Trinity College

SQRL

Chris Rowe (christopher.rowe@trincoll.edu)

Kirk Boyd (kirkland.boyd@trincoll.edu)

Digesh Chitrakar (digesh.chitrakar@trincoll.edu)

Divas Subedi (divas.subedi@trincoll.edu)

Mark Morales (mark.morales@trincoll.edu)

Hunter Badey (hunter.badey@trincoll.edu)

John Anthony Rosa (john.rosa@trincoll.edu)

Faculty Advisors: Professors John Mertens & Kevin Huang

Date Submitted: May 22, 2019

May 17, 2019

Dear IGVC,

Trinity College's entry this year is "SQRL" (also affectionately known as "squirrel"). SQRL has undergone a complete software and electrical system redesign from scratch. Much of the mechanical system has been redesigned and rebuilt. We have a team of seven students this year. Six are participating on our iGVC team for the first time this year. We are making our final decision as to our participation on our May 23rd demo day. We hope to see you in Michigan!

Best,

Dr. John D. Mertens
Professor of Engineering
Trinity College
300 Summit St.
Hartford, CT 06106

Team

Introduction

SQRL is an autonomous ground vehicle constructed by Trinity College Robotics Team. It is designed to navigate the IGVC track based on GPS waypoints while avoiding obstacles. It was built to withstand light rain conditions and travel across difficult terrain.

Organization

Members	Major	Role
Chris Rowe 20’	Computer Engineering	Software, Electrical
Kirk Boyd 21’	Mechanical Engineering	Mechanical, Electrical
Digesh Chitrakar 22’	Electrical Engineering	Software
Divas Subedi 22’	Electrical Engineering	Software
Mark Morales 22’	Mechanical Engineering	Mechanical
Hunter Badey 22’	Mathematics	Mechanical
John Anthony Rosa 21’	Biomedical Engineering	Mechanical

Design Methodology

Although SQRL inherited the main chassis and various electrical components from previous Trinity IGVC iterations, the majority of the features are newly designed. Using the main chassis as a base, a new housing structure was built. Using the power converter and fuse assembly, a new electrical system was designed. From there, the software for perception, mapping, localization and path planning were implemented using new algorithms and packages.

Innovation

- The frame holding the payload is adjustable to maximize stability of the payload.
- Various parts were 3D printed to custom fit the dimensions and needs of the robot.
- A new housing design was for the electrical system was designed, emphasizing compactness, modularity and plug-and-play functionality.

Mechanical Design

Overview

The chassis of SQRL is a modified PerMobil Trax all-terrain wheelchair. This chassis can support a payload of over 250 lbs and has a footprint of 55" by 28". It features a differential front wheel drive system – a pair of 500W Leroy Somer MBT1141S motors - and a pair of rear mounted casters. The motors are geared with a 25.8:1 ratio, providing 15 ft-lb of torque. 1”x1’’ T-slotted aluminium extrusions are used for the structure upon which an acrylic outer shell is mounted. These allowed for quick and easy component layout without compromising mechanical strength. The framings are mounted with M5 screws and drop in nuts. Mounting frames were constructed using 1.5”x1.5” extrusions. To reinforce the acrylic outer shell there are 3D printed corner pieces which minimize tension. The payload rests on the back face of SQRL for easy access, while the battery sits inside the robot.

Safety and Improvements

The electronic stack is designed for accessibility and efficiency. The entire stack is 3D printed with precise measurements to allow for quick assembly. There is a 3D printed board mounted to the voltage converter board that allows it to simply slide in.

In terms of improving the weather-proofing of SQRL, an acrylic outer shell was constructed to give SQRL resistance to water and wind. Joints and seams of the outer shell are sealed with silicone caulk. Furthermore, each outer sensor on the robot is encased in its own 3D printed enclosure that serves to not only weatherproof the devices but also to stabilize them during rough operation.

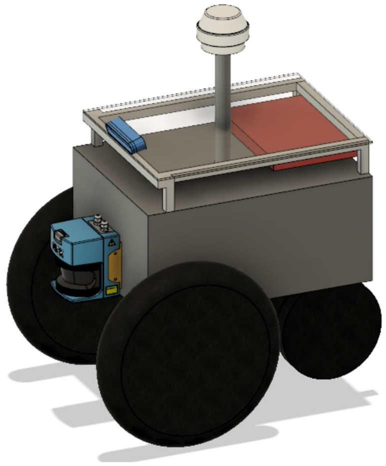

Figure 1: CAD Model for the chassis

Electrical System

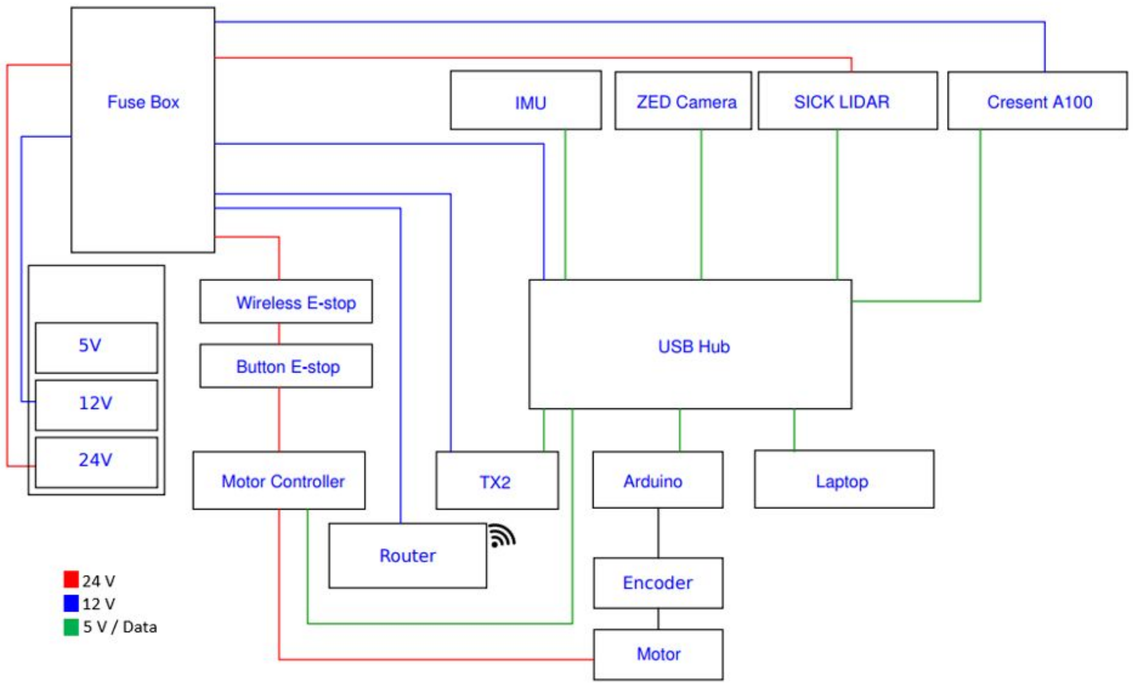

Figure 2: SQRL Electrical Diagram

Power

Power is provided using two 12V batteries in series. The voltage converter board from previous iterations features separate DC-DC converters that output 24V, 12V and 5V. To secure sensitive electrical components from transients, high frequency feedback, and surges from the motors, a 10A Radius Power Filter model RP220-10-4.7 was added to the motor line. In addition to the filter, fast-blow fuses have been added for all major electrical components to secure them in the case of accidental overloads.

To distribute power, previous iterations used 3 simple power busses, one for each voltage. Each connection was made using ring or spade terminals for both PWR and GND, and secured using screws. This required over 40 wires to connect each of the rails to their voltage terminal, and each sensor to their voltage rail. Due to this, it was not an easy task to detach and re-attach any singular component. The new design features a more elegant solution.

Electrical Stack Design

The electrical system was stripped bare, its mapping re-evaluated, and renewed with fresh wires. As part of the re-evaluation, emphasis was placed on modularity. The electrical stack is composed of 3 modules that stack together. Each module houses one of the three subsystems for power distribution. At the base is the voltage converter, which takes the raw 24V from the battery, passes it through an EMI filter and uses DC-DC converters to step-down the voltage to 12V and 5V. The second layer comprises the rails for the raw 24V. This supplies the motor controller circuit directly and serves as the main switch for the voltage converter.

The top layer functions as a hub, supplying a row of ports for each of the three voltages. From below, the converted voltages connect to their individual fuses, situated at one side for easy inspection and replacement, and then to their respective row. Terminal connectors for all components were replaced with DC barrel jack connectors (Type-D). This eliminates the need for two individual cables for PWR and GND, and allows all components to be easily plugged in and out.

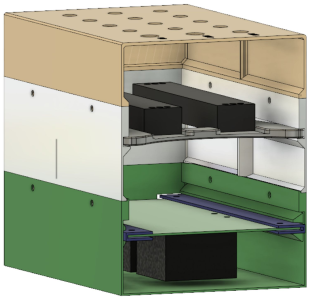

Figure 3a: Layers of the E-Stack

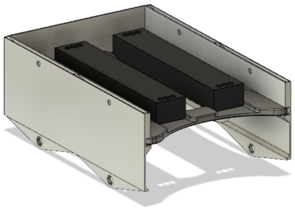

Figure 3b: Middle Layer of the E-Stack

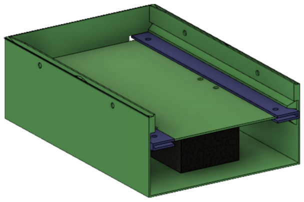

Figure 3c: Bottom Layer of the E-Stack

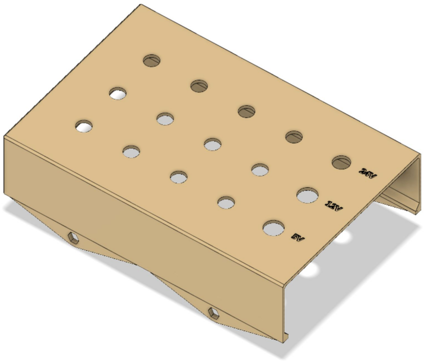

Figure 3d: Top layer of the E-Stack which houses the female barrel jack connectors.

Figure 3e: Power rails that slide into middle layer of the E-Stack

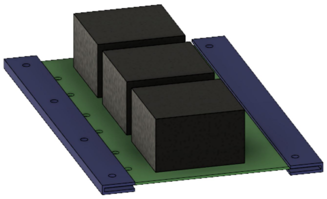

Figure 3f: Voltage converter that slides into the bottom layer of the E-Stack

Processors and Sensors

- The NVIDIA Jetson TX2 board functions as the main processor. The images are acquired using the ZED stereo camera. The path is executed via an AX3500 motor controller which controls the two DC motors.

- The SICK LMS 291-S05 sensor is used for obstacle detection. This sensor is powered by 24V and has a 180° field of view. The SICK can display information up to 80 meters (262.46 ft) away. Once the data is received it is sent to the computer through a USB-Serial adapter and publishes a `LaserScan` message to the `/scan` ROS topic.

- The Crescent A100 Differential GPS is used to find the position of the robot in latitude and longitude. It is accurate to less than 60 cm. It uses Space Based Augmentation Station (SBAS), more specifically, the Wide Area Augmentation System (WAAS) to calculate its coordinate location. The GPS is placed on top of the chassis of the robot so that nothing obstructs the GPS. The Crescent A100 is connected to 12V and data is sent through a USB-Serial adapter and publishes a `NavSatFix` ROS message under the `gps/` namespace.

- The Adafruit BNO055 Absolute Orientation Sensor (IMU) is used to measure linear acceleration, angular acceleration, and heading.

- Arduino is used to interface with the IMU and wheel encoders. It functions as the ROS NodeHandler for those two sensors and publishes to the `/imu` and `/odom` ROS topics respectively. Additionally, Arduino is used as the microprocessor for the control panel.

Software

Overview

The NVIDIA Jetson TX2 is flashed with an Ubuntu 16 derived Operating System called Linux for Tegra (L4T). The main development tool used is Robot Operating System (ROS), due its extensive library of perception, mapping and navigation algorithms. Additionally, Arduino is used to interface with several hardware components as mentioned above, and interfaces with ROS through the `rosserial_arduino` package.

Autonomous Navigation

The workflow of SQRL's autonomous navigation software can be broken down into three stages: building a map and localizing the robot within that map, planning the best path to reach the destination according to that map, and converting the map to motor controls.

The ROS navigation stack provides the platform for autonomous navigation. It performs as a system of packages, each providing a core functionality. The `move_base` package incorporates GPS, odometry and sensor data to evaluate the best path, and provides the robot with the associated velocity commands. It provides the interface between the path-planning nodes, `base_local_planner` and `base_global_planner` and the costmap nodes, `local_costmap` and `global_costmap`.

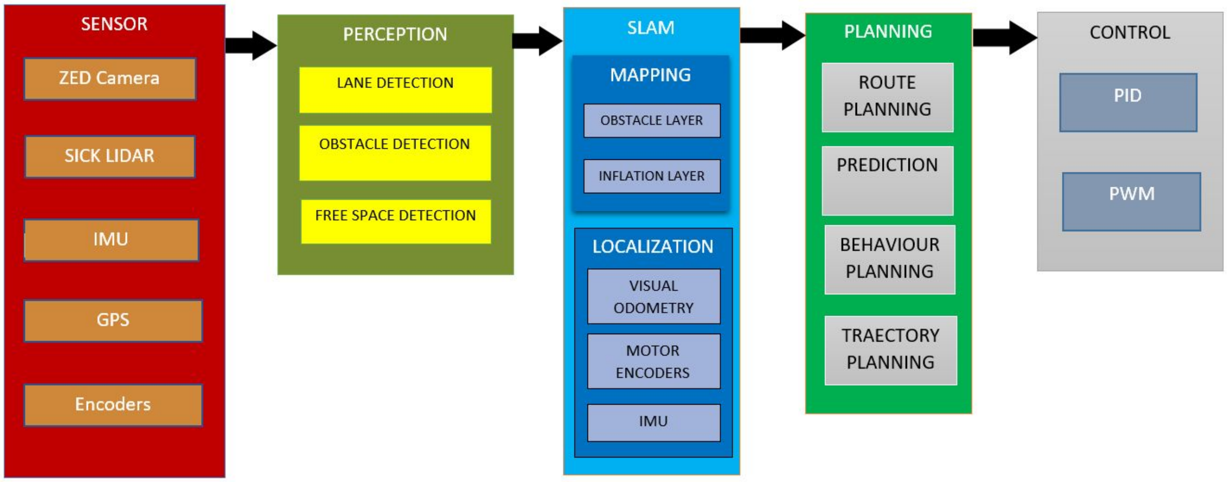

Figure 4: Autonomous Navigation Workflow

Simultaneous Localization and Mapping (SLAM)

Odometry is obtained by fusing the visual odometry from the ZED, the wheel movement from the encoders and inertial data from the IMU. Incorporating this, the `amcl` node performs probabilistic localization by implementing the Adaptive Monte Carlo algorithm using particle filters, thus keeping track of the robot's pose in relation to its environment.

The `rtabmap` package uses stereo image data to perform Real Time Appearance Based (RTAB) SLAM. Real Time Appearance Based mapping is a variation of GraphSLAM. The stereo camera captures image and depth data concurrently, which is used to identify features in the environment. When encountering a match of features between two poses, loop closure occurs, which greatly increases the accuracy of the map. Otherwise, when similar features are detected at different poses and loop closure is not used, that location is identified as new, rather than one previously encountered, thus creating duplicates of the same area.

By fusing the resulting obstacle point clouds along with obstacles detected by the LIDAR, a 2D occupancy grid map can be generated, where each point in space is designated as free, occupied or unknown. Additionally, A costmap is then generated by converting the ternary representation of the occupancy grid map into a spectrum of 255 possible cost values. It uses the robot's footprint to propagate the cost values out from occupied cells that decrease with distance. This inflated map is then used to plan a path.

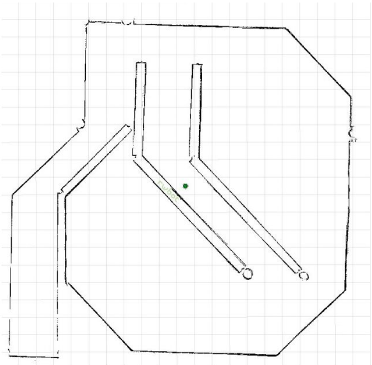

Figure 5a: Occupancy grid map

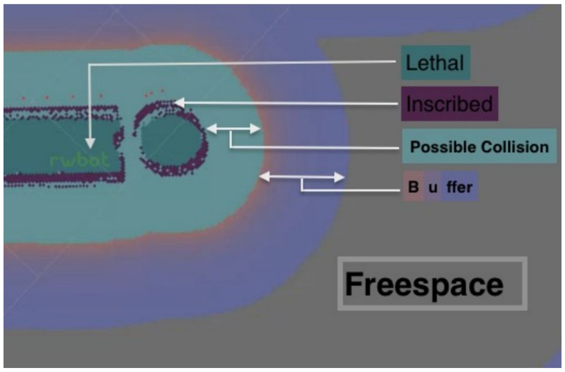

Figure 5b: Local costmap value spectrum

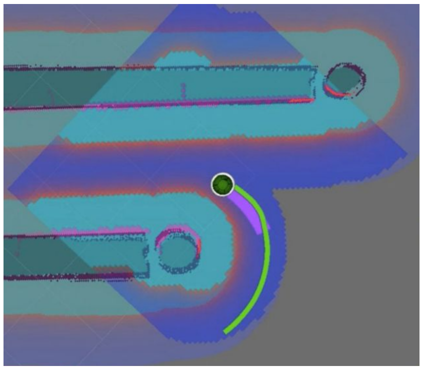

Figure 5c: Planning lowest cost path

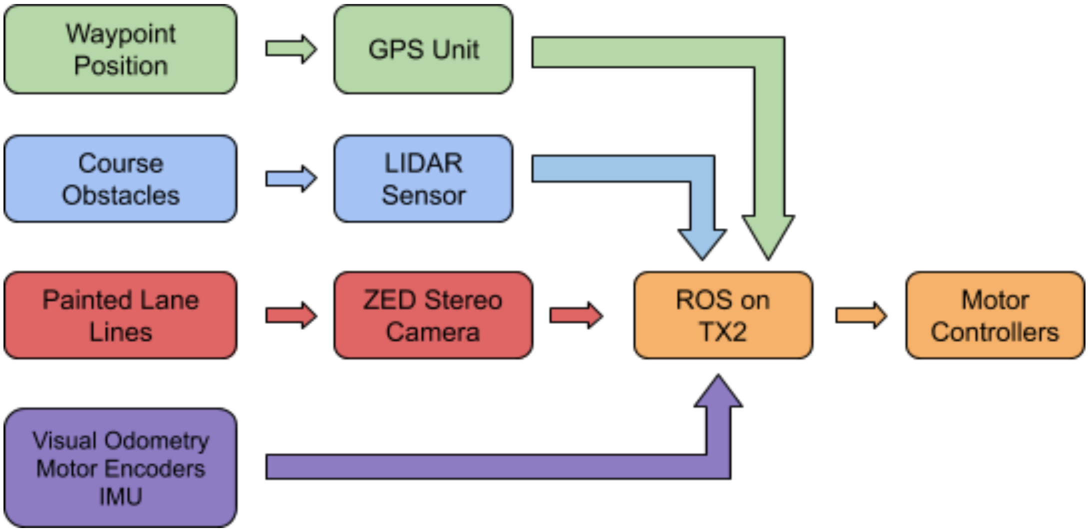

Figure 6: Sensor Integration Overview

Obstacle Detection

Obstacles that will be encountered on the course include construction barrels/drums of various colors, trees, shrubs, light posts and street signs, and are randomly placed on each run. The SICK LIDAR will be used to generate a planar scan of the 180° field of view in front of SQRL. The SICK is mounted on the front bumper of SQRL, and detects all objects at a height of 0.38m. While this only detects objects at that specific height, none of the possible obstacles include any objects where part or all of its body is hanging above that range, so this solution will have no blind spots.

Additionally, the 3D point cloud generated by the ZED will also used to identify any obstacle. Any objects detected within the height range of SQRL will be treated as a possible obstacle. Additionally, there will be potholes simulated as solid white circles 2 feet (0.6m) in diameter. However, since these are not physical, they require a different method of detection.

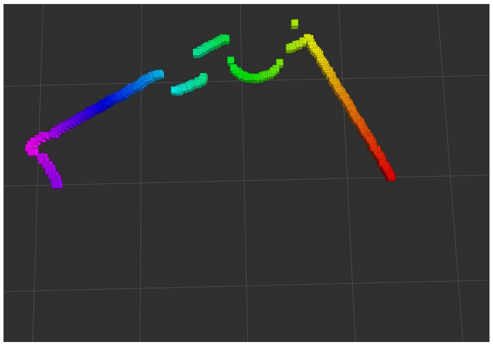

Figure 7a: Example point cloud received from SICK scan

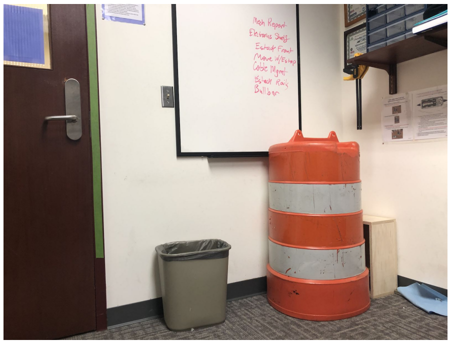

Figure 7b: Ground truth of SICK scan

Lane Detection

The ZED stereo camera generates a 3D point cloud of the target area. Similar to a normal camera, the scene is composed of RGB values. But additionally, each RGB value has a depth value, which corresponds to its XYZ point in space with respect to the camera frame. To exclude the white lines present on the construction barrels, a pass through filter is applied. This effectively crops out all points with a Z value greater than the 0.15m, leaving only points on or very near to the ground.

To detect the lanes, the scene must be examined from a bird's eye view. The image from a non-stereo camera would need to undergo a perspective transform to achieve this view, but since the X, Y and Z values are known for each pixel, the point cloud can simply be rotated into a bird's eye view. Using this top-down view, the scene can be treated as a 2D image. To extract the lanes, a threshold filter is applied to remove all non-white components, and then a 2D canny edge detection and hough transform is performed. With the lanes extracted, the image is compared with its top-down point cloud counterpart. The X and Y pixel values of the lane are matched to its corresponding voxels. The identified voxels are extracted and used as input to update the occupancy grid map.

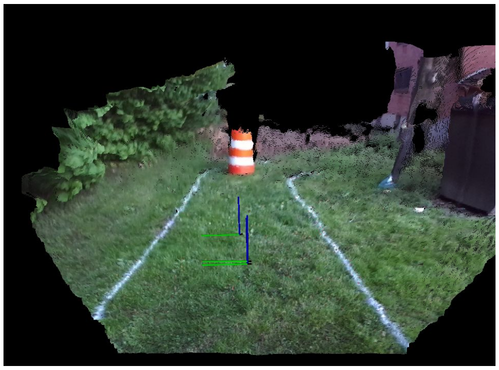

Figure 8a: Example Point Cloud

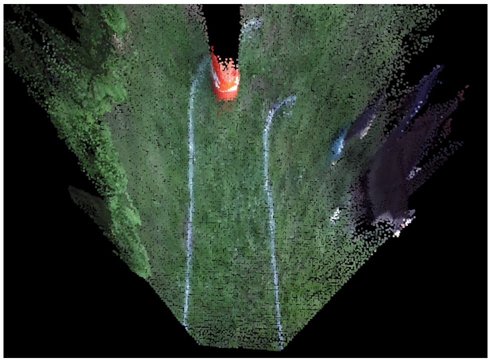

Figure 8b: Bird’s Eye View of Point Cloud

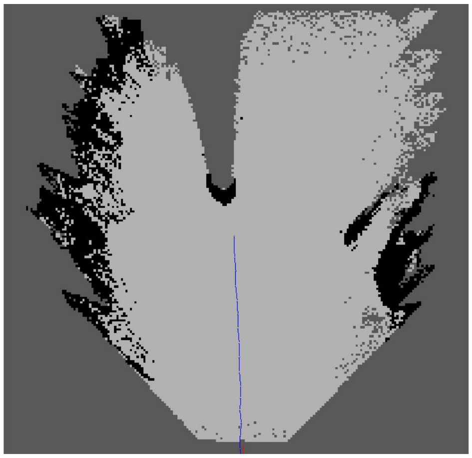

Figure 8c: Occupancy Grid Map from Point Cloud

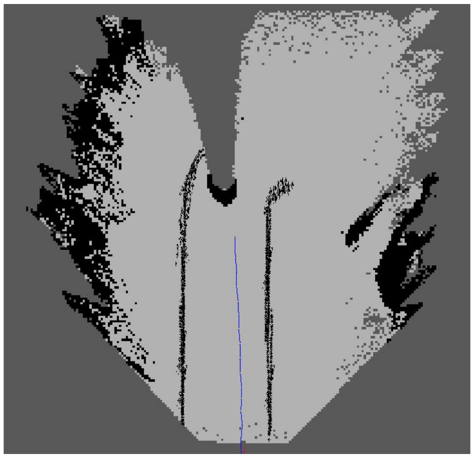

Figure 8d: Detected lanes overlayed onto Occupancy Grid Map

This method also succeeds in detecting the potholes. The criteria used for the lanes, namely low height and continuous white segments, also matches the features of the simulated potholes, which are 2 feet (0.6m) diameter solid white circles.

Safety

SQRL is a 250+lb machine with very powerful motors. Consequently, several safeguards are implemented in the event that the autonomous software fails and SQRL exhibits erratic and dangerous behavior.

Emergency Stop

The Roboteq AX3500 motor controller features separation of the microcontroller power supply and motor power supply. Additionally, the power stage supplying the motors features RDSon MOSFET transistors. This enables a convenient method of killing the motors since the microcontroller needs to be active to enable the motor's power stage MOSFETs.

The hardware E-Stop functionalities are implemented using this feature. The mechanical E-Stop is performed using a big, red button on the top, center, rear of the chassis. When the button is depressed, the motor controller microcontroller is grounded, and unable to enable the motor's power stage MOSFETs. The wireless E-Stop interface is implemented using NRF24L01 2.4GHz transceiver modules. With an attached RP-SMA antenna, it can achieve a range of over 1 km. Additionally, the motor controller has a watch-dog functionality, where if the motor controller stops receiving motor commands from the TX2, it shuts down the motors until commands are being sent again, at a steady rate.

Within the autonomous navigation program, software versions of the emergency stops are also implemented. The software watch-dog listens to the `/cmd_vel` topic, and publishes zero velocities if the topic has not been updated.

Vehicle Failure Points

SQRL is front-heavy, which makes its moment of inertia a potential failure point. This issue makes it so that whenever SQRL comes to an abrupt stop, it will tip forwards. To reduce tipping, the payload was placed further behind the robot's geometric center to shift the moment of inertia. Additionally, by smoothing the deceleration of SQRL the tipping can also be reduced.

Another potential failure point of SQRL is that the dimensions of the base of the robot and the dimensions of the frame of the robot are different. As these dimensions did not match the frame of SQRL it was difficult designing various aspects of the robot. Also, SQRL is designed to withstand light-rain weather conditions. Through the acrylic outer shell the wiring, the E-stack, and the computer are kept safe from exterior contaminants. To prevent any damage to internal hardware 3D printed casings were place around the sensors and other essential hardware.

Bill of Materials

Component List	Price	Cost Incurred
NVIDIA Jetson Tegra TX2 Development Board	$350	$350
ZED Stereo Camera	$450	$450
SICK	$4500	$0
Crescent A100 DGPS	$1500	$0
Roboteq AX3500 Motor Controller	$400	$0
24V battery	$100	$100
Caster Assembly	$150	$0
Chassis	$650	$0
Motors	$1000	$0
Voltage Converter	$150	$0
Remote Control	$50	$50
Wiring/Electrical	$100	$100