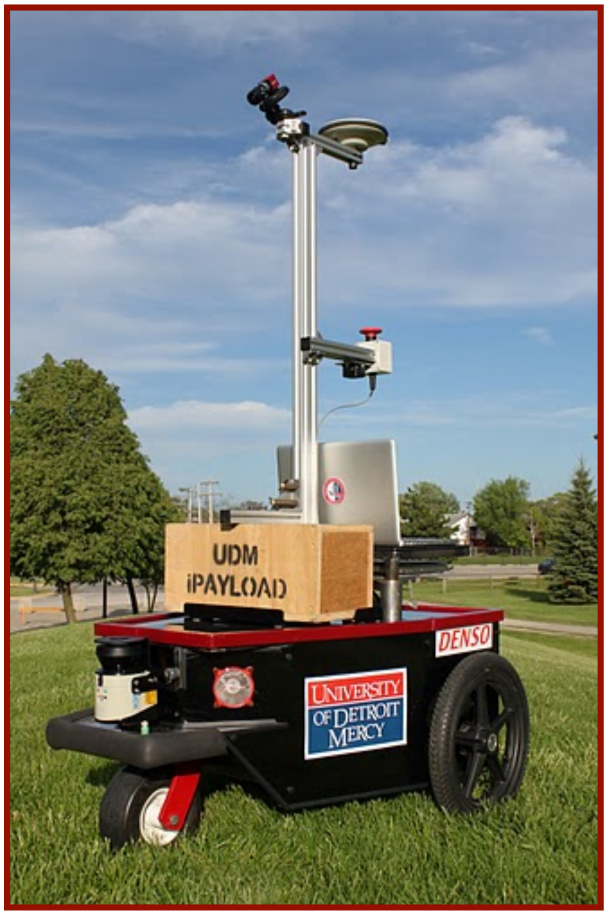
The University of Detroit Mercy Presents

Cerberus

IGVC 2010 Design Report



Team Members

-Matt Parrish

-Jason Osbourne

-Alex Szmatula

-Jean Carrier

-Richard Fatyma

-Elona Palushaj

-Sergei Gage

-Tom Wogaman

-David Shelly

-Rebeca Espinoza

-Maen Hammod

-Qing Wang

-Yue Chen

-Qingxiao Ni

-Jie Li

Faculty Advisors

-Prof. Utayba Mohammad

-Dr. Chaomin Luo

-Dr. Mohan Krishnan

-Dr. Mark Paulik

Consultants

-Cheng-Lung Lee
-Chris Sassak
-Modar Horani
-Yazan Aljeroudi
-Yasser Alnounou

We certify that the engineering design in this vehicle undertaken by the student team, consisting of undergraduate and graduate students, is significant and qualifies for course credits in senior design and in the Master's program respectively.

____

Prof. Mohammad, Dr. Luo, Dr. Krishnan, Dr. Paulik

1. INTRODUCTION

The University of Detroit Mercy (UDM) is introducing a brand new vehicle for the 2010 IGVC. Meet *Cerberus,* shown on the cover page, a differential drive vehicle with front caster incorporating improved software intelligence, electrical enhancements, and some significant new innovations. The team that designed and built *Cerberus* is a cross-disciplinary team consisting predominantly of senior undergraduates, with graduate students addressing some focused needs. The creation of *Cerberus* is based on the principle of continuous improvement, building upon our years of previous IGVC experience. The result is a safe, reliable, and robust vehicle that is capable of meeting the needs of the competition.

2. DESIGN INNOVATIONS

While there are many aspects of *Cerberus* that are improvements based on performance assessment of previous UDM robots, we list here a subset of the enhancements that we consider *innovations* – those that are specifically geared towards a more user-friendly, safe, and effective robot. The four innovations that we have chosen to highlight are briefly described below; more detailed discussions are contained in the sections that follow.

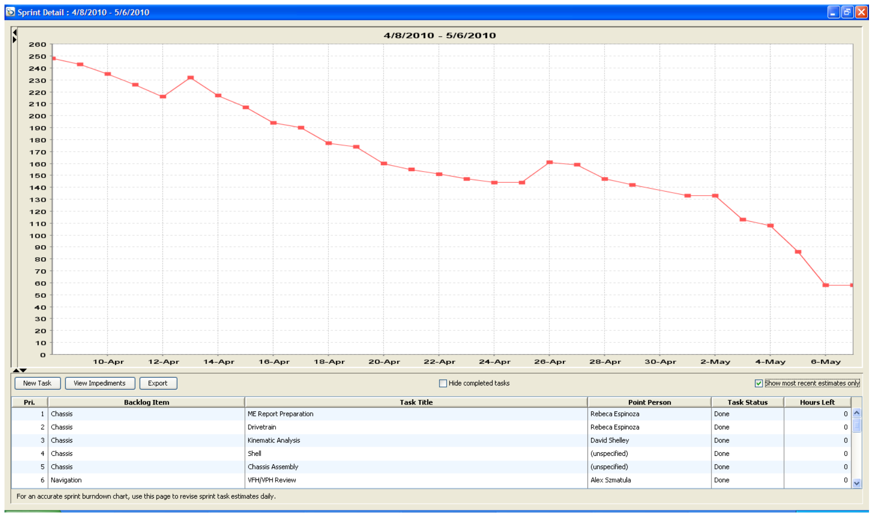
- New vehicle design*: Cerberus*, with its front caster differential drive system, convenient swivel computer trays, 270^0 form-fitting LIDAR, and auxiliary safety features (details in Section 4).
- Ramp-mode indicator: An enhanced vision algorithm based on evaluation of color and geometric aspects of the scene to detect ramps, which was a problematic feature of earlier courses in the Autonomous Challenge (details in Section 6.3.1).
- D*Lite debugging tool: A tool graphically displaying the path planning results of applying the D*Lite algorithm in the Navigation Challenge, which is very useful in troubleshooting (details in Section 6.4.2).
- The magic glove: An accelerometer-based embedded system that enables hand gestures to be used in controlling the robot (details in Section 5.6).

3. PROJECT MANAGEMENT

3.1 Design Process

The team wanted to approach the project in an organized manner that facilitated continuous communication between all team members, allowing for sharing of key information for the various sub-tasks to be completed and identifying and resolving project bottlenecks in a timely manner. In order to accomplish this an adaptive project framework was used that could accommodate variable scope for the individual project tasks. This is an iterative project management framework that seeks to optimize performance for each task in the design-implementation cycle. Furthermore, an agile development methodology, *Scrum*, was employed for short iteration cycles with rapid prototype/module validation. It is critical for an integrated project such as this, that teamwork, collaboration, and cross-functional reviews take place on a frequent basis. The ScrumWorks® project management software system was used to track and record task-group progress, and generate the different burn-down charts for each stage or sprint.

The team used multiple sprints in order to organize and detail plans for each phase. The sprints were titled to represent the phases of the project including reflection, development, testing and implementation. Each of these phases had sections for each of the sub-teams, which were then broken down further for specific tasks the sub-teams needed to address. Throughout the duration of the sprint, the team members updated their status regularly. The updates included progress, problems solved, and problems encountered. When the sprint was updated, a burn-down chart was also updated and overall progress of the sprint was then viewable by the team. Along with the burn-down, the team members were able to post 'impediments' to progress, requesting help from teammates and faculty. Figure 1 shows a burn-down near the end of one of the team's sprints.



Pri.	Backlog Item	Task Title	Point Person	Task Status	Hours Left
1	Chassis	ME Report Preparation	Rebeca Espinoza	Done	0
2	Chassis	Drivetrain	Rebeca Espinoza	Done	0
3	Chassis	Kinematic Analysis	David Shelley	Done	0
4	Chassis	Shell	(unspecified)	Done	0
5	Chassis	Chassis Assembly	(unspecified)	Done	0
6	Navigation	VFH/VPH Review	Alex Szmatula	Done	0

For an accurate sprint burndown chart, use this page to revise sprint task estimates daily.

Figure 1: ScrumWorks Burn-Down Chart

3.2 Team Composition

The cross-disciplinary team was made up of a majority of undergraduate students (8 EEs and 2 MEs) with 5 graduate students working on specialized tasks. Task assignment was based on area of expertise, interest, and the project needs. Figure 2 indicates the team structure and the assumed responsibilities. The total time devoted to the project was approximately 3000 hours.

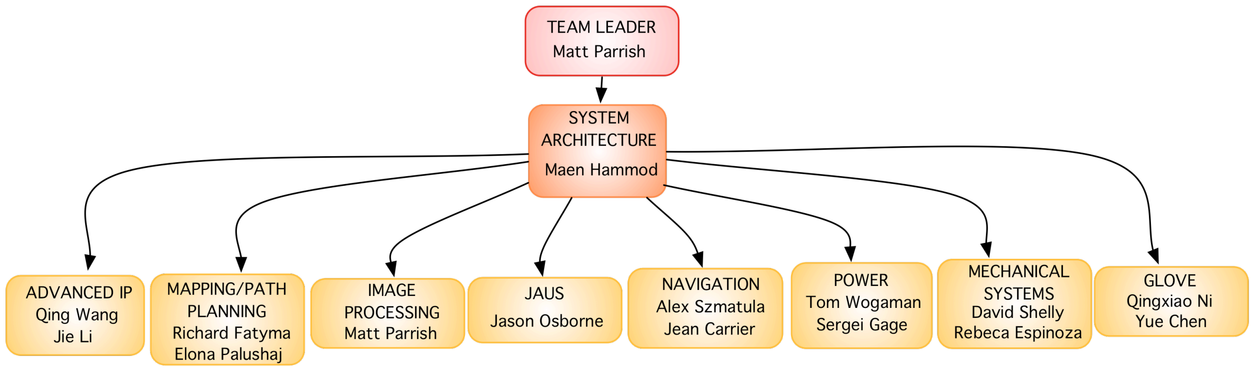


Figure 2: Team Organization

3.3 Design Objectives

The development of *Cerberus* was based on the principle of continuous improvement - UDM has participated in the IGVC for a number of years. A few weeks after the 2009 IGVC ended, faculty and team members met to review our own vehicle's performance as well as that of the competition. The document generated at that meeting served as the basis for the development of the design objectives for *Cerberus*. These are:

1. Meet IGVC requirements.
2. Improve ergonomic design with a focus on easier access for simultaneous use of multiple computers.
3. Improve/replace keyway component of drive train system.
4. Create debugging tool to accompany the use of D*Lite path planning in the navigation challenge.
5. Develop prototype gesture-based guidance system for the vehicle.
6. Enhance color processing vision strategy to deal with problematic situations such as ramp identification and robot turn-around.
7. Reduce weight of vehicle and extend battery life.

3.4 Cost Summary

Cerberus is a developmental vehicle which was built keeping cost-effectiveness in mind in the choices made. The systems incorporated were either designed and built in-house or purchased so as to best suit the project objectives in terms of functionality, performance, cost and development time. Table 1 shows the approximate retail cost of the various sub-systems in *Cerberus* as well as the team cost (some items were donated, reduced in price, or used from an earlier project).

Table 1: Cost Summary

Description	Retail Cost	Team Cost	Comments
Frame/Body	$746	$746	Volunteer work was involved
Drive Train	$2471	$2471	Purchased new
Rear Wheels(2) & Front Caster	$600	$0	
Batteries(4) & Charger	$430	$430	Purchased
Power PCB	$1100	$1100	Designed in-house
Remote PCB	$304	$104	Tranceiver donated by Aerocomm
Camera, Lens, Adapter	$937	$898	
LIDAR	$5500	$5000	Purchased
DGPS & Antenna	$6000	$3500	
Digital Compass	$1096	$0	Donated by PNI Corporation
MacBook Computers(2)	$5000	$5000	
OmniSTAR HP	$1250	$0	Donated
Magic Glove	$183.98	$183.98	
TOTAL	**$25,635.98**	**$19,432.98**	**Savings=$6,203**

4. MECHANICAL SYSTEM DESIGN

Cerberus (shown on the cover page) is designed to be more durable, more accessible, and more versatile than previous competition designs. Some examples of this are in the use of more robust frame materials and the addition of a dual-user laptop carrier. Major improvements to the drive train resulted in a simpler and more compact design than seen in our previous competition vehicles.

4.1 Chassis and Body

The driving function in *Cerberus'* design was to allow the utilization of a 270° field-of-view LIDAR, which translated into a forward-facing wedge body design. This concept eventually morphed into a 3-wheel vehicle, with two rear driven wheels and a leading caster. *Cerberus* has a width of 0.8 meters, a length of 1.04 meters, and a height of 1.8 meters. The total weight of the vehicle, not including the payload, is 110 kilograms.

Cerberus' frame is made out of 20 mm square steel tube stock. The tube sizing was primarily chosen because of its excellent strength and the ease it provides for mounting other components directly to the frame. The vehicle's rectangular extruded aluminum mast is an improvement over the heavy masts used in previous years. Lightweight and versatile for sensor mounting, it is directly attached to the frame on its shortest side. Leaving the longest side parallel to the direction of travel allows *Cerberus'* mast to resist front-to-back vibration, thereby adding stability to imaging sensors.

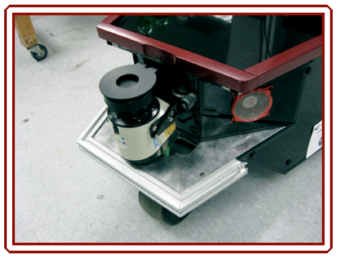


Figure 3: LIDAR Mounting

Cerberus' shell is made of pre-painted 6.35 mm Alumilite sheets. This material has the same weight as 1 mm thick aluminum but is 50 times stronger, adding robustness while minimizing the vehicle's weight. Jagged edges left exposed after the material was cut were filed down and covered with plastic molding for safety.

One concern brought up by the Electrical team was that the mounting of the LIDAR exposed it to frontal impact. This was addressed by adding a steel and aluminum bumper (see Figure 3) as protection. Recognizing that this impact could be with either fragile objects or people, an outside foam-padding layer covers the bumper.

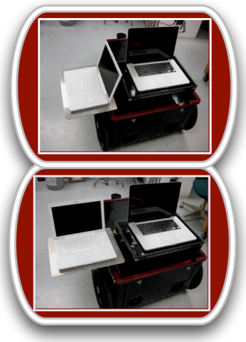


Figure 4: Laptop Swivel Tray Design

The ability to work on two separate laptops simultaneously is a major accessibility improvement that is also new to *Cerberus*. This was achieved by a laptop tray design (see Figure 4) that consists of an upper fixed shelf and a lower shelf that pivots on an arm. Both laptops can be viewed from the rear of the vehicle for side-to-side screen comparison, or one can be rotated for individual work. To protect the computers from water damage in case of rain exposure, a clear polycarbonate shell was built that could easily be placed over the tray structure.

4.2 Drive Train

Cerberus has a simple, yet solid drive train configuration (see Figure 5). It features a front caster and two rear driven wheels. Our two side-by-side 24V motors provide a continuous stall torque of 4.77 N-m. The motor's output is coupled with a 12:1 planetary servo gearhead, which has built-in bearings rated at 1400 N for radial loading and 3000 N for axial loading. It also features an output flange instead of a shaft, which takes away the need for an alignment coupling, along with two tapered roller bearings that would be required to allow for radial and axial loads. Torque is transmitted from the gearhead to the wheel through a two-part connector; it consists of a custom-connecting hub, which is bolted onto the gearhead. Four press-fitted pins transfer torque from the hub to the wheel, and a custom shaft securely clamps the wheel between the hub and itself with three bolts. The use of three bolts instead of one adds strength to the design; a single bolt could easily become detached, but since the load is symmetrically distributed between three bolts, this is less likely.

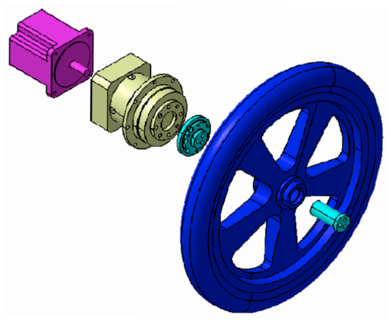


Figure 5: Drive Train

5. ELECTRICAL & ELECTRONIC SYSTEMS

As the electrical system from last year proved to be functional and reliable in general, the goal for this year's electrical systems design team was improvement and innovation. Based on a review of previous performance, some upgrades and enhancements were added that specifically improve safety, efficiency, and functionality. The design changes led to the creation of a schematic for a new and improved printed circuit board (PCB) using EAGLE® CAD software. The improvements incorporated into this new board are discussed below.

5.1 Improvements

Each of the improvements is primarily geared towards enhancing safety and ease-of-use. A new 112-pin microprocessor replaced the old 80-pin microprocessor, thus enabling additional display capability such as each sub-system's power consumption. A single-push start button arrangement is an improvement over last year, when two switching events had to take place in the right order to power the motors and the 24 V network. Finally, an audio alarm to indicate a low-battery condition has been provided; this allows for a safe shutdown of the vehicle or switching to an alternate energy source.

5.2 Power Distribution

In order to provide adequate power to all sub-systems of *Cerberus*, the requirements of each were first assessed under normal and worst-case conditions. The power for *Cerberus* comes from two 12 V 55 Ah Powersonic gel-sealed batteries connected in series, and is distributed via a custom-designed PCB to the vehicle sensors, wireless router, motors, and two Macbook Pro computers. The overall power distribution scheme is shown in Figure 6.

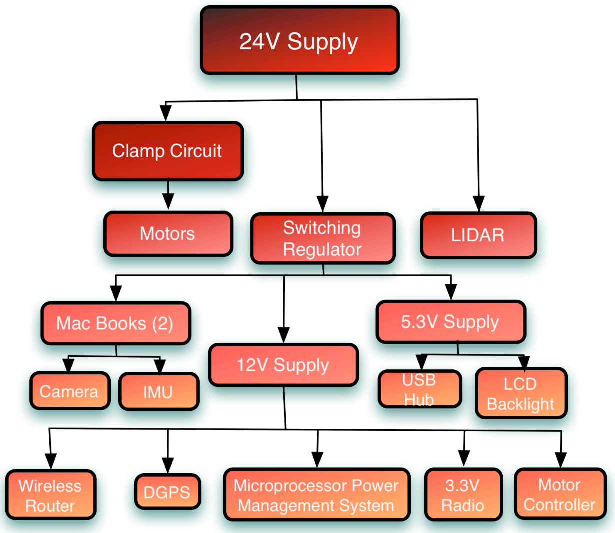


Figure 6: Power Distribution Scheme

5.3 Electronic System Protection and Safety

To help ensure that *Cerberus* is safe, reliable, durable, and easily serviceable, several special features have been incorporated into the power distribution system. The PCB is designed such that high power components are isolated from lower power components. Fuses are strategically positioned on the PCB to prevent electrical damage due to unexpected current surges. The incorporation of high efficiency switching regulators provides stable outputs with low ripple. In addition, these regulators have been designed to protect the PCB from low battery voltage levels, short circuiting, and overheating, thereby extending the life cycle of the circuitry. A clamper circuit is connected to the motor power supply to absorb the motor's back EMF. The status of the power box is conveyed via a series of panel-mounted light emitting diodes (LEDs). Finally, vehicle-wide systems integration is addressed by the use of a real-time current and voltage monitoring system that sends status information from the power box to the main laptop through a USB connection. Thus, if a problem occurs, its source can be located quickly and diagnosed.

Beyond protecting *Cerberus* from possible internal problems, the electronics are protected from both water and dust. The vehicle is weather proofed such that light rain will not cause electrical short circuits. This involves the incorporation of NEMA enclosures for the power distribution system, as well as a shell that surrounds the vehicle chassis and the various components.

Lastly, for user safety, *Cerberus* is equipped with hard, soft and remote E-stops controlled by a mechanical button, the microcontroller, and the remote control respectively. The remote control, which can operate in one of two modes Computer Controlled (PC) or Remote Controlled (RC), is made up of a custom designed PCB housed within a durable Futaba remote control shell. When the remote control is set to operate in PC mode, it transfers control of the motors to the computer (retaining E-stop control). If placed in RC mode, the operator can manually drive the vehicle.

The transceivers that are used in *Cerberus'* design are Aerocomm AC4490-200A transceivers. Although the vehicle only needs to be controlled from a maximum distance of 50ft, with the implementation of the aforementioned transceivers and full antenna extension, the vehicle is capable of being controlled from nearly a mile away. A twist-to-release remote E-Stop button is integrated into the remote control unit. As an added measure of security and immunity to interference, we transmit encrypted data over a spread spectrum wireless link for two-way communication between the vehicle and remote.

5.4 Sensor System

Cerberus incorporates five sensors into its compact design: a camera, a LIDAR, a DGPS, a digital compass, and an IMU. Each sensor is enclosed in a waterproof case and firmly mounted to the vehicle while, at the same time, permitting relatively easy removal for servicing. The following is a brief description of the sensors that are used by *Cerberus* as shown in Figure 7.

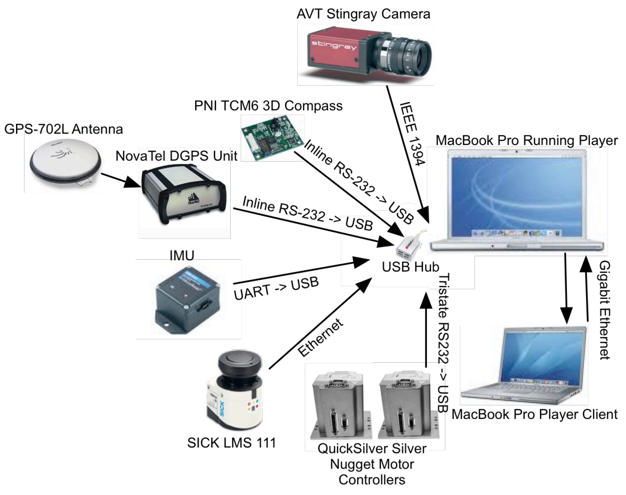


Figure 7: System Communication

Camera: The AVT Stingray F-080C 1/3" CCD camera was selected as the vision sensor for this vehicle. This camera uses the IIDC IEEE 1394B protocol to relay images, which is ideal for machine vision applications, because the frames are uncompressed and various options such as region of interest and lookup tables can be set and executed in hardware. Also, the camera's progressive scanning and high frame rates minimize motion blurring. The CS-Mount lens design enables the camera to accept a very wide-angle low-distortion lens, which provides a 125^0 field-of-view, which makes navigation heuristics easier to implement.

LIDAR: A 270^0 SICK LMS111 LIDAR unit was employed for the purposes of obstacle detection. The unit is capable of collecting data over a 270° field-of-view with 0.25° resolution, a maximum range of 20 m, and a 25 Hz scanning rate.

DGPS: To obtain positioning data in the Navigation Challenge, Novatel's ProPak-LB Plus DGPS system was selected. The DGPS antenna is mounted to the top of the vehicle's mast while the receiver is securely positioned inside the chassis. Using Omnistar HP's DGPS system, the signal is corrected to provide up to +0.1m accuracy. This system provides data at a rate of 20 Hz, which is adequate for *Cerberus'* expected speed and desired performance.

Digital compass: A PNI TCM6 digital compass was integrated into the vehicle to help determine vehicle heading. This compass provides a heading accuracy of 0.5° and updates at 20 Hz, which again is sufficient for the vehicle's speed and desired performance.

IMU: MicroStrain's 3DM-GX2 is a high-performance gyro enhanced orientation sensor which utilizes miniature MEMS sensor technology. It contains a 3-axis accelerometer, 3-axis gyro, 3-axis magnetometer, temperature sensors, and an on-board processor running a sophisticated sensor fusion algorithm. The 3DM-GX2 outputs include Euler angles, rotation matrix, deltaAngle & deltaVelocity, acceleration and angular rate vectors with a maximum data refresh rate up to 256 Hz.

5.5 Data Communication Interfaces

The various electrical and electronics systems were interfaced in the manner illustrated in Figure 7. The AVT Stingray camera is connected via Firewire (1394B) to the MacBook Pro. The DGPS system is connected to the computer through a USB interface using an inline RS-232-to-USB adapter. The SICK LMS111 LIDAR is connected to the computer via an Ethernet link. The PNI TCM6 digital compass also uses RS-232. Finally, all the computers are networked via gigabit Ethernet.

5.6 The Magic Glove

A demonstration system was developed to enable human hand gestures to be used in driving the robot. We call this system "the magic glove" (see Figure 8). A 3-axis accelerometer is used to capture the orientation of a human hand. A microcontroller gathers and interprets the acceleration data signals and transmits them through a wireless Bluetooth link to a Macbook computer on the robot. The computer then controls the robot appropriately through the Player interface. The design allows for the interpretation of 5 commands: start, move forward, turn left/ right, stop, and E-stop. Two speeds are also incorporated. The complete system - accelerometer, microcontroller, and Bluetooth wireless transmitter – is small enough to fit within a lightweight and flexible glove.

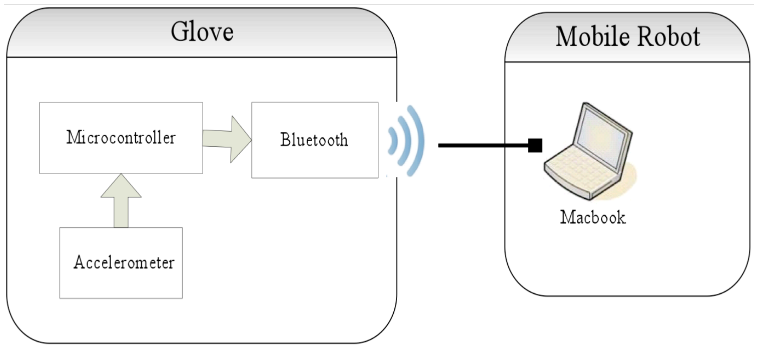


Figure 8: The Magic Glove

6. SOFTWARE DESIGN

The primary goal concerning software this year was to improve upon last year's software structure, integration, and performance. Accordingly, more failure-tolerant algorithms were developed, and user-friendly debugging tools were created. This was accomplished through identifying problematic vehicle behavior, reverse engineering existing approaches, developing and validating the new approaches through simulation, and finally deploying the successfully validated algorithms and testing them on the new vehicle.

6.1 Development Environment

This year the *Cerberus* team continued the use of a multi-layer environment for software development shown in Figure 9.   These layers make communication between hardware and software possible. The first is the server layer which is occupied by Player™, an open-source, Unix-based (Linux or Mac OS X) robotic software system that serves as an interface between robotic algorithm implementations and the vehicle systems. Specifically, it provides standard interfaces for a typical set of robotic peripherals (LIDAR, cameras, motors, etc.) that can be accessed from the local computer or any other computer over a TCP/IP network. The second layer is the client layer occupied by user programs written in MATLAB®©. These layers communicate with Player™ over a TCP socket to acquire data from sensors and send actuator commands, which Player™ then passes on to the vehicle. The writing of efficient wrappers to enable communication between the two layers is what has enabled our program developers to utilize MATLAB®, with its wealth of proven and powerful resources, for algorithm development. The construction of this environment is the single most important development over the past few years of IGVC participation. It provides the stable foundation that facilitates seamless integration of the algorithms of future generations of developers in support of this effort. The pipeline to MATLAB®-based development, a contribution of one of our "IGVC alums" to the Player-Stage™ open source community is particularly useful!

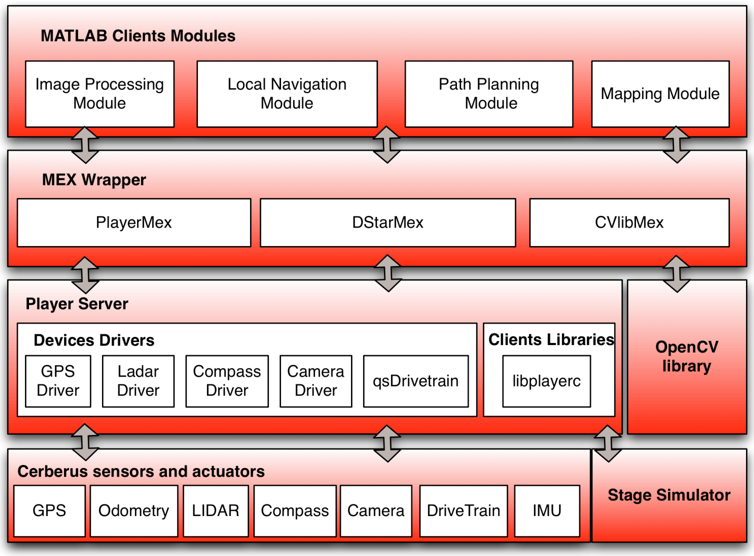


Figure 9: Software Development Architecture

This environment is effective not only for implementation but also for simulation as it allows for the use of Stage™. Throughout the progression of algorithm development, Stage™ allowed for simulated testing rather than on-vehicle testing. The Stage™ simulator was a time-conscious, accurate simulation environment complete with the robot, obstacles, GPS, camera and LIDAR. The availability of Player™-compatible drivers for all of *Cerberus'* hardware facilitates robot software development and testing by allowing algorithm code to run unmodified on either the simulator or actual vehicle. Also, Stage provides a graphical depiction of robot motion within its environment, which offers a powerful tool to gauge the effectiveness of algorithms.

6.2 Data Acquisition and Processing

With abundant data from numerous sensors being processed at once, there are multiple computers used to ensure that all needed data is acquired and processed as quickly as possible. *Cerberus* distributes the processing tasks among these computers as explained below.

Sophisticated vision algorithms are central to a successful strategy for the Autonomous Challenge due to the complex obstacle, lane, and terrain features present. If, when implementing these algorithms, the associated computational complexity causes the overall image-frame processing rate to drop too far, the vehicle may not be capable of operating effectively at higher speeds.

In order to favorably address this tradeoff, a multi-pronged strategy to increase the frame rate was adopted. *Cerberus* distributes its computational tasks over two laptop computers (with a total of 4 processor cores). Using

Player™ as a server facilitates setting up this distributed computing architecture and provides a wealth of existing open-source code to draw from. The top computer, a MacBook Pro, is entirely devoted to running components of the overall vision algorithm. This computer utilizes the MATLAB® parallel processing toolbox to spawn multiple workers which exploit parallelism in several of the IP routines (e.g., color segmentation and adaptive thresholding). This enables effective utilization of the multiple processor cores. When running the Navigation challenge, vision is not utilized; and the top computer is assigned to perform the heavy computations of path planning. A second MacBook Pro, on the swivel, is responsible for accepting the data from the vehicle's sensors, implementing heuristics on the vision results, generating a map, selecting a goal, and navigating the vehicle towards that goal.

6.3 Software Design: Autonomous Challenge

6.3.1 Image Processing

In order to address the increasing complexity of the vision task in the Autonomous Challenge course, *Cerberus* has adopted a new image processing methodology. In the past, images were captured in RGB format, which represents the image color components as red (R), green (G), and blue (B). Lane lines were then identified based on a gray scale image that promotes the lane line color combinations and demotes those of other objects in the scene. This grayscale image was derived with simple scaling and additions of the R, G, and B planes. Recently, colored barrels have been added to the autonomous course, and have made the gray-scale image formula and thresholding very sensitive to lighting conditions and scene content. Therefore, *Cerberus* has moved to a heavily color-dependent recognition algorithm that is more stable and robust.

HSV Thresholding: *Cerberus* processes images using hue (H), saturation (S), and value (V) planes. One aspect that makes the HSV representation appealing is that the lighting information is contained in the value plane, while the color information can be found using the hue and saturation planes. Each pixel in the image can be defined by these three values. The hue specifies the color, the saturation specifies the amount of color, and value specifies the intensity information. The image processing algorithm identifies lanes and obstacles. This is particularly useful since the LIDAR is unable to identify lanes and saw horses that are completely flat or which have portions below its field of vision. The output of the image processing algorithm is passed on to a data fusion process that fuses the LIDAR readings and lane-obstacle information into one obstacle data set.

The image processing algorithm works as follows: first, the captured RGB image is converted into an HSV image, and then thresholding is applied to the hue and saturation planes to remove grass from the image. This leaves behind anything that is an obstacle or is a lane. Finally, thresholding is performed again to pull out anything that is white. In the HSV representation of an image, anything that is white will have a fairly low saturation value and can have almost any hue or intensity value. The algorithm then checks for connected components and removes anything that is small and thus considered to be noise. Figure 10 shows a captured image and the corresponding output of the image processing algorithm.

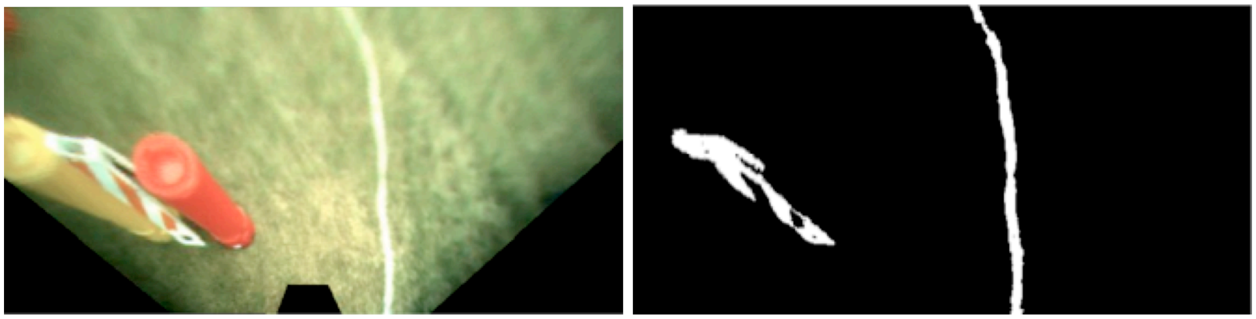


Figure 10: aptured Image on Left/ Processed Results on Right

Ramp Detection: because of its glossy paint and bluish color component, the ramp confuses normal color segmentation algorithms and often shows up as an obstacle under bright lighting conditions. This calls for a ramp identification strategy and a corresponding lane detection algorithm. By studying the histogram of the saturation plane in the HSV image, the ramp can be successfully identified based on number of pixels in a particular range of intensities. Figure 11 shows two histograms, one for a regular section of last year's autonomous trial course, and the other for a ramp section of the same course.

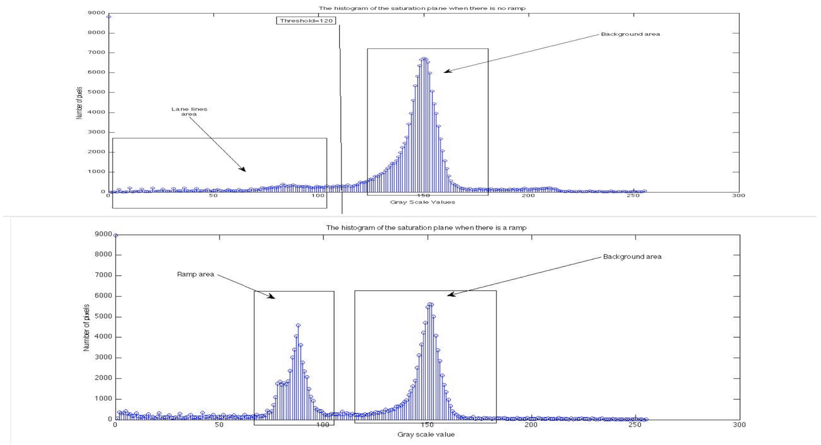


Figure 11: The top histogram represent a section of the autonomous course without a ramp, the bottom histogram represent a ramp section of the autonomous course.

After identifying the ramp a series of transform operations is performed so that only the ramp vertical edges are retained. The final output is an image that only contains the ramp edges.

6.3.2 Goal Selection

The goal selection algorithm is concerned with determining the "forward" direction. For the Navigation Challenge this is relatively easy, as forward is towards the next waypoint. However, in the autonomous challenge, the forward direction is less easy to determine, since it requires the vehicle to "go around the course". The ability to successfully navigate the autonomous course requires the use of image processing, obstacle avoidance and goal selection. These three facets allow *Cerberus* to remain in the lane lines and avoid obstacles, while continuing to move in a forward direction through the course. Keeping the robot moving in the forward direction is the

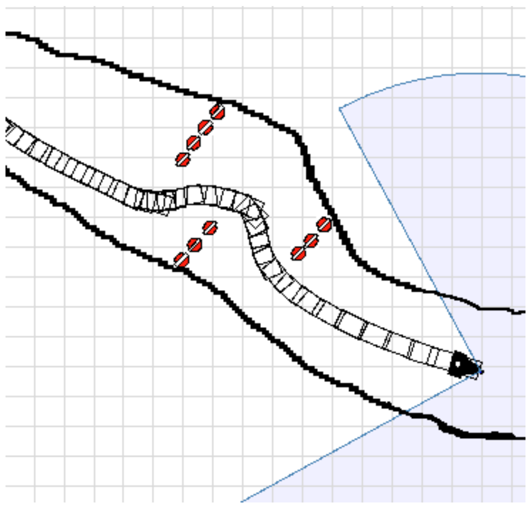


Figure 12: Switchback Navigation Scenario

biggest challenge, which is handled by goal selection. The forward direction then has to be established from the results of the Image Processing algorithms, which are not 100% reliable. Further complicating the situation is the presence of course features such as switchbacks and ramps, which can create apparent traps.

To deal with this situation, a heuristic layer is created. This layer combines two pieces of information to set the goal direction. The first is a "GPS history" direction established by GPS coordinates 4 meters apart from the immediate travel history of the robot. The second is the result of the Image Processing algorithm with its built-in level of confidence measure. When the IP results are less reliable, greater reliance is placed on the tail in setting the goal direction and vice versa. This approach contributes to improved navigation of switchbacks and traps. The effectiveness of this algorithm can be seen from the Stage™ simulation of Figure 12 which shows the robot navigating a switchback smoothly.

6.3.3 Navigation

The LIDAR data is read as a set of polar distances to obstacles not including lane lines and pot holes. While this data is processed in a similar manner using the VFH+ algorithm, there are differences between LIDAR data processing for autonomous navigation and waypoint navigation. The LIDAR data is taken from the degree array and distance array and converted into a polar histogram. This polar histogram is divided into 54 sectors as every sector is 5° and the LIDAR has a range of 270°. The obstacles are enlarged to further eliminate the chance of the robot hitting any obstacles and to provide smoother motion for the vehicle. After the obstacles are enlarged, the autonomous navigation algorithm then incorporates an obstacle-grouping technique (one of the features of the VPH + algorithm). The obstacle-grouping prevents the robot from choosing paths that would lead into a concave obstacle. This creates a smoother path for the robot and reduces navigation time. Each sector is determined to be blocked or not blocked by means of a threshold that is based on the distance of the object. The best sector to go through is then used to guide the vehicle based on a weighted formula that combines deviation from desired direction and associated obstacle densities.

6.4 Software Design: Navigation Challenge

Given a set of waypoints and possible starting positions, a matching lookup table for waypoint sequencing is created. The D*Lite algorithm then provides an initial path (breadcrumbs) between pairs of waypoints, and the VFH + navigation algorithm drives the robot according to those breadcrumbs. The following sections elaborate on the main software modules used in the Navigation Challenge.

6.4.1 Mapping

Mapping is an important capability for autonomous robots that facilitates good decision making. It is particularly beneficial in the Navigation Challenge, since it enables the use of path planning algorithms to determine the optimal route between waypoints. However, even in the Autonomous Challenge, the ability to maintain a global map could be used to enable the robot to partially retrace its path when it "thinks" it is in a trap. Mapping requires an accurate estimate of the robot pose (X, Y, Yaw) so that precise registration of the local map on the global map can be carried out. A Kalman Filter was implemented to estimate vehicle pose reliably by fusing data from the motor encoders, the DGPS, and the digital compass. This also allows GPS outages to be managed if they occur. The creation of a global map enables path planning to be used in the Navigation Challenge to optimize travel.

6.4.2 Path Planning

Performance in the Navigation Challenge is considerably enhanced if path planning is utilized. Path planning is carried out using the D*Lite algorithm, which provides the best route between waypoints. D*Lite can work off a partially complete map of the field, and progressively re-plans the optimal route when the map is augmented with new information as the robot explores. This is where the innovative D*Lite debugging tool proved its worth. Although D*Lite is a very effective algorithm to use for unknown terrain, its integration with a specific navigation task is difficult without a built-in debugging tool. Therefore, we designed a debugging tool which shows the breadcrumbs path from one waypoint to another as D*Lite plans (see Figure 13). Our program prints out these breadcrumbs, which update continuously according to the movement of the robot. Because the planned path can now be visualized by the programmers, debugging can be done by means of simulation.

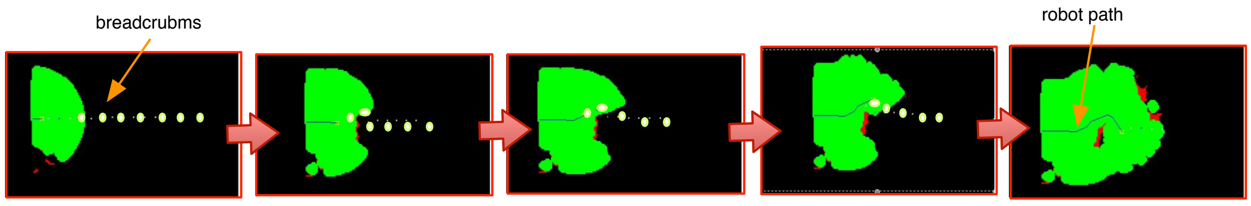


Figure 13: Path Planning Debugging Tool

6.4.3 Navigation

The robot navigation is carried out using a modified version of VFH+ algorithm. In this challenge obstacle grouping is not incorporated, as the challenge does not necessitate tight maneuvers, rather it requires smooth driving and careful path planning. The algorithm parameters (thresholds, swelling factor, and sector selection weights) were optimized to best serve the navigation challenge environment.

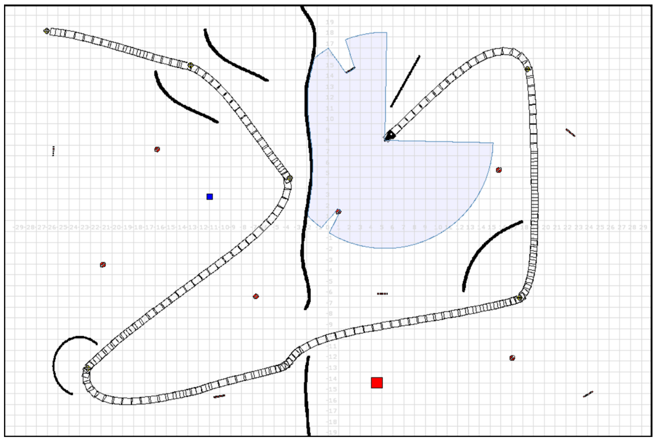


Figure 14: Navigation Simulation

6.5 Simulation

As stated earlier, the software development environment was based on Player- Stage™; the features and merits of this choice have been discussed in Section 6.1. Stage™ provides a powerful simulation environment that can be used to develop and test algorithms in environments similar to those expected in actual competition. A substantial benefit accrues from the use of such a simulation system - the team can construct highly complex course scenarios, test and assess the performance of algorithms, and make necessary corrections much faster than with the actual vehicle. Figures 12 and 14 show sample Autonomous and Navigation Challenge simulation runs.

7. SYSTEM INTEGRATION

This project was divided into subtasks to facilitate development and assignment of tasks to individuals. However, this then requires a process to integrate all the parts into a single, working product. The team utilized the Player/Stage™platform from the beginning. All hardware interaction was done through Player's common interface. This meant that all algorithmic code, being a Player client, could be developed and tested using the Stage™ driver set. Software system integration for *Cerberus* was tested on a test course built on campus to represent possible scenarios that might be encountered at the IGVC. The physical integration of the subsystems, to fit properly and make efficient use of space, was given careful consideration during the chassis design process, by taking their dimensions and locations into account. This allowed the designers to allow for the expected footprint of each subsystem as well as the space needed for their interconnection. After the vehicle chassis was built, the subsystems were positioned in their intended locations before finalizing their secure placement.

8. JAUS

Our goal for the JAUS challenge was to implement a system that would meet the SAE JAUS requirements, while at the same time be compatible with each of the research robots used in our Advanced Mobility Lab (see Figure 15). We chose to implement JAUS using Jr Middleware. It is SAE AS-4 AS5669A compliant software that handles routing JAUS messages on a network. It provides an API that allows a program to create and send out a message through the use of a function call.

Our JAUS implementation interfaces with player to obtain information, and perform JAUS-related tasks. The advantage is that our code could be compiled and run on any system that uses Player.

To verify the functionalities of the system a COP program was written that would send out the messages expected at the competition and display the incoming messages from our system. This allowed us to verify the functionality of messages that are not supported this year by the JAUS validation tool.

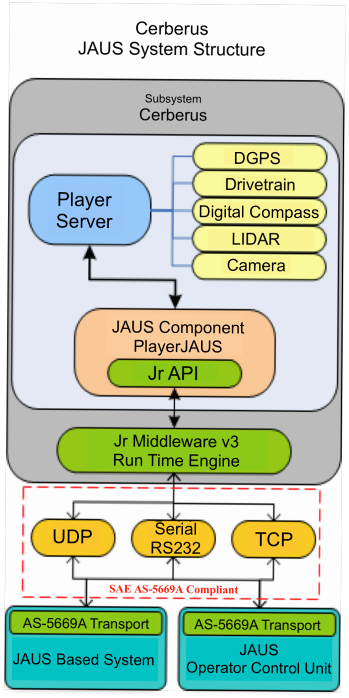


Figure 15: Cerberus JAUS Architecture

9. PERFORMANCE

9.1 Speed

The theoretical maximum speed of *Cerberus* is approximately 5.67 meters/second given its 40.6 cm wheel, 12:1 gear ratio, and 1600 rpm speed at optimal torque. However, the maximum speed that *Cerberus* will reach during competition is capped at 2.24 meters/second (5 mph).

9.2 Ramp Climbing Ability

Based upon the rated torque output of the motors, the size of the vehicle's wheels and the selected gearing, calculations and testing have revealed that *Cerberus* has ample torque to ascend an incline with a gradient of up to 30% (16.7°) without stalling. According to the IGVC rules, the vehicle needs only to be capable of climbing a 15% (8.5^0) incline.

9.3 Reaction Time

For the Autonomous Challenge, it takes approximately 100 ms (10 frames per second) to run the system algorithms (based on software timing estimates). At 5 mph, which is the maximum permitted speed, this cycle time translates to a decision being made approximately every 22 cm of travel. In the Navigation Challenge, the algorithms take approximately 40 ms to complete. At the 5 mph speed limit, this cycle time corresponds to a decision being made approximately every 9 cm.

9.4 Battery Life

Table 2 lists the power consumed by the vehicle components under normal as well as worst-case operating conditions. Using these values, it is expected that the vehicle will be able to run for approximately 5.7 hours under normal operating conditions and slightly over 2 hours under the worst-case conditions. These estimates have been exceeded in actual runs. A 480 W DC battery charger is positioned inside the vehicle to enable the batteries to be recharged.

9.5 Distance at which obstacles are detected

The LIDAR unit on the vehicle is capable of detecting objects at a distance of 20 meters; for *Cerberus*, the LIDAR is configured for a range of 10 meters. The camera is set up to view a shorter range to reduce glare and horizon effects (approximately 5 meters).

Table 2: Power Consumption Estimates

Power Distribution						
	Normal Conditions			Worst-Case Conditions		
Device	Volts	Amps	Watts	Volts	Amps	Watts
LIDAR	24	0.4	9.6	24	0.5	12
Laptop	16.5	2	33	16.5	3.6	59.4
Laptop	16.5	2	33	16.5	3.6	59.4
DGPS	12	0.2	2.4	12	0.2	2.4
Compass	5	0.02	0.1	5	0.02	0.1
Camera	12	0.17	2.04	12	0.17	2.04
Motor/Controllers	24	6	144	24	20	480
Wireless Router	12	0.5	6	12	0.5	6
TOTAL POWER			**230**			**621**

9.6 Accuracy of arrival at waypoints

The waypoints at the competition will be designed as concentric 2m and 1m radius circles centered on the GPS coordinates of the waypoints. *Cerberus'* DGPS system provides an accuracy of + 0.1 meters in DGPS mode, and + 0.01 meters in real-time kinematic (RTK) mode. It can be seen that this accuracy is more than sufficient. This has also been demonstrated both via simulation and actual experimentation.

10. SAFETY, RELIABILITY, DURABILITY

Even though *Cerberus* is a developmental vehicle, it is important for it to operate in a safe and reliable manner as well as be durable, just like any other product. The durability of its mechanical and electrical/electronic systems can be counted on because the design builds upon what worked well in our previous vehicles, while at the same time upgrading and/or redesigning what needed to be improved. *Cerberus* includes several features that not only contribute to its performance, but also increase its safety, reliability, and durability. Three E-Stop systems are implemented to ensure that the vehicle can be stopped safely, quickly, and reliably. These are the soft, hard, and remote E-Stops, which are controlled by the microcontroller, the manual mechanical button on the rear of the vehicle, and the remote control, respectively. The vehicle is weatherproofed such that light rain will not cause electrical short circuits. This involves the incorporation of NEMA enclosures for the power distribution system, as well as a shell that surrounds the vehicle chassis and the various components. All electrical circuits are carefully fused to prevent electrical damage. Furthermore, individual currents and voltages are monitored in all circuits. Diagnostic software and LED indicator systems were developed so faults could be quickly identified and repaired.

Cerberus implements three levels of "watchdogs" on the motor controllers to prevent unintended vehicle operation. The first watchdog is a hardware watchdog, which prevents vehicle operation in the event of a hardware failure. Every 500 ms the computer must send a specific message to the motor controller. If the message is not sent, an E-Stop is triggered. In the event of a hardware failure or computer crash, the message will not be received by the controllers and the vehicle will stop. The second watchdog is a software watchdog, to prevent vehicle operation in the event of a software failure. The motor driver will expect a new velocity command from the software algorithm at least every 2 seconds. If such a command is not received, the driver will halt the motors until a new command is received. The third watchdog monitors smooth wireless data transmission between the remote control and the vehicle. Any failure of the remote control or jamming of the wireless signal will trigger the E-Stop, acting as a hardware watchdog.

11. CONCLUSION

The UDM team is excited at the prospect of defending our title with *Cerberus*. As was evident during the free-for-all session towards the end of the Autonomous Challenge runs in the last competition, luck is important - a team from another University effortlessly circumnavigated the course in the dying minutes and became the only team to do so. But it is also important to have a good process of preparation for the competition in place, without which one would not be in a position to capitalize on luck.