Georgia Institute of Technology
RoboJackets

Swervi

Project Manager
Charles Li | cli651@gatech.edu
Mechanical Subteam Lead
Nicholas Vellenga | nvellenga3@gatech.edu
Electrical Subteam Lead
Indraja Chatterjee | ichatterjee9@gatech.edu
Software Subteam Lead
Vivek Mhatre | vmhatre3@gatech.edu

Members

Mechanical

Matthew Fernandez	mfernandez64@gatech.edu
Elizabeth Goetz	egoetz6@gatech.edu
Gabriel Gutierrez-Ruiz	gabrielg@gatech.edu
Waid Dunlop	wdunlop3@gatech.edu
Rohan Ravikanti	rravikanti3@gatech.edu
Yohan Venkateswaran	ysv3@gatech.edu
Joshua Frost	jfrost36@gatech.edu
Brayden Reaney	breaney3@gatech.edu
Daniel Ehret	dehret6@gatech.edu
Hayden Johnson	hjohnson84@gatech.edu
Bryan Yembiline	byembiline3@gatech.edu
Marvin Ren	mren36@gatech.edu
Findlay Townsend	ftownsend3@gatech.edu
Ashima Taneja	ataneja34@gatech.edu
Yongjae Won	ywon30@gatech.edu
Tomas Osses	tomas_osses@gatech.edu
Cameron Loyd	cloyd6@gatech.edu
Daniel Kilgore	dkilgore8@gatech.edu
Alexander Shih	alexander.shih@gatech.edu

Electrical

Andrew Roach	aroach34@gatech.edu
Somil Joshi	somiljoshi@gatech.edu
Hong Ze Khor	hkhor6@gatech.edu

Software

Matthew Hannay	mhannay3@gatech.edu
Aidan Stickan	astickan3@gatech.edu
Priyanka Rajan	prajan31@gatech.edu
Calix Tang	calix.tang@gatech.edu
Kaylia Mai	kmai30@gatech.edu
Sunidhi Dhawan	sunidhi@gatech.edu
Maya Pillarisetti	mpillarisetti3@gatech.edu
Matthew Kerner	mkerner3@gatech.edu
David Gorin	dgorin6@gatech.edu
Henry Liao	hliao62@gatech.edu

Faculty Advisor
Frank L. Hammond III | frank.hammond@me.gatech.edu

May 10th, 2022

1 Who We Are

1.1 Introduction

RoboNav is a team within RoboJackets, an organization within Georgia Tech that contains a wide variety of robotics teams. Our team is focused on autonomous navigation, and as such, we chose IGVC as a competition. Because our previous robot, Jessiii, was the third iteration of its original design, we decided to create a new robot from scratch for this year's competition. Inspired by the swerved-drive-driven robots featured in other robotics competitions, we decided to apply it to IGVC and build a robot with a swerve drive. With this robot, Swervi, we hope to succeed in IGVC through successful implementation of this new technology combined with the lessons learned from the previous year's competition.

1.2 Organization

Our team consists of three subteams: mechanical, electrical, and software. Each subteam consists of numerous members, one of which is the subteam lead, who is responsible for developing and enforcing a timeline for robot work, as well as keeping a positive work atmosphere and ensuring member retention. The subteam leads are managed by the project manager, who oversees the development of the robot as a whole.

Position	Name	Standing	Major
Project Manager	Charles Li	Senior	Aerospace Engineering
Mechanical Lead	Nicholas Vellenga	Junior	Mechanical Engineering
Mechanical	Cameron Loyd	Graduate	Mechanical Engineering
	Tomas Osses	Graduate	Mechanical Engineering
	Youngjae Won	Graduate	Mechanical Engineering
	Daniel Kilgore	Graduate	Aerospace Engineering
	Elizabeth Goetz	Graduate	Mechanical Engineering
	Alexander Shih	Senior	Mechanical Engineering
	Matthew Fernandez	Freshman	Mechanical Engineering
	Gabriel Gutierrez-Ruiz	Freshman	Mechanical Engineering
	Waid Dunlop	Freshman	Mechanical Engineering
	Rohan Ravikanti	Freshman	Aerospace Engineering
	Yohan Venkateswaran	Freshman	Mechanical Engineering
	Joshua Frost	Freshman	Mechanical Engineering
	Brayden Reaney	Freshman	Aerospace Engineering
	Hayden Johnson	Freshman	Biomedical Engineering
	Bryan Yembiline	Freshman	Mechanical Engineering
	Ashima Taneja	Freshman	Computer Science
	Marvin Ren	Sophomore	Mechanical Engineering
	Findlay Townsend	Sophomore	Mechanical Engineering
Electrical Lead	Indraja Chatterjee	Sophomore	Computer Engineering
Electrical	Andrew Roach	Sophomore	Math & Comp. Science
	Somil Joshi	Senior	Electrical Engineering
	Hong Ze Khor	Sophomore	Electrical Engineering
Software Lead	Vivek Mhatre	Senior	Computer Science
Software	Matthew Hannay	Junior	Computer Engineering
	Aidan Stickan	Freshman	Computer Science
	Priyanka Rajan	Freshman	Computer Science
	Calix Tang	Freshman	Computer Science
	Kaylia Mai	Freshman	Computer Science
	Sunidhi Dhawan	Freshman	Computer Science
	Maya Pillarisetti	Freshman	Computer Science
	Matthew Kerner	Freshman	Aerospace Engineering
	David Gorin	Freshman	Computer Science
	Henry Liao	Freshman	Computer Science

1.3 Design Process

The design of Swervi began immediately after the 2019 competition, and the team put a rough total of 4600 hours working on it.

Our design process for Swervi started by analyzing the limitations of Jessiii. While the differential drive design of Jessiii was quite simple, it limited Jessiii's mobility and maneuverability. As a result, Swervi was designed from the ground up with the goal of incorporating swerve drive to eliminate any mobility and maneuverability limitations. The design of Swervi underwent multiple design reviews and each review helped incorporate the lessons learned from Jessiii's performance in the previous competition.

2 Innovations

2.1 Swerve Drive

On Swervi, swerve drive control offers independent driving and steering for each wheel, affording the robot two translational degrees of freedom. This allows for a higher level of speed and maneuverability compared to other drivetrains such as Ackermann steering, differential drive, and omni wheel drive. A comparison of the mobility of the differential drive used in previous robots and the swerve drive is shown in Figure 1. Swervi utilizes four independent four swerve modules, as shown in Figure 2. Each swerve module has two motors: a drive motor to roll the wheel along the ground in a single direction, and a steering motor to change the direction the wheel is facing. While this model offers greater maneuverability, swerve drive requires a more complex control algorithm to adjust the speed of each wheel's drive motor and steering angle position for the robot to reach its target position.

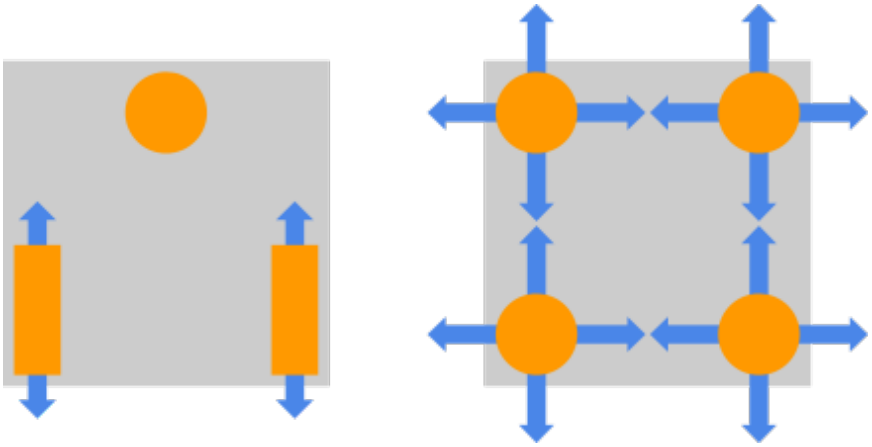


Figure 1: Comparison between differential drive (left) and swerve drive (right). Arrows indicate potential drive directions for each wheel (orange).

Figure 2: Single swerve module

2.2 Swerve Drive Wheel Odometry

Given the new swerve drive model of Swervi, a new wheel odometry model needed to be developed. Since true nonholonomic swerve drive is an over constrained system, there are multiple solutions to a given position and orientation. This means that there is no unique body position and orientation that exists given a set of wheel angles and velocities. In order to track the wheel odometry of Swervi, the instantaneous center of rotation (ICR) of the chassis was utilized to describe the motion. This approach is further explained in section 5.4.

3 Mechanical Design

3.1 Overview

The design approach for Swervi resembles that of past years, but differs in the extent of design alterations: the team analyzed previous designs for their strengths and weaknesses, proposed alternative solutions to the problems identified, selected the most desirable design, modeled it in CAD, reviewed the design to catch any potential issues, and proceeded to manufacture and assemble the machine to specification.

The robot contains four main mechanical subassemblies: the chassis, the swerve modules, the cover, and the sensor tower. The chassis acts as a central supportive structure to which the other assemblies are mounted. The swerve modules make up the drivetrain, enabling robot motion. The cover protects the electrical tray from the weather. The sensor tower holds various position, inertial, and photo optic sensors at a high vantage point above the robot.

3.2 Structure

The chassis's geometric form accomplishes a number of key structural tasks. The flat upper section connects to the base plate, which holds the electrical tray and cover, and the sensor tower support, which fixes the sensor tower mast in place. Below this flat section, the swerve modules are mounted at each corner, offset by a few inches to prevent the chassis from scraping the ground when traversing the apex of a ramp. The undercarriage contains dedicated compartments for the battery, which is walled in and latched shut with a hinged door, and for the payload, which is secured by aluminum bars fastened with screws. Mounting these heavy objects close to the ground lowers the center of mass to improve stability.

The chassis must support all forces exerted on it without deflection to maximize the accuracy of Swervi's motion, and so the construction material of choice was 1in (2.5cm) thick 80/20 Inc. aluminum T-slot extrusion. The material is ideal for its relative cost efficiency, modularity, ease of assembly, and high strength. Most of the members are attached to each other using end fasteners that facilitate reliable butt joinery, with small L-brackets used in locations where geometry made such a solution infeasible. All other connections to the chassis, such as the swerve modules and base pate, were made using 80/20 Inc.'s standard T-slot nuts.

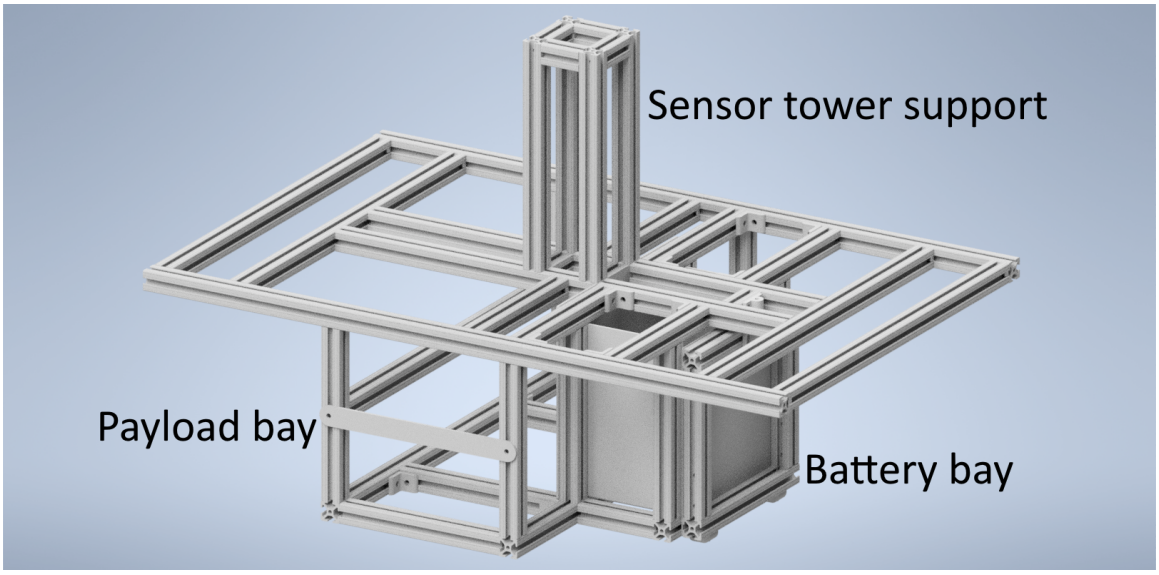


Figure 3: Chassis substructure.

3.3 Weatherproofing

The cover consists of three bent and stamped sheet aluminum sections, one fixed in the center around the sensor tower and two moving doors on front and back. These slide can be opened on drawer slides to allow access to the electronics, or, if needed, removed completely. Two latches on each door prevent water ingress by compressing a rubber strip placed at the interface between segments.

Our sponsor Protocase provided the cover at a discount. This allowed us to outsource a significant amount of potential work to professionals and ensure the final product would be of high quality.

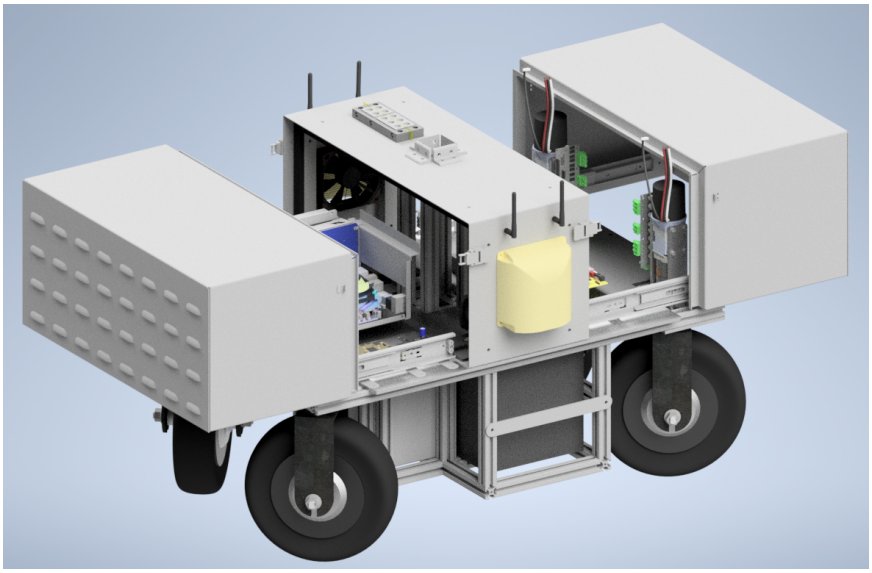


Figure 4: Cover opened to show electronics tray.

Proper heat dissipation is critical to avoid shortening the lifespans of the electrical components within. Two computer fans are mounted in the sides of the center cover section, each with a plastic cover to ensure air is drawn from below to exclude rainwater. These force cool air in and over the electronics tray and out through vents in the front and back of the robot. The vents are louvered to block rain at a wide range of angles, with their size and quantity selected to maximize airflow. Airflow is optimized when the intake surface area of the fans equals the exhaust surface area of the vents. A white powder coat on the cover increases albedo and keeps the interior cool by reflecting away more of the sunlight that hits it.

Some of the electronics mounted on the sensor tower, such as the GPS, indicator light, LiDAR, and emergency stop button, are sold as waterproof, but the IMU and cameras each require a rain shield incorporated into its 3D printed mount.

Figure 5: Sensor tower with mast and mounting bar.

3.4 Sensors

The sensor tower consists of a 2in (5cm) mild steel box extrusion mast which raises the top surface of the sensor mounting bar, a piece of 80/20 T-slot aluminum, nearly five and a half feet (1.7m) off the ground. The raised platform of the sensor tower provides the GPS access to a better signal, the cameras a better viewing angle, the indicator light better visibility, and minimal electromagnetic interference for IMU readings. The LIDAR is mounted on a separate plate aluminum bracket about 4ft (1.2m) above the ground, aimed in a downward direction to focus identification of obstacles between 3ft (0.9m) and 18ft (5.5m) in front of the robot. The emergency stop button box also lives on the sensor tower mast for ease of access.

3.5 Drivetrain

A 10in (25cm) diameter E-Tech brushless DC hub motor drives each swerve module. Collectively, the swerve modules have a max speed of 25mph (40km/h) and max load of 330lbs (150kg) per wheel. The aforementioned

capabilities of the swerve modules are well above what is necessary for the 15° slopes outlined in the competition rules, however what makes each swerve module unique is their compact form factor. Each swerve module is only 6 3/4in long and 5in wide, making it easy to fit all four swerve modules in Swervi's chassis. Originally designed for use in electric scooters, the hub motors eliminate the need for an auxiliary gearbox, affording the swerve modules a small footprint while still outputting sufficient power. By using Bipolar Nema 17 stepper motors with integrated incremental encoders, we ensure precise angling of the wheels to adhere as closely as possible to the target trajectory determined by the software stack.

4 Electrical Design

4.1 Overview

Swervi's electrical system boasts several significant improvements over its predecessor, including a longer battery life, a cleaner layout of electrical components and wires, and better sensors. We also made it easier to modify Swervi's computer by replacing the robot-mounted computer monitor with a remote desktop system. To successfully implement swerve drive, our team chose new actuators and their corresponding control systems in addition to repositioning sensors to accommodate the 360 degrees of mobility. We expect that these design decisions will enable Swervi to maintain an increased average speed with improved reaction times compared to Jessiii.

4.2 Power Distribution

Swervi is powered by a 24V 30Ah lithium ion battery. This was a much-needed upgrade over Jessiii's lead-acid batteries. The lithium ion batteries have resulted in quicker charging times, fewer maintenance issues, and an astonishing 8 - 10 hours of battery life during standby in addition to having a smaller physical footprint.

Swervi has a 24V power rail and a 12V power rail. A few electrical components, such as the cameras and the IMUs, are powered by USB via Swervi's onboard computer. Electrical components attached to the power rails are protected by fuses, which help prevent permanent damage in the case of excess current. Notably, all of the motors' power is routed through a solenoid, which can be triggered either by the red button on the mast panel or by a remote switch. The solenoid functions as Swervi's hardware-based E-Stop; once the solenoid is triggered, all of the robot's actuators are immediately disconnected from the 24V power rail.

4.3 Computer and Microprocessors

Since we switched from an Intel NUC and an Nvidia Jetson to a custom-built computer in 2019 containing an Intel Core i7-8700 CPU, a NVIDIA GeForce GTX 1060 GPU, and 32GB of RAM, our software stack has not been hindered by the limitations of our current computer hardware. As a result, Swervi's main computer has remained unchanged since the 2019 IGVC.

The primary microprocessor on Swervi is the MBED LPC1768. The LPC1768 is responsible for processing speed and rotation commands provided by the computer and returning encoder estimates of the eight motors back to the computer. The LPC1768 also communicates with several miniboards for diagnostic information.

Figure 6 displays how signals are processed on Swervi. Components that pass a lot of data, such as the computer, the LPC1768, and the LiDAR, are on a Local Area Network and communicate using TCP. Then, the LPC1768 exchanges information with the motor controllers and the diagnostic miniboards over a CAN Bus.

Swervi also has a router onboard, which allows us to remotely access Swervi's onboard computer. Remote access has been more convenient and has saved battery life compared to mounting a computer monitor on the robot. The router can be configured for both LAN access and WAN access, meaning that one could edit and update Swervi's software stack without being physically present.

4.4 Motor Control

The E-Tech brushless DC hub motors are controlled by an Odrive brushless DC motor controller. The Odrive motor controller has a wide variety of settings and capabilities that allows for precise control. One such capability is a built-in PID controller. In the past, we had to implement a PID controller on the LPC1768 to

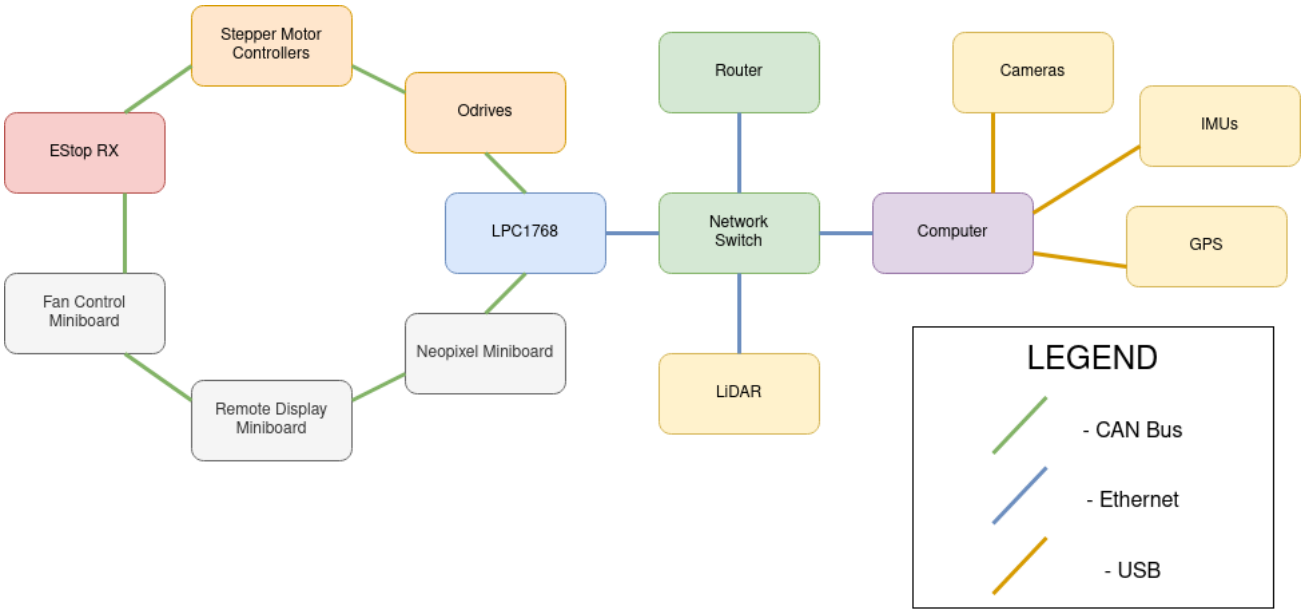


Figure 6: Diagram showcasing signal processing between modules

control the velocity of the drive motors. Now, the LPC1768 can simply provide the Odrive a velocity and the Odrive will perform the PID calculations itself. Another desirable capability of the Odrive motor controller is its wide variety of peripherals. We used the Odrive's USB connector for testing and fine-tuning and the CAN peripheral for normal operation. All in all, the Odrives reliably manage the speed of the E-Tech hub motors with the help of the motors' built-in hall effect encoders and can be accessed over the CAN Bus by the LPC1768.

The Nema 17 Stepper motors are controlled by a generic bipolar stepper motor controller. Stepper motors excel at holding their position, which makes it ideal as a steering motor. An AVR microcontroller listens on the CAN bus for position commands from the LPC1768 and translates those into pulse commands to step the steering motors.

4.4.1 Sensors

Swervi utilizes several different sensors for obstacle detection and localization. The Velodyne Puck VLP-16, a 3D LiDAR, is used for primary obstacle detection. Three Logitech c920 1080p cameras on the top of the robot are used for detecting lines.

For localization, Swervi uses the Hemisphere R330 receiver with an A21 antenna for the GPS. The GPS sensor enables Swervi to make a long-term goal based on the given GPS waypoints. A combination of wheel encoders and two IMUs are used for estimating Swervi's state in the short term. We use two Hillcrest Labs FSM-9 IMUs, one on the mast to minimize electromagnetic interference for magnetometer readings, the other between the motors to minimize lever arm effect on accelerometer and gyroscope readings.

The Odrives report encoder estimates based off of the drive motors' hall effect encoders, and the steering motors primarily use the integrated incremental encoder in the Nema 17 stepper motors. However, an AMT132 incremental encoder with a index signal is used for finding the steering motors' "home" position. When Swervi powers on, the robot doesn't know exactly where each wheel is pointing. Once the index signal is found, then the steering motor controller can point all of the wheels in the same direction, preparing Swervi for nominal operation.

The software section goes into detail of how each sensor plays a role in making control decisions, including localization, mapping, and obstacle detection.

4.5 Safety Devices

The wireless E-Stop module is another custom designed PCB on the robot. For this iteration, while the functionality of the system remains with all E-Stop circuitry on the PCB, the design was changed to integrate with the greater diagnostic system, improve reliability of the wireless E-Stop, and improve user experience. The previous design relied on having two sister boards only differing in what components were populated and the

loaded program, whereas this iteration separated both the transceiver and receiver design into two separate PCBs to create a more compact design for both. The receiver board is able to trigger the 24V signal used in the E-Stop, connect to the diagnostic CAN network, and connect to an RF antenna mounted on the front of the robot. The transceiver board is able to connect to the receiver through the same RF module and placed within a 3D printed case with an external push button to trigger the wireless E-Stop and external LEDs to indicate the status to the user. Similarly to the previous iteration, an ATmega328p microcontroller is responsible for generating and receiving E-Stop messages and processing CAN messages. The wireless component is handled with a RFM95W LoRa module running at 900 MHz for greater range and reliability and interfaces with the ATmega328p through SPI.

Figure 7: Left, RF transceiver. Right, RF receiver.

In order to integrate with the mechanical E-Stop, each system does not override each other. If either physical switch is enabled or the remote is enabled, the robot will enter the E-Stop state. This ensures that in order to run the robot, both the mechanical and wireless E-stops are in agreement that it is safe to run the robot.

In addition to the mechanical and wireless emergency stop, Swervi is also equipped with a safety light that indicates the operating mode of the robot. Upon powering up, the yellow light will illuminate, indicating that the computer is powering up. Once the computer connects to the lower-level hardware, the green light will illuminate. When Swervi shifts into autonomous mode, the green light will flash. Upon triggering the E-Stop, the red light will illuminate.

4.6 Mini Boards

Swervi contains three "miniboards" that provide diagnostic information. These miniboards include the NeoPixel miniboard, the Remote Display miniboard, and the Fan Control miniboard. Although these miniboards aren't essential to the core functionality of Swervi, they help make Swervi easier to use and a bit safer.

The NeoPixel miniboard controls a NeoPixel light strip. The NeoPixel light strip acts as an advanced status indicator. For instance, the Odrive motor controller has various errors that are hard to diagnose from a distance. In response to an Odrive erring, the Neopixel strip can flash a special pattern to display what type of error occurred.

The Fan Control miniboard drives two PC fans on either side of Swervi. This board helps keep the electrical tray cool to give a slight boost in power efficiency. In addition, the Fan Control Miniboard tracks the temperate inside the robot.

While the status lights serve as the primary method of receiving diagnostic information, the Remote Display miniboard serves as an auxiliary method of obtaining low-level information about Swervi. The miniboard receives diagnostic information from Swervi's E-Stop radio and displays it on an LCD screen. Since the E-Stop board is connected to the CAN bus, the Remote Display miniboard can report the status of any low-level module on the CAN bus.

4.7 Electrical Design Process

The beginning of our electrical design process for Swervi began in CAD. Most of our boards did not need to be designed from the ground up; more rather, our job for this year was to remove design flaws from and add CAN functionality to all of our custom electronics boards. We planned on using a CAN Bus because of the ability to easily add or subtract nodes from the CAN Bus.

One of the most crucial components of Swervi to develop and test early on were the swerve modules. Because the mechanical team would need time to construct Swervi's chassis, we used a smaller test rig comprised of two swerve modules and two caster wheels. The biggest problem we ran into with our original design of the swerve modules was the inability to determine the absolute position of the steering motors. We solved this problem by adding the AMT132 encoders that had an index signal we could use to calibrate the position of the steering motors upon startup.

Another critical component to Swervi was her emergency stop (E-Stop) system. For the most part, we worked on improving upon the functionality and reliability of our previous E-Stop system on Jessiii. To improve the signal strength of the wireless E-Stop radio on Swervi, we attached external antennae mounts to the sides of Swervi.

One major shortcoming of Jessiii was the poor arrangement of her electrical tray. Swervi's electrical tray was designed with accessibility and organization in mind. To accomplish this, we avoiding stacking modules on top of one another and used the geometry of Swervi's interior to space out modules. We also used cable ties to organize wires and to prevent clutter. As a result, we could comfortably fit all of our modules while having easy access to each one.

Once the individual modules were designed, built, and tested individually, we integrated the modules together on the CAN Bus. Although debugging communication issues on the CAN bus was challenging, we found the CAN Bus to be much more flexible compared to past communication arrangements. The CAN Bus allows for each node to operate independently from each other without direct management from the LPC1768.

The final steps of the electrical design of Swervi involved integrating Swervi's electrical system with Swervi's software stack. One major step was to ensure that requests from the computer were handled by the LPC1768. We also had to setup the computer for Swervi's sensors and configure the router for remote access. Once Swervi's computer could interface with the lower-level hardware and sensors, software could move from simulation into the real world.

5 Software Strategy

5.1 Overview

For Swervi, we've continued to use the Robot Operating System (ROS) framework, an open-source robotics middleware suite, for the software stack. ROS allows for a modular system and functionality is broken up into independent processes called nodes. Each node in ROS communicates which other nodes using TCP (Transmission Control Protocol) which allows for asynchronous message passing. In the software stack, nodes are divided into five different groups: Sensors, Perception, Navigation, Localization, and Controls.

A rough outline of the RoboNav software stack is shown below in Figure 9. As illustrated, Swervi has five different sources of data: LiDAR, cameras, GPS, wheel encoders, and IMU. To construct a map of the surrounding environment, Swervi utilizes its cameras to map line boundaries and its lidar to construct a heightmap and perform a traversability analysis. The resulting camera and LiDAR information are passed to the mapping module where a costmap is constructed. Using the generated costmap, Swervi performs path planning and time optimal trajectory to calculate motions for each wheel.

5.2 Obstacle Detection

Swervi identifies obstacles in its vicinity by utilizing LiDAR and vision based perception strategies. Obstacles are then mapped onto the traversability map and marked as untraversable. Using the updated traversability map, the path planning module finds an optimal path for Swervi to reach its destination.

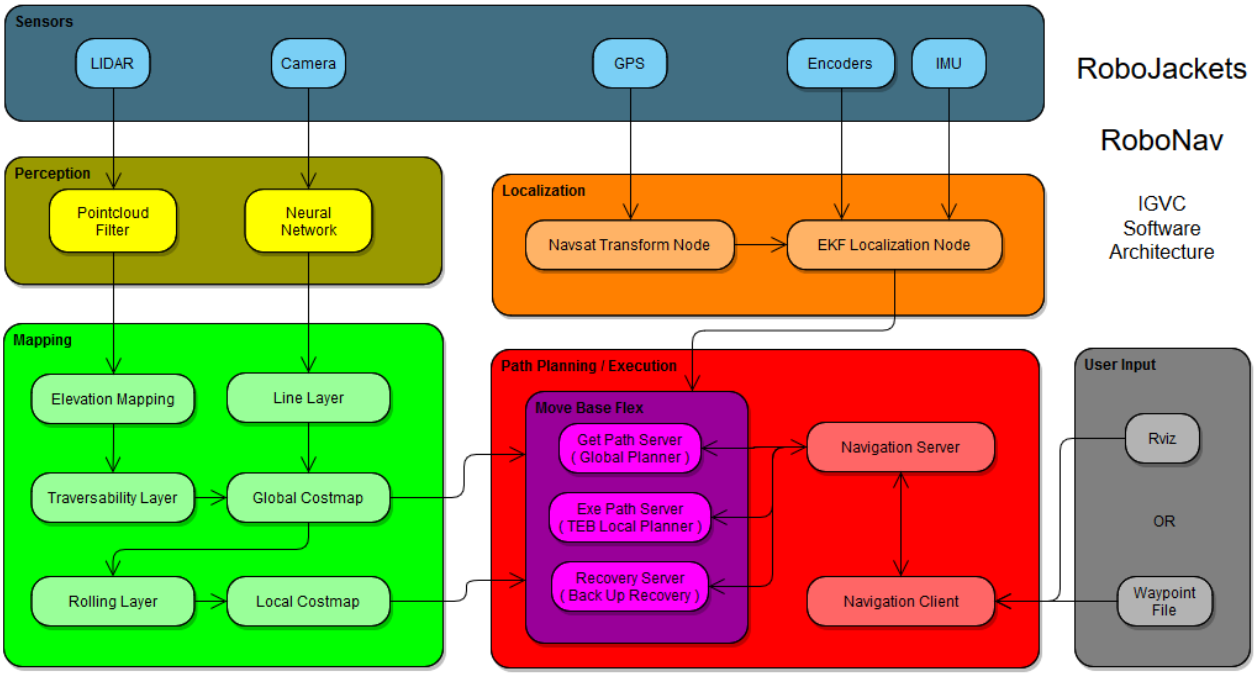


Figure 8: RoboNav Software Architecture outline.

5.2.1 Vision-based Perception

This year, Swervi's cameras were upgraded to new wide angle cameras to accommodate for its holonomic motion. Like its predecessor, Jessiii, Swervi still has three cameras. However, each of Swervi's cameras now have a 120 degree field of view. Thus, when combined the three wide angle cameras offer Swervi a 360 degree field of view which eliminates all possible blindspots. The camera configuration on Swervi is shown in Figure 10. Given the lack of blindspots, Swervi can instantly move in any direction all while avoiding obstacles.

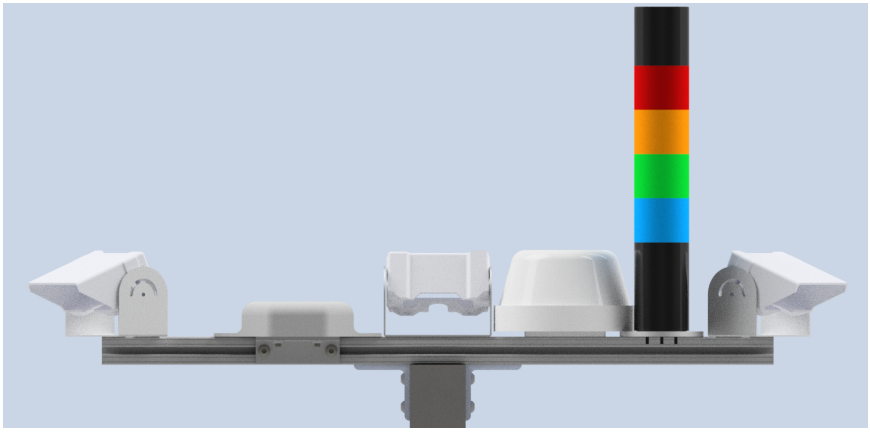


Figure 9: Swervi camera configuration.

In order to map obstacles detected in images onto the traversability map, each of the new cameras went through camera calibration. Camera calibration is the process of estimating the characteristics of a camera. By knowing a camera's characteristics, one can determine an accurate relationship between a 3D coordinate in the real world and the corresponding 2D coordinate in the camera image.

To detect obstacles in images, Swervi performs multiclass semantic segmentation classifying the pixels in each image as a line, barrel, or neither. The segmented image is then integrated into the traversability map accordingly. The architecture of the neural network is a U-Net with a pre-trained EfficientNet [3] encoder. By utilizing a pre-trained EfficientNet encoder, our model is able to extract important features from raw images and

take advantage of pre-existing datasets containing millions of images such as ImageNet. The effectiveness of the EfficientNet encoder for feature extraction combined with the accuracy of the U-Net decoder for segmentation map reconstruction allows our model to require fewer training images and produce more precise segmentations.

The segmentation map output is then projected onto an assumed flat ground using basic camera geometry. While the flat ground plane assumption may not be correct, our mapping module utilizes probabilistic mapping and increases the covariance on further points due to projection error to correct for errors introduced by this assumption.

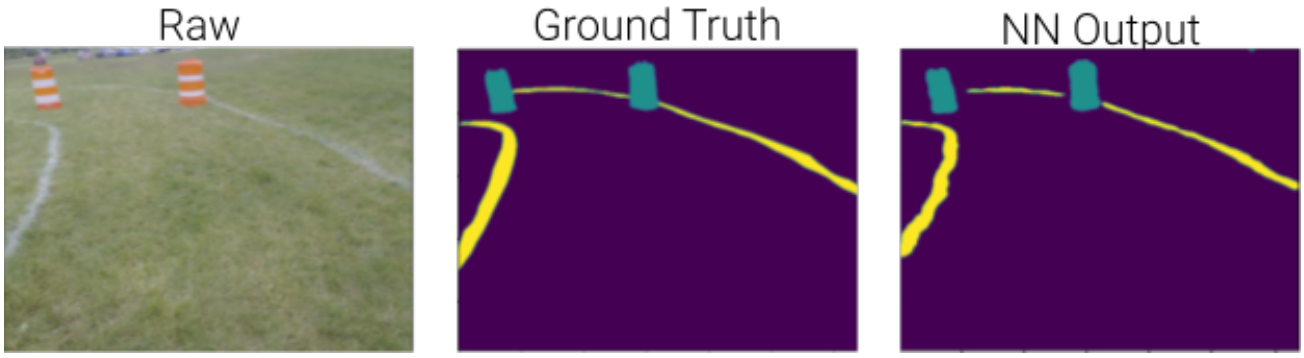


Figure 10: Visualization of Multiclass Segmentation Results.

5.2.2 3D LiDAR

This year, we made a few improvements to our existing traversability analysis. When testing, we found that abrupt orientation changes introduced quite a bit of noise in the LiDAR readings which in turn produced an incorrect traversability analysis. The 3D LiDAR on Swervi has a range of 100m and a 30 degree vertical field of view which allows Swervi to perform a traversability analysis.

To address this issue, after mapping the 3D environment around Swervi with a heightmap and performing a traversability analysis, we removed noise by iterating over the traversability map and performing morphological openings. By applying morphological openings, our mapping module is now able to produce clean traversability maps which allows for a single unified framework to handle uneven terrain, ramps, and barrels.

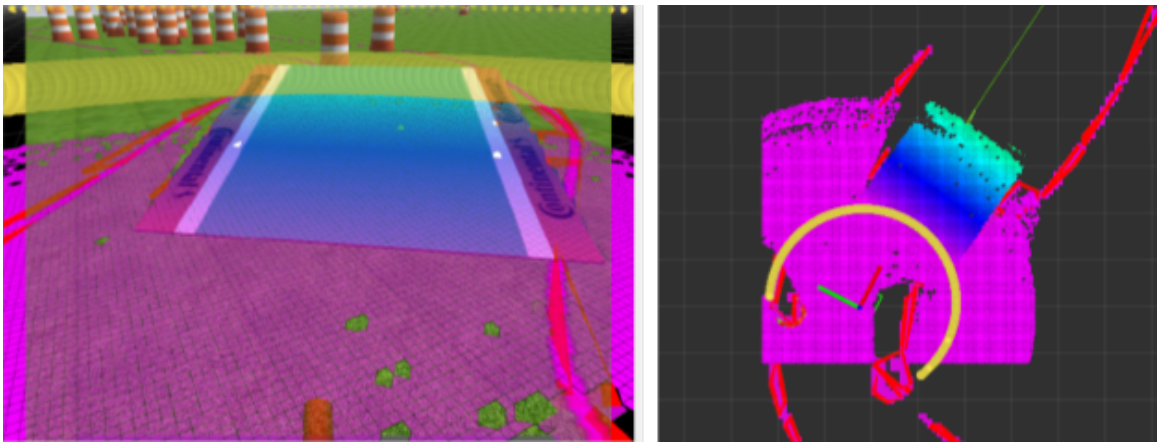


Figure 11: The traversability analysis and the resulting traversability map.

5.3 Path Planning and Software Architecture

To handle navigation to waypoints, a navigation client sends the waypoint to the navigation server which handles navigating Swervi towards the waypoint until it reaches its destination. A tolerance of 20 centimeters is given upon reaching the destination waypoint, and subsequently the next waypoint is sent to be processed.

When the navigation server first receives a waypoint it passes the waypoint to `move_base_flex`'s `get_path` client which returns a global path to the waypoint. The global path is then sent to `move_base_flex`'s `exe_path` which then generates a local path and executes it. After a specified interval the path is planned again by sending the goal to `get_path`, and a new local path is generated using `exe_path`. The replanning cycle allows for the discovery of new optimal paths and continually repeats until the goal is reached.

If `exe_path` fails or the robot is unable to make progress for an extended period of time, the recovery behavior is activated. Currently, the robot's recovery behavior is to back up as far as possible. After the recovery behavior has been executed, a new path is calculated by sending the goal to `get_path`.

In the `exe_path` client of `move_base_flex`, local paths are generated using the `teb_local_planner`. When generating local paths, `teb_local_planner` uses timed elastic bands which locally optimizes the trajectory of the robot with respect to trajectory execution time, separation from obstacles and compliance with kinodynamic constraints at runtime. By using timed elastic bands, Swervi is able to stay a safe distance away from any obstacles while following the global path.

A simplified image of the navigation state machine is shown in Figure 13. In the diagram the ellipses at the bottom represent final states, blue rectangles are actions, and the bottom rectangles describe special arrows.

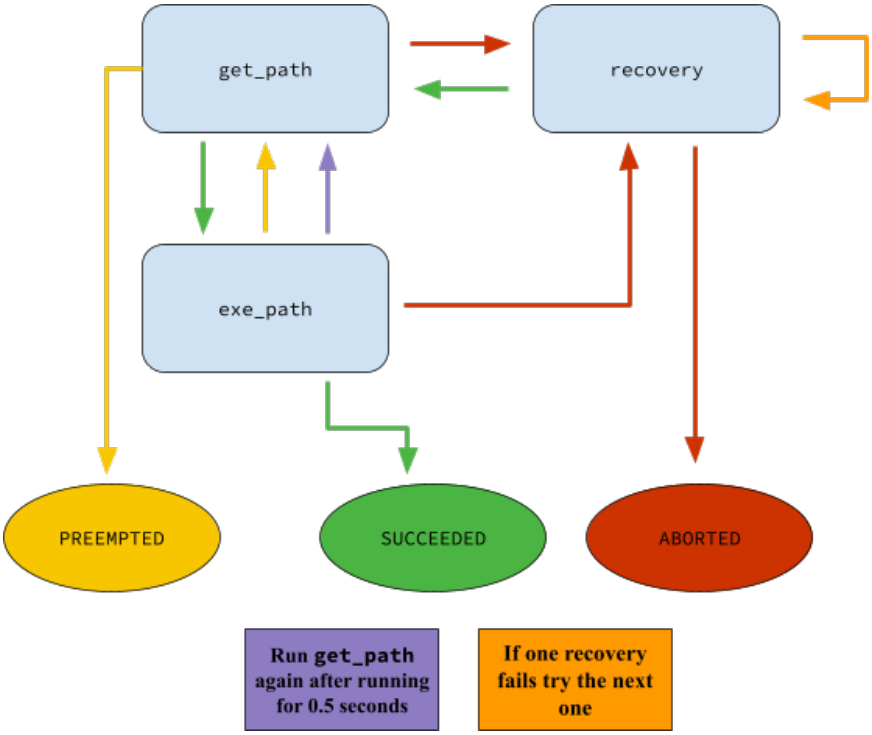


Figure 12: Swervi Navigation State Machine.

5.4 Swerve Drive Wheel Odometry

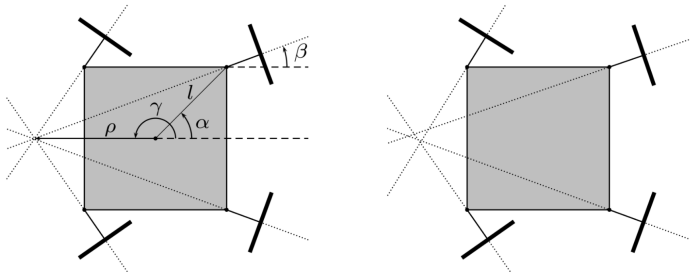
With the introduction of swerve drive, a new odometry model was needed to track the position of the robot since true nonholonomic swerve drive is an over constrained system. Thus, a new odometry module was developed using the instantaneous center of rotation (ICR) of the chassis. The ICR is defined as being the point at which the frame of the robot rotates around and can be used to describe the motion of a robot [2].

Consider a typical wheel. A wheel is forced to have a zero lateral motion about its axis due to the sliding constraint of the wheel. To visually illustrate the sliding constraint, a "zero motion line" can be drawn through the propulsion axis of the wheel which represents the direction that the wheel cannot move in. Thus, each wheel moves along some circle with radius R such that the center of the circle is along the zero motion line. For a robot with many wheels, the intersection of the zero motion lines for each wheel is the ICR.

When the ICR is a non zero constant, the robot is rotating around a point. When the ICR is at infinity, the robot is moving in a straight line.

While the ICR wheel odometry model is drastically more complex than the previous differential drive model, ICR is quite adaptable and can track the motion of any robot. For example, while an Ackermann vehicle has a different drive train than Swervi, there exists a single ICR and thus a single solution for the motion of the vehicle.

Unfortunately, due to sensor error and noise, the intersection of the propulsion axes are not perfect which makes it difficult to determine the true position of the ICR. To solve this issue, a least squares approach is used to create an estimate for the ICR [2]. While the estimate for the ICR allows for some error, sensors such as the GPS and IMU help correct for this in the localization module.



(a) Mathematics and notation (b) Reality

Figure 13: ICR defined by the intersection of the propulsion axes.

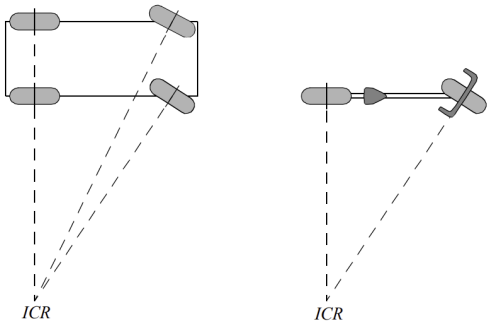


Figure 14: Left, ICR of an Ackermann vehicle. Right, ICR of a bicycle.

5.5 Swerve Drive Kinematics

Given a desired translation and rotation by the navigation module, the mapping to motor velocities for each swerve module is performed using the following equation:

$$\vec{output}_n = \vec{v} + \omega \cdot perpendicular(\vec{m_n}) \tag{1}$$

In the equation above, $\vec{v}$ is the desired translation and ω is the desired rotation given by the planner. m_n is the distance from the chassis center to swerve module n. $\vec{output}_n$ is the velocity vector for swerve module n. For each swerve module, the steering angle and velocity is determined from the corresponding velocity vector and then sent through the motor firmware. When the desired motor velocity is above the limit of the physical motor, the motor velocities are normalized to the maximum velocity. The post normalization performed by the swerve drive controller allows for optimal control outputs and full utilization of each motor.

5.6 Map Generation

When mapping its surroundings, Swervi utilizes a multilayer costmap. Each layer adds more information to the costmap, and allows for sensor fusion between our vision and LiDAR based perception. The four layers utilized in the costmap are the traversability layer, the line layer, the inflation layer, and the rolling layer. The rolling layer contributes to the local costmap and rest of the layers make up the global costmap.

As described in section 5.2.2, the traversability layer utilizes LiDAR readings to generate a heightmap and perform a traversability analysis. To generate the heightmap, we utilize the elevation_mapping library which utilizes kalman filters and covariances to accurately predict the height at each cell. Then, the elevation map is smoothed and the gradient at each point is calculated. If the gradient is too large at a cell, the cell is marked as lethal and thus is not traversable.

The line layer utilizes the segmented image produced by the multiclass segmentation model described in section 5.2.1 to probabilistically map the detected lines onto the costmap using basic camera geometry. Cells on the costmap that contain lines are marked as lethal which prevents the robot from traveling outside the line boundaries. Only obstacles within ten meters of the robot are mapped onto the costmap while the rest are ignored.

Unlike the other layers, the inflation layer doesn't add data based on sensor input. Instead, the inflation layer adds a radius to all the known high-cost cells, surrounding them with low-cost cells. By inflating the high cost cells, the global planner is encouraged to avoid going near obstacles when possible and take a safer path.

The rolling layer views the current Global Costmap and takes a square area that surrounds the robot. When extracting the square area, the rolling layer ignores any cells created by the inflation layer. This creates a local costmap that only contains obstacles.

To summarize, the Global Costmap is used for long-term path planning and the Local Costmap is used for small-scale path following.

6 Failure Points

6.1 False Positives in Neural Network

The mutliclass segmentation model produces a few false positives when exposed to extreme lighting conditions and environments not seen in training images. These false positives may result in certain sections of the map falsely labeled as lines and thus impassable. While a pretrained Efficient Net model is used as the backbone of the model, additional training data that capture edge cases is needed to further train the model and address this failure point.

6.2 Sensor Biases

Each sensor on Swervi implicitly has a small amount of error which can introduce issues in our current localization strategy. Currently, our localization strategy does not perform online bias estimation for the gyroscope or accelerometer which can result in our heading estimates drifting over time. While the IMU does perform online bias estimation of the magnetometer, this takes a non-negligible amount of time to converge. Addressing this failure point would require keeping track of biases either with another Kalman Filter, or moving to another method of localization.

6.3 Swerve Wheel Odometry Convergence

When Swervi executes a complex turn it is sometimes possible for the Instantaneous Center of Rotation (ICR) to experience a lack of convergence. When the ICR does not converge, the wheel odometry model fails to produce an estimate which can introduce a small amount of error into the localization strategy. To address this failure point the wheel odometry is assigned a high covariance, whereas more precise sensors such as the GPS are assigned a low covariance. The localization module then utlizes the covariances for each sensor measurement to account for errors such as the ICR not converging.

7 Simulations

Gazebo, a 3D robot simulator, played an integral role of the software testing and prototyping process for Swervi. This year, an entirely new model for Swervi was created in Gazebo and detailed in a Universal Robotics Description Format (URDF) file, allowing accurate modeling of the real-life dynamic interactions of the system. Gazebo plugins allowed for accurate simulation of various sensors on Swervi which enabled the entire software stack to be tested in simulation. Using the simulation, each new feature is able to be tested thoroughly in various scenarios.

8 Performance Testing

This year, we have made testing a priority and as a result our code coverage has increased drastically. Using the testing library developed last year, unit tests for any node are able to be created quickly and efficiently which allows for more rigorous testing of our software stack. In each test, a mock publisher and subscriber are created to assert that each node is able to send and receive the correct messages.

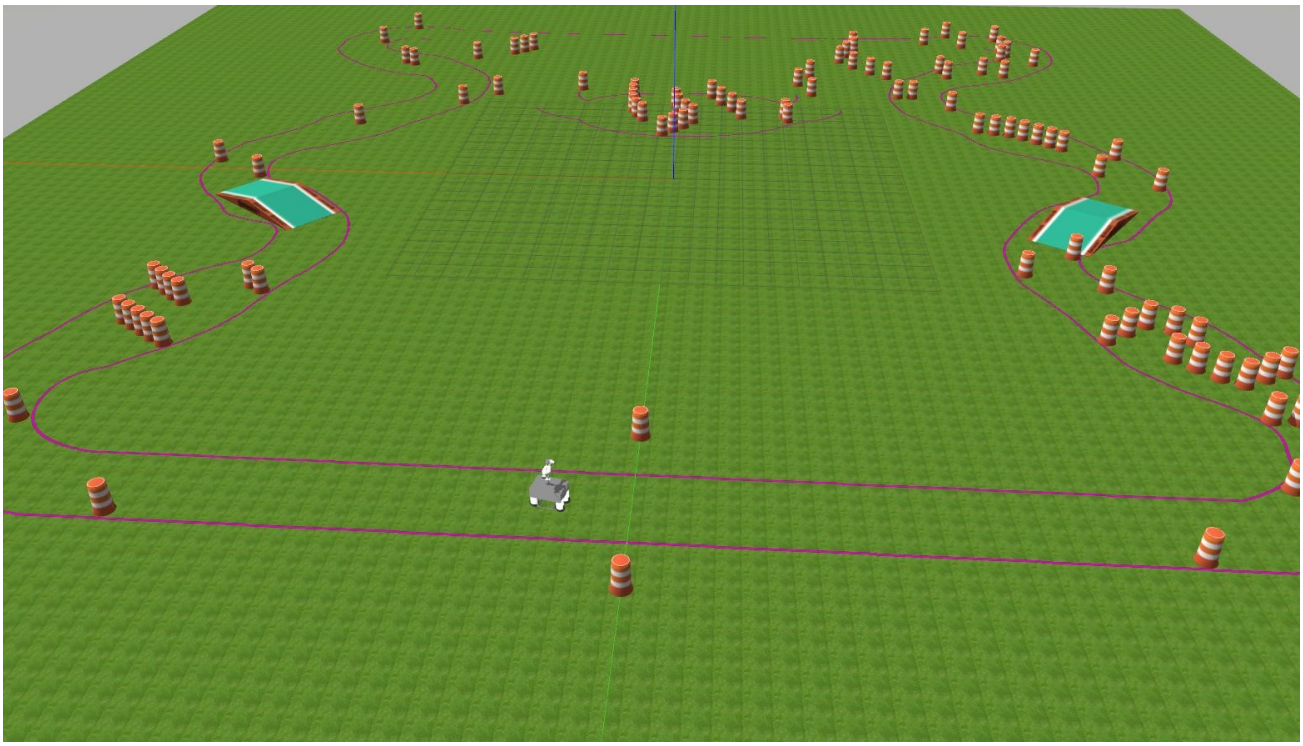


Figure 15: The simulated AutoNav course in Gazebo.

9 Initial Performance Assessments

Max Speed	4.79 mph
Acceleration	$5\,\mathrm{m\,s^{-2}}$
Ramp Climbing	25 degrees
Battery Life	10 hours standby 3 hours with motors running

10 References

[1] Clavien, Lionel, Michel Lauria, and François Michaud. "Instantaneous centre of rotation estimation of an omnidirectional mobile robot." 2010 IEEE International Conference on Robotics and Automation. IEEE, 2010.

[2] E. H. Binugroho, A. Setiawan, Y. Sadewa, P. H. Amrulloh, K. Paramasastra and R. W. Sudibyo, "Position and Orientation Control of Three Wheels Swerve Drive Mobile Robot Platform," 2021 International Electronics Symposium (IES), 2021, pp. 669-674, doi: 10.1109/IES53407.2021.9593947.

[3] Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." International conference on machine learning. PMLR, 2019.