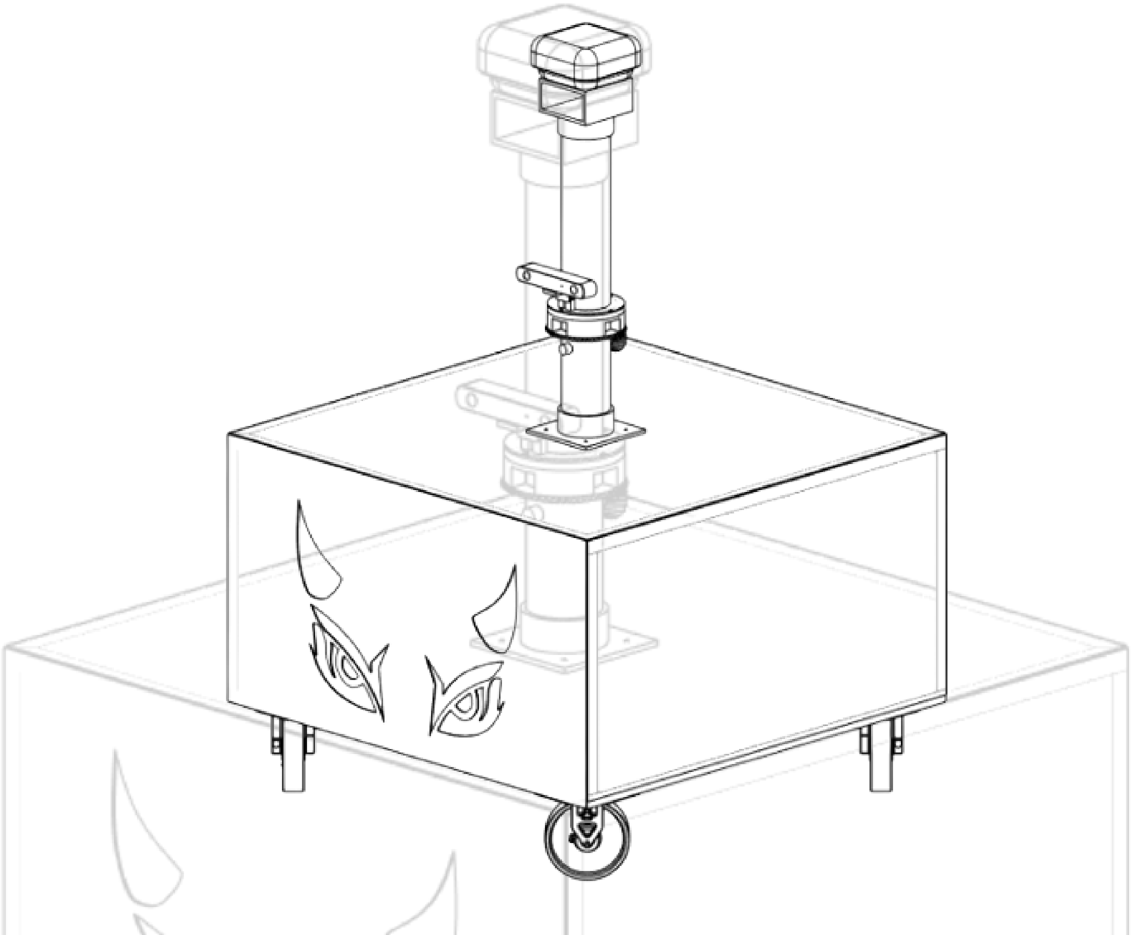
Lawrence Technological University

T.B.D.: The Blue Devil



5/14/2022

Captain: Roger Franzel II (rfranzel@ltu.edu)
Team: Marisa Assink (massink@ltu.edu), John Hall (jhall@ltu.edu), Ethan Harsh (eharsh@ltu.edu), Kyle Hendrian (khendrian@ltu.edu), Jerich Lopez (jlopez@ltu.edu), Grant Marshall (gmarshall@ltu.edu), Alecxander Pilon (apilon@ltu.edu), Mark Zammit (mzammit@ltu.edu)

"I certify that the design and engineering of "T.B.D.: The Blue Devil" by the 2021/2022 Lawrence Technological University Robotics Team has been significant and equivalent to what might be awarded credit in a senior design course"

Advisor: Dr. Giscard Kfoury

1. Design process, team organization

1.1 Introduction

We are T.B.D., a team competing on behalf of Lawrence Technological University with our robot The Blue Devil. Our team's aim is to engineer the best autonomous vehicle we can while innovating on as many aspects we see can be improved. We are a team made entirely of robotics engineering majors, using this opportunity in IGVC AutoNav to teach ourselves as much as we can about design, fabrication, electrical wiring, and programming of an autonomous system. This robot is an entirely new design compared to LTU's robots in previous years of AutoNav, however our design choices were made by learning from the previous iterations.

1.2 Organization

The team is composed of nine robotics engineers split into four subteams, each with a team leader. Each subteam has two co-leads who are responsible for planning and meeting their goals in their subteam. Two supporters fill in when their current workload is not demanding.

Team leader: Roger Franzel II				
	Fabrication	Design	Programming	Electrical
Leads	Alecxander Pilon & John Hall	Kyle Hendrian & Ethan Harsh	Marisa Assink & Grant Marshall	Jerich Lopez & Mark Zammit
Support	Jerich Lopez Ethan Harsh	John Hall Grant Marshall	Mark Zammit Kyle Hendrian	Alecxander Pilon Marisa Assink

1.3 Design assumptions and design process

At the start of our design process we first discussed different ways we could maneuver our robot. There were four main methods; ackermann steering, tank drive, crab drive, and swerve drive. First eliminated was crab drive due to the complexity and how unnecessary the design would be for this competition. Ackermann steering followed after that. This was due to the new rule of needing a reverse function and the other was needing to have the camera in a forward facing position. Then it was up to tank or swerve drive. We chose to utilize the swerve drive system in the end. This decision came down to the course being on asphalt and how easily modifiable the swerve drive systems we found were.

Second part of our design process was to model a robot that would incorporate the swerve drive system. Following some research we found that having the root have a square, symmetrical frame would be beneficial for the coding of the motors. We looked over the rules to identify the smallest square size we could have, which was found to be 3 ft x 3ft. Then we decided to have our frame made of aluminum tubing. We assumed that this would be our best choice due to having members experienced working with aluminum and its durability to cost ratio.

Third we discussed motors needed. The four major points we focused on were; cost, torque, power, and additional components. This led us to choosing the Nema 23 Stepper Motors. This had a good price point at $34 per motor, as well as about 20 lbs more torque than what we calculated was needed(49 needed, 72 found). The power source these would need would be a 24V battery that has 18 AH. Though sadly, unlike some of the other motors, this did not have any built in encoders.

Lastly we deliberated on the forms of sensing. For forms of detecting the obstacles in the course we had the option of utilizing a depth camera, a lidar, or both. Both the depth camera and the lidar were from previous teams and had been donated. In the end we decided on only the depth camera with a mount that could swivel it around. One of the leading factors in this decision was that a member of the coding team had worked with depth cameras before. Another was that due to the barricades that are in the course itself the lidar would have difficulty detecting it. It can only detect at one level.

2. Effective innovations in vehicle design

2.1 Innovative concept(s) from other vehicles designed into your vehicle

One concept that we considered incorporating into our robot was a steering design similar to that of the new GMC Hummer EV. It is equipped with a “crab walk” steering system that keeps the vehicle facing forwards while being capable of steering in a different direction. The idea of the crab drive was modified into the idea of a swerve drive, examples of which we had heard of or seen from some FIRST Robotics Competition teams’ robots.

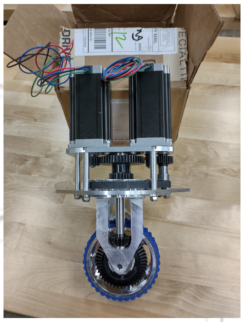


Figure 1: Swerve drive module

Another innovative concept is the use of the depth camera. An idea we had witnessed from a team in the 2021 IGVC’s competition was using the depth camera to create a 2D layout of the obstacles and boundary limits around them, an idea we will be using for our vehicle. Additionally, we decided that our robot would be relying only on the depth camera with no LIDAR supporting it. We also are going to be able to use the camera in every direction by taking and adding a system that rotates the camera around so it can view 360 degrees of the robot. It will be mainly so we can look behind while backing up so we do not run into obstacles behind the robot.

Figure 2: Camera mount

2.2 Innovative technology applied to your vehicle

Our robotic system incorporates the use of swerve drive. Each wheel has two stepper motors, one for movement and another for rotation, which enables movement of all 4 wheels’ angles independently. With the use of the swerve, we can achieve a near-zero turn radius while also allowing the robot to move sideways or diagonally without changing the direction it is facing.

3. Description of mechanical design

3.1 Overview

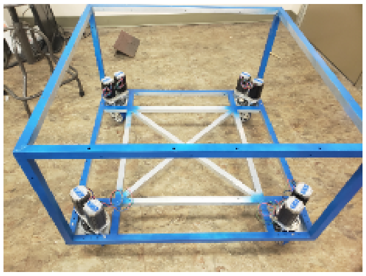


Figure 3: Robot frame with swerve modules

The robot consists of a square frame, 3 feet by 3 feet, with a height of 1.5 feet made of 1” thick aluminum tubing with ⅛” walls. Wheels add an additional 6 inches of height, but the center of gravity is still low enough to avoid tipping on ramps. The drivetrain is based on a swerve drive system where each wheel can operate independently from the others. Each module consists of two motors, one to steer and one to drive. Within each module, there are many moving gears and shafts that create rotation and drive motions, which moves our robot .The internal components will either be sitting on the bottom of it or on a diagonal panel to grant easy access to working on it.

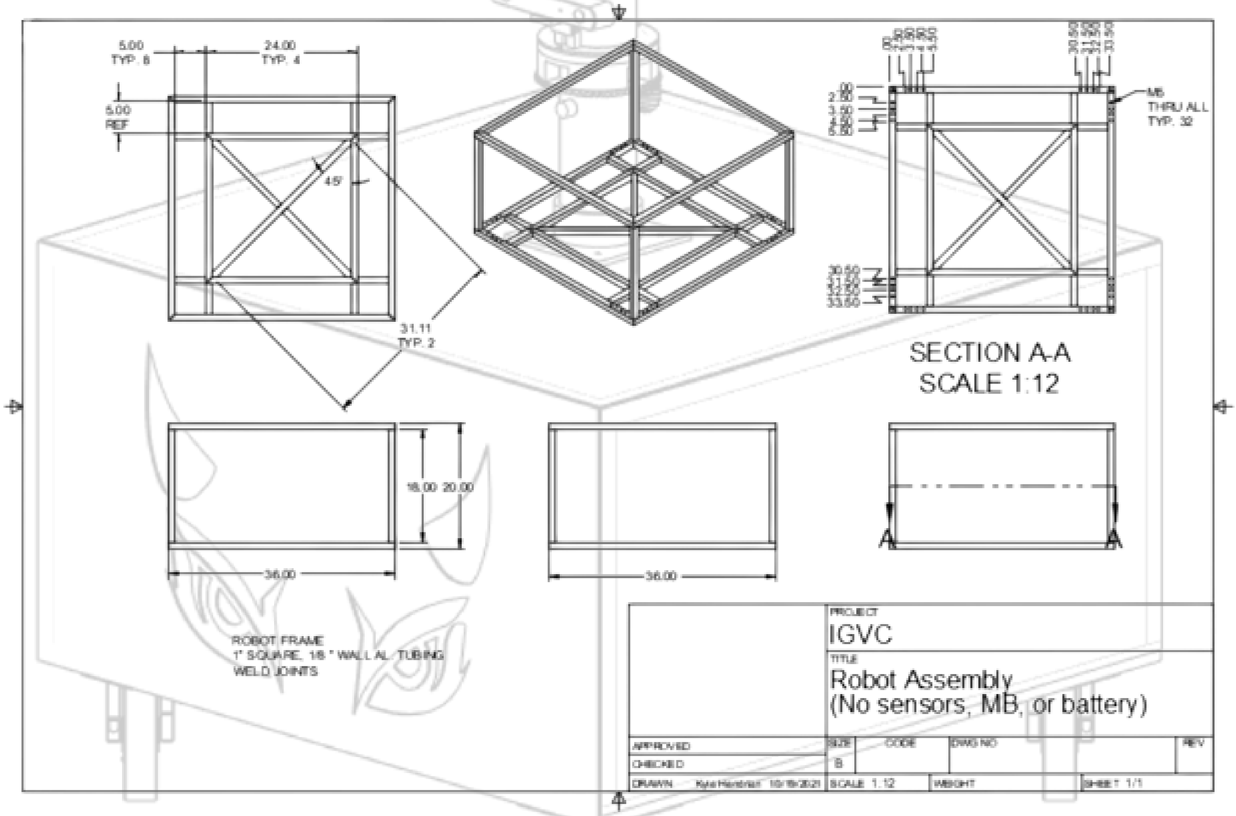


Figure 4: CAD drawing of frame dimensions

3.2 Decision on frame structure, housing, structure design

The frame of the robot was decided on after we thought through a few different options for the drive system. With the assumption that a swerve drive would be best, we chose to make the base of the frame a 3 foot by 3 foot square. The reasoning for this was that we wanted the robot to be as small as possible, but still square, which left us having to follow the minimum 3 foot limit from the length restrictions. The height was chosen to be 1.5 feet high, excluding the wheels. This was to keep the center of gravity low enough to avoid tipping when on 10 degree inclines.

The material was chosen by looking at the price, weight, strength, and machinability of different possible materials. The main two choices were steel or aluminum for their strengths, but steel seemed far too heavy for these purposes. The downside of aluminum would normally be its weldability, however our fabrication team had experience with welding aluminum and seemed confident with it. So after looking at costs, we went with 1 inch thick square tubing with 1/8th inch thick walls. To verify this would work for our purposes we tested a basic static stress FEA using Autodesk Fusion 360's simulation software by simulating what we estimated the total weight would be including motors, batteries, and payload.

The internal electronics were to be housed within the frame of the robot where they would be protected from weather and wind. We wanted them to be easily accessible from the front of the robot while leaving room for the payload and batteries behind, so to accommodate these requirements we mounted them on a slanted plastic board on the inside of the robot.

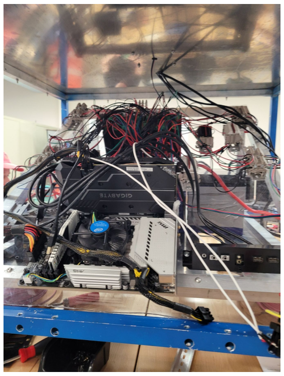


Figure 5: Internal electronics

3.3 Suspension

In terms of suspension, our robot is not equipped with a designated system. This means that we do not have shock absorbers or springs. The only suspension traits will come from the tires, which will provide a slight amount of rebound when our robot encounters a bump.

3.4 Weatherproofing

The side and top panels have a rubber seal that keeps water from seeping through into the interior of the robot. The panels that have designs cut into them (front and rear) will also be stuck to an opaque, back-lit acrylic sheet. The lighting is implemented to help the panel designs stand out. Any holes on the top of the robot will be covered with silicone to allow the wires to go through, but to block out any water that could come in.


Figure 6: Outer robot enclosure

4. Description of electronic and power design

4.1 Overview

The robot will have a two battery system. The 24V battery system will consist of the motors, motor controllers and the light stack. The 12V battery system will consist of the computer, ZED camera, GPS, IMUs, e-stops, and the monitor. There will also be a voltage step down that converts 12V to 5V to accommodate the differences in voltage. There will be an electrical panel that will have the computer and the motor controllers attached to it. All eight motors will be on the drive train and wired to the controllers.

4.2 Power distribution system (capacity, max, run time, recharge rate, additional innovative concepts)

For our 24 volt system, we required a battery with 21.8 amperage.

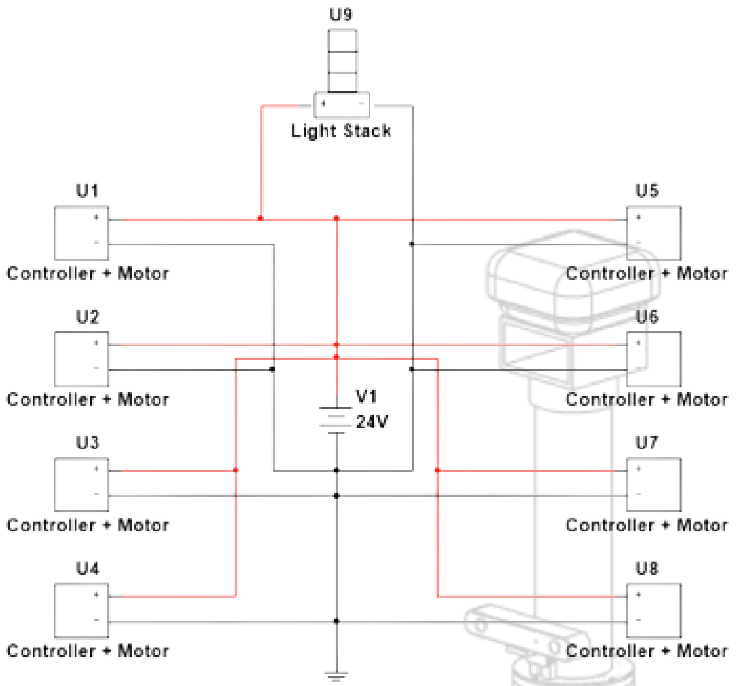
Component	Voltage	Amperage(per)	Amperage(total)
Drive Motors	24V	4.2A	16.8A
Steering Motors	24V	1A	4A
Light Stack	24V	1A	1A

Figure 7: Components Running off 24V system

For our 12 volt system, we required a battery with 17.6 amperage.

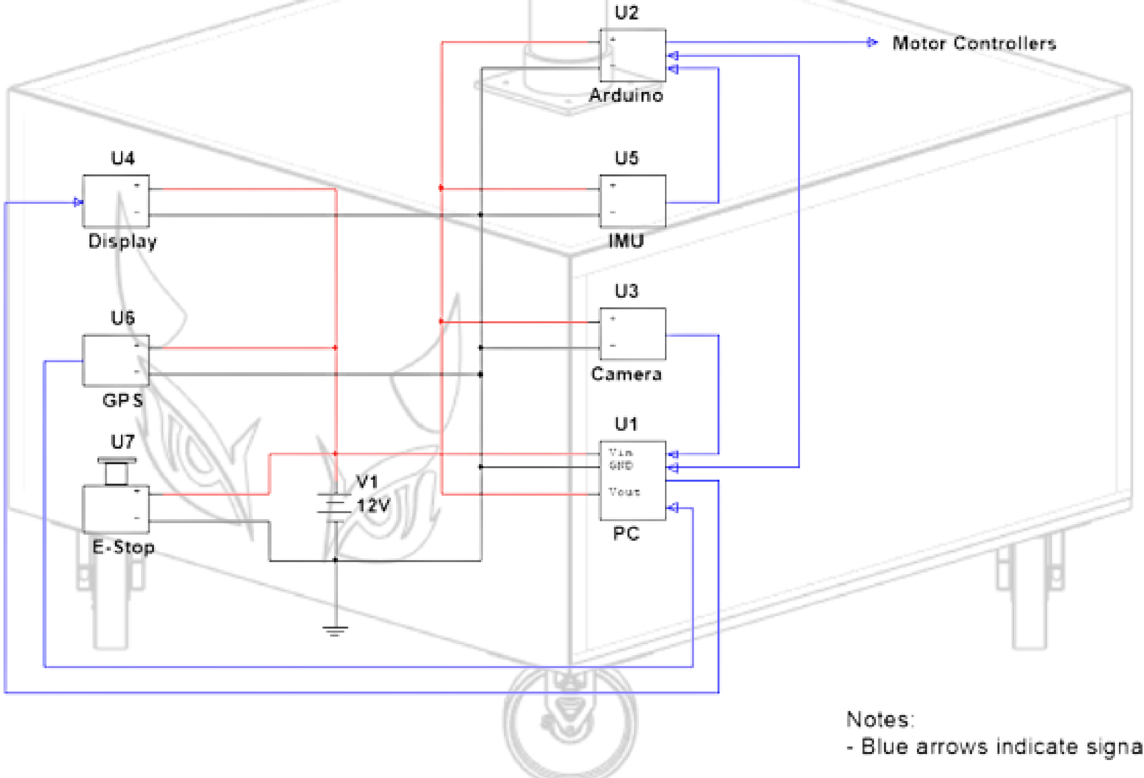
Components	Voltage	Amperage (per)	Amperage (total)
Teguar 12” Monitor	9-36V	7A	7A
PC	5V	5A	5A
Atlas GPS	7-32V	.28A	.28A
Servos	12V	2A	4A
ZED Camera	5V	.38A	.38A
E-STop	5V	1A	1A

Figure 8: Components running off 12V system



Notes:
- Detailed signal and power connections for the motors and motor controllers are shown in sheet 2.
- 24V is wired into the controllers and not the motors directly.

Figure 9: Electrical Schematic of the 24V system



Notes:
- Blue arrows indicate signal flow.

Figure 10: Electrical Schematic of 12V system

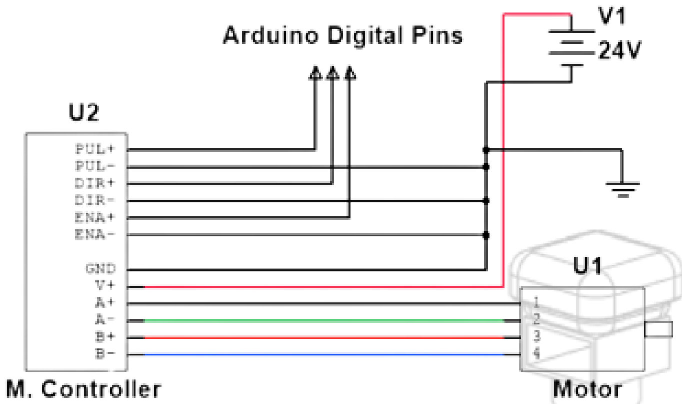


Figure 11: Electrical Schematic of motor connections

4.3 Electronics suite description including CPU and sensors system integration/feedback concepts

The GPS used will be the Atlaslink GNSS Smart Antenna. The GPS will be used to identify the current position of the robot as well as the waypoint data. To process the GPS data, the programming team has found some open source code repositories that could be modified to work with the current setup.

4.4 Safety devices and their integration into your system

The safety devices incorporated into our system are as follows: A safety light stack, as well as wireless and mechanical emergency stop buttons.

The light stack is mounted to the top of our robot for visibility. It is attached to the camera mount in order to optimize our space. We have it implemented in the code that when the robot is not in motion or in operation, the light is green, and when the robot is in motion, the light is yellow. In the case there are any fatal errors or unexpected issues, the light turns red to denote that the system needs to be stopped.

The mechanical e-stop is also mounted on the top of the robot in order to allow for easy access when necessary. It is connected directly to the power supply which allows us to cut power immediately in the event there is something wrong or the robot needs to be stopped for any other reason.

Figure 11: Light stack mount

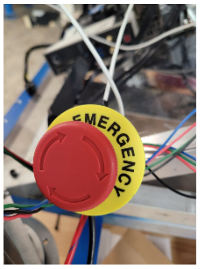


Figure 12: Mechanical E-Stop

5. Description of software strategy and mapping techniques

5.1 Overview

Our software strategy relies on our ZED depth camera and Adafruit IMU sensors. The information obtained from the sensors will be used to create a 2D cost map, which will then be used to determine a path to a goal point further down the course.

5.2 Obstacle detection and avoidance

Obstacle detection will be achieved through the ZED camera. It has the capability of detecting objects at different depths. Furthermore it can tell the distance between them by using the pixels from the camera. Using the ZED camera, it is quite easy to cross reference the data obtained from the picture camera and the depth camera. Objects will be detected by looking at groups of pixels that are the same color, and then checked by making sure the pixels within the object boundary are viewed at the same depth. The software will then determine if the obstacle is a traffic barrel or a barrier. The 2D dimensions of the obstacle will then be added to our map.

In order to handle lane following, the ZED camera will also be utilized. Its ability to analyze the environment and send that information to the PC in high-resolution images is unparalleled. It will easily be able to handle obstacle avoidance as well as stay inside the designated area.

5.3 Software strategy and path planning

The software we decided to use is a Robot Operating System (ROS) & Linux based system, using ROS nodes to send information back and forth in a quick and efficient manner. The main computer will communicate with the Arduino Mega through serial communication over USB, sending signals for various controls to the Arduino Mega. The Arduino Mega will be used for controlling the 8 stepper motors on the drive as well as the one motor on the camera.

5.4 Map generation

We will be using a ROS package that provides a 2D cost map, which we will then use for path planning. The map will be updated as we move through the course using information from the ZED depth camera and the Adafruit IMU. The map will only contain information for a set radius around the robot, and information that has moved beyond that reach will be dumped.

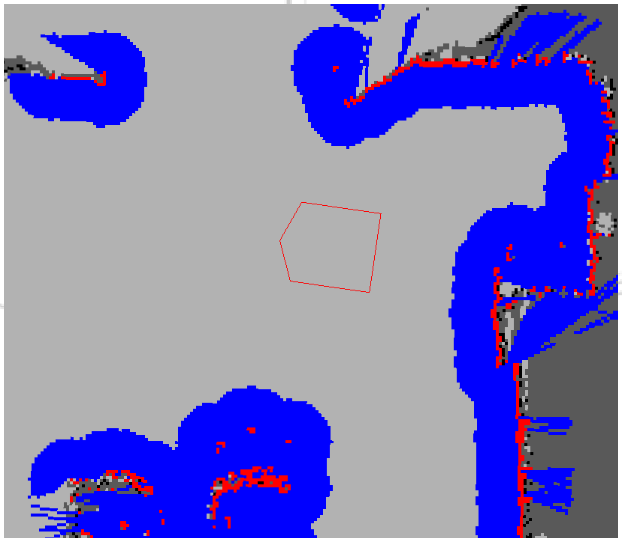


Figure 13: Cost map provided by costmap_2d. Red pixels are obstacles. Blue pixels account for the circumference of the circumscribed area of the robot and are areas that the center of the robot

cannot move through without the robot hitting an obstacle. Example robot outline shown in the middle of the diagram.

Obstacles will be detected and then matched to one of the obstacle objects, which will contain information about the dimensions, and have a variable set with the distance from the robot. The obstacle object will then be sent and projected onto the cost map.

5.5 Goal selection and path generation

Goal selection will be determined by maintaining the direction of progress along the course. Then, a goal position will be selected in the middle of the lane a certain distance from the robot. The software will then reference the cost map, and come up with a path solution that minimizes the distance traveled, while avoiding obstacles.

Using swerve drive allows for a turning radius of zero but in order to turn each wheel they need to be turned to different angles to allow for better control of the robot. The equation for determining the wheel angle assumes the center of the robot base as the outer edge of the turn. From the center of the robot is 14.5 inches both in the x and y directions to each of the drive modules.

$$\text{Drive module \#1 } Tan^{-1}\left(\frac{x-14.5}{14.5}\right)$$
$$\text{Drive module \#2 } Tan^{-1}\left(\frac{x+14.5}{14.5}\right)$$
$$\text{Drive module \#3 } Tan^{-1}\left(\frac{x-14.5}{-14.5}\right)$$
$$\text{Drive module \#4 } Tan^{-1}\left(\frac{x+14.5}{-14.5}\right)$$

Figure 14: The equation uses the forward direction of the wheel as zero degrees and x as the turning radius

6. Description of failure modes, failure points, and resolutions

6.1 Vehicle failure modes (software, mapping, etc) and resolutions

The drive module equations can get the proper wheel angles at any turning radius. except for the turning radius of 14.5 inches or -14.5 inches two of the drive modules will have an incorrect output. When the turning radius is 14.5 inches module 1 and 3 have incorrect outputs and when the turning radius -14.5 inches module 2 and 4 have incorrect outputs. To solve this problem a special case for these turning radius.

6.2 Vehicle failure points (electronic, electrical, mechanical, structural, etc) and resolutions

The main way that the structure of the robot will fail is that a weld will break. If a weld breaks, the team will patch it by welding it back together, or if a welder is not available, a plate with bolts will hold the members where the weld broke. This will in the end be stronger, but not look as pretty. Plates will be ready and some angle brackets will be ready as well to fix these problems.

Mechanically, our system is quite complex so naturally, there are many different ways that the system could fail. Mainly, the pulley inside of the gearbox breaks. The team will have extra pulleys so it can be replaced, and several members of the team have taken the gearboxes apart and back together so the reassembly should be quick. Another place where there could be failure is the gears. The gears inside of the gearbox could fail as well so we will need to have extras of each gear as well so we can replace them if needed. The traction material on our wheel could also wear out and break off. The team will have extra material on hand so they can be replaced if needed. We are also looking into using a more durable material so we do not have the concern of needing to replace them.

The electrical system poses the biggest threat in terms of failure points. Potential failure points could be a result of: faulty wire connections, overheating, insufficient power to vital components, component failure, drained batteries or a short circuit. To prevent these things, the team has made extra sure that each wire is secure in their respective connections. This reduces the possibility of wires coming loose, which could cause minor malfunctions or even short circuits. If the wires do come loose, it is usually a simple fix as long as the wire does not cross into a portion of the electrical board that would cause a short circuit. The team will also have battery chargers during competition so that the batteries can maintain a full charge. The worst point of failure would be a component failure. In other words, if a component were to completely be fried, it would not be able to be replaced in short notice.

For our software strategy, failure points mainly occur from the sensors providing bad information. If we have too much glare on the course for the camera to detect obstacles or receive a good picture, the robot will not be able to progress, nor should it. If the GPS cannot receive a good signal to determine its location within the no man's land portion of the course, it cannot navigate, and will not be able to progress. If we miss a step on the stepper motors, we can cross reference our distance travel with the IMU. However, if the IMU sends bad data, we trust it more than the stepper motors, and will have bad information within the map about obstacles along the side of the robot or behind it. Final failure points occur within our algorithms, and we plan to test them as much as possible to ensure they are correct and provide good navigation.

6.3 All failure prevention strategy

Our all failure prevention strategy is as follows:

Step one - Physical Element Inspection. This step has a two part process which includes looking over the robot and environmental issues. The robot inspection would include looking over the charge of batteries first. Then we would look for any damage done to the drivetrain or sensors. Following this we would make sure there are no loose bolts or components causing any failure. If all these pass inspection then we move onto the environment. Here we are looking for any glare specifically. Environment inspection also includes checking the temperature and humidity to ensure that the robot can function under those conditions. If any errors are found in the physical portion then we would look for backup parts to replace the broken or malfunctioning parts with.

Step two - Electrical Inspection. First part of this step is to take a multimeter and make sure every electrical component is receiving the proper voltage and current. From there we will proceed to turn on system by system to make sure everything powers on properly. If any issues are found then we would proceed to rewire the necessary components to ensure power distribution.

Step Three - Software Inspection. In this step we send our own sample code to make sure that the motors are working, as well as all sensors and components that are communicating with each other. If issues are found we can make any fixes if necessary. From there we can upload the proper code and make sure it works.

6.4 Testing (mechanical, electronic, simulations, in lab, real world, etc.)

The frame was tested using Fusion 360's simulation software under the worst-case scenario of the robot going up a 10 degree incline with just under 200 pounds of total weight distributed around it. The constraints were set so the rear two brackets with the assumption those would be the point it would pivot on going up the incline. As expected the most stress was seen around the brackets, but nothing was under enough stress to be of concern. The result of the analysis can be seen below:

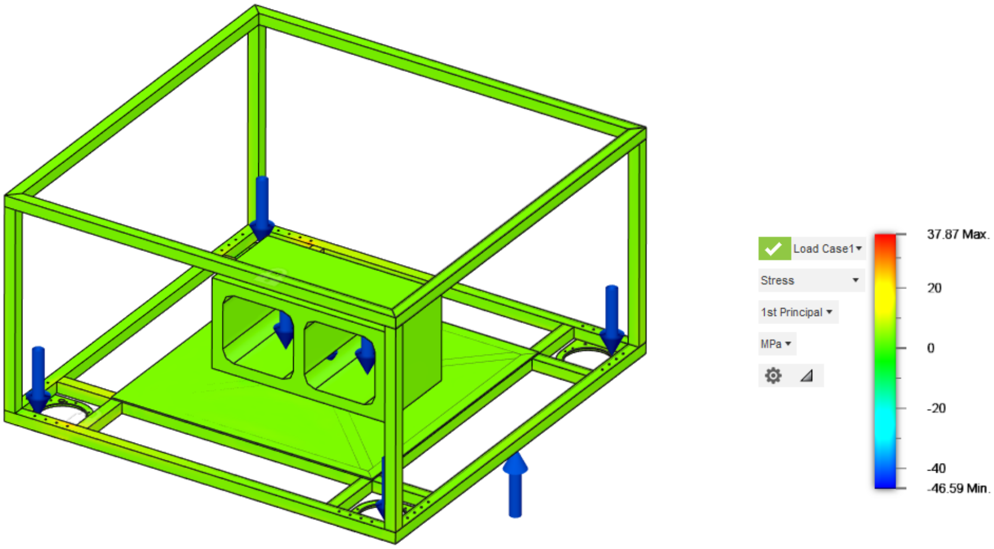


Figure 15: FEA Analysis

The frame's strength was tested in the real world by supporting the weight of a full grown adult male after the welding was finished and the drive modules were installed. This was later proven tested again once we had installed the electronics and the batteries and let the robot move on its own under that weight. As can be seen below, the batteries make up a significant portion of the weight the robot endures:

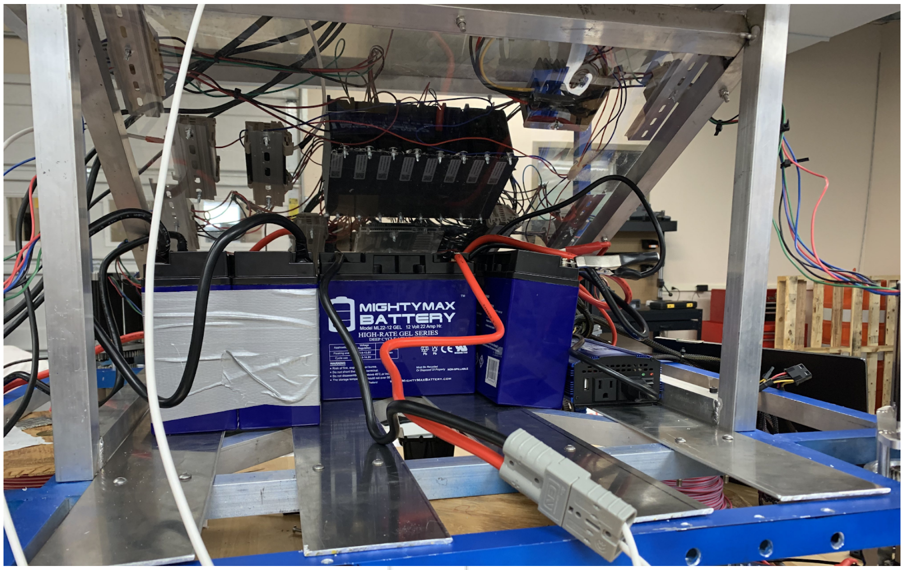


Figure 16: The batteries on the robot

The motors have gone through a multitude of testing. We have tested the motors individually, as well as testing each individual drivetrain system. This was done by just reading the correct pin readings on our Arduino Mega. Further testing with the motors will include giving an input of degrees to turn our steering motors while the robot is moving forwards.

We were able to test our motors with the wired e-stop. This was done while we were testing the motors and varying speeds they could run at. When it ran long enough, we tested the e-stop by pressing the button and it worked with cutting power to the motors.

Further testing on the electrical system included testing that everything turned on properly. The first attempt at this test went poorly and damaged a wire and motor controller. After making further modifications in adding in the jumpers into our terminal blocks we were able to safely power everything on and run.

6.5 Vehicle safety design concepts

When designing the robot, the team wanted to be sure that the robot would be safe to operate in a lab, outside, on a course, and when on display. In order to do this, the team was looking at several ways to be sure that the robot was safe in any environment that it would be in.

The main priority for the safety of the robot would be a mechanical e-stop. This will allow any member on the team to be able to shut off the robot just by running up and hitting the button.

Another safety feature that was a main priority was a light stack. This light will show different lights and signals to show what it is doing. This will allow people to see if the robot is active or is idle without a program running.

The other main feature to be employed is a wireless e-stop. This is to allow the holder of the e-stop to be able to stop the robot while it is moving and someone does not need to approach the vehicle to shut it off if something goes wrong. This is being considered by using a bluetooth keyboard for the computer to send a stop command to the main arduino. Other possibilities include using an ESP32 or Arduino with HC-05 (bluetooth) module to shut off a relay, but the keyboard control is most likely at the moment of writing this report.

A concept that the team is debating is a system that will turn the computer off or will warn the user when the computer gets too hot. This could be implemented to make sure that none of the devices overheat so there is less risk of them dying.

7. Simulations employed

7.1 Simulations in virtual environment

In testing the software, we created a simulated environment mimicking the expected course, and created a cost map of the obstacles encountered. Then, we were able to test our path planning to make sure the algorithm navigated correctly.

8. Performance Testing to Date

8.1 Component testing, system and subsystem testing, etc.

We were able to test the computer by turning it on and seeing if it can be powered by the battery system that we have on the robot. This allows us to have robot power and allow it to function without an external power source.

We have worked with our sensors in multiple tests. Our camera turns on and communicates between it and the computer. Its depth sensing capabilities have also been tested within our lab. We have also done a lot of tests with the GPS, and have established communication between it and the computer. Our GPS tests have all been conducted inside, and thus we have not gotten good results on GPS positioning.

Using the Arduino software, multiple simple tests have been done to all of the stepper motors and motor controllers, in order to confirm they are all in proper working order. Also, tests have been run with each motor running alone as well as all of the motors running simultaneously.

9. Initial Performance Assessments

9.1 How is your vehicle performing to date

As of May 15th, 2022 the robot has individual components and individual systems communicating with each other, but the robot as a whole does not communicate with each other. The arduino and motors talk with each other, and the computer and the depth camera communicate with each other. The motors are capable of turning and moving the robot. This includes being able to control the speed of the robots and tell it a position to rotate to. Though it still needs to communicate with the depth camera to take in input for when to turn, and how much to turn the wheels. The depth camera can send signals and can see depth. Though we are still trying to get the 2D mapping to work. The GPS is not currently working due to the program required to run the GPS giving us issues. Everything powers up properly and is wired. The next steps will be to polish up individual component's operations and then get them to communicate with each other.