INTELLIGENT GROUND VEHICLE COMPETITION

Final Report

by

Renard Dichoso, EE - AutoCad Design & Editing
Justin Heisterkamp, EE/CpE - Computer Vision & Project Management
Kyle Hernandez, EE - Power Design & Simulation
Trevor Novisk, EE - E-Stop Configuration
Warren Shields, CpE - Software Dev. & Navigation
Christophe Vautier, EE - Power Distribution & Management
Jonathan Zeigler, EE - IMU configuration & Data archival

Old Dominion University
Electrical and Computer Engineering Department
Fall 2021

Faculty Advisor:
Dr. Lee Belfore

Honor Pledge:

"I [we] pledge to support the honor system of Old Dominion University. I [we] will refrain from any form of academic dishonesty or deception, such as cheating or plagiarism. I am [we are] aware that as a member of the academic community it is my responsibility to turn in all suspected violations of the Honor Code. I [we] will report to a hearing if summoned."

Abstract

The purpose of this project is to prepare Old Dominion University's intelligent ground vehicles (Little Blue v1.0, Little Blue v2.0, and Monarch I) for the 29th Intelligent Ground Vehicle Competition (IGVC) in 2022. The next competition occurs during the first weekend of June 2022 (the 3rd to the 6th) at Oakland University in Rochester, Michigan. Little Blue v1.0 and Little Blue v2.0 are uncrewed robotic vehicle designs intended to participate in the IGVC's "Auto-Nav Challenge." Monarch I is a crewed (two-person capacity) autonomous vehicle design based on the Polaris GEM e2 electric vehicle that is intended to participate in IGVC's "Self-Drive Challenge." Little Blue v1.0 was constructed by a previous team but requires further software development and possible updates to the hardware and sensor components. Little Blue v2.0's chassis and suspension are currently under construction by our complementing mechanical engineering team, and it also requires a power and drive system design from our team. Monarch I's design is near completion but still requires extensive software development to integrate the sensors with the drive-by-wire system. Our team's goals are to finish the software development for Little Blue v1.0; aid the mechanical engineering team's construction and development of Little Blue v2.0; and complete the software development for Monarch I. Design objectives include using the US Army's Robotics Technology Kernel (RTK) which is based on the Robot Operating System (ROS). ROS nodes facilitate the communication framework between the various sensor and processing systems, e.g., computer vision for lane following and object detection. Furthermore, the software frameworks will need to support the construction and analysis of navigational cost maps, onboard inertial navigation, and the Global Positioning System (GPS) for localization.

Contents

List of Figures

List of Tables

1. Introduction

1.1. Design Problem

The Intelligent Ground Vehicle Competition, IGVC, challenges student teams to design and build two autonomous vehicles (AVs), an uncrewed robotic vehicle and crewed AV - each designed to navigate obstacle courses specific to their respective format. Specifically, the robotic AVs compete in the Auto-Nav Challenge and the crewed AVs compete in the Self-Drive Challenge. Both variants are eligible to participate in the IGVC's Design Competition Challenge (see the following subsections for details) [1]. Our senior design team's objective was to prepare ODU's legacy and prototype vehicles Little Blue v.1.0, Little Blue v2.0, and Monarch I for the upcoming 2022 IGVC. Our design process to meet this goal tested our teams' ability to analyze, implement, and integrate computer vision, machine learning techniques, sensors, electrical control, and power systems.

1.2. Auto-Nav Challenge

The objective of the Auto-Nav challenge is to design a robotic ground vehicle that autonomously navigates an outdoor course while maintaining a minimum speed of one mile per hour (mph). The vehicle must not exceed five mph, remain within the course's lanes, and avoid obstacles. The challenge will be on an asphalt pavement course that will be approximately 100 feet wide by 200 feet deep - resulting in a 600-foot long track. Three-inch wide painted white lines will define the track that the vehicle must detect and not breach, with the consequence of point deductions. The obstacle course will consist of ramps not exceeding 15% inclines, construction barrels, and cones with varying colors randomly placed on the course. Both natural and artificial obstacles such as trees, shrubs, light posts, and street signs may also appear on the course with randomized placement. The ideal performance of the AV should be focused on identifying obstacles and moving objects. Additionally, traffic violations such as crashing, crossing the painted boundary, and driving under the minimum speed limit will result in deducted scores or automatic disqualification. See Appendix A for details on the vehicle configuration requirements.

1.3. Self-Drive Challenge

The purpose of the Self-Drive challenge is to autonomously navigate a course that simulates many different traffic situations involving staying inside a lane, avoiding pedestrians,

identifying and obeying traffic signs, and parking. The vehicle must be two-person, four-wheel equipped with a mechanical and wireless e-stop. The e-stop must bring the vehicle to a complete stop. The vehicle requires yellow reverse lights and red brakes lights. For additional safety, a strobing roof light should indicate when the vehicle is operating autonomously. See Appendix B for details on the vehicle configuration requirements.

1.4. Design Competition

The Design Competition will require the team to submit a report which presents the vehicles' design processes and components. Submitted reports must document the used hardware and electrical components such as sensors, computers, actuators, along with the design planning process and analyses that lead to predicted overall performance. The report must describe safety, reliability, and durability considerations of the design. This competition requires an oral presentation that highlights the written report and any updates of the design. The team's vehicle will be examined and judged based on its packaging, serviceability, ruggedness, safety, originality and appearance.

2. Design Approach

2.1. Design Strategy

Our team was focused on vertically integrating the legacy designs with our work on existing IGVC vehicles: Little Blue v1.0, Little Blue v2.0, and Monarch I. Both Little Blue v1.0 and v2.0 are designed to compete in the Auto-Nav Challenge, and Monarch I the Self-Drive Challenge. A previous team assembled the hardware and sensor components for Little Blue v1.0, but the software development was incomplete. Little Blue v2.0 is an updated architecture, hardware and sensor setup that is intended to integrate modern technologies, such as the ZED 2 stereo camera. Monarch I is a Polaris GEM e2 vehicle that has been equipped with a drive-by-wire system made by AutonomouStuff, a company that specializes in AV technology for research and development.

Our team's goals are to finish the software development for Little Blue v1.0; aid our mechanical engineering team's construction of Little Blue v2.0 and develop a power and drive system; and complete the software development for Monarch I. Design objectives include using the US Army's Robotics Technology Kernel (RTK) which is based on the Robot Operating System (ROS). ROS nodes facilitate the communication framework between the various sensor

and processing systems, e.g., computer vision for lane following and object detection. Furthermore, the software frameworks will need to support the construction and analysis of navigational cost maps, onboard inertial navigation, and the Global Positioning System (GPS) for localization. Development on all vehicles will focus on receiving, interfacing, and processing computer vision, lidar, global position system (GPS) data, and inertial measurement sensor data to perform lane detection, object detection, avoidance, and localization.

2.2. Economic Constraints

Each version of AGVs contains a separate budget for the forward development of the individual vehicles. The Monarch I is being developed under a contract with the US Army Combat Capabilities Command (DEVCOM) Ground Vehicle Systems Center. The Little Blue v1.0 / v2.0 do not have an allocated budget from the ECE 487 team for the current Fall 2021 semester. Any funding for the Little Blue v1.0 / v2.0 will be at the discretion of the ECE department of the Frank Batten College of Engineering until the ECE 487 and MAE 435 team receive a budget. The following sections describe the constraints based on the current inventory, which is provided in lists by vehicle and universally interchangeable parts in Appendix C.

2.3. Sensor Constraints

Our current sensor inventory includes two ZED 2 Stereo Cameras, one RPLIDAR A2, one Velodyne VLP-16 Lidar, two ZED-F9P RTK GPS/GNSS breakout, and two XSens MTi-670 DK IMUs. The ZED 2 Stereo Camera has a 120-degree wide-angle field of view with a built-in Inertial Measurement Unit (IMU), barometer, magnetometer, and temperature sensors. The ZED API includes neural network libraries ready for object detection, depth sensing, and point cloud analysis. The RPLIDAR A2 and Velodyne puck LiDAR are sensors that provide ranging data in a 360-degree field of view. The ZED-F9P RTK GPS/GNSS breakout boasts positional data accuracy within millimeters. However, this requires access to a Networked Transport of RTCM via Internet Protocol (NTRIP) enabled base station for correctional data. Finally, the XSens MTi-670 DK IMU is an additional inertial measurement unit for tracking localization data. The XSens MTi-670 DK IMU also includes a Global Navigation Satellite System (GNSS) on the daughter card.

2.4. Hardware Constraints

See Appendix C for a full list of the hardware components specific to each vehicle.

Little Blue v1.0 - The competition rules have several requirements for qualification - please see Appendix A and B for a complete list. These outline the permitted vehicle specifics such as propulsion methods, vehicle size, and safety requirements. For example, each vehicle in the Auto-Nav Challenge must be able to travel within a 1-5 mph velocity range and may not use fuel cells or combustion engines. Thus, Little Blue v1.0 and Little Blue v2.0 have been designed to primarily run from battery powered electric drive systems.

Little Blue v2.0 - The four Kunray hub-motor wheels ordered for this design each came with a control unit but the forerunning teams experienced precision issues with them. Our team has contributed to an improvised control system design to replace the factory control units but will need to pass the design work onto the succeeding team. During our design process we found a sustained maximum current draw (14 A) from all four wheels, simultaneously, has the potential to drain the dedicated 24V 40Ah battery in approximately 40 minutes. A run through the 600-foot long Auto-Nav Challenge track at one mph should take ~11 minutes. Thus, the current battery capacity is expected to be suitable for the course but this constraint must be considered during prolonged testing.

Monarch I - The Polaris GEM e2 is equipped with the long range AGM battery bank that consists of 8 - 6V 224 Ah batteries wired in series. This provides the power source for the 48V AC induction electric motor for the front wheel drive system. The GEM e2 is advertised to have a maximum range of 35-60 miles, depending on the environment and operating conditions. With speeds capable of 25 mph, we are only required to run the Self-Drive Challenge at 1 to 5 mph, which will not demand as much power from the battery. The vehicle AGM battery should be recharged after extensive and frequent use to maintain a healthy state of charge and life cycle, however the service manual only requires charging the vehicle every 10 days while idle.

2.5. Software Constraints

The team will continue to use ROS as the primary software framework due to the contractual agreement with our sponsor. The US Army requires using their Robotics Technology Kernel (RTK), a ROS-based modular autonomy software library [2]. This core library requires ROS version “Kinetic Kame,” which relies on Ubuntu Xenial v16.04 as the underlying operating system to support the RTK. Fundamentally, ROS package nodes wrap each control component interface to publish or subscribe data to other nodes [3]. This allows several different

components to communicate and synchronize autonomous actions, such as the stereo camera and sensor data inputs for orchestrating decisions. Our team will continue to use the RVIZ and Gazebo packages for ROS visualization and open-source simulation to test the proposed and existing navigation algorithms for modification and updating.

To further satisfy the contractual agreement with the US Army, any hard drive partitions that include RTK packages and software must be encrypted. Consequently, a LUKS-encrypted installation of Ubuntu Xenial v16.04 with ROS Kinetic Kame and the RTK suite have been written to an image file for dissemination among the approved team members for software development. This image also contains a Moodle course provided by Bob Jones University which covers both ROS and RTK.

2.6. Software Logic Design Philosophy

Our software logic design philosophy for Little Blue v1.0/v2.0 and Monarch I follow the model provided in Figure 1. Note that this is a generalized abstraction and that each platform has nuances from this model. Upon initialization our AVs must immediately identify their current status, map their surroundings, and establish localization data. GPS, Inertial Measurement Unit (IMU), Stereo Camera, and Lidar sensor systems will be used to collect positional, orientation, visual, and ranging data for the ROS navigation stack.

The ROS navigation stack consists of packages that receive data from odometry, sensor streams, an assigned waypoint, or goal pose to compute output velocity commands. The navigation stack includes robot_localization and move_base. The robot_localization package is a collection of state estimation nodes that track the 15-dimensional states of the vehicle - estimations of X, Y, Z relational coordinates and roll, pitch, and yaw [4]. The move_base package allows movement of the vehicle to designated destinations by using the navigation stack [5]. The navigation stack serves as a major component of the C&D system as it makes the navigation decisions based on the received inputs.

An OpenSlam Gmapping (ROS software package) node has been tested to collect the Simultaneous Localization and Mapping (SLAM) data after we installed the sensor array on Monarch I. This information is critical for the future SLAM data integration nodes and navigating the vehicle. These nodes will establish a cost-map (See Figure 2) that will be used to identify available routes. Integrating this data with the inertial navigation systems (INS) and GPS

data will generate accurate localization data for the core Command and Decision (C&D) unit to process. C&D will be responsible for handling all the data streams to produce the vehicles' appropriate acceleration and maneuvering response signals for the drive controllers.

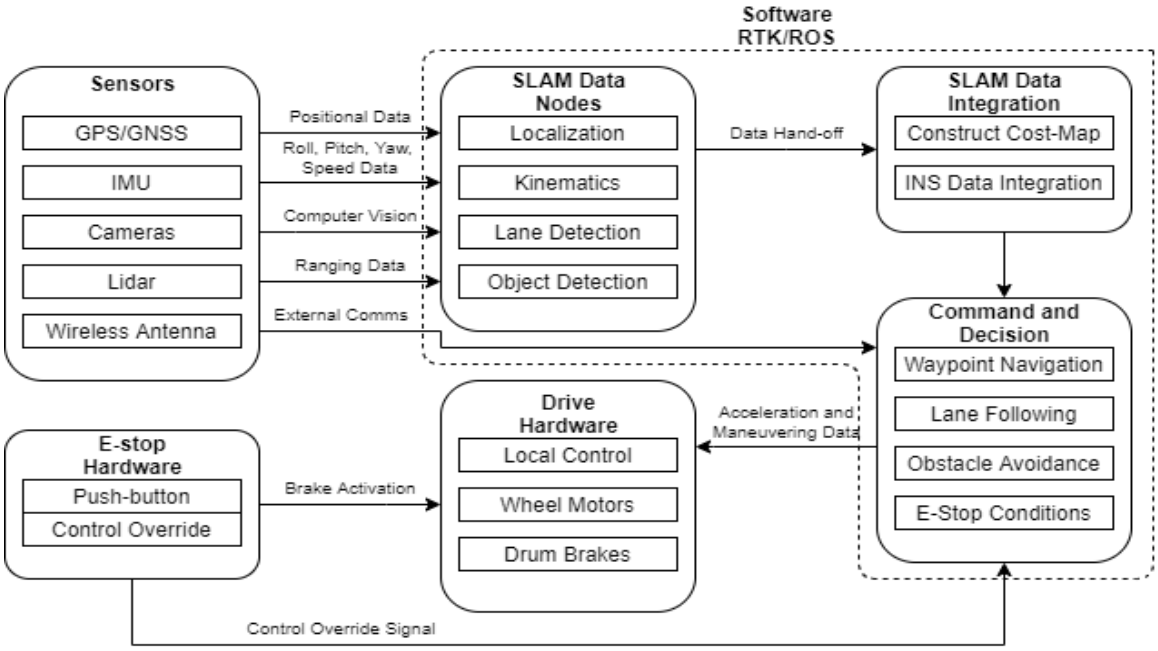


Figure 1. Command & Decision Logic Flow (Abstraction)

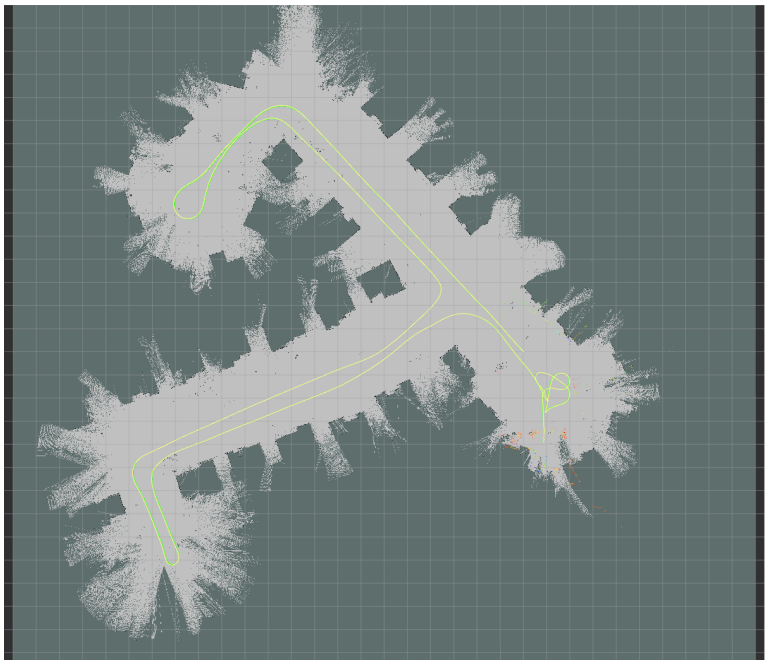


Figure 2. OpenSlam Gmapping - Cost-map Example

A local override controller for the emergency stop system will need to monitor external communications for the wireless E-stop signal. If the wireless or mechanical E-stop is activated, a “Control Override” signal should be produced to stall the C&D system while brakes (mechanical or electrical, depending on the vehicle) are activated. This will help ensure the vehicle will be safe to approach while it has power. Due to the differences in mechanical configurations between Little Blue and Monarch I, the emergency stop systems will need to be tailored to match their unique setups. See Appendix D for the E-stop activation circuit installed on Monarch I.

2.6.1. Velocity-to-PACMod Bridge

A major component of the Monarch I’s by-wire kit is the PACMod. This component provides a throttle and brake-by-wire controller module and a steering and shifting-by-wire controller module. Figure 3 shows how the PACMod communicates with the Spectra computer and Monarch I. Note that a Kvaser CAN to USB adapter is necessary to establish connection between the CAN and Spectra.

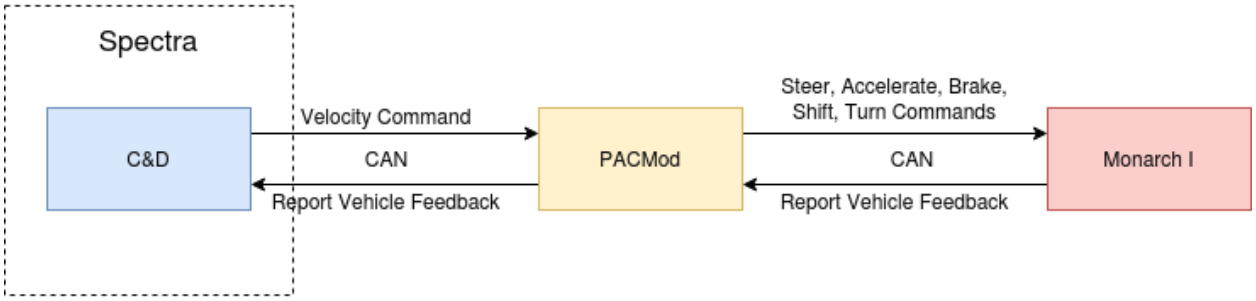


Figure 3. Autonomous Stuff Drive-by-Wire Kit Setup

As shown in Figure 3, the velocity commands from the C&D system are sent to the PACMod. The move_base node publishes velocity commands as a rostopic called “cmd_vel” of rostopic type geometry_msgs/Twist. This type of rostopic consists of a 3D vector for linear velocity and a 3D vector for angular velocity. Velocity commands of this type cannot be directly inputted to the PACMod. Autonomous Stuff provides a ROS PACMod node which allows ROS nodes to send commands to the physical PACMod by subscribing to topics for acceleration commands, brake commands, shift commands, steer commands, and turn signal commands. The PACMod node also subscribes to an enable topic which gives it permission to control the

vehicle. Additionally, the PACMod node subscribes to a CAN topic which it parses to provide feedback from the vehicle. (See Appendix G).

We design a Velocity-to-PACMod ROS node called “control_pacmod_node” to transform the command velocity we derive from our C&D system to appropriate rostopics that the PACMod node can subscribe to. Figure 4 depicts the communications between control_pacmod_node and the PACMod node. It can be seen that the pacmod node is subscribed to the as_rx rostopics that are derived and published by control_pacmod_node. Node control_pacmod_node is subscribed to the parsed_tx and as_tx topics that are published by the pacmod node as well as the velocity command, cmd_vel, which was manually derived for this test. Note that this graph was generated by ROS’ rqt_graph package which generates a graph of nodes and topics that are currently active.

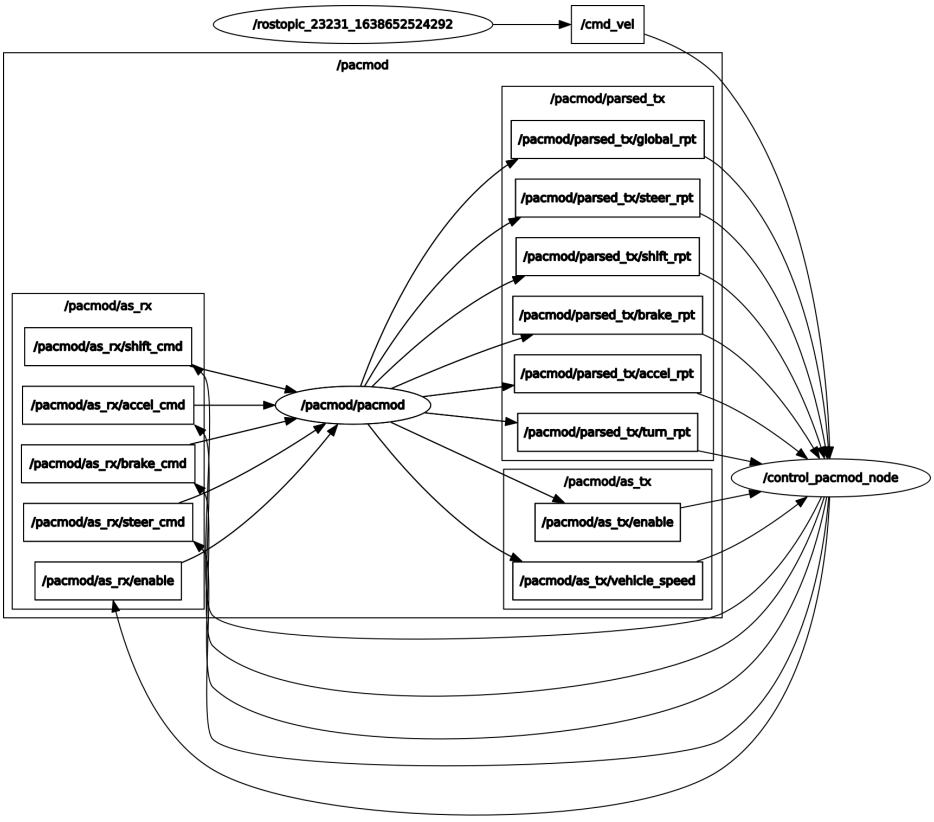


Figure 4. Communication between the pacmod node and control_pacmod_node

Node control_pacmod_node enables the PACMod to control the vehicle and derives a value for the gas pedal position, brake pedal position, angle of the steering wheel, velocity limit

of rotation of wheel, and gear. The angular position of the steering wheel is computed as the following:

$$steering\ wheel\ angular\ pos. = \ cmd\_vel_{angular(z)}.$$

The angular velocity limit of the steer is set to 1.0. Figure 5 depicts the transmission and throttle control in our design. The transmission and throttle control ensures that the transmission must shift to park before changing from forward to reverse and vice versa. It also ensures that the vehicle is completely stopped before shifting to park. As shown in the figure, the x component of the linear command velocity determines whether the vehicle is accelerating or not and its magnitude determines whether it is driving forward or backward.

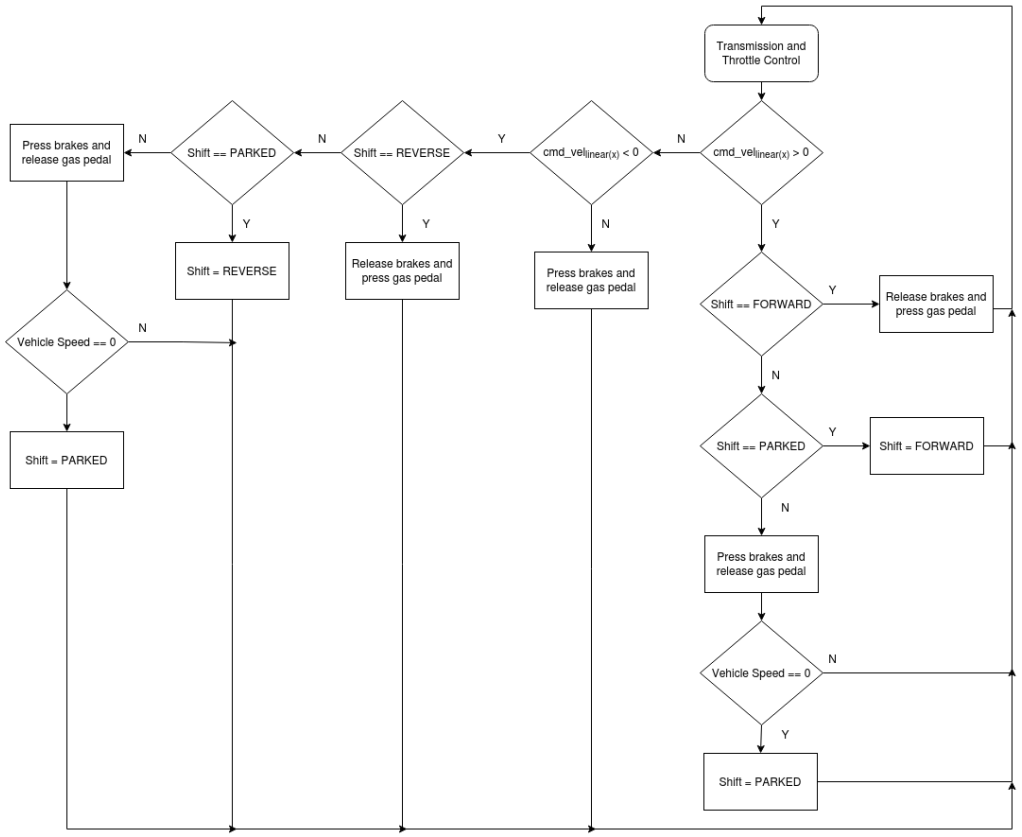


Figure 5. control_pacmod_node transmission and throttle control

The design currently sets the position value of the gas pedal to 0.4 (range of position is 0.0 - 1.0) when the vehicle is intended to be accelerating. This was determined to be a safe amount of press on the gas for the time being. The position value of the brake pedal is set to 1.0

when the vehicle is intended to brake. A future design could utilize the vehicle feedback published from the pacmod node to derive a more dynamic computation of the positioning of the pedals. Feedback of the current velocity and intended magnitude of velocity can be used to derive a gradual brake and acceleration. Testing of the Velocity-to-PACMod bridge is provided in Section 3.3.

2.7. Little Blue v1.0 Power Distribution

Currently, Little Blue v1.0 power supply consists of a 24V Battery with a 24-12V and 24-5V DC Converter to distribute power to the CPU/GPU hardware, sensors, emergency stop, and electric drive system as illustrated in Figure 6. The current CPU/GPU hardware, sensor setup, the E-stop controller, and the electric drive controller require either 5V or 12V supplies (except for the ZED-F9P GPS unit, which requires 3.6V).

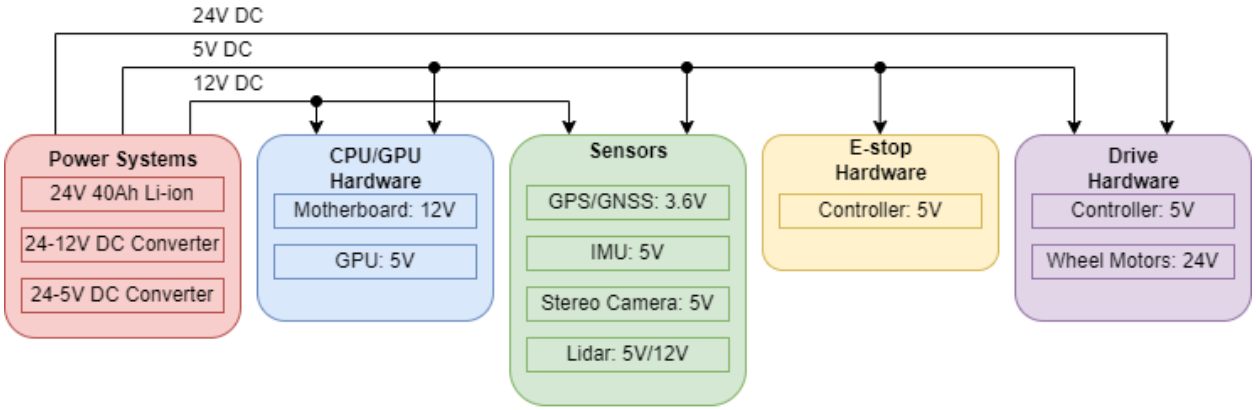


Figure 6. Little Blue v1.0 Power Distribution

2.8. Little Blue v2.0 Power Distribution

The Little Blue v2.0 will run in a similar manner which can be seen in Figure 7. Little Blue v2.0 Kunray wheel local drive controllers will be supplied by a 24V source. The Arduino Microcontroller will then provide the speed control characteristics to the motor drive controller. The emergency stop button will be implemented in between the Arduino Uno microcontroller and the motor driver to gradually reduce the power to the wheels to zero. The wheels on both Little Blue v1.0 and v2.0 will have a dedicated 24V power supply managed by a motor driver. Until additional sensors are implemented into Little Blue v2.0, the system will be operating under a user controlled device.

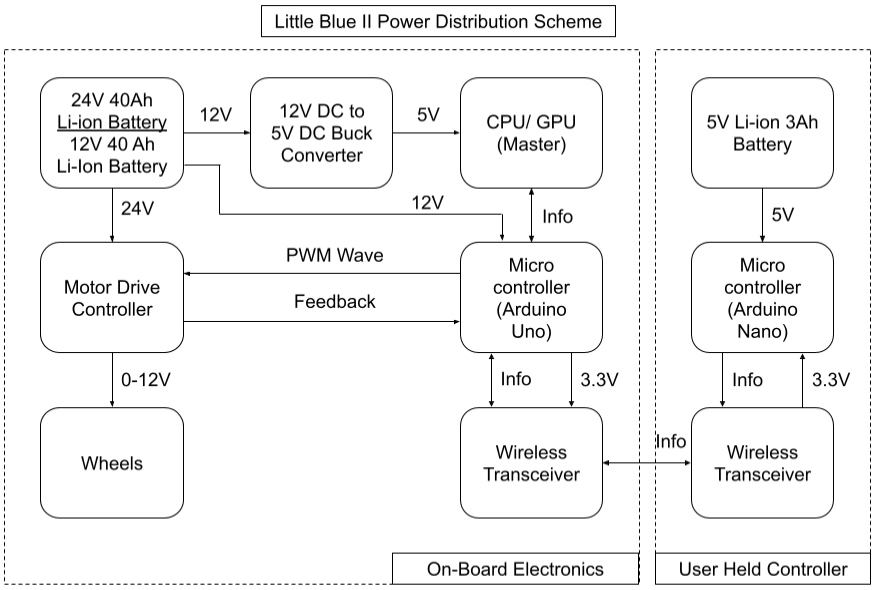


Figure 7. Little Blue v2.0 Power Distribution

2.9. Monarch I Power Distribution

Monarch I's Power Distribution System is managed by a multiplexed vehicle electrical center (mVEC) unit that distributes 12V into eight 12V, 20A fuse-protected outputs and four unprotected 12V lines. The mVEC has a built-in 12V to 24V DC to DC Converter to provide power to the onboard Spectra computer (See Appendix H). Currently, six of the eight protected 12 VDC outputs are routed and reserved for the onboard display, the Velodyne VLP 16 Lidar, the Triton Camera, the Kar-Tech Wireless E-stop, the Nvidia Jetson TX2, and an auxiliary line (future use). The mVEC receives its input from a 12V accessory battery that is indirectly charged through a DC to DC converter that steps down the voltage of the primary battery bank from 48V to 12V. The 48V battery bank is composed of eight golf-cart 6V AGM batteries that are charged by the Delta-Q QuiQ charger - which connects to any wall outlet (120V 60Hz AC).

3. Testing, Results, and Analysis

3.1. Monarch I Battery Troubleshooting and Testing

Upon returning from the 2021 IGVC competition, Monarch I's vehicle state of charge was depleted and no longer accepted charging from the 120 VAC wall charger. Our team began troubleshooting and testing the main vehicle battery, 8 - 6 VDC deep cycle AGM batteries in

series (48 VDC combined) and came up with open circuit voltage readings of approx. 45.5 VDC. After reading the manufacturer's troubleshooting manual and utilizing the flowchart shown in Appendix E Figure E.1 for the Delta-Q QuiQ charger, we found that the vehicle would accept charging with any pack reading over 36 VDC. These findings led us to look at the onboard 12 VDC battery under the dashboard which supplies power to all of the vehicle safety equipment and control relays from the Battery Management Controller (BMC). We took open circuit voltage readings and found the battery reading approx. 3.6 VDC. After removing and manually charging the battery up to around 10.5 VDC we returned the battery and its electrical connections to attempt charging. The vehicle's onboard charger detected power from 120 VAC and began charging. As a result, the group has begun testing the vehicle to locate and isolate the cause of battery depletion and our theory is a parasitic draw is occurring while the vehicle is shutdown leading to battery drain. The group is continuing its efforts of testing circuits and equipment powered from this battery to come up with a solution for our current theory. Due to the uncertainty of the vehicle main and accessory battery parasitic draw when correct startup and shutdown steps are not taken, we have designed a procedural flowchart shown in Appendix Figure E.2 to ensure we are not faced with consistent depletion and degradation of the vehicles 12 VDC and main 48 VDC batteries.

3.2. ZED 2 Stereo Camera Familiarization and Lane Detection Algorithm

Stereolab's ZED 2 stereo camera includes a robust sensor set and a software development kit (SDK) that supports libraries in C++, Python, and C# to facilitate computer vision on several platforms. As shown in Figure 8, the ZED's Depth Viewer sample program exemplifies the camera's ability to build a 3D point cloud from an input video stream of images (sample image shown in Figure 9). Theoretically, we could derive the location data of a lane within a region of the 3D point cloud by analyzing the ZED's input camera stream with a lane detection algorithm. Therefore, our goal was to integrate the ZED 2 camera with a lane detection algorithm that utilizes OpenCV's (open source computer vision library) Hough Line Detection method to define lane data with a set of cartesian coordinate points that correspond with specific pixels in the ZED's video stream. With these points the ZED's SDK functions can define a lane-region within the point cloud and relay the spatial information to localization ROS nodes.

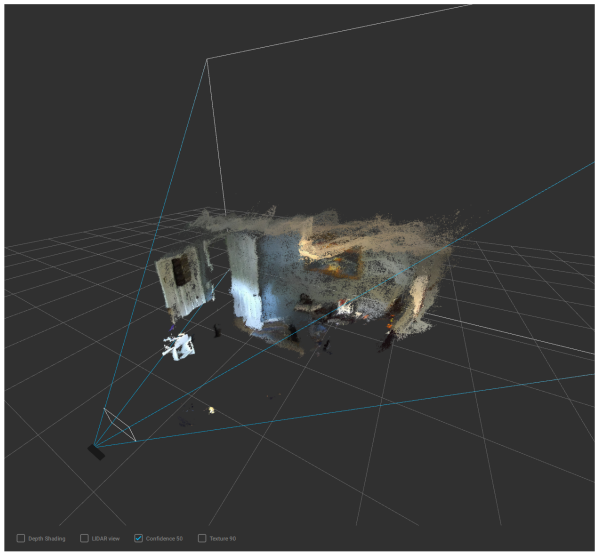


Figure 8. ZED Depth Viewer Point Cloud

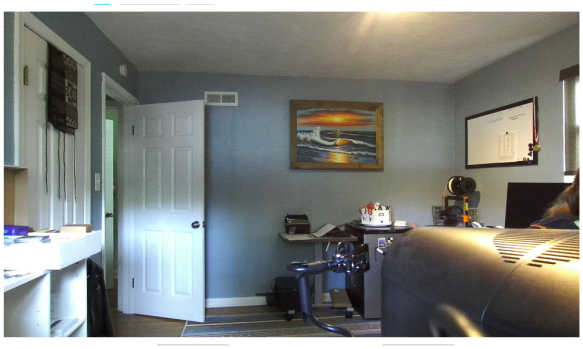


Figure 9. ZED 2 Sample Image

Since C# is primarily for the Windows .NET framework and C++ typically outperforms Python, we decided to employ C++ in our software designs for this device because the hough line detection method can be computationally heavy. This is because the Hough Line Detection function evaluates each image pixel in 180 degree sweeps (at an assigned degree interval) to search for pixels that have identical hues to the queried pixel to determine if a line exists. This is accomplished by recording the number of identical pixels into a mapped accumulator with a number of key value elements equal to the range of possible lines at the specified degree interval. These elements are typically categorized in their polar line equation format,

$$\rho = x \cos\theta + y \sin\theta$$

This avoids divide-by-zero errors in the vertical slope calculations of cartesian line data. Once each pixel has been evaluated, the function loops through the populated accumulator and evaluates the votes for each possible line to a threshold value (arbitrarily assigned) and validates line data with votes above the threshold.

An overview of the ten-step algorithm we developed is illustrated in Figure 10. Our C++ program continually samples the live video feed from the ZED 2 stereo camera and applies the algorithm in a while-loop that is only terminated when the keyboard ‘q’ is pressed. While the

program is running the user may choose to display steps 2-6, 10, and the predefined masking image for debugging purposes without interrupting the live stream but this isn't necessary for the final application. A visual representation of each step defined in Figure 10 (except for steps 7-9) are provided as they appeared during debugging in Figure 11 below.

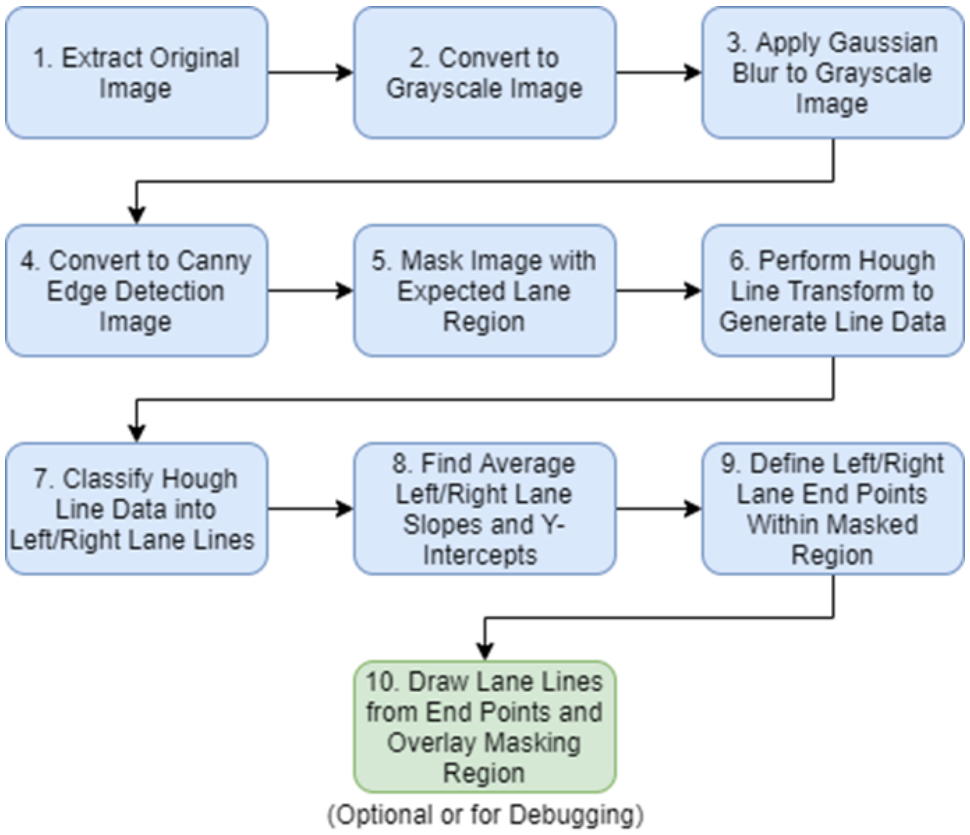


Figure 10. Lane Detection Algorithm Steps

The images sampled from the ZED 2 stereo camera's video feed must first be converted into grayscale to collapse the pixel hue ranges into single channel values, 0-255, for compatibility with the OpenCV functions ("GaussianBlur()", "Canny()", and "HoughLinesP()"). The "GaussianBlur()" function is used on the image to reduce noise before applying the Canny Edge Detection function. The "Canny()" function is applied to create a black image (0 pixel value) with only the contour lines illuminated in white (255 pixel value). A trapezoidal masking image is ANDed with the Canny Edge Detection image to define a perspective-based region the camera is expected to find a lane. ANDing the pixel values of the mask and the Canny image

eliminates any edges outside of this region to reduce noise that would interfere with the lane line calculations. A Hough Line Transform function, "HoughLinesP()," is used to probabilistically derive the cartesian end-point data, (x_1, y_1, x_2, y_2), of each detected line respective to the Canny image. Note the "HoughLinesP()" function performs these calculations more efficiently than the standard OpenCV "HoughLines()" function which derives line data in polar coordinates, (θ, r_θ) [6].

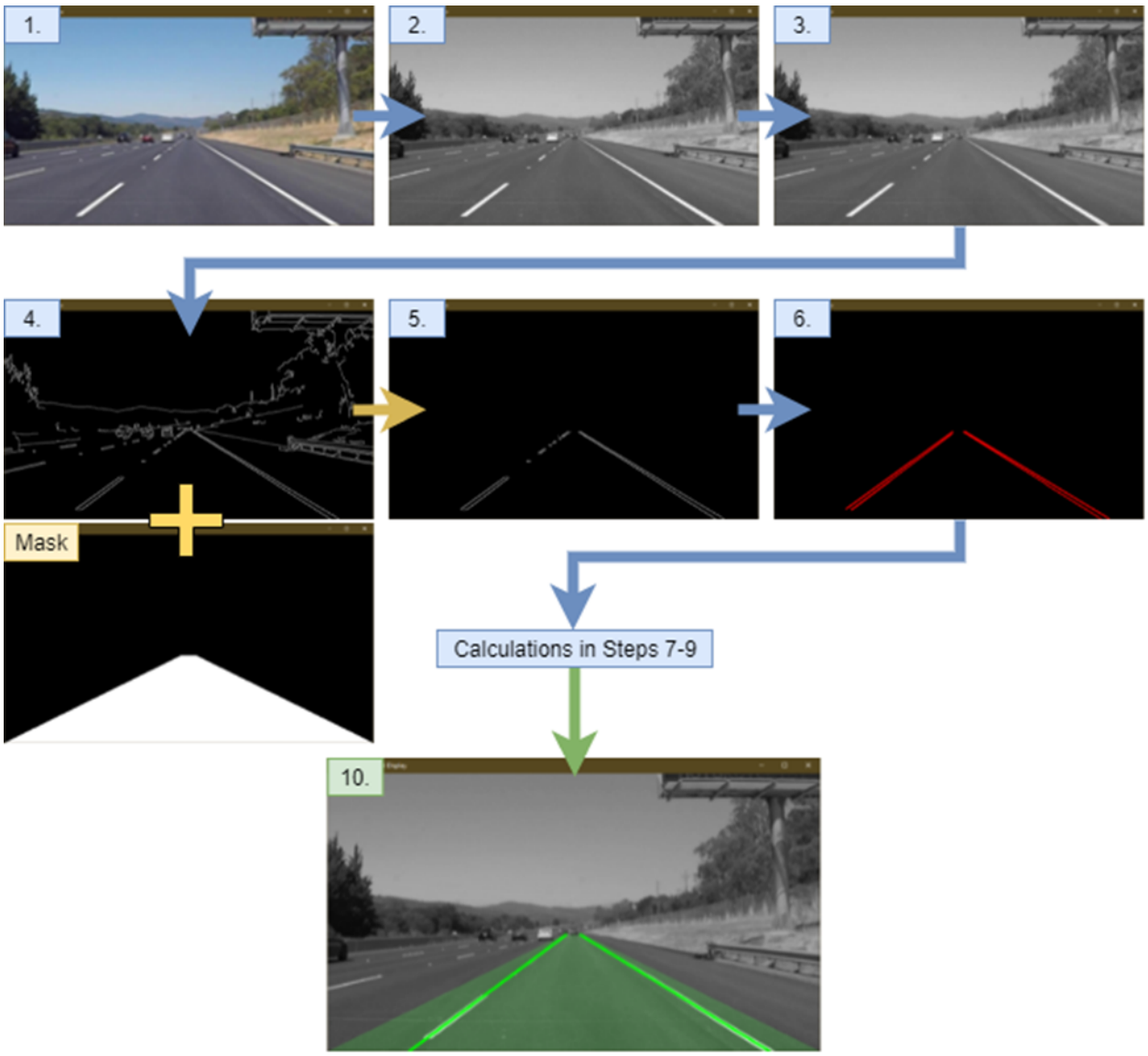


Figure 11. Lane Detection Algorithm Visual Representation

The algorithm's calculations in Steps 7-9 extrapolate four critical points from the Hough Line Transform function's line data sets. Two of these points correspond to the endpoints of the

left lane and the other two correspond with the endpoints of the right lane both bound by the top and bottom edges of the masking region. These are derived by, first, classifying the detected lines as right lane lines and left line lanes from their slopes. Positive slopes are identified as potentially right lane lines and negative slopes as left lane lines. Note these slopes are relative to the top horizontal as the x-axis and the left vertical as the y-axis with positive orientations to the right and down, respectively (See Figure F.1 in Appendix F). Once the right and left lane lines are classified, an average slope and y-intercept value is derived from the two data sets. These values are used to calculate the critical points of each lane line.

At this point, two lines can be drawn over the original image (or grayscale) between the endpoints of each line as illustrated in Figure 11, Step 10 – which also includes a transparent green overlay of the masking region. However, this is just to visually represent the extracted lane data for debugging purposes. Further testing and familiarization with the ZED 2 camera are required to develop a method that utilizes this data in the autonomous actions.

3.3. Velocity-to-PACMod Bridge Testing

We test the velocity-to-PACMod bridge, control_pacmod_node, by running the node with the pacmod node and manually publishing the command velocity (cmd_vel), feedback status of the vehicle's current speed in m/s (as_tx/vehicle_speed), and feedback status of the vehicle's current gear (parsed_tx/shift_rpt) at a rate of 1Hz. Manual publishing of rostopics are achieved by using the "rostopic pub" command in the terminal. To publish cmd_vel we use the following template: "rostopic pub cmd_vel geometry_msgs/Twist -r 1 '[2.0, 0.0, 0.0]' '[0.0, 0.0, 0.0]'" where 2.0 is the x component of the linear velocity. To publish parsed_tx/shift_rpt, we use the following template: "rostopic pub -l /pacmod/parsed_tx/shift_rpt pacmod_msgs/SystemRptInt -r 1 "{header: auto, output: 1}"" where the output parameter gives the status of the shift. Note that PARKED = 0, REVERSE = 1, and FORWARD = 3. The template to publish as_tx/vehicle_speed is "rostopic pub -l /pacmod/as_tx/vehicle_speed std_msgs/Float64 -r 1 "{data: 1.0}"" where data represents the speed in m/s. We use "rostopic echo" to view control_pacmod_node's published topics to verify that the node behaves as expected. The acceleration command (accel_cmd), the brake command (brake_cmd), the shift command (shift_cmd), and steer command (steer_cmd) are the published topics we observe. Recall from the design approach that accel_cmd will be 0.4

when acceleration is occurring and brake_cmd is 1.0 when the brakes are pressed. The following test cases were completely isolated from the vehicle.

Test Case 1: Shift from Reverse Gear to Forward

NICE!

1) $cmd\_vel_{linear(x)}$=+2.0, parsed_tx/shift_rpt.output = 1, as_tx/vehicle_speed.data = 1.0

   Outputs: accel_cmd = 0.0, shift_cmd = 0, brake_cmd = 1.0

2) $cmd\_vel_{linear(x)}$=+2.0, parsed_tx/shift_rpt.output = 0, as_tx/vehicle_speed.data = 0.0

   Outputs: accel_cmd = 0.0, shift_cmd = 3, brake_cmd = 0.0

3) $cmd\_vel_{linear(x)}$=+2.0, parsed_tx/shift_rpt.output = 3, as_tx/vehicle_speed.data = 0.0

   Outputs: accel_cmd = 0.4, shift_cmd = 3, brake_cmd = 0.0

Test Case 2: Shift from Forward Gear to Reverse

1) $cmd\_vel_{linear(x)}$=-2.0, parsed_tx/shift_rpt.output = 3, as_tx/vehicle_speed.data = 1.0

   Outputs: accel_cmd = 0.0, shift_cmd = 0, brake_cmd = 1.0

2) $cmd\_vel_{linear(x)}$=-2.0, parsed_tx/shift_rpt.output = 0, as_tx/vehicle_speed.data = 0.0

   Outputs: accel_cmd = 0.0, shift_cmd = 1, brake_cmd = 0.0

3) $cmd\_vel_{linear(x)}$=-2.0, parsed_tx/shift_rpt.output = 1, as_tx/vehicle_speed.data = 0.0

   Outputs: accel_cmd = 0.4, shift_cmd = 1, brake_cmd = 0.0

Test Case 3: Brake

1. $cmd\_vel_{linear(x)}$=0.0

   Outputs: accel_cmd = 0.0, brake_cmd = 1.0

Test Case 4: Command Steering

1. $cmd\_vel_{angular(z)}$=-1.8

   Outputs: steer_cmd.angular_position = -1.8, steer_cmd.angular_velocity_limit

All test cases show correct behavior for control_pacmod_node. Test cases 1 and 2 both show that the vehicle first comes to a stop and then shifts to park before shifting gears and then accelerates. Test case 3 shows the proper operation of braking and test case 4 shows how the z component of the angular command velocity is used to control the steering wheel. The angular velocity limit is set to 1.0 as mentioned in the design section.

3.4. Monarch 1 E-Stop Configuration and Testing

The original design for the E-stop configuration worked properly if only the primary transmitter was turned on and used. After getting the receiver and transmitter diagram and wiring configuration from the manufacturer, there was proper power put through the primary transmitter which would always cause it to work, this can be seen in Appendix E.7 and in the different tests in Appendix E.5. When trying to use the secondary transmitter in the original design, since there wasn't a proper output wired into the circuit, it would cause the circuit to malfunction, sputter between on and off, then keep the system on. However, when trying to use just one singular transmitter, the system works properly and will shut off once it's pressed. Another problem that occurred was that if the primary transmitter's "line of sight" to the receiver was blocked by the secondary transmitter, the primary transmitter did not work at all.

After reworking the circuit dozens of different ways, the manufacturer suggested adding a 1k Ohm resistor in the circuit to eliminate any error codes for unused outputs. After trying this suggestion dozens of ways, the same results arose. The design shown in Appendix E.8 is the most recent circuit setup that was made with the current components. The best way to fix the problem would be to change the relay. The relay in this system only allows the system to swap between two outcomes, which would only allow for one of the transmitters to function properly. The troubleshooting analysis can be seen in Appendix E.6, which is the same as in the previous design due to the same relay being used. As mentioned, a possible solution would be to get a relay that has the design like the third or fourth relay in Appendix E.4. By using one of these two relays, it will allow for each transmitter to be connected to the system, that way whenever both are on and one is pressed, the system will turn off and stay off.

3.5. Alternative Designs

Currently our primary focus is on the hardware and software testing and implementation of the Monarch I and Little Blue v2.0, as we are continuing to collaborate with other ODU

engineering groups and departments our focus and increased teamwork capabilities has led us to look into not only improving our current robot designs, but coming up with an innovative design approach such as the Little Blue v2.0 which is currently in its research and prototype stages, where along with the mechanical engineering group we are testing suspension and drive improvements as well as increasing the efficiency of onboard power consumption from the motors and sensors. Our goal for future competitions will be to qualify this design for competition and continue to look for ways to improve and come up with alternative solutions.

For more precise positional accuracy testing, a local base station could prove beneficial. There is a base station in range of the competition location but not in range for local testing. The closest base station to our location is in North Carolina. As a result, the team ordered three additional components to create an artificial base station: an RTK Surveyor, GPS Antenna Ground Plates, and a GNSS Multi-Band L1/L2 Surveying Antenna. Building a local ground station from the ordered equipment would provide the team with more accurate testing and offer more reliable data versus the use of the station located in North Carolina due to the difference in distances.

3.6. Little Blue II DC Motor Simulation

To properly define a system for the DC motors present on LBII, a model was used to show the interactions between a battery source, pulse width modulated (PWM) control wave, and motor drive controller. The basic cascade control structure of the simulated model can be seen in Figure 12. For simplification, the model contains a PWM wave, an error amplifier, correction factor, negative feedback loop, and final output. In Figure 13 it can be seen that there is a subsystem that makes use of the same elements. Let it be noted that the top figure in Figure 13 is the subsystem present in the bottom figure of Figure 13. The three separate feedback loops allows for a high level of correction that may occur with any changes in speed or load torque (ie. getting stuck, going uphill, or adding weight while running).

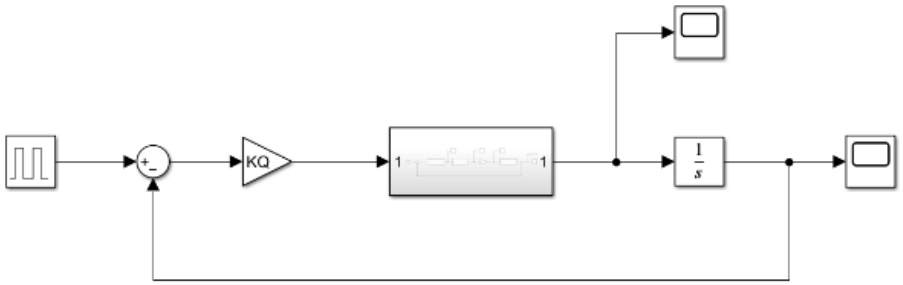


Figure 12: DC Motor Simulated Model: Position Control Belt

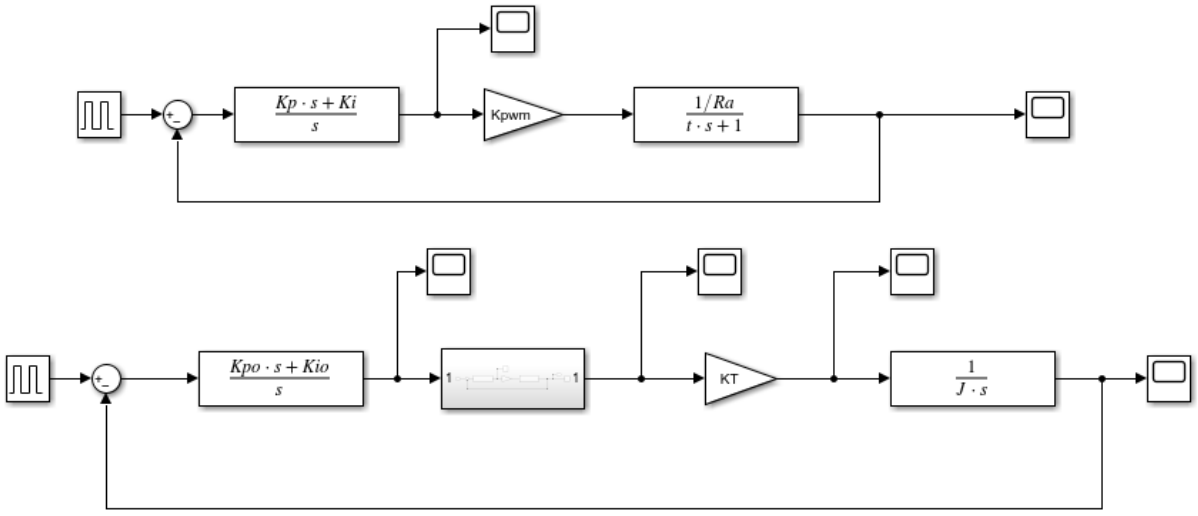


Figure 13: Speed Control Subsystem Belt (Top), Torque Control Subsystem Belt (Bottom)

The basic premise follows. The source is provided to the motor drive controller. The PWM control wave is then introduced to the source wave, thus giving it a high frequency PWM wave. This allows for the alteration of the average voltage provided to the motor. During each change in current due to the PWM wave, a set of error amplifiers are implemented to reveal the current spikes on a more noticeable scale. A correction factor follows the error amplifier which attenuates the sudden spikes present at periods of change. Prior to transient reduction, the current spikes can be clearly seen as a hazardous issue in the left image of Figure 14. Figure 14 right is the cleaned up voltage signal output from the motor drive controller.

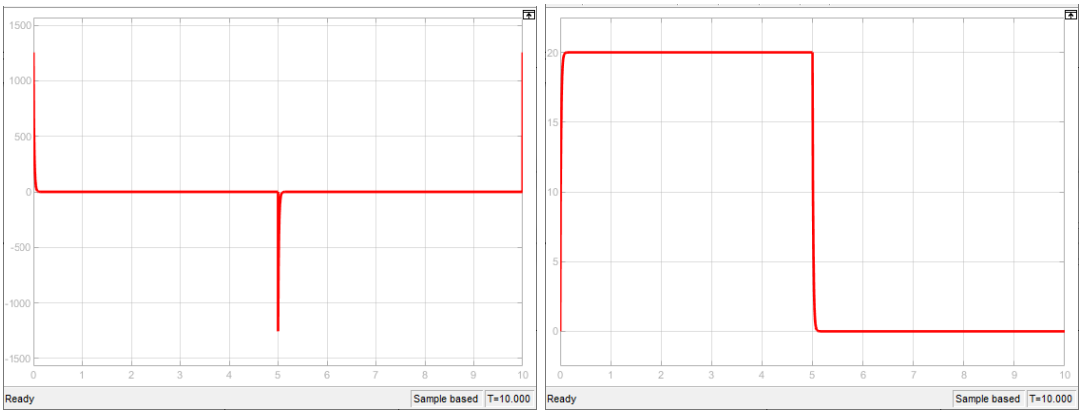


Figure 14: Transient Response Prior to Attenuation (Left),
Voltage Signal After Transient Attenuation (Right)

4. Project Management

4.1. Team Organization

The team held weekly meetings on Mondays at 5:00 PM via Zoom and Thursdays at 3:00 PM with Dr. Belfore to discuss progress, plans, and deadlines. A shared Google Drive folder was utilized for team, software, and hardware documentation. The team also regularly met once a week to make modifications and improvements to the Monarch I.

4.2. Tasks and Responsibilities

Renard Dichoso - Verified the designs and documentation for Little Blue v1.0, Little Blue v2.0, and Monarch I, which was tested and calibrated, to be compliant with the Statement of Work and recent IGVC rules and created and updated all power distribution, one-line diagrams, and IGVC rules, team and, design information on AutoCAD Drawings and PDF's, which will act as an efficient, accessible living document for past, current, and future students.

Justin Heisterkamp - Routed Monarch I's power distribution and helped troubleshoot the battery bank. Performed testing with the ZED 2 API in C++ and Python with the ZED 2 SDK. Developed a crude lane following algorithm using the ZED 2 stereo camera and OpenCV.

Kyle Hernandez - Power systems control and design are my primary applications. Working to ensure sufficient energy is distributed to mechanical and electrical devices without unnecessary losses. Simulation via Simulink and PSpice with a focus on overall operation and

efficiency when related to DC motors and drives. Secondary focus on troubleshooting electrical issues with newly integrated devices.

Trevor Novisk - Analyzed different components to help design the mechanical and electrical systems operate proficiently. Tested different simulations via PSpice and Simulink that helped design a proper solution to the E-stop features for Monarch 1. Troubleshooted multiple systems to make the wireless E-stop work properly with both receivers.

Warren Shields - Developed ROS package that bridges communication between velocity commands and the PACMod to provide C&D navigation to the Monarch I. Setup an encrypted installation of Ubuntu 16.04 with ROS, RTK, and its necessary drivers and packages and wrote the disk to an image for more efficient distribution among approved members.

Christophe Vautier - Worked on hardware and sensor equipment management and analysis. Tested, designed and optimized vehicle local and remote wireless e-stop, onboard vehicle power distribution hardware and sensor circuitry, hardware and sensor placement and optimization, tested and troubleshooted vehicle battery and charge systems and design. Designed vehicle startup and shutdown procedure, guidelines and schematics.

Jonathan Zeigler -My primary focus was on gathering information on the IMU configuration. I also documented the components on LB-1's wireless controller.

4.1. Lifelong Learning Discussion

Renard Dichoso - Before ODU, I started working as an electrician at the shipyard right after high school, but during those four years, I became more interested in the distribution and design of electrical systems. After the shipyard, I started working for a private engineering company that specializes in plumbing, mechanical, and electrical design solutions for renovations, repairs, and new construction and started transferring to ODU from TCC for Electrical Engineering at the same time. I started as a CAD Technician, where they promoted me to Electrical Design Engineer after my first two years in the company. With the experience from my jobs and courses that have helped me including Power System Design Analysis and all my Engineering Management classes, it has helped my group and I with the particulars of smaller scale electrical design and the employment of various engineering management, engineering

writing, and CAD Drawings. The collaboration in the group and the project itself has helped grow my continuous interest in both my job and Electrical Engineering in general.

Justin Heisterkamp - Naturally, this project and our faculty advisor has helped me (and the team) strengthen several engineering skill-sets, such as programming and troubleshooting. However, this project has taught me a lot about project management and other soft skill-sets, such as communication within a team. It has also revealed areas of improvement to me that would be prudent for me to address in my future career. Currently, I have prospects towards a career in semiconductor manufacturing as a process engineer, which is much different from this project. Despite that I'm certain the lessons I've learned will still be useful in my professional life. Despite the contrast, I plan to continue practicing development as a hobby to help keep a sharp mind and possibly lead to other career opportunities. I decided to do a dual major to cultivate a broad horizon of knowledge in engineering and I plan to continue to have a mind-set to pursue knowledge, just for its own sake and to pass that knowledge on.

Kyle Hernandez - Prior to Old Dominion University I attended Northern Virginia Community College. After eventually finishing my associates degree in electrical engineering, I transferred to ODU where I took on many courses pertaining to power electronics. During my time at ODU I interned with the Vandenberg Space Force Base, where I was introduced to the complexities of real life power engineering situations. Following graduation, I will study for the FE exam in hopes of pursuing the professional engineering certification. Due to the nature of the field I have chosen, I will pursue a masters either in electrical engineering or business administration.

Trevor Novisk - Before attending ODU, I attended Tidewater Community College and completed an associates degree in engineering. This kickstarted an interest in electrical engineering, which allowed me to focus more on electronic systems and the hardware aspect. These skills will help aid the team's designs and goals throughout the scope of the projects regarding the ground vehicles. The knowledge I gained primarily from ODU and some through self teachings, have helped myself and the team to keep up with tasks, as well as create designs that will benefit the initial design and the goal for the team. This project alone has been the most complex and actually a real project, which has helped in trying harder and wanting to learn more as an engineer.

Warren Shields - Through my tasking, I have strengthened my expertise in C++ and overall programming/debugging skills and adaptability by learning the ROS API. After graduation, I will be converting to full-time employment with my current internship as a Software Engineer and these skills will be directly transferable. I will be graduating with a concentration in Data Analytics Engineering and also have plans for earning a M.S. in Data Analytics Engineering or a related field after working for some time. The acquired knowledge from this project benefits me in that.

Christophe Vautier - As an EE major at ODU with an associates from TCC, currently working in marine engineering with military service as a shipboard electrician. I have acquired knowledge in power systems and renewable energy to aid in my pursuit of working in power systems related fields. This project has allowed me to gain knowledge in various aspects of electrical engineering, such as DC power, hardware and sensor electronic systems. The teamwork and collaboration during my time at ODU has greatly improved my skills in communication and critical thinking. I plan to focus my knowledge and experience gained on AC and DC power systems to work with a company looking for testing and design engineers.

Jonathan Zeigler - Before attending ODU, I attended TCC with the intent of going into the arts. I started seeking an Engineering Associates after realizing I desired a different kind of challenge. My background has been varied with technical jobs, and creative projects while pursuing my Bachelor's degree in Electrical Engineering and my minor in Computer Engineering. I will continue to pursue technical and creative projects outside of school after joining the workforce.

4.2. Deliverables and Timeline

Task	Start	End
E-Stop	Aug-28	Dec-11
Lane detection	Aug-28	Dec-10
Disk Encryption	Aug-28	Nov-20
Navigation/Costmap Integration	Oct-20	Dec-10
Midterm Report	Oct-7	Oct-14

Final Report	Nov-23	Dec-6
Final Presentation	Dec-1	Dec-9

Table 1. Milestone Timeline

5. Engineering Standards

5.1. ROS Enhancement Proposals (REPs)

REPs are design documents published by ROS developers that provide information to the ROS community, or describe new ROS features, processes, or the environment [7]. There are three types of REPS: Standard Tracks, Informational, and Process. Standard Track REPs describe new features. Informational REPS provide general guidelines as informational advice. Process REPs describe procedures, guidelines, and changes in the decision making process, tools, or environment used in ROS development. Though these documents are not officially recognized engineering standards, they do provide standard guidance for the use of ROS nodes. Moreover, familiarity with REPs will aid in the development in conjunction with industry standards and help the team determine the applicability of them.

5.2. 1873-2015 - IEEE Standard for Robot Map Data Representation for Navigation

Robot navigation, whether indoor or outdoor, critically relies on two-dimensional costmaps, topographical maps, or voxel maps. This document defines an IEEE standard for formatting map data that facilitates common robot navigation and interoperability methods [8]. However, after review of this document with the ROS REP information in mind, it appears this standard is not ideal for runtime costmap construction. Rather it is normally used to help disseminate pre-existing map data to a network of robots. This standard may only be useful for providing our system's pose and speed with the Common Operating Picture (COP) software specified under the Interoperability Profile Challenge requirements.

5.3. 1609.12-2019 - IEEE Standard for Wireless Access in Vehicular Environments (WAVE) - Identifiers

Our project consists of designing and developing a fully autonomous robot that requires wireless communication to obtain real-time data and feedback. The 1609.12 IEEE engineering standard establishes a protocol to standardize data and information transmission between different systems or devices. Our autonomous robot will need to receive and provide feedback to

a host system or data hub during the IOP challenge. Provider Service Identifiers (PSIDs) have two uses specified in IEEE Std 1609.3. First, a service provider identifies advertised service opportunities by the PSID values in WAVE Service Advertisement messages that it or another entity may transmit" [9]. The WAVE Short Message Protocol (WSMP) will deliver this message based on a set of values given by the sender [9]. This standard allows and authorizes the use of these PSID values and testing PSIDs for us to test and develop to improve on our robot performance and information processing.

5.4. 1872-2015 - IEEE Standard Ontologies for Robotics and Automation

This IEEE standard defines specifications for the representation of knowledge and vocabulary in the context of robots and artificial systems to pass unambiguous knowledge among "humans, robots, and other artificial systems" [10]. This project involves the communication between various sensors and hardware components that employ computer vision algorithms and several ROS nodes to build an artificial system. The datastreams, vernacular, and knowledge-base involved with this is extensive. A standard for communicating this ontological modality between the team members and the AVs machine learning capabilities and status is critical. Furthermore, it's also important to meet a standard format for communicating our methods to the public.

6. Broader Impacts

6.1. Ethical Implications

If artificial intelligence has to choose between two or more possible adverse outcomes, which one will it pick? Suppose the outcomes are hitting a pedestrian and saving the driver or avoiding the pedestrian and risking the driver. How will the AV determine a best outcome on moral grounds, and how should it determine worth? All humans have a right to safety, but the decision will lie on the group of engineers when programming and creating the AV. Current technology can only operate on preprogrammed instructions that may calculate a casualty-cost decision structure based on the number of pedestrians detected versus the number of passengers. Other factors may not be as easily determined, such as the age of those involved or a pedestrian's reaction. The IEEE Code of Ethics states that engineers shall hold paramount the safety, health and welfare of the public [11]. Ethical scenarios raise concerns, but the overall safety of this technology may outweigh its risks. Projections report that removing human error with AVs may

decrease automobile accidents by 90% [12]. Despite valid ethical concerns, the likelihood of such an event occurring in an AV compared to a human-operated vehicle may reduce these.

6.2. Knowledge of Contemporary Issues

Just as increased automation has already replaced many jobs, professional drivers may now be at risk. Since heavy truck and tractor-trailer driver salaries totaled nearly $91.87 billion in 2019 [13], an increase in AVs has the potential to harm the economy short term. This impact could affect millions of workers in those industries, leaving them unemployed.

The effects of transportation on the environment present a climate change issue. According to the US Environment Protection Agency, the transportation sector accounts for about 14% of all greenhouse gas emissions in the United States [14]. A different study suggests that the increase of AVs throughout the ground vehicle transportation sector could increase vehicles' efficiency by up to 80% [15]. The efficiency of vehicles alone could reduce the US emission output significantly.

An additional topic worth exploring is the availability of AVs. Due to the technology required to achieve autonomy, AVs' cost is much greater than a standard vehicle. This adds an economic barrier for autonomous vehicles. Many researchers suggest that AVs will lower fuel costs, reduce travel time, and potentially reduce insurance costs which may be viewed as a socioeconomically restricted technology [15].

6.3. Global Impacts

As autonomous vehicle technology and research continue to grow globally, this technology's potential benefits can have major positive impacts if adopted and appropriately implemented in the transportation and environmental sectors. 95% of the world's population lives in areas exceeding the World Health Organization (WHO) guideline for healthy air [16]. This statistic illustrates the importance of innovation and autonomous vehicle research and development. If this technology can benefit the environment with the use of more environment-friendly technologies such as fully electric, demand and funding for this technology will increase and have a positive global impact.

6.4. Economic Impacts

In this digital era, integration of intelligent sensor systems in the automotive industry have become more necessary. Car manufacturers are equipping vehicles with systems that monitor and display aspects of vehicle status and safety hazards. We now find ourselves on the precipice between actively driven cars and AVs for civilian use. The growing interest in these machines and the dramatic increase of sensors used in vehicles has led to immense data collection that forecasts severe economic changes. The heaviest impacts will occur in the costs of freight, the insurance industry, passenger transportation, the auto industry, and the auto repair/maintenance industry. Despite most of these areas initially suffering from large economic losses, estimates suggest that AVs full adoption could save the U.S. up to $1.3 trillion per year and $5.56 trillion globally [17].

As the number of personal vehicle sales increases on an annual basis - 95 million in 2018 and a projected 110 million vehicles in 2022 globally [18] - the auto industry has a clear incentive to stay market competitive. However, as rideshare systems increase in popularity and the facilitators of these systems gain more interest in AVs, private vehicle ownership may decline as AVs gain more traction.

The counterpart to the auto industry, the repair and maintenance industry, may also take a big hit - presuming that AVs performance is adequate in decreasing traffic incidents [18]. Yet, the demand for routine maintenance would still be present. The reduction of incidents also implies that insurance companies would be under fiscal pressures from their clients for lower service premiums. Currently, companies like Uber and Google gather more driver information than incumbent insurance companies [19]. As the number of intelligent vehicles increases, the collected data may suggest that insurance companies' historic risk calculations are obsolete, especially considering an estimated 90% of car accidents are due to human error [19].

6.5. Environmental Impacts

It is no surprise that personally owned vehicles (POVs) have a significant impact on the environment. In fact, approximately 34% of carbon dioxide emissions in the United States originate from the transportation sector [20]. Car sharing programs can reduce greenhouse gas emissions by virtually replacing personal transportation [21]. However, these programs require more innovative solutions for the logistic issues regarding personal transportation convenience

and readiness. Long wait times due to latency from large distances between the person and the nearest available manually operated vehicle are problematic [21].

By creating a networked system of shared autonomous vehicles (SAVs), a person may use their cellphone to obtain a ride in much less time than to search or wait for a non-autonomous vehicle [21]. Advances in artificial intelligence and machine learning techniques present an opportunity for carsharing organizations to match demand by allowing the network to develop optimal routes and pick-up schedules more rapidly. These developments could benefit the environment by reducing habitat destruction as parking requirements would lessen with a decline in POVs. A 2013 study found "North American carsharing members reduced their driving by 27%" [19, p. 2]. Furthermore, automated acceleration and braking systems associated with AVs can reduce fuel consumption due to increased precision and performance [20]. Development of AVs shows the potential for decreasing pollution, traffic congestion, and space consumption.

6.6. Societal Impacts

When considering the impacts of AVs on our society, the most obvious of these impacts are often extreme examples. Rates of death among young drivers, increasing federal and state legislation, or even travel times are often remarked upon regarding AVs. One of the leading causes of death for young adults and children is traffic accidents, alongside high injuries and property damage [22]. Through vehicle autonomy, countless lives could be saved by letting computers manage situations where human ability fails. While it can protect, AVs also give an ability to enable. Now people with disabilities or other restrictions could become passengers of AVs. AVs can decrease travel time through reduced accident traffic and increased roadway efficiency [22]. A thought that comes to mind when discussing AVs is the legal aspect. When it comes to vehicles, regardless of autonomy or not, risk is always a present factor. A common dilemma of AVs is ethical decision-making. Following the article [23], there is a legal discussion that has not been tackled. Car manufacturers have to decide on priorities. Who is to be saved in a worst case scenario? Society will have to determine the ethical reasoning of harming the pedestrian by saving the driver or saving the pedestrian while risking the driver. There is no doubt that AVs provide extensive benefits to humanity by saving lives and increasing the quality of life.

7. Summary and Conclusion

AVs are necessary for technological advancement, and the work to progress the field's knowledge furthers the possibilities. Our work in preparation for the IGVC and its subsections will prepare the next group of engineers for the technical and ethical uncertainty in advancing AVs. We achieved that through extensive testing, implementation, and integration of sensor technologies, electrical controls, power systems, and software development. We have developed a lane detection algorithm that is integrated with our ZED 2 camera as an input for our C&D unit and have developed the necessary bridge between our C&D and the PACMod controller to the Monarch I via the control_pacmod_node. Future work should involve further testing of our developed software and in a simulated environment along with the integration of an object detection algorithm, the LiDAR, and building a costmap with the received sensor data.

References

[1] J. Lane and A. Kosinski, "The 28th Annual Intelligent Ground Vehicle Competition (IGVC) & Self-Drive," 8 September 2021. [Online]. Available: http://www.igvc.org/2022rules.pdf. [Accessed 17 September 2021].

[2] "Introduction to Robotic Technology Kernel (RTK)," U.S. Army. [PowerPoint slides]. Available: http://www.gl-systems-technology.net/uploads/3/4/5/7/34572805/introduction_to_rtk_-_may2018_-_dista.pdf. [Accessed 16 March 2021].

[3] "ROS," [Online]. Available: https://www.ros.org/. [Accessed 16 March 2021].

[4] "robot_localization wiki," [Online] Available: http://docs.ros.org/en/kinetic/api/robot_localization/html/index.html. [Accessed 1 December 2021]

[5] "move_base," [Online] Available: http://wiki.ros.org/move_base. [Accessed 1 December 2021]

[6] OpenCV. "Hough Line Transform." doxygen. https://docs.opencv.org/3.1.0/d9/db0/tutorial_hough_lines.html (accessed Nov 17, 2021).

[7] K. Conley. "REP Purpose and Guidelines," [Online]. Available: https://www.ros.org/reps/rep-0001.html#id4 [Accessed 21 April 2021]

[8] "IEEE Standard for Robot Map Data Representation for Navigation," *1873-2015 IEEE Standard for Robot Map Data Representation for Navigation*, vol., no., pp.1-54, 26 Oct. 2015, doi: 10.1109/IEEESTD.2015.7300355.

[9] "IEEE Standard for Wireless Access in Vehicular Environments (WAVE)--Identifiers," in IEEE Std 1609.12-2019 (Revision of IEEE Std 1609.12-2016) , vol., no., pp.1-17, 22 Oct. 2019, doi: 10.1109/IEEESTD.2019.8877516.

[10] "IEEE Standard Ontologies for Robotics and Automation," in IEEE Std 1872-2015 , vol., no., pp.1-60, 10 April 2015, doi: 10.1109/IEEESTD.2015.7084073.

[11] "IEEE Code of Ethics," *IEEE*. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 02-Mar-2021].

https://www.nspe.org/sites/default/files/resources/pdfs/Ethics/CodeofEthics/NSPECodeofEthicsforEngineers.pdf. [Accessed on March 4, 2021].

[12] M. Ramsey. "Self-Driving Cars Could Cut Down on Accidents, Study Says," *The Wall Street Journal,* Mar 15, 2015. Accessed on March 3, 2021. [Online]. Available:https://www.wsj.com/articles/self-driving-cars-could-cut-down-on-accidents-study-says-1425567905

[13] "Heavy and Tractor-trailer Truck Drivers," *US Bureau of Labor Statistics,* September 16, 2020. Accessed on March 4, 2021. [Online]. Available: https://www.bls.gov/ooh/transportation-and-material-moving/heavy-and-tractor-trailer-truck-drivers.htm

[14] "Global Greenhouse Gas Emissions Data," *United States Environmental Protection Agency,* Accessed on March 17th, 2021. [Online]. Available: https://www.epa.gov/ghgemissions/global-greenhouse-gas-emissions-data#Sector

[15] B. Opher, B. Oded, N. Mehdi, "Introducing Autonomous Vehicles: Adoption Patterns and Impacts on Social Welfare,, Accessed on March 17th, 2021. [Online.] Available: https://pubsonline-informs-org.proxy.lib.odu.edu/doi/pdf/10.1287/msom.2020.0955

[16] David Rojas-Rueda, Mark J. Nieuwenhuijsen, Haneen Khreis, Howard Frumkin, Autonomous Vehicles and Public Health Annual Review of Public Health 2020 41:1, 329-345

[17] B. Van Meldert and L. De Boeck, "Introducing autonomous vehicles in logistics: a review from a broad perspective," *FEB Research Report KBI_1618*, 2016.

[18] E. Alonso, C. Arpón, M. González, R. Á. Fernández, and M. Nieto, "Economic impact of autonomous vehicles in Spain," (in English), *European Transport Research Review*, vol. 12, no. 1, Dec 2020

[19] P. Coppola and D. Esztergár-Kiss, *Autonomous Vehicles and Future Mobility*. San Diego, UNITED STATES: Elsevier, 2019.

[20] P. Kopelias et al., "Connected & autonomous vehicles – environmental impacts – a review," *Science of the Total Environment*, vol. 712, no. 135237, 2020, doi: https://doi.org/10.1016/j.scitotenv.2019.135237.

[21] D. J. Fagnant and K. M. Kockelman, “The travel and environmental implications of shared autonomous vehicles, using agent-based model scenarios,” *Transportation Research Part C: Emerging Technologies*, vol. 40, pp. 1-13, 2014, doi: https://doi.org/10.1016/j.trc.2013.12.001.

[22] James M. Anderson et al.“The promise and perils of autonomous vehicle technology.” *Autonomous Vehicle Technology: A Guide for Policymakers*, RAND Corporation, 2014, pp. 9–40. *JSTOR*, www.jstor.org/stable/10.7249/j.ctt5hhwgz.9

[23] M. Cunneen, M. Mullins, F. Murphy, D. Shannon, I. Furxhi, and C. Ryan, “Autonomous vehicles and avoiding the trolley (dilemma): vehicle perception, classification, and the challenges of framing decision ethics,” *Cybernetics and Systems*, vol. 51, no. 1, pp. 59–80, 2019.

Appendix

A. Auto-Nav Vehicle Configuration Requirements

Design	- Ground vehicles must be propelled by direct mechanical contact to the ground by wheels, tracks, pods, or hovercraft.
Length	- Minimum: 3 feet - Maximum: 7 feet
Width	- Minimum: 2 feet - Maximum 4 feet
Height	- Not to exceed 6 feet (excluding E-stop antenna)
Propulsion	- Power must be generated onboard the vehicle - Fuel storage, running combustion engines, and fuel cells are not permitted in the team's maintenance area
Average Speed	- above 1 mph over the entire run
Minimum Speed	- 1 mph
Maximum Speed	- 5 mph
Mechanical E-stop	- Must be a red push-button at least 1 inch in diameter - mounted center, rear of the vehicle - at least 2 feet from the ground - no more than 4 feet from the ground - must be hardware based, not software controlled - must bring the vehicle to a complete stop
Wireless E-stop	- must be effective for a minimum of 100 feet - must be hardware based, not software controlled - must bring the vehicle to a complete stop - the wireless E-stop will be held by the Judges
Safety Light	- must have easily viewed solid indicator light which is turned on whenever the vehicle power is turned on - light must go from solid to flashing whenever the vehicle is in autonomous mode - light must go back to solid as soon as the vehicle comes out of autonomous mode
Payload	- payload specifications: 18" X 8" x 8", 20 pounds - payload must be securely mounted on vehicle - if payload falls off during a run, the run will be terminated

B. Self Drive Vehicle Configuration Requirements

Design	- Side by Side 2-person, 4-wheel vehicle - Electric, no gas
Length	Maximum: 115 in
Width	Maximum: 60 in
Height	Maximum: 75 in
Weight	Maximum: 1500 lbs
Average Speed	- above 1 mph over the entire run
Minimum Speed	- 1 mph
Maximum Speed	- 5 mph - 2 mph (Reverse)
Mechanical E-stop	- push to stop, red, minimum of one inch in diameter - must be located inside the cabin, outside on sides, and rear of vehicle - hardware based, not controlled through software - must bring vehicle to quick and complete stop
Wireless E-stop	- effective for a minimum of 100 feet - wireless E-stop will be held by judges during competition
Safety Light	- brake lights: red - reverse lights: yellow - strobe light must be mounted on roof and activated when vehicle is under robotic control - light must be blinking during autonomous mode - light must be off otherwise

C. Inventory

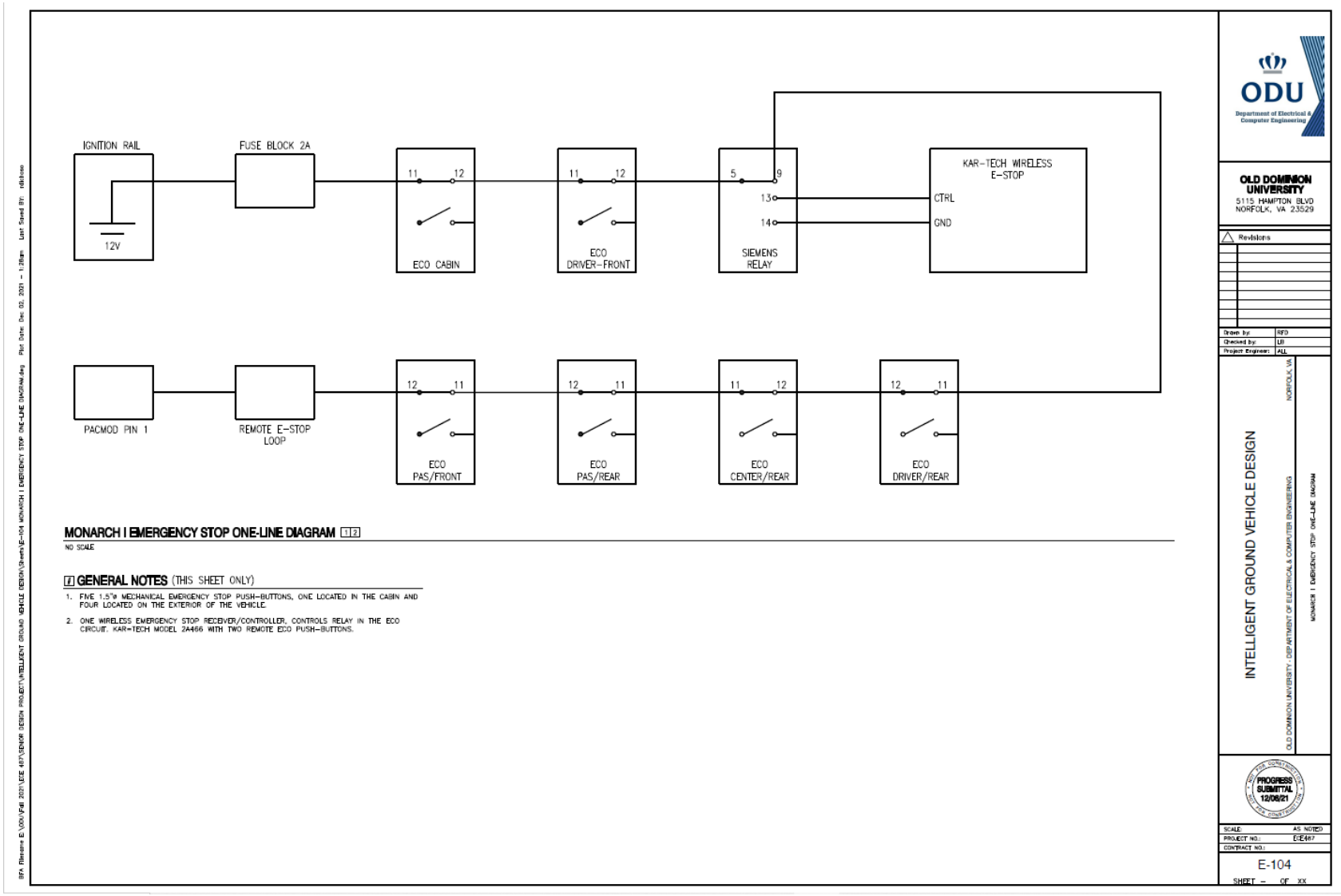
Little Blue v1.0 Specific Items		
Component	**Quantity**	**Description/Use**
Gigabyte Z390 UD ATX Motherboard	1	Chipset
Intel Core i7-9700K 3.6Ghz 8-Core	1	CPU
NVidia GeForce RTX 2060	1	GPU
Raspberry Pi 4 4GB	3	Computer vision processing nodes
Raspberry Pi V2-8 Cameras	3	Facilitates computer vision
Arduino Nano	1	Drive system microcontroller
XSenx Mti-670 IMU	1	Inertial Measurement Unit
ZED-F9P GPS Unit	1	Global Positioning System Break-out
RPLIDAR A2	1	360 degree Lidar sensor

Little Blue v2.0 Specific Items		
Component	**Quantity**	**Description/Use**
Kunray 24V 350W hub-motor wheels	4	Drive system
NVidia Jetson TX2	1	CPU and neural network capabilities
12V, 40 Ah Li-ion Battery	1	Power for processing unit, electronics, and sensors
24V, 40 Ah Li-ion Battery	1	Power for drive system
ZED 2 Stereo Camera	1	Camera for lane following, object detection, and ranging
ZED-F9P GPS Unit	1	Global Positioning System Break-out

Monarch I Specific Items		
Component	**Quantity**	**Description/Use**
Intel Xeon® E3 v5 CPU	1	Spectra CPU
EVGA GeForce RTX 2080 Super GPU	1	Spectra GPU
Velodyne VLP-16 Lidar	1	360 degree Lidar sensor
Triton Camera	2	Camera
Distance AGM Battery Bank	1 (8)	Power for front-wheel drive system - consists of eight 6V, 224 Ah batteries
XSenx Mti-670 IMU	1	Inertial Measurement Unit
ZED-F9P GPS Unit	1	Global Positioning System Break-out
ZED 2 Stereo Camera	1	Camera for lane following, object detection, and ranging

Universal Vehicle Support and Spare parts		
Component	**Quantity**	**Description/Use**
ZED-F9P GPS Unit	1	Global Positioning System Break-out
Sparkfun RTK Surveyor	1	Base Station NTRIP Server unit
GNSS Multi-Band L1/L2 Surveying Antenna	1	Base Station NTRIP Server support

D. Monarch I: Power Diagram & Emergency Stop One-Line Diagram



E. Monarch 1 Vehicle Procedure Flowchart and E-Stop Schematic

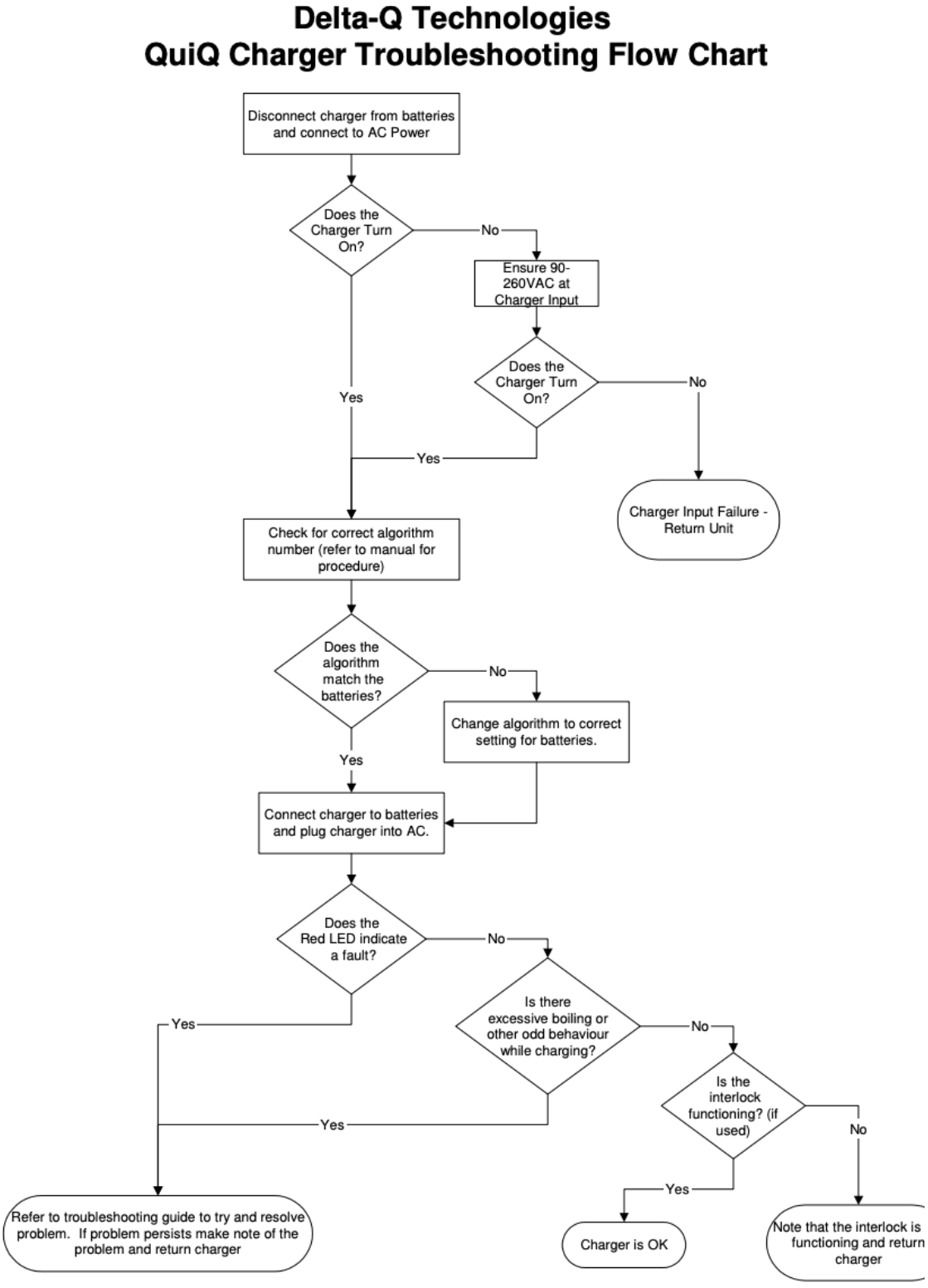


Figure E.1: Delta-Q Charger Troubleshooting Flowchart

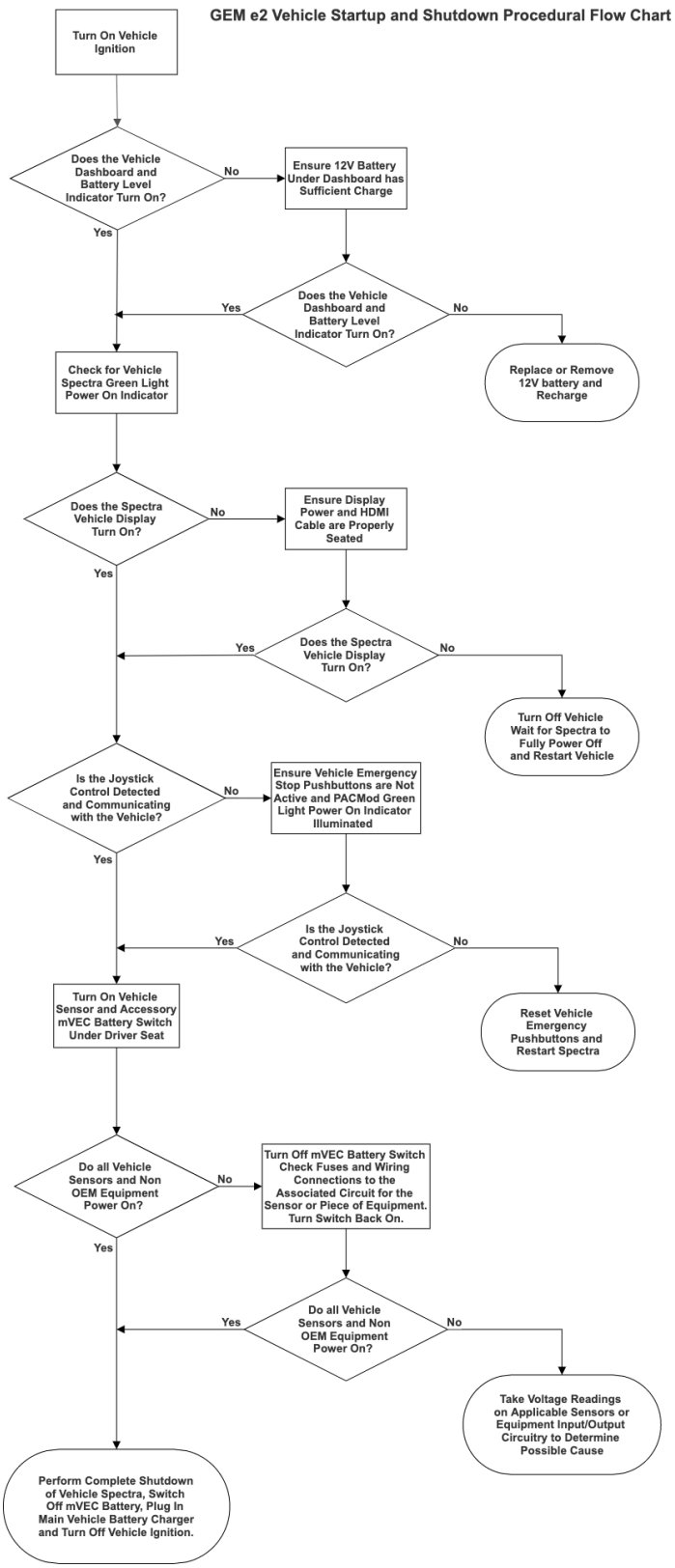


Figure E.2: Monarch 1 Vehicle Startup and Shutdown

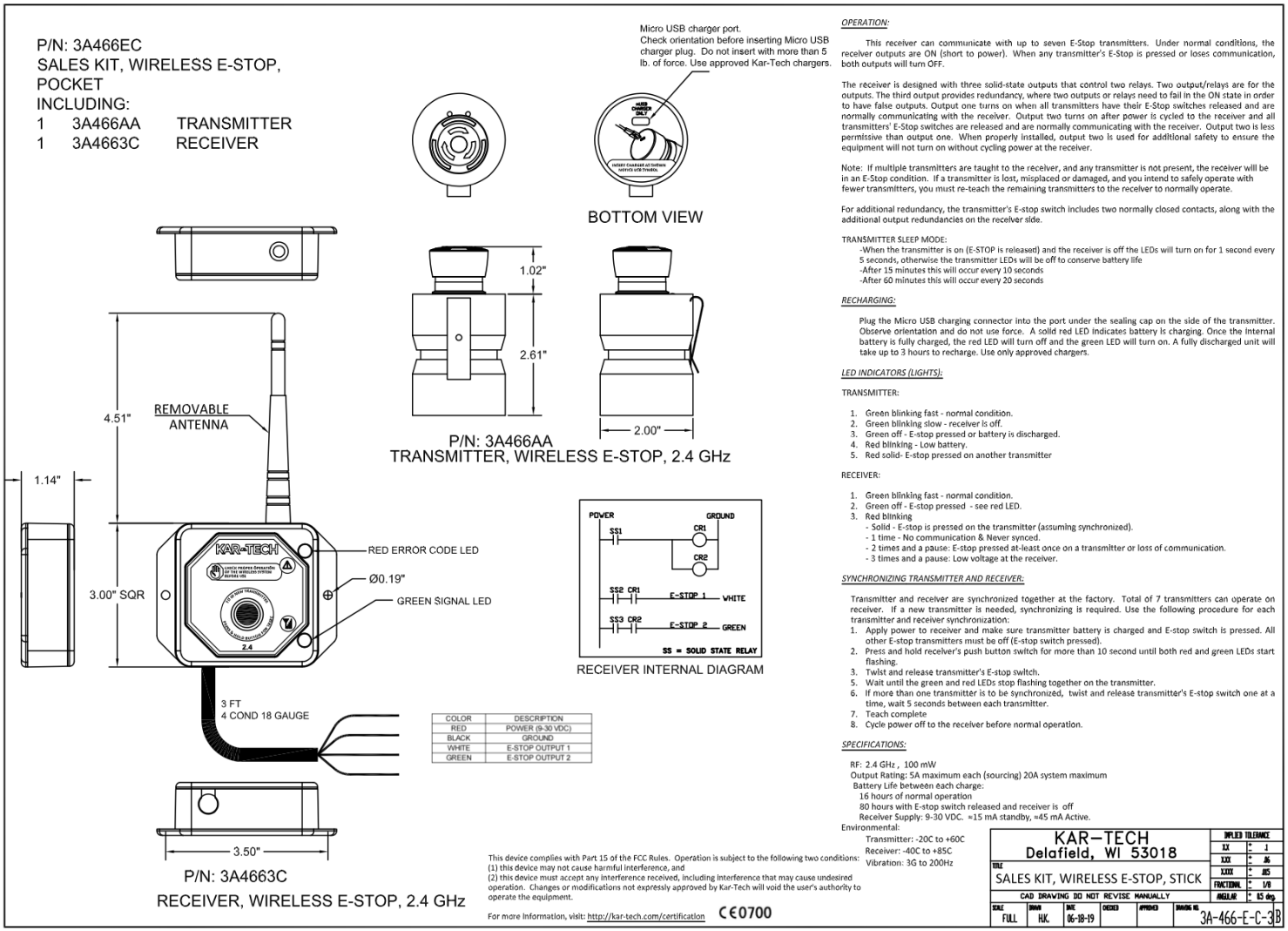


Figure E.3: Wireless E-Stop Schematic

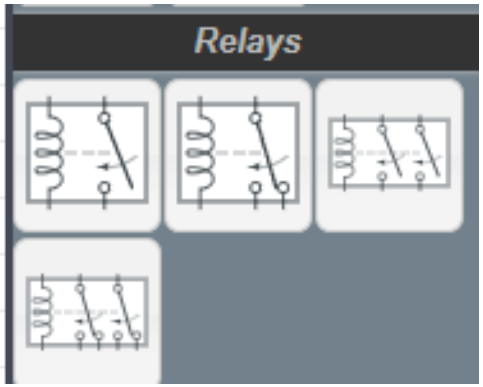


Figure E.4: Wireless E-Stop Relay

Original Design

Primary Power	Secondary Power	System Before Button Pressed	Primary Button	Secondary Button	System After Button Pressed
On	Off	On	Pressed		system stays off
On	On	On	Pressed		system turns off for 4 seconds, then turns back on
On	On	On		Pressed	system sputters, then stays on
Off	On	On		Pressed	system stays off

Figure E:5 Wireless E-Stop Original Design Troubleshoot

New Design

Primary Power	Secondary Power	System Before Button Pressed	Primary Button	Secondary Button	System After Button Pressed
On	Off	On	Pressed		system stays off
On	On	On	Pressed		system turns off for 4 seconds, then turns back on
On	On	On		Pressed	system sputters, then stays on
Off	On	On		Pressed	system stays off

Figure E:6 Wireless E-Stop New Design Troubleshoot

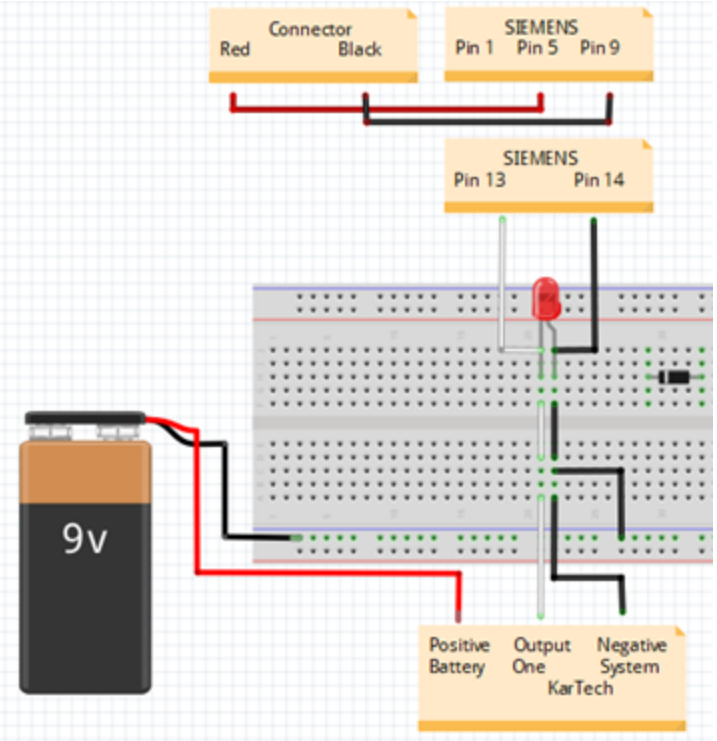


Figure E:7 E-Stop Original Design Circuit

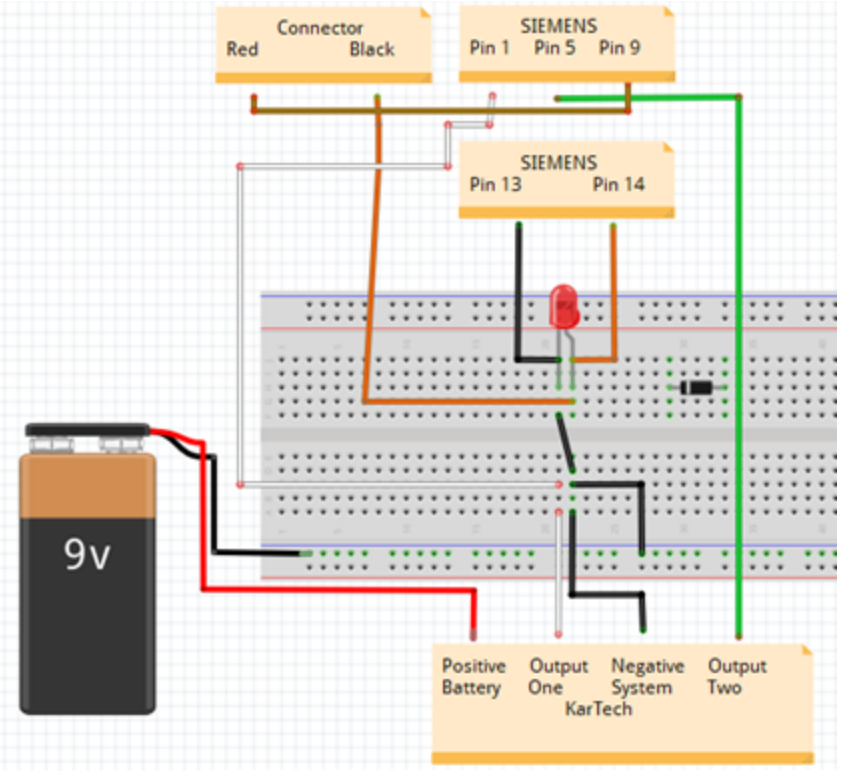


Figure E:8 E-Stop Revised Design

F. Lane Detection Images Blown-up

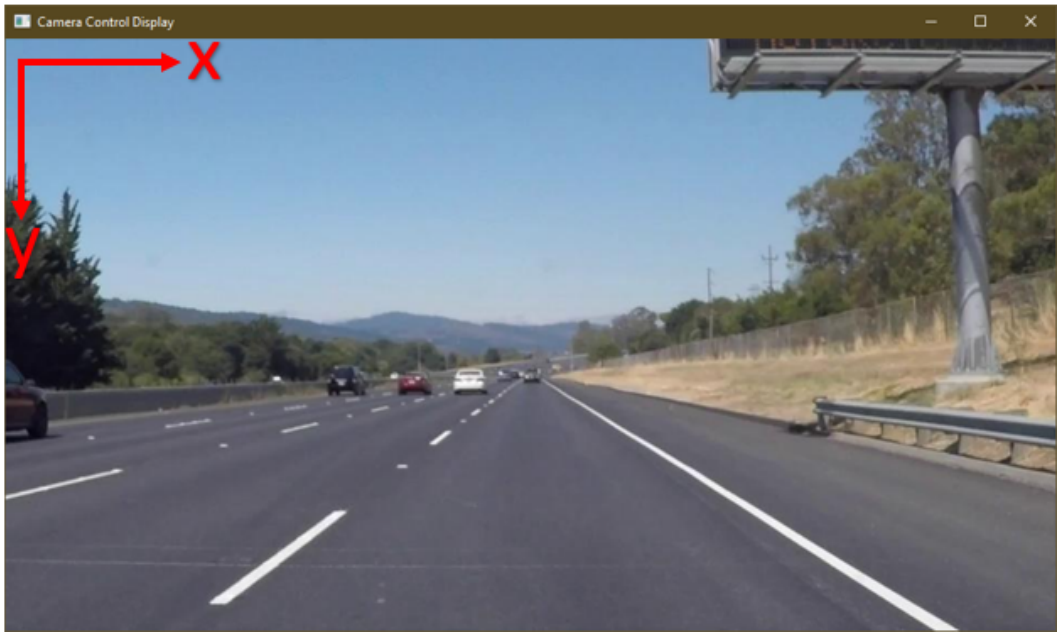


Figure F.1: Original Image

Figure F.2: Grayscale Image

Figure F.3: Grayscale Image with Gaussian Blur (5x5 pixel kernel)

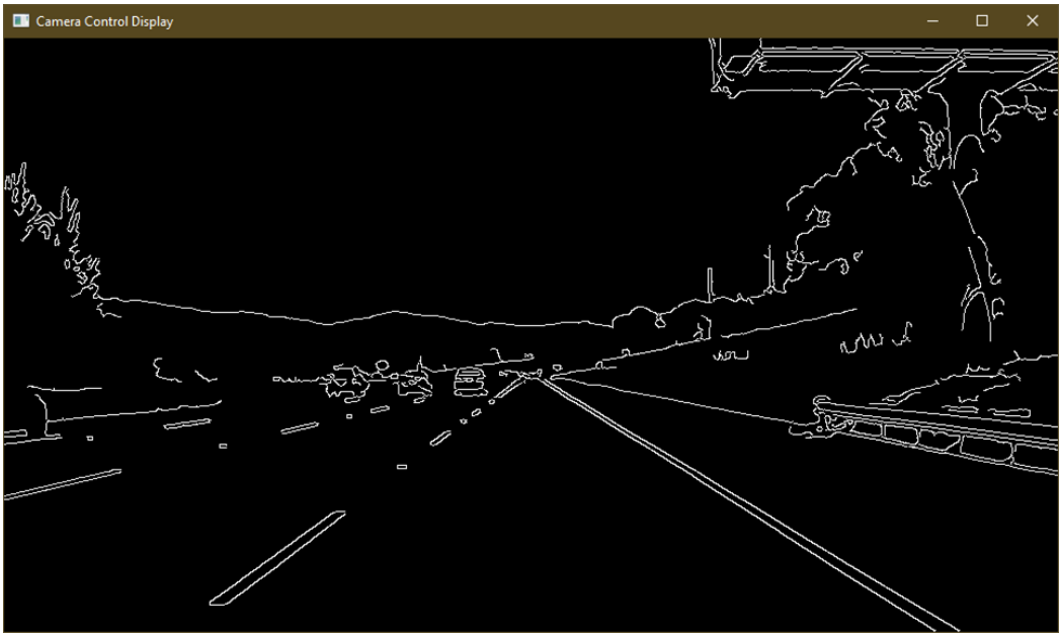


Figure F.4: Canny Edge Detection Image

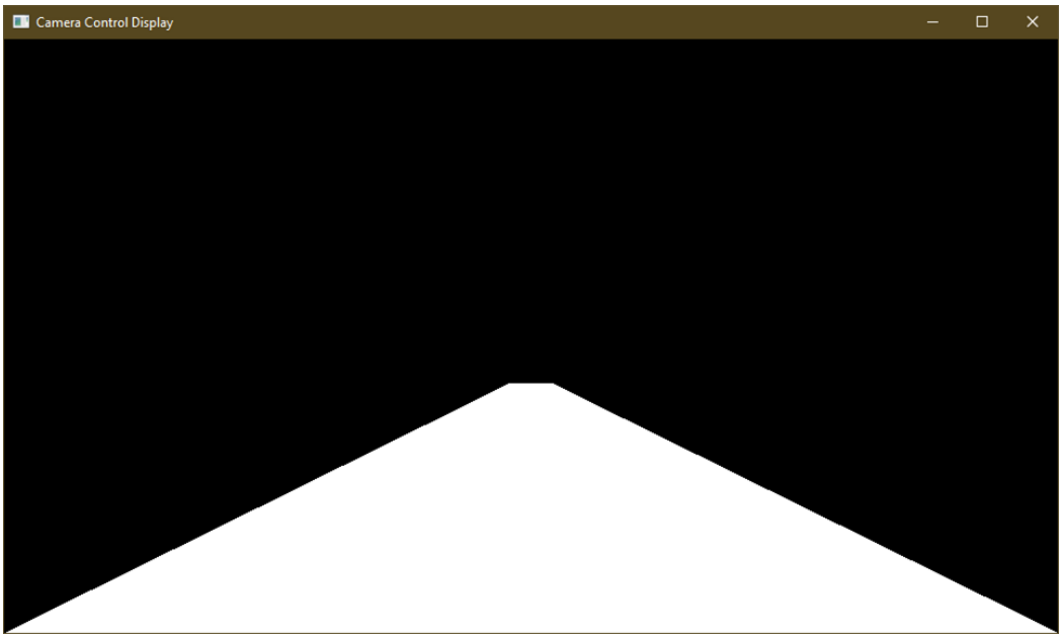


Figure F.5: Masking Image

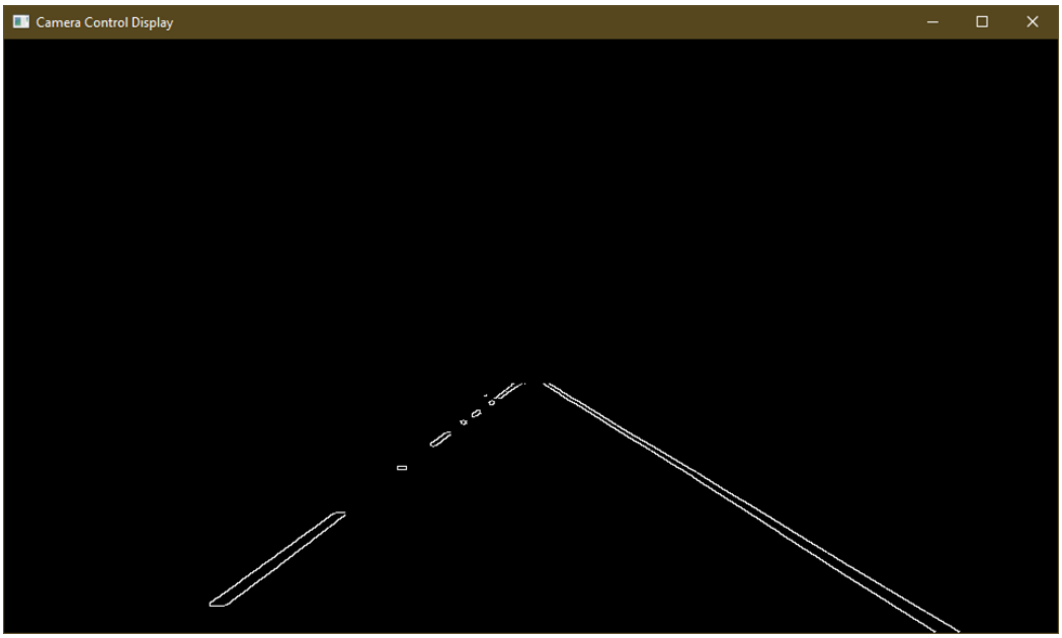


Figure F.6: Canny Image ANDed with Masking Applied

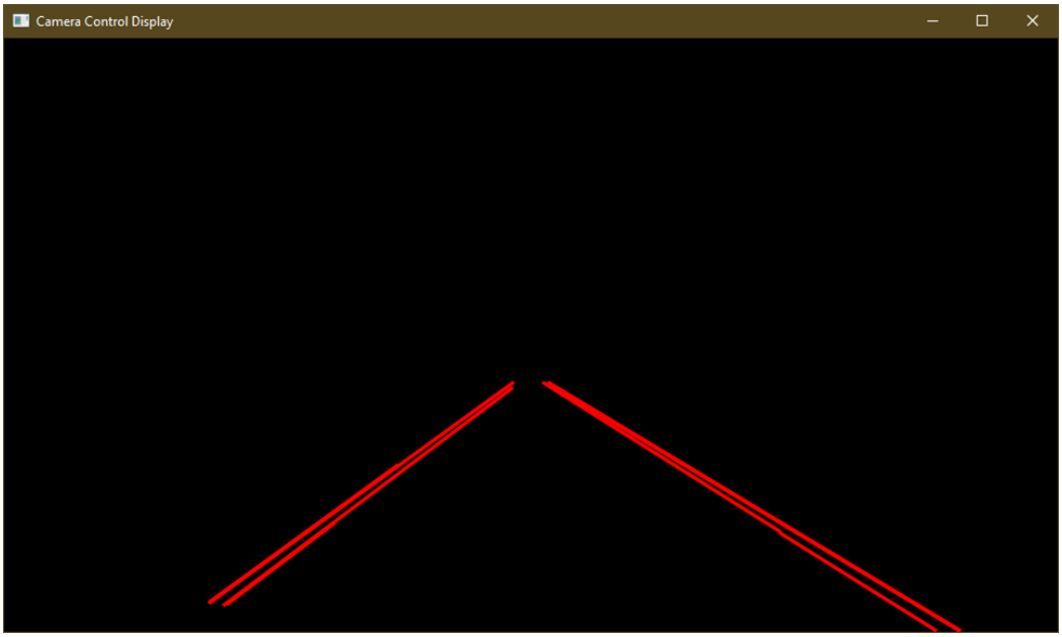


Figure F.7: Probabilistic Hough Line Detection on Masked Canny Image

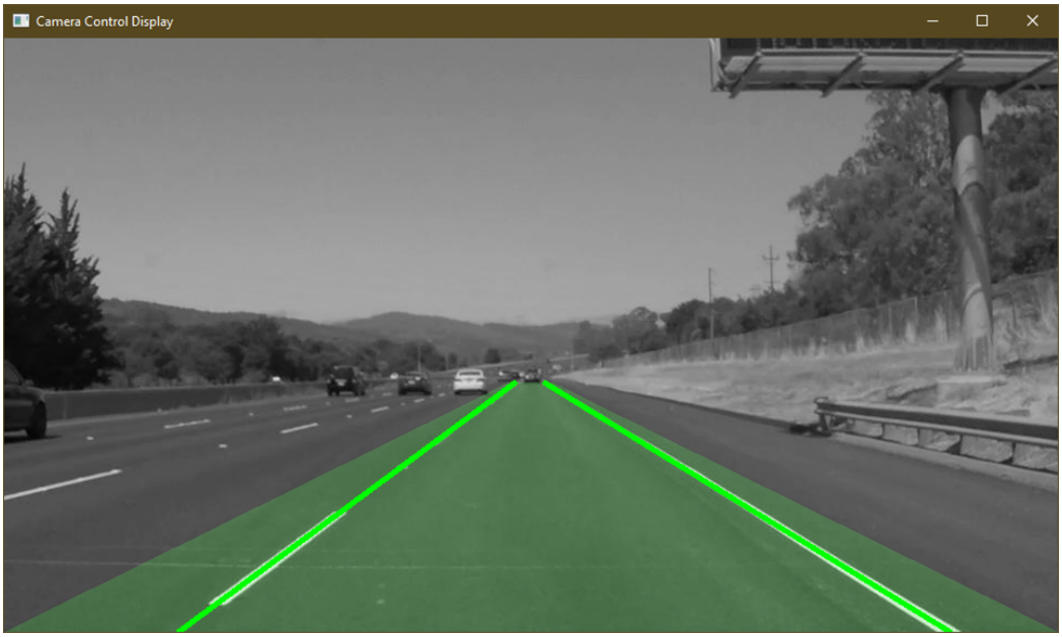


Figure F.8: Averaged Lane Line Image with Detection Region (Mask) Overlay

G. PACMod Topics Explained

Rostopic	Message Type	Description
can_tx	can_msgs/Frame	Data to be parsed by the PACMod driver via CAN interface.
as_rx/accel_cmd	pacmod_msgs/PacmodCmd	Commands gas pedal position [0.0 to 1.0]
as_rx/brake_cmd	pacmod_msgs/PacmodCmd	Commands brake pedal position [0.0 to 1.0]
as_rx/shift_cmd	pacmod_msgs/PacmodCmd	Commands transmission
as_rx/steer_cmd	pacmod_msgs/PositionWithSpeed	Commands steering wheel angle (rad) and angular velocity limit (rad/s)
as_rx/turn_cmd	pacmod_msgs/PacmodCmd	Commands turn signal
as_rx/enable	std_msgs/Bool	Enables PACMod's control of vehicle when true

Table G.1: PACMod Subscribed Topics

Rostopic	Message Type	Description
can_rx	can_msgs/Frame	Data sent to PACMod system via CAN interface
parsed_tx/global_rpt	pacmod_msgs/GlobalRpt	Data for entire PACMod system
parsed_tx/accel_rpt	pacmod_msgs/SystemRptFloat	Parsed values for throttle subsystem
parsed_tx/brake_rpt	pacmod_msgs/SystemRptFloat	Parsed values for brake subsystem
parsed_tx/turn_rpt	pacmod_msgs/SystemRptInt	Parsed values for turn signal subsystem
parsed_tx/shift_rpt	pacmod_msgs/SystemRptInt	Parsed values of the transmission subsystem
parsed_tx/steer_rpt	pacmod_msgs/SystemRptFloat	Parsed values (rad) of the steering subsystem
as_tx/vehicle_speed	std_msgs/Float64	Vehicle’s current speed (m/s)
as_tx/enable	std_msgs/Bool	Status of the PACMod’s control of the vehicle. True if enabled.

Table G.2: PACMod Published Topics

Adapted from http://wiki.ros.org/pacmod

H. Monarch I - mVEC Power Distribution System (PDS)

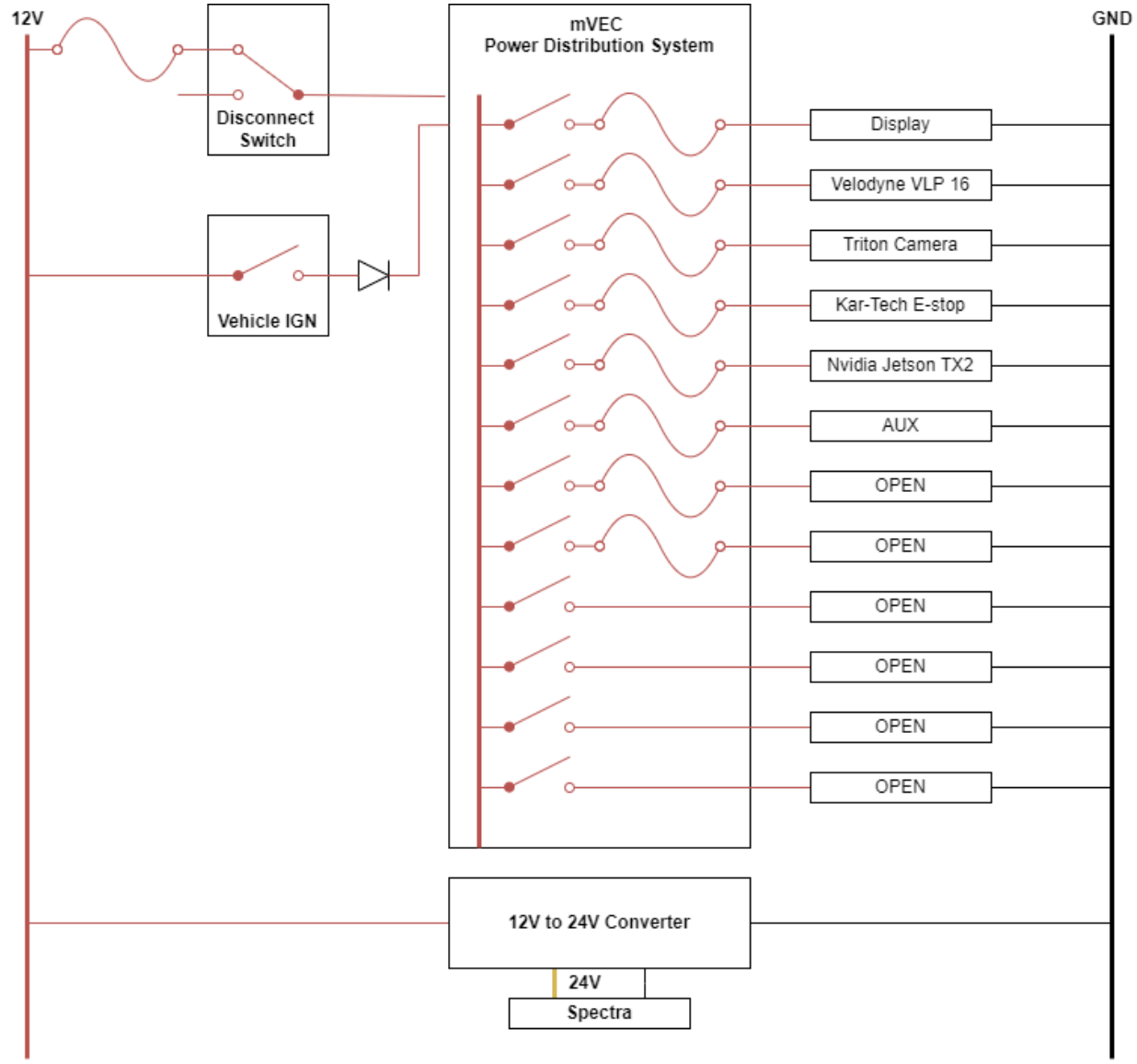


Figure H.1. Monarch I Power Distribution