Autonomous Campus Mobility Platform

A MAJOR QUALIFYING PROJECT REPORT

WORCESTER POLYTECHNIC INSTITUTE

Submitted to Project Advisor:

Cagdas Onal, WPI Professor

Submitted by:

Garrison Hefter

Mitch Read

Dylan Roncati

Md Shamsur Rahman Saikat

Date: April 26, 2018

Abstract

This Major Qualifying Project (MQP) is based around the development of a robotic vehicle for use in improving mobility. The main objective was to create an autonomous vehicle capable of navigating a person or cargo back and forth from Higgins Laboratory on the Worcester Polytechnic Institute (WPI) main campus to the Robotics Laboratory located at 85 Prescott Street, approximately 0.6 miles away. An autonomous robot was uniquely designed as a personal mobility platform to navigate its environment using onboard navigation and sensing system.

We considered various levels of autonomy, researched the options for platforms, and established the primary and secondary requirements for our vehicle. Based on the specified requirements and budget we settled on a longboard design for the chassis. The lightweight, compact and portable design makes it ideal for driving on a sidewalk. Battery powered motors along with custom steering assembly and a control system allow for locomotion and navigation through information derived from various interfaces.

Acknowledgements

The team would like to express many thanks toward our project advisor, Professor Cagdas Onal, for his assistance and guidance throughout this project.

Table of Contents

List of Figures

1. Introduction

This Major Qualifying Project was done to create an autonomous mobility platform to transfer a person from the main WPI campus to the robotics building at 85 Prescott Street. The goal of this platform is to provide a light, effective, and autonomous solution to transportation on and around campus. While there are existing solutions serving similar goals, our design and implementation varies from these to provide benefits in that it offers the possibility of travel to individuals with compromised mobility and those without mobility issues, which is not offered in any other research space currently.

After careful consideration and discussion in the early stages of this project, the team chose to design and build a solid wood longboard chassis as the basis for this mobility platform. This base was chosen for its compact size and portability in contrast to the cumbersome existing wheelchair and scooter technologies. The chassis includes a battery-powered motor for locomotion, a handle for steering, control system, and various sensors for environmental orientation. The robot is designed to navigate its environment using GPS navigation system, and LIDAR. Navigational decisions are made by the robotic vehicle via information derived through various interfaces.

This platform completes the following requirements to demonstrate fulfillment of requirements differing from any existing system.

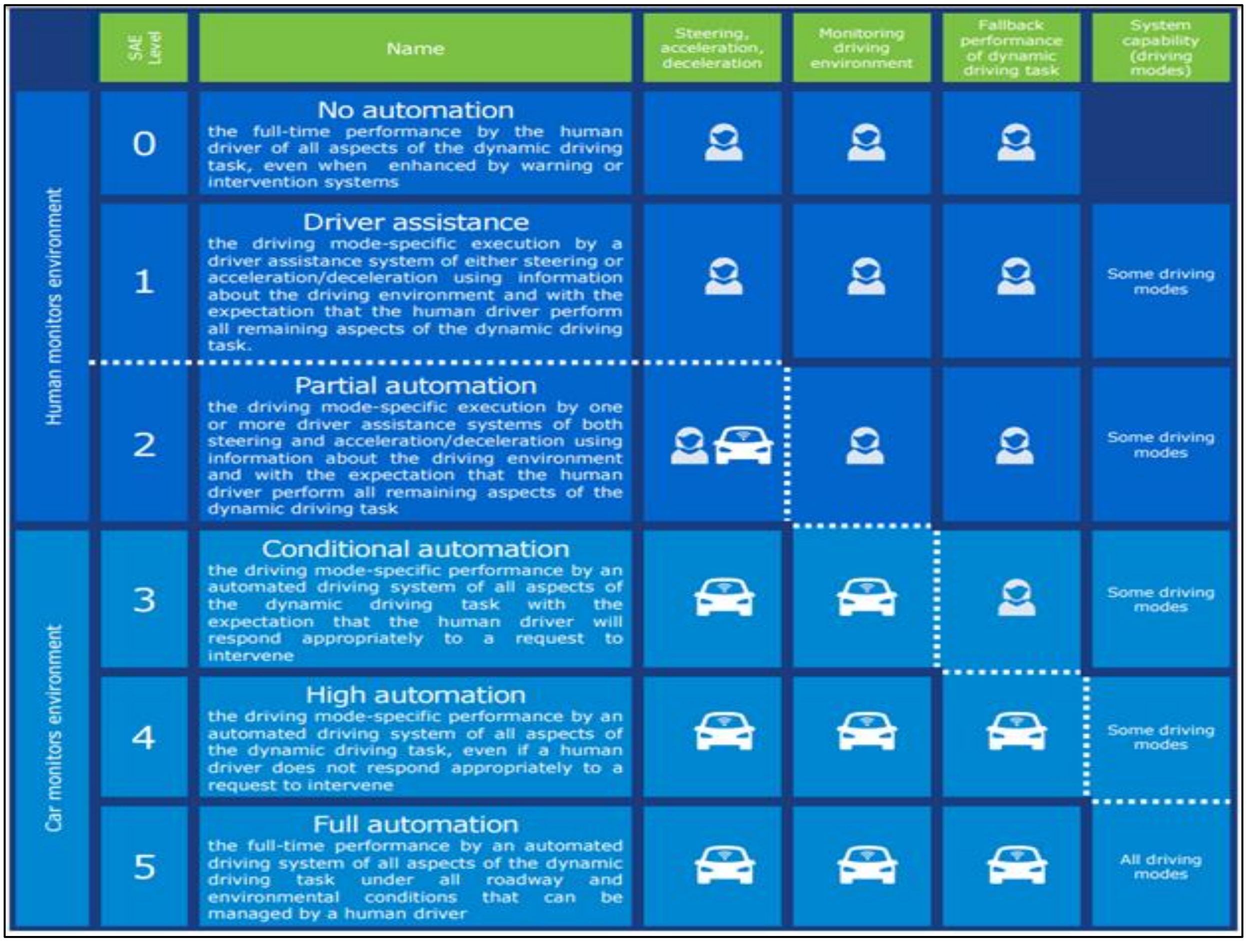
- The platform is light and compact enough for an adult to lift and carry
- The platform is able to autonomously navigate a person or cargo back and forth from a specified location on campus and back.
- The platform supports up to 200 lbs of cargo/passenger.

2. Background

Autonomous systems have been used at various scales for transportation for decades. They are used in aviation, weaponry, automobiles, industrial robotics, among others. The goal of this project inherently defines the scope as such that it is most relevant to technologies used in autonomous cars and personal mobility devices.

2.1 Levels of Autonomy

The capabilities of autonomous systems generally vary widely upon the application. According to the Society of Automotive Engineers (SAE) standard, vehicular automation can be classified into six levels, Level 0 - Level 5. The classification is primarily based on level of human intervention required to accomplish a task. The following visual (Figure 1) from a report published by the International Transport Forum summarizes these different levels of autonomy:



	SAE Level	Name	Steering, acceleration, deceleration	Monitoring driving environment	Fallback performance of dynamic driving task	System capability (driving modes)
Human monitors environment	0	**No automation** the full-time performance by the human driver of all aspects of the dynamic driving task, even when enhanced by warning or intervention systems	👤	👤	👤	
	1	**Driver assistance** the driving mode-specific execution by a driver assistance system of either steering or acceleration/deceleration using information about the driving environment and with the expectation that the human driver perform all remaining aspects of the dynamic driving task.	👤	👤	👤	Some driving modes
	2	**Partial automation** the driving mode-specific execution by one or more driver assistance systems of both steering and acceleration/deceleration using information about the driving environment and with the expectation that the human driver perform all remaining aspects of the dynamic driving task	👤 🚗	👤	👤	Some driving modes
Car monitors environment	3	**Conditional automation** the driving mode-specific performance by an automated driving system of all aspects of the dynamic driving task with the expectation that the human driver will respond appropriately to a request to intervene	🚗	🚗	👤	Some driving modes
	4	**High automation** the driving mode-specific performance by an automated driving system of all aspects of the dynamic driving task, even if a human driver does not respond appropriately to a request to intervene	🚗	🚗	🚗	Some driving modes
	5	**Full automation** the full-time performance by an automated driving system of all aspects of the dynamic driving task under all roadway and environmental conditions that can be managed by a human driver	🚗	🚗	🚗	All driving modes

Figure 1: Levels of Vehicle Autonomy (Breznak 2008)

Recent advances in self-driving vehicle technologies have enabled the achievement of Level 4: High Automation, albeit existing systems are still under testing and development. Equipping a system with Level-4 automation requires it to be data driven. Information is required to be constantly fed into the system and processed by complex algorithms in order to monitor changes continuously and perceive the dynamic environment (Autonomous Intelligent Vehicles, 2011). Figure 2 shows a control scheme for a generic autonomous mobile robot.

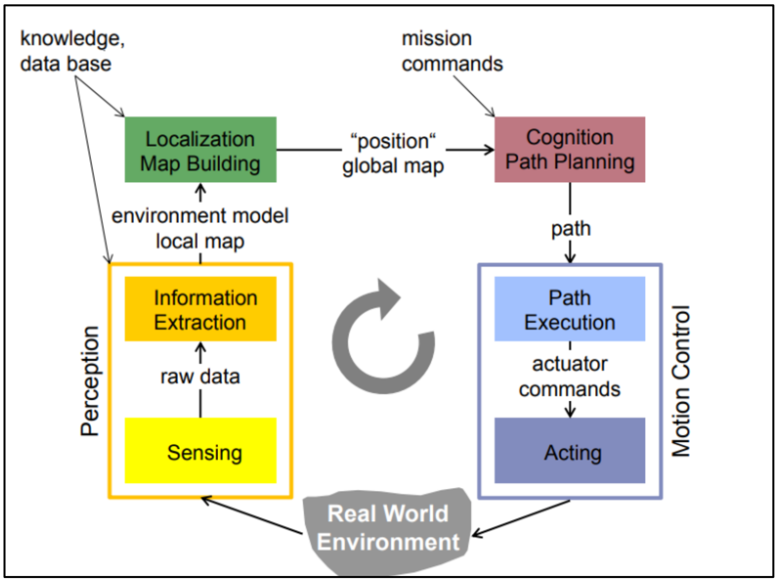


Figure 2: Closed-loop control of autonomous machines can be demonstrated in a flowchart.

The chain of actions shown in this figure involves four main processes: localization, path-planning, locomotion and obstacle avoidance. In order to move from point A to point B, the robot needs to know its current position and destination, how to navigate in physical space to get to the desired position without hitting obstacles, and how to mechanically drive itself (Autonomous Intelligent Systems, 2011). Each of these processes require fusion of information from all of the subsystems. Even though the intelligent systems in autonomous outdoor vehicles are based on similar technologies to indoor mobile robots, it is significantly more challenging to integrate them in a vehicular platform. Developing a reliable intelligent transport system requires practical resources and innovation, supply of which can be justified by fulfilling a big market segment. Thus, currently we have large automotive companies competing for self-driving cars and small scale commercial products or research projects for personal mobility platforms as an assistive device. Consequently, there lies an empty niche that can potentially bridge autonomous mobility and personal transport.

2.2 Personal Autonomous Vehicles

In this section, we examine and discuss the most current existing personal autonomous vehicle technology. Our research into these vehicles allows us to determine the attributes to consider in the development of this project. We explored varying chassis styles, sensor technologies, motors, power sources and sensors.

Determining which chassis to use for our vehicle began with an exploration of existing personal autonomous vehicles with a view toward creating a more lightweight and portable base. We discovered that most personal mobility devices have historically built on either wheelchair or scooter bases where the rider must travel in a seated position. More recently, however, the development of powered longboards has opened the possibility for upright autonomous mobility.

2.2.1 Powered Wheelchair Bases

There has been considerable research and development into adapting a standard powered wheelchair base as an autonomous mobility platform. We focused our research on the most recent projects and considered their individual attributes for possible incorporation in our project design.

In 2016, a semi-autonomous wheelchair was developed at Worcester Polytechnic Institute (WPI) using an existing powered wheelchair base. The project was designed to accommodate individuals with compromised physical mobility who are unable to use a traditional joystick controller. In order to accommodate individuals with such a degree of immobility, it is necessary to provide a seated position for travel. This platform included a laser rangefinder (LIDAR) to generate a map of the surrounding environment and localize the vehicle, and ultrasonic and infrared range sensors were used for obstacle and collision avoidance (Qiao, 2016).

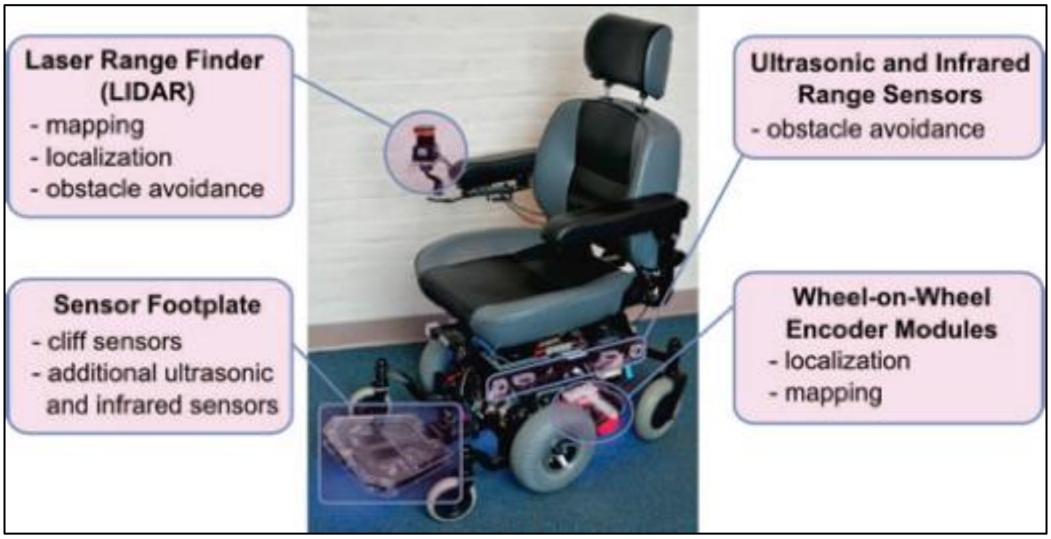

Laser Range Finder (LIDAR)
- mapping
- localization
- obstacle avoidance
Ultrasonic and Infrared Range Sensors
- obstacle avoidance
Sensor Footplate
- cliff sensors
- additional ultrasonic and infrared sensors
Wheel-on-Wheel Encoder Modules
- localization
- mapping

Figure 3: Sensor Components of WPI Semi-Autonomous Wheelchair (Qiao, 2016)

Similar autonomous mobility technology on a powered wheelchair base is under development at Massachusetts Institute of Technology (MIT). This technology is appropriate for people who have lost mobility due to brain injury or the loss of limbs, but who retain speech (MIT 2017). Project objectives center around the implementation of a voice-operated device that learns its environment and is capable of detecting objects and avoiding collisions. This required a speech interface to interpret commands, a wireless location device and motor-control software to direct the wheelchair's motion (MIT, 2017). This robotic wheelchair is capable of learning the layout of its environment and moving to any previously visited location (MIT, 2017).

Figure 4: MIT Autonomous Wheelchair Demonstration (Teller)

Using a Segway wheelbase, the SCEWO autonomous wheelchair has as project objectives self-balancing technology to enable it to rotate in place and the ability to climb steps. The design is functional and allows for many adjustments in the seating position but does not allow travel in an upright position.

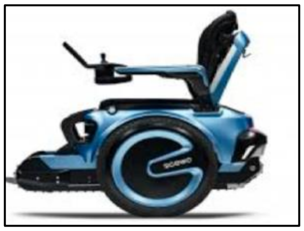


Figure 5: SCEWO Wheelchair with Segway Base (SCEWO, 2017)

2.2.2 Scooter Bases

Recognizing the limitations of the powered wheelchair as a base, particularly for outdoor road travel, research at MIT has centered on the creation of a door-to-curb-to-highway autonomous transportation system for the mobility impaired. Rather than the two wheeled power chair base, the chassis is a three-wheeled powered scooter that provides more flexibility and better handling. The system includes several layers of software including low-level control algorithms that enable a vehicle to respond immediately to changes in its environment, such as a pedestrian darting across its path (MIT Scooter). In addition, route-planning algorithms, localization algorithms and map-building algorithms orient the vehicle to its environment and enable efficient navigation (MIT Scooter). A scheduling algorithm is designed to allocate fleet resources and an online booking system that allows users to schedule rides (MIT Scooter).

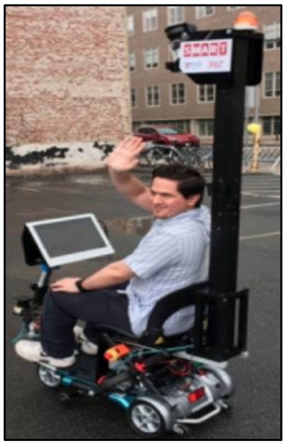


Figure 6: MIT Autonomous Scooter (Hardesty, 2016)

2.2.3 Longboard Bases

Powered skateboards and longboards are increasingly popular as a lightweight and portable mode of transportation. Currently, there exist some concept models and startup products that integrate smart technologies with a skateboard or longboard base. For instance, Audi has recently unveiled its autonomous longboard, which is made of aluminum and carbon fiber and can be used in one of three modes. It can be mounted and ridden in a standing position as normal longboard, used as a scooter by deploying the integrated handlebar, or put in a cargo mode where it can be loaded with cargo and follow its user autonomously by tracking a phone or smartwatch (Ziegler, 2016). Audi's board goes over seven miles on each charge and can travel at speeds up to 18 miles per hour (Ziegler, 2016).

Other longboard designs include Marbel 2.0 and the XTND, both commercial products fundraising on Kickstarter. The Marbel 2.0, already available in the market, was developed by a relatively new e-board brand Marbel based in Tampa, Florida. Their main feature is a downloadable mobile application that acquires data from every ride to allow the rider to customize the experience by fine tuning various parameters such as acceleration, speed, braking. The XTND, however, raises the bar significantly higher. With the first release slated for January 2018, the XTND board designed by a Czech startup is reportedly the first of its kind to integrate Artificial Intelligence (AI). In addition to similar capabilities of the Marbel 2.0 in terms of ride customization, this board can also fine tune its own parameters in an intelligent way to assist the rider to learn, as well as offering a smooth ride. Among many other features, the XTND offers route optimization based on previous ride data; has step recognition to sense presence of a rider; anti-collision/fall off system to minimize rapid speed change or vibrations at high speed; and automatic sleep mode to conserve battery.

Figure 7: XTND Board (XTND, 2017)

2.2.3 Conclusion

While both the autonomous wheelchair and scooter offer the possibility of travel to those with compromised mobility, such bases do not allow for travel in an upright position nor are they an appropriate means of travel for those who do not have mobility issues. Our project aims to meet the needs of both.

There are many individuals with other mobility issues that do not require the use of a wheelchair, autonomous vehicles that require the user to be seated unnecessarily confine and restrict the movement of those who are capable of standing. The presumption that everyone with a mobility issue needs to be seated in order to travel autonomously is unwarranted. There are many types of mobility impairments that may require a mobility aid but do not necessarily preclude the individual from remaining in a standing position. Such impairments may include blindness or visual impairment, heart conditions, respiratory disorders, hearing or other sensory impairments. While individuals with these issues may need assistance with transfers and ambulation, they may not need to be restricted to a seated position to receive it.

Additionally, our project brings autonomous transportation to people who are not disabled but who simply need or prefer a lift to their destination. The ability to travel at speeds faster than and without the effort required for walking is appealing to many.

While there are many electric and powered longboards, our project is distinct in that it provides the ability for both abled and disabled riders to travel autonomously.

2.3 Sensors and Communication Hardware

Autonomous vehicles detect their surroundings primarily using RADAR, LIDAR, GPS, odometry, and camera-enabled computer vision. All the sensory data from the preceding sources is then analyzed by the control system. We researched and investigated various detection systems to determine which were best suited to the safe navigation of our vehicle. Each system has different attributes, not all of which are relevant to our goals.

Many autonomous vehicles employ forward RADAR for navigation. Forward RADAR involves reflected microwaves used to identify the location and speed, but not always the type, of nearby objects. Radio waves are sent out and bounced off objects. A major benefit of RADAR is the ability to detect objects at great distance. It allows the vehicle to see up to 100 meters away in dark, rain, snow or vision impaired circumstances. While this is an important consideration for night mobility and all types of weather, RADAR cannot differentiate between objects.

Another popular detection system involves the use of LIDAR, essentially the word *laser* and *RADAR* combined. LIDAR works by sending out light pulses and reflecting them off objects. The distance of an object is then measured by detecting its return signal. LIDAR works well in the dark and offers continuous scanning to provide 3D omni-directional view of surroundings.

The ability to map surroundings is also critical to autonomous vehicles. Once a local map is updated, software processes this information, plots a path and sends instructions to the "actuators" on the vehicle which control acceleration, braking and steering (Self Driving, 2017). Hard-coded rules, obstacle avoidance algorithms, predictive modeling and "smart" object discrimination which determines the difference between a person and an object and help the software follow a desired path and navigate obstacles (Self Driving, 2017). There are two possibilities for mapping: (1) an intensive process of manually capturing a detailed map of the vehicle's surroundings and programing the vehicle. (2) by teaching the vehicle to learn its environment in much the same way as a person would whereby the vehicle is taken around once on a guided tour, with important places identified along the way using GPS receivers (MIT, 2017). Indoor mapping, however, cannot rely on GPS but may be able to achieve using Wifi signals, with wide-field cameras and laser rangefinders, coupled to computer systems that can construct and localize within an internal map of the environment as they move around (MIT, 2017). GPS combined with high-precision mapping is used to determine a vehicle's position on the road.

Forward-facing camera uses image-processing software to detect lane stripes, signs, and other objects (Gates, 2016). Cameras take images that are interpreted by computer. This method of detection is limited to interpreting only what the camera can see. Depending on the type of camera and its position on the vehicle, the camera may only be capable of detecting a small portion of the environment. In addition, it relies on the installation and integration of lane stripes and other objects into the surrounding area to facilitate orientation of vehicle.

Autonomous vehicles use computer vision to convert visual images derived from their detection systems into descriptions to interface with other processes and elicit appropriate actions by a machine such as braking or turning to avoid an object (Stephens, 2016). A multidomain controller on board the vehicle manages input from mapping and navigation data from the above detection systems to form navigational decisions.

2.4 Motors and Power Sources

Options for power sources for electric longboards are typically either single or dual drives. Each offers benefits worthy of consideration.

Single drive longboards involve one motor, one mount and one Vedder Electronic Speed Controller (VESC). Having one of each component saves space on the board and keeps the board light. A single drive is more economical than having two of each component. Importantly, a single drive long board can achieve the same speed as a dual drive. However, acceleration is considerably slower and braking and turning are less efficient with only one motor.

Dual drive longboards are more expensive since there are two motors, two mounts. However, there are many benefits to having dual drives. They have the benefit of reducing the stress on each drive system. In a dual drive, each drive shares the amperes so one motor is not doing all of the work. This translates to motors that run cooler, more efficient braking and faster acceleration. In addition, dual drives work better in off road and harsh weather conditions like snow and heavy rain as they provide better traction.

There were several considerations regarding the drive and selection of appropriate power sources, not the least of which was cost. Overall, while speed is the same whether using single or dual motors, the dual motors provide better acceleration and better braking. Clearly, having dual motors is optimal.

In determining whether to use a single or dual drive on this project, however, cost was a driving factor. A single drive does mean inefficient turning, slower acceleration and less efficient braking. However, with only one motor, one mount and one VESC, the cost is half that of a dual drive. While our vehicle would likely benefit greatly from dual drives, the cost of having to purchase two of everything would have been prohibitive given our restricted budget for this project.

One more motor or actuator is required to steer the board. While steering can be handled in a number of ways, mounting a sprocket to the truck is very simple. The sprocket can be turned with chain back to the motor or actuator.

3. Goal Statement and Objectives

This project aims to find a means of transporting a person or cargo autonomously from one point to another. Our prototype involves the design and creation of a chassis, drivetrain, sensing and control system for a new autonomous vehicle capable of transporting a person from Higgins Laboratory to 85 Prescott and back.

To achieve our overarching aim, the objectives of the project are defined as follows:

1. To produce a robotic mobility vehicle
2. To design and build a powered chassis
3. To design and build a sensor system allowing the robot to navigate efficiently
4. To develop a control system to analyze data from sensors about the environment and permit the robot to operate safely in its environment

4. Task Specifications

This project development was based on two main categories of considerations from which individual attributes were developed.

4.1 Primary Specifications

The primary considerations for the project included physical and mechanical attributes, power and performance requirements and parameters for sensing and autonomy.

In terms of physical and mechanical attributes, we have the following specifications encoded in “Task Spec - Primary - Number” format (TSPXX):

- **TSP01**- The platform must support and carry 200lbs of load, either cargo or a rider.
- **TSP02**- The platform must be portable with a total mass of less than 10 kg.
- **TSP03**- The platform must have a safety feature that assists rider to remain on the platform such as a handle, foot straps, or other means of securing the rider or load.
- **TSP04**- The platform must have a top speed of 10 mph or below.
- **TSP05**- The platform must be capable of making multiple trips back and forth from Higgins Labs to 85 Prescott Street on one full charge.
- **TSP06**- The platform must be able to reach full charge in less than 5 hours.
- **TSP07**- The platform must be able to locate itself outdoors (using global positioning system (GPS)).
- **TSP08**- The platform must be able to map its environment at a suitable range for obstacle detection, and minimum stopping distance (which will be determined based on features of design components).
- **TSP09**- The platform must have a load detection system to sense when a rider or load above a weight of 50lbs is present.
- **TSP10**- The platform must not exceed the measurements of 12 inches wide, 38 inches long, and 8 inches high.
- **TSP11**- The cost of the project must not exceed 800 USD.

4.2 Secondary Specifications

Additional mechanical attributes that may be taken into account include ergonomic design elements to minimize discomfort for the rider. In terms of these attributes, we have the following secondary considerations encoded in "Task Spec - Secondary - Number" format (TSSXX):

- **TSS01**- The platform should be waterproof for outdoor use
- **TSS02**- The platform should have a commute mode
- **TSS03**- The platform should have a sport mode
- **TSS04-** The platform should be able to reach full charge in less than 3 hours
- **TSS05**- The platform should make independent decisions without user intervention
- **TSS06**- The platform should be capable of adjusting torque according to terrain.

5. Community Interest

WPI has a total student population of approximately 6,300 comprised of approximately 4,200 undergraduate and 2,100 graduate students. The number of employees including professors, administrative and other staff is in the hundreds.

To calculate the proportion of students at WPI who could benefit from a robotic campus mobility platform, we conducted a survey to determine the need or desire for an autonomous mobility platform. There were 400 responses were generated and analyzed to determine the results.

The overwhelming majority of responders, 96%, were students with the remaining 4% made up of faculty and university administration. Of the population, 20% indicated that they had mobility issues that affected their ability to navigate on campus. The majority spend between five and ten minutes walking between campus buildings every day, including trips between the main campus and off campus buildings.

The overall results of the survey were positive for interest. Of the surveyed individuals, 70% were interested in using an autonomous platform to get around campus. Of those, 69% indicated that they were likely to use our particular powered longboard campus mobility platform.

The development and applicability of the mobility platform envisioned by this project is not limited to use on the WPI campus. Such technology can have applications in other domains, including commercial development in the same manner as other autonomous transportation technology including personal vehicles, trains, aircraft and undersea vehicles.

6. Design Overview

After careful consideration of the possible chassis options for this project, we narrowed down our selection to a longboard deck. It offers the most universal use and appeal to those needing assistive mobility devices who are not confined to wheelchairs as well as use by those who are completely able bodied.

In order to simplify the overall design process, the platform has been divided into 5 subsystems according to functionality. The subsystems/subassemblies are tabulated as follows:

Chassis	**Drivetrain**	**Power distribution**	**Steering**	**Sensing/Control Unit**
Deck	Motor	6S2P LiPo Battery	Seat motor	Raspberry Pi 3
Handle	Motor pulley	3V LiPo for LIDAR	Sprocket	Arduino Uno
Trucks	Wheel pulley		Belt/Chain	Teensy 2.0
Motor mount	Drive Belt		Thrust bearing	LIDAR
Acrylic Support	Wheels		Adapter	GPS
			Hex bearing	IMU

7. Mechanical Design

7.1 Chassis

Preliminary design of the structure of the longboard chassis involved research and consideration of various deck designs and multiple considerations for construction, including choice of materials. The design of the handle was conceived to allow it for stability for users in need of extra stability, but with the flexibility of a fold down storage position for those who do not. Trucks and motor mount selections were made with consideration for the overall design and specifications of the longboard deck, drivetrain, power and sensing units.

7.1.1 Longboard Deck Design and Construction

The first step in design conception of the deck involved consideration of six basic deck shapes typically seen in powered longboard construction: cruiser, carver, slider, free rider, racer, and dancer.

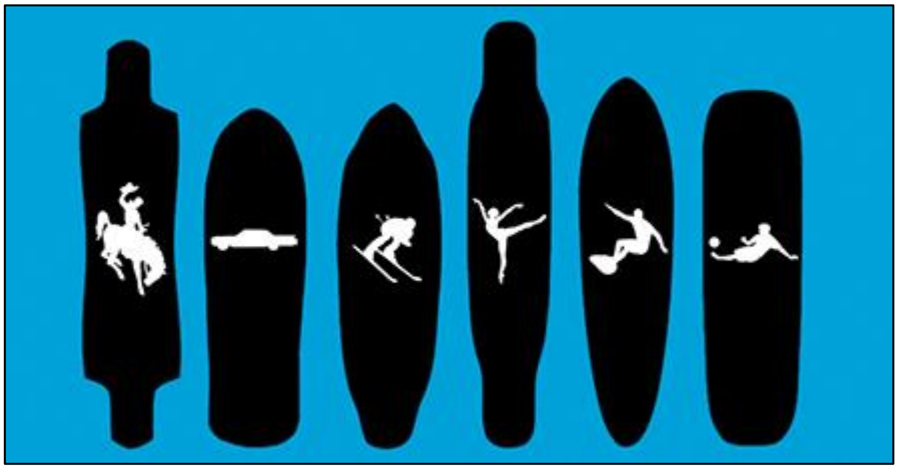


Figure 8: Longboard deck shapes left to right: Free rider, Cruiser, Racer, Dancer,

Each of these deck shapes offers a different ride experience and maneuverability. For our purposes, we chose a modified version of the free rider. This design works well on hills and is ideal for speed. The wheel cutouts allow for maximum handling and turning.

Calculating the space needed for the fold-down handle, a single drive, battery and sensing control units, we settled on deck dimensions of a length of 30 inches (76.2 cm) and a width of 9 inches (22.86 cm) at the widest area of the board and a thickness of 0.625 inches (1.59 cm).

Our next consideration involved choosing the materials to build the deck. The choices included Carbon Fiber, Baltic birch and Maple plywood, Bamboo, and Maple. Each material offered a different combination of benefits and inconveniences.

Carbon fiber offers the most ideal attributes for this project. It is lightweight, durable, stable and can be customized to cover the electronic components. This would be extremely beneficial to protect these components from the negative effects of rain and snow. Unfortunately, this material is extremely expensive and given the budgetary considerations of this project, it was not feasible to use it to construct the deck.

Moving forward, we analyzed the characteristics of the remaining materials before making our final selection. Plywood, while cheap and strong is prone to splintering and is very stiff, which may present difficulties in ease of maneuverability. Bamboo is extremely light, which is an asset in keeping the weight of the board under 10 kg, however it is very brittle, cracks easily and is rather expensive. We concluded that construction a deck out of stacked maple allows for easy customization and offers the rigidity we were looking for at a moderate price.

Rather than making a deck from one solid piece of wood, the deck was constructed by layering the wood in a cross-grain pattern to create a strong board that would be highly resistant to warping or splitting, two of the main disadvantages of natural wood. This project achieved construction of the longboard deck using nine layers of pressed maple veneer, laminated together using modern wood glue and clamping techniques.

7.1.2 Board Design and Removable Handlebar Design

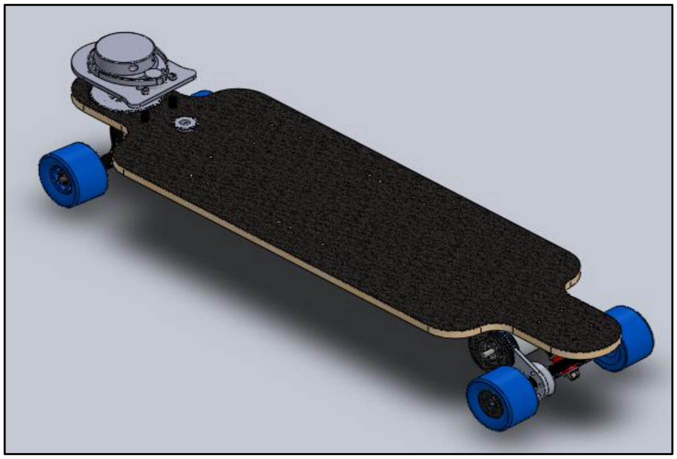


Figure 9: Preliminary Board SOLIDWORKS Design

According to our third technical specification (TSP03) the platform must have safety features to assist the rider. One of the safety features we designed was an adjustable handlebar. The handlebar has a max extension of 40.0 in with a minimum collapsed distance of 26.0 in. The 14 in adjustable height allows riders of various heights to hold the handlebar at a position which is most comfortable for them.

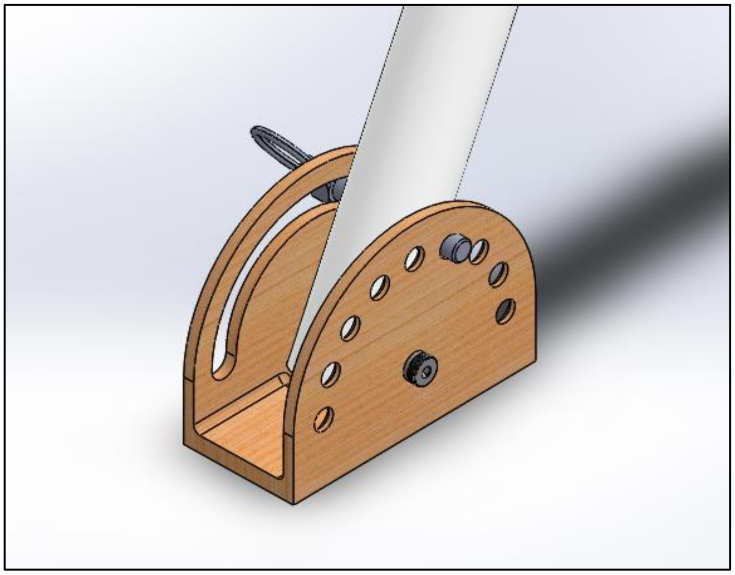


Figure 10: Preliminary Handle Locking design

The base for the handlebar is designed so that the handlebar can adjusted to different angles. By pulling the pin and supporting the shaft, the rider can adjust the angle of the handlebar to a comfortable position when riding. The handlebar also has the capability to collapse and lay parallel to the platform for easy storage.

We decided that the most effective way to build the handlebar would be buy the handle and support pieces and to build the base plate. We were able to find precut PVC to dimensions we needed as well as the pull pin we needed on McMaster Carr. We decided to make the base plate out of the maple we used to build the deck. Maple was chosen as it is easy to machine and is strong enough to support the rider and handlebar. This initial design was ultimately not implemented, due to concerns for its appropriateness for a disabled rider.

7.1.3 Trucks

After researching different options, we found that the best way to mount the wheels and steering is using existing longboard trucks that we can purchase, then modify them according to our design. The front truck will need minor modifications to handle steering and the rear truck will need some minor modifications to handle power input to the rear wheels. We have selected a truck set that we have found is the best for our application.

7.1.4 Board and Component Protection Considerations

Electronic components need to be protected from impact and from wet weather. To best achieve this protection, components benefit from dual protection. The first layer of protection usually involves treating the components with a conformal coating. While there are several spray coatings on the market, they are extremely expensive and beyond the budget of this project. For this reason, we decided to not treat our components. Instead, we opted for protection for the electronic components by a protective hard case cover.

The basic considerations for a hard case to protect our electronic components include durability, weight, impermeability, ground clearance and price. There are several such enclosures available for purchase that fit many of our requirements.

We investigated the available designs and concluded that none actually would work perfectly with our specific placement of electronic components on the board. Additionally, after consideration of our budget, we realized that buying a premade enclosure would be prohibitively expensive and not necessarily offer us the characteristics we require in terms of weight and ground clearance.

Instead, we considered three primary ways of creating our own enclosure: 3D printing (ABS or Nylon), repurposing of existing products, and vacuum forming thermoplastic. There are many positive aspects to 3D printing including custom fit, lightweight and easy of creation.

Depending on the materials used, this type of enclosure has variable durability for our application. We experimented with a basic design in the initial stages of consideration to determine exactly what would be needed.

However, we quickly concluded that the time spent to create the design for 3D printing would likely change throughout the project each time we made adjustments to the placement of our electronic components. In addition, we considered that 3D printing can be time consuming and we may have to print out several prototypes along the way before getting the proper fit. Furthermore, in the event that an enclosure breaks, it would potentially leave our components unprotected and cause delays in testing while we wait for a replacement.

In the interest of budget and time, we briefly considered repurposing materials such as used plastic plant trays and old baking tins as covers for our electronic components. Both are lightweight and waterproof with the plastic being less durable. Neither was particularly pleasing aesthetically, nor did they provide the custom protection we prefer for our components.

Vacuum formed enclosures provide a custom fit, are lightweight and durable and waterproof. To produce such an enclosure is considerably more work intensive than 3D printing. A vacuum former machine and a mold of the electronic components have to be hand created in advance. The machine consists of a hollow box with a suction port and a grid of holes. Inexpensive thermoplastic ABS or polycarbonate sheets are heated close to their melting point and then placed on the machine over a mold of the components. Air is vacuum pulled out of the box with a wet dry vacuum and the plastic sheets harden over the mold. This yields an absolutely custom enclosure.

After careful consideration, we determined that the vacuum forming machine was best suited to our purposes and needs. While initially quite labor intensive, once the vacuum forming machine and mold were and constructed, we could use them to make as many enclosures as we wanted, allowing us to quickly experiment with different designs along the way without losing time.

7.1.5 LIDAR Support

On the front of the board we designed, and laser cut an acrylic support to hold the LIDAR, Arduino and Teensy as well as protect the sprockets and chain from being stepped on by the rider.

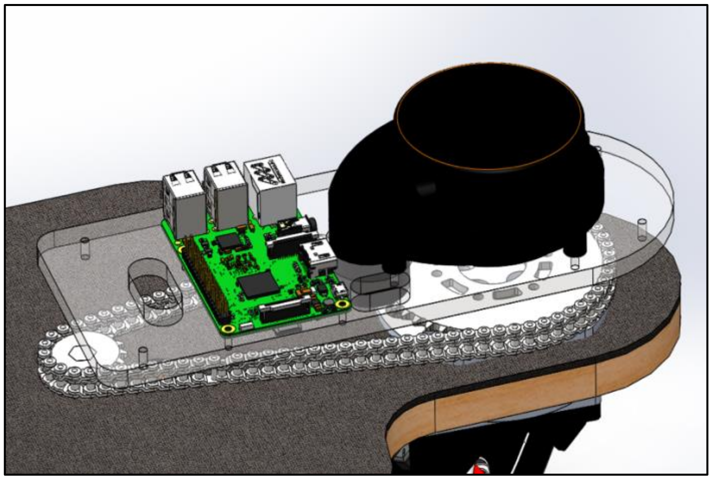


Figure 11: Acrylic LIDAR Support

7.1.6 Longboard Construction

Nine thin sheets of maple veneer were coated with standard polyvinyl acetate wood epoxy and then stacked and put into a press to set. To get the strongest bond, the glue was allowed to dry completely for several days before attempting the shaping of the board.

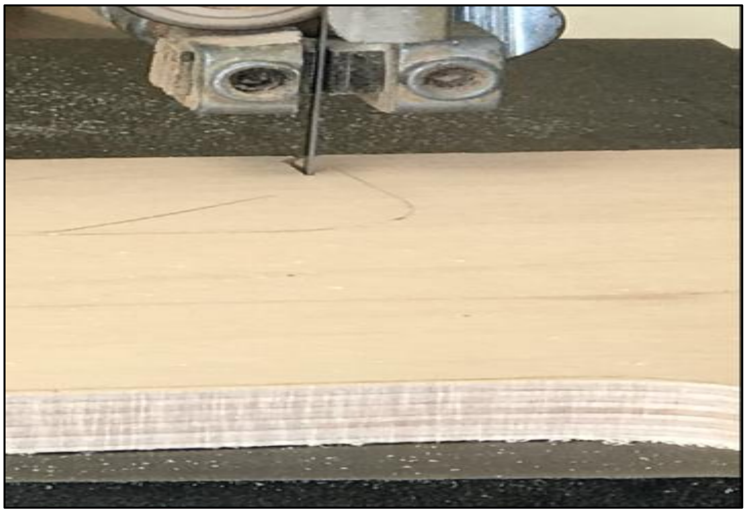


Figure 12: Rough cutting on the bandsaw.

A pattern for the board was made based on the design specifications. The pattern was transferred by tracing onto the blank deck. The board was rough shaped first on a table saw to cut down on the length. From there, the ends of the board were shaped on a bandsaw. Because of the curves in the design, this took several passes, each time removing a bit to avoid splitting the deck or overcutting the rounded areas.

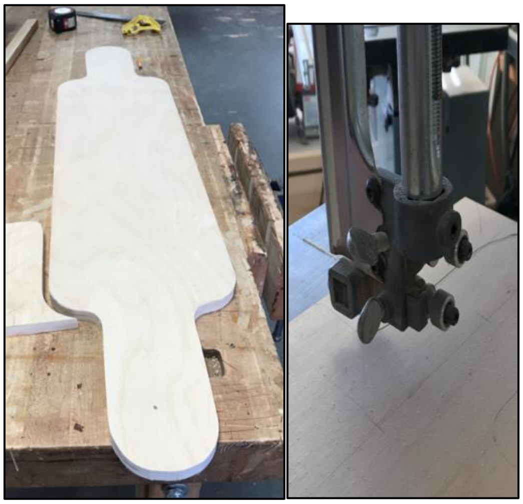


Figure 13: a) Left: Verifying against pattern b) Right: Trimming with router

Once the shape of the board closely matched the pattern, we used a router all around the perimeter of the board to trim off and refine the shape.

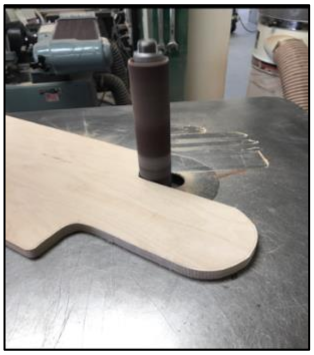


Figure 14: Finish on spindle sander

The final step, was to smooth out the board's edges on the spindle sander.

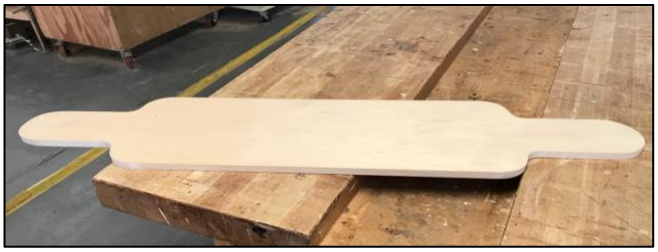


Figure 15: Finished longboard deck

7.1.7 Electronic Component Enclosure

Construction on the vacuum forming machine began with a design in the woodshop based on the size of our board and budgetary considerations. We used available scrap materials to cut down on costs. We constructed the frame of our hollow box out of maple so that it would be solid and hold up to the air pressure that would be present during the vacuum forming process. We cut the boards in two lengths so that when constructed they would form a rectangular box a bit larger than our desired enclosure. In one of the sides, we bored a hole the size of the hose of a standard shop vacuum.

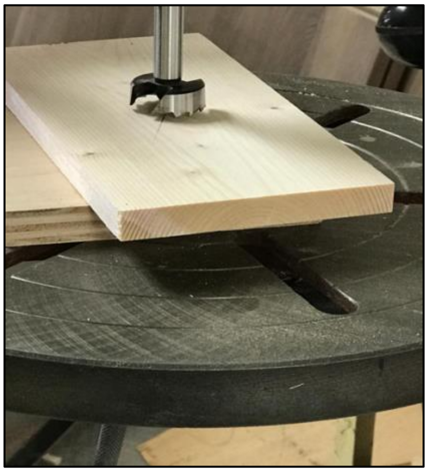


Figure 16: Boring hole in side of vacuum forming machine

There are several methods to joining the wood together to form the box. We had to be mindful of the force of the air pressure during the vacuum forming process and ensure that our joints were solid. Rather than attach the sides of the frame with glue and brads, we opted to use a joiner and biscuits. In this method, a machine is used to cut slots into each piece of wood to be joined. Compressed wood biscuits are placed in slots and glue is added to strengthen the bond. The parts are clamped and allowed to dry completely overnight. This procedure for joining is especially effective when dealing with 90degree angles such as those in our box construction. We were lucky to have access to a Domino, a tool that creates flat biscuits that resist torqueing. This process, while also work intensive, gave our box much stronger joints and better resistance in our application.

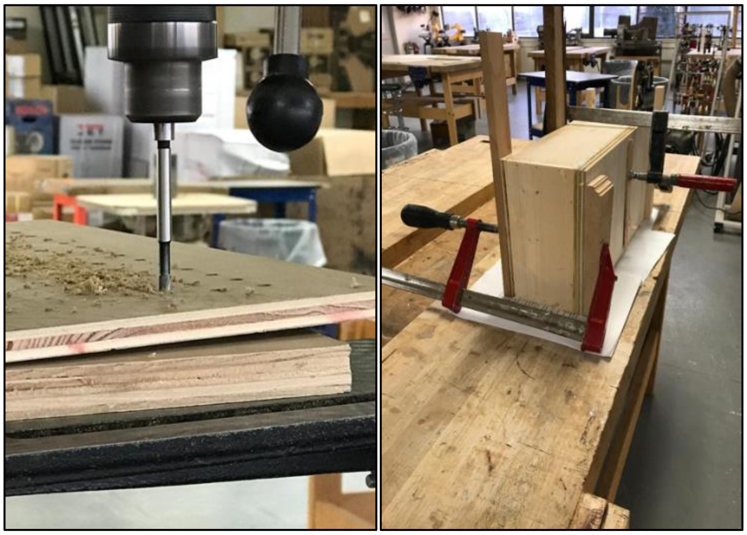


Figure 17: a) Left: Drilling of holes and b) Right: clamping during construction of vacuum

For the top grid plate, each hole was drilled using a bench drill. The placement of the holes was designed so that air would be removed uniformly across the thermoplastic sheet during the forming process.

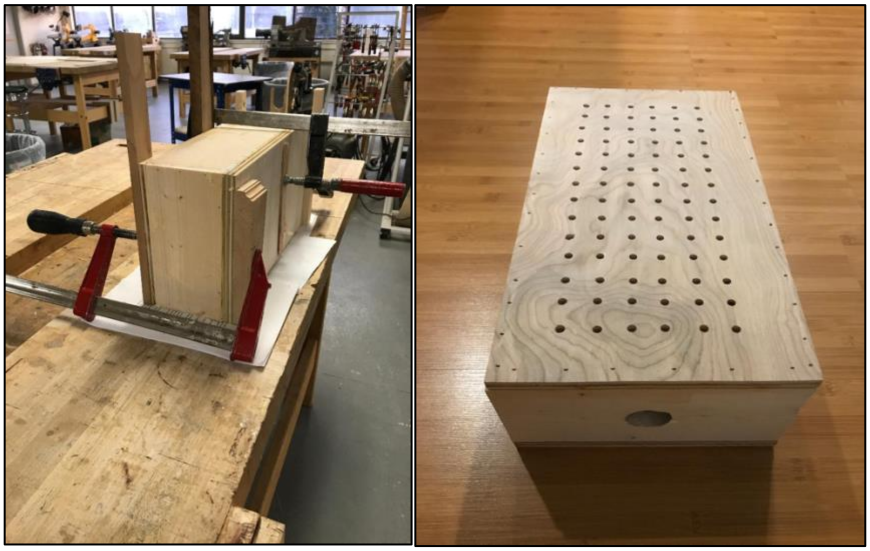


Figure 18: a) Left: Construction of vacuum forming machine and b) Right: Final

In addition to the box, we had to construct a double frame to hold the thermoplastic sheets inside during the heating and forming process. Again, we chose maple for durability and used the same procedure to join the sides with the Domino to ensure resistance to air pressure during the vacuum sealing process. The dominoes were reinforced with polyvinyl acetate wood glue, clamped and allowed to dry overnight.

The final step in finalizing the vacuum forming machine was to create a mold of the layout of our components. The mold was 3D printed based on the size and configuration of the electronics placed at the front of the board. Unfortunately, the printing of the mold could not be achieved without holes which created significantly difficulty in the vacuum forming process. As a result, the holes had to be covered and solid wood molds were created in the same shape to facilitate the thermoforming process.

We were fortunate to have several sheets of commercial grade polycarbonate donated to our project by Mayfield Plastics of Sutton, Massachusetts to be used in creating our custom cover. Polycarbonate is a difficult material to work with as it is sensitive to temperature and humidity. Before attempting to form the enclosure on the vacuum forming machine, each sheet had to first be dried in the oven to prepare it for the process. Although the polycarbonate sheets all were similar in appearance, the time required for drying each varied considerably. On average, the drying process required two hours of baking at 250 degrees Fahrenheit under careful scrutiny to avoid over drying and cracking. Each sheet had to be dried immediately prior to the thermoforming process and could not be done in advance. Sheets were individually placed in the wooden frame constructed for this process.

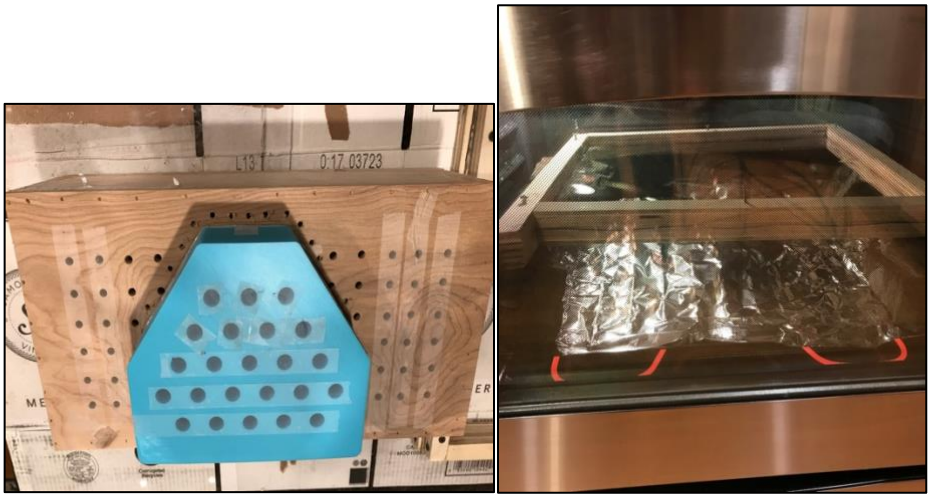


Figure 19: a) Left: 3D mold and b) Right: Frame view during process of oven

Once dried, each sheet was subsequently heated until the material achieved a softened and pliable state necessary for thermoforming. If the polycarbonate was heated for too long, bubbles formed, and it became stiff. Inadequate heating left the polycarbonate rigid and unable to form around the mold. After much trial and error, we determined that heating the polycarbonate to 350 degrees Fahrenheit for thirty-two minutes yielded the ideal conditions for creating a cover on our vacuum forming machine.

Once removed from the oven, however, the polycarbonate would seize up quite quickly and become totally unusable for our purposes. Having the box and mold in place with the vacuum attached reduced the risk of loss of use of each polycarbonate sheet.

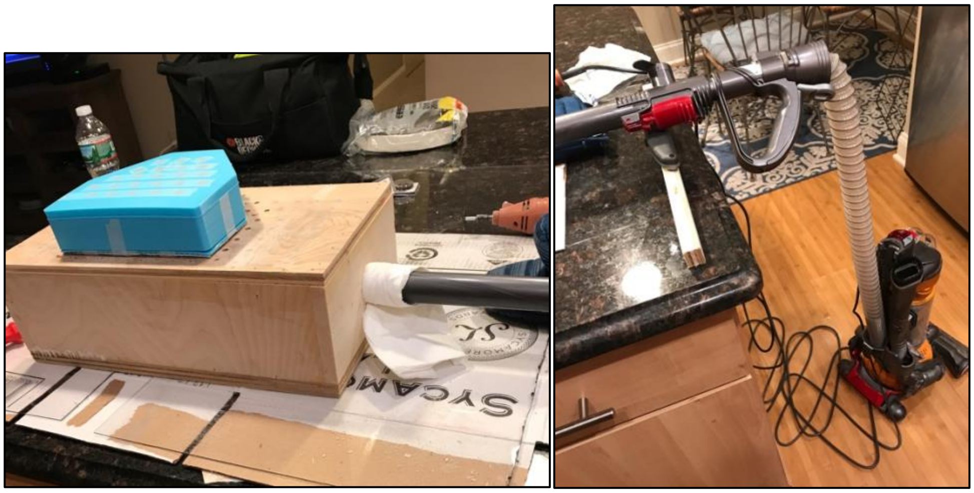


Figure 20: a) Left: Vacuum forming machine and b) Right: Attached vacuum

Several attempts were made to obtain a cover that was clear and free of bubbles and folds. Our final model was refined with the help of a heat gun used to smooth out the folds on the corners. After many attempts, we achieved a clear cover in the appropriate shape and size required for our purposes.

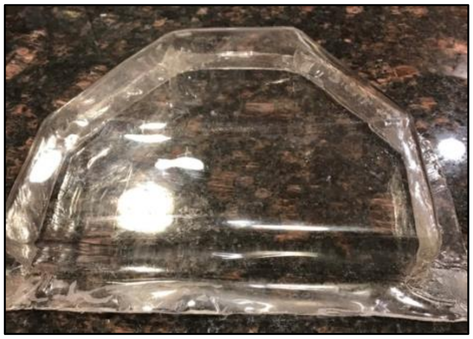


Figure 21: Final cover

Once the cover was ready, we needed to design a base with which the polycarbonate could be attached to the board. We determined that attaching the polycarbonate directly to the board would ultimately result in issues with possible breakage and damage to the board each time the fasteners needed to be undone. Realizing that the cover would likely need to be removed frequently to access the electronics, we needed to consider alternate ways of attaching to the board. Our main focus was to maintain the integrity of the board and cover while allowing accessibility to the components.

To resolve this issue, we designed a base out of one piece of solid maple for rigidity and durability. After the design and pattern was created, the base was cut on a band saw. Holes were drilled in both the base and polycarbonate cover. We subsequently permanently affixed the maple base to the board with screws and the polycarbonate to the base with wing nuts. This design allows for easy removal of the cover for access to the electronic components. A watertight seal was added around the perimeter of the cover to further protect the electronics from the elements during operation.

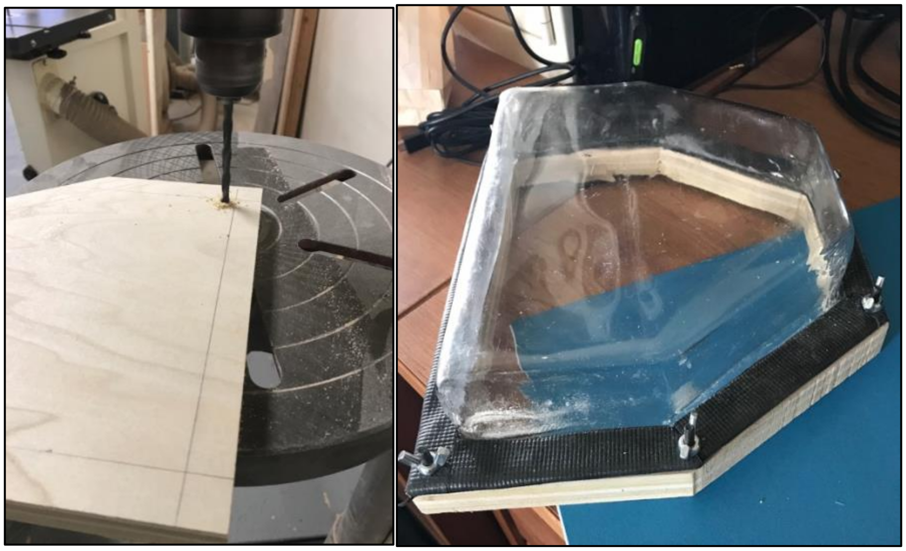


Figure 22: a) Left: Construction of cover base and b) Right: Final assembled cover

7.2 Drive System

7.2.1 Drivetrain Design and Assembly

The drive motion of the longboard is achieved by a high torque motor connected to one of the rear wheels by a belt drive. There are four parameters that primarily contribute towards the design of the drivetrain:

i) the gear ratio (m_G) - dictates the mechanical advantage at the wheel

ii) motor Kv - RPM per volt input (without load attached)

iiI) wheel diameter (d) - determines distance covered per revolution of wheel

iv) voltage input to the motor - power supply to the system

Combination of these parameters give us the desired top speed and acceleration. According to the task specification, the maximum speed is 10 mph. Using the parameters defined above, we can calculate the maximum speed, υ, output as follows:

$$Motor\ RPM\ =\ Motor\ Kv\ \times\ Voltage$$

$$\upsilon\ = Motor\ RPM\ \times d\ \times \Pi\ \times 1/m_G\ \times \varepsilon_{transmission}$$

In the above equation, ε is multiplied to factor in the estimated efficiency of the transmission. The drive train had been initially constrained by the wheel diameter size in order to ensure the steering motion does not cause collision of the wheels with the deck. Thus, the diameter was limited to below 3.5" which gives the necessary clearance between the deck and the wheels. Second parameter that partially constrained the design was the teeth ratio of the timing belt pulleys commonly available for purchase and were small enough to fit under the board. Taking these constraints into account, with some iterations the following were finalized: 83 mm wheel, 16 teeth motor pulley, 36 teeth wheel pulley, 260 Kv brushless DC motor. Estimating an efficiency factor of 0.7, the required 10 mph is achieved with a voltage supply of 13 V.

Another parameter considered in the design were center distances of the pulleys. An optimum distance of 65-70 mm was determined to maximize the number of teeth in mesh to prevent belt slippage and minimize force acting on the motor mount. Thus the optimal belt length was calculated to be 265 mm.

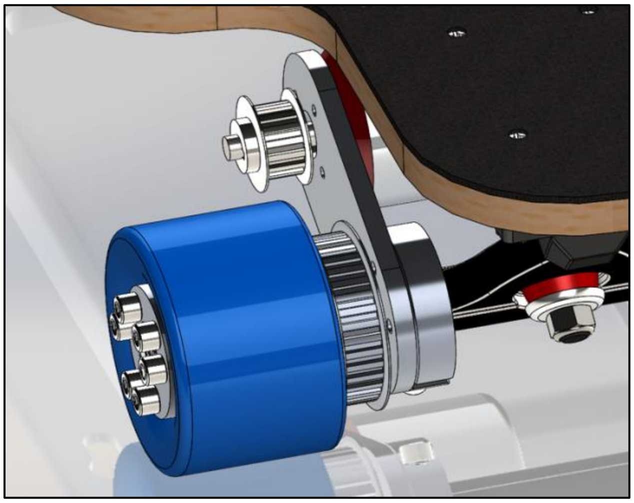


Figure 23: Basic Drivetrain setup without belt

Once the components were selected, we began the assembly by attaching the wheel pulley with the rear wheel. The aluminum pulley originally comes with tapped holes for fastening. It was securely attached to the wheel using 6 M5 bolts. However, first the polyurethane wheel had to be drilled in the same bolt pattern using a hand drill and 6mm drill bit for the bolts to go through.

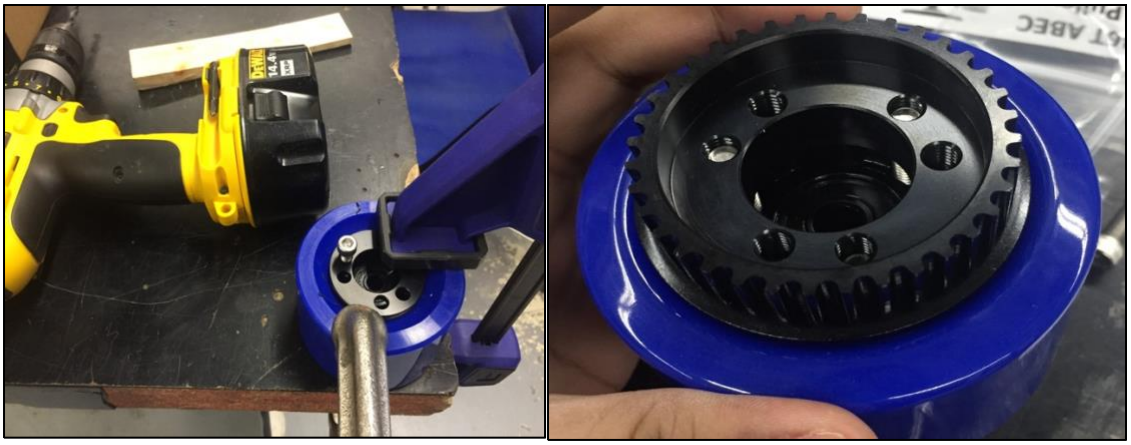


Figure 24: Wheel clamped for drilling bolt holes.

After attaching the pulley, the belt was put around the teeth, and the wheel was attached to the truck using a locknut. The motor pulley was secured on the motor shaft using a 3X3 mm key and two grub screws. Before tightening the screws, a small amount of Loctite was applied for thread locking to ensure robust assembly. The motor was mounted using four M4 bolts through counterbored slots on the motor mount. The slots allow sliding of the motor along the length of the mount in order to adjust the tension in the belt. Once the belt was taut, the set screws on the motor mount were tightened to complete the drivetrain assembly.

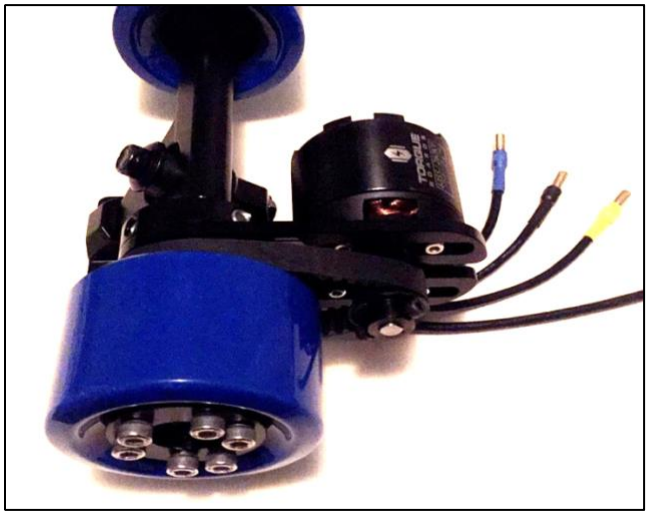


Figure 25: Assembled drivetrain on rear truck

7.2.2 Motor Mount

The motor was mounted on the truck using a 2-piece motor mount. The first piece is a collar around the truck and is fixed to the truck using 4 set screws radially. The second piece is a 6 mm thick plate that is bolted to the collar on the truck and on the other side contains counterbored slots to mount the motor using bolts. The 2-piece mount provides a rigid fixture preventing bending of the mount or relative motion between the components.

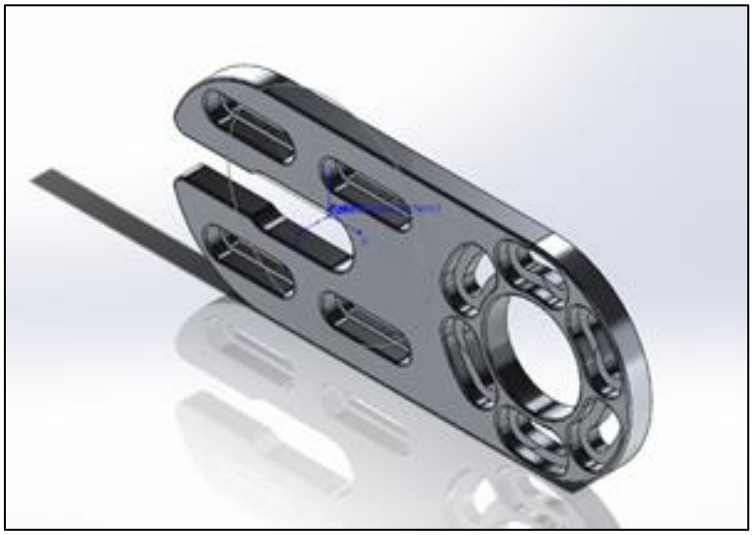


Figure 26: Motor mount SOLIDWORKS model

During our background research, we came across a number of reports that inadequately designed motor mounts undergo bending and warping. Thus, we decided to conduct a static analysis in SOLIDWORKS in order to simulate the forces the motor mount would withstand and the results thereof.

In this study, the material Al 6061 was assigned to the part. It has a yield strength of 270 MPa. It is assumed that the bolt connections between the mount and the collar around the truck create a fixed geometry on the faces highlighted in the figures below.

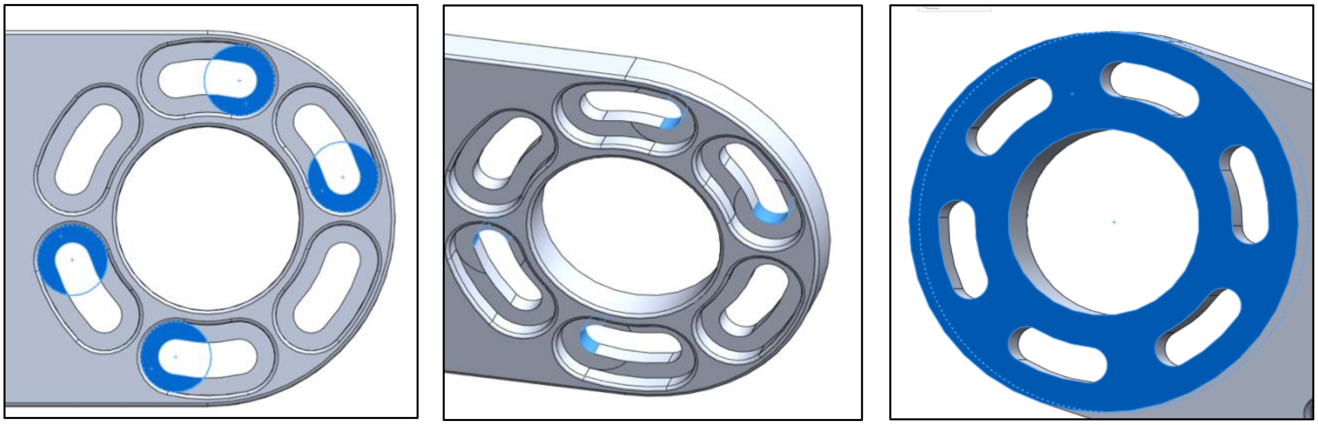


Figure 27: Faces used for fixed geometry constraint

The back side of the mount where the collar acts as support, is also assumed as fixed geometry.

Following the fixtures, the loads were assigned in 2 parts. First load simulates the moment created by the motor’s weight acting towards the end of the mount. Since the weight acts at a distance away from the longitudinal axis of the mount, the moment was calculated as:

$$M = W \times r = 10N \times 0.025 = 0.25\, N.m$$

Since this moment could not be directly applied in the software, we assumed a distributed loading on the footprint of the motor on the mount, equivalent to the moment about the longitudinal axis. The figure below shows the load distribution from the back and side view.

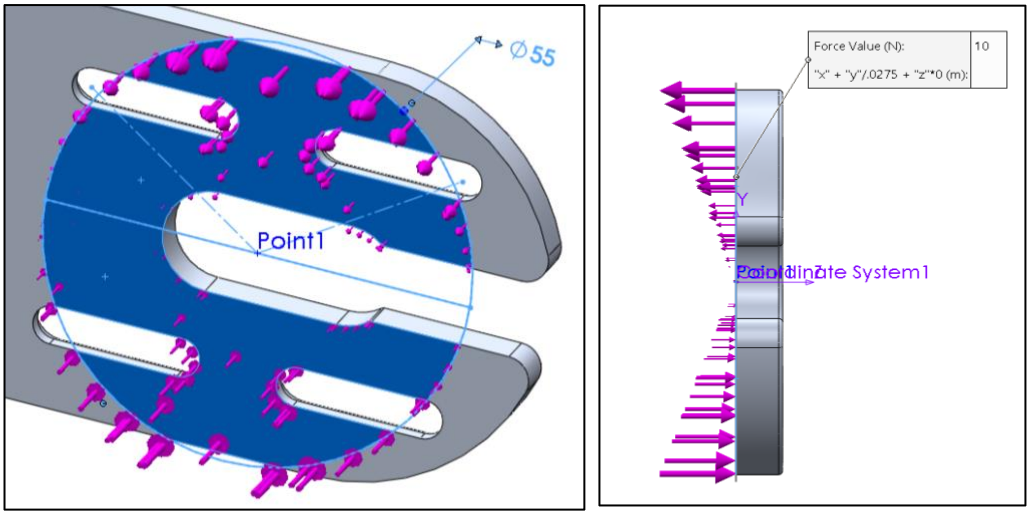


Figure 28: Modeling of the moment load created by motor weight

The second load simulates the motor torque. We assume the worst-case scenario and use the value of the motor stall torque for maximum loading. The 2.7 Nm torque is applied to the faces where the bolts connect the mount and motor, shown in figure below.

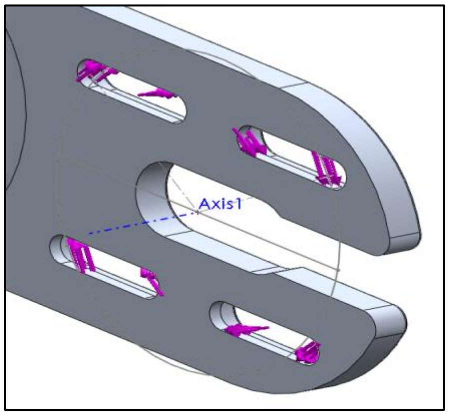


Figure 29: Motor torque load

With these fixtures and loading, we ran our study and obtained the following results. The image below shows the stress distribution and an over-scaled deformation. The red zones show maximum stress occurring where the bolts are connected in the slot. The von Mises stress in that area is 45 MPa, which sums up to a safety factor of 1.2. Since we already assume the worst-case scenario, the small safety factor was deemed acceptable.

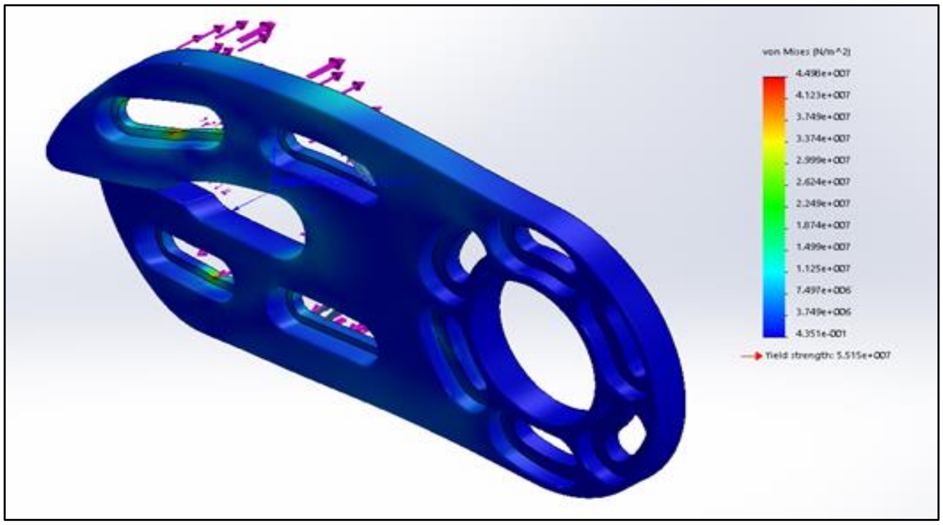


Figure 30: Stress distribution on the mount

7.3 Steering System

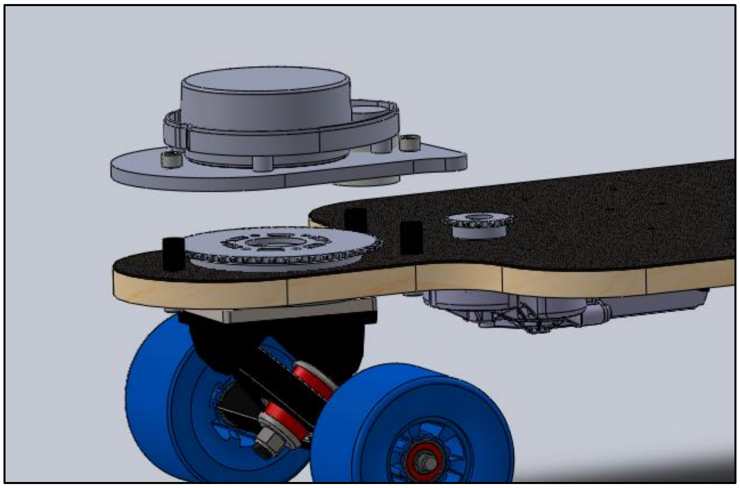


Figure 31: Preliminary model for steering

Steering will be handled by one high-power servo motor driving a 16-tooth sprocket, via chain, to a 48 tooth sprocket that is attached to the front truck, thereby enabling steering of the board. The servo motor we are using is a Bosch seat motor commonly used for raising and lowering powered seats, generously donated by andymark.com (Figure 32, below).

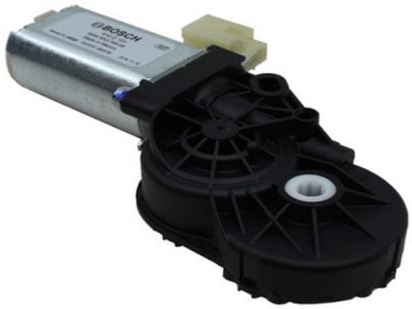


Figure 32: Bosch seat motor servo for steering

The rated stall torque of 22 Nm with a 3:1 reduction, assuming a transmission efficiency of 90%, provides a maximum output torque of

$$\text{Tout} = (22\text{ Nm})(48/16)(.9) = 59.4\text{ Nm}$$

to the front truck assembly, in order to allow turning of the wheelbase and thus, steering. The wheelbase has a diameter of 180 mm, thereby applying tangential forces of

$$((59.4\text{ Nm})/2)/((.180\text{ m})/2) = 330\text{ N}$$

to each of the wheels, which was deemed sufficient to allow turning under regular use.

The top assembly plate holding the sprocket is bolted to the lower plate via four ½” bolts, and rests on an off the shelf thrust bearing in order to minimize friction between the mating surfaces (Figure 33, below). The thrust bearing is secured into the cutout in the board by a pair of counterbored plates, with the mating surfaces resting on it, with the aforementioned bolts passed through the central opening. The lower assembly is itself bolted to the truck mounting plate by an additional four ½” bolts, thus completing force transfer in a fairly efficient manner.

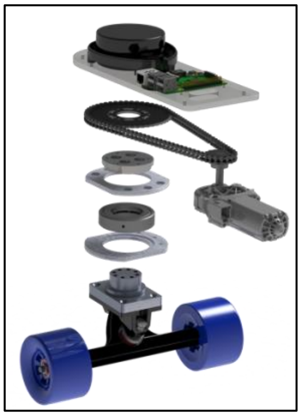


Figure 33: Steering assembly view

This design was chosen due to its relative simplicity, and the inexpensive (in terms of direct, budgeted costs) process of milling extruded aluminum bar stock into the desired form using the university's machining resources. Of course, in production the machining process would actually represent one instance of an additional cost over the prototype, since the machining equipment and time could no longer be provided for free by the university and MQP students, respectively.

All of the custom-made components of the board were first designed on SOLIDWORKS and then run through simulations. Once the parts were made we were able to test different materials to see which would not fail under the constraints we had. At first we decided on steel which passed the simulation tests and seemed like the best choice. Once we started machining the parts and assembling the board we realized that steel was too heavy to use so we went back to the SOLIDWORKS models and tested the components using 6061 Aluminum. After analyzing each component, a second time we determined that 6061 Al is a more suitable material. We came to this conclusion but figuring out that the component with the greatest stresses on the part "adaptor_2".

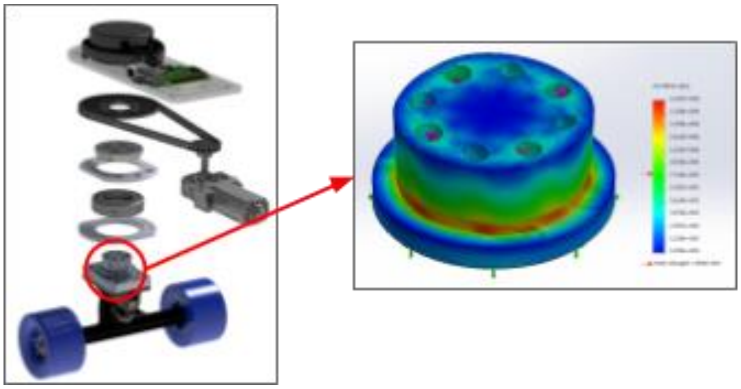


Figure 34: Adaptor 2

For the adaptor_2 component we applied a force of 200 lb acting down on the board. The adaptor_2 component is attached to the sprocket using four of the eight holes on the part. Since only four of the holes are being used we applied a disturbed torque of 380 ft.lb across the alternating holes. Using a tensile strength of 310 MPa for 6061 Aluminum the simulation provided a safety factor of 5.3 which deemed reasonable for this component. Since the component with the greatest stresses acting on it was efficient we concluded that all the components would work. We then machined the parts out of aluminum and installed the components. Once the components were installed we stood on the board to and tested with various weights and none of the Aluminum components fractured. We also tested the Aluminum parts as a steering assembly and the wheels turned with minimal friction resulting in smooth data. The rest of the analyses of the other components can be seen in the Appendix D.

7.4 Structural Analysis of Steering Assembly

Assume a 200 lb person is traveling along at a leisurely pace of 1.4 m/s (normal walking speed for an average person). They hit a large buried rock, or the motor seized up, or anything happens to stop them dead. The handlebars stop their forward momentum in 0.1s (quick arrest being vital for the disabled rider to have any chance of maintaining balance).

$$F = \frac{J}{t} = \frac{(90.72\,kg)(1.4\,m/s)}{0.1\,s} = 1270\,N$$

average force is exerted by handlebars on rider. Assuming the handle is a three-foot lever arm, torque is applied on the base plate.

$$(1270\,N)(0.9144\,m) \;=\; 1161\,N \cdot m$$

Assuming a brand-new assembly with no wear, defects, or loose fasteners, assuming the steel bolts do not cause the weaker aluminum to yield at all, either through shearing off the threads or pulling through the 0.08” (< 1.5*diameter = 0.195”) of material between the edge of the bolt and edge of the part on the bottom aspect, and assuming 80% of the counter torque is taken up by the bearing surface (design RoT),

$$\frac{(1161\,N \cdot m)(0.20)}{0.01194m} = 9{,}728\,N$$

tension is applied through the single loaded M4 bolt in the baseplate to supply the remaining counter torque.

The minimum ultimate tensile load of M4x0.7 grade 4.6 bolt (per AMES calculator) is 3,510 N << 9,728 N. Assuming the bolts were grade 12.9 and the handle was turned perfectly so that two bolts were engaged evenly, so in this case the bolts would not fail given these assumptions.

$\frac{9728\,N}{2} = 4{,}864\,N \;<\; 10{,}700\,N$(MUTL for grade 12.9 M4X0.7 bolts),

Realistically, under those conditions the mode of failure is more likely to be tear out (or fracture) through the edge of the plate or by shearing through the (aluminum) female threads, since the application of load over time will place significant stress on the hole edges, which are a scant 0.08” (0.61*hole diameter) from the edge of the plate on the inferior aspect, and four of the six bolts are 0.117” (0.9*hole diameter, see Figure 34) from the edge of the large bolt counterbores on the superior aspect. These stresses are apt to result in small deformations, particularly in light of the grain structure of the extruded bar stock from which the part was turned, the play from which will make these sorts of failure even more likely.

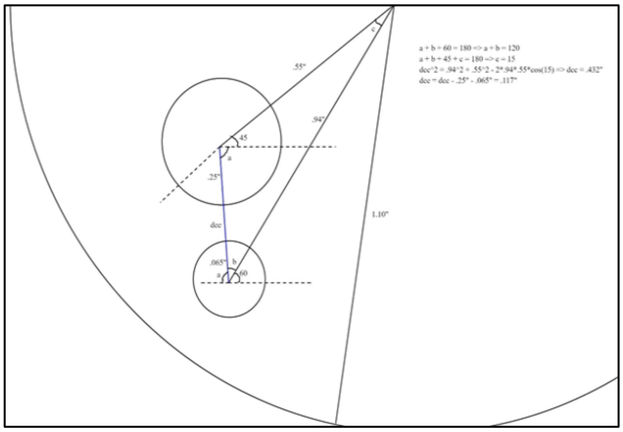

c
.55"
.94"
45
a
.25"
dee
.065"
b
a
60
1.10"
a + b + 60 = 180 => a + b = 120
a + b + 45 + c = 180 => c = 15
dcc^2 = .94^2 + .55^2 - 2*.94*.55*cos(15) => dcc = .432"
dee = dcc - .25" - .065" = .117"

Figure 35: Base plate hole dimension analysis

Assuming the bolt is threaded and engaged for the entire depth of the part, and again assuming brand new parts with no creep and none of the above-described non-idealities we have

$$(0.5)(\pi)(10.668\,mm)(4\,mm-(0.64952thr)(\frac{1}{0.7\;thr/mm})=51.48\,mm^{2}$$

effective thread shear area. Given the above-described scenario,

$$\frac{9728\,N}{51.48\cdot 10^{-6}\,m^{2}}=189\,MPa\;<207\,MPa$$ (typical shear strength of 6061 aluminum).

Now, assuming that no shearing of the threads occurs, tear out through the plate can be considered. If an M4 bolt is torqued in such a manner as to bear toward the counterbore of the large bolt on the superior face, and the plate edge on the inferior edge, it is therefore relying on .08” of aluminum (0.61*bolt diameter) on the inferior face and .117” (0.9*bolt diameter) on the superior face to retain it. To oversimplify a complex situation, considering this scenario as a simple shear tearout through the strongest aspect,

$$F_{fail}=(30000\,psi)(2)(0.117"+0.065")(0.20")\;=\;2{,}184\,lbf\;=9{,}714\,N,$$

somewhat greater than the order of what has been encountered in the other settings. Determining a value for the force applied is difficult given all that is going on, but failure here also seems unlikely, despite the grain structure of extruded aluminum, which would make the plate more prone to fracture along that grain.

Finally, turning the handle against an immobilized wheelbase would result in single shear on the six bolts. For grade 4.6 bolts,

$$F_{fail} = (6)(USS)(BSA) = (6)(310\ MPa)(8.78\ mm^{2}) = 16{,}330\ N.$$

This shear force would result from an axial torque of

$$\tau_{fail} = (16330\ N)(0.0238\ m) = 388.6\ N \cdot m.$$

Assuming the handlebars are one foot wide from grip to grip, an applied torsional force of

$$F_{bar} = \frac{388.6\ N{\cdot}m}{0.1524\ m} = 2{,}550\ N$$

at the handlebars would be sufficient to shear the M4 bolts off, so were the stop to occur in such a way that the wheels were immobilized, and one side of the handlebars took the brunt of the force, failure is somewhat likely.

Analysis: given the multiple possible modes of failure, securing the joint with the weaker fasteners may be necessary in order to act as a mechanical fuse. Grade 4.6 bolts provide the lowest bar to failure in all areas, and therefore would serve this purpose of failing without destroying the assembly. Notably, assumptions had to be made regarding the stopping time; no consistent and reliable information could be located on reasonable values, and we lacked the equipment to test this empirically. This means that, depending on the stopping time, the calculated stresses experienced could be higher or lower. Due to this, more rigorous testing will have to be implemented prior to release of the project as a product, and a software-based speed control to limit the forces experienced will have to be implemented to ensure safe operation of the device.

The remaining parts have the same bearing surface area and the same number of the same-sized fasteners resisting the same loads as the first part in compression and tension, and thus should yield the same (well within any reasonable margin of safety) results in those aspects, so we will forego repeating those calculations.

For the adaptor, tearout through the "plate" edge, assuming the minimum diameter throughout, much higher than is likely.

$$F_{fail} = (30000\ psi)(2)(0.10" + 0.10")(0.85") = 10{,}200\ lbf = 45{,}371\ N,$$

For all of the parts secured for torsion by four ¼” bolts, assuming the single-shear characteristics are the same, and the bolts are grade 2, if the bearing part is immobilized and a force is applied to the part in question, produces single shear failure of the four bolts.

$$F_{fail} = (USS)(4)(BSA) = (45000\,psi)(4)(0.04908\,in^{2}) = 8{,}716\,lbf = 38{,}770\,N$$

This corresponds to an axial failure torque of much greater than that of the M4 bolts.

$$\tau_{fail} = (F)(r) = (38770\,N)(0.01397\,m) = 541.6\,N \cdot m,$$

Analysis: the remaining parts, all essentially thick plates with large bearing surfaces, taking minimal load and secured by four large fasteners, are not likely points of failure for this assembly.

7.5 Power Distribution

The board components are all powered by a 6S2P (6 series, 2 parallel) 18650 (Samsung 20R cells) battery pack, sourced from diyelectricskateboard.com and designed for this specific purpose. This particular pack contains both the standard 6S (6*3.7 V = 22.4 V) output, as well as two USB ports providing standard USB 5V to power a Raspberry Pi and an Arduino, as well as an integrated battery management system designed for motorized skateboards such as this one. Power was drawn from the battery via all three routes, as seen below in Figure 35.

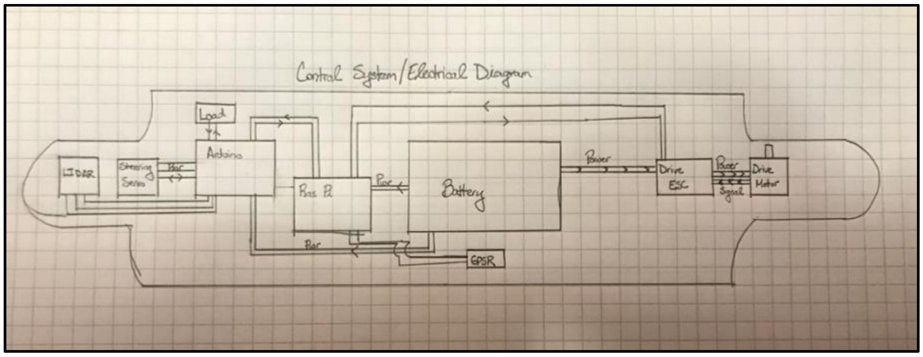


Figure 36: Electrical Diagram

The primary (6S) power output was routed to the electronic speed controller (ESC) for the motor, which in turn managed the power supplied to the drive motor, and thus ultimately the drive speed. The control electronics, specifically the Raspberry Pi and Arduino, were powered by the two USB ports, as per the designs on diyelectricskateboard.com. The steering servo motor was powered via a Pulse-Width Modulation (PWM) output from the Arduino; there was initially concern that the current supplied by the Arduino might not be sufficient to provide adequate torque for this purpose, but that does not appear to have been an issue with appropriate mechanical advantage from the chain and sprocket described elsewhere.

Using the provided 2A charger, the skateboard battery was able to be charged in 2-3 hours, again per the given specifications, thus fulfilling the secondary specification TSS04. This could be further improved by purchasing the 4A fast charger also offered by diyelectricskateboard.com, though it was ultimately decided to forego this for budgetary reasons, particularly as the specification had already been achieved.

The 6S2P 18650 pack provides 4000 mAh rated, for 25.2 V * 4 Ah = 100.8 Wh rated power supplied, while typical power consumption from the ESC under typical conditions was found to be 13V. From our experiments the battery last 2 hours at 50% idle time.

7.6 Thermal Design

The inside of the electronics enclosure contains many heat-sensitive components and heat generating components. The steering motor controller is the largest heat generator. As the steering motor has a worm gearbox it is <20% efficient. This lack of efficiency causes the motor and motor controller to give off the excess energy as heat. We ran a simulation in SOLIDWORKS to estimate the heat generation and temperature inside the enclosure.

To validate the study, we placed temperature sensors inside the enclosure at the points specified in figure 36.

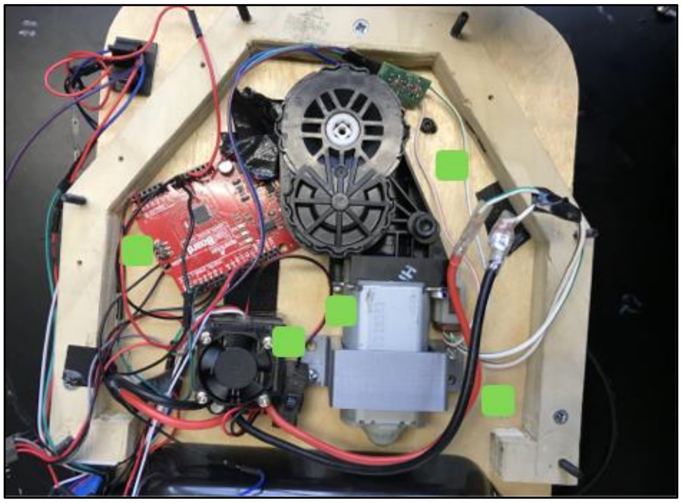


Figure 37: Temperature sensor points are marked with green squares.

First, the board was switched on and steering was run intermittently for 10 minutes without a fan on the motor controller. The temperature was recorded with an Arduino Uno and all components were allowed time to cool. The fan was then mounted onto the motor controller and the test was re-run. The results from these tests are shown in figures X and Y.

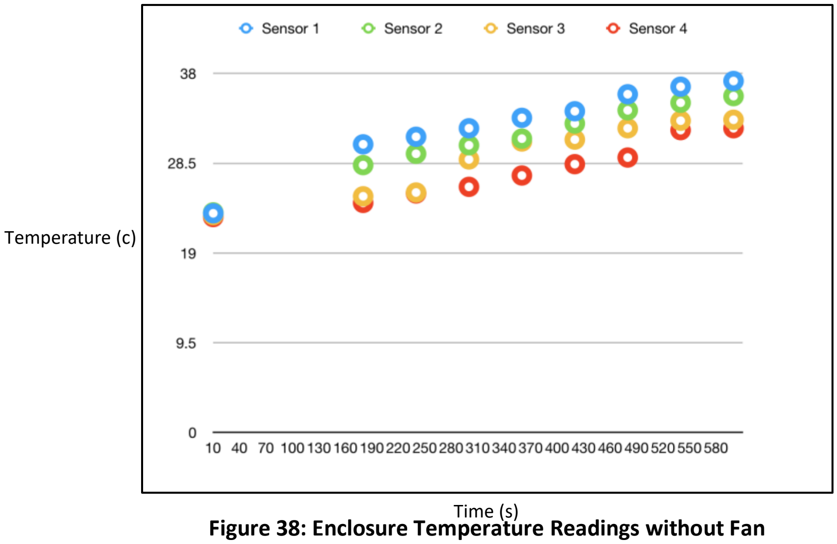


Figure 38: Enclosure Temperature Readings without Fan

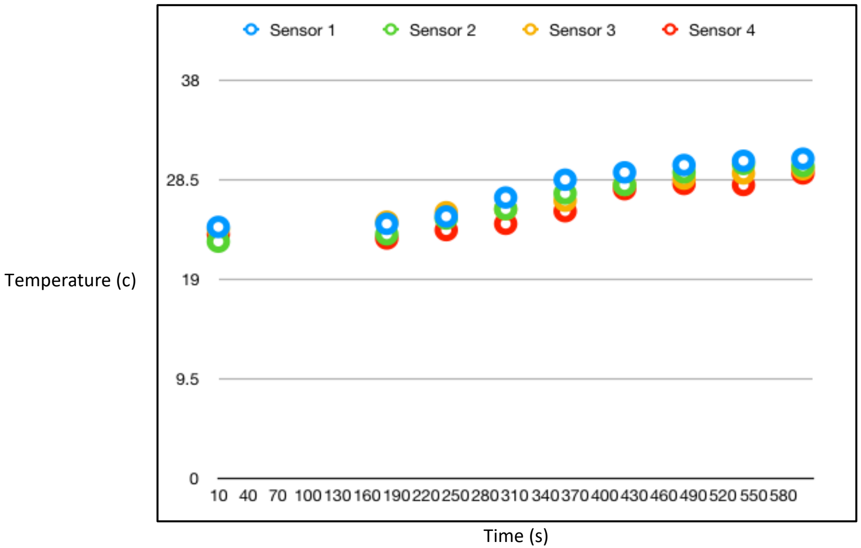


Figure 39: Enclosure Temperature readings with Fan

The normal operating temperature of the enclosure drops approximately 5 degrees with the addition of a fan on the motor controller.

8. Sensing and Control

8.1 Sensors

The sensors used on this robot serve two primary purpose- robot localization and obstacle detection. In order to aid localization of the robot in the global coordinate frame, we have used a Global Positioning System (GPS) module and a magnetometer as a digital compass. Both of these sensors are part of the BerryGPS-IMU v2 board which also includes an accelerometer and a gyroscope as an inertial measurement unit (IMU). For obstacle detection, we have used XV-11 LIDAR. These components can be seen in Figure 40, below.

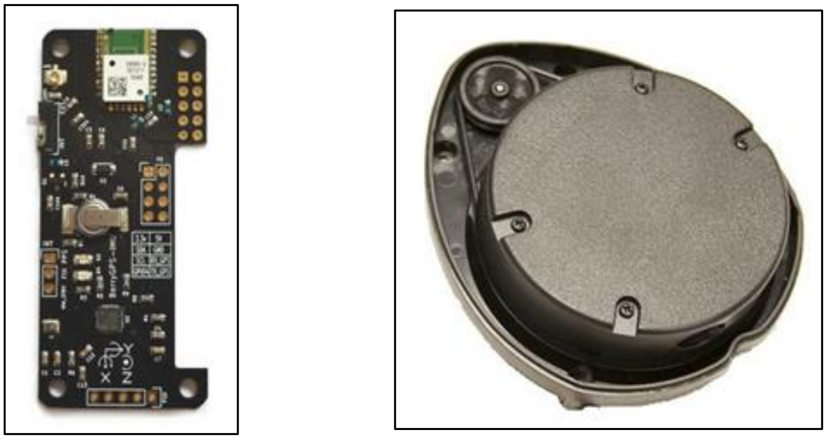
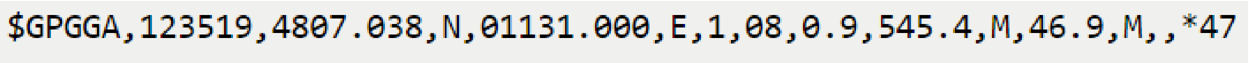


Figure 40: BerryGPS-IMU v2 (left), XV-11 LIDAR (right)

8.1.1 Global Positioning System (GPS)

For localization and producing reliable position estimates, we used the GPS module M10478-A2 on the BerryGPS-IMU v2 board. It has an accuracy of +/- 2.5m (from datasheet). The module outputs data in the form of National Marine Electronics Association (NMEA) sentences, a particular format used by most GPS devices. All relevant data, such as latitude and longitude were extracted using a python script on the Raspberry Pi. The image below shows an example of output data from the GPS sensor.



```
$GPGGA,123519,4807.038,N,01131.000,E,1,08,0.9,545.4,M,46.9,M,,*47
```

Figure 41: Raw GPS data format

From the image above, the second comma-separated number is the latitude (48° 07.038' N), and third is longitude (-11° 31.000' E).

8.1.2 Digital Compass

In order to measure the heading of the robot, we used the magnetometer as a digital compass. A magnetometer is a micro-electromechanical system (MEMS) instrument for measuring the strength and direction of an ambient magnetic field— commonly the Earth's, in units of Tesla. The output from the sensor are signed values for the magnetic field in x, y and z-axis. The heading angle with respect to magnetic north can be calculated mathematically by using the calibrated readings of X and Y components (see Appendix A for magnetometer calibration).

$$heading = \tan^{-1}\frac{magY}{magX}$$

The compass heading ranges from 0 to 360 degrees, so a negative angle is added by 360. Furthermore, since the magnetic north is at an angle from true north, an offset, called magnetic declination, needs to be applied to this angle obtained. This offset varies based on the geographical location and also fluctuates slightly over time. The magnetic declination offset of where this project was carried out was approximately -14°, obtained from the website: http://www.magnetic-declination.com.

8.1.3 LIDAR

The LIDAR sensor we chose is the sensor from a Neato XV-11 vacuum robot. We selected this specific LIDAR because it easily interfaces with our existing system and there is good existing documentation on how to work with and interface it with Arduino and Raspberry Pi. It is also relatively low-cost compared to other options available. The LIDAR from the XV-11 robot also provides 360-degree scanning. The LIDAR is not used for any sort of mapping or localization; instead, it is used only for obstacle/sidewalk detection. The idea was that we will be able to "bound" where the sidewalk is and control the robot such that it maintains its relative position on the sidewalk except in the event of an obstacle that needs to be avoided. The LIDAR is controlled by a Teensy 2.0 control board. The board provides the LIDAR with power for the motor and uses I2C communication to send and receive data with the sensor. With our limited implementation, the LIDAR is used to avoid oncoming obstacles. The sidewalk positioning could be potential future work.

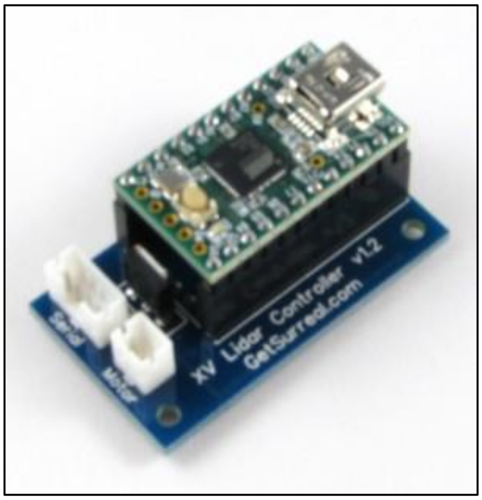


Figure 42: XV-11 LIDAR Controller

The Teensy communicates with the Raspberry Pi via serial communication. Sending alerts when objects come into the “detection range” of the board.

To calibrate the sensor and ensure that it was reading correctly, it was set up on the table with objects a set distance around it at 8 points in a circle. Each object was initially placed approximately 200 mm away from the sensor and moved back 100 mm from the original position during the duration of the test. This was done to ensure correct dimensional readings as well as the ability to track a moving object over time. The results of the test are shown in Figure 41 below.

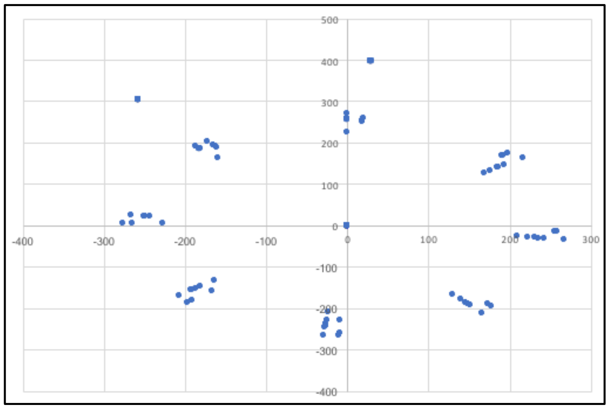


Figure 43: LIDAR Calibration Visual

8.2 System Architecture

The main CPU of the robot is a Raspberry Pi 3. It runs the main navigation code and communicates with all of the other sensors and sub processors. For navigation, the Raspberry Pi communicates with the GPS and IMU via I2C. For steering, the Raspberry Pi sends data to an Arduino Uno. The Uno then sends the correct PWM signal to the steering motor controller.

For Object detection and avoidance, the XV-11 LIDAR is controlled by a Teensy 2.0 control board. This interacts with the Raspberry Pi via USB. The next sections will go into further detail about the interactions between the sensors.

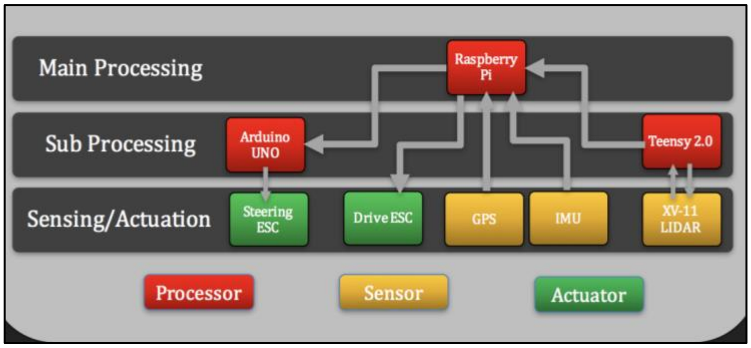


Figure 44: System Architecture showing communication between processors, sensors and ESCs.

It is important to note that the Raspberry Pi I/O operates at a 3.3V level while the Arduino Uno uses a nominal 5V level. This makes the two processors incompatible without a level shifter. The data sheet for the level shifter we used is included in Appendix X.

8.3 Navigation and Control

In order to achieve autonomous navigation and control, we programmed the robot to localize itself in the global coordinate frame, compare its location and heading to a target location and desired heading, and execute the necessary motor functions required to move to the target. We used Python 2.7 to write most of the program used for navigation and control.

8.3.1 Program Logic

It is important to note that the robot does not do path planning itself. The desired route is broken down into a list of waypoints pre-assigned to the program. The program initializes by checking the robot’s current position using GPS. We assigned a threshold such that the program will only continue if the current position is within 2 meters of the starting point in the route. The routine executed after initialization can be outlined by the following steps:

Once the program enters the main loop, it fetches the immediate next waypoint in the list as the target. For the target waypoint alone, the program enters another loop which executes tasks to reach the waypoint.

A state estimate is then generated, allowing the robot to localize itself with information about its position, velocity and heading. Since the position estimate obtained by Odometry is susceptible to accumulation of error, we implemented a Kalman filter (detailed below) for sensor fusion of the GPS data. The GPS measurement combined with those from Odometry generates a more statistically reliable estimate of the robot's location.

The position is in Cartesian form with X and Y coordinates corresponding to East and North, respectively, with the origin at the start location. With the position known, the straight line or line of sight (LOS) distance to the target waypoint (also in Cartesian form) is calculated by simple distance formula.

$$r = \sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}$$

This distance is compared against a threshold of 1m to check if the robot is close enough to consider target waypoint reached or not (which it isn't the first time loop is executed). This is what tells the robot to start or continue driving.

For course adjustment, the robot compares the current heading, obtained from the IMU module, with the desired heading. Desired heading is the angle with respect to North, calculated using simple trigonometry based on current position and target position. The difference in the two angles is the adjustment required and can be positive or negative. If the course adjustment required is over a certain threshold (in our case, 5 degrees) then steering is done for course correction.

The new state is computed again to check how far the robot has travelled and the loop keeps going until the target waypoint is reached. Once the waypoint is reached, the program exits the inner loop and starts back in the main loop which gets a new waypoint if any is remaining. If all waypoints have been reached, the robot has reached its destination and the program stops.

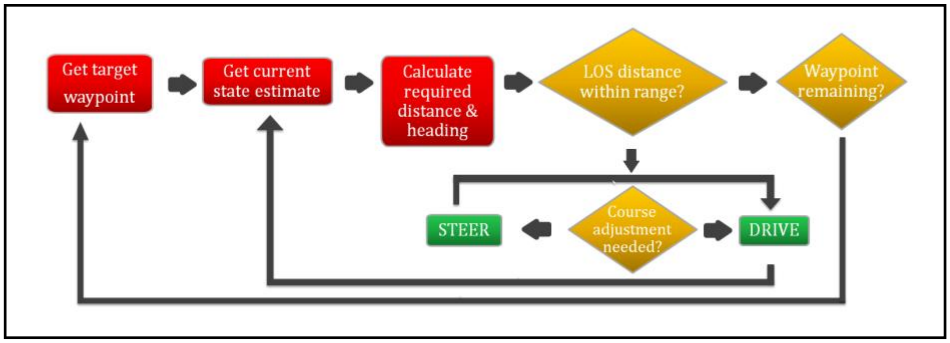

Get target waypoint
Get current state estimate
Calculate required distance & heading
LOS distance within range?
Waypoint remaining?
STEER
Course adjustment needed?
DRIVE

Figure 45: Flowchart summarizing navigation logic.

8.3.2 Locomotion and Odometry

As mentioned in section (8.2), System Architecture, the drive function is controlled directly by the Raspberry Pi (RPi), whereas the steering is done by the Arduino Uno. However, the computation that results in the output for both is performed and delegated by the RPi.

The drive module on the RPi is python script that controls the duty cycle output to the drive motor via the ESC. We mapped the range of the drive duty cycle (driveDC) to a scale of 5 – 10. We then mapped the rpm range of the motor using this scale by a linear regression. This was done by monitoring the rpm using VESC tool (see Appendix B) at different points on the scale. While theoretically the midpoint should be 0 rpm, 7.0 to 7.2 was actually dead band. The maximum value, 10, resulted in approximately 5000rpm of the motor and theoretical wheel rpm of 2222 (16:36 gear ratio). We used the range of theoretical wheel rpm to compute robot speed (V) and distance travelled by using a kinematics model (detailed below). In order to improve the accuracy of the model and account for losses, we performed several tests to find a suitable scale factor that result in output of the model to be very close to the measurements.

$$RPM_{motor} = \frac{(driveDC - driveDC_{min})}{driveDC_{range}} \times RPM_{motor\_max}$$

$$V = sf \times gr \times \frac{(RPM_{motor} \times \pi \times d_{wheel})}{60}$$

The steering motor is also controlled by PWM signal from the Arduino. Unlike the drive module, steering is done with only three values corresponding to stationary, clockwise (cw) and counter-clockwise (ccw) rotation of the front truck. The PWM values were mapped to +/- 3 degree measured through experimentation. Based on the state at a given time, the corresponding input angle (δf) was used in the kinematics model along with the drive module parameters. At any point during the program, the state of the steering is chosen based on digital output from the RPi monitored by the Arduino. The logic used based on the output from RPI is shown in the state table below:

RPi Pin 23	RPi Pin 24	Steering
0	0	Stationary
0	1	CCW
1	0	CW
1	1	Stationary

Using the aforementioned control inputs, V and δf, the bicycle kinematics model (Polack, Altché, d'Andréa-Novel & de La Fortelle, 2017) was implemented for the robot locomotion. In this model, as in a bicycle, the four wheels of the board are grouped as one front and one rear wheel on the longitudinal axis. Besides the simplicity of the model, it has an added advantage of bearing resemblance with the steering system on the longboard since the entire front truck is allowed to rotate unlike an Ackermann steering. The model can be described as follows:

$$V_x = V\cos(\Psi + \beta(\delta_f))$$

$$V_y = V\sin(\Psi + \beta(\delta_f))$$

$$\Psi = \frac{V}{l_r}\sin(\beta(\delta_f)) \times \delta t$$

$$\beta(\delta_f) = \tan^{-1}(\tan(\delta_f) \times \frac{l_r}{l_r + l_f})$$

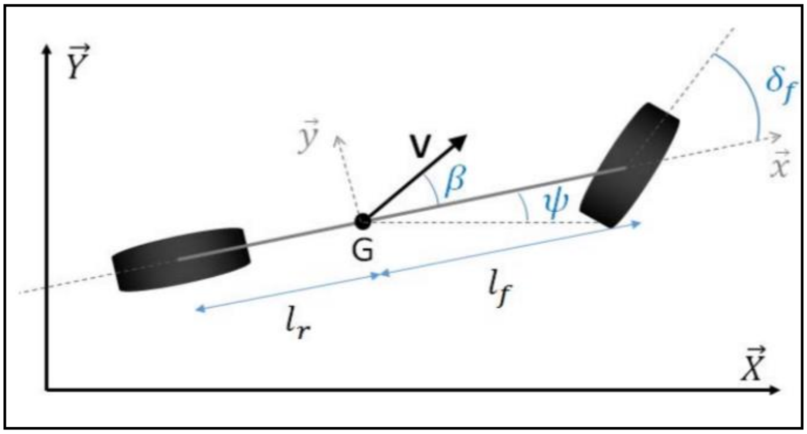


Figure 46: Kinematic bicycle model for the longboard adopted from Polack et al., 2017.

In the description above, V_x and V_y represent velocity components in the x and y components of velocity in the inertial frame. δ_f, β, and Ψ represent the steering angle, slip angle at center of gravity (G), and yaw, respectively. The parameters l_r and l_f are distances from G to rear truck and front truck, respectively.

It should be noted that the velocity components derived from the kinematics model is in the inertial frame. Hence, it is necessary to perform a rotation of these vectors to the absolute coordinate frame. This is done by simple matrix multiplication using the initial heading at the beginning of the robot's journey.

$$\begin{bmatrix} V_E \\ V_N \end{bmatrix} = \begin{bmatrix} \sin\theta & -\cos\theta \\ \cos\theta & \sin\theta \end{bmatrix} \begin{bmatrix} V_x \\ V_y \end{bmatrix}$$

The instantaneous velocity components are integrated to compute position in Cartesian coordinates in the inertial

$$x_t = x_{t-1} + (V_E \times dt)$$

$$y_t = y_{t-1} + (V_N \times dt)$$

8.3.4 State estimation using Kalman Filter

The Kalman filter is one of the most commonly used data fusion algorithms that is used as an optimal estimator for linear systems. It is an iterative mathematical process used to estimate a system's state, such as position and velocity, when an accurate measurement is not available. The measurements are "fused" with an initial guess of the state produced by a mathematical/theoretical model (in our case, the bicycle kinematics model). The computation used in a Kalman filter is capable of quickly estimating true value, even when sensor measurements have random error or variation. It does so by taking into account the uncertainty of sensor measurements and that of the process itself. In other words, it factors in the process variance and noise of the measurement data. Another advantage of using the filter is that, data from each iteration is not required to be stored to compute the estimate. The filter only uses information from the previous state to produce the current state estimate (Kim, 2017).

Prediction Step:

$$\bar{x}_t = Ax_{t-1} + Bu \qquad A = \begin{bmatrix} 1 & dt \\ 0 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} \frac{1}{2}dt^2 \\ dt \end{bmatrix}$$

Generally, the equation is used for state prediction, where A is the state transition matrix, B is the control matrix and "u" is the acceleration vector. However, since we mapped the velocities directly using the duty cycle, we can drop the control terms and use our estimates from the kinematics model. This equation is just a combined matrix representation of:

$$x_t = x_{t-1} + (u \times dt) + \frac{1}{2} a\, dt^2$$

$$v_t = v_{t-1} + a \times dt$$

Based on the kinematics model elaborated in the section above, the filter produces an initial guess of the position and velocity. Because a state has two parameters of interest, we represented them as vectors and used matrices for efficient computation. The vector $\bar{x}_t$ (not to be confused with coordinate axis "x") represents the predicted state at time t in either x or y direction.

$$\bar{x}_t = \begin{bmatrix} pos \\ vel \end{bmatrix}$$

$$Q = \begin{bmatrix} sd^2 & 0 \\ 0 & sd^2 \end{bmatrix}$$

$$\bar{P}_t = AP_{t-1}A^T + Q$$

$\bar{P}_t$ is the predicted covariance, P_{t-1} is the actual covariance from last step, A^T is A transposed, Q is the process covariance matrix which is a diagonal matrix we constructed using our experimentally determined standard deviation of the robot's position from mathematical model value.

Kalman Gain:

$$K = \frac{\bar{P}_t C^T}{C\bar{P}_t C^T + R}$$

K is the kalman gain calculated using C which is the sensor (GPS) model matrix, in our case $[1 \quad 0]$. R is the sensor measurement variance. The standard deviation (2.5m) for this was directly used from the datasheet.

Update:

$$x_t = \bar{x}_t + K(Z_t - C\bar{x}_t)$$

$$P_t = (I - KC)\bar{P}_t$$

This is the step where the program takes the measurement from GPS and refines our estimated position and covariance. Z_t, unlike x_t is not a vector. It only has position estimate, since we did not measure velocity from GPS. The difference in forms of Z_t and x_t does not affect our computation because our sensor model matrix $[1 \quad 0]$ removes the velocity estimates. This was necessary because we were unable to extract reliable velocity estimates from the sensor.

The position values we used for Z_t were in Cartesian form. Using the Haversine formula (see Appendix C), with the latitude and longitude obtained at a given position during the robot's journey and those at starting point, the program calculates the distance between them. We used the same set of coordinates to calculate the bearing between them, which paired with the distance, allows projection of the spherical coordinates onto a 2D Cartesian coordinate system.

8.4 Obstacle Avoidance

In order to avoid obstacles, the robot is equipped with a LIDAR and corresponding control board. The sensor scans 360 degrees. The control board filters out all information except for +/- 45 degrees from the centerline of the robot. The distance from the LIDAR to the nearest obstacle at a given heading is reported by the sensor to the control board. With an array of angles and their corresponding distances, a "map" is formed of all objects with the target distance of the robot. Next, the Teensy processes and determines which objects are within a certain "cutoff range." If the object is closer than 2 m then the Teensy alerts the Raspberry Pi. This distance was calculated using the reaction time and stopping distance of the robot at our nominal velocity.

If it is determined that there is an obstacle within 2m of the board along the current path, the status of left and right are checked. If there is no obstacle to the left, it is deemed safe for the board to turn left around the obstacle. If there is an obstacle left, right is then checked. If right is clear, then the board will steer right. If right is also blocked by an obstacle, then the board will come to a stop until the obstacle is cleared.

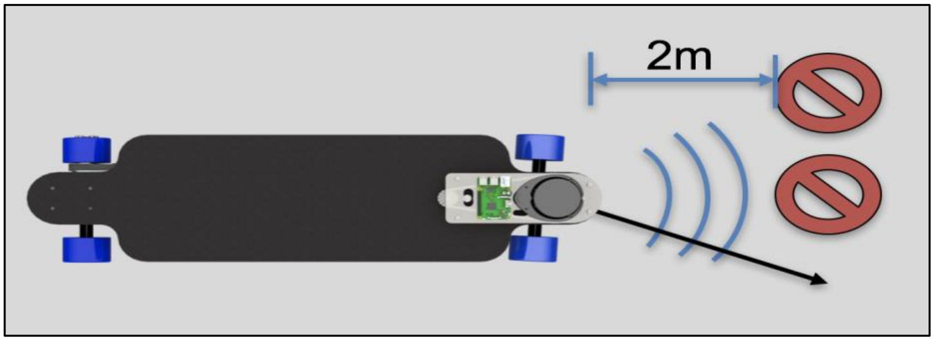


Figure 47: Obstacle detection range using LIDAR.

9. Results

9.1 Summary of Technical Specifications

9.1.1 Primary Specifications

Upon completion of the project we revisited the technical specification we originally stated and determined whether or not we met each specification and how we met each one.

TSP01	Load bearing capacity: 200lb +	✓
TSP02	Weight ≤ 10kg	✗
TSP03	Safety feature: Handle Bar	✓
TSP04	Speed of ≤ 10 mph.	✓
TSP05	Battery capacity for multiple round trips to 85 Prescott St from HL.	✓
TSP06	The platform must be able to reach full charge in less than 5 hours.	✓
TSP07	The platform must be able to locate itself outdoors using GPS	✓
TSP08	Detect obstacle at safe stopping distance.	✓
TSP09	Load sensor to detect rider.	✗
TSP10	Dimensions must be 12 inches wide, 38 inches long, and 8 inches high.	✓
TSP11	The cost of the project must not exceed 800 USD	✗

TSP01 - The task was met and proven successful through SOLIDWORKS simulations as well as physical tests from us standing on the board and applying loads up to 200 lbs.

TSP02- The platform weighs 12kg which is greater than the 10kg but we determined that the board is still portable compared to other personal mobility platforms like a Segway. The board can still be easily picked up and carried.

TSP03- The platform has a detachable and collapsible handlebar which we designed but was not build. The handlebar was created as a support for the rider to lean on while riding on the longboard. We also affixed grip tape along the top of the board to provide a better surface for securing the rider and reducing the risk of slippage.

TSP04- The platform's drive system is capable of reaching 24 mph but the software limits the top speed to around 6 mph.

TSP05- The platform's battery has the capability of lasting around 5 miles with the full electronics suite, which covers multiple round trips to 85 Prescott Street which is approximately a half mile each way.

TSP06- The platform's battery has a quick recharge ability which is 3 hours.

TSP07- Using GPS sensor the robot logs GPS coordinates periodically and sends it to the Raspberry Pi.

TSP08- Using a Teensy 2.0 the LIDAR monitoring range is set to 2 m which is more than braking distance of the robot at operating speed of 6 mph.

TSP09- Due to time constraints board could not be tested with a rider, so a load sensor was not implemented.

TSP10- The board is 10 inches wide, 37 inches long and 7.5 inches high.

TSP11- That total cost of the project was approximately 1,100 USD. The cost is made up of materials for each of the components, two motors, LIDAR, RASPI, TEENSY, materials for thermoforming the covers and hardware needed to assembly the board.

9.1.2 Secondary Specifications

Additional mechanical attributes that may be taken into account include ergonomic design elements to minimize discomfort for the rider. In terms of these attributes, we have the following secondary considerations encoded in “Task Spec - Secondary - Number” format (TSSXX):

TSS01	The platform should be waterproof for outdoor use.	✗
TSS02	The platform should have a commute mode.	✗
TSS03	The platform should have a sport mode.	✗
TSS04	The platform should be able to reach full charge in less than 3 hours.	✓
TSS05	The platform should make independent decisions without user intervention.	✗
TSS06	The platform should be capable of adjusting torque according to terrain.	✗

TSS01- The platform should be waterproof for outdoor use: Not tested

- The board was treated with three coats of clear waterproof varnish
- Electronic components are protected by a custom polycarbonate cover with watertight seal.

TSS02- The platform should have a commute mode: Not achieved

TSS03- The platform should have a sport mode: Not achieved

TSS04- The platform should be able to reach full charge in less than 3 hours: Achieved

TSS05- The platform should make independent decisions without user intervention: Not achieved

TSS06- The platform should be capable of adjusting torque according to terrain: Not achieved

9.2 Test Results

Once we had the robot assembled, we aimed to test the its autonomous navigation and driving capability in a controlled environment. Therefore, we chose to test it on the outdoor track on the university campus. Figure # shows a snapshot from Google maps showing the path (an arc) chosen for test. We obtained a list of waypoints along the arc using Google maps. However, before testing the robot as a whole, we tested the subsystems to assess their functionality.

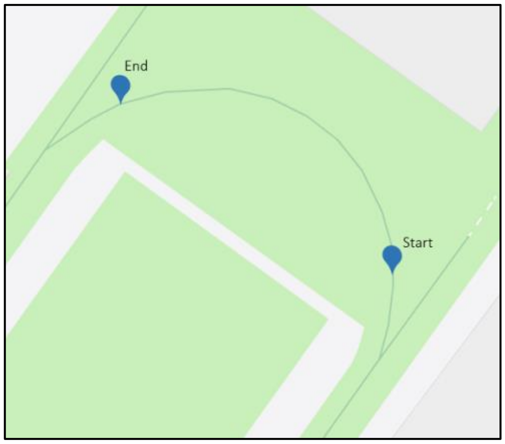


Figure 48: Google maps view of path chosen for test on WPI track.

9.2.1 GPS Module

Testing the GPS module accuracy was relatively simple. We connected it to the Raspberry Pi and using a Python script logged the coordinates at specified interval while walking on the path we had chosen for test. Then we plotted the path produced by our GPS data against those obtained from Google maps. The result is shown in figure #. The maximum deviation for this particular test was approximately 1m.

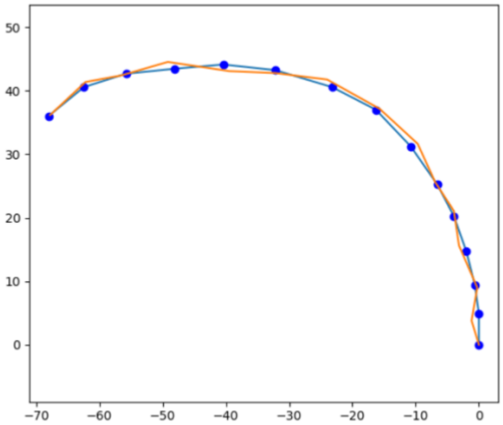


Figure 49: GPS position (orange) given by our sensor along the waypoints from google maps (blue)

9.2.2 Drive Module Test

In order to test the drive function and reliability of the position estimates produced by our kinematics model, we collected data from several experimental runs. The test was devised to vary the duration of drive keeping the input duty cycle (hence the command velocity) constant. For each run, we measured the distance travelled using a tape measure. The estimated and measured distances for each run were plotted against the respective durations. The experiment was repeated after refining the mathematical (using a scale factor) model and the results were very close to the measurements. This can be seen from figure #. Red line corresponds to our model before adjustment, green one corresponds to the model after adjustment, and blue line was the actual measurement.

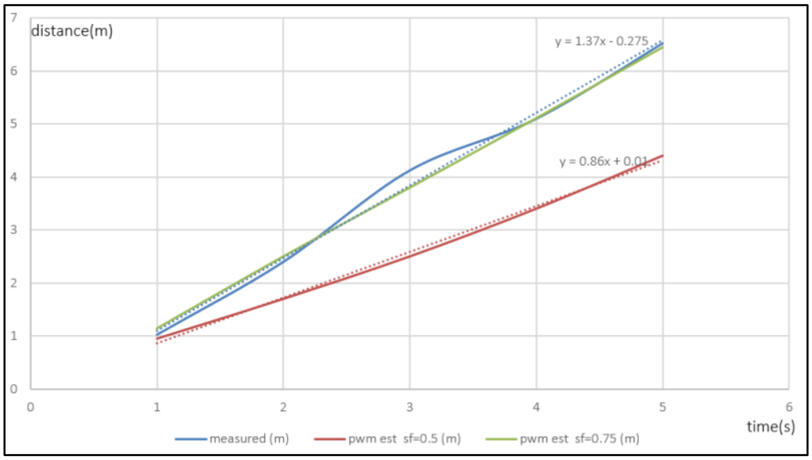


Figure 50: Estimated position using different scale factors compared against measured position.

9.2.3 Steering Module Test

We tested the steering assembly by measuring the actual angle turned by the truck and comparing it with our estimate from pwm mapping. We measured the actual angle by mounting a smart phone on the truck in horizontal position, and using the tilt angle measurement feature in the phone once the truck turned. The angle measurements were generally consistent within a degree. However, we discovered that sometimes the signal was noisy, which resulted in twitchy behaviour of the steering. After several test attempts, we determined this issue to be inconsistent in nature. This rendered the steering unreliable for independent course adjustment during autonomous navigation.

9.2.4 Simulation Results:

In order to test the behavior of the robot without fully functioning locomotion, we decided to simulate the autonomous navigation and obstacle avoidance. The simulation was created to serve as a proof of concept and observe how certain parameters, such as steering angle and drive speed, affect the functionality of the robot. Additionally, it resulted in being an efficient tool for debugging and improving the system in general. The simulation was programmed in Python 2.7. The program runs the simulation in real time, meaning the robot's motion can be observed as it follows the assigned path.

For the simulation, we used the same list of waypoints we created for the outdoor test on the track. In addition, we placed two obstacles on the path to demonstrate obstacle avoidance. The navigation logic implemented in the simulation is same as outlined in section 8. For avoiding the obstacles, the simulated robot steers in the direction that requires least deviation from its normal trajectory.

Test 1:

In this test we simulated the robot's motion without obstacle avoidance. For this simulation, the steering angle was set to a constant of +/- 3 degrees. This was based on how the robot was actually programmed initially for proportional navigation. The drive velocity was set to 1.4 m/s by keeping the duty cycle at a constant value. For this and all subsequent tests, the threshold of "line of sight" (LOS) distance that determines the range in which the robot is considered to have reached a target waypoint is set to 1m. Figure 51 shows the simulation result.

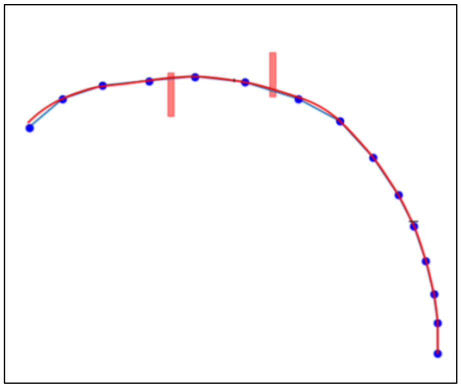


Figure 51: Simulated autonomous navigation without obstacle detection.

From the plot, we see that even though the modeled robot was able to hit all the waypoints, it deviated from the path significantly more wherever the gradient of the path changed more than average, or suddenly. For instance, at the curve right before the end and before the first obstacle. The limitation of this method of navigation and control was more evident in the next test.

Test 2:

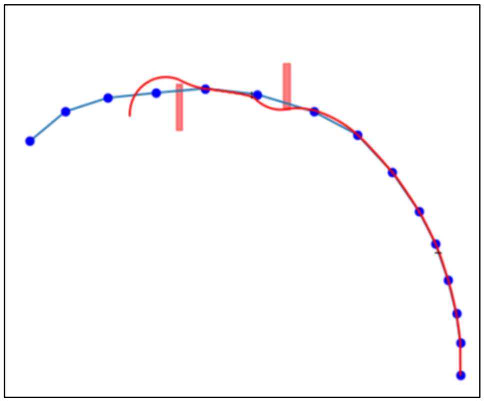


Figure 52: Control parameters leading to divergence from trajectory.

In this test, we enabled obstacle avoidance, keeping all parameters same as before. We can see that the control behavior was not able to follow the path once it tried to avoid the obstacles. It barely cleared the first obstacle and returned to the path. However, the second obstacle caused the robot to deviate so much from its path that with steering constrained at 3 degrees, it was not able to hit the waypoint immediately after the second obstacle. The velocity being constant, the robot was unable to correct its heading before passing the immediate next waypoint. At the same time, since it was never within the LOS threshold, the program did not realize that the waypoint was passed. Thus, it kept trying to go back to the waypoint, which consequently led to the erroneous behavior.

Test 3:

In test 3, we adjusted the steering angle to 10 degrees for avoiding the obstacle and 15 degrees for returning to the path. This was done because naturally the return mechanism requires more course change than deviating from the path. Additionally, the drive control was adjusted to lower the velocity to 1m/s for instances when the robot is trying to do course correction. The implementation of this can be seen to positively change the robot's motion in the following figure.

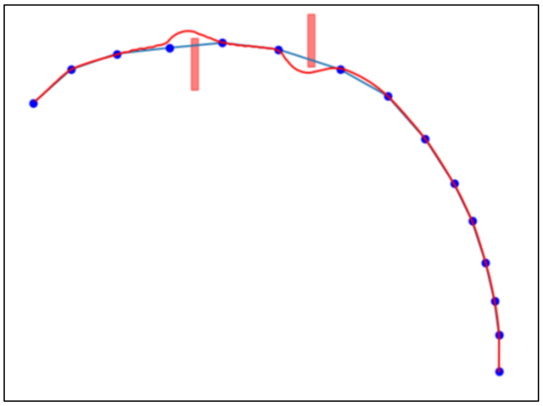


Figure 53: Obstacle avoidance with minimum steering angle.

However, if we take a closer look at the instances where the robot is deviating and returning to the assigned trajectory, another problem becomes apparent. The return instance after the second obstacle succeeds by a narrow margin. It is very close to the threshold, so the robot could have missed the waypoint by a slight more initial deviation from its path. We can change the program logic to consider such obstacles as unavoidable to begin with, so that the robot just stops its journey. However, the control behavior can be changed further to enhance the obstacle avoidance ability. This is illustrated in the next test.

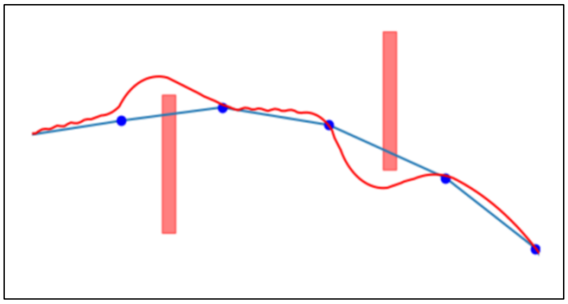


Figure 54: Close up view of obstacle avoidance.

Test 4:

In this test, we simply increased steering angles' limit for all instances. The plot indeed verifies the expected result that the model is able to avoid diverging from the trajectory too much. The increased steering angle enabled the robot to make sharper turns and stay close to the path. However, an undesired effect of this is the overshoot during course correction. The robot responds to each instance of course correction by oversteering, which is seen by the squiggly path drawn in red. This would be impractical to implement on the longboard since it will keep steering continuously back and forth making it uncomfortable and potentially dangerous.

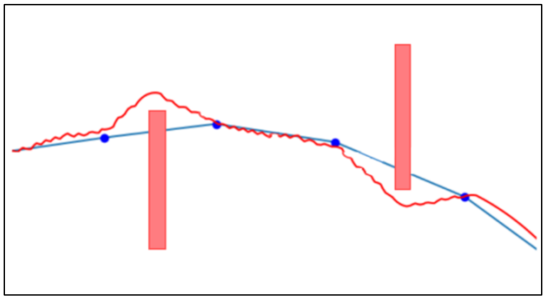


Figure 55: Close up view showing overshoot during steering corrections.

Test 5:

In this final test, we have simulated an optimal solution – proportional/closed loop control. In this scenario, the program determines a suitable steering angle and velocity based on the amount of course correction required. The result is seen from the figure below. The path taken by the robot is relatively smooth and close to the assigned path.

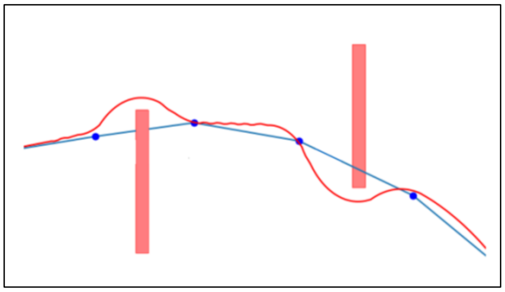


Figure 56: Simulated proportional control resulting in smooth trajectory.

The results of the simulations above allowed us to improve the program used by the robot for navigation and control significantly. This was particularly beneficial because despite not being able to test the actual system, we were able to validate the robot's behavior to a certain extent in a short span of time. Based on these results, we were able to make important future recommendations for the project.

10. Conclusions

Our goal for the project was to create an Autonomous Mobility Platform capable of self-navigating around campus. We designed, manufactured, built, and programmed a machine capable of autonomous navigation. The board is also light, portable, and charges quickly. While similar mobility systems exist, our solution fills a gap previously unfilled. We implemented and tested each module of the system independently and integrated them into a complete robot. In the end, the robot successfully fulfilled 8 of the 11 primary specifications. The results showed a sound proof of concept for achieving the goal envisioned in this project.

This project taught us nearly all different aspects of mechanical and robotics engineering including, but not limited to, stress analysis, thermo-fluid design, design of machine elements, power systems, localization and mapping, navigation, guidance, and control.

Bibliography

VIDEOS

1. ASLCornell. “Introduction to ASL's Robotic Segways.” YouTube, YouTube, 27 Feb. 2014, www.youtube.com/watch?v=qZQ_48zmKGU.
2. Freego2012. “Freego WheelChair WC-01.” YouTube, YouTube, 13 Sept. 2016, www.youtube.com/watch?v=LyrOAhZ8nss.
3. leorobotics. “Turning a Segway into a Personal Robotic Beer Delivery Cart.” YouTube, YouTube, 17 June 2011, www.youtube.com/watch?v=2o4Ug6ERKyo.
4. MobileRoboticsOxford. “Lisa - Segway Based Robot on a Data Collection Run.”
5. YouTube, YouTube, 24 Sept. 2011, www.youtube.com/watch?v=7TCerC3SOqk.“Ogo Episode 21 - Going Places with Indiegogo!” YouTube, YouTube, 8 Mar. 2016, www.youtube.com/watch?v=WwmXz99J9ZU.

ARTICLES

1. "Automated and Autonomous Driving Regulation under Uncertainty." International Transport Forum, cyberlaw.stanford.edu/files/publication/ files/15CPB_AutonomousDriving.pdf
2. Autonomous intelligent vehicles (systems used in autonomous https://link.springer.com.ezproxy.wpi.edu/content/pdf/10.1007%2F978-1-4471-2280-7.pdfehicles)
3. Desmond Caulley, Nadav Nehoran, Sherry Zhao. "Self-Balancing Robot." Welcome - School of Electrical and Computer Engineering - Cornell Engineering, Cornell University, people.ece.cornell.edu/land/courses/ece4760/FinalProjects/f2015/dc686_nn233_hz263/final_project_webpage_v2/dc686_nn233_hz263/index.html.
4. Gates, Guilbert. (2016, December 14). The Race for Self Driving Cars. Retrieved from: https://www.nytimes.com/interactive/2016/12/14/technology/how-self-driving-cars-work.html
5. Hall-Geisler, Kristen. "Oxbotica's Software For Self-Driving Cars Doesn't Need GPS Signal." Popular Science, 18 Aug. 2016, www.popsci.com/software-for-self-driving-cars-without-gps.
6. Johnson, Benjamin, et al. "Experimental Evaluation and Formal Analysis of High-Level Tasks with Dynamic Obstacle Anticipation on a Full-Sized Autonomous Vehicle." Journal of Field Robotics, 14 Mar. 2017, onlinelibrary.wiley.com/doi/10.1002/rob.21695/full.
7. Larry Hardesty | MIT News Office. "Driverless-Vehicle Options Now Include Scooters." MIT News, 7 Nov. 2016, news.mit.edu/2016/diverless-scooters-1107.
8. Kim, Xiongzhe HanHak-Jin, et al. "Development of a Low-Cost GPS/INS Integrated System for Tractor Automatic Navigation." International Journal of Agricultural and Biological Engineering, ijabe.org/index.php/ijabe/article/view/3070/pdf.
9. Li, Zhijun, and Yang Li. "Motion Control of an Autonomous Vehicle Based on Wheeled Inverted Pendulum Using Neural-Adaptive Implicit Control." pdfs.semanticscholar.org/bfd2/c59e12f93d2eba6b2b630a8646204f31e98f.pdf.
10. Meiners, Jens. "Dorks and Mall Cops Take Heed: Volkswagen to Sell Foldable Segway-Type Thing." Car and Driver Blog Dorks and Mall Cops Take Heed

Volkswagen to Sell Foldable SegwayType Thing Comments, Hearst Communications, Inc., 6 July 2015, 10:44, blog.caranddriver.com/dorks-and-mall-cops-take-heed-volkswagen-to- sell-foldable-segway-type-thing/.

11. Nosowitz, Dan. “Test Drive: GM's EN-V Electric Transporter, Half a Smart Car On a Segway Base.” Popular Science, 9 Jan. 2011, www.popsci.com/cars/article/2011 -01/gms -en-v-electric-car-half-smart-car-segway-base.
12. Österling, Anders Eskil. “On the Skateboard, Kinematics and Dynamics.” By Anders Eskil Österling :: SSRN, Stockholm University, 26 Apr. 2011, poseidon01.s srn.com/delivery.php?ID=882123066117006090065027003019007104017037064079033004076088092068091124087005127000036041017035040124044003112071000067001067038045044040053069030005102122114112102002081037124067126021115103085089015108118007091126103110123012099024118071108111025067&EXT=pdf.
13. Pilonieta, Carlos. “Gyro Two Wheeled Gyroscopically Stabilized Electric Vehicle.” Tuvie, 2017,www.tuvie.com/gyro-two-wheeled-gyroscopically-stabilize d-electric-ve hicle-by-carlos-pilonieta/.
14. Polack, Philip. “The Kinematic Bicycle Model: A Consistent Model for Planning Feasible Trajectories for Autonomous Vehicles?” The Kinematic Bicycle Model: A Consistent Model for Planning Feasible Trajectories for Autonomous Vehicles? - IEEE Conference Publication, 31 July 2017, ieeexplore.ieee.org/document/7995816/.
15. Qiao, Junqing. “Semi-Autonomous Wheelchair Navigation With Statistical Context Prediction.”WORCESTER POLYTECHNIC INSTITUTE, 2016, web.wpi.edu/Pubs/ETD/Available/etd-053016-140503/unrestricted/Thesis.pdf.
16. Scalevo Is Now Scewo!” Scalevo, Elegant Themes, scalevo.ch/.
17. Self Driving Cars Explained. (2017, January 27). *Concerned Scientists*. Retrieved from: http://www.ucsusa.org/clean-vehicles/how-self-driving-cars-work#.We90pBOPLqQ
18. Stephens, Leah. (2016, April 28). How Driveless Cars Work. Retrieved from: https://interestingengineering.com/driverless-cars-work
19. Teller, Seth, and Nick Roy. “Autonomous Wheelchair.” CSAIL, MIT,

www.csail.mit.edu/videoarchive/research/robo/autonomous-wheelchair.

20. "The Autonomous Mobile Robot SENARIO: a Sensor Aided Intelligent Navigation System for Powered Wheelchairs." The Autonomous Mobile Robot SENARIO: a Sensor Aided Intelligent Navigation System for Powered Wheelchairs - IEEE Journals & Magazine, IEEE, Dec. 1997,ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=637806.
21. Vlahos, James. "A Self-Balancing, Two-Wheeled Car." Popular Science, 13 May 2015, www.popsci.com/self-balancing-two-wheeled-car.
22. Volkov, Alexander, et al. "Segway Robot." ECE 4760 Segway Robot, Cornell University,people.ece.cornell.edu/land/courses/ece4760/FinalProjects/f2014/avh34/Website/Website/index.html.
23. Ziegler, Chris. "This Audi Has a Beautiful Electric Longboard Hidden in Its Bumper." TheVerge, 25 Apr. 2016, www.theverge.com/2016/4/25/11503794/au di-electric-autonomous- longboard-hidden-bumper.

WEBSITES

1. Campbell, Prof. Mark. “Main Page.” Autonomous Systems Laboratory Wiki, Autonomous Systems Lab, cornell-asl.org/wiki/index.php?title=Main_Page.
2. “Frontier V6, Indoor & Outdoor Wheelchair, Innovation in Motion.” Innovation In Motion, CLP Marketing LLC, www.mobility-usa.com/frontier-v6-wheelcha ir.html.
3. Leavitt, Austin. “The XTND Board Is the World's First Electric Skateboard.” Gear For Life, 31 May 2017, gearforlife.com/xtnd-board-worlds-first-electric-skateboard/.
4. “Papers.” Oxford Robotics Institute, ori.ox.ac.uk/publications/papers/.
5. Siegwart, Roland. “Autonomous Mobile Robots.” pdfs.semanticscholar.org/463 5/5ad96cd566 a5c74515ab252877e29f63e3e8.pdf.
6. Skating, E. “How to Choose Electric Longboard Deck Shapes.” The Electric Skateboard Drive That Fits under Any Skateboard, E-Skating, 23 Feb. 2017, www.mellowboards .com/en/blog/how-to-choose-electric-long board-deck-shapes/.
7. “The Core of Your Mobile Robotics Solution.” Segway Robotics, SegwayRobotics Inc., rmp.segway.com/2009/12/15/cornell-search-and-rescue-robots-to-compete-internationally/++.
8. xu.bai@ninebot.com. “Segway Robotics.” Segway Robotics, SegwayRobotics Inc., www.segwayrobotics.com/.

Past MQP Projects

1. Breznak, Robert S, et al. “Autonomous Vehicle Control.” Worcester Polytechnic Institute,WPI, 21 Apr. 2008, web.wpi.edu/Pubs/E-project/Available/E-project-042108-160447/.
2. Marbel. “Marbel 2.0 Twin Drive - Electric Skateboard | Electric Longboard.” MarbelBoards.com, MARBEL, 2017, marbelboards.com/twin/.
3. Sahay, Prateek. “Robocart: Autonomous Ground Vehicle.” web.wpi.edu/Pubs/E-project/Available/E-project-043015-034638/unrestricted/Final_Report.pdf.
4. Siegward, Ronald, et al. “Autonomous Mobile Robots - Spring 2017.” Autonomous Mobile Robots - Spring 2017 – Autonomous Systems Lab , ETH Zurich, www.asl.ethz.ch/education/lectures/autonomous_mobile_ro bots/spring-2017.html.
5. “XTND - World's First Electric Board with AI.” Kickstarter, XTNH, Inc., 2017, www.kickstarter.com/projects/910224259/xtnd-worlds-first-electric-board-with -ai.

Appendices

Appendix A - Magnetometer Calibration

Magnetometers need to be calibrated for *hard iron distortion* caused by presence of other electronics or magnetic materials around. The hard iron distortion results in a constant bias in the sensor output. On the longboard, the BerryGPS-IMU board is mounted directly on top of the RaspberryPi GPIO pins and is surrounded by other electronics. Thus, the sensor bias is significant.

Such a bias can be observed if the sensor is rotated 360 degrees, and the X, Y component values are plotted. In an ideal scenario, the plot is a circle centered at the origin. However, presence of hard iron distortion shifts the plot away from the center. The offset needed to compensate for bias is calculated by averaging the maximum and minimum values in each axis. The respective offsets are subtracted from the raw readings in each axis to obtain calibrated readings. Furthermore, the shape is usually elliptical rather than a circle because of the presence of what is known as a soft iron distortion. Because of the complexity of compensation for a soft iron distortion, and its effect being relatively small, we only proceeded with calibration for hard iron distortion.

In order to compensate for that bias, we rotated the entire longboard holding stationary the end where the sensor is mounted. This allowed the sensor to read the magnetic field in all directions (2D), in presence of the same hard iron distortion that would be present during usage. The readings were recorded and plotted in excel.

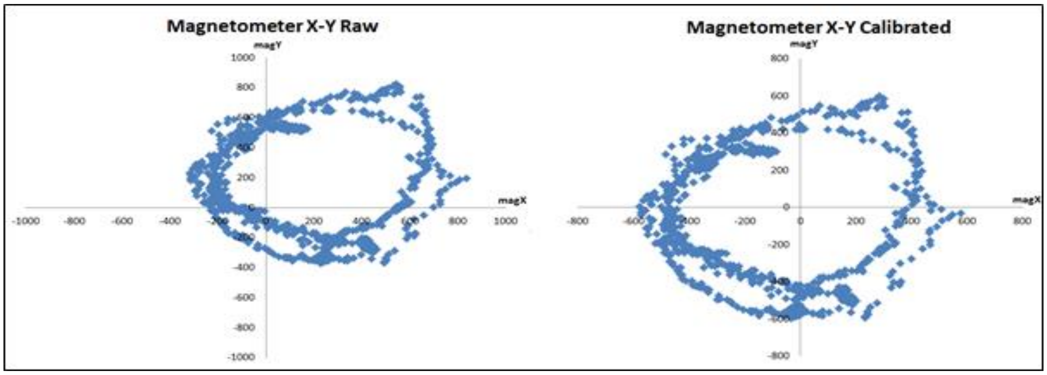
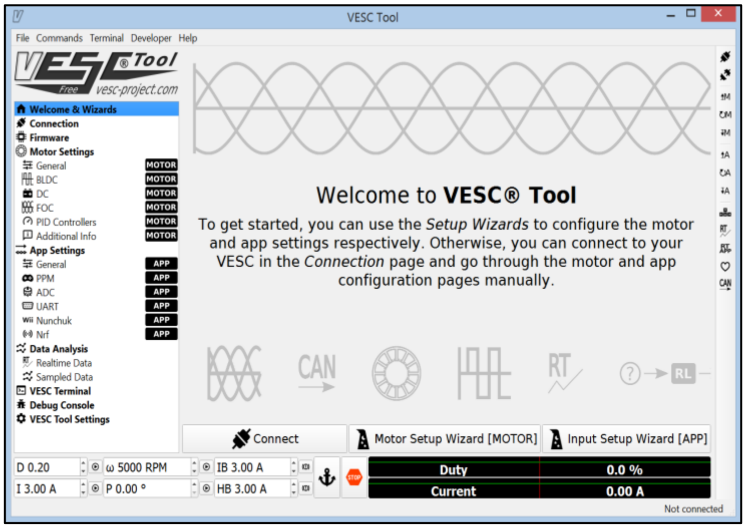


Figure 57: Magnetometer readings from rotating the sensor, before and after calibration.

Appendix B - VESC Tool and Motor Controller Configuration



Welcome to **VESC® Tool**

To get started, you can use the *Setup Wizards* to configure the motor and app settings respectively. Otherwise, you can connect to your VESC in the *Connection* page and go through the motor and app configuration pages manually.

D 0.20
ω 5000 RPM
IB 3.00 A
I 3.00 A
P 0.00 °
HB 3.00 A

Duty	0.0 %
Current	0.00 A

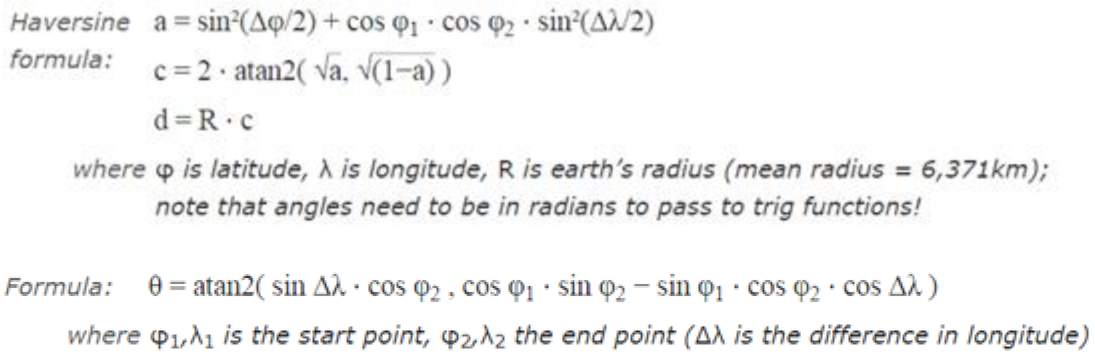
Not connected

Figure 58: User interface of the application used to configure the drive motor.

The VESC is an open source platform developed by Vedder. It is designed to work with a large range of brushless DC motors. To configure the motor controller to work with our chosen brushless motor, we used the VESC tool. This is also an open source program that gives a user full control over the settings of the motor controller. First the motor controller is connected to the laptop via USB. The “Motor Setup Wizard” is then run which sets the voltage limits, current draw limits, and hall table values. We set the motor controller up to work with a PWM input. It is also capable of other communication types such as UART.

Appendix C - GPS Measurement to Cartesian Coordinates

In order to project the spherical coordinates from GPS sensor to a 2D cartesian coordinate frame, the Haversine formula can be used to calculate distance between two points on a sphere. Using the bearing formula below, this distance can be decomposed into cartesian components. This way localization can be done relative to starting point.



Haversine formula:

$$a = \sin^2(\Delta\varphi/2) + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \sin^2(\Delta\lambda/2)$$

$$c = 2 \cdot \operatorname{atan2}(\sqrt{a}, \sqrt{(1-a)})$$

$$d = R \cdot c$$

where φ is latitude, λ is longitude, R is earth's radius (mean radius = 6,371km);
note that angles need to be in radians to pass to trig functions!

Formula: $\theta = \operatorname{atan2}(\sin\Delta\lambda \cdot \cos\varphi_2 , \cos\varphi_1 \cdot \sin\varphi_2 - \sin\varphi_1 \cdot \cos\varphi_2 \cdot \cos\Delta\lambda)$

where φ_1,λ_1 is the start point, φ_2,λ_2 the end point ($\Delta\lambda$ is the difference in longitude)

Appendix D - Stress Analysis

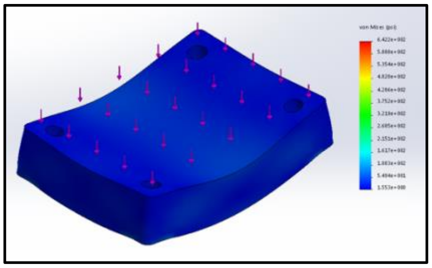


Figure 59: Rear Truck Support

The rear truck support is made of 6061 Al and has forces from the rider and weight of the board applied downward on it.

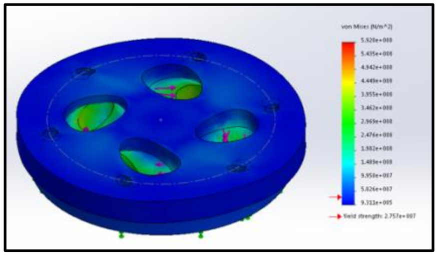


Figure 60: Drive Truck Support

The drive truck support is made of 6061 Al and has a torque applied to the four inner bolt holes where it attaches to the adaptor_2.

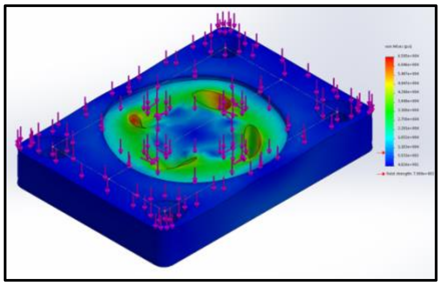


Figure 61: Front Truck Adaptor

The front truck adaptor is made of 6061 Al and has a torque applied to the four inner bolt holes where it attaches to the adaptor_2 component. The part also has a downward force applied to it from the weight of the rider and other components.

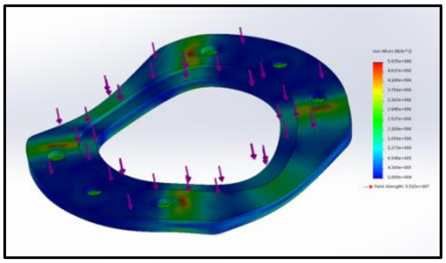


Figure 62: Bottom Bearing Support

The bottom bearing support is made of 6061 Al and has a downward force applied on it from the weight of the rider and board.

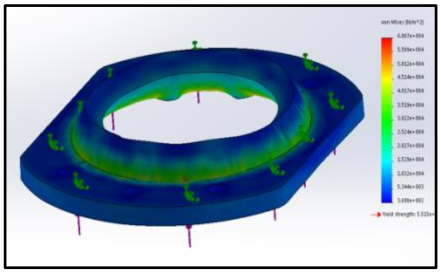


Figure 63: Top Bearing Support

The top bearing support is made of 6061 Al and has a force acting upward on it from the thrust bearing it is supporting.

Appendix E - Future Recommendations

As is the nature of projects, there were a great number of challenges we encountered that we were unable to complete during the duration of the project.

- More accurate speed Control of drive system
- Position control of steering system
- On-board path planning and waypoint generation
- Improve Kalman filter estimate
- Add sidewalk detection system
- Add rider detection system
- Implement a “smarter” obstacle avoidance system

These are just a few examples of potential future work on the project.