Oakland University IGVC 2024 - Horizon

THE HORIZON INTELLIGENT ROBOTICS PLATFORM

Faculty Advisor: Dr. Jun Chen
Team Captain: Daniel Mocnik (dmocnik@oakland.edu)
Date Submitted: May 15, 2024

STATEMENT OF INTEGRITY:
I certify that the design and engineering of Horizon by the current listed student team has been significant and equivalent to what would be awarded credit in a senior design course at Oakland University.

[illegible] 05/15/24

Faculty Advisor: Dr. Jun Chen, Ph.D.

Team Member

Sivasakthi Muthukumar (smuthukumar@oakland.edu)
Andrew McGhee (amcghee@oakland.edu)
George Trupiano (gtrupiano@oakland.edu)
Owen Bricker (obricker@oakland.edu)
Luis Gomez (ljgomez@oakland.edu)
Ian Keefer (imkeefer@oakland.edu)
Jacob Hernandez (jdhernandez@oakland.edu)
Harisakthi Muthukumar (hmuthukumar@oakland.edu)
Samuel Walker (swalker4@oakland.edu)
Linda Tir (ltir2@oakland.edu)
Grace Szpytman (gszpytman2@oakland.edu)
Jacob Chiappelli (jdchiappelli@oakland.edu)
Julien Mayberry (jmayberry2@oakland.edu)
Devon Roodbeen (dhroodbeen@oakland.edu)
Valerie Nielson (vnielson@oakland.edu)
Benjamin Braniff (bbraniff@oakland.edu)
Zachary Lain (ztlain@oakland.edu)
Emilio Eslava (emilioeslavabar@oakland.edu)

Table of Contents

Introduction

The Robotics Association at Oakland University is proud to introduce Horizon, our latest intelligent robotic platform for the 2024 Intelligent Ground Vehicle Competition (IGVC). Horizon houses significant design improvements compared to previous competition seasons. The design choices for this new platform build on the team's previous platforms, as well as the implementation of some new innovations.

Team Organization

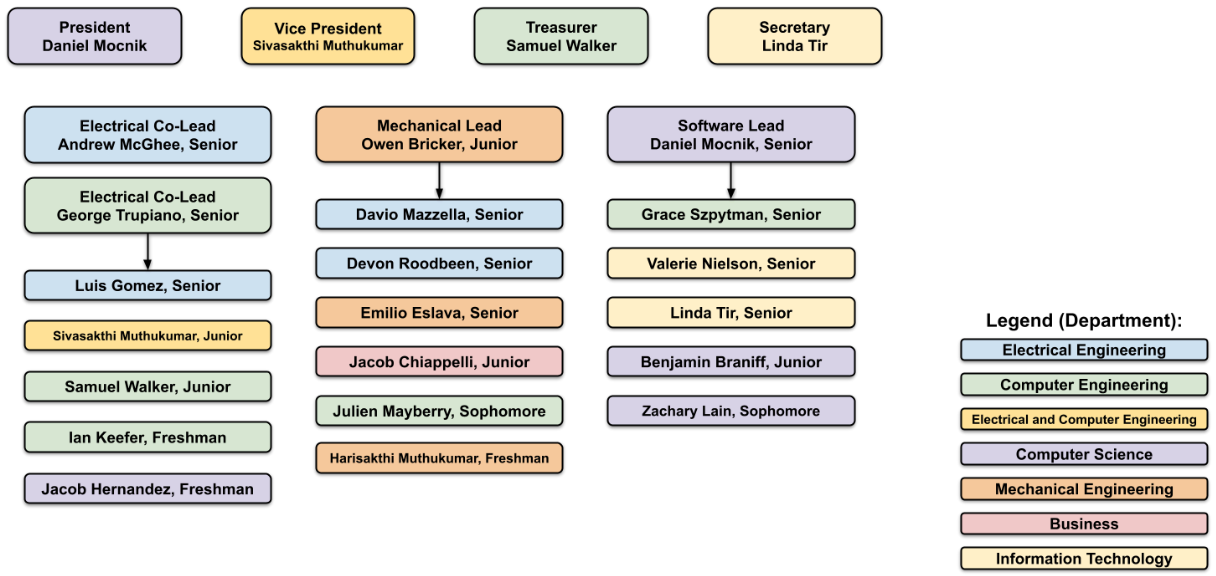


Figure 1: Organization Chart

The Robotics Association at Oakland University is organized to promote collaboration between all areas and members of the team. The team is structured with the executive board, subteam leads, and general team members, and is further comprised of three subteams: electrical, mechanical, and software. The executive board is in charge of administrative work so that members can be provided with a more technical-focused environment. All members are involved with the design and engineering of the platform. Both subteam meetings and regular general body meetings were held to help ensure communication and collaboration between the subteams, and the team as a whole. About 1,200 person-hours were used to facilitate the development of this platform.

Subteam	Subteam Hours	Number of Members	Total Man Hours
Electrical	63	7	441
Mechanical	62	7	434
Software	62	6	372
		Total Team Hours	1,247

Table 1: Breakdown of Person Hours Expended

Design Assumptions and Design Process

When designing Horizon, the requirements and constraints set by IGVC were prioritized, along with insights from previous years' teams. Safety measures were also a key focus, with fuses integrated into the circuitry to protect critical components. The design process was iterative, utilizing older designs and concepts from previous years as models for the new design. Initial ideas were converted into rough drafts for visualization and then reviewed and refined by the entire team. One design was selected and thoroughly analyzed to assess its feasibility and identify potential failure points. Adjustments were made to ensure the design met performance, integrity, and functionality standards. This process was repeated until the final design met all necessary criteria.

Effective Innovations in Vehicle Design

Innovations on Horizon prioritize design simplicity and efficiency. Horizon has moved away from the wheelchair base used by Hindsight, as the base posed problems for documentation and operating voltage requirements. The electrical box has iterated on the tray design of previous years to allow for the electrical components to be accessed on a sliding tray, allowing for efficiency in accessing and modifying them. The electrical system has been modified to remove the usage of buck converters, as all components operate on 12V.

Description of Mechanical Design

Overview

The mechanical sub-team is responsible for creating an enclosure to accommodate Horizon's electrical and software elements to create a dynamic and adaptive design suitable for the IGVC course.

Design on Frame Structure, Housing, and Housing Design

Frame. The frame was built using square aluminum tubes fixed together with custom water-jetted steel brackets. Aluminum tubes were cut using a band saw and the angles were refined using a belt grinder. Aluminum was selected to reduce the overall weight of the system while still having enough structural integrity, while the hollowness of the tubes provided space for electrical routing through the robot. The brackets were designed in Fusion 360 to ensure the configurations would fit in their specified places within the system, and they were cut using a water jet in a machine shop at Oakland University. Steel brackets were individually bent to specific angles to bring the overall chassis of the robot together, necessary for structural integrity and stability in the system. The chassis can be opened and reassembled for future modifications, adding to the modularity of the entire robot.

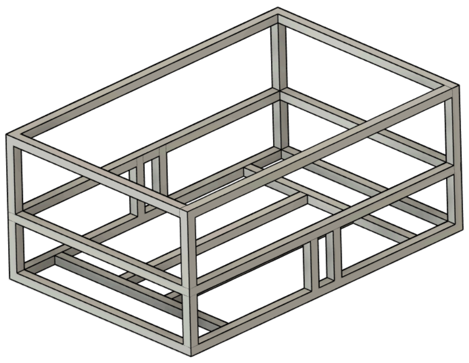


Figure 2: Robot Frame

Tower Structure. The camera and GPS sensors need to be at an optimal height for line perception and GPS signaling. The tower structure was created for this purpose, composed of aluminum columns connected with an aluminum beam across the top. The tower beams are secured at points to the exoskeleton at the base. The hollow beams allow wiring to reside within the structure, protecting them from the elements. The two GPS sensors are located at opposite ends of the joining top beam to avoid interference from the other sensors. The camera is centered along the same beam and angled downward slightly for better line perception.

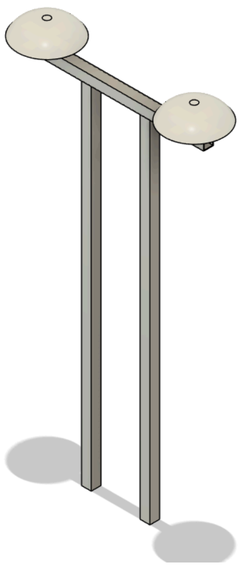


Figure 3: Tower Structure Assembly

Electrical Box. The electrical box was built as a plexiglass drawer system to be lightweight and easily accessible. The box features two layers for protection, sits within the frame, and is held in place by brackets. Having a central location to house the electronics allows for quick removal and adjustments. The wires coming out of the box are organized using cable sheathing.

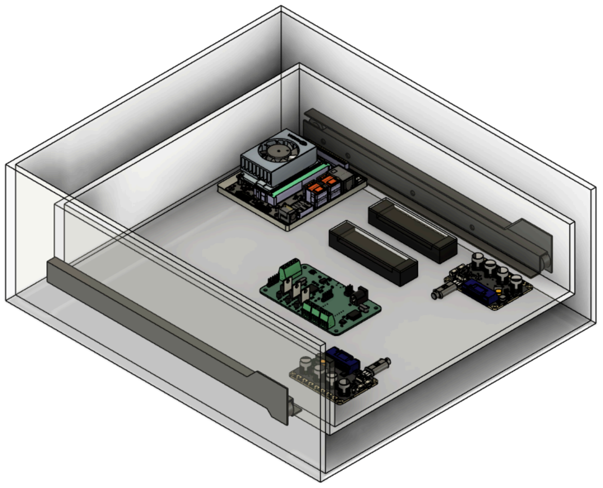


Figure 4: Electrical Box Assembly

Description of Electronic and Power Design

Overview

Each decision behind Horizon's electrical design was made to maximize safety, reliability, and efficiency. Horizon's electrical system is enclosed on a slidable tray for easy access, modification, and protection from rain and debris. The electrical tray is strategically placed in a central location, allowing for shorter cable runs, and helping reduce current loss. Low-level control of Horizon is accomplished with a custom PCB containing the ATmega32U4 microcontroller chip, which directs CAN communication to control system-state lighting and the movement of the two brushless NEO motors via the ODrive S1 motor controllers. External shaft encoders placed on the NEO motors provide more precise movement control. The SparkFun-RTK Global Positioning System (GPS) and ZED 2i camera provide locational information to assist with the autonomous aspect of Horizon. An ESP32 provides Bluetooth capability for the PlayStation 5 controller, which operates Horizon in manual control mode. The PlayStation controller can initiate a software emergency stop via the R2 shoulder button. Safety components, including circuit breakers and fuses, limit the current throughout the system and prevent unwanted spikes that can harm the electrical equipment. Finally, the safety lights indicate what mode Horizon is operating at, denoting the operation mode of the platform- whether it is running autonomously or under user control.

Power Distribution System

The majority of components in the electrical subsystem operate on 12V. The unity in operating voltage means that buck converters can be removed from the design of the power distribution system. Consequently, Horizon operates directly from a single NERMAX 12V 30Ah Lithium LiFePO4 battery. This specific amp-hour rating was chosen due to the power consumption of the electrical components, as the two NEO brushless motors draw an estimated 20A and the rest of the components would draw a cumulative 5A. To ensure a longer runtime, two 30Ah batteries were acquired, minimizing downtime when the in-use battery is drained. With this combination of current draw and supply, Horizon is estimated to operate continuously for approximately 45 minutes, an improvement over previous years' runtime.

Electronics Suite Description

Horizon's electrical box was designed and implemented considering the power distribution. The two NEO brushless motors are each powered and controlled using an ODrive S1 motor controller. The ZED 2i camera, BNO055 IMU, and AMT132S-V incremental rotary encoders are used to track and measure the needed feedback to determine the position and speed of the platform at any given time.

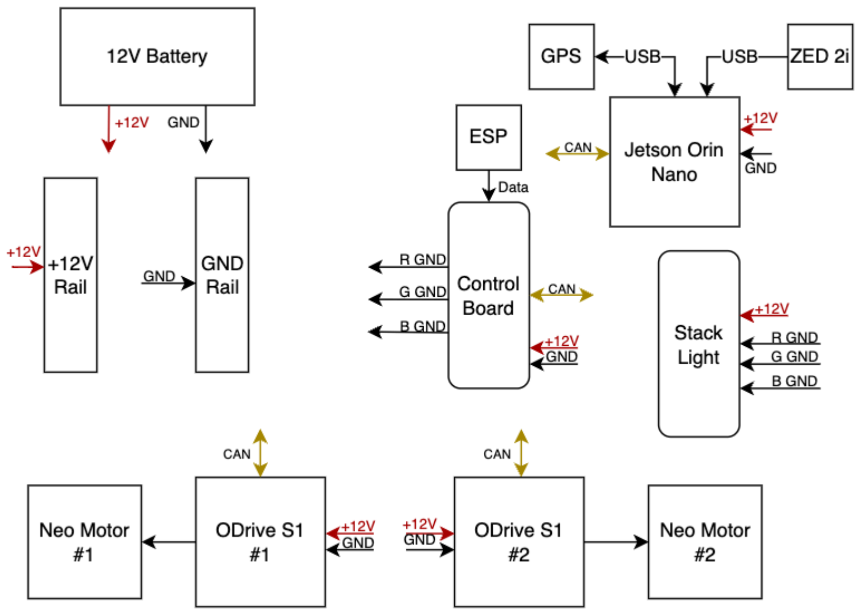


Figure 5: Electrical Box Diagram

The motor driver is controlled using the custom control board PCB. The ATmega32U4 microcontroller chip on the custom board is powerful enough to handle computations for quadrature decoding, IMU reading, RC request handling, and sending and receiving messages across a ROS topic. As previously mentioned, a custom control board was designed for Horizon to combine the functionalities of an off-the-shelf microcontroller, similar to that of an Arduino board, along with the team's designs for Horizon, such as the inclusion of CAN bus communication, a stack light circuit, and an IMU mount. The custom board reduces external connections, reduces size, and improves the overall organization of the electrical box.

The following components were used on Horizon to collect and interpret data from the surroundings for auto-navigation decisions:

Main computer. The Jetson Orin Nano is powered by an NVIDIA Orin system-on-chip design, boasting an ARM Cortex CPU and an integrated GPU using the Ampere architecture. It includes several connectivity options, such as USB 3.0, HDMI, Ethernet, and GPIO pins, allowing it to interface with a wide range of peripherals. The Jetson was chosen for its image processing capabilities, as well as its strong community support, and documentation.

GPS. The SparkFun GPS-RTK-SMA module, which is a breakout board for the uBlox ZED-F9P, was selected for its high precision and reliability. This module is powered by the same 12V batteries that power Horizon. Operating at 20Hz, it offers an impressive accuracy of up to 10mm. Its robust performance ensures that Horizon can navigate with exceptional precision.

Camera. Horizon is equipped with a ZED 2i, a stereo depth camera capable of capturing images at a maximum resolution of 4416 x 1242 (2208 x 1242 per camera) at 15 frames per second. For this application, the camera is configured to operate at a resolution of 2560 x 720 (1280 x 720 per camera) at 30 frames per second. Both power and data for the camera are managed through a single USB connection to the NVIDIA Jetson.

Custom control PCB. The board is based around an ATmega32U4 microcontroller, commonly used on consumer boards such as the Arduino Leonardo. The board can be programmed using Arduino IDE, a familiar IDE to all members of the engineering program. The chosen microcontroller chip has built-in USB communication to exchange data and power the board. Following the design of an Arduino board, the control board has both 8 MHz and 16 MHz crystal oscillators, a voltage regulator, and a normalized voltage circuit. In addition, the board includes a circuit for the indication light used to alert others nearby of the operation mode of the robot. An IMU mount is also incorporated onto the board, with internal connections made to the respective pins on the microcontroller chip. These additional circuits on the control board remove the need for an additional perfboard in the electrical box, which posed an issue in previous years' electrical design.

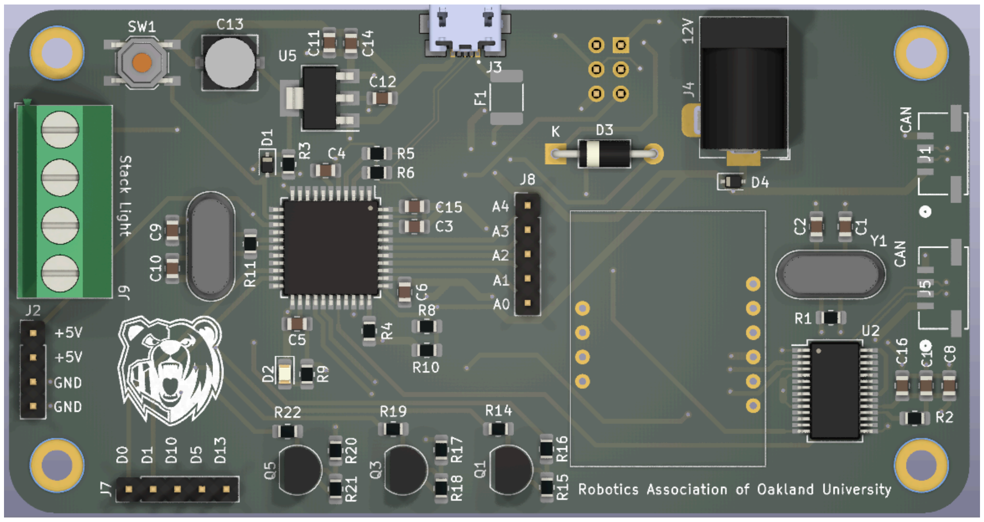


Figure 6: KiCAD Control PCB Design

External encoders. The AMT132S-V incremental rotary encoder is used to get more accurate position and speed values in comparison to the encoder included within the NEO brushless motors.

IMU. The BNO055 IMU is used to measure the acceleration and rotation of the robot to get the robot's positional data at any given time. Although IMUs are conventionally placed in the center of the robot to get more accurate readings of the position and orientation, it is mounted on the control board to reduce wiring, and the offset of the IMU is accounted for in the software.

Safety Devices and Integration

Safety Light. The safety light is a multi-color LED strip wrapped around a vertical pole. Commands to set the state of the safety light originate from the desired state that is controlled by the custom control board. The safety light has modes for indicating when Horizon is operating autonomously (in which the safety light flashes), and manual control (in which the safety light is solid). When Horizon is sent an E-Stop command, the safety light will be set to red.

E-Stop. E-Stop is implemented through two separate systems: one through hardware and another through software. Hardware E-Stop is implemented with a physical emergency stop button, connected to the custom PCB on its 5V line. When the detection pin is pulled high, the PCB immediately sends commands to the motor controllers on the CAN bus to initiate their E-Stop protocols. Because the NEO Brushless motors are three-phase, their motor controllers must work to shut them down as manually cutting power could lead to an inrush or excessive drawing of current upon subsequent power-up. Software E-Stop causes the same chain of events to occur as Hardware E-Stop, with the main difference being that the ESP32 invokes the command after detecting the R2 Shoulder button being pressed on the PlayStation 5 controller.

Description of Software System

Overview

Horizon's software is run on the Jetson Orin Nano, and enables the robot to accurately navigate the IGVC course by processing data from its sensors (ZED 2i, GPS, and IMU) and computing a linear and angular velocity. This is then communicated to the electrical system to move the robot. The system is running on ROS (Robot Operating System) 2 Humble Hawksbill on Ubuntu 22.04. The system also makes use of Nav2, a robot navigation framework that handles mapping, path planning, and localization. Many of these components within Nav2 are configured by modifying simple text-based configuration files and are plugin-based, meaning that a particular plugin can be swapped out with a different plugin (such as one with a different planning algorithm) if desired. This setup allows the robot to be able to navigate environments autonomously, with a minimal amount of manual programming required.

Sensor Processing

Horizon's main sensor for vision and obstacle detection is the ZED 2i. Empowering the ZED 2i is the ZED SDK, which integrates very nicely with ROS and allows the camera's data to be easily subscribed to via different ROS topics, such as a point cloud from the depth data or the raw camera images. By utilizing this point cloud, Horizon can create a costmap of the course via Nav2's mapping capabilities. Objects on the IGVC course (such as barrels) are added to this map so that the path planner can navigate around them. To detect the lane lines of the IGVC course, a basic OpenCV pipeline was written. This pipeline receives the raw color camera image from the ZED 2i and applies a grayscale conversion, region of interest masking, Gaussian blurring, and finally color thresholding. The color threshold is then used as a mask over the raw depth image, creating a “lane line filtered” depth image, which is finally published as a separate point cloud (that contains just the white lines) for the Nav2 costmap to consider.

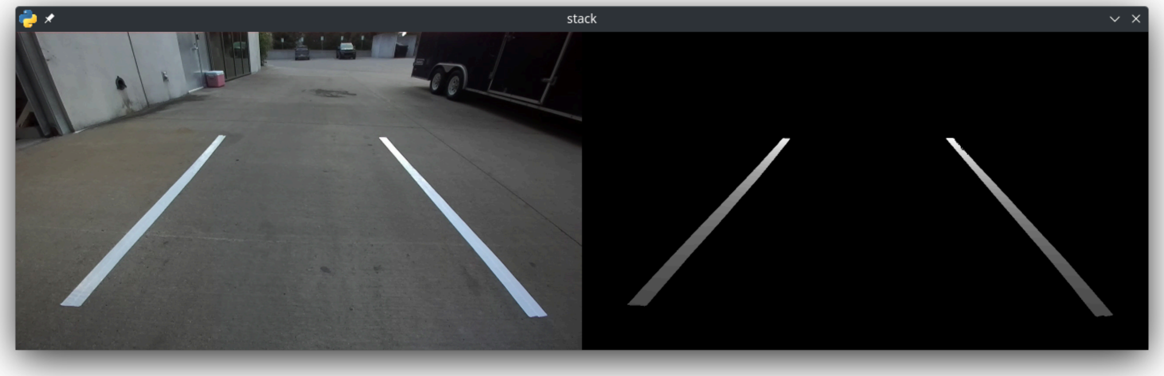


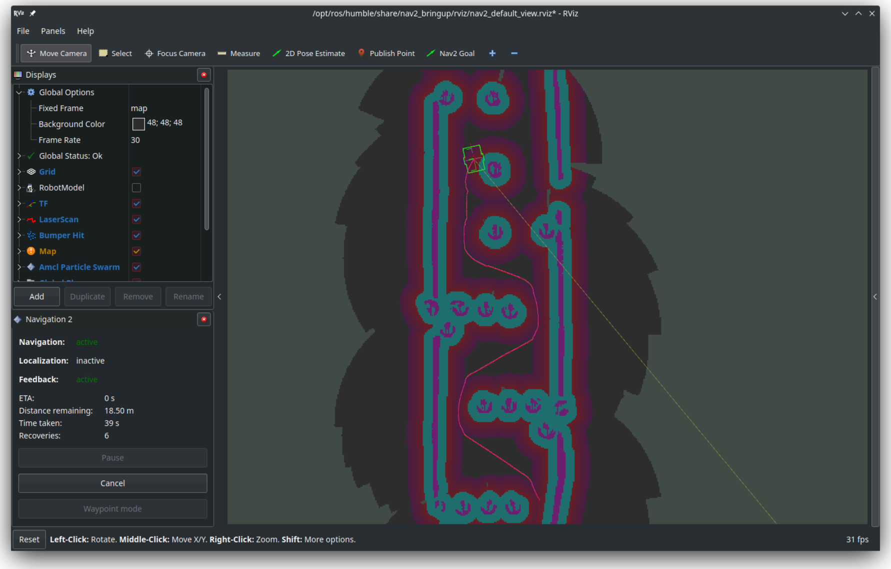
Figure 7: OpenCV Pipeline for Lane Lines

Mapping

Horizon uses the built-in functionality provided by Nav2’s Costmap2D package. Here, the representation of the area around the robot is defined as a costmap, where each map consists of many cells that have a numerical cost, which can represent different occupation states such as “free,” “occupied,” or “unknown.” Each costmap consists of one or more “layers,” where each layer is a way of tracking obstacles or programmatically modifying the costmap’s values. Horizon has both a global costmap (everything the robot has seen, but updates less frequently), and a local costmap (the immediate area around the robot, but updates more frequently). The global costmap consists of an obstacle layer, a voxel layer, and an inflation layer. The obstacle layer subscribes to the point cloud from the ZED 2i and uses 2D ray casting to be able to locate obstacles within the point cloud and mark them as occupied on the costmap accordingly. Similarly, the voxel layer subscribes to the “lane line filtered” point cloud and uses 3D ray casting to convert the point cloud to a group of voxels (cubes), which then marks obstacles on the costmap by projecting a top-down view of the voxels. The voxel layer is also configured to only include points within a specific range of the ground plane to ensure that only lines are included. Finally, the inflation layer simply takes all the obstacles that have been marked on the costmap, and adds an area of slightly lesser cost around the obstacles. This is to help discourage the path planners from creating paths that might cause the robot to be dangerously close to an obstacle. On the other hand, the local costmap has a voxel layer of both the previously mentioned point clouds and an inflation layer.

Path Planning

Horizon makes use of two techniques to be able to plan through and traverse its environment. The robot makes use of “behavior trees,” which are a conditional list of tasks for the robot to attempt to accomplish. In the context of the robot, the robot’s overarching task is to navigate to each of its goals (e.g. the GPS waypoints), and back to its original starting point. To navigate to a goal, a path first needs to be computed to the selected goal. If no valid path can be computed, the behavior tree can trigger different actions to help the robot be able to compute a new path, such as backing up, spinning, or clearing the costmap to populate it with new data. The behavior tree can also dynamically switch between different path planners based on certain conditions being met, such as the goal being changed and/or reached, or an amount of time passing. To plan a path to the robot's next goal, one of Nav2's built-in algorithms is used. This algorithm is the NavFn Planner, which, when given the global costmap, plans a path to the goal using Dijkstra's algorithm. Finally, Nav2's controller is used to follow the path and take action to dynamically navigate around any obstacles that may appear in the robot's local costmap.



Global Options
Fixed Frame map
Background Color 48; 48; 48
Frame Rate 30
Global Status: Ok
Grid ✓
RobotModel ☐
TF ✓
LaserScan ✓
Bumper Hit ✓
Map ✓
Amcl Particle Swarm ✓

Navigation 2
Navigation: active
Localization: inactive
Feedback: active
ETA: 0 s
Distance remaining: 18.50 m
Time taken: 39 s
Recoveries: 6

Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click: Zoom. Shift: More options.

Figure 8: Robot Costmap and Path Planning

Localization

For localization, Horizon uses a dual-EKF (Extended Kalman Filter) setup. Each EKF takes data from the sensors and fuses them to get a better estimate of where the robot is within each of its maps. The first is for localizing the robot globally, and fuses data from the GPS, IMU, VIO (Visual-Inertial Odometry) from the ZED 2i, and the wheel encoders. The second EKF is for computing the robot's odometry locally, or how it has moved from its starting position. This EKF fuses the same data sources as the global EKF, minus the GPS. Finally, since Nav2 treats all location data in cartesian coordinates, a node runs alongside the EKFs that converts the raw GPS coordinates to cartesian, to allow the data to be fused.

Cyber Security Analysis

The NIST RMF (Risk Management Framework) is a tool for controlling and taking account of risks that pertain to information security. It describes the steps necessary to employ proper security measures to reduce the potential impact of threats. The 7 steps are as follows: prepare, categorize, select, implement, assess, authorize, and monitor. If a malicious user were to access the robot, the extent of their damage should be minimal at best. Due to the simplistic nature of the robot's hardware, the only software portion that could be tampered with is the Jetson Orin Nano. To prepare for an event of this nature, all members of the software team are familiar with the Jetson, the operating system running on it, and its capabilities. This event could be categorized as having low confidentiality, and high integrity and availability since there is not much sensitive information residing on the Orin Nano, but having the integrity of the software on it as well as its availability being impacted could present problems to the operation of the robot. To mitigate this, the SP 800-53 Control Enhancement that could be implemented is IA-05(01), which pertains to passwords. The Robotics Association already maintains a list of passwords for all accounts owned by the club, as well as the passwords for the various computing devices in our possession, as well. Each password is unique and complex to ensure that they are hard to guess and secure against unauthorized access. We continually make sure that all assets are secured in this manner and if we encounter that they are not, take the appropriate steps to rectify the situation.

Complete Vehicle Analysis

Lessons Learned

One of the major challenges in designing Horizon was accurately estimating the time required for various tasks. The team realized that starting early is crucial for success as it allows ample time for thorough planning, fabrication, and testing. By beginning the design and development process early, potential issues can be identified and addressed before they become bigger issues. Starting early also provides more opportunities for team members to learn and refine important skills. Creating an effective schedule and sticking to it—while also being flexible to changes, as they come up—is an important skill that many have picked up this year.

Potential Hardware Failures

Horizon's simplicity means that the quantity of potential hardware failure points has been reduced compared to the previous year's design. However, care must still be taken with the remaining potential failure points.

Cable management is a priority for Horizon's design, especially since the electrical components are housed on a moveable tray. Zip-ties, cable sleeving, and velcro secure hanging wires to prevent issues. Loose wires can be dangerous and can disconnect, causing damage by getting caught in moving parts, especially when opening the electrical box tray.

Mechanical failures such as weatherproofing, joint fatigue, and sensor instability can affect the robot's efficiency when outdoors. Joint weakening can occur from carrying loads or driving on uneven terrain, so extra brackets were added to support major joints. Sensor instability was addressed by adding anchor points to the tower to ensure reliable data collection. Weatherproofing is essential to prevent damage to electrical components, so caulk was applied around the plexiglass sheets to maximize Horizon's resistance to water and dust.

Horizon is equipped with several safety precautions to handle potential failures. It has two emergency stops: one physical e-stop button on the front and top of the robot's frame, and a switch on the remote controller. Activating either of these stops cuts power to the motors. Additionally, if the remote controller disconnects or the motor controller fails, the motors will not receive power from the batteries. A circuit breaker and fuses are also in place as safety measures. If a motor stalls and draws more current than expected, the in-line fuses will blow to protect the electrical system or the circuit breaker will trip and cut off power to the entire robot.

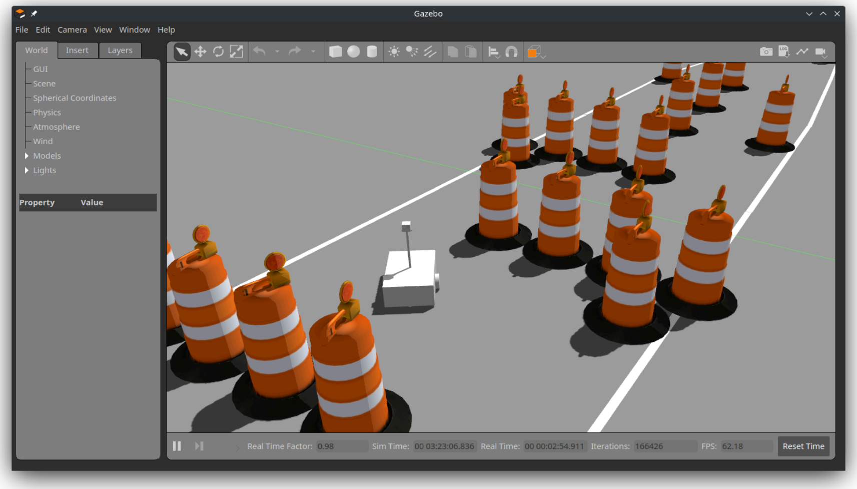
Two 2" caster wheels were used in addition to the bigger wheels on Horizon. These wheels can be somewhat flimsy, with poor stability and traction. They can sometimes struggle with tight turns and rough terrain, often leading to reduced maneuverability and increased risk of getting stuck. Also, their small size can result in frequent maintenance and potential damage to the robot.

Software Testing, Bug Tracking, and Version Control

For all software testing, debugging, and version control the team utilized GitHub. This online system facilitated easy sharing of code allowing for the team to stay on top of any updates made. Testing was individually done as the team members saw fit. Some members frequently pushed smaller updates and changes, checking in with the rest of the team to see how these changes were impacting the functionality. Other members would push commits infrequently, the scale of which was quite large with full functionality. Bug tracking was noted when any member would make commits to GitHub. The summary section was frequently edited to contain information about the most recent attempts made, errors returned, and possible solutions for consideration. GitHub was also used for version control. This allowed multiple different members of the team to work on similar parts of the robot at the same time without concern for overwriting others' work, sharing progress made in real-time whilst providing a place to look back on previous work, from both this season and previous seasons. The software lead was the main point of contact when merging branches of the repository were needed.

Simulation

For testing purposes, the team used Gazebo simulation software. A recreation of the previous year's IGVC course was built in Gazebo including lane lines, barrels, and a ramp. Gazebo supports the use of spherical coordinates, functioning similarly to GPS waypoints which were used to replicate the waypoint requirement. A variation on the competition robot was then inserted into the world to test vision, mapping, and path-planning capabilities.



Real Time Factor: 0.98 Sim Time: 00 03:23:06.836 Real Time: 00 00:02:54.911 Iterations: 166426 FPS: 62.18

Figure 9: Robot Navigating Simulated IGVC Course

Performance Testing to Date

Performance testing has primarily been completed on the subteam level to ensure that each subteam's system is operating at an acceptable level before beginning integration with the other two subsystems. The subteams can perform tasks pertaining to what is required during an IGVC course run with relative ease. For the software subsystem, in simulation, Horizon is able to perceive and map obstacles, plan a path to a selected goal, and output the desired wheel velocities to physically navigate the robot to the goal. It is anticipated that the system should work well with the electrical system with some extra parameter tuning, adjustments, and integration. For the electrical subsystem, the E-stop circuit, motor controllers, control PCB, and safety components have been individually tested to verify functionality. They are currently all functioning well together and are expected to perform correctly when integrated with other systems.

Initial Performance Assessments

Each subsystem appears to be performing as intended, based on the extensive testing completed by each subteam. There is, however, room for improvement in each subsystem, and new adjustments made to each system are currently being thoroughly tested. It is anticipated that these adjustments will improve the overall performance of the robot as one complete system.

Cost Report

Part	Model	Date Ordered	Quantity	Price / Unit	Cost Total	Cost to Team
Motors	NEO Brushless V1.1	10/2023	2	$65	$130	$130
Motor Controllers	ODrive S1	11/2023	2	$168	$336	$336
Microcontroller	ESP32 WROOM-32	2/2024	1	$5	$5	$5
Controller	Sony DualSense Wireless Controller	—	1	$70	$70	$0
Computer	NVIDIA Jetson Orin Nano	11/2023	1	$500	$500	$400
Control Board	PCB Shield	3/2024	1	$15	$15	$15
Encoders	Automation Direct AMT132S-V	3/2024	2	$35	$70	$70
GPS	SparkFun GPS-RTK-SMA	5/2024	1	$350	$350	$350
Camera	ZED 2i	—	1	$550	$550	$0
Batteries	Nermak LIFEPO4 12V 30Ah	2/2024	2	$187	$374	$374
Gearboxes	AndyMark EVO Slim	12/2023	2	$150	$300	$300
Mechanical	Raw Materials*	—	—	—	$601	$601
Electrical	Assorted Materials**	—	—	—	$100	$100
				Total Cost	**$3,501**	**$2781**

* Includes aluminum beams, steel sheets, polycarbonate sheets, clamps, and wheels.
** Includes wires, crimps, solder, flux, resistors, capacitors, and other electrical components.--

Table 2: Cost Breakdown of Horizon to Date