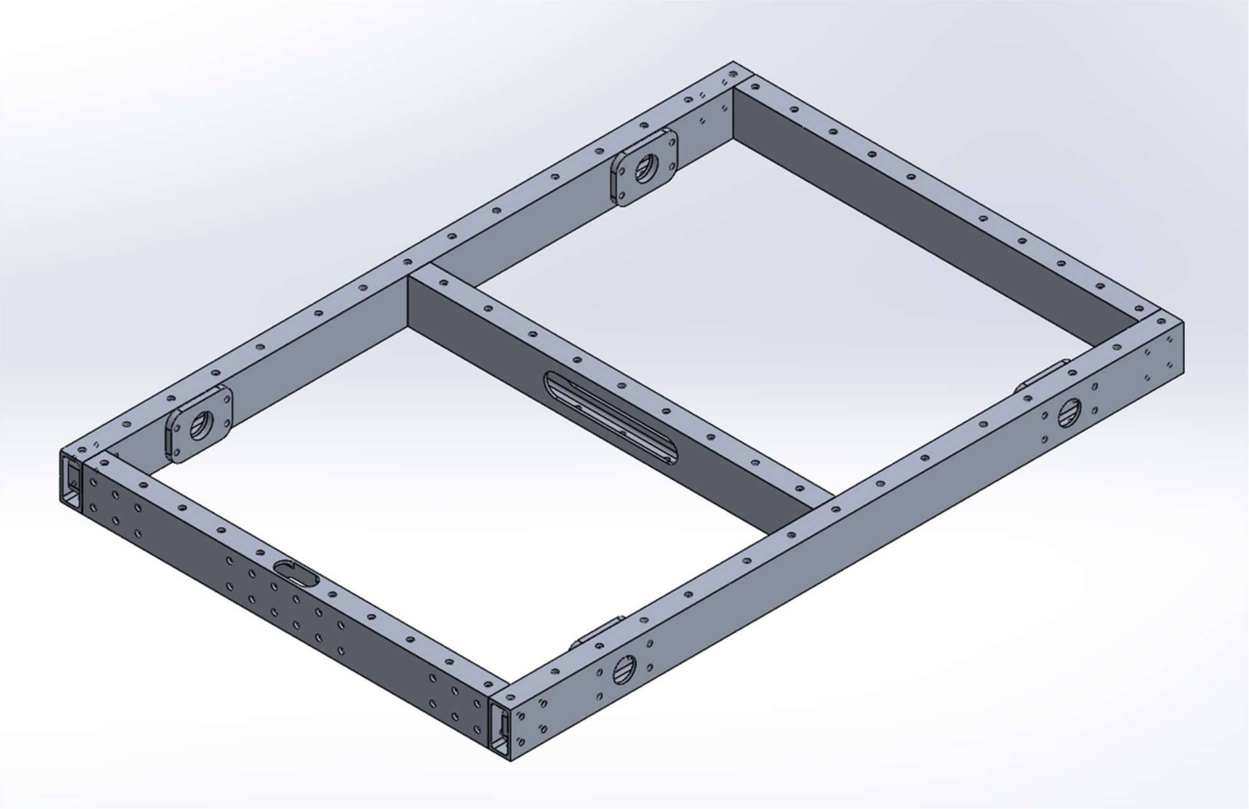


Figure 28 Updated chassis base

Most structural components were constructed primarily from 1/16” thick 2”x1” aluminum 6061 T-6 rectangular tubing. Each piece of tubing is joined together with pieces of aluminum sheet metal of varying thicknesses and 1/4” rivets.

The entire robot underside is covered using an 0.080” thick sheet of laser-cut aluminum. The top uses a 3-piece design: one piece is a central structural component and the other two are easily removable to service the electronics (*Figure 29*).

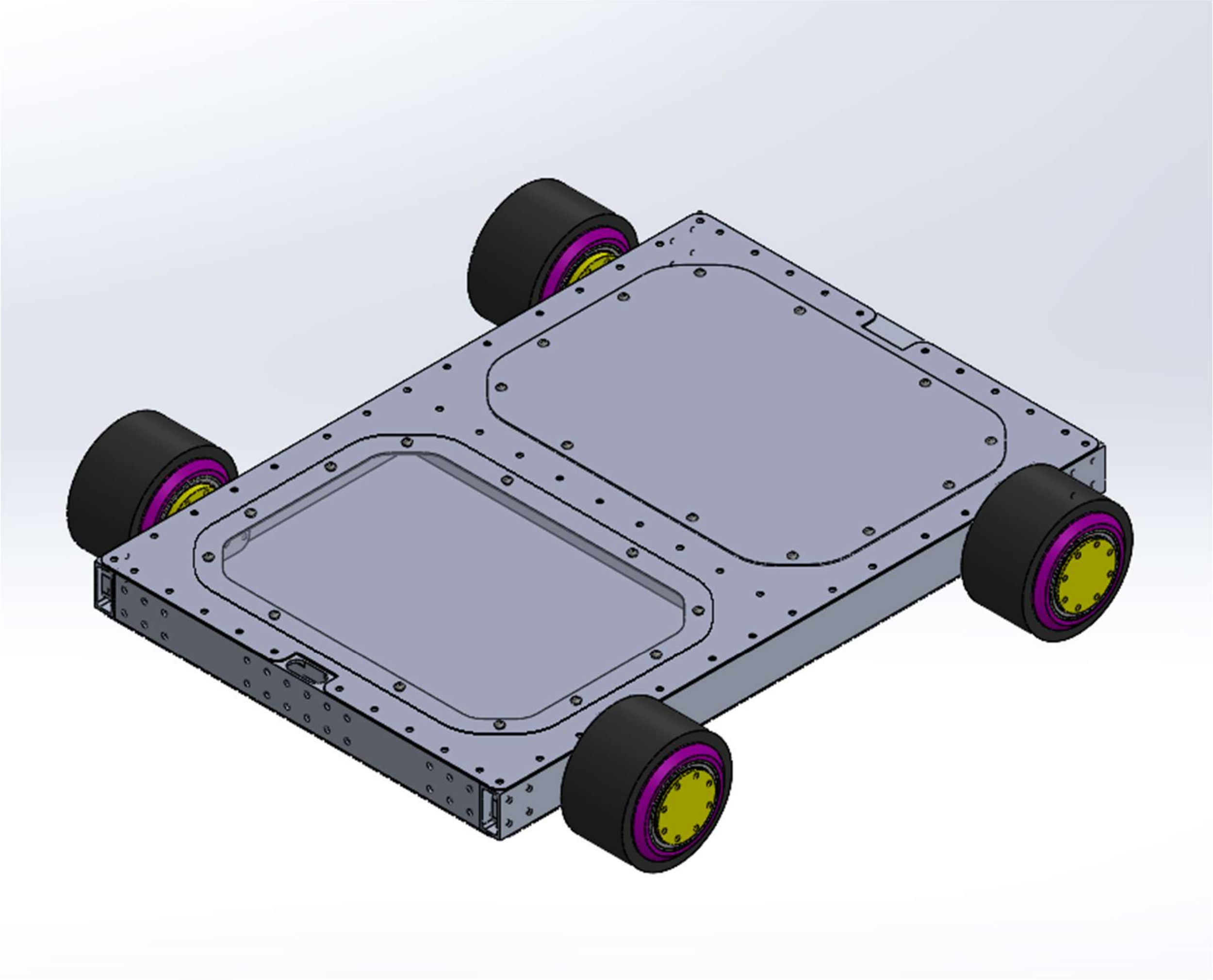


Figure 29 Robot Chassis with top and bottom plates

Payload Storage Design

Payload storage construction follows the same design philosophies as the main chassis and utilizes the same fabrication and assembly techniques. Several tubes have cutouts for wire pass-throughs to power all wheels from a single central electrical system. Payloads were carried using an Amazon storage container that is made out of polyethylene fabric **(***Figure 30***)** and retained using an aluminum structure.

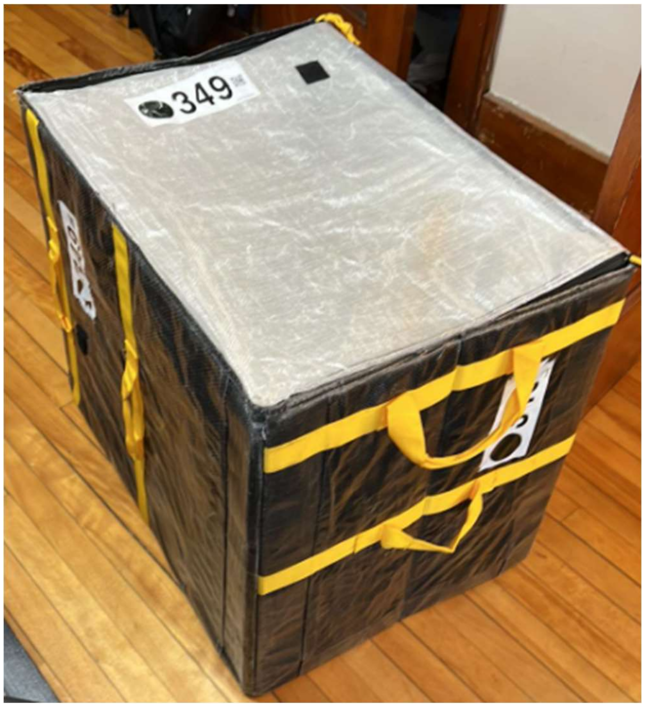


Figure 30 Amazon polyethylene box used to store payloads

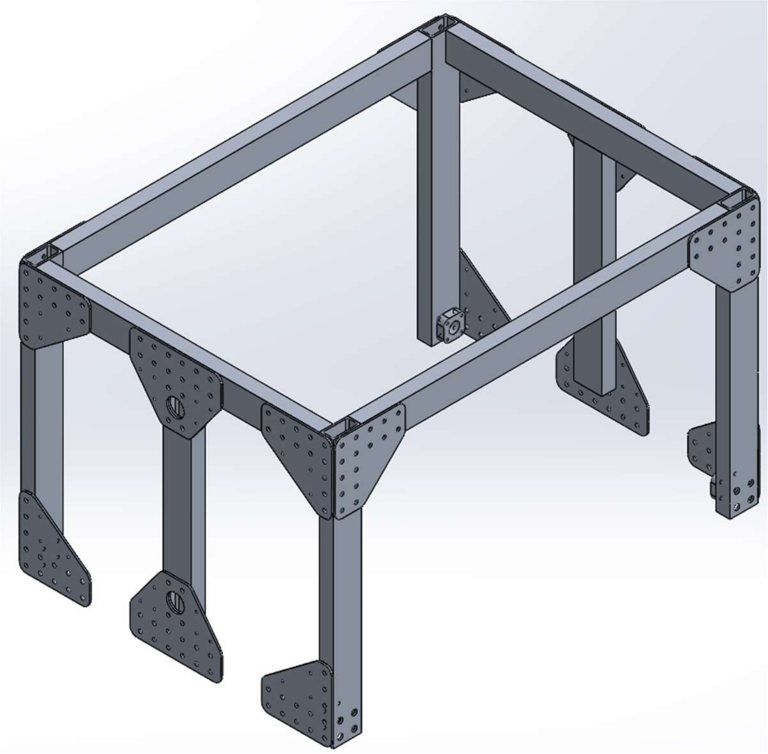


Figure 31 Aluminum structure used to retain the box

Full System

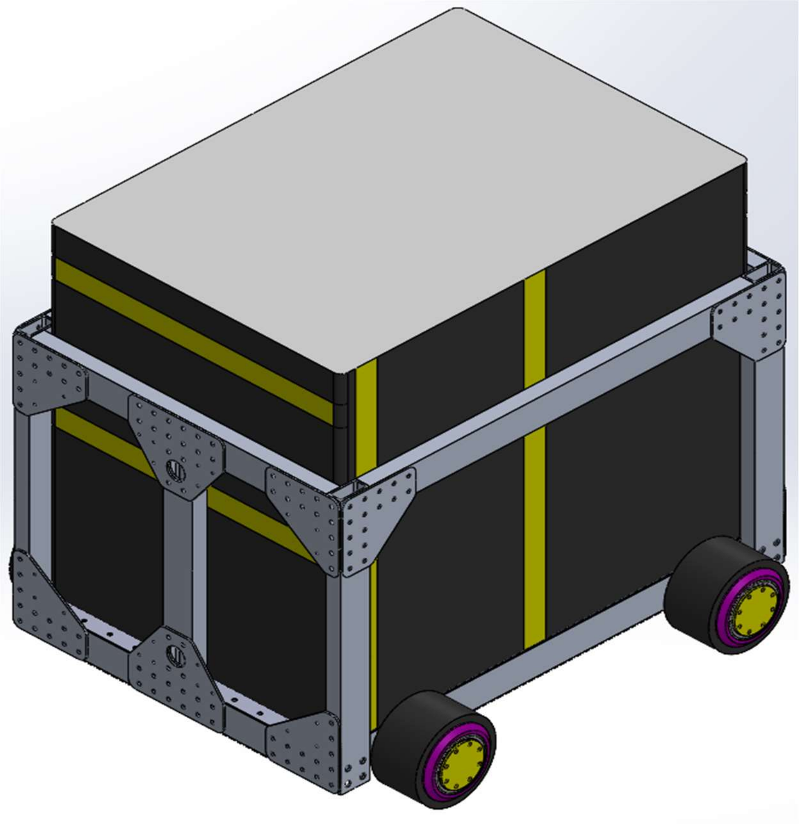


Figure 32 Full system CAD

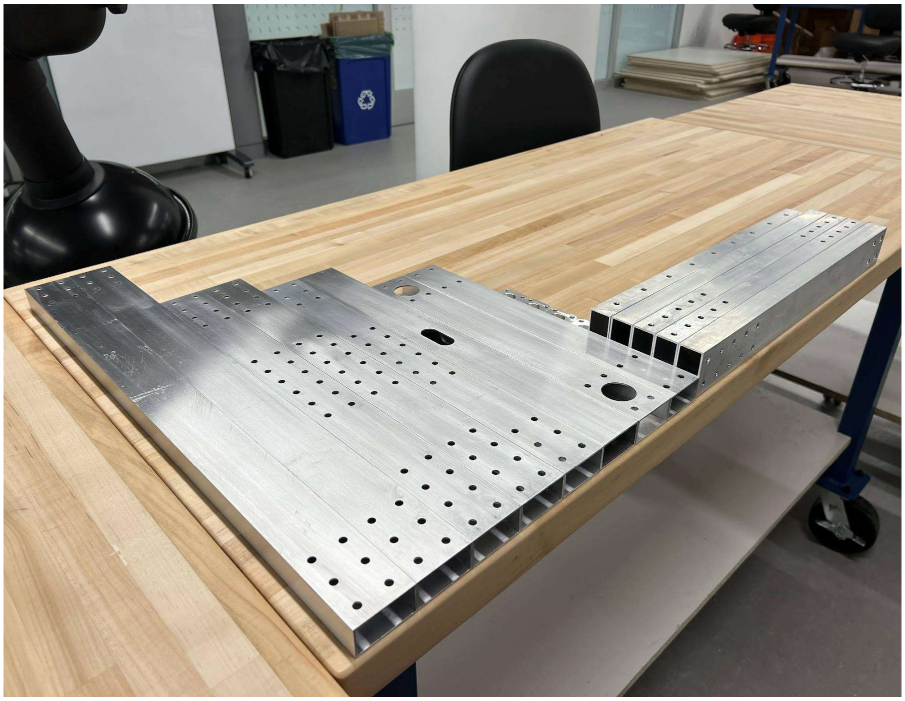


Figure 33 Chassis and payload storage structure components

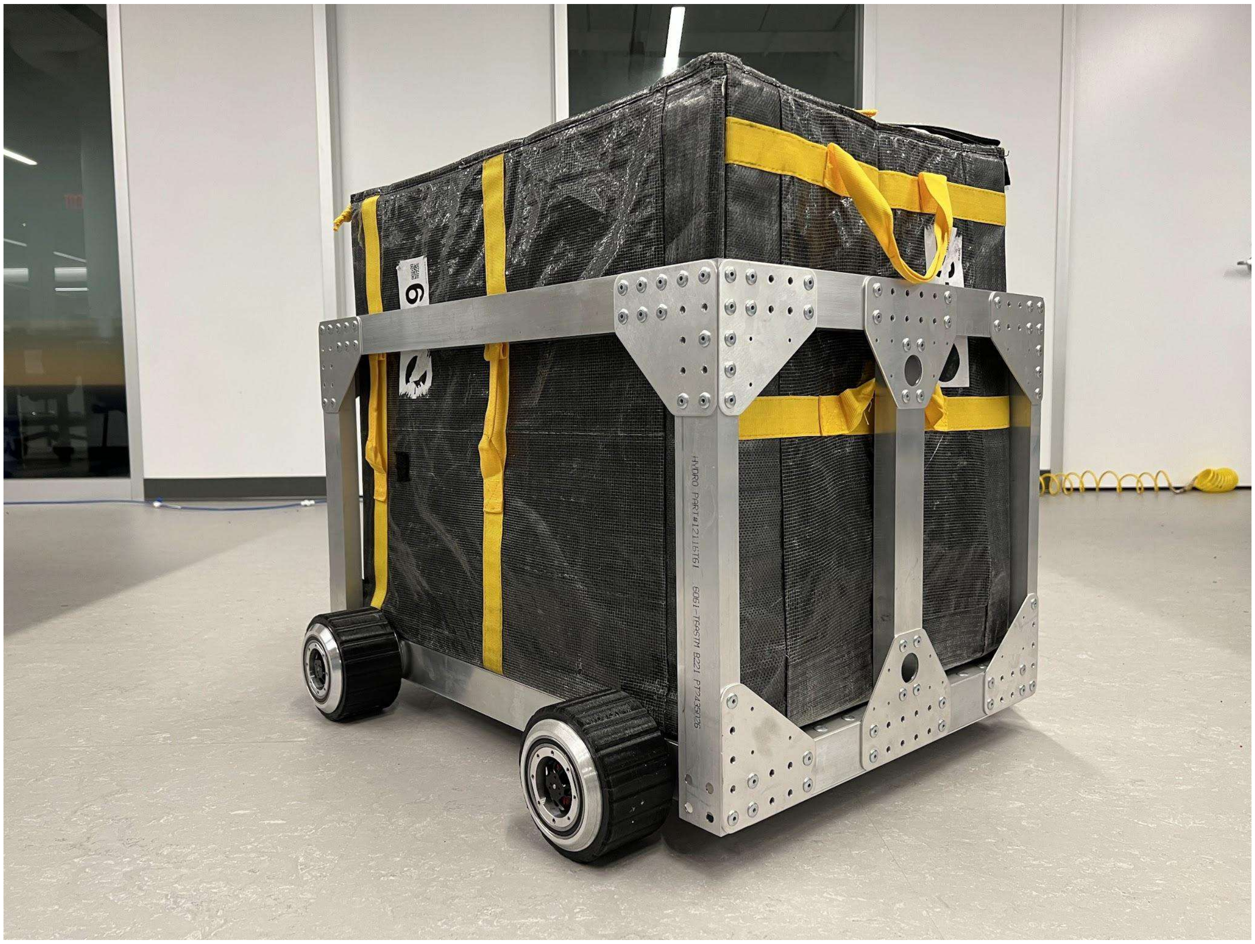


Figure 34 Full assembly of final robot

Belt Box Testing Methodology

The novel reduction system was tested by applying various loads and measuring the electrical inputs and mechanical outputs of the system. The startup torque of a wheel assembly was tested by inserting a hex shaft into the TPU print with a near tangential mounting approach, as shown in *Figure 35*. The shaft began in contact with a scale and the scale was tared. The motor was then powered and pressed the shaft into the scale to measure the force applied by the wheel at a given radius. The continuous torque was measured by pressing a scale tangentially into the wheel and measuring the torque reading when the assembly stalled. The coefficient of friction between the wheel and scale was calculated by taking the tangent of the angle of the scale at which the wheel could slip down the scale surface. The angle was measured with an iPhone measure app. The speed of the motor was tested using ERPM readings from the motor and dividing by the pole count to get the actual motor RPM. This RPM value was verified using a strobe light tachometer.

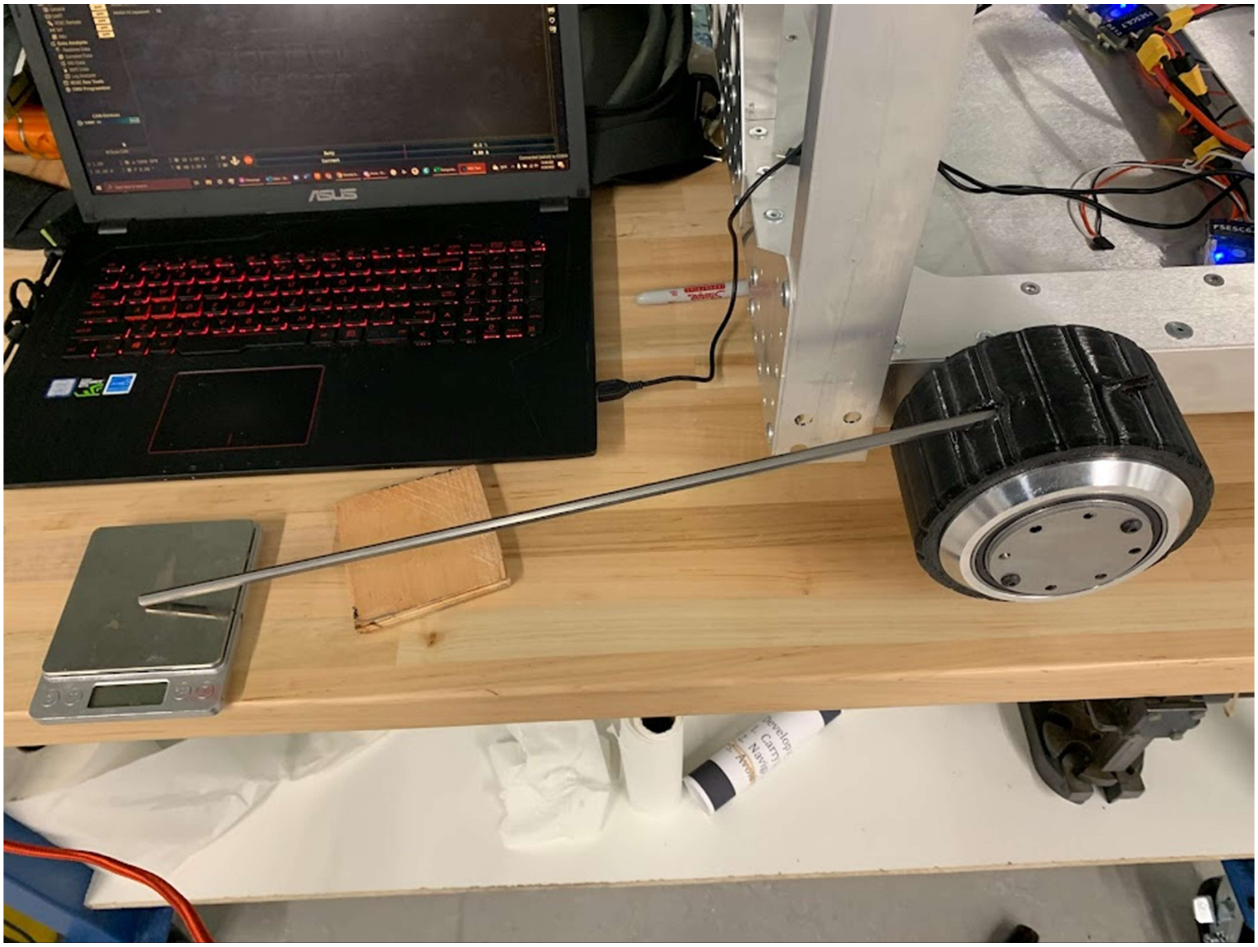


Figure 35 Startup torque testing of wheel assembly

Electronics

Central Processor

The Raspberry Pi 3B+ was initially selected as the central processor due to its wide availability, extensive documentation, and low cost. Initial testing on the prototype chassis was completed with this processor, but it lacked sufficient computational power for the applications of this project. This was evident when a graphical user interface was loaded onto the device and simple tasks, such as opening a web browser, were unacceptably slow. As a result, the Raspberry Pi 4, 4GB version, was subsequently selected. While the increase in computational power was evident through the user experience while interacting with the graphical user interface, the processor generated much more heat. This high heat output was addressed through the usage of adhesive heat sinks on the core components.

GPS

The team determined that the best way to track the precise location of the robot was through the use of a high-precision GPS. The GPS was selected as the primary localization method due to the high precision of existing solutions, which can have centimeter-level precision, theoretical ease of implementation, and robustness against the accumulation of error over time. It was determined that the optimal GPS for this application would be the U-Blox F9P. This GPS was preferred due to its dual-frequency nature, resulting in centimeter accuracy, as well as its ability to use internal RTK processing or RTKLib processing. However, due to its relatively high cost, at $275, this component was outside of the limited budget.

Due to the aforementioned financial limitations, our next choice of GPS was the U-Blox M8T, which cost $70 for similar accuracy. The reason for the significant price decrease is that the M8T module lacks internal RTK, which could be made up by using the external RTKLib instead and being single frequency rather than dual. The single frequency could have potentially caused problems because the recommended range to a CORS station was 10km for single-frequency receivers, 20km for dual-frequency receivers, and the closest station was 12km away. Despite this slight deviation from the recommended precision, we determined that the M8T GPS would be a good option for our application.

Before purchasing the M8T GPS, the team discovered that another MQP team had 2 GPS units, one of which could be used for this delivery robot to reduce project costs. This GPS is a U-Blox C94-M8P which was intended to be used in a base-rover pair, but was able to be reprogrammed, using UCenter, to act as two rovers interfacing with RTKLib and a CORS reference station.

To pull the RTK base data from the CORS station, RTKLib was run on the Raspberry Pi and connected to the MACORS network, specifically the Northborough station. The data was pulled from the MACORS server using ntripclient and then fed into RTKLib along with the GNSS data from the M8T GPS. RTKLib is then able to process the data to determine a precise location for the robot.

The team was unsuccessful in properly integrating the RTKLib data with the GPS reading because the GPS was only able to support one-way communication. As a result, we returned the C94-M8P to the previous team and purchased the NEO-M9N. The NEO-M9N cost $69.95, which is within the limits of our budget, and guarantees 2-meter accuracy, which produces satisfactory performance when accompanied by computer vision. This GPS was easily integrated with ROS through referencing code from the UBlox GPS GitHub.

Lidar

Although the GPS can provide a precise location for the robot, a GPS is unable to detect transient obstacles in the robot's path. Thus, lidar was added to the robot to support obstacle detection and collision avoidance. Initially, an "LDS-01 360" Lidar was used due to its availability from the WPI RBE3002 course, which was not in session during initial testing. This lidar was able to be successfully implemented into the Raspberry Pi, visualized in RVIZ, and incorporated into a script that avoids obstacles. At the end of A term, the lidar had to be returned to the school for use in RBE3002 and a new lidar had to be sought out.

The lidar decided on was the RPLidar A1M8. The manufacturer, Slamtec, is a reputable supplier and is known to have well-developed ROS integration. This lidar also features a relatively long-range (12m), high sample rate (8000 times per scan), and high speed (2-10 Hz). The primary two drawbacks are the relatively high price (~$100) and potential failure in direct sunlight. We determined that the long-range and manufacturer reputation was worth the high cost to mitigate the risk of a cheaper lidar being unable to meet our specifications.

Camera

In addition to the GPS, the camera assists the robot in staying on the path. The camera was intended to point toward the ground in front of the robot such that it can see approximately 10 feet in front of the robot and 3 feet to either side of the robot. While the robot is navigating, the camera could take images of the path ahead, to be processed and determine the edges of the path, the camera's orientation relative to the path, and thus the robot's location on the path. This could assist the robot in staying centered on the path.

Encoders

Encoders were intended to be used in the project for two purposes: to track robot position relative to the last location derived from the GPS or camera and provide sensor feedback to the ESCs.

The robot’s position can be tracked with encoders through the use of odometry, utilizing the position feedback of each wheel to determine the robot's change in position. This allows real-time position tracking with the fastest update rate of the implemented sensors. Odometry, however, does not have absolute feedback, only relative change, resulting in the accumulation of error over time resulting in gross inaccuracies. Thus, the encoder position tracking alone is not sufficient in this project.

Another use of the encoder is to provide motor positional feedback to the electronic speed controller (ESC). By providing the ESC with the angle of the motor shaft and stator, the electric pulses through the motor phase wires can be better timed to maximize the efficiency and power of the motor, especially at low speeds. This allows for a significantly higher motor torque when driven at low speeds which are especially important when carefully navigating around people.

The initial encoders we selected utilized an AS5600 microchip which was selected for its low cost. It brought about two issues, however. The first issue was that each of the encoders shared the same I2C address which meant that a 12C breakout board was required to utilize multiple encoders interfacing with the Raspberry Pi. The second issue was that the encoders were not designed for high-speed continuous motors. Their peak operating speed was experimentally discovered to be approximately 4000rpm and the Aerosky motors used on this delivery robot were intended to be run at up to 10000rpm. While this would allow for proper sensoring of high torque and low-speed control, the encoder also only reported readings from 0-360 and the Raspberry Pi is unable to read at a frequency high enough to determine whether a rotation has been completed.

An alternative encoder, AS5048A, was selected due to its increased compatibility with the high motor speeds and VESC integration. It was able to significantly improve the motor startup torque, but due to the difficulty of assembling the magnets into the motor shaft, the encoder was ultimately not implemented into the final assembly.

VESC

The VESC motor controller was connected to the robot's power system using an XT60H connector and to the motor using an MR30 connector. For calibration, the VESC was connected to a Windows laptop, using a micro-USB cable, running the VESC tool. The first calibration step was to update the VESC to the latest publicly available firmware to get the most up-to-date features. Then, the motor was initially calibrated using the auto-calibration features to measure the motor properties, including the resistance and inductance. Safety limits, such as peak current and battery voltage cutoffs were also set before further testing. The primary calibration target was the sensorless startup settings using the HFI mode. Sensorless calibration was used because the required sensor magnets were not implemented into the brushless motor shafts. The motor properties and settings were adjusted continuously until the VESC was able to successfully spin the motor. Some additional tuning was completed to improve the efficiency and acceleration of the system. This process was repeated for each of the VESC, and motor combinations and the motor tuning parameters were saved. The ESCs were tested under a PID speed control loop as well as a duty cycle control.

Compass

To effectively determine which direction the robot must drive to reach its goal, a high-precision compass was selected, the CMPS14 from RobotShop. This compass has an integrated gyro, magnetometer, and accelerometer which are utilized together for a more precise heading. The readings from each of these sensors, raw and corrected, are also made available to the Raspberry Pi for further sensor fusion if desired with future iterations. The compass communicated with the Raspberry Pi through the I2C communication protocol to transmit the current heading, acceleration, angular velocity, and magnetometer readings. To ensure a reliable connection between the compass and the Raspberry Pi, a prototype perf board was used. The perf board housed a row of female headers to allow easy connection and removal of the compass PCB. A 2x6 row of male header pins was also added to the end of a wire length to ensure a rigid connection to the Raspberry Pi GPIO pins.

Software Systems

Operating System

To run ROS on the Raspberry Pi, the processor ran an Ubuntu distribution due to the incompatibility of Raspbian with ROS. To maintain a robust system going forwards and use the most modern technology available, the Ubuntu distribution chosen was 20.04 to run ROS Noetic. While the Ubuntu distribution was initially installed as a server, it was later upgraded to have a desktop environment to easily visualize the lidar data and provide easier troubleshooting.

ROS

This project uses the Robot Operating System (ROS) as the main software framework. ROS is a collection of software libraries and tools for building robot applications and simulations. Due to the running operating system (Ubuntu 20.04), ROS noetic was installed. This version of ROS primarily runs Python 3 over the deprecated Python 2.

OpenCV Computer Vision

Although the hope was to keep the robot centered on the sidewalks using solely the GPS, it was acknowledged that the surrounding buildings and trees may obstruct the GPS that such accuracy may not be possible. In such an event, the robot would be able to stay centered on the sidewalk by detecting the edges of the sidewalk and its own position relative to the edges such that it is able to center itself.

The first attempt to detect the edges of the sidewalk utilized the canny edge detection algorithm with blurring. The blurring was achieved through repeated convolutions with a low-pass filter kernel. The canny filter was successful in detecting edges, but did not isolate the major road edges, but rather identified the edges of every blade of grass. The blurring was successfully able to reduce the number of minor detections, but in doing so, made the primary detections composed of many smaller lines rather than long, continuous lines.

The next attempt at OpenCV utilized the WPI-developed tool GRIP which enables the user to rapidly modify the image processing pipeline to see the results of novel pipelines in real-time. This tool was useful in testing edge detection in a range of pictures, and it was determined that the color-based approach of OpenCV was not well suited for the detection of sidewalk edges, due to the varying colors of the sidewalk, surroundings, and sunlight.

TensorFlow Computer Vision

Due to the difficulty in utilizing OpenCV to detect the edges of the sidewalk, a TensorFlow image processing model was developed to isolate the path, which could then be used with OpenCV to detect the edges and the robot's orientation concerning the path. The training process involved first importing a three-layer (RGB) image of the road environment and a one-layer image (black and white), mask, that isolates the path area. These images were combined into a single TensorFlow object which underwent image segmentation training. Image segmentation is the process of assigning a label to every pixel of an image, rather than the image as a whole, such that the relevant area of the image can be identified and isolated.

The first attempt to create a suitable TensorFlow model was inspired by the image segmentation example in the TensorFlow tutorials [14]. This model assisted in the configuration of TensorFlow on a windows PC and served as an introduction to the TensorFlow pipeline. This tutorial used image segmentation to determine the area of an image containing pixels of a pet, and the type of pet shown. While this tutorial was useful in demonstrating the effective use of image segmentation in TensorFlow, it proved difficult to modify this pipeline for the custom sidewalk dataset.

The next attempt mirrored the image processing pipeline presented by Yann Le Guilly [15]. This process used the ADE20K dataset, a custom dataset provided by MIT that included road segmentation. The custom dataset, while less efficient than the TFRecords demonstrated in the pet segmentation, was easier to replicate and the training process was more closely aligned with our goals. Through the modification of this dataset, a custom model was developed which successfully performed image segmentation to isolate roads from given images.

Campus Map

The first step in planning the robot's trajectory was to generate a precise map of the WPI campus. The map was generated by walking with the C94-M8P GPS along the desired paths of the robot. The expected setup was simulated as much as reasonably possible and carried by one of the team members along the center of the WPI sidewalks while recording the GPS position.

The positions collected during the human walkthrough were connected to form a series of nodes for the robot to travel between during its autonomous navigation.

The following GPS data points were collected and processed to generate the following map.

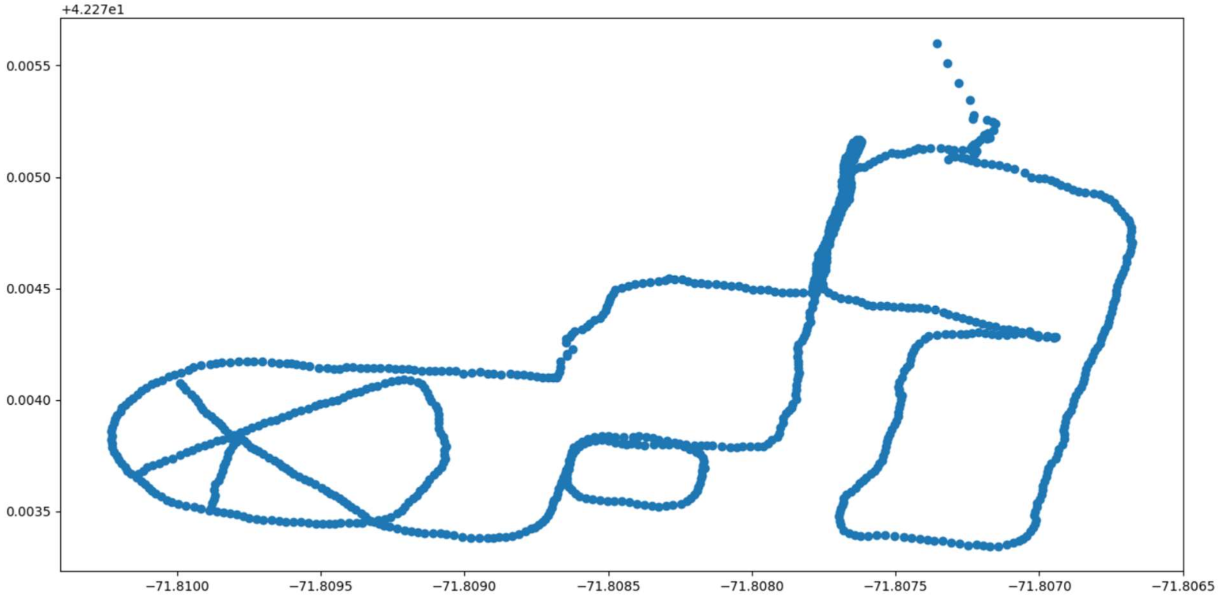


Figure 36 Raw GPS coordinate data collection

The data was processed from the map such that there are no repeated points along each road, and data collected from indoor spaces were removed. From the remaining data, points along the defining features of the map were kept and used to serve as navigation points in a search graph. Edges were constructed manually based on the existing roads on campus. The resulting campus map is shown below.

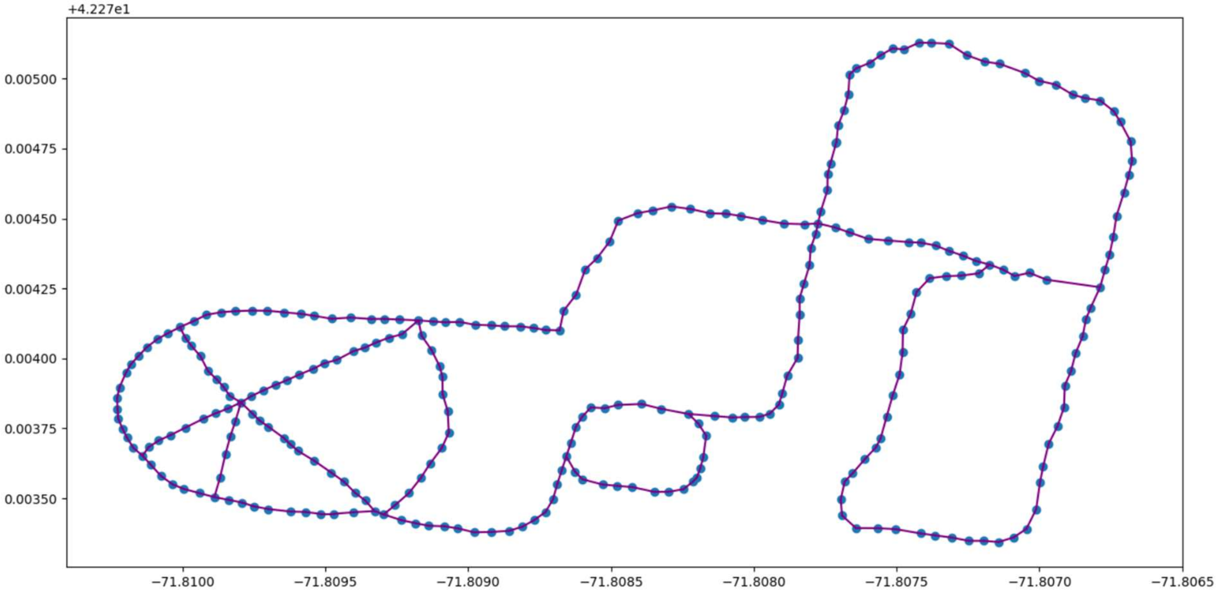


Figure 37 Graph for campus map after data filtering

Because the C94-M8P had to be returned, we reconducted the experiment of walking around the campus using the new NEO-M9N. The results are shown below, where the orange points are collected using the new GPS, and the blue points are collected using the old one. There was significantly more noise with the new GPS data, due to the decreased accuracy of the cheaper GPS compared to the previous one. Due to the consistency of the new data with the old, but higher quality data with the original GPS, the original GPS data was used to construct the map of campus for future navigation.

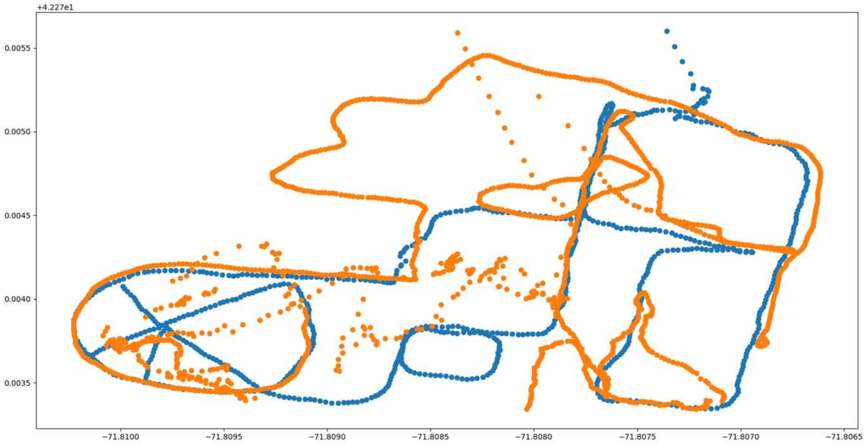


Figure 38 NEO-M9N coordinate data vs. C94-M8P coordinate data

User Interface

A user interface was created for the members of the community to interact with the robot, as shown in *Figure 39.* The interface was designed as an HTML webpage that is connected to WPI Wireless for accessibility. The red circles on the map indicate possible destinations for the robot. When the user selects a destination, the robot will receive the location as its goal. The start and stop buttons control whether the robot drives. The HTML webpage communicates with the ROS network, running on the Raspberry PI, through the ROSBridge library. This communication allows the webpage to send goal positions and drive commands to the ROS programs.

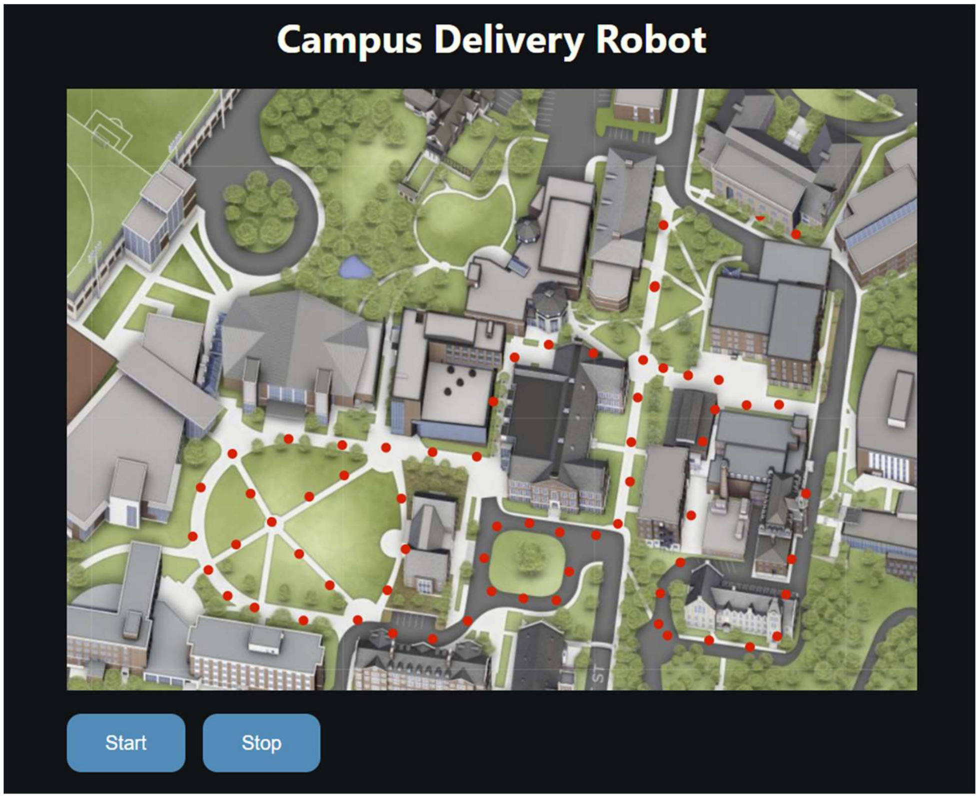


Figure 39 User interface for choosing goal destination

Navigation

After receiving a command to travel to a desired location from the HTML UI, the robot utilized an A* algorithm to find the optimal path from its current location to the destination. The A* search used the node map of collected GPS data points with a distance-based heuristic function to minimize the distance to travel. The next point in the path from the robot's current location was used as an intermediate setpoint for the robot to travel to.

The robot attempted to autonomously navigate to the destination by driving toward each setpoint in the path toward the destination. To navigate to a setpoint, the robot utilized a point-and-shoot driving strategy, in which the robot determined the angle between itself (using the compass) and the setpoint (using the GPS). If the angle was within a set tolerance (10 degrees), the robot would drive straight. If the angle was outside of this tolerance, the robot would turn such that it pointed in the direction of the setpoint. Every time the GPS transmitted an updated robot location, the A* algorithm was repeated such that a new path was determined to correct for any erroneous and optimize the path if a new route becomes preferable. This process was repeated until the robot was within a given tolerance (3m) of the destination.

Internet Connectivity

This robot is continuously connected to the internet through the use of the WPI Wireless Network. The internet connection enables the robot to receive a travel destination and could be used for future authentication and RTK data processing. The network was configured through the use of WPI certificates and authentications as described on the WPI Hub [16].

Collision Avoidance

The robot avoids collisions with obstacles on the given path through the use of a lidar. The lidar is used to scan the surrounding area and determine the location of obstacles on all sides of the robot. While ROS supports the mapping of the environment through Lidar data, this robot does not create and update the map due to the dynamic nature of a sidewalk, with constantly moving people. Instead, the polar readings from the lidar are converted to cartesian coordinates and analyzed such that if any obstacles appear in the immediate path of the robot, the robot will stop moving and wait for the obstacle to move out of the path such that the robot can safely continue forwards.

Discussion & Results

Mechanical Evaluation

FEA

To determine the theoretical strength of the chassis and wheel assemblies before manufacturing and without destructive testing, Finite Element Analysis (FEA) studies were simulated in Fusion 360. Forces were applied to the various regions of the top of the chassis to simulate various payloads. The wheels also had radial and tangential loads applied to simulate a payload and driving friction.

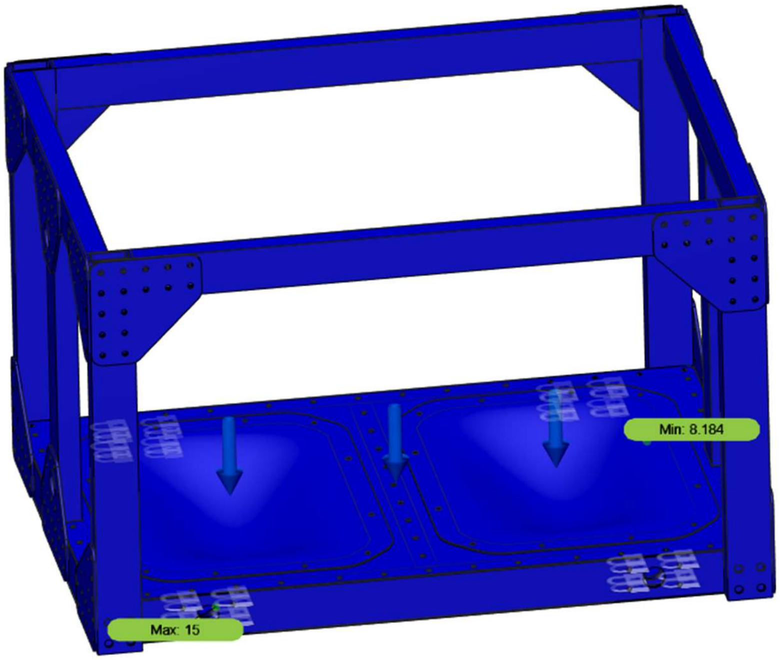


Figure 40 Safety factor on frame with 500lb load evenly distributed

To simulate an evenly distributed load over the entirety of the chassis, a load of 500lb was applied across the top frame and plates. The wheel mounting holes were fixed to simulate the resulting normal force of the ground on the wheels. This simulation shows that the aluminum plates were weaker than the aluminum tubing and that an approximate theoretical maximum load of 25,000lb could be applied before the frame mechanically failed. The simulation result is shown in *figure 40*.

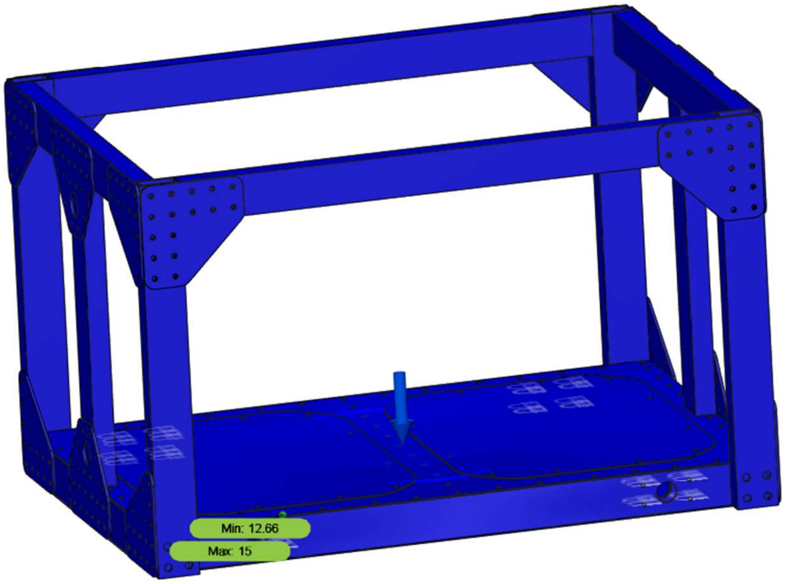


Figure 41 Safety factor on frame with 500lb load distributed over center of frame

In the event that the package is rigid and unable to deform to match the deformation of the top plates, the load would be carried primarily by the center support beam. A load on the center support of 500lb is applied to the center rail and the system is fixed in the same manner as in the previous study. The resulting minimum factor of safety is 12 (shown in *Figure 41*), greater than the fully distributed force because the plates are not bearing the payload weight.

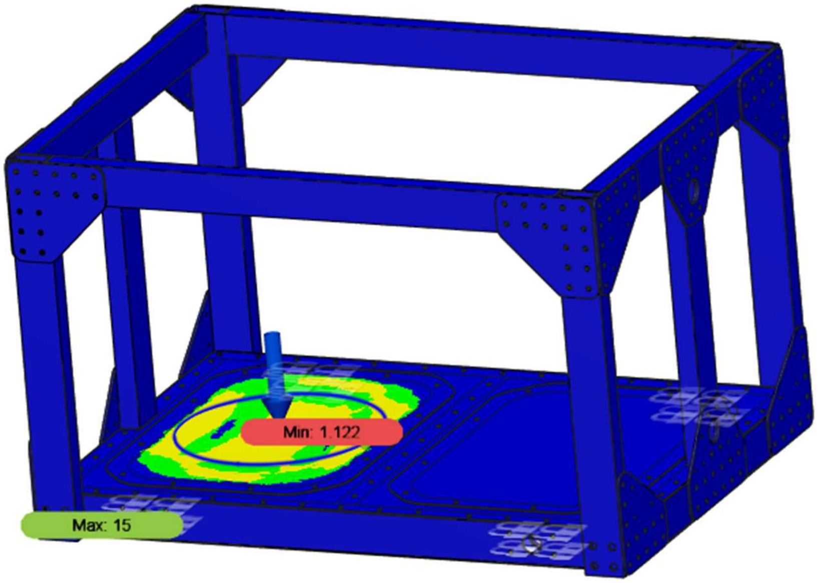


Figure 42 Safety factor on frame with 500lb load distributed over center of panel

In a worst-case scenario, the full payload could be applied to a single top plate. This scenario is simulated in *Figure 42* with the same constraints as the previous simulations and with an applied load of 500lb. This load case provides the lowest factor of safety (1.122) but still suffices to bear a load of 500lb on a single top plate.

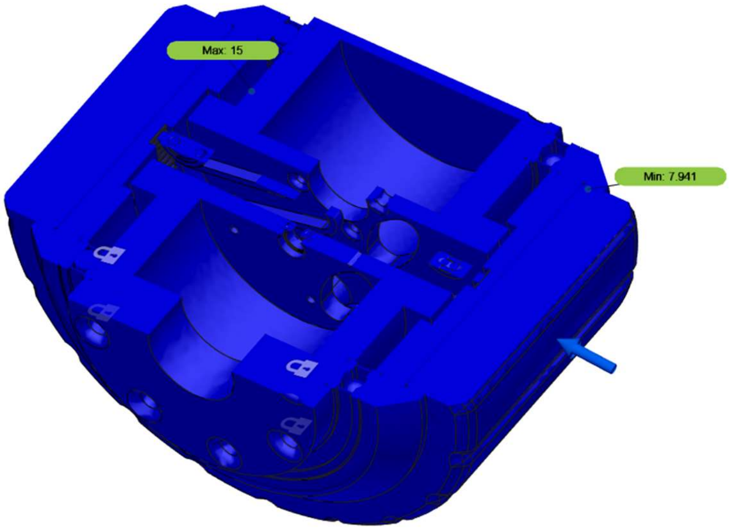


Figure 43 Safety factor on wheel assembly with 100lb load distributed over wheel side

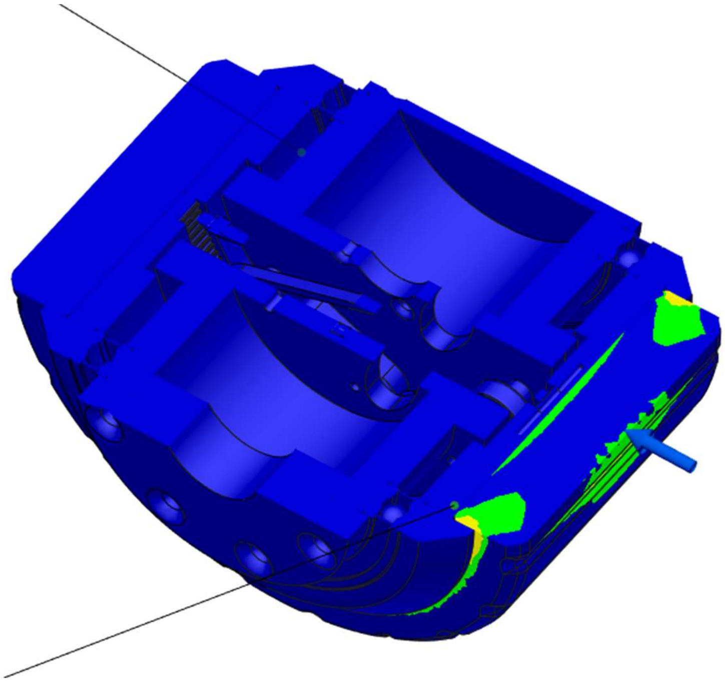


Figure 44 Exaggerated radial safety factor on wheel assembly

The first wheel simulation modeled a heavy payload, as modeled in the previous chassis simulations. The wheel was fixed on the mounting holes and a 100lb force was applied radially to the wheel, simulating the ground normal force from payload weight, shown in *Figure 43*. The minimum safety factor for this 100lb load was nearly 8, implying that up to 800lb of weight per wheel (over 2000lb total package weight) could be carried before the weight of the package crushed the wheel assembly. *Figure 44* exaggerates the applied load to highlight the local areas of deformation and potential failure. The first component to theoretically fail was shown to be the tire, given a heavy payload.

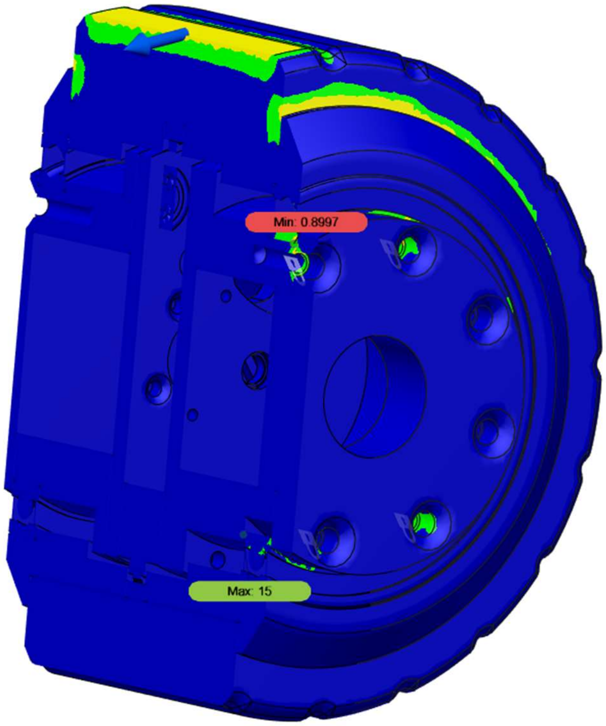


Figure 45 Exaggerated tangential safety factor on wheel assembly

An excessively great tangential force (1000lb) was applied tangentially to the tire to determine which structural component of the wheel was weakest during robot acceleration. The weakest location, as shown in *Figure 45*, was the mounting hole to the chassis and the only other notable weak areas are the outside of the tire, as would be expected to wear under normal operation.

Physical Testing

During full system testing with the final robot, several issues arose with the drivetrain system. When the transmission system experienced load, several belt box systems experienced failures. The first issues occurred during wheel assembly. Some assemblies had significantly more assembly resistance and some wheels had significant friction when rotated. Multiple modifications were made to the components to reduce friction, including enlarging slip fit bores for bearings, testing several types of grease on the belts, and replacing the bearing grease with an oil-based lubricant. Although these modifications did reduce friction, the friction between assemblies remained inconsistent. These inconsistencies may have been due to manufacturing tolerances. During testing, the critical failure sustained by multiple wheel assemblies was the breakage of the timing belt after a few minutes of loaded operation.

The most significant source of friction in the belt box reduction was between the neoprene belt and acetal sliders. The first lubricant tested was a 50/50 mixture of dish soap and water, inspired by the process which some divers use to slide into their wetsuits. Despite an initial success, as the solution dried, the belt became stickier, likely due to the dish soap removing oils from the belt. The next lubricant tested was general assembly lube with a low viscosity. This lubricant suffered a similar shortcoming to the soapy water mixture: as the lubricant dried, it became less effective and caused the system to have high friction again. The best lubricant tested was Loctite 8042, a white lithium grease. This lubricant was more viscous than the other lubricants listed, resulting in a longer functional time before drying out or exiting the hub.

Unfortunately, additional funding for this project was only obtained during the second semester of the project, making it difficult to improve and manufacture new components after failures were observed. A tolerance analysis of the wheel transmission system could have improved the fit of system components.

Several mechanical challenges could have been avoided with larger pitch belt. The initially small belt pitch was selected to reduce the size of the COTS pulley, achieving a higher reduction within the same wheel diameter. However, a larger pitch belt would have been stronger and less prone to stripping, making it a better choice for this application.

ERPM	RPM	Current (A)	Input Power (W)	Notes
4000	364	4	4	Minimum functional speed
20000	1818	5.5	5.5	
50000	4545	-	-	
60000	5454	-	-	Belt broke after short period

Table 2 Free spin speed measurements

Wheel/scale slip angle (deg)	Wheel/scale coefficient of friction	Measured scale mass (kg)	Measured scale force (N)	Frictional Force (N)	Wheel radius (m)	Frictional torque (Nm)
25	0.45	11.5	113	51	0.07	3.57

Table 3 Kinetic torque testing

Arm radius (m)	Measured scale mass (kg)	Measured scale force (N)	Torque (Nm)	Current (A)
0.48	0.5	4.9	2.4	13

Table 4 Static torque testing

In order to determine if the wheels were suitable for this robot, they underwent testing until failure. The free spin of each wheel was recorded in *Table 2* using PID speed control. While the wheels were tested at speeds of 50,000 and 60,000 ERPM, the belt snapped before the current and input power could be recorded. The similar current draw between 4,000 ERPM and 20,000 ERPM highlights the inefficiency of brushless motor at low speeds.

The kinetic wheel torque was measured through an external frictional force (shown in *Table 3*). A scale was pressed into the spinning wheel until the scale was overloaded at 11.5kg, at which point the wheel was significantly slowed but not fully stopped. By using the coefficient of friction between the scale and wheel, the frictional force exerted at scale overloading was 3.57 Nm. This provides a lower bound for the wheel torque output and the actual peak torque output is greater than 3.57 Nm by an unknown amount.

Startup torque, while significantly lower for brushless motors than brushed, is a significant factor in the capabilities of the wheel. For example, while the continuous torque may be sufficient to drive up a hill, if the robot stops on the hill and the startup torque is insufficient to start driving up the hill again, then the hill is not safely drivable by the robot. This startup torque was measured in *Table 4* and was measured as the point in which the belt slipped, and the startup torque of the motor was able to slip the belt. The 2.4 Nm of stall torque enabled by a single wheel is sufficient to drive up a 10-degree incline with a maximum of 20 kg of load supported by each wheel (*Equation 1*), allowing for a maximum weight capacity of 80 kg (175 lb) on the WPI campus while maintaining full mobility.

Large variations and inconsistencies between the performance of each wheel (both relative to itself through testing and after reassembly, and to other wheels), made testing difficult and the results inconclusive. Belt boxes would frequently partially lock up, requiring a high torque to rotate the wheel through some portions of travel. Occasionally this could be fixed by manually rotating the wheel and causing the belt to fall back into alignment. The setscrews on the motor can and belt pinion also frequently came loose, allowing the motor to free spin without rotation of the wheel assembly. Additionally, after extensive testing, the belt was prone to snapping. Removal of the outermost bearing or cap allowed the assembly to spin more freely (approximately half the current required at an equivalent speed), but this allowed the ring pulley to slide laterally and disengage the belt. The torque to slip the belt on the pinion also greatly varied. During some tests, the pinion would slip at 1 Nm, while remaining engaged up to 2.4 Nm during other stall tests. Some kinetic tests would slip the pulley at an equivalent torque and current (13A) while other tests allowed the current to increase to 23 Nm before slipping occurred. Slipping at high speeds was shown to be more problematic than at low speeds as a high pinion rotation speed would cause the belt to teeth to be shredded and the belt to break more quickly.

Electronic Evaluation

Central Processor

A Raspberry Pi 4 served as the central processor for this robot. The robot uses the full capabilities of this processor, running multiple demanding applications, including ROS and TensorFlow. From our initial findings, the Raspberry Pi 4 appears powerful enough to run ROS locally without significant performance losses, but the performance of TensorFlow running on the pi was not evaluated in this project. Excessive heat from the processor was successfully dissipated through the use of adhesive heat sinks, although the placement of the heatsinks could have been more aesthetically aligned. *Figure 46* shows the heat sinks on the Raspberry Pi 4.

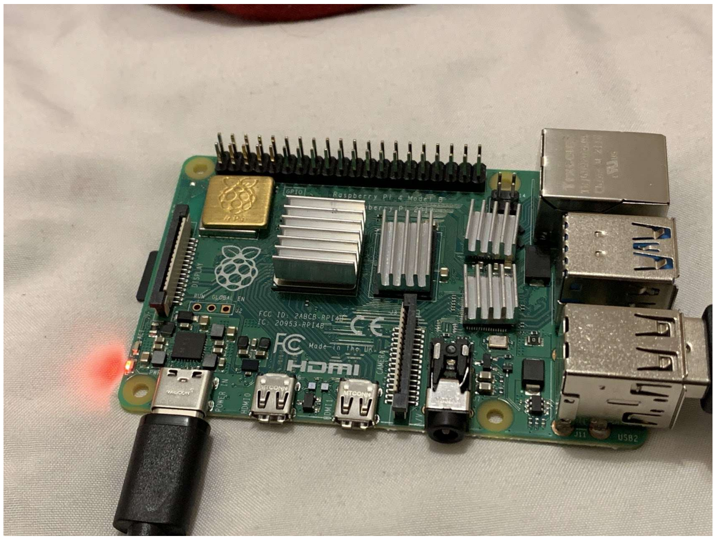
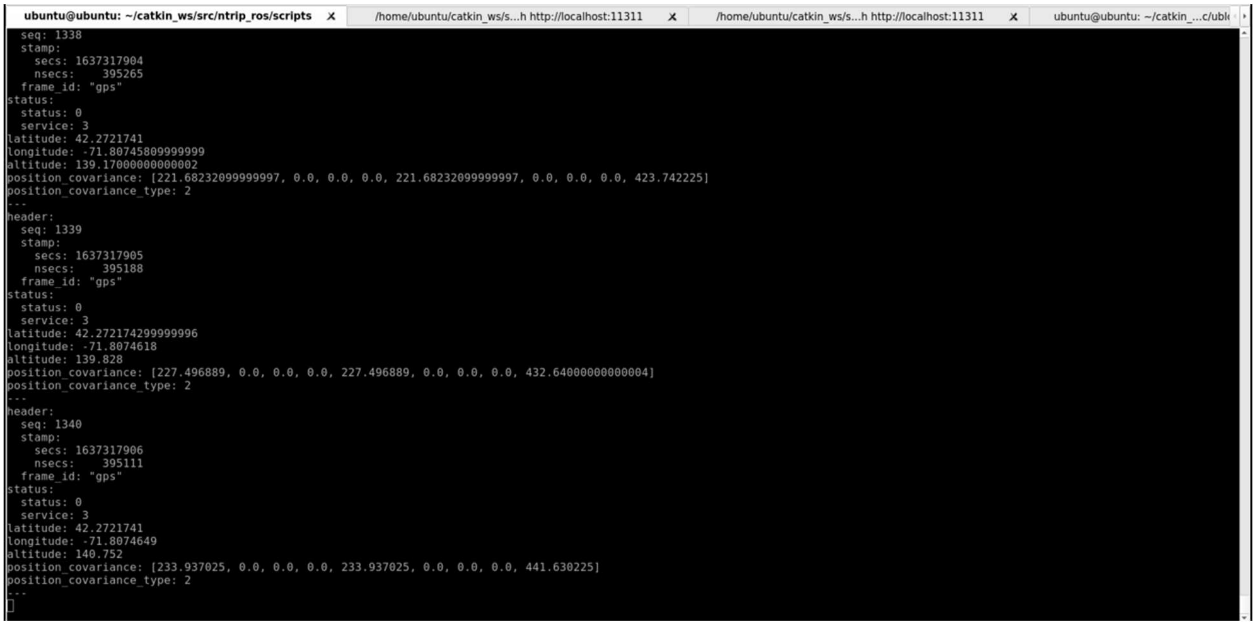


Figure 46 Raspberry Pi with heatsinks

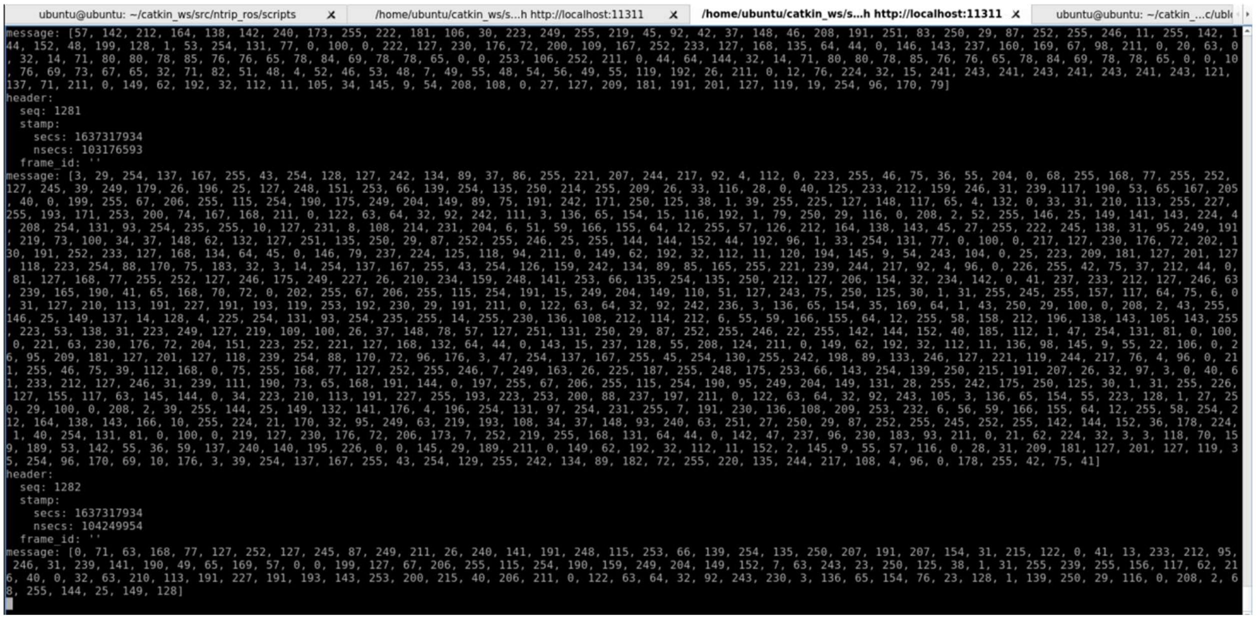
Mapping

The C94-M8P has RTK integration features to achieve higher precision for localization. The device was successfully able to receive a GNSS fix, as shown in *Figure 47*, and RTCM messages from the Northborough MaCORS station, as shown in *Figure 48*. However, the GPS was unable to combine the two data sources into a more precise reading, so failed to achieve an RTK fix.


ubuntu@ubuntu: ~/catkin_ws/src/ntrip_ros/scripts
/home/ubuntu/catkin_ws/s...h http://localhost:11311
/home/ubuntu/catkin_ws/s...h http://localhost:11311
ubuntu@ubuntu: ~/catkin_...c/ubl

```
  seq: 1338
  stamp:
    secs: 1637317904
    nsecs:    395265
  frame_id: "gps"
status:
  status: 0
  service: 3
latitude: 42.2721741
longitude: -71.80745809999999
altitude: 139.17000000000002
position_covariance: [221.68232099999997, 0.0, 0.0, 0.0, 221.68232099999997, 0.0, 0.0, 0.0, 423.742225]
position_covariance_type: 2
---
header:
  seq: 1339
  stamp:
    secs: 1637317905
    nsecs:    395188
  frame_id: "gps"
status:
  status: 0
  service: 3
latitude: 42.272174299999996
longitude: -71.8074618
altitude: 139.828
position_covariance: [227.496889, 0.0, 0.0, 0.0, 227.496889, 0.0, 0.0, 0.0, 432.64000000000004]
position_covariance_type: 2
---
header:
  seq: 1340
  stamp:
    secs: 1637317906
    nsecs:    395111
  frame_id: "gps"
status:
  status: 0
  service: 3
latitude: 42.2721741
longitude: -71.8074649
altitude: 140.752
position_covariance: [233.937025, 0.0, 0.0, 0.0, 233.937025, 0.0, 0.0, 0.0, 441.630225]
position_covariance_type: 2
---
```

Figure 47 C94-M8P coordinate data stream with GNSS fix


ubuntu@ubuntu: ~/catkin_ws/src/ntrip_ros/scripts
/home/ubuntu/catkin_ws/s...h http://localhost:11311
/home/ubuntu/catkin_ws/s...h http://localhost:11311
ubuntu@ubuntu: ~/catkin_...c/ubl

```
message: [57, 142, 212, 164, 138, 142, 240, 173, 255, 222, 181, 106, 30, 223, 249, 255, 219, 45, 92, 42, 37, 148, 46, 208, 191, 251, 83, 250, 29, 87, 252, 255, 246, 11, 255, 142, 1
44, 152, 48, 199, 128, 1, 53, 254, 131, 77, 0, 100, 0, 222, 127, 230, 176, 72, 200, 109, 167, 252, 233, 127, 168, 135, 64, 44, 0, 146, 143, 237, 160, 169, 67, 98, 211, 0, 20, 63, 0
, 32, 14, 71, 80, 80, 78, 85, 76, 76, 65, 78, 84, 69, 78, 78, 65, 0, 0, 253, 106, 252, 211, 0, 44, 64, 144, 32, 14, 71, 80, 80, 78, 85, 76, 76, 65, 78, 84, 69, 78, 78, 65, 0, 0, 10
, 76, 69, 73, 67, 65, 32, 71, 82, 51, 48, 4, 52, 46, 53, 48, 7, 49, 55, 48, 54, 56, 49, 55, 119, 192, 26, 211, 0, 12, 76, 224, 32, 15, 241, 243, 241, 243, 241, 243, 241, 243, 121,
137, 71, 211, 0, 149, 62, 192, 32, 112, 11, 105, 34, 145, 9, 54, 208, 108, 0, 27, 127, 209, 181, 191, 201, 127, 119, 19, 254, 96, 170, 79]
header:
  seq: 1281
  stamp:
    secs: 1637317934
    nsecs: 103176593
  frame_id: ''
message: [3, 29, 254, 137, 167, 255, 43, 254, 128, 127, 242, 134, 89, 37, 86, 255, 221, 207, 244, 217, 92, 4, 112, 0, 223, 255, 46, 75, 36, 55, 204, 0, 68, 255, 168, 77, 255, 252,
127, 245, 39, 249, 179, 26, 196, 25, 127, 248, 151, 253, 66, 139, 254, 135, 250, 214, 255, 209, 26, 33, 116, 28, 0, 40, 125, 233, 212, 159, 246, 31, 239, 117, 190, 53, 65, 167, 205
, 40, 0, 199, 255, 67, 206, 255, 115, 254, 190, 175, 249, 204, 149, 89, 75, 191, 242, 171, 250, 125, 38, 1, 39, 255, 225, 127, 148, 117, 65, 4, 132, 0, 33, 31, 210, 113, 255, 227,
255, 193, 171, 253, 200, 74, 167, 168, 211, 0, 122, 63, 64, 32, 92, 242, 111, 3, 136, 65, 154, 15, 116, 192, 1, 79, 250, 29, 116, 0, 208, 2, 52, 255, 146, 25, 149, 141, 143, 224, 4
, 208, 254, 131, 93, 254, 235, 255, 10, 127, 231, 8, 108, 214, 231, 204, 6, 51, 59, 166, 155, 64, 12, 255, 57, 126, 212, 164, 138, 143, 45, 27, 255, 222, 245, 138, 31, 95, 249, 191
, 219, 73, 100, 34, 37, 148, 62, 132, 127, 251, 135, 250, 29, 87, 252, 255, 246, 25, 255, 144, 144, 152, 44, 192, 96, 1, 33, 254, 131, 77, 0, 100, 0, 217, 127, 230, 176, 72, 202, 1
30, 191, 252, 233, 127, 168, 134, 64, 45, 0, 146, 79, 237, 224, 125, 118, 94, 211, 0, 149, 62, 192, 32, 112, 11, 120, 194, 145, 9, 54, 243, 104, 0, 25, 223, 209, 181, 127, 201, 127
, 118, 223, 254, 88, 170, 75, 183, 32, 3, 14, 254, 137, 167, 255, 43, 254, 126, 159, 242, 134, 89, 85, 165, 255, 221, 239, 244, 217, 92, 4, 96, 0, 226, 255, 42, 75, 37, 212, 44, 0,
 81, 127, 168, 77, 255, 252, 127, 246, 175, 249, 227, 26, 210, 234, 159, 248, 141, 253, 66, 135, 254, 135, 250, 212, 127, 206, 154, 32, 234, 142, 0, 41, 237, 233, 212, 127, 246, 63
, 239, 165, 190, 41, 65, 168, 70, 72, 0, 202, 255, 67, 206, 255, 115, 254, 191, 15, 249, 204, 149, 110, 51, 127, 243, 75, 250, 125, 30, 1, 31, 255, 245, 255, 157, 117, 64, 75, 6, 0
, 31, 127, 210, 113, 191, 227, 191, 193, 119, 253, 192, 230, 29, 191, 211, 0, 122, 63, 64, 32, 92, 242, 236, 3, 136, 65, 154, 35, 169, 64, 1, 43, 250, 29, 100, 0, 208, 2, 43, 255,
146, 25, 149, 137, 14, 128, 4, 225, 254, 131, 93, 254, 235, 255, 14, 255, 230, 136, 108, 212, 114, 212, 6, 55, 59, 166, 155, 64, 12, 255, 58, 158, 212, 196, 138, 143, 105, 143, 255
, 223, 53, 138, 31, 223, 249, 127, 219, 109, 100, 26, 37, 148, 78, 57, 127, 251, 131, 250, 29, 87, 252, 255, 246, 22, 255, 142, 144, 152, 40, 185, 112, 1, 47, 254, 131, 81, 0, 100,
 0, 221, 63, 230, 176, 72, 204, 151, 223, 252, 221, 127, 168, 132, 64, 44, 0, 143, 15, 237, 128, 55, 208, 124, 211, 0, 149, 62, 192, 32, 112, 11, 136, 98, 145, 9, 55, 22, 106, 0, 2
6, 95, 209, 181, 127, 201, 127, 118, 239, 254, 88, 170, 72, 96, 176, 3, 47, 254, 137, 167, 255, 45, 254, 130, 255, 242, 198, 89, 133, 246, 127, 221, 119, 244, 217, 76, 4, 96, 0, 21
1, 255, 46, 75, 39, 112, 168, 0, 75, 255, 168, 77, 127, 252, 255, 246, 7, 249, 163, 26, 225, 187, 255, 248, 175, 253, 66, 143, 254, 139, 250, 215, 191, 207, 26, 32, 97, 3, 0, 40, 6
1, 233, 212, 127, 246, 31, 239, 111, 190, 73, 65, 168, 191, 144, 0, 197, 255, 67, 206, 255, 115, 254, 190, 95, 249, 204, 149, 131, 28, 255, 242, 175, 250, 125, 30, 1, 31, 255, 226,
 127, 155, 117, 63, 145, 144, 0, 34, 223, 210, 113, 191, 227, 255, 193, 223, 253, 200, 88, 237, 197, 211, 0, 122, 63, 64, 32, 92, 243, 105, 3, 136, 65, 154, 55, 223, 128, 1, 27, 25
0, 29, 100, 0, 208, 2, 39, 255, 144, 25, 149, 132, 141, 176, 4, 196, 254, 131, 97, 254, 231, 255, 7, 191, 230, 136, 108, 209, 253, 232, 6, 56, 59, 166, 155, 64, 12, 255, 58, 254, 2
12, 164, 138, 143, 166, 10, 255, 224, 21, 170, 32, 95, 249, 63, 219, 193, 108, 34, 37, 148, 93, 240, 63, 251, 27, 250, 29, 87, 252, 255, 245, 252, 255, 142, 144, 152, 36, 178, 224,
 1, 40, 254, 131, 81, 0, 100, 0, 219, 127, 230, 176, 72, 206, 173, 7, 252, 219, 255, 168, 131, 64, 44, 0, 142, 47, 237, 96, 230, 183, 93, 211, 0, 21, 62, 224, 32, 3, 3, 118, 70, 15
9, 189, 53, 142, 55, 36, 59, 137, 240, 140, 195, 226, 0, 0, 145, 29, 189, 211, 0, 149, 62, 192, 32, 112, 11, 152, 2, 145, 9, 55, 57, 116, 0, 28, 31, 209, 181, 127, 201, 127, 119, 3
5, 254, 96, 170, 69, 10, 176, 3, 39, 254, 137, 167, 255, 43, 254, 129, 255, 242, 134, 89, 182, 72, 255, 220, 135, 244, 217, 108, 4, 96, 0, 178, 255, 42, 75, 41]
header:
  seq: 1282
  stamp:
    secs: 1637317934
    nsecs: 104249954
  frame_id: ''
message: [0, 71, 63, 168, 77, 127, 252, 127, 245, 87, 249, 211, 26, 240, 141, 191, 248, 115, 253, 66, 139, 254, 135, 250, 207, 191, 207, 154, 31, 215, 122, 0, 41, 13, 233, 212, 95,
 246, 31, 239, 141, 190, 49, 65, 169, 57, 0, 0, 199, 127, 67, 206, 255, 115, 254, 190, 159, 249, 204, 149, 152, 7, 63, 243, 23, 250, 125, 38, 1, 31, 255, 239, 255, 156, 117, 62, 21
6, 40, 0, 32, 63, 210, 113, 191, 227, 191, 193, 143, 253, 200, 215, 40, 206, 211, 0, 122, 63, 64, 32, 92, 243, 230, 3, 136, 65, 154, 76, 23, 128, 1, 139, 250, 29, 116, 0, 208, 2, 6
8, 255, 144, 25, 149, 128]
```

Figure 48 C94-M8P RTCM messages for RTK data

Nevertheless, the precision of the C94-M8P was tested by leaving it stationary outdoors, without RTK input data, and the coordinate readings are shown in *Figure 48*. The results of this experiment demonstrate that the standard deviation for the latitude and longitude of the GPS, while stationary, is $4.03 \cdot 10^{-5}$ degrees and $2.97 \cdot 10^{-5}$ degrees, respectively. Through converting the degree precision into meters, the precision was found to be approximately 4 meters latitudinally and 3 meters longitudinally. The data summary for latitude and longitude readings from the GPS is shown in *Figure 50*.

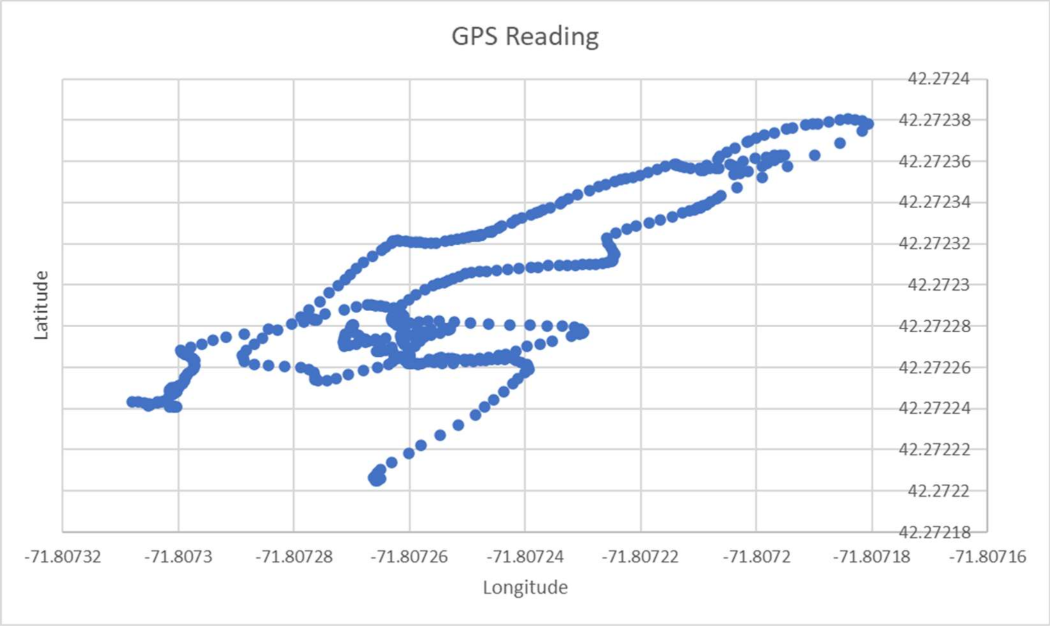
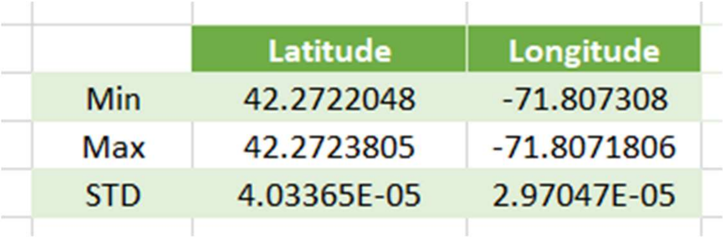


Figure 49 GPS coordinate reading while stationary in outdoor setting



	Latitude	Longitude
Min	42.2722048	-71.807308
Max	42.2723805	-71.8071806
STD	4.03365E-05	2.97047E-05

Figure 50 Data summary for latitude and longitude readings from GPS

The C94 GPS, while unsuccessful in obtaining an RTK fix, was able to maintain sufficient accuracy and precision to create a navigable map of the WPI campus. The collected data points were overlaid onto a satellite map and visually corresponded with the intended campus paths (*Figure 51*). There was some deviation between the data points and campus paths resulting from the locational tolerance of the GPS. These points, which did not lie on the intended path, had the potential to direct the robot to drive off the sidewalk or into a curb. The GPS update frequency was sufficient for manual data collection while walking at a brisk pace and slow autonomous travel. The positional accuracy was sufficient to tell if the GPS was on or off of the sidewalk if the baseline data point was centered on the sidewalk.

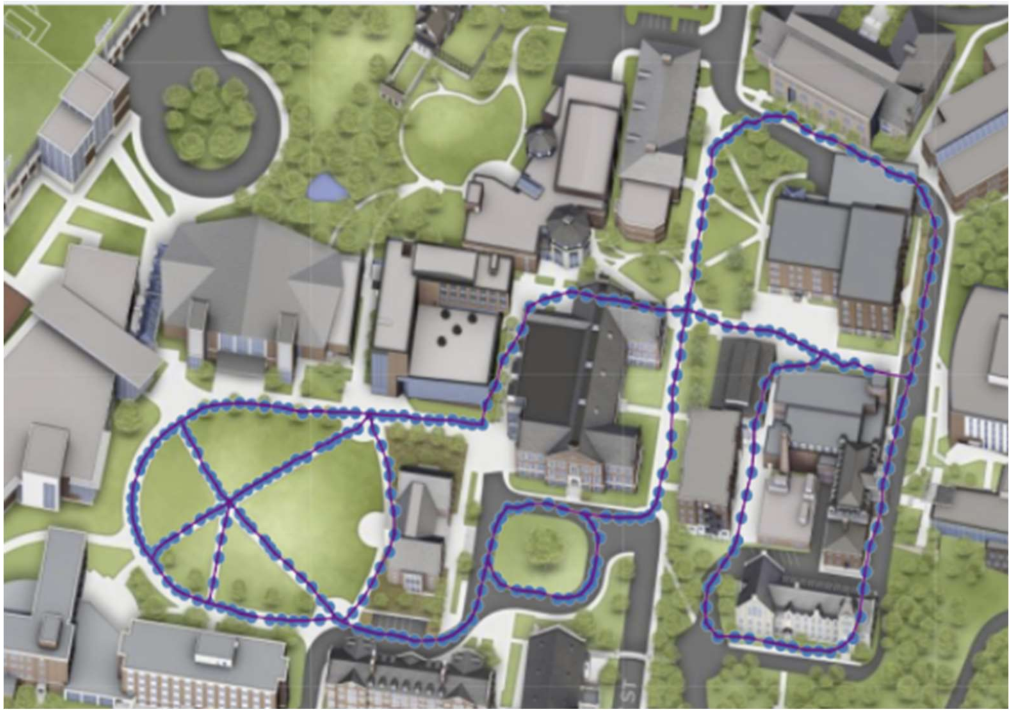
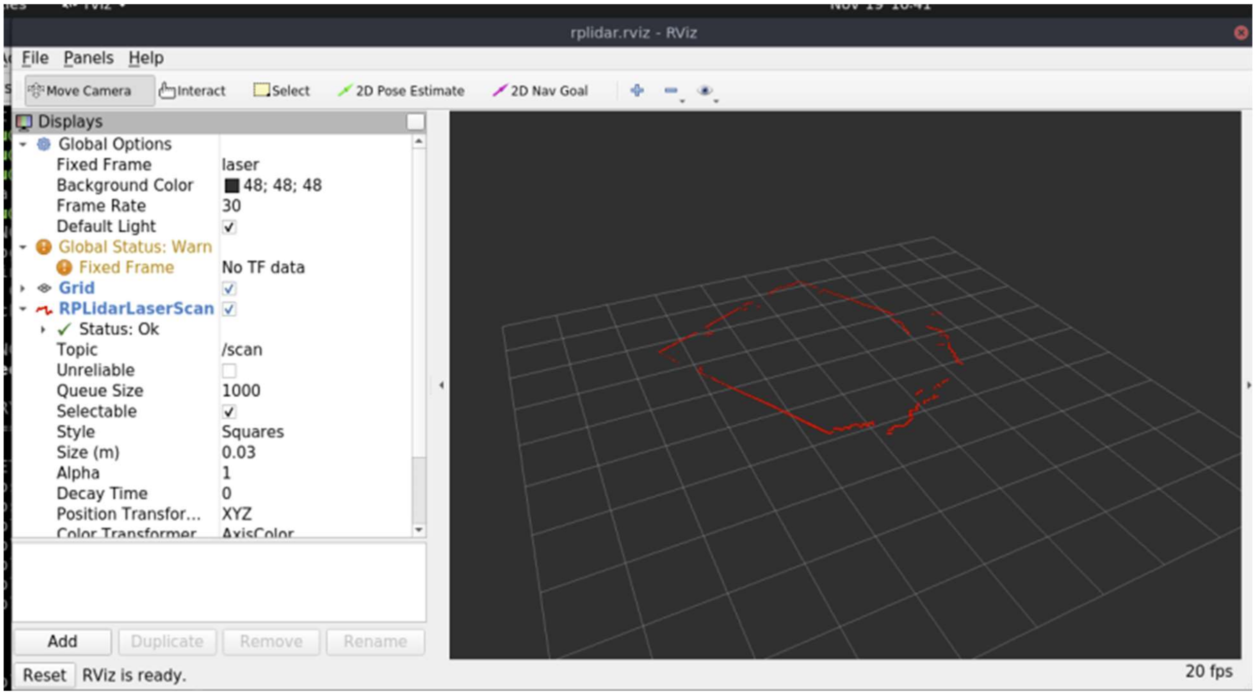


Figure 51 GPS data overlaid on campus map

Lidar

The verification for a functional lidar was conducted by integrating it into ROS and running a launch file that reads its laser data. The laser data was visualized through the ROS visualization tool, Rviz. As shown in *Figure 52*, the lidar was able to fully map the testing room and update its readings at a rate of 10Hz.



Global Options	
Fixed Frame	laser
Background Color	■ 48; 48; 48
Frame Rate	30
Default Light	✓
Global Status: Warn	
Fixed Frame	No TF data
Grid	✓
RPLidarLaserScan	✓
✓ Status: Ok	
Topic	/scan
Unreliable	□
Queue Size	1000
Selectable	✓
Style	Squares
Size (m)	0.03
Alpha	1
Decay Time	0
Position Transfor...	XYZ
Color Transformer	AxisColor

RViz is ready.

20 fps

Figure 52 Lidar scan data visualized in Rviz

VESC

Although the VESC did not receive magnetic encoder feedback from the motor, it successfully spun the motor. The motor startup sequence was inefficient due to the lack of encoders, sometimes drawing excessive current and causing a current fault that would prevent the motor from spinning until a 0-speed command was sent to reset the system. The motor acceleration was also heavily limited. The cause of this acceleration limit was not determined within the allocated time. This slow acceleration caused the robot to occasionally overturn as the motors would not decelerate quickly enough after the intended speed was reached. The VESCs ran the motors with relatively high torque at high speeds, but the motor torque was severely limited at low speeds. The minimum functional duty cycle on the belt box reduction was approximately 20% while the minimum duty cycle on the Mantella test platform was 8%. The PID speed control, as compared to duty cycle, drove the robot in a straighter line because it was able to correct for varying resistances in each wheel system, but this came at a great cost to efficiency and minimum motor speed. The speed control architecture was better suited for the belt box transmission system where high speeds were permissible and there were large variations in wheel efficiency. The duty cycle architecture was required for the Mantella platform because of the low speed and smooth driving that duty cycle control permitted.

Software Evaluation

User Interface

The user interface could successfully send destination messages to the ROS nodes running on the Raspberry Pi, through the WPI Wireless Network. Furthermore, the user interface was able to send start or stop messages to the robot. The response time from the robot after a request was made through the user interface was instant. The communication reliability between the user interface and the Raspberry Pi was dependent on the WPI Wireless internet connectivity. As a result, on-campus areas where the internet connectivity was poor, such as in the middle of the Quad, the messages sent from the user interface would get lost, causing the robot to not respond accordingly.

The map portraying the WPI campus used in the user interface is an artist's rendition, so the streets do not match exactly the collected GPS coordinates.

Navigation

The A* algorithm was chosen as the main navigation method for its optimality, efficiency, and heuristic approach. A preliminary implementation was written in JavaScript to visualize the algorithm’s behavior. Sample nodes and edges were added for easy testing before data filtration was complete. *Figure 53* showcases the successful implementation of A* finding the shortest path between two chosen points on the campus map.

Prior to full assembly and testing on a mobile robot base, the integration of the GPS and compass were tested on a handheld mobile testbench. The angle between the next setpoint and the current reading from the compass was used to determine how much the system must turn to face toward the setpoint and this information was relayed to the operator through the use of a rotating arrow on a micro monitor.

The navigation system was also tested on the small combat robot chassis, Mantella, prior to the final assembly of the delivery robot base. This experimentation also tested the implementation of the motor control software. Mantella was able to successfully navigate between two locations on campus, and initial testing included navigation from the Fountain to the Quad and from Washburn Shops to Atwater Kent. However, since the Mantella chassis is low to the ground, it had difficulty traveling across uneven surfaces, needing assistance when getting stuck in crevices or on debris.

These tests successfully guided an operator and Mantella along the WPI sidewalks, but occasionally directed the system into a wall or curb, especially around sharp turns. These turns were improved by adding more data points to the node map. Persistent angular errors in the measured heading were corrected by adding an angular offset to the compass such that the front of the testbench was aligned with the 0 angle of the adjusted compass measurements. Some data points, such as one in front of the Washburn Shops, were found to stray too far from the path center and must be remeasured for improved navigation.

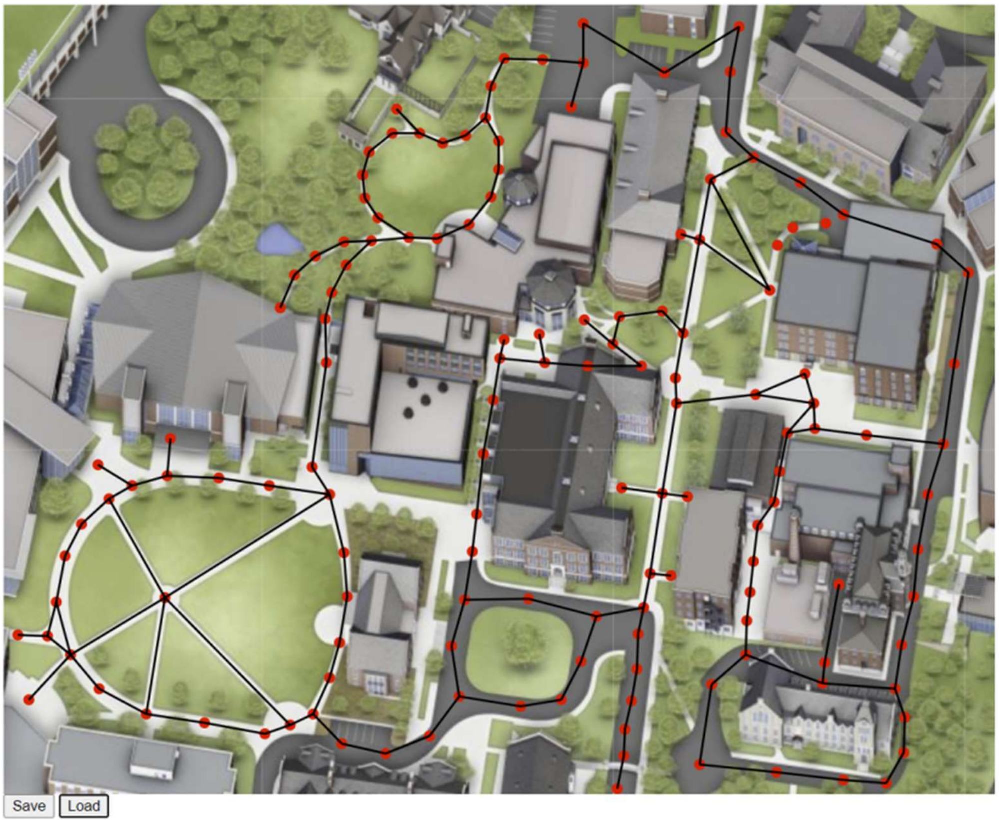


Figure 53 Sample campus map for A implementation*

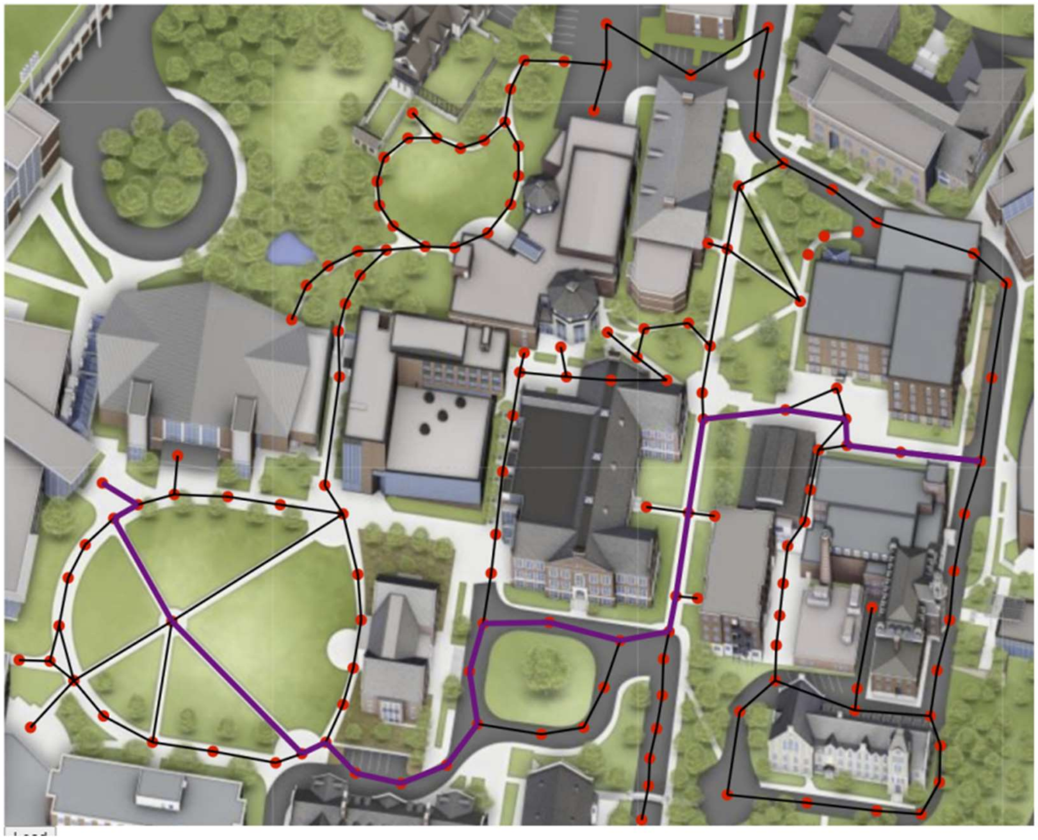


*Figure 54 Shortest path between two points, returned by A**

Computer Vision

While difficult to train, the TensorFlow model has proved very efficient, performing image segmentation to isolate the street in approximately 0.03 seconds. Using the SparseCategoricalCrossentropy loss function from Keras, the segmentation loss was reduced to 0.023, a reasonable accuracy for this process. The comparison between a provided mask and the predicted mask can be seen in *Figure 55*and the training loss over time can be seen in *Figure 56*.

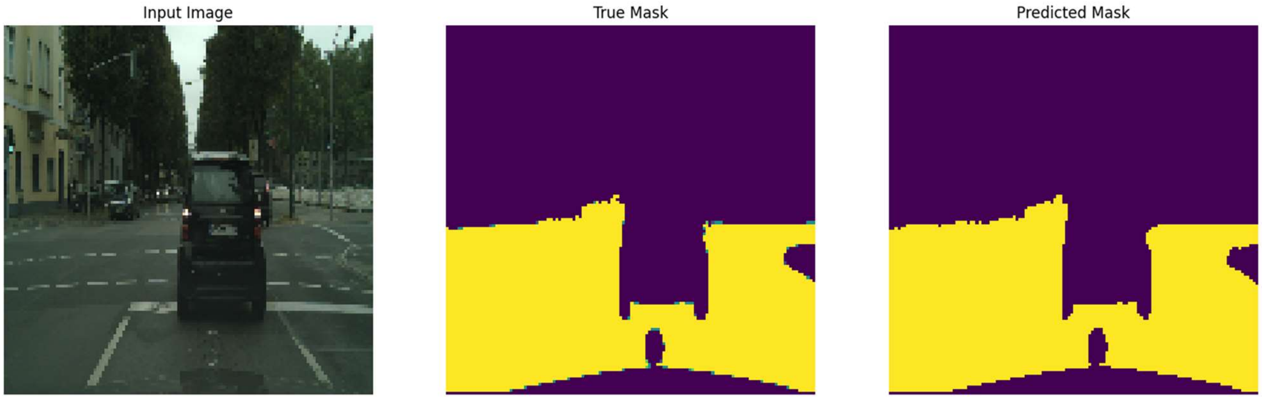


Figure 55 Comparison between input image, true mask, and predicted mask

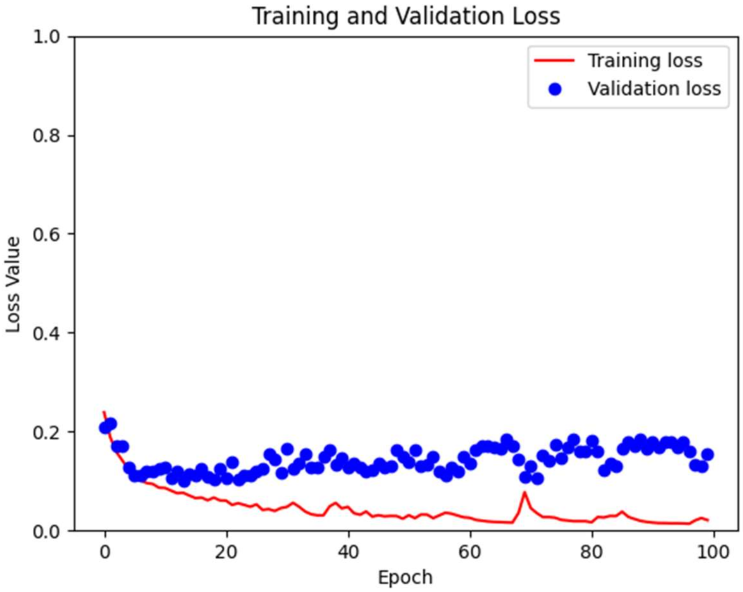


Figure 56 Training and Validation Loss over Time

Using the predicted mask from the TensorFlow model, an OpenCV script was run to isolate the edges of the mask/sidewalk such that the relative position of the robot can be found. The process of simplifying the edges of the sidewalk is accurate to a reasonable degree. Sample images isolating the edges of the sidewalks are shown below in *Figure 57*.

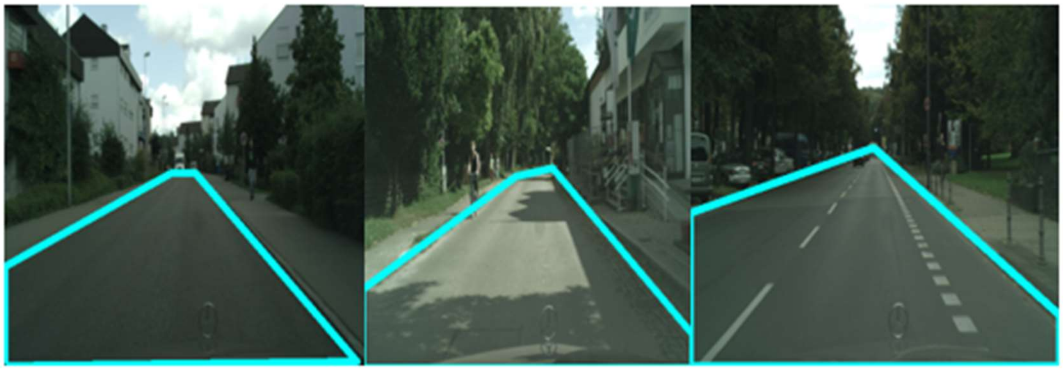


Figure 57 Computer vision road detection

Having outlined the road, the first step in localizing the robot's position was to isolate the left/right edges of the road from the rest of the road. The initial implementation utilized an angle-based approach, measuring the angle from vertical to each line in the road boundary. Lines outside the desired angle range were discarded and for simple roads (straight and without intersections), this was sufficient. Using the left/right road edges, the intersection of such lines was calculated to determine which direction the robot will eventually need to travel to successfully reach the destination. The program also determined the robot's current position relative to the road by determining where the midpoint between the base of the road edges lies concerning the camera. Despite these successes, the CityScapes dataset, on which this model was trained, does not sufficiently match the WPI sidewalks and a new training dataset and TensorFlow model must be trained for the WPI campus.

Internet Connectivity

The internet connection around the WPI campus has been shown to be sufficient for preliminary testing of the GPS with RTK data. The WPI Wireless Network reaches speeds over 300 Mbps, which is greater than what would be requested from this robot, but the strength of the network varies greatly. While speeds over 300Mbps can be reached in many indoor parts of the campus, isolated outdoor areas, such as on the quad, have reduced connection speeds, reaching only 10 Mbps. Due to the numerous access points, the Raspberry Pi will temporarily lose internet connection when transitioning from one access point to another, so a continuous internet connection cannot be assumed. These momentary disconnections and areas with reduced speeds will not affect the robot's localization (because GPS precision is no longer reliant on RTK data) and will have a negligible impact on the robot's ability to receive navigation commands/requests (so long as the robot does not come to rest in a problematic area which can be easily avoided). If such connection issues are shown to be problematic, then the signal strength of the network across campus can be mapped and used as part of the A* heuristic to maximize signal strength and robot reliability while traversing the WPI campus.

Collision Avoidance

Initial testing of the “LDS-01 360” liar on the prototype chassis resulted in frequent collisions. These collisions may have been the result of slow processing onboard the robot or limited low-speed robot control. When the RPLidar replaced the original lidar and was implemented on the final robot chassis, the lidar was successfully able to determine when an obstacle was in front of the lidar and stop the wheels. The lidar scanning range was 1m and this range was sufficient to decelerate the robot to zero velocity before hitting the obstacle. This testing was performed with the robot moving at approximately 20% of the maximum speed.

The integration of the lidar was tested by walking directly in front of the robot while it was driving, to simulate a situation where students on campus block its path. During the experimentation, the robot was successfully able to stop its movements when there was an obstruction in front of it. When the obstructions cleared, the robot resumed its normal operations.

Conclusion

While this research project was successful in autonomously driving a robot across the WPI campus and in developing a robust robot chassis, the shortcomings of the implementation of the novel belt box transmission system prevented the main robot from completing deliveries. The GPS, compass, and lidar system were able to guide the robot along WPI pathways without collisions but required occasional human intervention to return to the planned paths, as expected for a single year of development into a custom autonomous robot. To improve the reliability of the transmission system, it is recommended that a larger belt profile be used (to increase the belt strength and prevent breakage) and the bearing slip fits be looser (to allow for more alignment of the internal components). To better stay on the WPI pathways, it would be recommended to implement an RTK GPS such that centimeter-level precision can be used in path planning.

Additionally, retraining the TensorFlow image segmentation model using a custom dataset of the WPI sidewalks would allow for the robot to be less reliant on the GPS data and stay on the intended path. Collision avoidance could also be improved by detecting the sides of an object and creating a route around objects. Furthermore, the robot could navigate around the object, instead of waiting for the obstruction to move. The robot motion can also be improved through the implementation of arc turns rather than the current point-and-shoot motion for more efficient navigation. The security of the carried packages should also be improved by improving the cargo box to a stronger material that can not be easily broken into. And an electronic lock and authentication system should be implemented to prevent theft.

References

[1] Lee, I. (2021). Service Robots: A Systematic Literature Review. *Electronics*, *10*(21), 2658. https://doi.org/10.3390/electronics10212658

[2] Marketsandmarkets. *Service robotics market*. Market Research Firm. Retrieved April 25, 2022, from https://www.marketsandmarkets.com/Market-Reports/service-robotics-market-681.html

[3] West, D. M. (2022, March 9). *Brookings Survey finds 52 percent believe robots will perform most human activities in 30 years*. Brookings. Retrieved April 25, 2022, from https://www.brookings.edu/blog/techtank/2018/06/21/brookings-survey-finds-52-percent-believe-robots-will-perform-most-human-activities-in-30-years/

[4] Kim, S. (S., Kim, J., Badu-Baiden, F., Giroux, M., & Choi, Y. (2020, December 4). *Preference for robot service or human service in hotels? impacts of the COVID-19 pandemic*. International Journal of Hospitality Management. Retrieved April 25, 2022, from https://www.sciencedirect.com/science/article/pii/S0278431920303479

[5] Jennings, D., and Figliozzi, M. (2019). A Study of Sidewalk Autonomous Delivery Robots and Their Potential Impacts on Freight Efficiency and Travel, Forthcoming 2019 Transportation Research Record.

[6] Rogers, K. Higher Demand for Quick Delivery is Creating a Boom in Jobs. CNBC News. 6 2018. https://www.cnbc.com/2018/05/04/higher-demand-for-quick-delivery-is-creating-a7 boom-in-jobs.html. Accessed October 24, 2018.

[7] Pani, A., Mishra, S., Golias, M., & Figliozzi, M. (2020). Evaluating public acceptance of autonomous delivery robots during COVID-19 pandemic. Transportation Research Part D: Transport and Environment, 89, 102600. https://doi.org/10.1016/j.trd.2020.102600

[8] FedEx. *Meet Roxo, The FedEx Sameday Bot*. FedEx. Retrieved December 16, 2021, from https://www.fedex.com/en-us/innovation/roxo-delivery-robot.html

[9] Reporter, M. B. |. (2021, March 30). *How the Starship Robots Work*. BG Falcon Media. Retrieved December 16, 2021, from https://www.bgfalconmedia.com/news/how-the-starship-robots-work/article_84412830-6acc-11ea-bf09-1bb7696209a5.html

[10] Brown, D. (2020, July 22). *Meet Scout: Amazon Is Taking Its Prime Delivery Robots to the South*. USA Today. Retrieved December 16, 2021, from https://www.usatoday.com/story/tech/2020/07/22/amazon-taking-its-scout-delivery-robots-south/5485106002/

[11] WPI. *Makerbucks- the "Go and try" prototyping program*. WPI. Retrieved December 16, 2021, from https://www.wpi.edu/news/announcements/makerbucks-go-and-try-prototyping-program

[12] WPI. *Tinkerbox*. WPI. Retrieved April 26, 2022, from https://www.wpi.edu/about/innovation-entrepreneurship/programs/tinkerbox

[13] NHRL. *Mantella*. Mantella - NHRL. Retrieved April 26, 2022, from https://50day.io/wiki/index.php/Mantella

[14] TensorFlow. (n.d.). *Image segmentation :   Tensorflow Core*. TensorFlow. Retrieved April 26, 2022, from https://www.tensorflow.org/tutorials/images/segmentation

[15] Guilly, Y. L. (n.d.). *Semantic segmentation with tf.data in tensorflow 2 and Ade20k dataset*. Stochasticity and Chaos. Retrieved April 26, 2022, from https://yann-leguilly.gitlab.io/post/2019-12-14-tensorflow-tfdata-segmentation/

[16] The WPI Hub. (n.d.). *Article: Connect to WPI wireless using a raspberry pi with Gui*. The WPI Hub. Retrieved April 26, 2022, from https://hub.wpi.edu/article/361/connect-to-wpi-wireless-using-a-raspberry-pi-with-gui

Appendices

Appendix A

ROS workspace: https://github.com/TianYuFan1/mqp_ws

Appendix B

User interface workspace: https://github.com/TianYuFan1/html_ws

Appendix C

Video of initial reduction prototype motion: tinyurl.com/IBB-hand-spin

Appendix D

Video of first powered prototype reduction spin up: tinyurl.com/IBB-motor-spin

Appendix E

Video of turning tires down: https://tinyurl.com/boring-carefrees

Appendix F

Bill of materials

Item	Quantity	Total Cost	Link
Slamtec RPLIDAR A1M8 Lidar	1	$99	https://www.amazon.com/dp/B07TJW5SXF

C94-M8P GPS	1	$240	https://www.digikey.com/en/products/detail/u-blox/C94-M8P-2/6150728
Raspberry Pi 4 B 4GB	1	$55	https://www.raspberrypi.com/products/raspberry-pi-4-model-b/
Serial-USB Converter	1	$8	https://www.amazon.com/gp/product/B0753HBT12
AeroSky MC4830 420KV Brushless Motor	4	$67	https://www.ebay.com/itm/233139334246?hash=item36482fec66:g:t9IAAOSwwmlcbIN8
Brushless ESC	4	$119	https://flipsky.net/products/flipsky-mini-fsesc6-7-pro-70a-base-on-vesc6-6-with-aluminum-anodized-heat-sink
AS5600 Magnetic Encoder	4	$4	https://www.amazon.com/dp/B097QNG1CN
MR30 Connector	1	$10	https://www.amazon.com/dp/B0747MW9RX
XT60H Connector	1	$8	https://www.amazon.com/dp/B07Q2SJSZ1
18AWG Wire	1	$24	https://www.amazon.com/dp/B0793LLH9R
MPU9250 IMU	2	$28	https://www.amazon.com/dp/B01I1J0Z7Y
AS5048 Magnetic Encoder	4	$85	https://www.amazon.com/dp/B08CHJMQV

Item	Quantity	Cost	Link
Flexible 3D Printer Filament	1	$32	https://www.amazon.com/dp/B0865PFBLG
Box Tubing Stock	1	$250	https://www.onlinemetals.com/buy/material?q=%3Aprice-asc%3AMaterial%3AAluminum%3AShape%3ATube-Rectangle%3AAlloy%3A6061%3AHeight%3A1%2522%3AWall%3A0.062%2522%3AWidth%3A2%2522
70mm Bearings	1	$74	https://www.amazon.com/dp/B093247LY7
Large Wheel Hub Aluminum Stock	1	$336	https://www.mcmaster.com/catalog/8974K2
Aluminum Standoff Stock	1	$6	https://www.mcmaster.com/catalog/8974K24
Rivets	3	$43	https://www.mcmaster.com/catalog/97519A651
Rivet Nuts	1	$10	https://www.mcmaster.com/catalog/94020A375
M5 Flanged Bolts	1	$7	https://www.mcmaster.com/catalog/90909A722

M5 Countersunk Bolts	1	$11	https://www.mcmaster.com/catalog/91263A835
M5 Socket Head Bolts	1	$10	https://www.mcmaster.com/catalog/91292A196
Aluminum Flat Bar	1	$49	https://www.mcmaster.com/catalog/8975K211
GT2mm Belts	4	$44	https://shop.sdp-si.com/catalog/product/?id=A_6R51M193060
External Retaining Rings	1	$6.55	https://www.mcmaster.com/90119A108/
10mm Bearings	2	$20	https://www.amazon.com/gp/product/B082PSCNKB/ref=ppx_yo_dt_b_asin_title_o03_s00?ie=UTF8&th=1
GT2mm Pulley	4	$64	https://www.mcmaster.com/3764N103/
15mm Bearings	2	$20	https://www.amazon.com/gp/product/B09FJD3M78/ref=ppx_yo_dt_b_asin_title_o05_s00?ie=UTF8&psc=1
Steel Shaft Stock	1	$14	https://www.amazon.com/gp/product/B07KT3J7NZ/ref=ppx_yo_dt_b_asin_title_o00_s00?ie=UTF8&th=1
Aluminum Round Stock 3.75”	1	$138	https://www.mcmaster.com/8974K96-8974K961/
Alignment Dowel Pins	2	$22	https://www.mcmaster.com/91585A185/

Aluminum Round Stock 4.75”	1	$416	https://www.mcmaster.com/8974K2/
Send Cut Send Manufacturing	1	$389	https://sendcutsend.com/ (custom invoice)

Total: $2,709