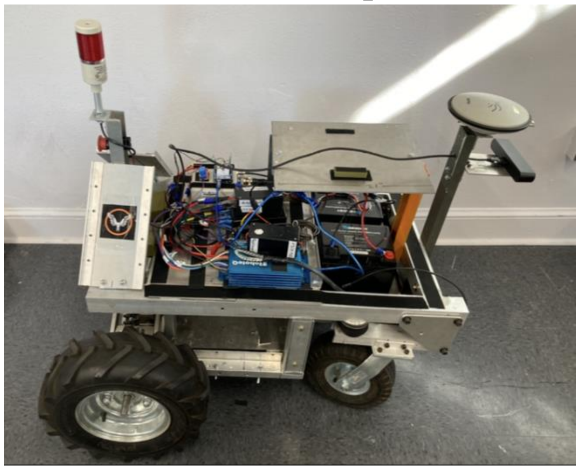
Virginia Polytechnic Institute and State University

VT CRO - Raphael



Submitted May 15, 2024

Team Member	Email	Major
Christian Hackett (Team Lead)	cjhackett50@vt.edu	Mechanical Engineering
Christopher Rosend	cmrosend@vt.edu	Mechanical Engineering
Jingcheng Luo	ljingcheng19@vt.edu	Mechanical Engineering
James O'Donovan	jamesjo@vt.edu	Electrical Engineering
Shreya Balaji	shreyabalaji@vt.edu	Computer Engineering
Harrison Bui	bharrison20@vt.edu	Computer Engineering
Jack Orr	jack524@vt.edu	Computer Engineering
Rahul Muthuraman Shanmugam	rahulms@vt.edu	Computer Engineering
Joe Bekiranov	jb5@vt.edu	Computer Engineering
John Shebey	jshebey@vt.edu	Computer Engineering
Timonthy Fish	timonthyfish@vt.edu	Computer Engineering

Advisors

- Marie Paretti, PhD, Professor of Engineering Education at Virginia Tech
- Stephen Moyer, Visiting Professor of Practice at Virginia Tech

Statement of Integrity

I hereby state the engineering design and development of Rapheal was completed by the current student team and is equivalent to a senior design capstone course at Virginia Tech.

[illegible]

Stephen Moyer
semoyer@vt.edu

Introduction

Virginia Tech has not participated in the Intelligent Ground Vehicle Competition (IGVC) since 2014. We are continuing from where the previous team ended. Our goal is to lay the foundation for future IGVC teams. For the 2024 competition, we improved the old 2014 robot by redesigning it solely for the Intelligent Ground Vehicle Competition. We focused on replacing components with upgraded systems, streamlining the design, and implementing processes necessary for the competition. The 2023-2024 IGVC team took an outdated robot and turned it into something competition worthy.

Organization

VT CRO is the Competitive Robotics Organization at Virginia Tech. It is made up of four engineering design teams including the IGVC Team. The IGVC Team was created in August of 2023, with the goal to compete in the 2024 Intelligent Ground Vehicle Competition. The team has seven senior engineering students and four junior engineering students. The team is divided into two sub-teams: mechanical and software. The team lead reports the team's progress to the President and other executives of VT CRO.

Design Process and Assumptions

The goal of this IGVC team was to satisfy all the requirements to qualify for the competition while staying within our very limited budget. In September 2023, we received three old robots built for mapping competitions. One of the three robots was sent to the 2014 competition as it passed the requirements for qualification. We fully disassembled this old IGVC robot to gain perspective on how the previous IGVC team engineered their robot. Upon reconstruction, we had a basic chassis which required major modifications. We designed the remainder of the robot based on ease of usability and cost effectiveness. To stay within our budget, the mechanical team recycled materials from the other two robots. With the limited funds, there was enough money to purchase new batteries, a camera, and a laptop for the software team.

Key Innovations

The most effective innovation to our robot design is the use of Velcro. To avoid the hassle of bolts and screws, we lined the electrical board with Velcro. Because of the use of Velcro, we can quickly remove and reorientate every electrical component on the robot. Due to our limited budget, our prototyping phase was very long. As we were redesigning this robot, we were constantly changing the layout of the board. The mechanical sub-team was frustrated with frequently drilling new holes, bolting and unbolting items to the robot. We all agreed that it would be easier to simply stick the components on the robot. We purchased a roll of Velcro to test our idea and it worked as intended. We have saved time by eliminating the need to drill new holes.

One of our key innovations in relation to software is our coordinate transfer system. We have utilized the ZED 2i stereo camera for vision and depth sensing. The depth sensor gives us coordinates from the robot's perspective. However, this can be difficult to visualize. To address this, we've devised a coordinate transfer system that converts these coordinates from the camera's viewpoint into overhead Cartesian coordinates, akin to a bird's eye view. This transformation enables us to seamlessly integrate the coordinates into a PyGame GUI, enhancing our ability to visualize obstacles in proximity to our robot.

Mechanical Design

Overview

According to their design document, the previous team's robot was retrofitted for the IGVC competition, so there were a couple design changes necessary for convenience and accessibility. First, we removed the old batteries and used the new space to hold the payload. The payload was then sealed off from the top so that an electrical board could be properly placed on the robot. This board was fitted with a lining of Velcro because the nature of the design phase forced us to move components around frequently. A separate stand was created for the laptop so that it would be separated from the nest of wires on the electrical board and could easily be accessed. Additionally, we added vertical beams to the front and back of the robot to attach the safety light, E-stop button, camera, and GPS. Lastly, since the competition will take place even if there is light rain, we used a plastic storage bin as a waterproofing device and added additional plates in the rear to protect the motors.

Chassis

According to the previous Virginia Tech team's report, their robot was retrofitted for the IGVC competition. After removing all unnecessary parts, we had a functional chassis that we could reuse and build upon. The chassis of Rapheal was originally built from 6061 Aluminum alloy. The frame is very durable as it was wielded together. Due to the solid construction of the chassis from the 2014 team, we are not concerned with stress on the frame. There are two driving wheels in the rear and two casters in the front. With all components and payload attached, we estimated the weight to be 180 to 200 pounds.

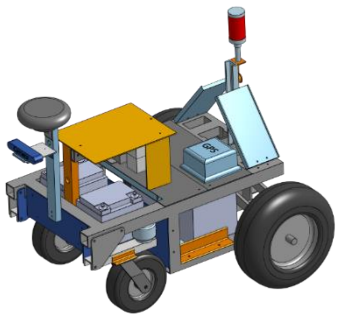


Figure 1: Raphael CAD model

Payload Storage

The first challenge the mechanical team faced was where to put the payload. There was not a clear spot on the robot to place a cinder block without damaging the electric components. Since the payload stays inside the robot, we designed a space under our electrical board as a storage bay. A cinder block can be slid between the frame of the robot and secured with L channels.

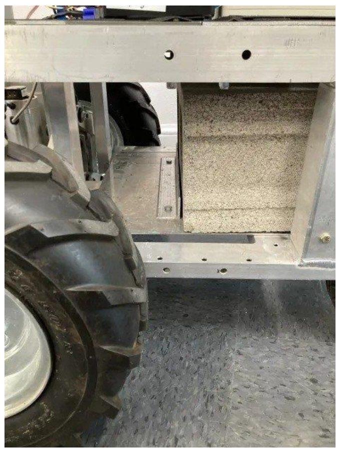


Figure 2: Payload storage area

Weatherproofing

Due to the large size of the robot, we had several different options for weatherproofing it. One idea was using plastic sheets and hinges to create a drawer system. We would be able to access the inside of the robot and close the door as needed with this system. However, this was deemed too costly because of our limited budget. The second idea we had was using a plastic storage bin to completely cover the robot with a single device. This was more cost effective, and we were able to purchase a storage bin which covered the entire body of the robot. To protect the motor wires, we 3D printed angled plates which insert between the gaps above the motors.

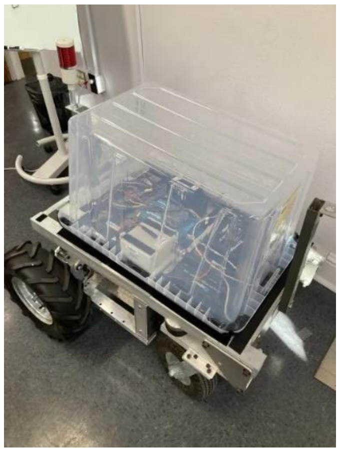


Figure 3: Weatherproofing

Electrical Design

Overview

Our robot uses nine main electrical components. The motor controller, two motors, safety light, and voltage regulator are 24V. The GPS is powered by 12V. The robot's laptop is powered independently and operates the Arduino and Zed2i camera. There are a few additional electrical components including resistors and microchips powered by the Arduino. Figure 4 is the overall circuit design for our robot.

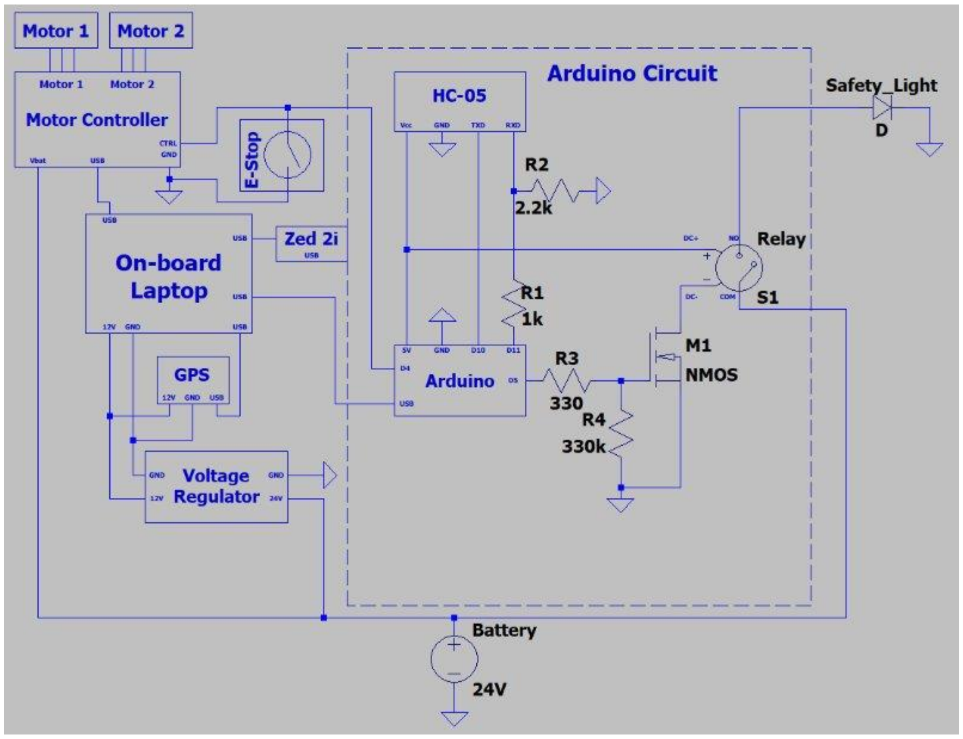


Figure 4: Overall circuit design

Batteries

We prioritized safety, performance, and the cost over size and weight. The batteries on the old IGVC robot were deemed unsafe to use. To power the robot, we purchased two 24V 25Ah lithium iron phosphate batteries. The batteries are wired in parallel to give a total of 50 Ah. These batteries are waterproof and protected from dust. They can operate in temperatures as low as –4 degrees Fahrenheit and as hot as 140 degrees Fahrenheit. With an operational life span of five years, they were the perfect batteries we needed for our robot.

Table 1: Current Draw Testing Results

Robot State	**Components Running**	**Current Draw (A)**
E-stop pressed	Safety light, GPS	0.51 A
E-stop unpressed, robot stopped	Safety light, GPS, Motor controller	0.63 A
Robot moving forward (no load)	Safety light, GPS, Motor controller, Motors	1.47 A
Robot moving forward (flat ground)	Safety light, GPS, Motor controller, Motors	6.63 A (Max) 2.73 A (Steady state)

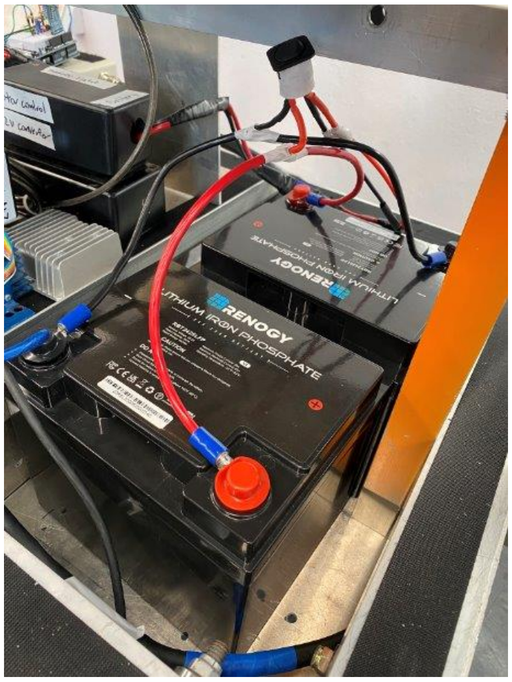


Figure 5: Batteries wired in parallel

Motors

We decided to use the motors from the old robot because they were still in good shape. The torque of the motors allows for 593.48 lbs. (269.2 kg) maximum weight of the robot to be pulled up the 15% gradient required for competition. Additionally, the full-load RPM along with the 16-inch diameter wheels gives a full-load speed of 2.570 mph which is within the 1-5 mph limit. The donated motors have the following statistics:

- Max Continuous Torque: 336 in-lbs. (37.96 Nm)
- Full-Load RPM: 54
- Operating Voltage: 24V
- Max Continuous Current Draw: 17A
- Add-ons: Hall Effect Encoder

Safety

To meet the requirements for safety, we are using a 24V LED Andon Stack Light. It draws 8W and is controlled using a relay. The light is powered when the robot is on and will flash whenever the robot is in autonomous mode. We used a relay to control the safety light to blink when in autonomous mode. We used a MOSFET transistor and a couple resistors because the relay draws 70 mA and the Arduino digital pin is 40 mA. A simple C++ code is used to control the blinking of the safety light

The robot also has a mechanical and wireless E-stop. The mechanical E-stop button is on the rear of the robot, right below the safety light. The motor controller we are reusing has a control pin that will automatically shut down the motor controller when connected to ground. Also, we are reusing the safety button from the old IGVC robot because of its functionality and adherence to the competition rules. When the E-stop is pressed, the control pin will connect to the ground and the robot will immediately stop.

To create a wireless E-stop, the motor controller ctrl pin was connected to one of the Arduino's digital output pins. The Arduino can directly turn off the motor controller without going through the main code. A Raspberry Pi controls the relay and is connected to a Bluetooth module. We designed a Bluetooth circuit by using an Arduino to communicate with an on-board laptop. We used an HC-05 Bluetooth chip and resistors to create the circuit. A voltage divider is used to prevent frying the HC-05 chip.   The theoretical range is at least 350 feet. During the testing phase, we were able to control the robot from over 150 feet away, well over the 100-foot minimum competition requirement.

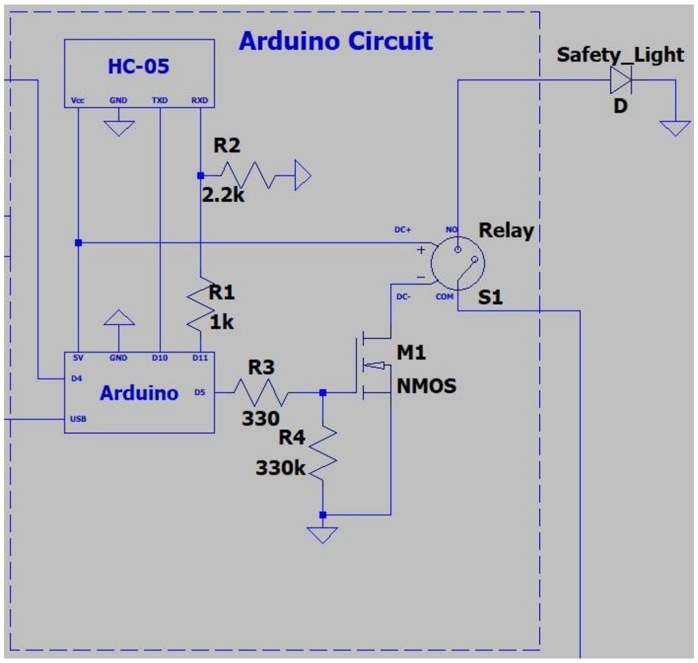


Figure 6: Arduino circuit design for safety light

<u>Software Design</u>

Overview

The core of our robot's software is written in Python. For obstacle detection, we rely on the depth data acquired from the stereo camera, while line detection utilizes the RGB image captured by the camera. Our code leverages a range of libraries, including OpenCV and PyZed, to facilitate these functionalities.

Obstacle Detection

To detect obstacles, we used a ZED 2i stereo camera to measure the distance away from the robot in each pixel of the robot's point of view. This camera was chosen specifically because of its built-in depth sensing software. To help the program run efficiently, only the middle line of pixels in the image was used. According to the rulebook, obstacles that will be used are construction barrels, drums, natural obstacles such as trees and bushes, and manmade obstacles such as light posts or street signs. This means, it is reasonable to assume these obstacles will be taller than the height at which the camera is placed on the robot (which is about 80 cm). While iterating through the pixels, any depth values which are less than a set threshold are added to a list of obstacles for the robot to avoid. If the robot is expected to collide into an obstacle on its current trajectory, the robot's path is adjusted by changing its turning angle until it is no longer headed towards this obstacle.

Line Detection

Line Detection is one of the main qualifications for the competition; the robot must follow and stay within two solid white lines drawn on asphalt. The line detection code uses various functions from OpenCV to process the image and detect the lines. The code begins by applying a white mask and identifying the region of interest. Then, a Gaussian Blur is added which gets rid of extra noise and makes the detection slightly more accurate. Next, a Canny Filter is applied to the image to obtain the edges of the images. The Canny edge detector is known for its ability to accurately detect edges while minimizing false positives. Once the edges are found, a Hough Lines Transform is used to find the lines. The Hough Lines transform returns two (x, y) coordinates for a line. First, the slope and intercept of each line is found. However, Hough Lines often outputs multiple small lines around the edges it detected in the Canny process. Hence, the average slope (and intercept) of all the found lines is taken. This average is then used to draw the line. If the slope is positive, the line is on the left side of the image, and if the slope is negative, the line is on the right side of the image. Since Canny returned the edges of the actual line, that is what Hough detects. Using the average function, two lines are drawn onto the image at the thickness of the real line. The below figure shows the steps listed above.

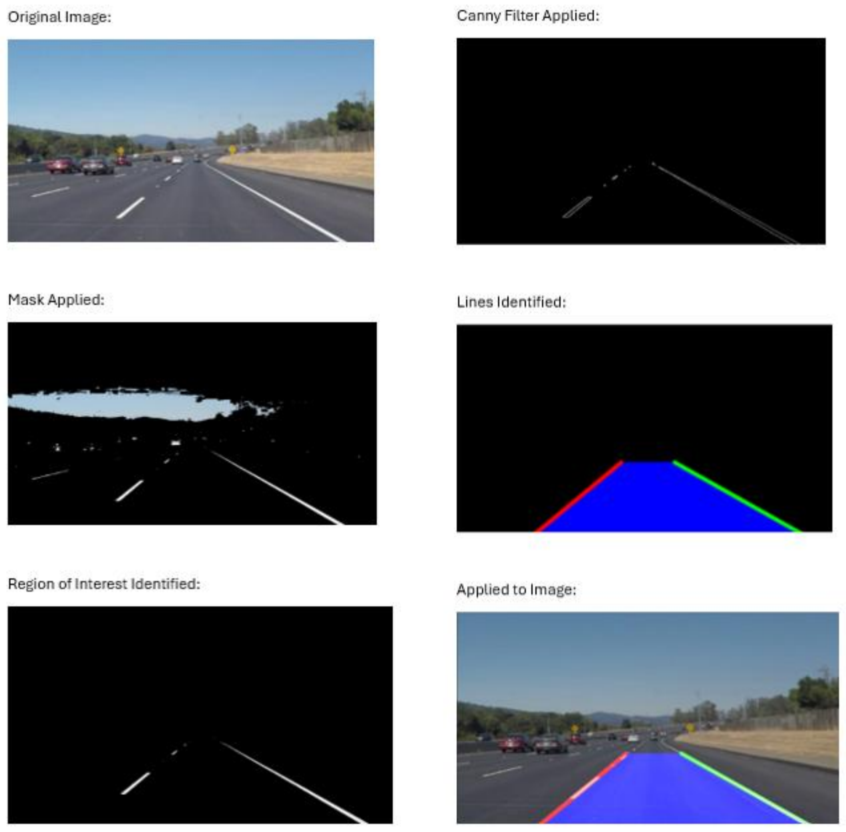


Figure 7: Line detection process

Path Planning

Our path finding is based on a combination of the measurements from the depth sensor and the location of the white lines. First, obstacle detection returns the objects' location through an array of pixels. Next, line detection draws out two lines on the image sent from the camera. These lines are then scanned, and the depths of each pixel are returned. These two arrays are combined into one obstacle list which is what the robot uses to path plan. For visualization purposes, the obstacle list is then converted from camera frame coordinates into overhead coordinates and is then plotted using a PyGame script. The logic for path planning is simple; the robot's pathing is to travel forward in a straight line while checking for any obstacles. If no obstacles are detected, the robot interacts with the motor controller and continues forward. However, if obstacles are present, the robot adjusts the angle of its path slightly until the path is clear. Below is an example of the GUI we have made to represent path planning. Figure 8 is the previously mentioned PyGame visualization, while Figure 9 is what the robot is physically seeing.

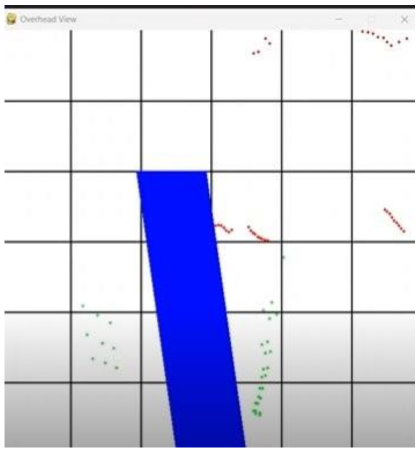


Figure 8: Path planning GUI

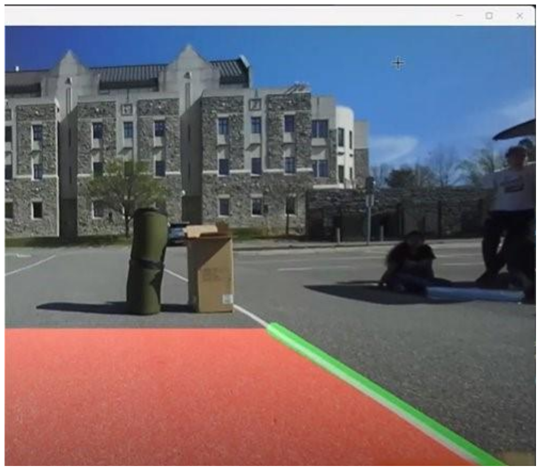


Figure 9: Path planning test

Testing Results

Table 2: Test Cases for Object Detection

Test	Results	Success?
Object far left	Robot went straight and did not try and avoid obstacles it didn’t need to. Line detection was glitchy when no lines were present so line detection was disabled for this test	YES
Object close left	Robot turned right just enough to avoid the obstacles. line detection was disabled for this test	YES
Object center	Robot turned left just enough to avoid the obstacles. line detection was disabled for this test	YES
Object close right	Robot turned left just enough to avoid the obstacles. line detection was disabled for this test	YES
Object far right	Robot went straight and did not attempt to turn to avoid the obstacles. line detection was disabled for this test	YES
No obstacle	Robot went mostly straight with a slight drift left. line detection was disabled for this test	YES
Straight lines no obstacles	Hugged left line but avoided crossing line until curve. A previous unrecorded attempt led to the robot successfully turning with the curve but partially crossed the line in the process	YES
Straight lines entering from angle	Robot self corrected and followed the lines	YES
GPS stop when entering waypoint radius	Robot followed lines and stopped within 6” of the waypoint. The gps uncertainty during the waypoint setting and testing was between $\pm$0.5m and $\pm$0.75m	YES
Straight lines obstacles in slalom	Robot curved back and forth between the obstacles like an elegant skier	YES
Speed test	Robot completed a 50 ft straight track in 17.01 s resulting in a speed of 2.939 ft/s = 2.0038 mph	YES
Bluetooth range test	Drove the robot over 100 ft away from the off-robot laptop and still maintained full control.	YES

Analysis of Complete Vehicle

The VT CRO-IGVC team completed their project goals. We took an old robot originally engineered for mapping and redesigned it to an autonomous robot with the abilities of obstacle avoidance and line detection. There were several roadblocks we experienced along the way. First, our budget was drastically cut halfway through the year. The mechanical team had to work with the leftover materials from old robots to complete our project. We were not able to make all of the modifications we wanted to increase maneuverability. Second, the software team had to write their code from scratch because we could not access the previous team’s work. Additionally, there were difficulties integrating the line detection with the obstacle detection.

The top hardware failure we could encounter would be improper wiring of the robot at competition. With several electrical components operating at different voltages, we needed to take steps to reduce the potential for improper wiring. To accomplish this, the mechanical team meticulously labelled all wires on the robot. Thus, anyone on the team can correctly wire the components on our electrical board.

As seen in Table 2, we have passed all appropriate tests required for the competition. When testing the obstacle detection, the robot would turn if an object was in its path. If an object was detected but not in the path of the robot, the path would remain the same. For the speed test, we recorded the time it took to travel 50 feet. The robot completed the track in 17 seconds at a speed of 2 mph. The last major test was the Bluetooth range test. We manually controlled the robot using a second laptop, successfully driving it from 153 feet away, well within the competition's required range.

Finally, the vehicle meets the requirements for the Intelligent Ground Vehicle Competition. However, there are several improvements we could make to the robot. The first is fully redesigning the chassis to increase maneuverability. Second, we would like to use a processor instead of a bulky laptop, meaning we would need to ensure the software libraries necessary for operating the robot are compatible with the operating system, hardware architecture, and other configuration requirements of the processor. Lastly, we want to improve the path planning of the robot by programming the robot to move backwards to correct itself. Now that we have laid the foundation for future IGVC teams, we hope VT CRO will have greater success in the years to come.