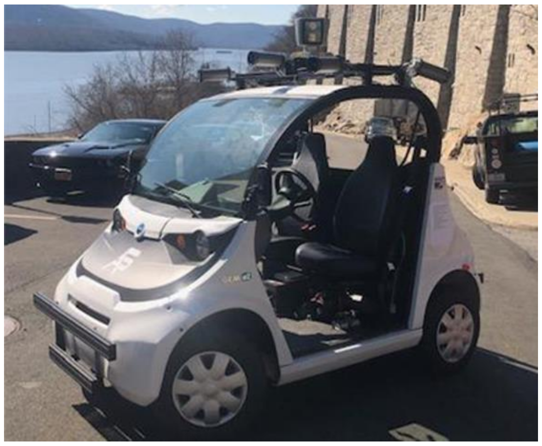
UNITED STATES MILITARY ACADEMY

Autonomous Vehicle Research and Design (AVRAD)

IGVC – Self Drive Design Report – 08 May 2020



Team Member	Contact Information
CDT Will Anderson	will.anderson@westpoint.edu
CDT Reed Burton	reed.burton@westpoint.edu
CDT Preston Draelos	preston.draelos@westpoint.edu
CDT Jarred Fassett	jarred.fassett@westpoint.edu
CDT Edward Kang	edward.kang@westpoint.edu
CDT Christopher Little – Team Captain	christopher.little@westpoint.edu
CDT Sean Min	sean.min@westpoint.edu
CDT Austin Morock	austin.morock@westpoint.edu
CDT Samuel Pool	samuel.pool@westpoint.edu
CDT Mitchell Stiffler	mitchell.stiffler@westpoint.edu

Faculty Advisor	Contact Information
COL Christopher Korpela	christopher.korpela@westpoint.edu
LTC Kathryn Pegues	kathryn.pegues@westpoint.edu
MAJ Joseph Cymerman	joseph.cymerman@westpoint.edu
MAJ Mark Lesak	mark.lesak@westpoint.edu
MAJ Courtney Razon	courtney.razon@westpoint.edu
MAJ (ret.) Dominic Larkin	dominic.larkin@westpoint.edu
Dr. Daniel J. Gonzalez	daniel.gonzalez@westpoint.edu
Dr. Peter Hanlon	peter.hanlon@westpoint.edu
Mr. Nicholas Livingston	nicholas.livingston@westpoint.edu

ABSTRACT

USMA Autonomous Vehicle Research and Design (AVRAD) is a team of 10 undergraduate engineers and computer scientists working to develop an unmanned ground system (UGS) capable of autonomous urban navigation, with the ultimate goal of competing and winning at the Intelligent Ground Vehicle Competition (IGVC). The software toolkits of winning IGVC submissions are integrated into the Robotic Technology Kernel (RTK), the Department of Defense's (DoD) platform-agnostic architecture for sensory perception and locomotion. RTK is employed on all Army unmanned ground systems. The existing, fielded RTK architecture is suited for autonomous navigation in ungoverned spaces, but lacks urban navigation capabilities. This capability gap limits the capacity of the modern soldier to utilize autonomous UGS in densely populated areas, given that the employment of UGS in the battlespace provides a significant technological advantage by increasing unit lethality and decreasing unit mortality. Given that each member of the AVRAD team will commission as an Army officer and lead soldiers upon graduating from USMA, AVRAD has a vested interest in developing a solution which wins at IGVC to bridge RTK's autonomous navigation capability gap, to effectively mitigate battlefield risk, and to empower the modern warfighter with cutting-edge technology.

1. INTRODUCTION

1.1. Acknowledgements

AVRAD greatly appreciates the support given to our team by Mr. Jerry Lane and his company, Great Lakes Systems & Technology LLC (GLS&T). Not only does GLS&T lead the charge in hosting the annual IGVC event in Warren, Michigan, but Mr. Lane has graciously engaged with the AVRAD project for years. AVRAD's success would not have been possible without his guidance. He joined the team in monthly teleconferences in which he reviewed our design progress and provided direct feedback on our results.

AVRAD also appreciates the guidance and financial support we have received from the engineers and scientists at Combat Capabilities Development Command Ground Vehicle Systems Center (CCDC-GVSC). CCDC-GVSC curates the RTK architecture, participates in hosting IGVC, and has provided direct support to our team by offering guidance and information without which our integration of RTK software packages would have been impossible.

Finally, AVRAD would like to thank the team of advisors from the USMA Departments of Civil & Mechanical Engineering (CME), Electrical Engineering & Computer Science (EECS), and Systems Engineering (SE) whose leadership set the conditions for our success in our endeavor: COL Christopher Korpela, LTC Kathryn Pegues, MAJ Joseph Cymerman, MAJ Mark Lesak, MAJ Courtney Razon, MAJ (ret.) Dominic Larkin, Dr. Daniel Gonzalez, Dr. Peter Hanlon, and Mr. Nicholas Livingston.

1.2. Problem Definition

AVRAD framed the urban navigation problem in terms of the rules and qualification requirements for IGVC 2020. At a minimum, the system needs to be capable of maintaining a controlled speed between 1-5 miles per hour, follow and remain centered within an 8 foot wide lane, detect and avoid obstacles, and react to stop signs and traverse intersections, all while navigating to within a 1 meter radius of a global positioning system (GPS) waypoint using a route requiring the shortest distance travelled in total. The system also needed to operate on software which could be integrated into RTK, and must be capable of conducting an emergency stop if a lane safety, grader, or the operator deem its behavior unsafe. Based on these requirements, the team drafted its problem statement, which would govern the focus and lines of effort for the duration of the project: *"AVRAD develops and tests a means of following lanes, avoiding road obstacles, and navigating intersections, in accordance with the rules and guidelines of the Intelligent Ground Vehicle Competition, and integrates these functionalities into the Robotic Technology Kernel."*

1.3. Team Composition

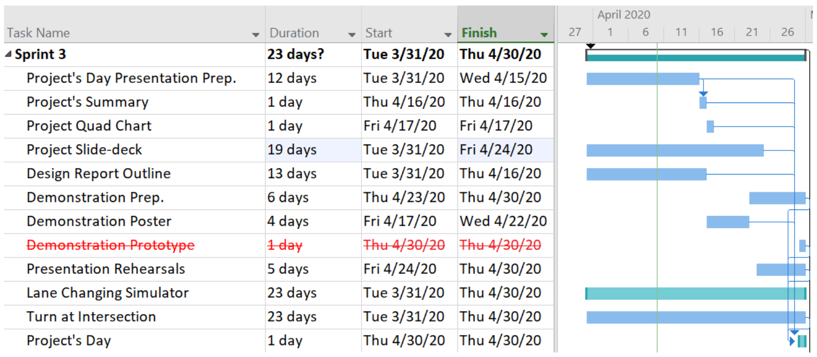
AVRAD is comprised of 10 future Second Lieutenants hailing from 6 technological disciplines offered by 3 academic departments. Each member of the team had a specific role in which he contributed to the success of the overall project by mastering a line of effort that corresponded to one or many of the design requirements of the vehicle (ref. Table 1). The team itself organized around two working sub-teams, each led by a sub-team captain who reported to a project manager who coordinated the efforts of the sub-teams. The sub-teams consisted of an administrative management team responsible for scheduling, inventory, sponsor communications, logistics, and budgeting, and a development team responsible for programming, sensor integration and troubleshooting, network management, and modeling, simulation, design, and testing of the vehicle and its various controllers.

Table 1: Team Composition by Line of Effort

NAME	DISCIPLINE	LINE OF EFFORT
Will Anderson	Computer Science	Computer vision and camera integration
Reed Burton	Computer Science	Code development, programming, circuit design
Preston Draelos	Mechanical Engineering	Vehicle dynamics; modeling, simulation
JD Fassett	Information Technology	Network engineering; LiDAR integration
Ed Kang	Systems Engineering	Logistics and resource coordination
Christopher Little	Mechanical Engineering	Project manager / team captain
Sean Min	Systems Engineering	Budgeting and scheduling; sub-team leader
Austin Morock	Mechanical Engineering	Controls engineer; controller design, testing
Sam Pool	Electrical Engineering	Vehicle/state management; sub-team leader
Mitch Stiffler	Computer Science	Machine learning and image classification

1.4. Project Management Plan

AVRAD employed the AGILE process framework to manage the complex nature of the project. Each project objective was broken down to a monthlong 'sprint', wherein the team would focus on accomplishing a specific goal related to one of the performance capabilities related to the vehicle. Each monthlong sprint would conclude with a performance review, a team realignment, an objectives and goals assessment, and planning for the next month's sprint (ref. Figure 1). Every collaborative work session began and ended with a 'scrum', a 10-15 minute session where each member of the team briefed his teammates on his progress since the last work session and received information and guidance from the project manager and sub-team leaders (ref. Figure 2). The AGILE process framework proved very effective for accomplishing a robotics research project. The sprint system facilitated the project goals being broken into manageable chunks and allowed for easy readjustment of goals if work was not being completed on schedule (due to advances or delays in progress). The daily scrums facilitated quick dissemination of information and guidance on tasks and deadlines, and helped hold each teammate accountable by forcing him to publicly recount his progress since the previous session to the entire team. A downside of the AGILE process is that it can become easy to lose focus by reducing the scope of a project into more digestible tasks without an involved project manager and a dedicated team member in charge of scheduling and task tracking. AVRAD mitigated this hurdle by having a dedicated administrative management team to focus and course-correct the work plans for the individual team members to



Task Name	Duration	Start	Finish
Sprint 3	**23 days?**	**Tue 3/31/20**	**Thu 4/30/20**
Project's Day Presentation Prep.	12 days	Tue 3/31/20	Wed 4/15/20
Project's Summary	1 day	Thu 4/16/20	Thu 4/16/20
Project Quad Chart	1 day	Fri 4/17/20	Fri 4/17/20
Project Slide-deck	19 days	Tue 3/31/20	Fri 4/24/20
Design Report Outline	13 days	Tue 3/31/20	Thu 4/16/20
Demonstration Prep.	6 days	Thu 4/23/20	Thu 4/30/20
Demonstration Poster	4 days	Fri 4/17/20	Wed 4/22/20
~~Demonstration Prototype~~	~~1 day~~	~~Thu 4/30/20~~	~~Thu 4/30/20~~
Presentation Rehearsals	5 days	Fri 4/24/20	Thu 4/30/20
Lane Changing Simulator	23 days	Tue 3/31/20	Thu 4/30/20
Turn at Intersection	23 days	Tue 3/31/20	Thu 4/30/20
Project's Day	1 day	Thu 4/30/20	Thu 4/30/20

ensure that the group was continually meeting deadlines.

Figure 1 (Left): Example of a sprint roadmap for the month of April 2020, outlining the events leading up to USMA Projects Day. This roadmap was adjusted, to include the cancellation of a live prototype demonstration, due to the unfortunate outbreak of COVID19.

Figure 2 (Right): The team conducts a daily scrum following a successful simulation demonstration of the vehicle's emergency stopping toolkit to the team captain, CDT Little. CDT Stiffler operates the computer while CDTs Draelos and Burton activate the emergency stopping package and CDT Morock photographs.

2. DESIGN PROCESS

2.1. Stakeholder Analysis

The team began our research by gathering information about autonomous vehicle technology, analyzing the IGVC and its past competitors, and interviewing the engineers who host the competition. Each member of the team conducted an individual review of the existing published literature revolving around autonomous vehicles, to include their mechanisms and algorithms for path planning, sensory perception, locomotion, and safety. The team then reviewed the rules of IGVC2020, as well as the design reports of previous competitor universities, to gather a working understanding of the competition's proceedings. Using this base of shared knowledge, AVRAD drafted a prioritized list of the functional capabilities and customer requirements that the vehicle must fulfill and a desired specification list for the system (ref. Table 2). The team then sought feedback from Mr. Lane, and then the engineers at CCDC-GVSC, on these priorities. Based on this analysis, AVRAD concluded that the overarching priorities for the vehicle were to qualify and score at IGVC, followed by the ability to integrate any designed software tools into the RTK architecture. These priorities informed the solution approach and component design of the system.

Table 2: Key Design Capabilities for IGVC Qualification

E-Stop connectivity out to 100ft.	Stop within 50ft of detecting stop sign.
E-Stop within 30ft after button is pressed.	Remain centered within a standard width 8ft lane.
Velocity control between 1-5mph.	Stop or change lanes within 50ft of detecting obstacle.

2.2. Engineering Standards

Throughout the duration of the project, the team adhered to the various standards governing the various engineering disciplines involved. The American Society of Mechanical Engineers (ASME) codes for mechanical design, and the ISO9000 principles of quality management were adhered to during all stages of development of the final product. The team placed great emphasis on continually maintaining contact and a healthy relationship with our primary customer, Mr. Jerry

Lane. AVRAD regularly sought his feedback, guidance, and leadership on all decisions that may affect the outcome of the final product to be delivered to him at IGVC in June 2020. Our approach to solving the urban navigation problem was process-based, rooted in the mechanical system design process embodied by all USMA accredited engineering disciplines. Our design decisions were informed with proper background research and literature review, consolidated academic evidence, and in-depth engineering analysis before prudent judgement was applied.

2.3. Project Constraints

The most pressing constraint placed on the completion of the project was time. Given that IGVC was scheduled for June 2020, and that work on the project began when the involved cadets were enrolled in Mechanical System Design on 19 August 2019, the cadets had a rigid framework by which to abide for scheduling manpower and allocating resources. The budgetary constraints of the project, discussed in Section 9 "Cost Assessment", were negligible due to the more than ample $30000 budget granted to the cadets by GVSC.

Key constraints to testing were dictated by the weather on and around garrison at West Point, NY. Given that the test plan, discussed in Section 8 "Test Plan and Performance Assessment", required ample road testing, the project's success could be altered by the severity of weather. The presence of snow on the ground severely inhibited the team from conducting testing during the winter months, as did the severe wind chill and its effect on delicate computer components if exposed for great lengths.

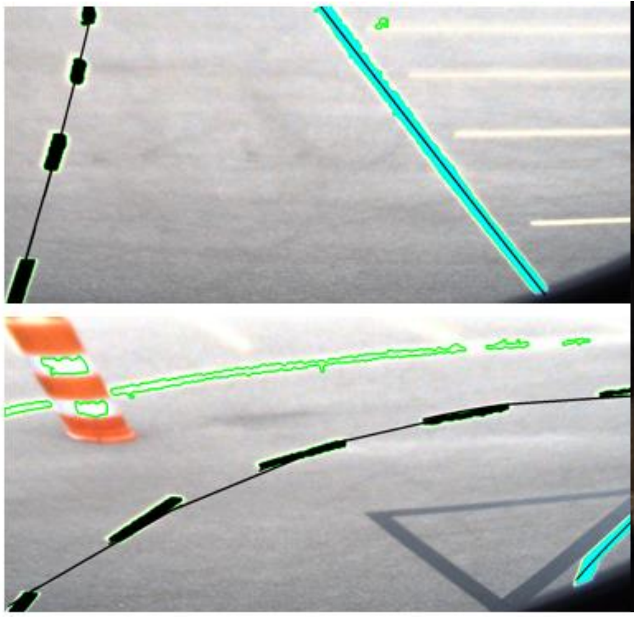
2.4. Design Constraints and Key Assumptions

IGVC specifies several design constraints by which all Self-Drive competitors must abide. Every submission must employ a battery electric vehicle: the vehicle must be electric, and can have no outboard power connection requirements. The vehicle must have exactly 4 wheels, in contact with the ground at all times. The vehicle must carry 2 passengers in a side-by-side configuration, cannot exceed 1500 pounds empty weight (passengers are not included in the weighing), and its maximum profile could not exceed 115" x 60" x 75" (length x width x height). The vehicle must maintain a speed between 1-5 mph. The vehicle needed emergency stop buttons located in the cabin, on each side, in the rear, and on a wireless key fob that could connect out to 100 feet. Every emergency stop button also needed to activate a safety light, which indicated whether the vehicle was operating in autonomous or manual mode.

AVRAD made several key assumptions in modeling and designing the vehicle. The first was that of a constant lane width of 8 feet. This is the standard lane width on the IGVC road course, and making this assumption simplified the calculations necessary to detect and follow lane lines. The team also assumed the vehicle could be modeled as a bicycle. The 'bike model', meaning that the vehicle's wheelbase could be reduced to a single front and back wheel centered on the vehicle midplane, drastically simplified the dynamic equations of motion utilized to design the vehicle's steering controller. This model will be further discussed in depth, and its validity will be evaluated, along with other key vehicle dynamics assumptions in Section 3.3 "Mechanical Design: Suspension and Steering".

2.5. Innovations

The team employed and developed several technological innovations. The use of computer vision and machine learning algorithms especially enabled the complex computing tasks required for a robotic system to detect and classify objects within images. The ability to capture, mask, and deconstruct images, then feed their constituent data components to convolutional neural networks and image processors allowed the ability to detect and classify objects and traffic signals for the uses of following lane lines (ref. Figure 3) and braking in response to stop signs (ref. Figure 4). The use of tools which allow environmental ranging and reconstruction aided the vehicle in perceiving its environment to ascertain the presence of both static and dynamic obstacles. Without these tools, the team would be unable to program the vehicle to make reactionary decisions to overcome its preselected path.



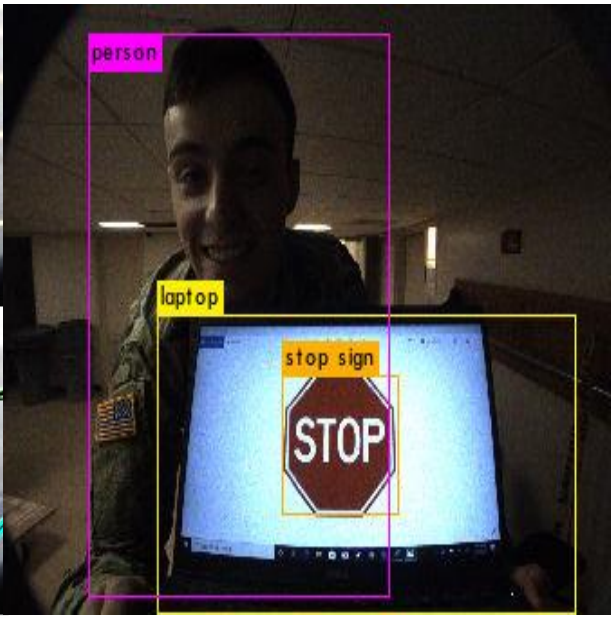


Figure 3 (Left): CDT Anderson's originally designed lane detector, which utilizes computer vision to mask and transform an image so that the GEM e2 can perceive lane lines.
Figure 4 (Right): CDT Stiffler demonstrates the image classifier he designed using machine learning algorithms to train a neural network to recognize everyday objects.

3. MECHANICAL DESIGN

3.1. Overview

Based on the design constraints and key function requirements that the vehicle must fulfill, the team chose to utilize a Polaris GEM e2 electric vehicle as its system platform. This vehicle is battery powered, sits in a 2 passenger side-by-side configuration, and has a gross empty weight of 1200 pounds. Its dimensions, 103" x 55.5" x 73", are within those specified by the competition guidelines. The GEM e2 has a top speed of 25 mph, so it is capable of meeting the 1-5 mph speed requirements of the road course.

3.2. Structural Design and System Configuration

The team had several sensors available for use in configuring the GEM e2 for success on the road course: a Velodyne HLD-64E Light Detection and Ranging (LiDAR) device, high resolution Mako cameras, a Global Positioning System (GPS), an Inertial Measurement Unit

(IMU), an Autonomous Stuff Spectra computer outfitted with an NVIDIA graphics processor, wheel encoders, and an odometer. Of these, AVRAD chose to utilize the LiDAR, cameras, GPS, IMU, and computer to accomplish the IGVC qualification requirements. The 'top-level' system configuration was achieved by mounting the cameras and LiDAR to a rack on the roof of the GEM e2 and wiring them through the roof to the Spectra, seated between the two passenger seats in the cabin of the GEM e2 (ref. Figure 5).

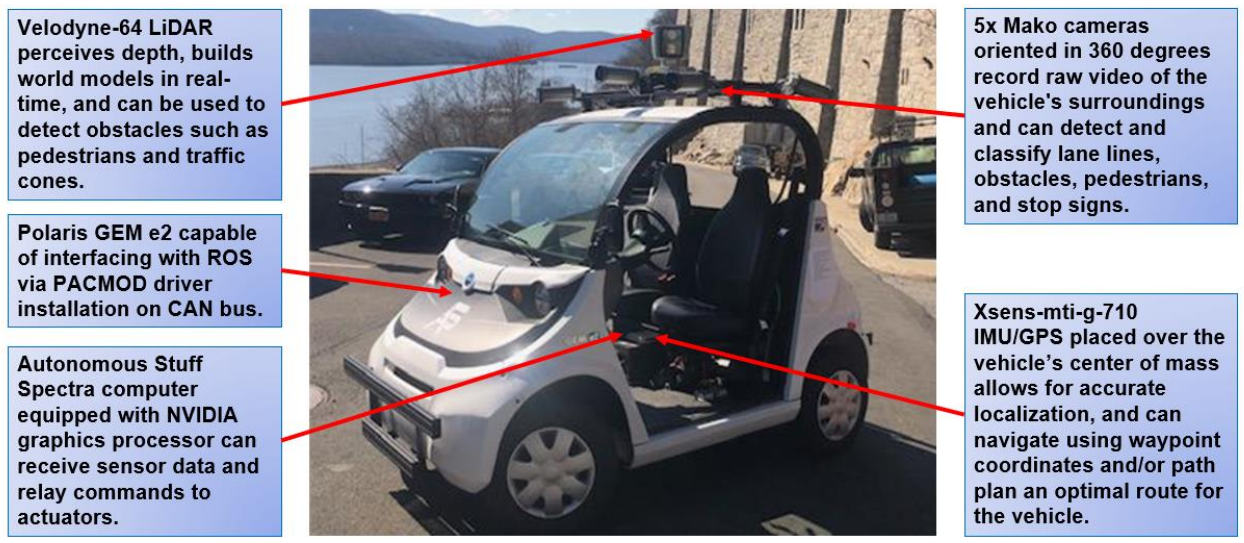

Velodyne-64 LiDAR perceives depth, builds world models in real-time, and can be used to detect obstacles such as pedestrians and traffic cones.
Polaris GEM e2 capable of interfacing with ROS via PACMOD driver installation on CAN bus.
Autonomous Stuff Spectra computer equipped with NVIDIA graphics processor can receive sensor data and relay commands to actuators.
5x Mako cameras oriented in 360 degrees record raw video of the vehicle's surroundings and can detect and classify lane lines, obstacles, pedestrians, and stop signs.
Xsens-mti-g-710 IMU/GPS placed over the vehicle's center of mass allows for accurate localization, and can navigate using waypoint coordinates and/or path plan an optimal route for the vehicle.

Figure 5: AVRAD's GEM e2, outfitted with the various sensors that the system utilizes to accomplish urban navigation.

3.3. Suspension and Steering

The Polaris GEM e2 utilizes a MacPherson strut suspension for its front driving and steering axle, and employs an independent trailing arm on its rear axle. This suspension combination, when coupled with the GEM e2's Ackermann differential, are designed to minimize both tire scrub during tight turns and resistive steering torque at low speeds while simultaneously maximizing the contact surface area between the vehicle's tires and the contour of the road. These qualities make the GEM e2 an ideal platform for the road course at IGVC, which features 90-degree turns at small radii of curvature and has a prescribed 5 mph speed limit.

The MacPherson strut and trailing arm combination allows the vehicle to carry a substantial payload with minimal shock deflection for various changes in payload configuration. This means the vehicle rides with a relatively constant clearance from the ground for a variety of weight ranges. Given that only a 300 pound difference separates the empty and maximum competition weights of the GEM e2, the team assumed that the distribution of weight forces across each of the GEM e2's tires was constant and equal for the duration of its operation.

An Ackermann differential (ref. Figure 6) minimizes positive scrub and maximizes tire contact during tight turns by allowing each tire to traverse the radius of the turn at a different rotational speed and angle relative to the vehicle's heading within the lane. This quality makes the GEM e2 a masterful vehicle for navigating extremely tight turns, with radii of curvature as low as 150 inches. The team assumed that the GEM e2's tires would in turn undergo a 'no-slip' condition during operation, which allowed AVRAD to neglect tire slip when evaluating the vehicle's motion.

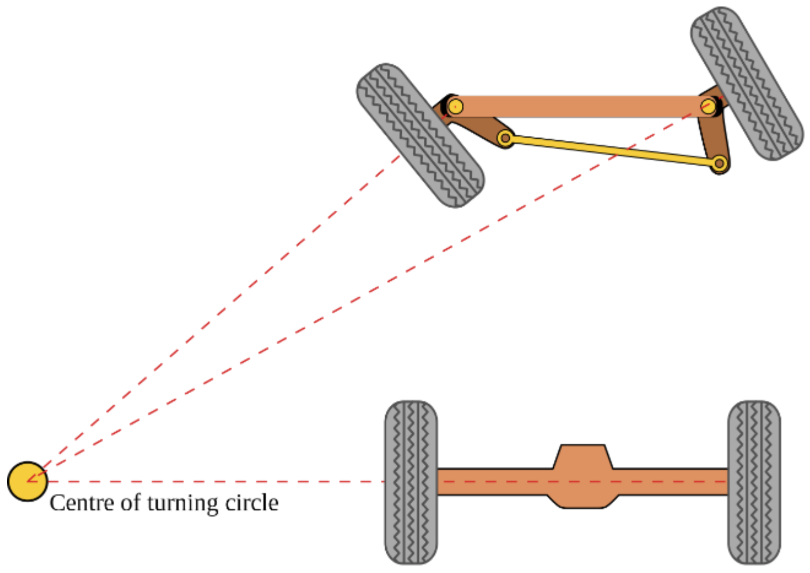


Figure 6: Image adapted from *Wikipedia, the Free Encyclopedia* illustrating the turn geometry of a vehicle chassis outfitted with Ackermann steering.

A downside of the Ackermann steer component is the complexity it introduces when attempting to model and analyze the GEM e2's dynamic equations of motion to design an appropriate steering controller. The multiple wheel angles enabled by the Ackermann steering geometry introduce multiple additional degrees of freedom to these equations. To simplify our analysis of the vehicle by reducing these wheel angles to a single degree of freedom, the team employed the 'bike model' (ref. Figures 12 and 13, Section 5.3.2) to develop a simulation, and later a steering controller, for the vehicle. By condensing the vehicle's wheelbase to a singular wheel model centered midplane on the vehicle's front axle, the lane following team only needed to compute a single steering angle to adjust the vehicle's orientation. The 'bike model' assumption is perfectly valid for the work performed by the team, because the low speed of the vehicle reduces the error associated with neglecting an extra degree of freedom within these steering calculations.

The steering and suspension components of the GEM e2, accompanied by its relatively slow speed on the IGVC road course, allowed the team to make the assumptions necessary to develop simplified equations of motion for the vehicle and begin work on designing a robust lane following controller, to be discussed in further detail in Section 5.3.2 "Software Design: Lane Detection and Following".

3.4. Weatherproofing

The sensors of the AVRAD GEM e2 are protected from environmental conditions, especially precipitation, through a number of means. The sensors and computer housed within the cabin of the vehicle are protected simply by mounting doors to the chassis. Each Mako camera is sealed within a watertight housing designed for underwater use. The Velodyne LiDAR is designed in part for use on maritime vehicles, and its product specifications indicate that it has a substantial tolerance for moisture and precipitation. The GEM e2 is not capable of handling severe winter conditions. Its front wheel drive and road tires would not fare well in icy conditions, and the lane detection and stop sign detection software developed by the team, discussed in Section 5.3 "Software Design: Autonomous System Design", would not work effectively in the presence of snowfall or other line of sight inhibitors.

4. ELECTRONIC DESIGN

4.1. Overview

The AVRAD system employs an Autonomous Stuff drive-by-wire kit, the Platform Actuation Control Module (PACMOD), to bridge the gap between its various sensors, their governing computer system, and the GEM e2's Controller Area Network (CAN-bus) which controls its actuators and physical automotive systems. The team also employed a variety of switches, relays, network hubs, and Arduino microcontrollers to complete the on-board network for the vehicle.

4.2. Power Distribution and Sensor Integration

The AVRAD GEM e2 utilizes a 72 volt battery consisting of six 12 volt cells, which were routed to a switch hub that provides power directly to the Autonomous Stuff Spectra computer and the 5 Mako cameras via an ethernet connection cable. The Autonomous Stuff Spectra in turn routes power to the remainder of the system (ref. Figure 7).

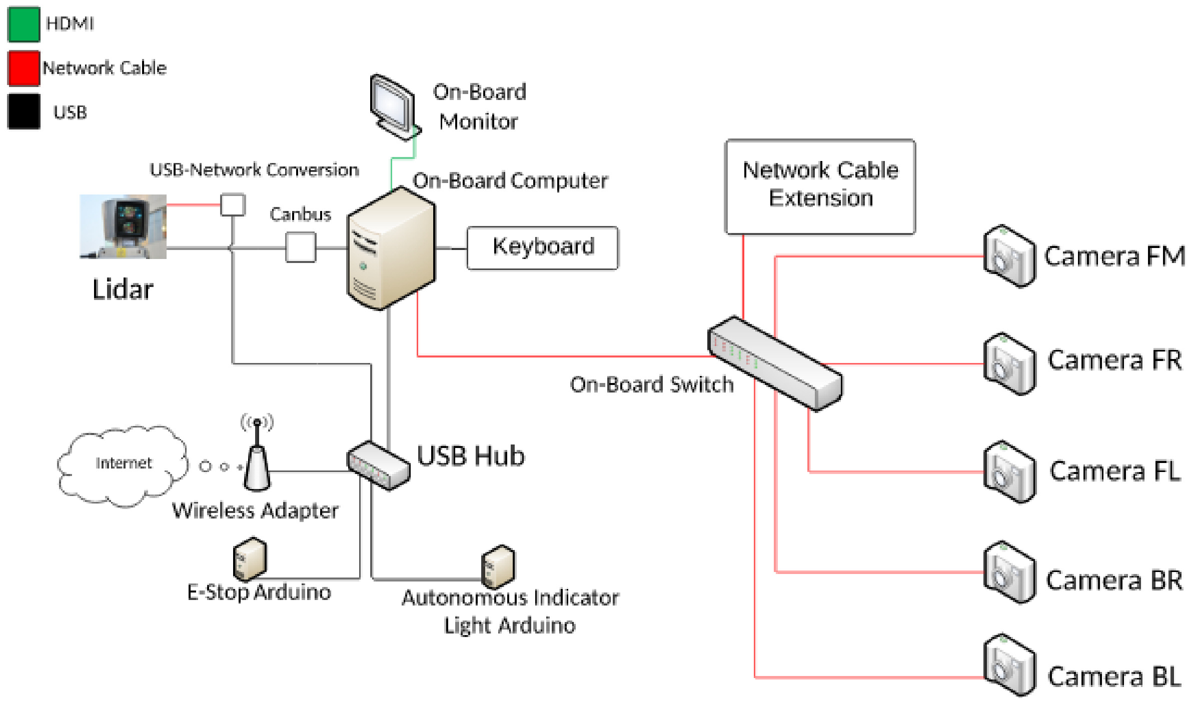


Figure 7: Wiring schematic showing the network configuration of the AVRAD GEM e2 and the connector cable used. Red denotes an ethernet cable, which transfers both power and data between the computer, sensor, and switch hub.

4.3. Safety Devices

The vehicle's primary safety devices are the emergency stopping kits the team created to fulfill the IGVC road course safety standards. The team wired 4 mechanical emergency stop buttons, one in the cabin of the vehicle, one on each side, and one in the rear, to an Arduino microcontroller which interacts with PACMOD, and subsequently the vehicle's actuators, to publish a brake command via the Robot Operating System (ROS) once a stop is initiated (ref. Figure 8). The software element of the emergency stop toolkit is discussed in further depth in Section 5.2 "Vehicle and State Management". The team also configured the Arduino microcontroller to receive a signal from a wireless key fob and publish a logical 'stop' value if the key fob is activated. The first Arduino connects to a second microcontroller which operates the vehicle's safety light and indicates the current operating state of the GEM e2.

Once pressed, each of the emergency stops causes the vehicle to execute a series of commands: the safety light activates, PACMOD activates the brake to bring the vehicle to a controlled halt, and control of all vehicle processes is returned to the driver.

Polaris also designed the GEM e2 with a proprietary kill switch in the cabin of the vehicle that kills all on-board processes and shuts off the power to all vehicle systems entirely. As a redundancy, AVRAD has tested the functionality of this kill switch to shut off autonomous processes, however this kill switch is utilized as a last resort due to the inability of the driver to control the GEM e2 after the kill sequence occurs. While this feature is valuable for protecting pedestrians in the event of a failure of the GEM e2's drive-by-wire kit, rendering a moving vehicle entirely powerless in heavy traffic is not desirable as this could result in severe harm or damage.

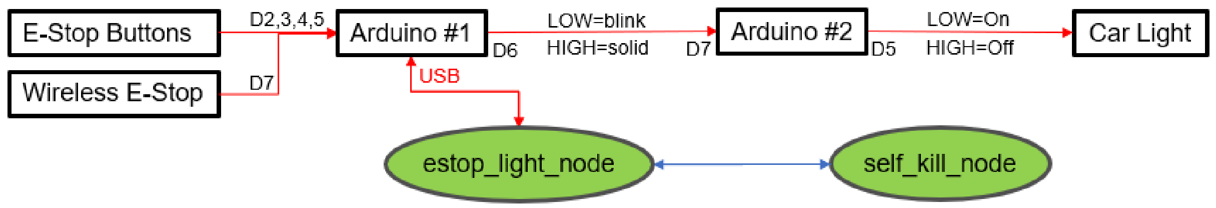


Figure 8: Diagram mapping the functionality of the AVRAD emergency stop toolkit. If an emergency stop button is depressed, a signal is sent to an Arduino microcontroller, which activates the software commands to kill the vehicle.

5. SOFTWARE DESIGN

5.1. Overview

The team built the software architecture for the vehicle within ROS, the same robotics middleware platform employed within the architecture of RTK. ROS is specifically designed to allow robotic integration and control of a multitude of sensors, and utilization of the same middleware would facilitate easy interoperability between the packages written by AVRAD and the packages already available to the team via the CCDC-GVSC repository.

5.2. Vehicle and State Management

The overarching architecture of the vehicle's software structure is nested within a vehicle management system housed within ROS (ref. Figure 9). A vehicle management node interacts with the user interface and manual controls of the system and publishes a state machine reflecting the various desired operating states of the vehicle: 'manual drive', 'teleoperation', 'autonomous drive', or 'off'. This vehicle manager node then relays information to and from PACMOD and a 'Self-Drive' package which contains the autonomous control system for the vehicle. At all times, the vehicle management node is pinging the emergency stopping node, checking for a signal indicating that the self-kill node should be activated in response to one of the emergency stop buttons having been pressed.

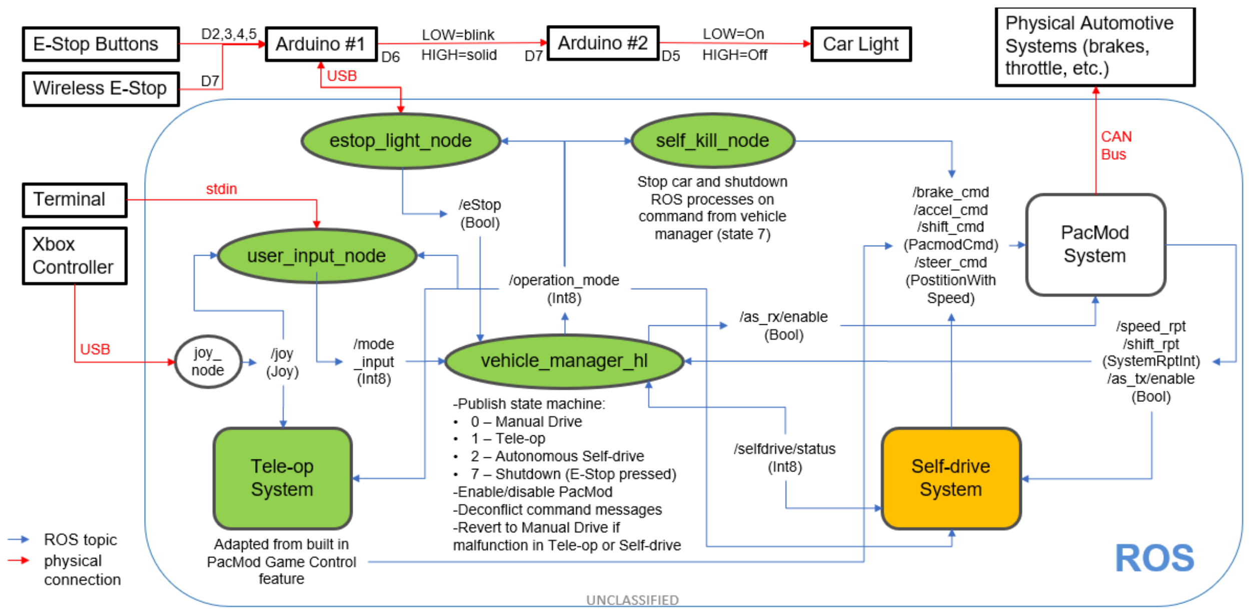


Figure 9: Software diagram mapping the vehicle management architecture comprising the high-level software system.

5.3. Autonomous System Design

Within the vehicle manager architecture, AVRAD designed a ROS package to conduct the various autonomous functions required by IGVC (ref. Figure 10). This autonomous system interacts with the various sensors to receive and process their data, allowing the GEM e2 to build a fundamental understanding of its proximate environment and make path decisions to avoid obstacles and attain the shortest distance route while obeying traffic laws and complying with desired user input commands.

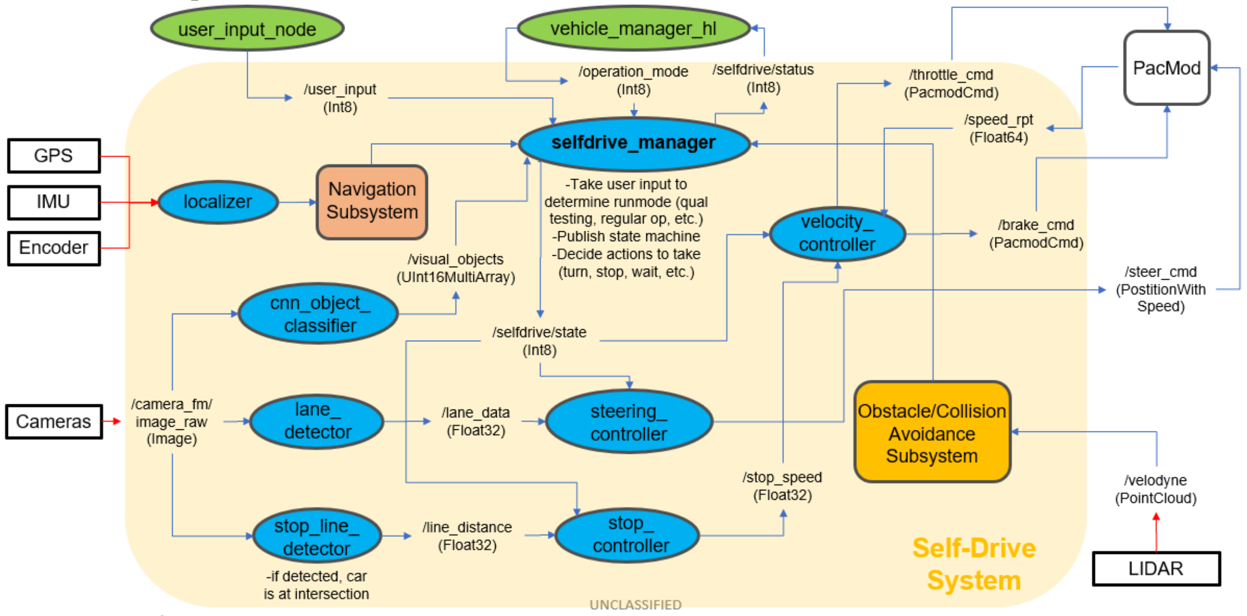


Figure 10: Software diagram mapping the GEM e2's autonomous 'Self-Drive' system. This system accomplishes the actual perception, locomotion, and navigation tasks while the vehicle is operating in autonomous mode.

5.3.1. Velocity and Brake Control

To accomplish maintaining the published speed limit on the IGVC road course, the team designed ROS node containing a basic proportional velocity controller. This node receives the desired user input speed (in this case 5mph for the competition constraint) and the actual vehicle speed from feedback published by PACMOD, calculates the difference between the two speeds, then computes and publishes an accelerator or brake command to PACMOD to correct for this difference.

5.3.2. Lane Detection and Following

To accomplish lane following, the team first accomplished lane detection. The team weighed several various alternatives (ref. Table 3), ultimately deciding to design our own lane detection software due to the inability of existing solutions to quickly process signal noise at slow speeds and the tendency of existing solutions to underperform when handling tight turns. This lane detector would consist of a custom image pipeline, dubbed the Close Quarters Lane Finder (CQLF).

Table 3: Lane Detector Alternatives Considered by AVRAD

Detection Alternative	Operating Principles
Udacity Histogram Algorithm	Open-source software that fits a probability distribution to visible lanes, transforms image to top-down view, and then estimates curvature based on lines of best fit equations.
DAVE2 Neural Network	Neural network specifically trained to detect images of lanes and map the correlation between the trajectory of the lane and a probable human steering response.
Custom Image Pipeline (“Close Quarters Lane Finder – CQLF”)	Raw image masked to find white pixels, then fit curvature equations to the lane lines which allows the vehicle to calculate its distance from center-line.

CQLF utilizes the front right Mako camera to capture a raw image of the lane, then transforms and masks the image to separate the black and white pixels representing the lane line drawn on asphalt. Using the known width of the vehicle and the assumption of a standard 8 foot wide lane, CQLF calculates the vehicle’s offset from the center of the lane based on its orthogonal distance from the rightmost lane line (ref. Figure 11). CQLF publishes this offset from the center of the lane, the cross-track error, to a controller designed by the team to steer the vehicle. By correcting for and minimizing this cross-track error, the vehicle centers itself within the lane and follows it at a relatively constant velocity.

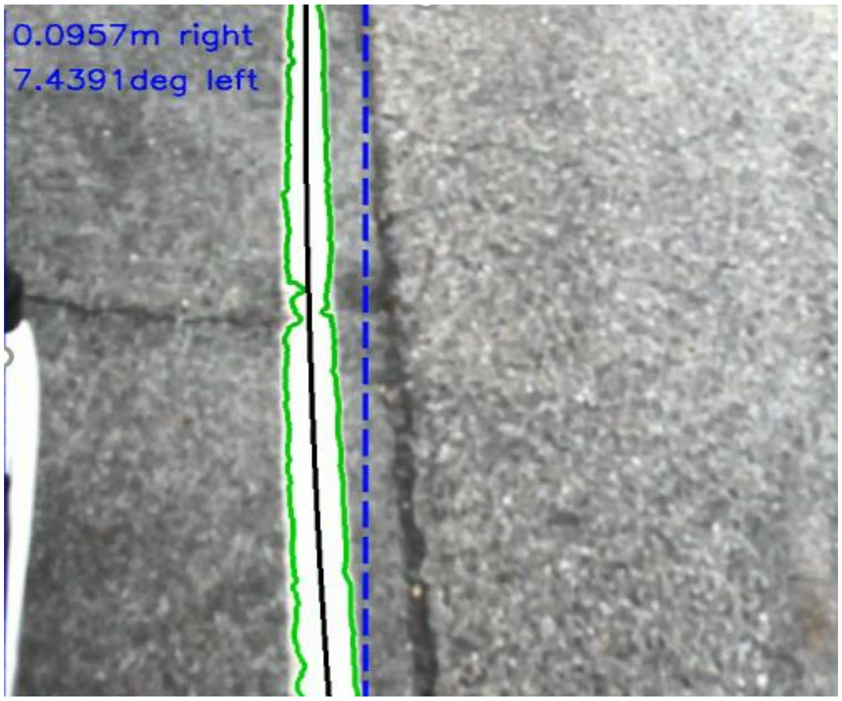


Figure 11: CQLF highlighting the detected rightmost lane line, the calculated trajectory of the lane (shown in blue), and the orthogonal distance from the lane line and necessary steering angle correction for PACMOD (top left).

Design of the lane following controller began by developing equations of motion for the GEM e2 based on its governing dynamics as it follows a lane (ref. Figure 12). As the vehicle follows a lane it encounters two sources of 'error', or difference between the actual and desired position and trajectory within the lane. The first is the cross-track error, or offset from the center of the lane, y. The second is the difference between the vehicle's heading Ψ and steering angle Ω.

$$\boxed{\begin{gathered} \boldsymbol{V = 5mph} \\ \boldsymbol{\dot{y} = -\sin(\psi - \theta)\sqrt{\dot{V}_y^2 + \dot{V}_x^2}} \\ \boldsymbol{y_D = \dot{y}_D = 0} \end{gathered}}$$

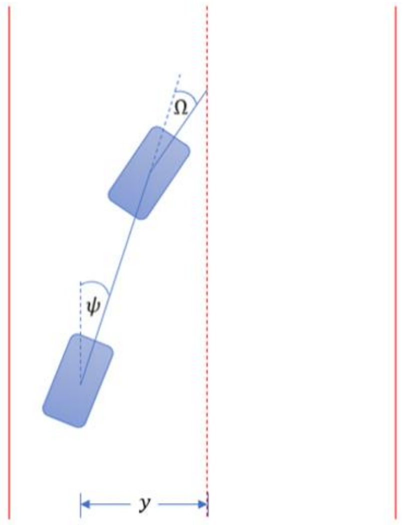


Figure 12: Diagram depicting the condensed wheelbase of the GEM e2 as it follows a lane (solid red) along with its steering angle, heading within the lane, and offset from the center of the lane (dashed red). The equations of motion on the left describe the vehicle as it moves down the lane: for any constant straight line speed of 5 mph, the vehicle changes cross-track error at a rate defined by its heading and velocity components. The desired cross-track error, and its corresponding rate of change, is set equal to 0 to keep the vehicle centered within the lane.

Over time, the team needed to control these two errors by manipulating the vehicle's motion to minimize them. Given that the vehicle's velocity for the competition is relatively constant, the only motion input that the team could control is the vehicle's steering angle. As the team developed the final equations of motion for the vehicle and began to explore the design of a controller, it became apparent that the team could employ the 'bike model' to simplify the analysis and design of the controller due to the vehicle's low speed and large margin for overshoot within the lane (ref. Figure 13).

$$\Delta\hat{e}_{x,f}{}' = \Delta n_f \cos(\Omega)$$
$$\Delta\hat{e}_{y,f}{}' = \Delta n_f \sin(\Omega)$$
$$\Delta\Psi = \frac{V}{L}(\Omega)$$
$$\Delta n_f = V\,\Delta t$$

$$L = Wheelbase\ Length$$
$$\Omega = Steering\ Angle$$
$$\psi = Heading\ Angle$$
$$y = Cross\ Track\ Error$$

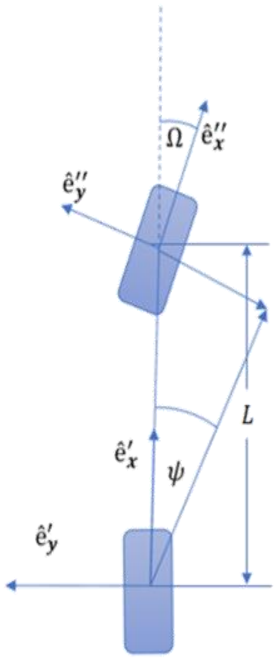


Figure 13 (Right): Diagram showing the bike model in context to the reference plane at the road's surface, alongside the initial control equations used to build the lane follower.

The team worked to develop a control law which could effectively allow the vehicle to achieve zero residual error in its motion along the path of a lane. The team began by using proportional control and evaluating the vehicle's response. Though proportional control effectively minimized the residual steady state cross-track error within the lane following system, this method of control was unstable (ref. Figure 14). The controller quickly centered the vehicle within the lane but contantly overshot the zero error cross-track line because there was a significant difference between the direction of the lane, the vehicle's steering angle, and its heading angle. As such the vehicle required continuous and drastic corrections to adjust for its previous steering responses. This type of instability is common in proportional controllers, as the correction from the vehicle is gauged by the magnitude of the error. By adding a derivative term to the controller, the team was able to drastically increase the system stability by decreasing the rate of cross-track error accumulation to 0. The addition of this derivative term was the minimum necessary requirement to bring the poles of the system into the stable region (ref. Figure 15). Proportional-derivative (PD) control increases the system stability by allowing the vehicle to reduce its cross-track error rate to zero: as the vehicle traverses the lane, over a series of step responses, the rate of error accumulation reaches zero and it requires increasingly less drastic steering responses to correct for increasingly smaller overshoots in heading error.

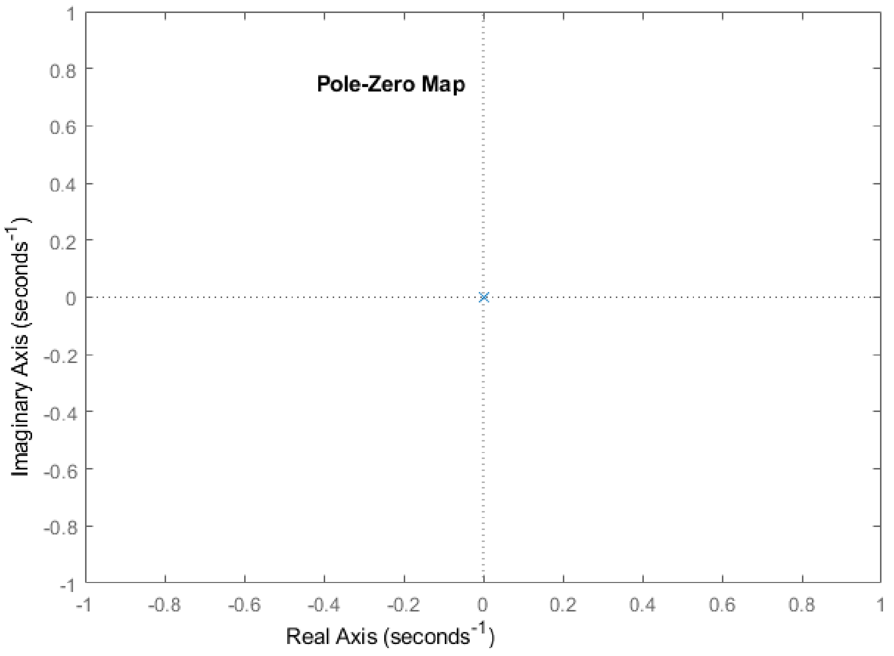


Figure 14 (Left): Pole-Zero Map showing the roots of the system using just proportional c ontrol. Here, the system is entirely unstable, and requires adding a derivative term to bring its roots into the stable region.

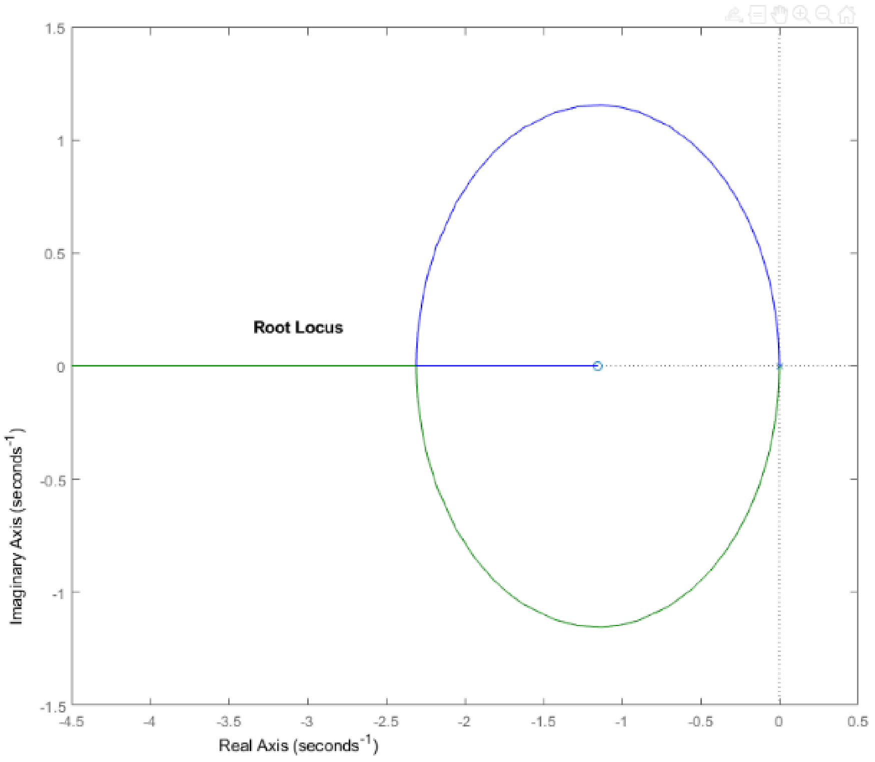


Figure 15 (Right): The root locus of the PD controller brought the poles of the system into the negative region of the real axis, indicating that the system response was stable .

AVRAD developed the closed-loop transfer function for the system to determine the system gains and begin tuning the controller (ref. Figure 16). Within a MATLAB simulation environment, as discussed in Section 7 “Simulations Employed”, the team scaled the gains of the controller and plotted the step response of the vehicle as it navigated from an arbitrary origin position, velocity, heading, and offset distance (ref. Figure 17). Based on these results, the team was able to adjust the controller to achieve a final maximum overshoot of 20% with residual error reaching zero at approximately 3 seconds. This means that, in context of an IGVC standard lane, the vehicle would never stray further than 1.77 feet from the center of the lane, finding and settling at the center of the lane approximately 3 seconds after motion is initiated. Integrating the controller into the CQLF ROS package produced a high-fidelity lane following solution which, when run on the Autonomous Stuff Spectra’s NVIDIA graphics processor using a live video feed from the right Mako camera, was able to detect the lane line and issue a steering correction at a rate of 12 Hz.

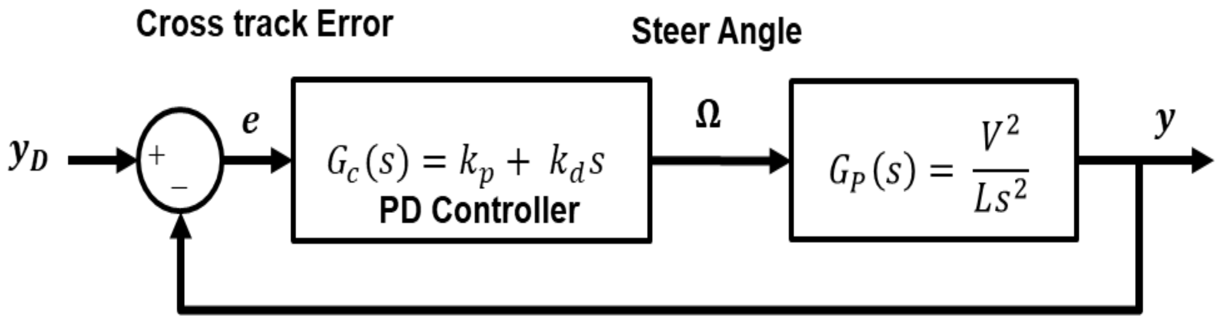


Figure 16: Block diagram showing AVRAD’s closed-loop feedback PD controller and associated transfer functions.

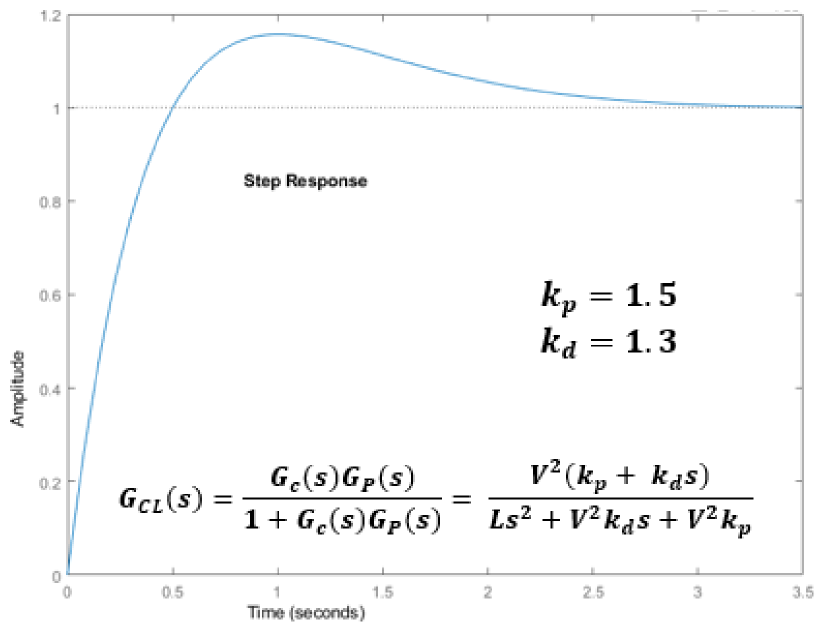


Figure 17: Step response in MATLAB showing the completed lane follower alongside its full governing control law and final gains after tuning.

5.3.3. Stop Sign Detection and Reaction

To react to stop signs, the team first needed to design a means of detecting stop signs using the Mako cameras mounted on the GEM e2's roof. AVRAD began by analyzing a variety of image classification tools (ref. Table 4) and ultimately decided to utilize the Darknet Neural Network framework trained on a YoloV3 urban dataset due to its rapid detection speed, high accuracy, and minimal training time.

Table 4: Stop Sign Detector Alternatives Considered by AVRAD

Detection Alternative	Operating Principles
Faster-R CNN	Image is broken into regions and each is processed for features indicating assigned classes.
Darknet YoloV3	Bounding boxes with weights/probabilities assigned to a grid overlying the image. Each box is processed to detect features of probably objects within the image.
Darknet Yolo-tiny	Same as above. Grid is smaller, so the classifier runs faster at the cost of greater accuracy.
Custom Neural Network	The team would collect a dataset and train its own convolutional neural network with custom classes.
Retinanet	Image is deconstructed and processed by layers to form a 'feature map'.
Simplified "Red Pixel" Finder	The team would develop a program that simply scans for 'blobs' of red pixels to approximate a stop sign.

The Darknet Framework scans images and assigns bounding boxes to groups of pixels with similar properties, then uses weighted classes to predict possible objects contained within that bounding box based on the properties of those groups of pixels. The YoloV3 dataset contains weights and classes tailored to urban environments, and training the Darknet Framework on this dataset yields a solution which correctly outputs the presence of pedestrians, vehicles, obstructions, and stop signs (ref. Figure 18).

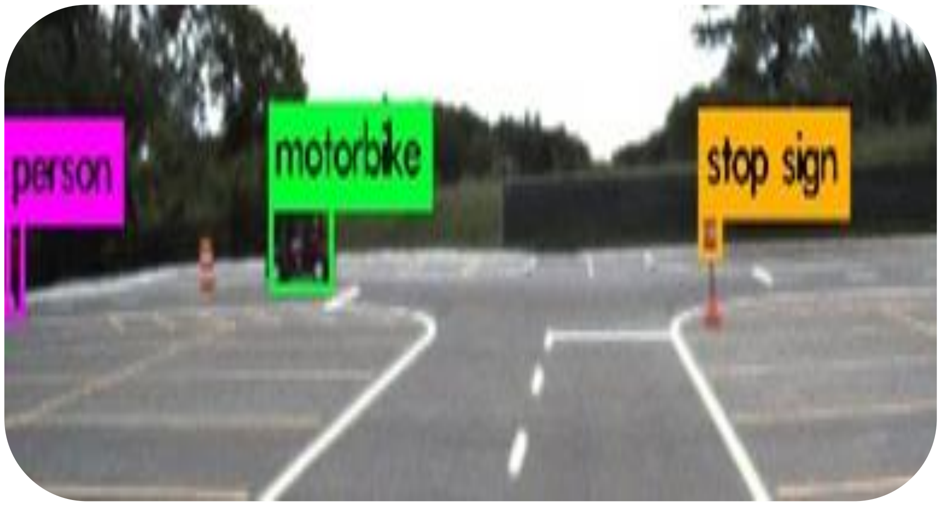


Figure 18: The AVRAD image classifier, after processing this camera image from IGVC 2019, correctly detected the presence of a stop sign in addition to a variety of traffic obstacles.

To design a reaction controller, AVRAD explored a variety of options (ref. Table 5) and decided to proceed with a fuzzy logic controller due to its ability to handle complex or uncertain inputs, its ease of programming, and its ability to produce a gentler stop for the operator. The fuzzy controller, unlike a PID-variant, deals with logical values that can be assigned to multiple 'memberships' that can be tailored to produce brake commands in PACMOD.

Table 5: Stop Sign Reaction Controller Alternatives Considered by AVRAD

Reaction Controller Alternative	Underlying Physics
Feedback Controller	PID control variant receives approach speed and distance and constantly updates brake value over distance until output speed is 0.
Fuzzy Logic Controller	Controller receives change in bounding box area and/or distance to stopping line from pixel change in camera image, then calculates logical value to publish brake command.
Pre-programmed Stop Command	ROS package calculates the approach speed and publishes a 'one and done' precalculated brake value to PACMOD.

The AVRAD fuzzy controller employs two inputs with various membership classes to calculate and publish a brake command to PACMOD (ref. Figure 19). The first input is calculated from the area of the bounding box, in pixels, of the stop sign as the vehicle approaches. The second input is calculated from the CQLF detecting the white line that demarks intersections on the IGVC road course and calculating its distance from the top of the image in pixels. Knowing the resolution of the cameras allowed the team to use trigonometric relationships to translate pixels into units of distance and area. Based on these distances and areas, AVRAD can calculate the approach distance, or the distance from the stop location to the vehicle at the moment of initial detection. Based on the changes in areas (in pixels) of these inputs, and the vehicle's approach speed, the fuzzy controller publishes a desired speed command to the velocity controller, which continually updates with feedback from the detection algorithm until the vehicle is at a halt.

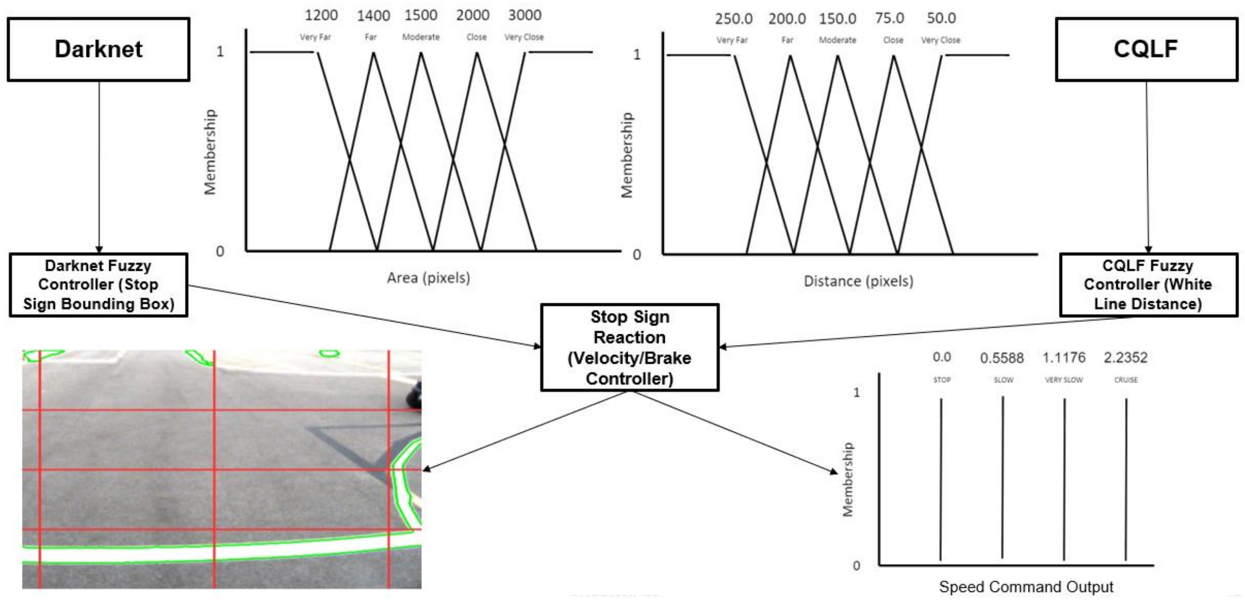


Figure 19: Diagram mapping the control input memberships and speed command response for stop sign reaction.

After training the network and employing the Compute Unified Device Architecture (CUDA) parallel processing tool in ROS to allow the detection algorithm to run in real time on the NVIDIA graphics processor, the team's entire stop sign detection and reaction controller was able to process images, classify a stop sign if present, and begin braking at a rate of 2 Hz. The final controller was tuned (ref. Figure 20) and adjusted to increase its robustness and produce a more gradual stop, similar in fashion to the way a human operator would apply a brake in a vehicle.

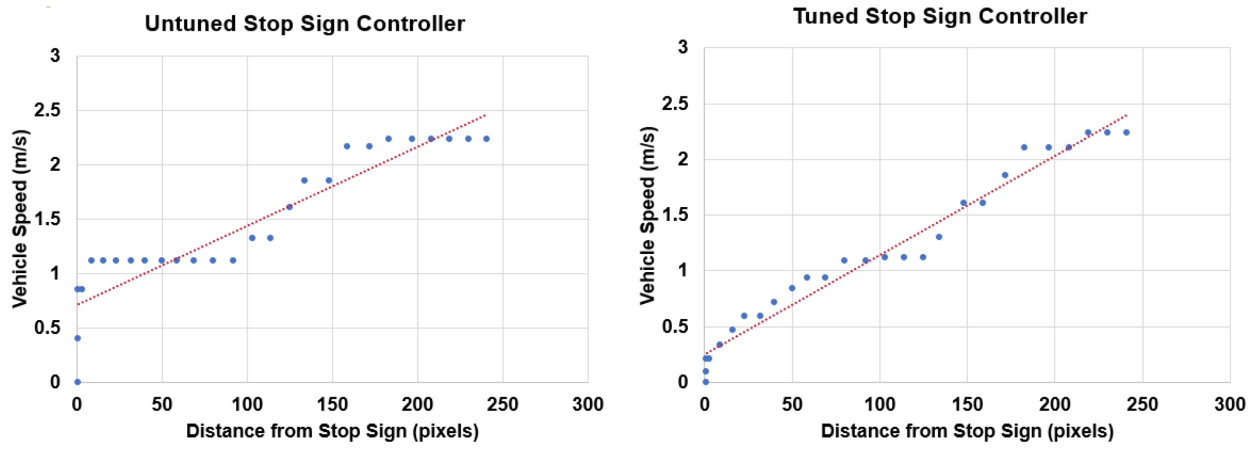


Figure 20: Tuning results for the stop sign controller. The relationship between the vehicle's speed and distance from a stop sign developed a linear correlation, representing a gradual application of the brake to bring the vehicle to a smoother, gentler stop.

5.3.4. Obstacle Detection and Avoidance

The team's obstacle detection and avoidance suite employs the Velodyne LiDAR to build a map of the vehicle's proximate surroundings to indicate the presence of inhibitors to the vehicle's path. Using an existing ROS package from the RTK architecture, the LiDAR reliably calculates the depth of these obstacles relative to the vehicle in terms of their spatial distance and direction. The team modified the CQLF to receive this information and calculate a new desired offset, which in turn forces the lane following controller to correct for this offset by merging lanes and thus avoiding the obstacle. Once the LiDAR no longer registers any inhibitor to the vehicle's forward motion, the lane offset is reset to match that which is necessary to center and follow the right-side lane of the road course. As of the writing of this design report, the team had developed the logic for these procedures, and modified and tested the updated lane follower in the MATLAB simulation environment, but due to the COVID19 outbreak and the subsequent switch to remote learning the team had not been able to assess the roadworthiness of these capabilities.

5.3.5. Map Generation, Goal Selection, and Path Planning

Due to the outbreak of COVID19, the team had to postpone all work on the navigation subsystem of the 'Self-Drive' package. The team's plan for this subsystem is to employ the GPS and IMU, along with localization and path generation algorithms available through RTK, to begin to design the vehicle's functional capability suite for determining its position, desired endpoint as dictated by the user, and the optimal route and the required decision tree to achieve that endpoint. Key to this subsystem is the development of logic capable of making decisions to navigate intersections. Once the stop sign controller has brought the vehicle to a halt, the navigation subsystem must reanalyze and reoptimize the route in terms of shortest distance or duration and subsequently publish a decision to the lane follower whether to turn left, turn right, or continue straight forward.

6. DESCRIPTION OF FAILURE MODES

The major failure modes for the AVRAD GEM e2 involve the possibility of a failure to conduct an emergency stop, and the possibility of system failure due to a compromised component. As discussed in Section 4.3 "Electronic Design: Safety Devices", the emergency stopping system is built with redundancy in mind, employing 2 Arduino microcontrollers and 5 total emergency stopping buttons, in addition to the kill switch designed into the GEM e2 by its manufacturer. The concept of component failure can be mitigated by constant monitoring of the health of all sensors and components in a vehicle health node within the ROS vehicle management system. This node could alert the user of possible disconnection or damage to a sensor or computing component if this were to occur.

7. SIMULATIONS EMPLOYED

To simulate the vehicle, the team employed two methods. First, the team utilized pre-recorded, or 'bagged', camera data which captured the vehicle's view of its surroundings and data representing human inputs. By testing our controllers on this bagged data, the team could debug the software and tune it until the outputs of the software most closely represent those expected by a skilled human driver.

The team also adapted a simulation environment in MATLAB in which the vehicle's steering controller was designed, tuned, and later adjusted to meet the criteria for obstacle avoidance (ref. Figure 21). This simulation environment provided AVRAD with the barebones tools for mathematically analyzing the vehicle without spending unnecessary time and manpower on developing a more robust or visually appealing simulation.

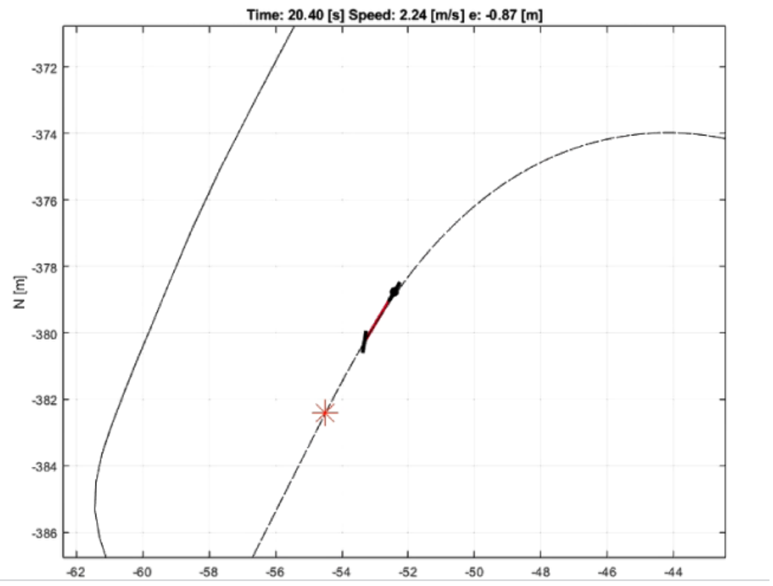


Figure 21: AVRAD MATLAB simulation environment, including the 'bike model' vehicle, an obstacle, the total run time, vehicle speed, cross-track error, and global coordinates.

8. TEST PLAN AND PERFORMANCE ASSESSMENT

The team's test plan was divided into three stages: simulation and debugging, validation, and tuning. As each software toolkit and its subsequent controller was designed, it was simulated on bagged camera data and within MATLAB to confirm that the software was mathematically correct and contained no major bugs or errors. Following the debugging phase, the controller was uploaded onto the vehicle computer and an initial road test was conducted to validate the results of the simulation, meaning that the controller could execute its desired function safely. Finally, after validation, the controllers were tuned, and road testing would occur to ensure that the vehicle could actually meet the performance specification necessary for IGVC (ref. Table 2).

As of the writing of this paper, the only controllers to be deemed fully road worthy were the velocity and brake controllers and the emergency stopping toolkit. The remainder of the controllers, most notably the stop sign reaction controller and the lane follower, were still in the tuning phase at the onset of the COVID19 outbreak and the subsequent switch to remote learning. While these controllers are in a debugged and testable state, they cannot be deemed fully roadworthy until a final in-person test is conducted to ensure that they meet the desired product specification from the tabulated design requirements.

9. COST ASSESSMENT

AVRAD has been graciously allotted an annual $30000 budget by GVSC, which to date has only been tapped for select maintenance actions and other general supplies related to the upkeep of the GEM e2 (ref. Figure 22). The largest portion of this budget has been dedicated to travel to Warren, Michigan for the physical judging of IGVC, but due to the outbreak of COVID19 and the subsequent mandates of social distancing, this portion of the budget has been absorbed back into the funds available for use. These funds will be stewarded and earmarked as necessary for keeping the GEM e2 properly maintained and updated with the most technologically advanced equipment as it becomes available for academic use.

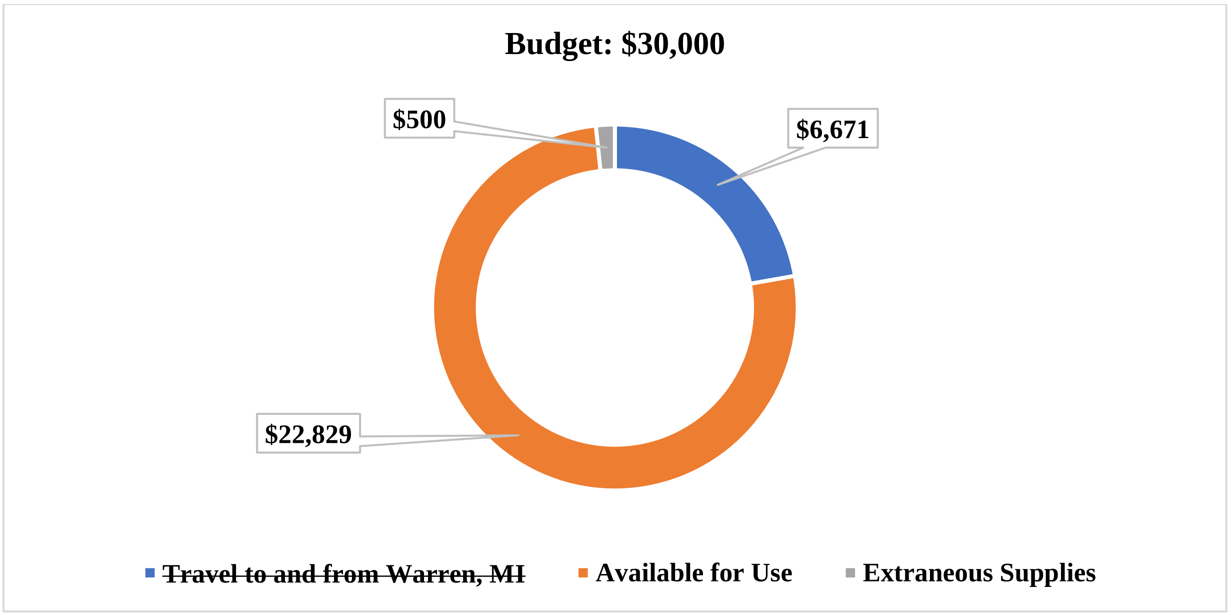
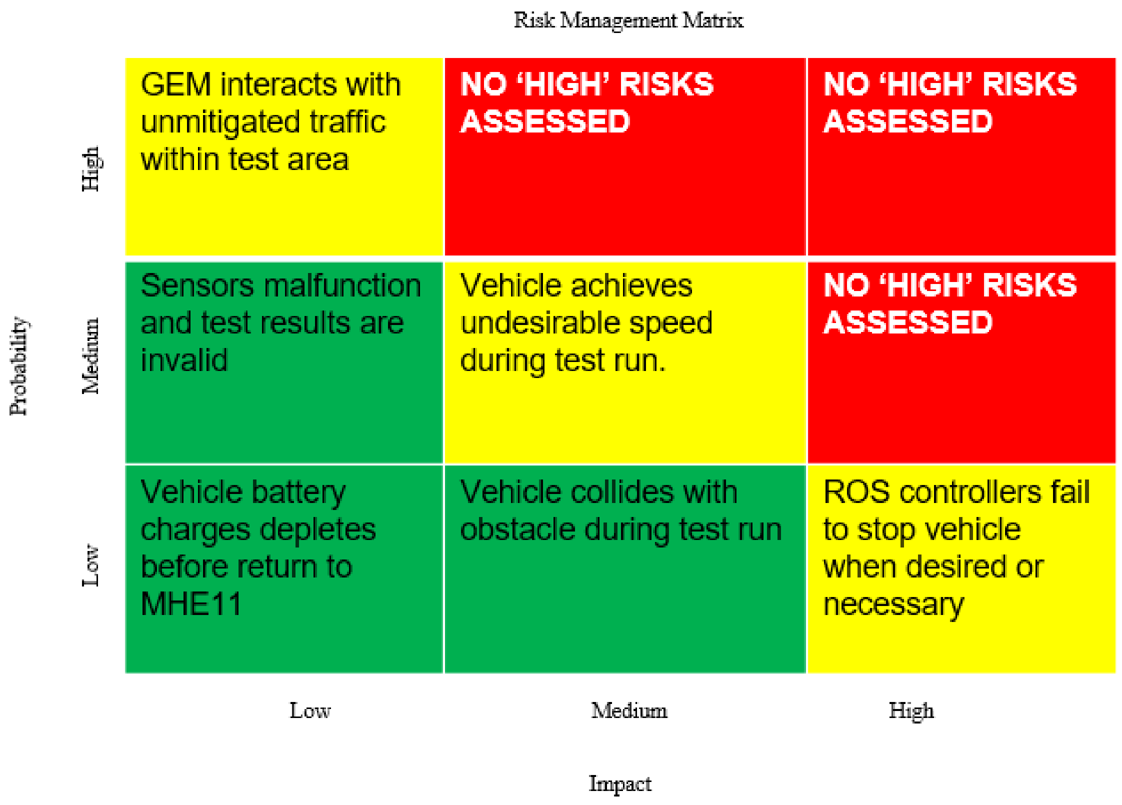


Figure 22: AVRAD budget for fiscal year 2020.

10. RISK ASSESSMENT

A standard risk assessment for the project was conducted using a DD2977 form, Deliberate Risk Assessment Worksheet (DRAW), in accordance with DoD policy. Based on the DRAW, the team employed a risk tracking cube (ref. Figure 23) to project the highest likelihood and impact risks respectively.



Risk Management Matrix

Probability	Low (Impact)	Medium (Impact)	High (Impact)
High	GEM interacts with unmitigated traffic within test area	NO 'HIGH' RISKS ASSESSED	NO 'HIGH' RISKS ASSESSED
Medium	Sensors malfunction and test results are invalid	Vehicle achieves undesirable speed during test run.	NO 'HIGH' RISKS ASSESSED
Low	Vehicle battery charges depletes before return to MHE11	Vehicle collides with obstacle during test run	ROS controllers fail to stop vehicle when desired or necessary

Figure 23: Risk cube used to assess the likelihood and impact of all immediate risks to the participants and equipment involved in the project.

The highest impact risk would be the failure of any given controller after road safety validation, which could result in the GEM e2 diverging from its desired path and causing harm to the cadets, pedestrians, or public or private property. To mitigate this, the cadets undertook several measures: the drivers wore seatbelts at all times, the vehicle operated at the IGVC-compliant walking speed at all times, and lane safeties were positioned inside and outside of the cabin of the vehicle at all times, ready to press the emergency stop buttons. Any and every instance the vehicle was immersed into a road testing environment, the functionality of the emergency stop package was confirmed beforehand with the vehicle mounted on stationary jacks.

The highest likelihood risk would be the interaction between the GEM e2 and any traffic or obstacles that might impede it during a test, thus skewing or invalidating the results of that test. The cost of this risk would merely be time lost, as the test in question would have to be reconducted. This risk was mitigated by marking off a controlled test environment before conducting all road tests.

11. CONCLUSION AND LESSONS LEARNED

The major takeaway from this project is how the functionalities provided by an autonomous ground vehicle can augment and improve the capabilities of soldiers on the ground by removing the complexity associated with navigation in urban environments. The ability to reliably employ an autonomous vehicle to navigate through densely populated areas would increase the lethality of modern soldiers by allowing units to reallocate manpower from focusing on operating the vehicle to instead focusing on engaging the enemy. In the future, the basic urban navigation software tools and systems AVRAD has developed may become more refined, allowing warfighters to refocus on the mission at hand while our team's product accomplishes path planning and locomotion.

In addition to the vast depth of robotics knowledge the team has accrued over the past year, this project provided the team with the opportunity to practice the analysis and system implementation necessary to develop engineering design solutions to solve complex and the organizational processes necessary for ensuring a project of such scope and resources is accomplished successfully.

12. FUTURE WORK

The AVRAD 2021 teammates who resume the work of this project after the conclusion of the COVID19 outbreak will first have to validate the functional capabilities of all work currently completed on the project, to include re-testing the packages and controllers designed by the current team. Then these cadets will finish the obstacle avoidance and path planning subsystems on the vehicle, after which it will be ready for testing to confirm its detail product specifications. Once the adjustments and tuning to the critical subsystems are performed and the vehicle is deemed roadworthy, i.e. ready for qualification at IGVC, future teammates will have the opportunity to make the capabilities of the vehicle more robust by developing solutions for more complex urban navigation tasks, including but not limited to merging at interchanges, driving in reverse to park and pull out of parking spaces, parallel parking, U-turns, and three point turns.

REFERENCES

Bries, Matthew and Towler, Jerry, "ROS-Military: Progress and Promise," In Proceedings of the Ground Vehicle Systems Engineering and Technology Symposium (GVSETS), NDIA, Novi, MI, Aug. 7-9, 2018.

Brothers, R and Bevly, D. "A Comparison of Vehicle Handling Fidelity Between the Gazebo and ANVEL Simulators", In Proceedings of the Ground Vehicle Systems Engineering and Technology Symposium (GVSETS), NDIA, Novi, MI, Aug. 13-15, 2019

J. Cymerman, S. Yim, D. Larkin, K. Pegues, W. Gengler, S. Norman, Z. Maxwell, N. Gasparri, M. Pollin,C. Calderon, J. Angle, J. Collier, "Evolving the Robotic Technology Kernal to Expand Future Force Autonomous Ground Vehicle Capabilities," In Proceedings of the Ground Vehicle Systems Engineerin g and Technology Symposium (GVSETS), NDIA, Novi, MI, Aug. 13-15, 2019. Ferguson, David, et al. *Methods for Interactions with Autonomous or Semi-Autonomous Vehicles*. 25 June 2019.

"HDL-64E User's Manual." Velodyne Acoustics, Inc., 2018. https://velodynelidar.com/lidar/products/manual/HDL-64E%20S3%20manual.pdf

Kania, Robert, et al. "Dismounted Soldier Autonomy Tools (DSAT) - From Concept to Deployment." 2014 NDIA Ground Vehicle Systems Engineering and Technology Symposium, 12 Aug. 2014.

Meinhold, Richard J., and Nozer D. Singpurwalla. "Understanding the Kalman Filter." *The American Statistician*, vol. 37, no. 2, 1983, pp. 123–127., doi:10.1080/00031305.1983.10482723.

Merriaux, P., et al. "Wheel Odometry-Based Car Localization and Tracking on Vectoral Map." *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014, doi:10.1109/itsc.2014.6957971.

(PDF) Mobile Interaction with and in Autonomous Vehicles. https://www.researchgate.net/publication/319439467_Mobile_interaction_with_and_in_autonomous_vehicles.

Pollin, Mary. Et al. *Intelligent Ground Vehicle Competition - Self-Drive Challenge 2019 Design Report*. IGVC, 2019, pp. 1–15, *Intelligent Ground Vehicle Competition - Self-Drive Challenge 2019 Design Report.*

Pleune, Mitchell. Et al. *ACTor: Design Report*. IGVC, 2019, pp. 1–17, *ACTor: Design Report*.

Real-Time Motion Planning for Agile Autonomous Vehicles. http://www.mit.edu/~dahleh/pubs/ 2. Real-Time Motion Planning for Agile Autonomous Vehicles.pdf.

Cymerman, MAJ Joseph. "project_v3.m", academic code published in R2019a, MathWorks MATLAB. Stanford, CA: date of publication unknown. Accessed 25NOV2019 from: https://usarmywestpoint. sharepoint.com/ sites/CMESelf- Drive/.

Gerdes, J. Christian. "ME227: Vehicle Dynamics and Controls: Controlling the Vehicle", assignment published on behalf of Stanford University. Stanford, CA: Spring 2017. Accessed 25NOV2019 from: https://usarmywestpoint.sharepoint.com/ sites/CMESelf-Drive/.

Kong, Pfeiffer, Schildbach, Borelli, et al. "Kinematic and Dynamic Vehicle Models for Autonomous Driving Control Design", Mechanical Engineering program, University of California Berkley. Berkley, CA: 2015. Accessed 26NOV2019 from: https://borrelli.me.berkeley.edu/pdfpub/IV_KinematicMPC_jason.pdf.

Open Source Robotics Foundation. “pacmod”, ROSwiki, ROS.org. Mountain View, CA: 16JAN2019. Accessed 26NOV2019 from: http://wiki.ros.org/pacmod.

Pepy, Lambert, Mounier, et al. “Path Planning Using a Dynamic Vehicle Model”, Institute of Basic Electronics, IEF, University of Paris. Orsay, Paris, FR: 2006. Accessed 26NOV2019 from http://www.cs.cmu.edu/~motionplanning/ reading/PlanningforDynamicVeh-1.pdf.

Polaris. “Polaris GEM e2 Owner’s Manual and Model Specifications”, Polaris. Medina, MN: 2019. Accessed 24NOV2019 from: https://gem.polaris.com/en-us/e2/specs/.

The 28th Annual Intelligent Ground Vehicle Competition (IGVC) & Self-Drive. “Official Competition Details, Rules and Format .pdf”, Oakland University. Rochester, MI: 22OCT2019. Accessed 24NOV2019 from: http://www.igvc.org/2020rules.pdf.

Unspecified author (chapter in publicly available textbook). “Chapter 2: Vehicle Dynamics Modeling”, VTechWorks Library, Virginia Polytechnic Institute and State University. Blacksburg, VA: date of publication unknown. Accessed 26NOV2019 from: https://vtechworks.lib.vt.edu/bitstream/handle/10919/36615/Chapter2a.pdf.

Khan, Mudassir “KMeans Clustering for Classification", Towards Data Science, 2017. [Online]. https://towardsdatascience.com/kmeans -clustering- for-classification -74b992405d0a. [Accessed: 12-Dec- 2019].

Gupta, Tushar “Deep Learning: Feedforward Neural Network", Towards Data Science, 2017. [Online]. Available: https://towardsdatascience.com/ deep-learning- feedforward-neural-network-26a6705d bdc7. [Accessed: 12-Dec-2019].

“Deep-neural-network-for-traffic-sign-recognition-systems", Github, 2019. [Online]. Available: https://github.com/ppriyank/Deep-neural-network-for-traffic-sign-recognition-systems.