**University of Michigan Dearborn**
**KiloOhm**


**Team Members**:
Sal Martinez: salatiel@umich.edu
Andrew Johnson: johnsoaa@umich.edu
Avery Girven: agirven@umich.edu
Evan Miller: evanemil@umich.edu
Eejoy Lim: eejoylim@umich.edu
Emma Hetrick: hetricke@umich.edu
Laura Sas: laurasas@umich.edu
Andrew Ealovega: avealov@umich.edu
Jordan Krasan: jkrasan@umich.edu
Zachary Fusinski: zfusinsk@umich.edu
Jadyn Smith: jadyns@umich.edu
Joseph Berna: joeberna@umich.edu
**Team captain:** Joseph Kennedy: josephpk@umich.edu
**Faculty Advisors**: Michael Putty Ph.D
DATE: 5/15/2022

I, Michael Putty Ph.D of the Department of Electrical and Computer Engineering at the University of Michigan Dearborn, certify that the design and development of KiloOhm is significant and unique, and is equivalent to what might be awarded credit in a senior design course.

**X_____Michael Putty_____**

## ABSTRACT

This paper presents KiloOhm, a robot designed and used by the University of Michigan - Dearborn for the 29th Annual Intelligent Ground Vehicle Competition (IGVC). The objective of this robot is to autonomously navigate through an outdoor obstacle course, navigating to GPS waypoints while avoiding barrels and staying within white lines. Data from a lidar, an IMU and GPS unit and cameras are processed, a map is generated, and the robot attempts to locate itself within that map. The robot can then determine where it wants to go and at what speeds it must spin the motors to get there.

While the general framework and function of the robot has remained the same over the years, the University of Michigan-Dearborn has continued to advance and upgrade this robotics platform with new hardware and software. Recently, the entire frame has been upgraded to a metal structure that is both strong and lightweight. This year the mechanical and electrical teams were busy determining how to switch over the old electronic systems to the new platform while reducing wiring mess and electric noise. The electrical team has also added features to improve the time needed to test the robot. The software team has replaced old software that did not function properly while also adding new software to analyze new sensors and react accordingly.

## INTRODUCTION

The Intelligent Systems Club of the University of Michigan - Dearborn enters the 2022 Intelligent Ground Vehicle Competition with 11 new and 2 returning members. With such a large influx of new members relative to those who were a part of the team before, a key goal of this year was to educate and train new members on the system as improvements were designed and tested. For example, the electrical circuit was broken down into several sections, and the new electrical team members were taught about how each section worked as new improvements were being added to that part of the system. The software team gathered resources to help the new members learn the basics of ROS coding and broke down large tasks into smaller, more manageable problems for the new members. The new members also benefited greatly from the hands-on experience they were able to obtain. The electrical members were able to directly observe the electric circuit and the wiring was transferred over to the new platform. The software team was able to observe the robot operating in simulation as the robot was created and tested in Gazebo. This approach of teaching new members the basic functionality of the robot while integrating in new features allowed the team to both train members and complete projects in the short period of time before competition.

The team consists of 13 undergraduate students studying different fields. The team member composition is displayed in **Table 1**.

**Table 1: Team KiloOhm Composition**

| Name | Email | Class | Role |
|---|---|---|---|
| Joe Kennedy | josephpk@umich.edu | Robotic Engineering, Senior | Captain |
| Andrew Johnson | johnsoaa@umich.edu | Electrical Engineering, Junior | Mechanical |
| Jadyn Smith | jadyns@umich.edu | Human-Centered Design and Engineering, Sophomore | Mechanical |

| | | | |
|---|---|---|---|
| Evan Miller | evanemil@umich.edu | Robotic Engineering, Freshman | Electrical |
| Laura Sas | laurasas@umich.edu | Electrical Engineering, Freshman | Electrical |
| Zachary Fusinski | zfusinsk@umich.edu | Electrical Engineering, Junior | Electrical |
| Joseph Berna | joebre@umich.edu | Electrical Engineering, Senior | Electrical |
| Emma Hetrick | hetricke@umich.edu | Computer Science, Junior | Software |
| Avery Girven | agirven@umich.edu | Computer Science, Senior | Software |
| Andrew Ealovega | avealov@umich.edu | Computer Science, Sophomore | Software |
| Jordan Krasan | jkrasan@umich.edu | Computer Science, Junior | Software |
| Eejoy Lim | eejoylim@umich.edu | Computer Science, Freshman | Software |
| Sal Martinez | salatiel@umich.edu | Computer Science, Freshman | Software |

## ORGANIZATION

This paper will begin with a description of design innovations, then cover mechanical electrical and software systems in finer detail. After those sections there will be a detailed description of the software strategy, an overview of failure modes, and the performance analysis to conclude the report.

## DESIGN ASSUMPTION AND DESIGN PROCESS

This year the team utilized an iterative design approach prioritizing core functionality across all subsystems first, through a 3-step design, test, improvement process. Each iteration added new features, and moves from functioning design to functioning design, a structure that was facilitated by the use of an existing mechanical and electrical platform from previous competition. For example, in the first iteration, the features being implemented only encompassed basic lane detection and obstacle avoidance, while leaving more advanced features like mapping or high-level path planning, for later iterations, work done in simulation and physically to allow for electrical and mechanical redesign in tandem with software work.

## DESIGN INNOVATIONS

This year, the team has several goals spread out across all three subsystems. **Table 2** describes these areas. **Table 3a-c** describes the costs of the robot.

**Table 2: Design Innovations and Reasoning**

| Areas to be Improved or Added | Reason for Improvement or Addition | Improvement Result |
|---|---|---|
| Designed new electrical enclosure | Previous enclosure would not fit on new frame, increased water protection was desired | Smaller, more serviceable electric enclosure |
| Improved wiring layout | With the new frame, the electrical circuit had to be placed in a new enclosure. | High and low voltage components separated, related |

| | | |
|---|---|---|
| | | components placed together |
| Added Hot Swapping function | Powering the robot down and back up takes time, so it would be easier if depleted batteries could be removed without causing the system to power down | Diodes in parallel were placed to wire connections to two batteries, ensuring the safety of each during hot swapping |
| Improved Power Distribution System | Voltage spikes caused by starting or stopping the motors caused spikes on the control line. It would be best to isolate the two systems | Completely separated the motor and controls battery. Each is now powered by a separate battery |
| Ignition Gazebo simulation | Software team did not have access to the physical robot while electrical and mechanical were rewiring the system | Robot and sensors can be tested in virtual environment composed of objects and white lanes |
| Reduced odometry drift and error | High odometry error or drift can cause the robot to believe it is somewhere it isn't | Extended Kalman filters which take in multiple sensor inputs and produce a filtered odometry stream |
| White line detection software rewritten | Previous algorithm was not accurate and was susceptible to errors in different lighting conditions | Software now has increased accuracy and can operate in different lighting conditions |
| Calibrated SLAM and navigation | Robot needs SLAM in order to both map itself and locate itself without relying purely on GPS | SLAM fully calibrated |

**Table 3a: Mechanical Cost**

| Mechanical | Qty | Unit cost | Price |
|---|---|---|---|
| Frame | 1 | $300 | $300 |
| Motors | 2 | $450 | $900 |
| Plastic (+) | 5 | $30 | $150 |
| **Total Mechanical Cost** | | | **$1,350** |

**Table 3b: Electrical Cost**

| Electrical | Qty | Unit cost | Price |
|---|---|---|---|
| 2D Sick LiDAR | 1 | $5,000 | $5,000 |
| GPS | 1 | $5,000 | $5,000 |
| Computer | 1 | $650 | $650 |
| Camera | 1 | $650 | $650 |
| IntelRealsense Camera | 1 | $400 | $400 |
| 6s LiPo battery | 6 | $80 | $480 |
| Battery Management | 1 | $100 | $100 |
| Motor controller | 1 | $390 | $390 |
| Misc | 1 | $100 | $100 |
| **Total Electrical Cost** | | | **$12,320** |

**Table 3c: Vehicle Cost**

| Overall Category | Price |
|---|---|
| Electrical | $12,320 |
| Mechanical | $1,350 |
| **Estimated Retail Price** | **$13,670** |
| **Actual Cost** | **$1,200** |

## MECHANICAL DESIGN

The vehicle is made primarily of aluminum and uses a differential drive steering control scheme which is aided by two casters. The CAD model of the robot is shown in **Figure 1**. **Table 4** provides the dimensions of the robot.

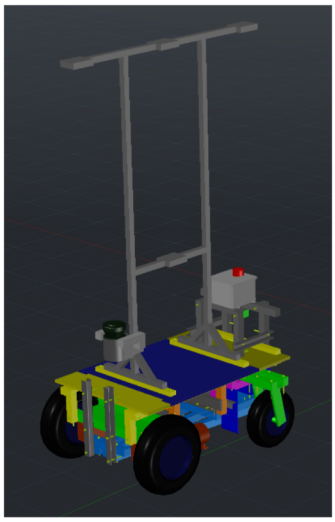

**Figure 1:** Robot design

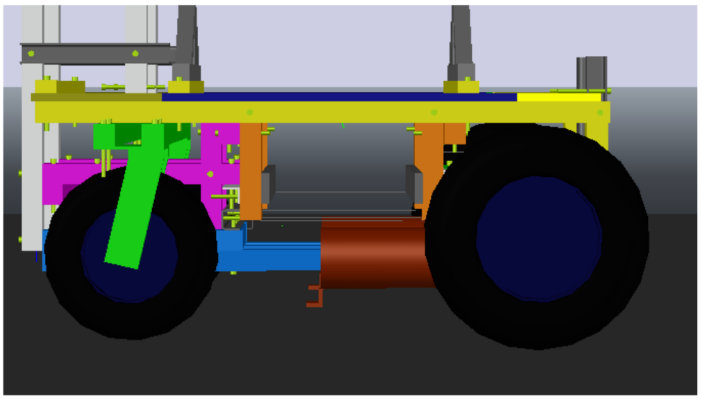

**Figure 2:** Electronics location

### Structure design

The robot is made almost entirely of aluminum with plastic enclosing the frame and electronics for environmental protection as well as polymer platforms for the payload, router, and battery casings. The robot has been manufactured from 1 ¼" x 1 ¼" aluminum square tubing, bolted together. The center of the frame houses the main electronics shown in **Figure 2.**

The robot utilizes two front casters and is propelled by twin 24 volt NPC B81 brushed DC motors with integrated 18:1 gearboxes, providing a maximum of .81 horsepower each and a maximum of 180 rpm. The tires are 0.33m and have been replaced recently to improve traction. There is an aluminum mast that serves as a mount point for the GPS, IMU, both cameras, and LiDAR. The batteries are housed in the front of the robot for accessibility.

**Table 4: Vehicle dimensions**

| | Vehicle | Requirements |
|---|---|---|
| Width | 0.78m | 0.61m - 1.21m |
| Length | 0.94m | 1.21m - 2.13m |
| Height | 1.79m | 1.82m maximum |

| | | |
|---|---|---|
| Mast height | 1.37m | - |
| Mast length | 1.02m | - |

**Mechanical Updates for this Season**

The mechanical team made several design changes which improved the weatherproofing and functionality of the robot. A new electrical enclosure, shown in **figure 3,** was designed to separate the high and low voltage, improve weatherproofing while also reducing heat buildup, and increase accessibility. A fan was integrated into the electrical enclosure to cool the computer and electronics, but the opening was protected by a sloping overhang so rainwater could not enter the box. All the electrical wires are fed out of the box underneath this overhang. To service the electronics, one would simply need to unplug the connections, remove the electrical enclosure, and then remove the lid to have access to the middle layer or remove the bottom panel to have access to the high voltage components. This design allows the electrical team to easily service or test any component in the electrical system.

**Figure 3:** New Electrical Enclosure

To further improve weatherproofing of some components, a specific mount was 3D printed to house and protect the Intel RealSense camera. To increase functionality, the orientation of the vehicle was reversed so that the drive motors were located in the rear and the casters in the front. The team determined that this orientation performed the best on pavement. To increase traction, the payload mounting point was placed directly over the drive wheels.

## ELECTRICAL COMPONENTS AND DESIGN

**Overview**

The electrical system is composed of a power distribution system, a safety system, and a sensor suite. Most of the components are housed in the electrical enclosure. This year, the team improved the layout and wiring of the components in the electrical enclosure, added the hot swapping functionality, and separated the power sources for the motors and control circuit.

**Power Distribution System**

There are three different voltage levels used by KiloOhm. Two 6S LiPo batteries provide the power to the system. The 22.2V supplied is then converted into 12V for the Safety system and 19V for the onboard computer. **Figure 4** shows the component voltage breakdown.

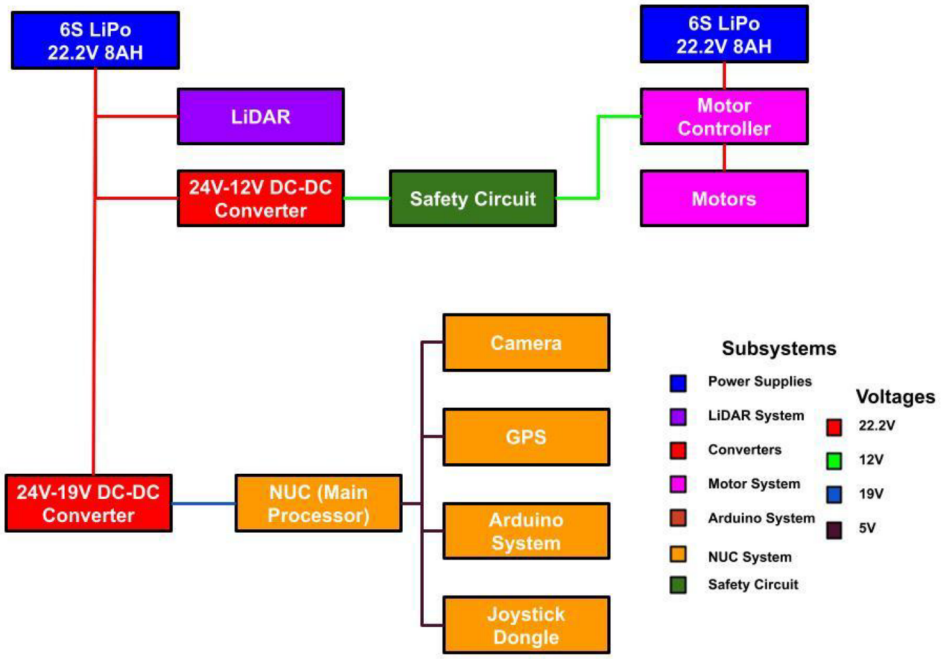


**Figure 4:** Component Voltage Breakdown

**Batteries**

This year, to protect our computer and LiDAR, our robot is powered by two LiPo batteries (specification provided in **Table 5**). The batteries are located inside a separate box on the front of the robot, which allows for easy access and swapping when necessary. Because of the new hot swapping circuit, the batteries can be replaced without powering down the computer and forcing a reboot of the software. An Arduino Uno is used to monitor the voltage levels of the control battery, and this is interfaced with the LED tower to indicate to the user when batteries are low.

**Table 5: Battery specifications**

| Capacity | Nominal voltage | Overcharge | Charge time |
|---|---|---|---|
| 8AH/12AH | 22.2v | 25.2v | ~3 hrs |

**Power Budget**

**Table 6** shows the power draw of major components in the electrical system both during average consumption and max consumption.

**Table 6: Power Budget**

1

| Component | Average Consumption | Max Consumption | Operating Voltage | Source |
|---|---|---|---|---|
| Motors (both) | 288W | 1680W | 24V-36V | 24V Motor Battery |
| LiDAR | 8W | 35W | 10V-30V | 24V Control Battery |
| Safety Circuit + Diagnostics | 5W | 10W | 12V | 24 - 12V converter |
| NUC | 20W | 70W | 19V | 24-19V converter |

| System | Average | Max |
|---|---|---|
| 24V Motor Battery | 327W | 1780W |
| 24V Control Battery | 327W | 1780W |

**Safety System**

To ensure the safety of our robot, a wireless relay control system was installed to work in tandem with a physical emergency stop button to control a solenoid. The wireless estop has been tested reliably out to 100ft. The wireless relay and physical E-Stop directly controls the solenoid which controls power to the motors. This means that the emergency and wireless E-Stops are purely achieved through circuitry and not software.

The wireless relay remote has two buttons that are configured into two modes: pause and kill shown in **Figure 5**. “Pause Mode” toggles the robot between an active and disabled state. This mode is intended for testing or when setting up the robot. “Kill Mode” will immediately cut power to the motors by activating the solenoid. In order to re-enable the robot after it has been “killed”, the physical emergency stop button must be cycled. This is intended to force the operator to walk up to the robot and make sure the robot is safe to operate again.

The robot also has a physical E-Stop. When the E-Stopped is pressed the power to the motors will be cut. To make the robot run again the E-Stop must be released.

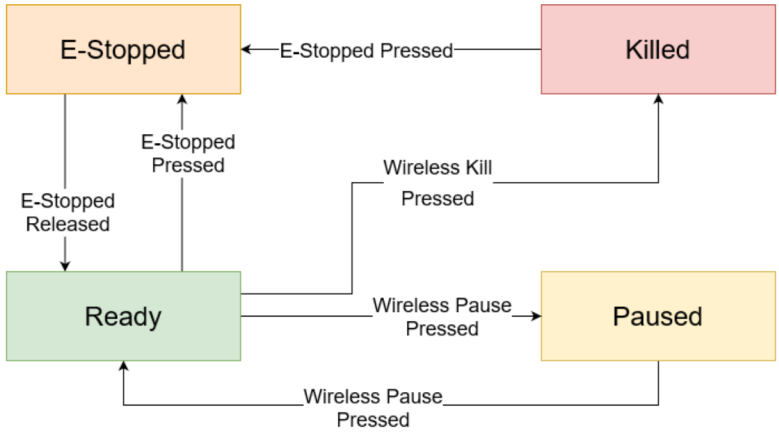

**Figure 5:** Safety System State Diagram

**Indicator System**

Ohm has indicators for many different aspects of the robot. It uses a bright LED light to indicate when the robot is on and what mode it is in. When the robot is in manual mode the light will be solid and when the robot is in autonomous mode the light will be blinking on and off. Ohm also uses the tower light to indicate the voltage of the battery. When the battery voltage is running low the red tower light will begin to flash.

**Sensor Suite**

Ohm uses four main input sensors (descriptions provided below) to help it navigate the course. These sensors are all processed on a laptop computer which then sends the appropriate commands to the motor controller as shown in **Figure 7**.

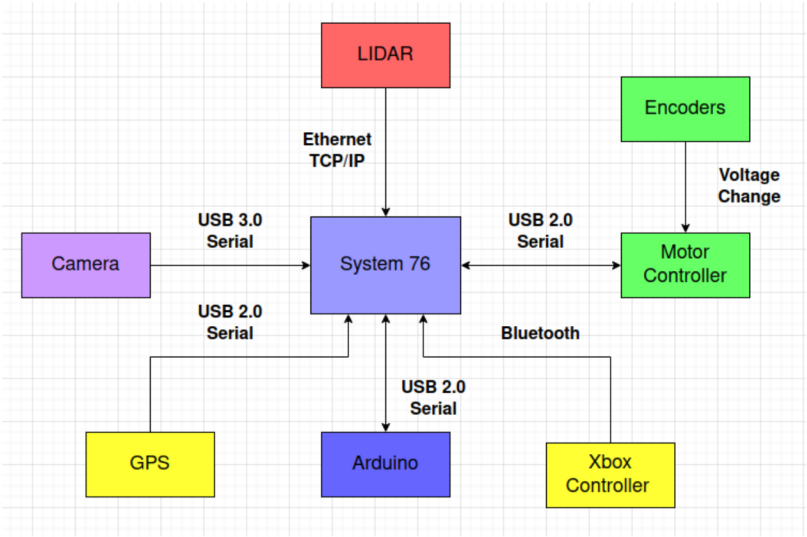


**Figure 6:** Hardware Interfaces

A SICK LMS-111 Lidar is used due to its high reliability and accuracy and has been implemented in various weather conditions. The scan range is 20m at a frequency of 25 Hz with an angular resolution of 0.25°, and has 270° field of view.

KiloOhm uses a combination of both the Intel Realsense D435 Depth Camera for detecting white lines and an Intel T265 Stereo Camera with IMU Sensor Fusion Intel Movidius Myriad VPU accelerator which calculates odometry. The main interface for the RealSense cameras is USB 3.1, which connects and communicates with the main robot controller. The main robot controller is responsible for delivering appropriate power to the sensor.

KiloOhm uses the VectorNav VN-300 dual antenna GPS. This system is used due to the increased positional and heading accuracy compared to a Garmin Marine GPS 19x HVS system the team has used in the past. It also has a built-in 10-axis IMU which it uses in conjunction with a built-in Kalman filter to prevent large jumps in heading and position. The IMU is also used for navigation.

To get accurate odometry, KiloOhm uses two Kubler Turck 200 PPR (pulses per revolution) quadrature encoders, one mounted on each motor. These encoders are connected to the Roboteq motor controller, which handles pulse counting. These values are polled by the software system to calculate the robot's position.

KiloOhm uses a System76 laptop to perform all computations and is the main interface for all sensors. It uses an I7-6770HQ quad core CPU with 16GB of RAM, running on Ubuntu 20.04 and ROS2 Galactic.

**Electrical Updates for this Season**

Because the electronics had to be transferred over to a new platform, the team redesigned the layout of the electronic components to improve serviceability, reliability, and organization. As mentioned before, the electronic components were divided into a high voltage and low voltage section, and each section was designed to easily be removed from the electronics enclosure and serviced. New wire terminals and Anderson connections were placed on most components, adding to the reliability of the system. Additionally, a fan was integrated into the electrical enclosure. The Arduino can measure the temperature of the electrical box and turn on the van when necessary.

The team implemented hot swapping in order to quickly change out the batteries during competition. To achieve this two different battery connections were added in parallel, and a diode was placed in series to each connection, as seen in **Figure 7**. When the current battery is drained and operating at a low voltage, a new battery can be connected to the parallel connection. The diode for the new battery will be forward biased, because the voltage of the new battery will be higher than the voltage of the old battery. This also means that the diode in series with the old battery will be reversed biased, and no current can flow to or from the old battery. This method ensures that the new battery will not attempt to charge the old battery and damage the entire system.

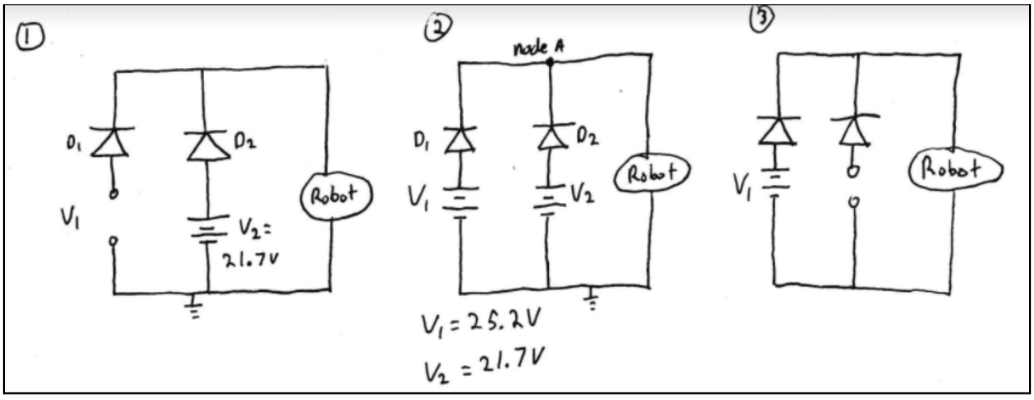


**Figure 7:** Hot swapping circuit

During testing, an electrical surge occurred on the main power bus, damaging some components. It was uncertain if this electrical surge occurred due to the LiPo battery discharging lower than its recommended voltage or if starting the motors created this spike. At this time the electrical system was powered by a single 6s LiPo battery, so the motors, Lidar, and control circuit shared the same 24V power. Later, the system was tested without the electrically sensitive components connected to the 24V power bus. An oscilloscope was connected, and the motors were started and stopped so that any potential noise would be detected. The

oscilloscope did detect electrical noise on the line when the motors were spinning, as seen in **Figure 8**.

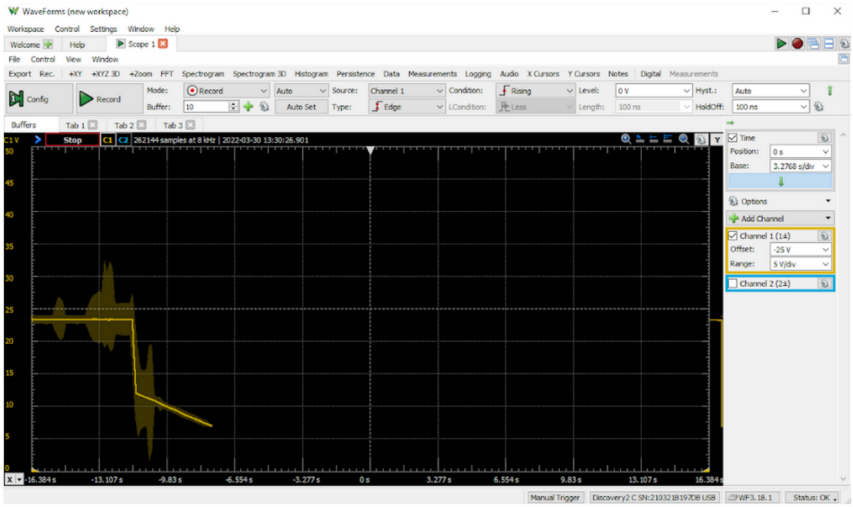

**Figure 8**: Electrical Noise Measured from the Motors

To prevent this electrical noise generated from the motors from interfering with the sensitive electrical components, the electrical team decided to revise the power distribution system. The motors would be powered by a separate 24V LiPo battery, so the electrical noise would remain on a separate 24V line. The wiring was also adjusted so that the noise motor wires would not be located next to the control wires. After these changes the system was measured again with the oscilloscope, and minimal electrical noise was measured on the control circuit.

## SOFTWARE

### Overview

KiloOhm's software has received a complete rewrite for the 2022 season. This was primarily performed to better utilize modular libraries the team developed for other projects while unable to compete in IGVC due to the COVID-19 pandemic. During this rewrite, we took the opportunity to begin using ROS2, which allows us to both utilize more modern libraries and avoid the approaching ROS1 EOL. Overall, the software stack resembles that of past years, but with much needed modernisation and a significant bump in code quality. Of note is the full implementation of mapping, which has been a long running goal for the Ohm series that has at last been completed in KiloOhm.

### Odometry and Extended Kalman Filter

To aid in localisation, KiloOhm uses four sensors. These are two Kubler Turck wheel encoders that are placed on each motor, a Intel Realsense T265 Tracking Camera, and a 10-axis IMU onboard the Vectornav VN-300 Dual-Antenna GPS Unit. The T265 Tracking Camera uses a combination of IMU and stereo cameras in order to perform onboard VIO (Visual Inertial Odometry). This odometry stream is then fused with both a second odometry stream

generated from the wheel encoder data as well as acceleration/orientation data given by the Vectornav's onboard IMU using an Extended Kalman filter. The filter takes in all the odometry sources provided and creates a single filtered odometry stream which allows us to minimize error from each of the sensors by fusing different pieces of data from each sensor that will cancel out error from other sensors. By utilizing this approach we create a superior odometry stream which will allow for greater precision in movement and obstacle avoidance.

**Slam**

KiloOhm performs simultaneous localisation and mapping (SLAM) using Slam-Toolbox, a modern SLAM library. SLAM takes scans from the LiDAR and white line detection to create a 2D occupancy grid (map) of its surroundings. By also taking odometry data from the EKF, Slam-Toolbox can effectively localize KiloOhm within the map. This map is continuously refined over time using loop closure. By producing a map in such a dynamic manner, KiloOhm can navigate in both dynamic and static environments, as well as save data between heats, if desired.

**Navigation and Obstacle Avoidance**

Navigation is handled by the successor of navstack called Navigation2. Nav2 works in almost the same way as its predecessor. Nav2 is broken up into a Global section and a Local section. The Global section encapsulates the Global costmap as well as the Global planner, and the Local section encapsulates the Local Costmap along with the Local Planner.

In order to navigate in its surroundings, Nav2 creates 2 Costmaps, a global map as well as a local map based off of the 2D occupancy grid received from Slam, the odom topic, and the laserscan topic. Both Costmaps are configured with different layers that stack on top of each other. Those layers include a static layer that creates cost around the occupancy grid from SLAM, an obstacle layer where we perform obstacle ray tracing from observation sources, and an inflation layer that inflates cost to a set radius and positions a cost scaling factor to create a gradient cost around the obstacles.

Nav2 uses the Global costmap to generate a path from its current location to a waypoint using a modified variation of A* called Smac planner. The difference between Normal A* and Smac is that smac produces kinematically possible paths, introduces a costmap sampler, and a path smoother. All of these improvements allow for KiloOhm to compute a smooth, feasible path at 75m within 144 miliseconds(Computer Dependent).

Once the Global path is computed and published, Nav2 takes the 5x5 meter local Costmap and uses a Local trajectory planner to compute the control efforts to move KiloOhm. The local trajectory planner called DWB which is a port of the Ros1 Dynamic window approach algorithm (DWA) takes the global plan, scores the best trajectory based on the plugin critics we provide and then publishes the computed velocity by summing up their scores. Since DWB is not an exact path following algorithm like Pure Pursuit, this gives us the ability to have dynamic obstacle avoidance by tuning in the path align critics to allow for swaying off the current path.

**White Line Detection**

The white line detection (WLD) algorithms found in prior Ohm models have historically had a difficult time handling variable lighting conditions. This is because they relied on a precalibrated threshold to determine the lunanance value white should be considered. While this was very effective in our labspace, the comparably dynamic environment of IGVC caused WLD to become flaky as clouds and shadows came into frame. Fixing this issue was the primary design goal behind the WLD rewrite for the 2022 season.

To improve operations in variable light environments, we now calculate the threshold value as the 3rd standard deviation from the mean luminance value in the image, which allows the threshold to vary per frame. This approach has proven very effective in rejecting changes in light in an IGVC-like environment, including in adverse conditions such as rain.

In order for navigation to recognize the white lines as an obstacle, we convert the detected white pixels to a LiDAR scan, and merge that scan with the real LiDAR scan. This causes SLAM to create obstacles in the occupancy grid in the location of the white lines. This approach allows for the white lines to persist over time, preventing the bot from reversing over them or otherwise navigating over the lines as they leave the frame.

The complete algorithm is as follows:

1. Receive image from Realsense D450 and convert to mono8.
2. (every 5 frames) Compute the average luminance of the image.
    a. Set the threshold value to the 3rd standard deviation from this mean.
3. Threshold the image.
4. Apply an erosion kernel to remove specular highlights, such as from streetlights on pavement.
5. For each remaining pixel
    a. Use a pinhole camera model to get the ray facing out of that pixel in the world.
    b. Solve the ray plane intersection problem between that ray and a flat plane at z=0.
    c. Add this point to a fake LiDAR scan.
6. Concatenate this fake LiDAR scan with the real one, and send that scan to SLAM.

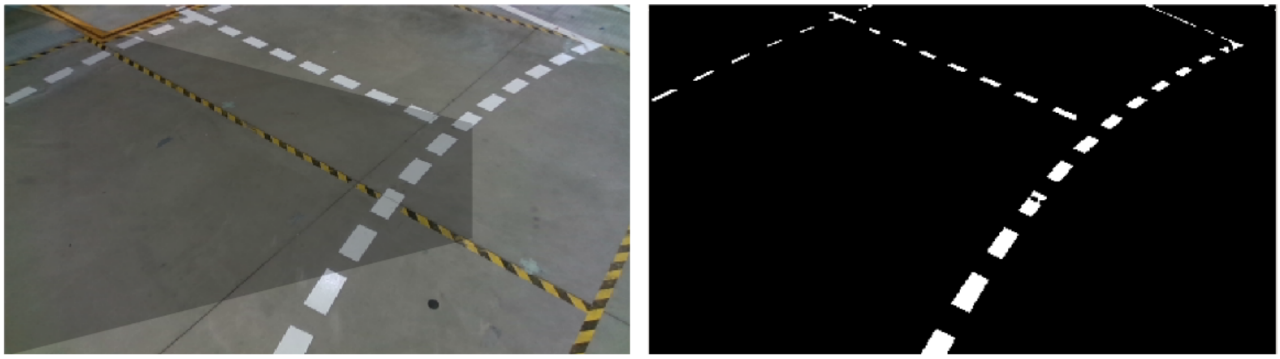

**Figure 9:** A visualization of white line detection

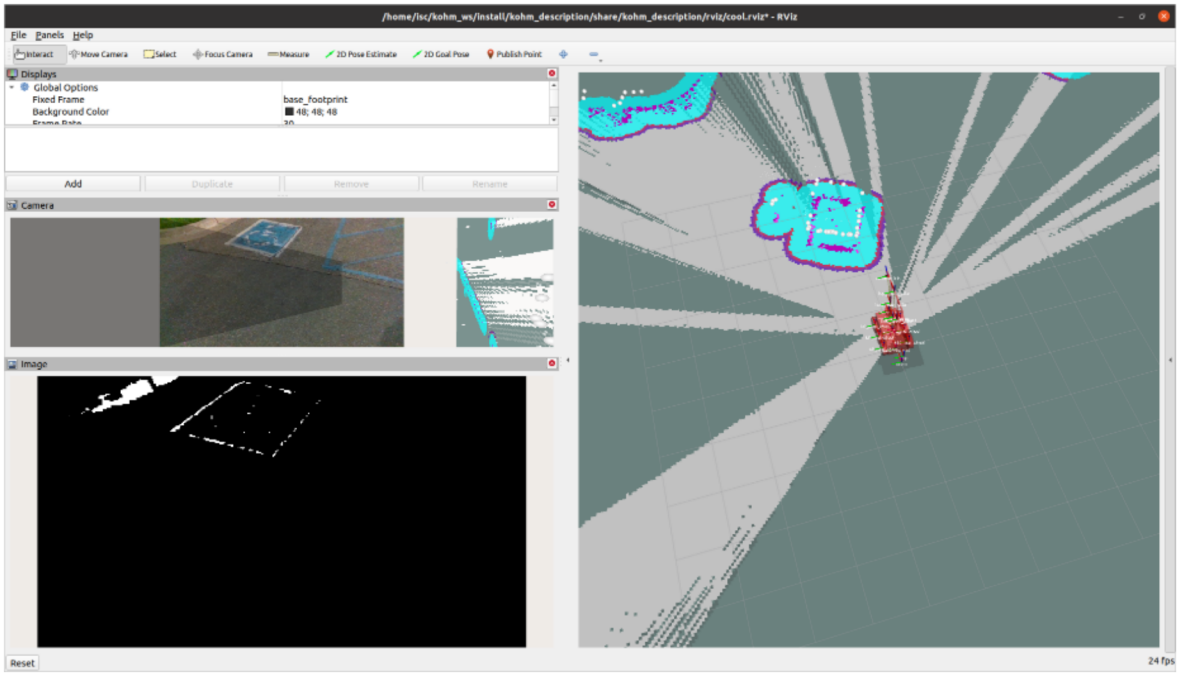


**Figure 10:** White line detection in the dark, with costmap visualization

### Waypoints and GPS

When the robot state transition from teleoperation to autonomous navigation first occurs, the bot will load a file containing a list of GCS latitude longitude coordinates. Once a GPS lock is acquired, KiloOhm will then convert those coordinates to local waypoints about itself using geodetic transformations in sequence. Finally, these waypoints are sent to the navigation stack, which will begin navigation. Navigation will continue until either all points are met, the robot determines a point is unreachable, or the bot is manually brought back into teleoperation.

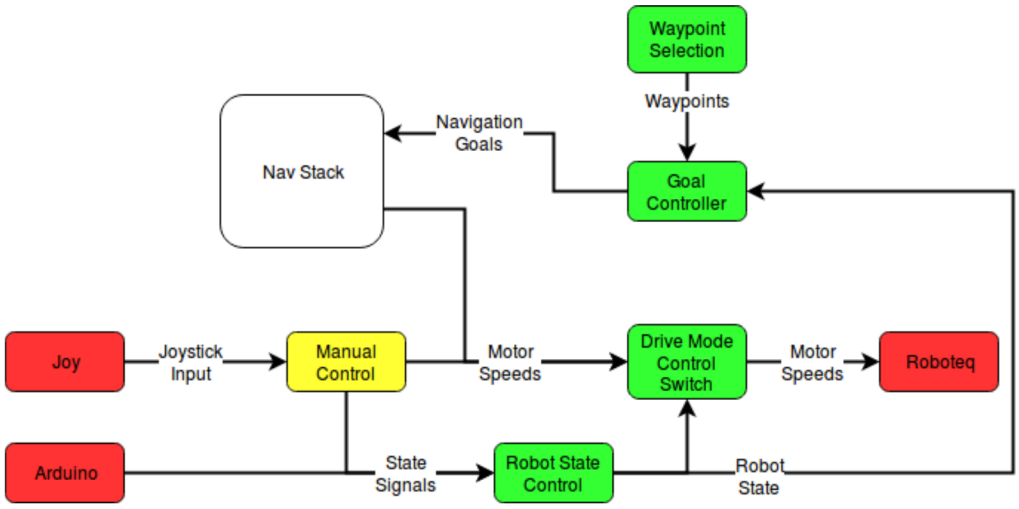


**Figure 11:** Robot state machine and associated modules

### Gazebo

In order to allow for ease of testing and development, we decided to utilize the Ignition Gazebo simulation environment. This piece of software allowed us to create a scale model of KiloOhm with simulated sensors. We were then able to load this model of KiloOhm into a world and feed our software data from the simulated sensors on where the robot is, what obstacles it perceives, etc. Within this world we could then build courses similar to the IGVC course and have the robot autonomously navigate them. This allowed us to debug our software and make improvements without needing the physical robot and allowed for electrical improvements to happen in parallel with software improvements.

**FAILURE MODES**

**Table 7a-c** describes the main points of failure for mechanical, electrical and software subsystems, along with severity and mitigation actions.

**Table 7a: Failure Modes Mechanical**

| Possible Failures | Likelihood | Severity | Action |
|---|---|---|---|
| Water infiltration damaging components | Low | High, Damaged components may end competition | Panels deflect water and electronics are enclosed. |
| Payload | Low | Moderate, End of run | Payload sits in an enclosure and is tied down |
| Casters lockup during turning | Moderate | Low | Test orientation of caster wheels in front and back |

**Table 7b: Failure Modes Electrical**

| Possible Failures | Likelihood | Severity | Action |
|---|---|---|---|
| Wires come unplugged from Arduino | Moderate | Low | Implement new Arduino screw shield |
| Solenoid breaks/ /fails to operate | Low | Low, fails safe | Ensure solenoid is used within specified limits |
| Physical or wireless e-stop fails | Low | High, End of run, possible DQ | Have spare relay, thorough testing to ensure risk mitigation |
| Battery dies early | Moderate | Low | Monitor battery with Arduino system |
| Wire comes loose and shorts to chassis | Low | Moderate, would trip circuit breaker | Ground chassis |

**Table 7c: Failure Modes Software**

| Possible Failures | Likelihood | Severity | Action |
|---|---|---|---|
| Poor GPS Fix | Moderate | Moderate, performance degradation | Convert waypoints to robot-local frame, wait for better conditions |
| Sensor Disconnect | Low | High | Secure sensor connections |

| | | | |
|---|---|---|---|
| | | | to mast |
| Inconsistencies in map | Moderate | Low | Connect inconsistencies in the map using image editors |
| Localization inaccuracy | Low | Moderate, performance degradation | Refresh map and position in map, provide more accurate initial position, slow down robot |

## PERFORMANCE TESTING

**Table 8: Performance Summary**

| Category | Requirements | Analysis |
|---|---|---|
| Speed | 2.2m/s | Tune software to limit speed as little as necessary |
| Ramp | Capable of climbing up to 30° incline. | Tested on varying inclines. Confident to 30° |
| Reaction Times | Maintain a system update rate of 10Hz | Take advantage of configurable output rates, and limit rates in software where necessary |
| Battery Life | 30 minutes in the parking lot. | Performed endurance test on grass. Actual runtime ~1 hour |
| Distance of Obstacle Detection | Maximum obstacle detection with LiDAR is 20m. Robot reacts within 2m. | Tune robot to react within larger radius if necessary |
| Distance of Lane Detection | Detect white lines | Camera can effectively see only ~4m ahead and 1.5m on either side |
| Behavior in Dead end situations | Capable of navigating out of a dead end | Have robot turn in place/backup until a path can be found |

## CONCLUSION

This year, the main goal was to fully implement and optimize the electrical circuit on the new mechanical chassis, implement simulation so that software could be tested during this period, and rewrite more robust and functional code for important modules, such as SLAM, encoders, and white line detection. Projects were broken down into small tasks that allowed new members to learn about the system while making progress on the robot. This allowed the large number of new members to gain hands-on experience while creating a robotics platform that had been repeatedly tested and optimized for competition.