Autonomous Ground Vehicle

A Project Report

Submitted to the Faculty

of the

UNITED STATES COAST GUARD ACADEMY

in partial fulfillment of the requirements for the

Degree of Bachelor of Science

in

Electrical Engineering

by

1/c Tomas Laucys

1/c Adrian Ulanoff

1/c Nathan Morello

Projects in Electrical and Computer Engineering
03 May 2007

Reviewed By:

LCDR David J. Godfrey, Project Advisor

LCDR David J. Godfrey, Class Coordinator

Professor Keith C. Gross, Chief, Electrical & Computer Engineering Section

Abstract

This paper provides the system design of the autonomous ground vehicle being constructed by the United States Coast Guard Academy in the school year of 2006-2007 for the Intelligent Ground Vehicle Competition (IGVC). Designing and building a fully autonomous vehicle for the IGVC put the theoretical knowledge of computer controls, electromechanical machines, communications, and system design into practice. The first objective was to build a robot platform for the sensors and control unit. The second objective was to integrate a laser rangefinder, one web camera, one electronic compass, two shaft encoders, and a differential global positioning system (DGPS) receiver to form a working sensor unit. The last objective was to implement a controller to guide the vehicle's movement.

During the school year, two major objectives were completed. The one not completed was the implementation of the vehicle's motor controller and will be resumed next year. This project is relevant to the USCG because it researches the control systems of autonomous vehicles and can be applied to the UAVs that will extend the capabilities of the fleet. The United States Coast Guard's (USCG) Command and Control Engineering Center (C2CEN) and Navigation Center (NAVCEN) are sponsoring the project. The autonomous ground vehicle project challenges students to apply the skills and knowledge acquired over the first four years of the EE curriculum at the USCGA as well as graduate level skills of the student from the University of Rhode Island.

Table of Contents

Introduction	page 1
Background	page 2
Objective	page 6
System Design	page 10
Results	page 26
Conclusions	page 27
References	page 29
Bibliography	page 29
Appendices	page 31
Appendix A: Publishing Notes	page A-1
Appendix B: Publishing Notes	page B-1
Appendix C: Publishing Notes	page C-1

Introduction

The goal of the project was to build an autonomous ground vehicle to compete in the 15th Annual Intelligent Ground Vehicle Competition (IGVC) in Rochester, Michigan on June 8-11, 2007. This would have marked the first time the United States Coast Guard Academy (USCGA) participated in the competition. The United States Coast Guard's (USCG) Command and Control Engineering Center (C2CEN) and Navigation Center (NAVCEN) are sponsoring the project. In order to compete, the autonomous ground vehicle should meet the design requirements specified in the competition rules. The competition consists of the design evaluation, an autonomous challenge and a navigation challenge [1].

The project is important and relevant through its many applications for military and civilian use. The USCG has a need for project managers and engineers with an understanding of how autonomous ground vehicles work. The IGVC project addresses these needs by providing students with the basic skills of project engineers. The work consists of constructing the physical robotic vehicle, incorporating various sensors and devices and acquiring the data from the sensors, and designing software which controls the movement of the robot and performs autonomous navigation.

The background section covers the IGVC rules and requirements. Previous year's submissions to the IGVC were studied to gain insight into the competition and autonomous vehicle designs. In the objectives section, the project goals and requirements are discussed. The system design section covers the plan to build an autonomous vehicle, how the plan was achieved, and the

major design decisions that were made. The results achieved for the project are the following: complete construction of the vehicle and its propulsion system, integration of laser rangefinder, one web camera, one electronic compass, two shaft encoders, and a differential global positioning system (DGPS) receiver, and design of the motor controller. The team also completed separate power systems for the motors and the sensors. The results are further explained in design section.

Background

The IGVC is an annual event hosted, since 1992, by Oakland University in Rochester, Michigan [1]. The competition is sponsored by organizations such as Tank Automotive Research and Development and Engineering Center (TARDEC), Department of Defense (DOD), and the Association for Unmanned Vehicle Systems International (AUVSI) [1].

The purpose of the IGVC is to offer students an opportunity to design, construct and test an autonomous vehicle to participate in a three-part competition. Furthermore, IGVC aims at providing a chance for students in the electrical engineering, mechanical engineering, and computer science and engineering fields to contribute their expertise toward creation of a single product that is a combination of technologies and knowledge of a broad range of disciplines [1].

An autonomous ground vehicle is a craft that is propelled through direct contact with the ground and is capable of automatic navigation. This craft can sense and react to its environment, and navigate solely on its sensors and the predefined objectives [2].

In order for a team's autonomous ground vehicle to participate in the competition, it must pass the qualification standards in the rules of the competition. The rules are explained in the objectives section. The vehicle is eligible to participate in the competition once it meets all qualification standards. The competition is broken into three events.

The first is the autonomous challenge. The objective is to test the machine's capability to travel through an outdoor course avoiding all obstacles on the track, while staying on the track within the 5 mph speed limit. Vehicles will have to travel up and down natural hills, with a gradient not more than 15%. The course consists of grass and pavement ground with obstacles and lines to direct the vehicles. The lines will be painted white or yellow and be approximately three inches wide. Obstacles on the course include; large orange traffic cones, natural objects such as trees and bushes, and potholes, indicated by solid black circles on the ground [1]. Figure 1 shows an entry from the 2006 competition during the autonomous challenge.

Figure 1: Virginia Polytechnic Institute and State University - Gemini (2006) (courtesy of www.igvc.org)

The second part of the competition is the vehicle design. This is the only part of the competition that is mandatory for all vehicles. Several judges will inspect the vehicle, read the vehicle's project report, and listen to the oral presentation from the group. The judges will score the vehicle's design separately from the other two challenges and will specifically emphasize and encourage innovation and creativity in the application of hardware and software. The robot does not have to win, or even participate, in other parts of competition to win the design part of the competition. [1]

The last part of the competition is the navigational challenge. The goal is to move the vehicle autonomously from an initial position to a number of waypoints and return to its starting point, given only the target location's latitude and longitude. Each vehicle will be allowed up to three attempts to navigate to as many waypoints as possible in seven minutes. The course is laid out on an area of approximately 1.8 acres. Barriers are set up around the course and include trees, barrels, fences, and tents. To successfully navigate to a waypoint the vehicle must come within two meters of the waypoint position. Figure 2 shows a practice field for the navigation challenge [1].

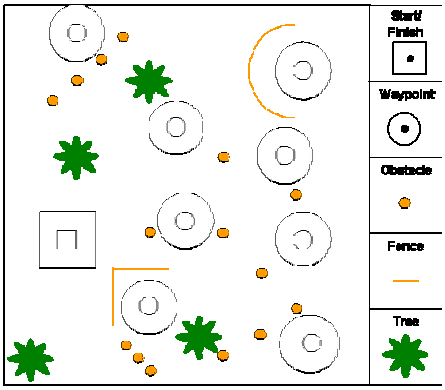

Figure 2: 2007 Practice Map for the Navigation Course (courtesy of www.IGVC.org)

The team examined the technology and designs of the previous participants of the competition before starting the design of the vehicle. Vehicles which were successful in the different challenges of the competition were identified along with their innovative features. The team also examined the design reports for the autonomous ground vehicles that performed poorly in order to avoid making the same mistakes. However, these reports never provided ample information regarding the cause of poor performances. [1]

In the 2006 IGVC, the Virginia Tech autonomous ground vehicle “Gemini” won the autonomous challenge. Gemini was a resubmission from the 2004 and 2005 competitions. Some of the attributes of Gemini that contributed to its success were a tubular aluminum frame, saving valuable weight; a two-body design with independent motors on both front wheels, giving a zero turning radius; a plastic paneling with weather stripping, which protected the electronics from rain; an absorbed glass mat dry cell lead acid battery, which is a sealed battery and can be used when rolled onto its side; an onboard battery charger; and a digital camera with FireWire cable. [3]

Virginia Tech’s “Johnny-5” won the 2006 IGVC navigation challenge. Johnny-5 was another resubmission from the 2004 and 2005 competitions. An attribute that contributed to Johnny-5's success was its onboard gas-electric hybrid power system, allowing for 10 hours of run time. Johnny-5 also incorporated a collapsible mast, allowing for easy transportation of the autonomous ground vehicle. When the vehicle had to be moved, its six foot mast would fold

onto the top of the robot allowing quick movement and transportation. All sensors were connected to a laptop via a serial port card. Lastly, Johnny-5 carried updated software, designed for seamless integration of all sensors in LabView [4].

No one design guarantees success at the competition, so it is important to identify features and techniques implemented in the past winning designs since they provide great foundations for building autonomous ground vehicle. From this point, the goal is to build an autonomous ground vehicle that not only will compete in the navigation and autonomous challenges faster than previous submissions, but also incorporate new and innovations designs.

Objective

The goal was to create a fully autonomous ground vehicle that met the requirement specifications of, and qualify for the 15th IGVC which will be held June 8th to 11th, 2007. To accomplish the goal, objectives were to design and construct the vehicle and its propulsion system, to set up and verify functionality of the sensors, and to design software to control and autonomously navigate the robot in accordance with the requirements specifications provided by IGVC. These objectives are covered in this section, and can be found in Appendix A.

The vehicle must be propelled by direct mechanical contact to the ground. To ensure safety of the people and the surroundings, hardware governed maximum speed of five miles per hour is mandated. The vehicle's propulsion system must have its power generated onboard. The vehicle must be between 3ft to 7ft long, between 2ft to 5ft wide and at most 6ft tall, excluding the height of the emergency stop antenna. The vehicle must also carry a 20 pound cinder block. Even though the vehicle navigation is autonomous, to ensure safety and control of the vehicle's

mobility a mechanical emergency stop (e-stop) must be present. The e-stop must be a red push button, a minimum of one inch in diameter. Its location must be in the center rear of the vehicle, at least two feet from ground level, but not to exceed four feet above the ground. This e-stop must be hardware based and not controlled through the software. In addition, activation of the mechanical e-stop must bring the vehicle to a complete stop.

The vehicle also must have a wireless e-stop. The wireless e-stop must have an effective range of at least 50 ft. Similar to the mechanical e-stop, the wireless e-stop must be hardware based and not controlled through software. The wireless e-stop must also bring the vehicle to a complete stop.

The last construction requirement of the autonomous ground vehicle is the vehicle's capability to perform in various weather conditions. The competition will take place in the event of light rain or drizzle, defined by the judges, but not in heavy rain or lightning. If the competition is held in wet conditions it is necessary the vehicle is waterproof since water inside the robot will ruin the electronic equipment and disable the vehicle.

The capabilities the autonomous vehicle must have are: ability to follow the lanes, detect and avoid obstacles and perform waypoint navigation. In the lane following phase, the vehicle must demonstrate that it can detect and follow lanes that are approximately three inches wide and painted on the ground. The competition will consist of a marked path over an area o approximately 60 to 120 yards long, by 40 to 60 yards wide. The vehicle must follow the

prescribed path to reach the target (end) point of the course. Figure 3 shows a 2006 participant performing lane following during the competition.

Figure 3: University of Detroit Mercy - THOR (2006)
(courtesy of www.igvc.org)

During the competition, the vehicle must demonstrate that it can detect and avoid obstacles because the course will have several different objects along the path of the course which the vehicle must navigate around. Obstacles on the course will be various colors and shapes that may include construction drums, 5-gallon pails, pedestals, cones, and barricades. In addition, natural obstacles such as trees, bushes or plants and man-made obstacles such as a light post or street signs may also appear on the course. Several types of the obstacles used in the competition can be seen in the figures 1, 2 and 4 from previous competitions.

Figure 4: University of Michigan-Dearborn - SMART 2005 (courtesy of www.igvc.org)

Last is the waypoint navigation challenge. In this challenge, the vehicle must prove it can find a path to several 2- meter navigation waypoints. The vehicle will have seven minutes to navigate to within two meters of as many waypoints as it can [1]. Depicted in figure 5, Brigham Young University's vehicle “Y-CLOPS” is participating in waypoint navigation.

Figure 5: Brigham Young University - Y-CLOPS (2006) (courtesy of www.igvc.org)

The goal for December was to construct the operational vehicle with the hardware, sensors and power supply in place and manually control the vehicle through graphical user interfaces on the laptop. The project plan contains more information on project goals and deliverable dates. It can be found in Appendix B. The goal for May 2 was to a have fully autonomous vehicle capable of participating in all parts of the autonomous ground vehicle competition and full documentation that describes how the project was developed.

System Design

Constructing an autonomous ground vehicle to meet qualification requirements can be divided into three sections: mechanical and propulsion system (vehicle design), autonomous navigation system (sensors), and power systems. The mechanical and propulsion system consisted of an aluminium frame, 4 ATV wheels, and a tank drive system. The power system consisted of 3-12 volt batteries, 1-9 volt battery, 2-5 volt batteries and laptop power. Lastly, the autonomous navigation system consisted of a Dell laptop, a Creative webcam, SICK laser range-finder, Honeywell 3000 electronic compass, shaft encoders, and a Trimble DGPS system. In figure 6, the relationship between the navigation, movement control, and hardware can be seen.

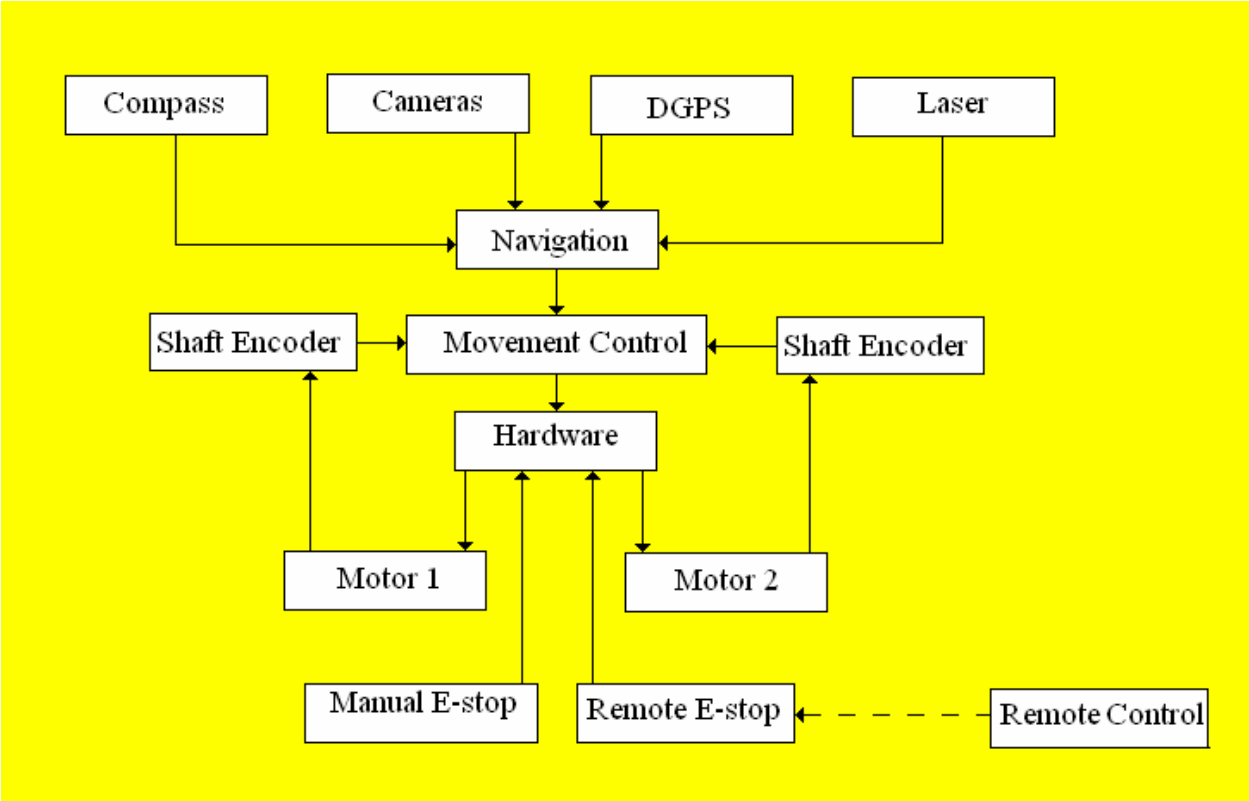

Figure 6: Interaction between the three sections

The electronic compass, cameras, DGPS, and laser are the sensors that aid in the autonomous navigation of the vehicle.

For DGPS, the DGPS receiver used for determining the vehicle's position is the Trimble 214 DGPS. MATLAB was used to retrieve the position (longitude and latitude) and the time stamp from its data string. The DGPS outputs its data in the National Marine Electronics Association (NMEA) string. The data type chosen for the NMEA sentence is the Recommended Minimum (RMC).

For the electronic compass, the Honeywell HMR3000 is being used to obtain the robot's heading. This electronic compass provides NMEA sentences in three different formats: HDG,

HDT, and HPR. The NMEA sentence chosen is the HPR. This data type represents heading, pitch and roll. Furthermore, MATLAB was used to obtain the heading from the HPR NMEA string to provide to the robot so that it could calculate the difference between the actual heading and the desired heading and make the appropriate turn to achieve that desired heading.

Another sensor utilized is a digital camera. One camera will cover a 180° sweep of the robot's front view. The camera will be placed on the center of the vehicle's frame (behind the laser) elevated 2.17 ft above the ground to capture images of vehicle's front view. The data retrieved from the camera is then filtered by the Hough transform (in MATLAB).

The laser rangefinder utilized is the SICK Laser LMS221. The data retrieved from this sensor is obtained via MATLAB. The data retrieved is a continuous real-time scan of the laser making a 180° sweep of the laser's front view. This data essentially looks like an upper semi-circle that is uniquely deformed depending on the environment the laser is viewing. The factors that affect the laser's scan are an obstacles size, distance from laser, and location.

To gather the wheel rotational speed and direction of movement, two simple shaft encoders were built and programmed using MATLAB. Three magnetic sensors were placed in the side of the vehicle next to each of the front wheels. The permanent magnet is mounted in the sprocket of the wheel. All sensors were connected to digital I/O pins on the A/D converter (PC board). The three sensors were labeled as a distinct variable with one sensor used to monitor the distance, speed, and rpm by calculating the time it took the magnet to come close in contact with it. The other two sensors were used only for determining which direction the shaft is rotating to display if the vehicle was moving forwards or backwards. MATLAB code interprets the data received from the

A/D converter and converts it into speed over ground, distance traveled and direction. The shaft encoder sensor assembly is depicted in figure 7.

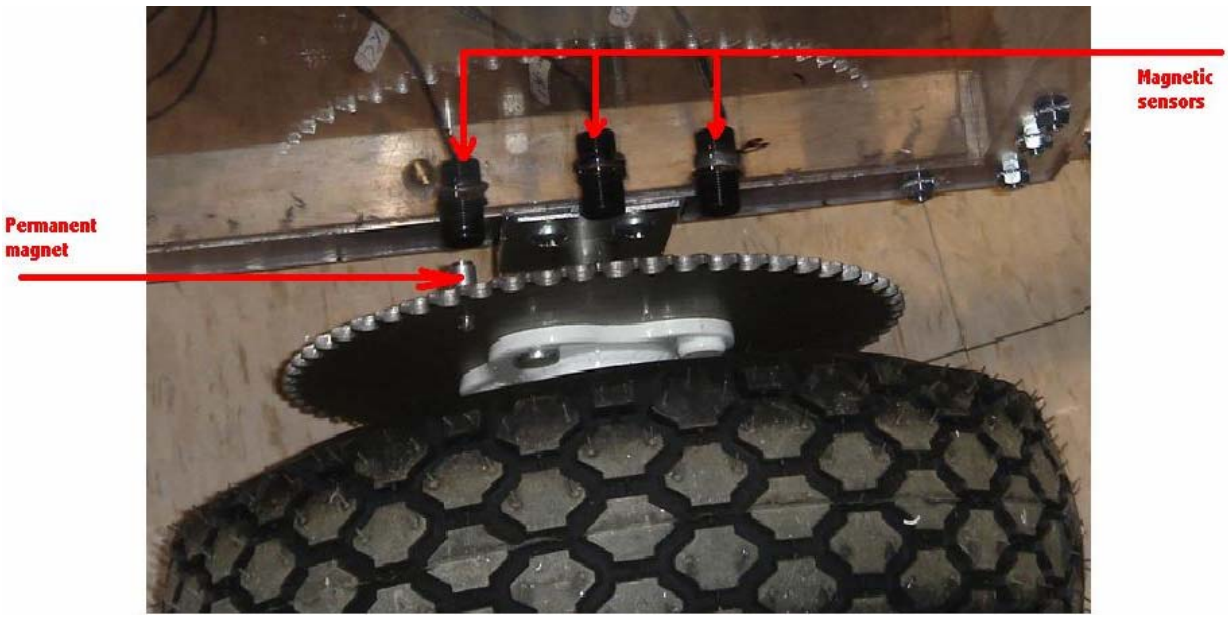

Figure 7: Shaft encoder assembly

The main concern for the vehicle was to easily incorporate equipment, and have compartments for all the equipment. The first step was to get a rough drawing of the vehicle, to see what it might eventually look like. After this was done, dimensions were added based on the IGVC requirements and what was required for size. Once dimensions were added, the vehicle was built in Solidworks using 1”x1” square tubing. The dimensions used were 3’x4’. This would provide a vehicle that is within the size constraints, but also give ample space for equipment. After designing this vehicle, a second vehicle was designed that was 2’x3’. This vehicle also met the requirements and provided enough space for all the equipment, but used less material. Figure 8: IGVC Vehicle Design shows the 2’x3’ vehicle frame from Solidworks**.**

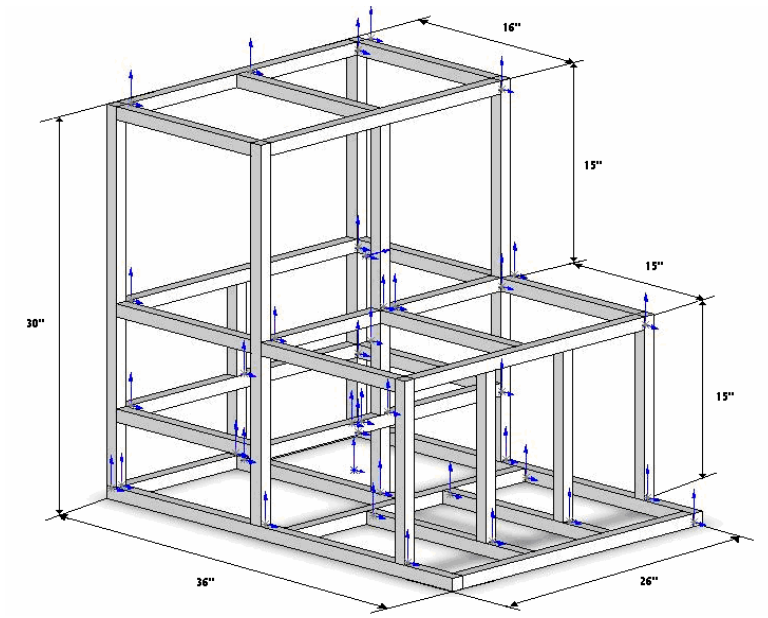

Figure 8: IGVC Vehicle Design

This design had several advantages for IGVC use. On the front, there is a lip for the laser to mount on, with two vertical tubes in the center for the laser to be bolted to. The compartment behind the laser mount is for batteries, and the charging system. On the back of the vehicle there are 3 compartments on top of each other. The lowest is for the motors, and the First Robotics controllers and systems. The second is for the computer to motor and sensors interface, DGPS receiver, serial connection box, and other equipment that the operator does not need to access to get the machine running. The top compartment is for the laptop computer and switch panel. Here the operator will be able to make changes as necessary and start the operation of the vehicle. The compartments were designed this way to simplify equipment installation as well as make the building of the vehicle simpler.

After designing the frame, the wheel mounts, and antenna mounts had to be designed. Figure 9 shows a completed vehicle.

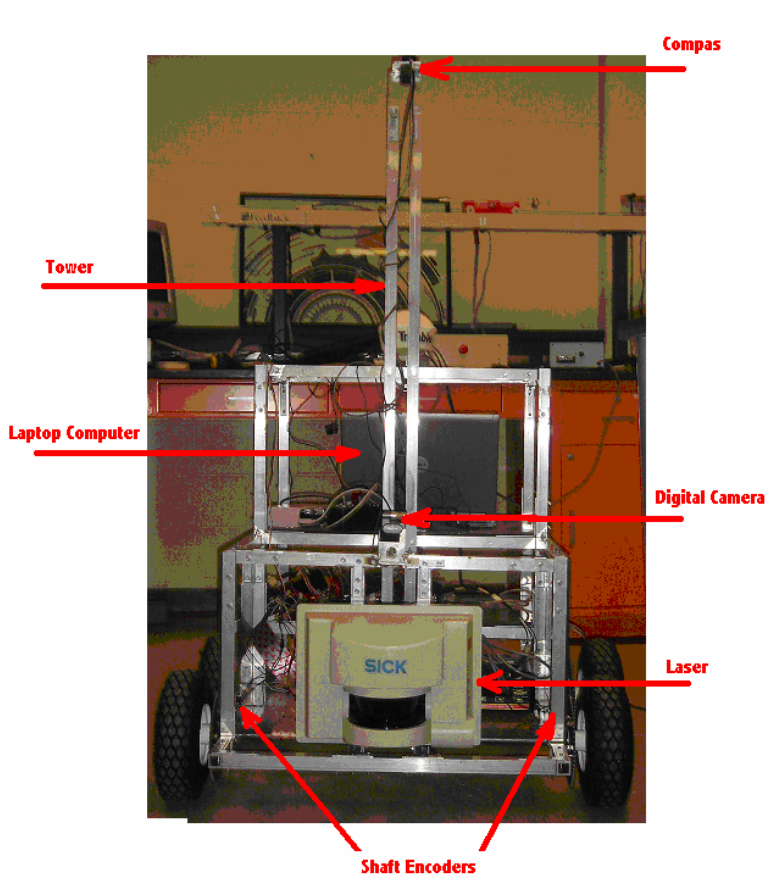

Figure 9: Front View of the Robot

The electronic compass had to be mounted as far from motors and shafts as possible due to the interference they created. This led to a tower being designed that is secured to the frame in several spots but is screw mounted so it can be removed if necessary. The tower can be seen in the front view of the robot in figure 9.

The DGPS antenna was mounted on the back part of the vehicle and can be seen in figure 10. The wheels were mounted to resemble a tank drive system in order to have a zero-turning radius. They can be seen inside picture of the robot in figure 10.

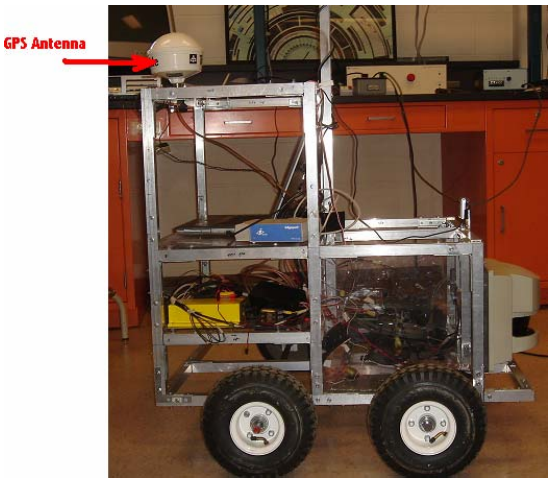

Figure 10: Side View of the Robot

The propulsion system for the vehicle is a tank drive system. It consisted of two motors coupled together on each side. The two motors per side gave more torque so the vehicle would move quicker. The system was set up so that the motors deliver power to all the wheels. This minimized the turning radius of the vehicle, thus providing better navigation. The motors are situated behind the back wheel so the chain runs forward to the back wheel sprocket, then forward to the front wheel sprocket.

Since the sensors, devices, and motors required different voltages, two power systems were created. The decision was made to split the power system into two was based off a couple of

factors. First, the electrical or mechanical disturbance from the power grid connected to the motors could interfere with the continuous operation of the sensors. For example, the start of the motors will cause a voltage drop which could introduce an anomaly to the system. We also anticipated an implementation of the emergency stops and predicted that a simultaneous cutoff of power for the motors and sensors would result in a lengthy reset of the laptop and all of the sensors. Two separate power grids eliminate this problem and allow us to stop the robot without turning of the sensors. There are no concerns with the power budget since all of the distinct batteries are adequately provided by the electrical engineering and mechanical engineering departments.

The power system for the sensors used two 12 volt batteries and three 9 volt batteries. The laser rangefinder required 24 volts to operate while the DGPS used 12 volts to function. In addition, one 9 volt battery is used for each e-compass and the wireless camera while the web cameras are powered by the laptop power. The second power system is for the drive train and control of the robot. This system requires a 12 volt battery to power two motors and a motor controller. Figure 11 shows the power supply we will build for the robot.

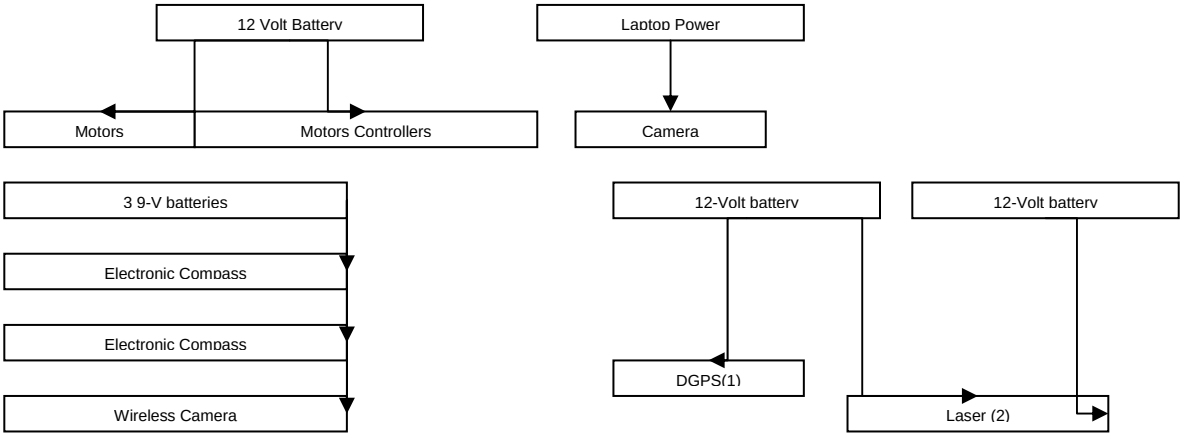

Figure 11: Power System for the Robot

The power supply of the vehicle consisted of a “black box” type unit with a modular wiring system. The wiring system was intended to lead to a single weather-proof outlet on the outside of the robot. The purpose of this is to provide the capability of recharging the batteries necessary for powering the laptop, different sensors, and motors of the vehicle.

The key component in the emergency e-stop system is an electric relay that cuts the power for the motors. When the motors are off, the gears stop the movement of the vehicle. The electric relay is controlled by an electrical circuitry (figure 12) that consists of the two transistors and an operational amplifier. The 12 volt sensor battery provides power for the e-stop circuitry. The circuitry receives a logic signal from the Futaba analog servo circuitry (0.04-5V). A wireless link is established using the Futaba remote control unit and the Futaba FM receiver that work for the distances greater than 50 feet and thus meet a design requirement. The receiver is powered by 5 volt rechargeable battery. The remote control unit has an internal rechargeable 12 volt battery.

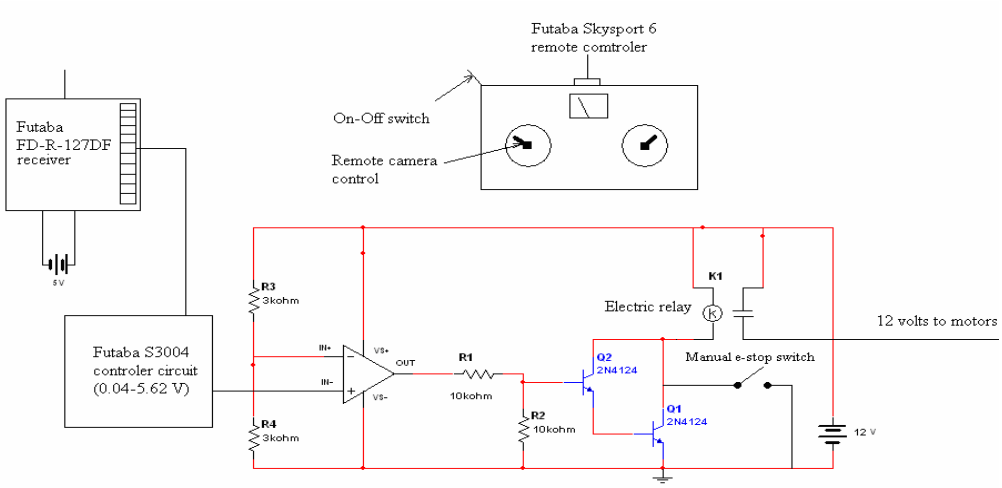

Figure 12: Emergency stop (wireless and manual)

A manual emergency stop is simply a pushbutton which is pressed grounds a negative terminal of the electric relay bypassing the electric circuitry of the wireless emergency stop. If the manual stop is energized the power of the motors can not be restored remotely. The manual e-stop switch can be seen in the lower right corner of figure 12.

The last component of the design solution is the control system. In this component, the development and implementation of a single PD controller for steering and distance control of the autonomous ground vehicle was created in order to enable the vehicle to travel from one location to another. The controller will manage all of the robot's movements and speed. In addition, it will work in unison with the data retrieved from the sensors to safely and accurately navigate on a prescribed path or to targeted waypoints. The autonomous control system will be comprised of four distinct state-space controllers. The results should verify that implementing

this PD controller will provide acceptable lateral and longitudinal control of the autonomous ground vehicle as well as excel in tracking accuracy and steady-state error.

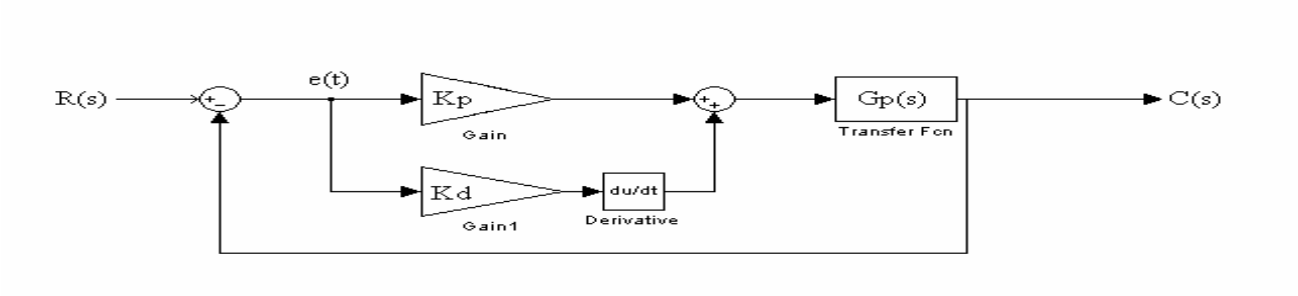

Figure 13: PD controller devised for the robot.

Figure 13 represents the PD controller devised for the robot. The coefficients have not been determined yet and are pending since the vehicle has not been completely assembled. When the vehicle is completely built, the system I.D. can be calculated and thus the Kp and Kd values can be found which depend on the plant Gp (also known as the system I.D.). The Kp value is proportional to the error while the Kd value monitors the rate of change. The controller designed for the vehicle is simplified since it is only concerned with turning to a specified angle and driving a certain distance. The controller was designed this way since the input that would be provided to the vehicle would be the desired heading and range to a particular location. Lastly, the R(s) represents the voltage input for both left and right motors necessary for the vehicle to reach the desired location.

The figure 14 presents a better illustration of how the entire vehicle control system works and the integration of the sensors. The system I.D. is modeled through the speed and heading transfer functions. Since the vehicle is not built yet, the equations can not be presented since they are not

accurate. Basically, what should be drawn from this figure is the user enters the desired position, the software (MATLAB code) controls the vehicle's movements (i.e. voltage to the motors) and acquires data continuously from the sensors and feeds the data back to the computer. The controller, which is essentially the big box encompassing the speed and heading transfer functions, makes sure the vehicle reaches the desired position by comparing it with the actual position and makes suitable corrections when necessary to ensure the vehicle is facing the precise heading and has traveled the correct distance.

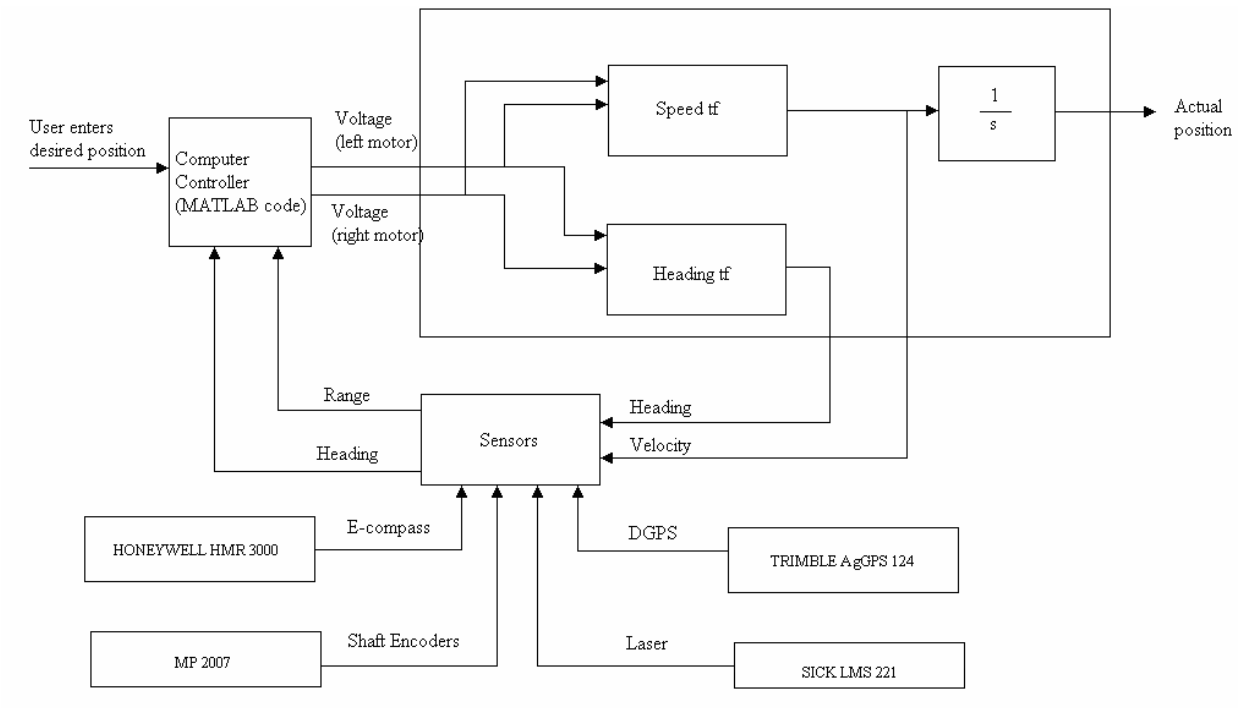

Figure 14: Vehicle mechanical control scheme

During the building of the vehicle, several decisions were made. It was decided to build a 2'x3' vehicle because it is more cost efficient to the project. A larger vehicle provides more room for equipment and a wider base for more stability, but comes at a higher cost and makes navigation in tight places more difficult. A smaller vehicle was cheaper, navigated more efficiently, and still had enough room for the equipment needed. The smaller vehicle was the best choice for this project.

After deciding on the vehicle size, the next decision was to determine which materials to use for constructing the frame. There were several choices. The first material was the aluminum used on the first robotics vehicle, which had predrilled holes for easily putting pieces together. It costs $39.95 per 6 foot segment. The second was aluminum bars, which is very strong and can support large amounts of weight but cost $56.72 per 6 feet. The third was aluminum tubing, which is strong, but is lighter weight because it is hollow. Aluminum tubing costs $20.78 per 6 feet. The decision was made to use the aluminum tubing for several reasons. The first reason was it provided the strength needed for our vehicle. Secondly, it was not hard to drill through aluminum. Finally, it was much cheaper than the solid aluminum bars, or the predrilled first robotics aluminum.

The third decision was determining what type of wheels to use. The competition requires a wheel that is capable to driving on pavement or grass. This meant the wheel needed to be well treaded and sturdy since the vehicle will be driving off road. The wheel chosen is shown in Figure 15: Wheel used on vehicle is a large in order to provide enough clearance for the bottom

of the vehicle. It also had a sprocket mounted that is the correct tooth size, which helped with connecting the wheels to the power system. The wheels have a shaft size of ¾”, which is larger than what was used on the first vehicle tested on. Having a wheel with a sprocket mounted eliminated the need for the purchase of a sprocket with a shaft size of ¾”.

Figure 15: Wheel used on vehicle

The final decision, after all parts were decided upon, was how to build the vehicle. The strongest, most permanent solution would be to weld all the parts together. Another solution was to drill holes and mount all the pieces together using bolts and brackets. It was decided to use bolts and brackets to put the vehicle together because welding requires practice and knowledge, which nobody on the team had. By bolting the pieces together, the team was able to build the vehicle solely by the team’s efforts.

The decisions on the propulsion system were driven by what parts were available at the USCGA. The team did not want to purchase parts for the propulsion system since the team already had access to a wide variety of equipment from the FIRST Robotics competition teams. The motors

and the coupling were used from the 2006 FIRST Robot. The rest of the propulsion system was taken from the 2003 FIRST Robot.

The Futaba transceiver and the receiver were chosen for the wireless link of the remote e-stop because of the economic considerations. The academy already possessed this equipment. The control circuitry for the electric relay was necessary because a sufficient current supply was necessary in order to power the electromagnet of the relay.

The team decided to build shaft encoders using three magnetic sensors, magnet and a MATLAB program because it was a cheapest option available. It was also faster to build the shaft encoder rather than ordering them. While there are numerous shaft encoders both coupled with motors and separate ones that are available on the market they tend to be expensive and in many cases incompatible with our vehicle design because of the motors that were already available and the wheel that are mounted on the stationary shaft. The shaft encoders were placed on the two front wheels of the vehicle since the vehicle is operating a left and a right motor to emulate a tank drive system. The shaft encoders are vital because they provide the distance traveled by the vehicle which is one of the parameters required by the PD controller in order to travel to a prescribed location. This sensor provides constant feedback as to whether the vehicle has achieved that prescribed distance.

The Trimble 214 DGPS receiver was used since it was provided from the beginning of the project. MATLAB was the chosen as programming language because not only is it a powerful and relatively easy-to-use computational tool, but it interacts very well with other sensors and devices. The RMC sentence was chosen because it provides the essential position, velocity and time data critical for autonomous ground vehicle to accurately navigate. In addition, the DGPS was chosen because it achieves an accurate position varying between 0.5 and 1 meter. Lastly, MATLAB was used for integrating all other sensors and devices because it was less confusing and difficult for the user when all programs were written in one common language. This also created more ease for the user to identify any possible errors in the code.

The Honeywell HMR3000 was used since this sensor was provided from the beginning of the project. The only relevant information from the HPR NMEA sentence used from the electronic compass was the heading. Pitch and roll were considered for determining if the robot is approaching a declining or inclining hill and how much speed to apply to the motors to ensure the robot can make way while maintaining the 5 mph limit. However, these variables were never used since the controller designed did not require this information.

A digital camera was used since the pictures it took were digitized which made it possible for digital image processing techniques. Initially, three cameras were intended to be used to get an accurate 180° sweep of the robot's front view. However, due to the lack of processing capabilities of the digital cameras presented to the team, the digital cameras were not able to take pictures simultaneously and thus fail to work afterwards. So the plan reverted to just using one digital camera. The purpose of using the digital camera was to capture several images in order to detect the yellow or white lanes which the autonomous ground vehicle must follow. The Hough transform was chosen because it identifies lines in an image. The Hough transform applies an

edge detection operator to a raw image (picture) and determines which lines fit most closely to the data in the image. To serve as an additional tool, the team decided to incorporate a wireless camera on the robot which displays on a remote laptop what the robot is viewing.

The SICK Laser LMS221 was the particular laser rangefinder chosen since this sensor was purchased before the team started working on the project. The purpose of the laser rangefinder to detect obstacles within 8 meters and safely navigate around those obstacles that may be too close or are along the path to a targeted destination (waypoint).

Results

The team was successful in completing a majority of the requirement specifications. A new robot was constructed as the base frame for the IGVC vehicle with a length of 3 ft, width of 2.17 ft and height of 5.17ft. This fulfils the dimension requirements 1-3 in appendix A. The vehicle's design has a compartment built in it with the dimensions 26" x 15" x 15" which is able carry a payload of 2 lbs with the dimensions of 18"x 8"x 8". The vehicle's four ATV wheels and a mechanical propulsion system ensure that the vehicle maintains constant contact with the ground. The vehicle has 4 motors that are controlled by a system of Innovation First robotic controllers and have enough torque to move a 198.7 lb robot with an additional 20 lb payload. The vehicle's propulsion system is independent since its power is generated onboard by three 12 volt batteries. One12 volt battery runs the motors while the other two 12 volt batteries provide power for the sensors. The mechanical E-stop is a mechanical push button located in the center

rear of the vehicle 2.5 ft above the ground. Furthermore, the E-stop is solely hardware based and completely stops the vehicle when activated. The wireless E-stop is a remote controller that is also solely hardware based and completely stops the vehicle when activated and worked well at distances of 50 ft and greater. All other requirements specifications were not met due to the time constraints.

Besides meeting most of the requirement specifications, the team made significant progress toward completing autonomous navigation which consisted of lane following, obstacle avoidance, and waypoint navigation. The laser-rangefinder, electronic compass, DGPS receiver and shaft encoders and web camera have been integrated with the laptop computer. MATLAB code was written for data acquisition from the mentioned sensors. This code can be found in Appendix C. An electronic compass is used within a MATLAB GUI to control the turning of the vehicle to any desired heading.

Overall, the team's success in accomplishing many goals this academic year, built a solid foundation for future teams to further develop the project and participate in the IGVC.

Conclusions

The goal of the Autonomous Ground Vehicle project was to build a vehicle to participate in the IGVC. During the 2006-2007 academic year, progress was made towards completing a vehicle to compete in 2008. All the sensors necessary to have a working autonomous ground vehicle were integrated via MATLAB. FIRST Robotics competition motor system to gain control of the movement of the motors. By outputting voltages thru the digital to analog card in the laptop, the computer was able to control the speed and direction that each motor rotated. At the end of the project, the vehicle was able to collect data from all sensors, turn a specified direction based off values provided from the e-compass, and move designated distances at constant speeds based

upon the input to the control function and the values read by the shaft encoders. The integration of the motor control system and the mapping program was never completed so the vehicle was never tested for operating as an autonomous ground vehicle. The most important lesson learned was to provide sufficient documentation for superiors on current project status and any progress achieved. In addition, it became evident that providing complete and detailed documents such as design specifications and requirements specifications are vital in determining when the project is finished and providing enough instructions so that others could reproduce the same product. While working with LT Everette, the team realized that MATLAB was insufficient because it could not process data fast enough to create a visual map for the vehicle to autonomously navigate. In the future, a project like this should be written in C or C++ so that the processing power is not lost in a supporting program. Designing and building an autonomous ground vehicle with four group members is not an easy task and requires all members of the group to be knowledgeable on all aspects of the vehicle operation. As a recommendation, future teams should consist of cadets from the electrical and mechanical engineering fields since the project requires knowledge and skills from both majors. If this recommendation is considered, successful completion of the project by the deadline becomes more feasible.

References

[1] Theisen, Bernard. "Official Competition Rules." Intelligent Ground Vehicle Competition. 1 Sept. 2006. Oakland University. 15 Sept. 2006. Available from: http://www.igvc.org/rules.html.

[2] Wit, Jeff, Carl D. Crane III, and David Armstrong. "Autonomous Ground Vehicle Path Tracking." Journal of Robotic Systems 21 (2004): 439-449. 27 Sept. 2006. Available from: http://cimar.mae.ufl.edu/CIMAR/pages/pubs/wit_2004_journal_rob_sys.pdf.

[3] Baity, Sean, Andrew Bacha, David Eargle, Brett Gombar, Jake Green, Bobby Mott, Colin Todd and Jon Weekley. "Gemini 2006 Design Report." Intelligent Ground Vehicle Competition. 01 OCT 2006. Virginia Tech. 09 OCT 2006. Available from: http://www.igvc.org/design/reports/dr124.pdf.

[4] Gombar, Brett and Andrew Bacha. "Johnny-5." Intelligent Ground Vehicle Competition. 01 OCT 2006. Virginia Tech. 09 OCT 2006. Available from: http://www.igvc.org/design/reports/dr125.pdf.

[5] Monteiro, Luís S., Terry Moore, and Chris Hill. What is the Accuracy of DGPS? Diss. The Univ. of Nottingham, 2004. 3 Feb. 2007. Available from: http://www.nottingham.ac.uk/iessg/research/res_publication/rp2004/dgps_enc04paper.pdf

[6] Phillips, Charles L., and Royce D. Harbor. Feedback Control Systems. 4th ed. Upper Saddle River: Prentice Hall, 2000. 69-114.

Bibliography

"Amigo." Intelligent Ground Vehicle Competition. 01 OCT 2006. Hosei University. 09 OCT 2006. Available from: http://www.igvc.org/design/reports/dr29.pdf.

"Artemis." Intelligent Ground Vehicle Competition. 01 OCT 2006. Virginia Tech. 09 OCT 2006. Available from: http://www.igvc.org/design/reports/dr36.pdf.

Avitabile, Michael, Timothy Judkins, Charles Lamb, Robert McNeal, Jamie Riggins and Brijesh Sirpatil. "Artemis 2001." Intelligent Ground Vehicle Competition. 01 OCT 2006. Virginia Tech. 09 OCT 2006. Available from: http://www.igvc.org/design/reports/dr39.pdf.

Balas, Christian, Hui-Shan Chang, Kuan-Chih Chen, Cheng-Lung Lee, Robert McKeon, Utayba Mohammad, Ricardo Moore, Niyant Patel and Benjamin Radke. "Thor." Intelligent Ground Vehicle Competition. 01 OCT 2006. University of Detroit Michigan. 09 OCT 2006. Available from: http://www.igvc.org/design/reports/dr150.pdf.

Bento, L. Conde, Urbano Nunes, Fernando Moita and Antonio Surrecio. Sensor Fusion for Precise Autonomous Vehicle Navigation in Outdoor Semi-structured Environments. Vienna: IEEE Conference on Intelligent Transportation Systems, 2005.

Carmer, Dwayne C. and Lauren M. Perterson. "Laser Radar in Robotics". Proceedings of the IEEE 84 (1996).

Cook, Jarriel, Pete Guion, Josh Guzman, Jeff Mason, Brian Renn and David Seigel. "Kenny." Intelligent Ground Vehicle Competition. 01 OCT 2006. University of Maryland. 09 OCT 2006. Available from: http://www.igvc.org/design/reports/dr5.pdf.

Dhareshwar, Rahul, Meyyappa Ganesh, Rob Hicks, Sampath Kanakaraju, Jaiganesh Karuppuswamy, Xiaoqun Liao, Nathan Mundhenk, Ramya Ravindran, Mike Rivett, Mayank Saxena, Sathish Shanmugasundaram, Vishnuvardhanaraj Selvaraj, Vijayakumar Sreekantan and Ramesh Thyagarajan. "BearCat II." Intelligent Ground Vehicle Competition. 01 OCT 2006. University of Cincinnati. 09 OCT 2006. Available from: http://www.igvc.org/design/reports/dr26.pdf.

Joerger, M and B.M. Pervan. Autonomous Ground Vehicle Navigation Using Integrated GPS and Laser-scanner Measurements. Position, Location, And Navigation Symposium, 2006.

Kaygisiz, B.H, A.M Erkmen, and I. Erkmen. GPS/INS enhancement using neural networks for autonomous ground vehicle applications. Proceedings Intelligent Robots and Systems, 2003.

Kedrowski, Philip, Justin Macken, David Parnell, Matt Peppers and Milivoy Villarroel. "Navigator." Intelligent Ground Vehicle Competition. 01 OCT 2006. Virginia Tech. 09 OCT 2006. Available from: http://www.igvc.org/design/reports/dr47.pdf.

Lucente, Anthony, Ross Marten, Jonathan Hyland, Hachem Nader, Ebrahim Nasser and Vin Varghese . "Road-Runner." Intelligent Ground Vehicle Competition. 01 OCT 2006. University of Michigan-Dearborn. 09 OCT 2006. Available from: http://www.igvc.org/design/reports/dr118.pdf.

Merz, T , P. Rudol and M. Wzorek. Control System Framework for Autonomous Robots Based on Extended State Machines. International Conference on Autonomic and Autonomous Systems, 2006.

Nister, D, O. Naroditsky and J Bergen. Visual odometry. Proceedings of the 2004 IEEE.

Ruggiero, Eric, Chris Cantrell, Fernando Perez, Peter Boelhouwer, Roy Robertson and Chris Terwelp. "Maximus." Intelligent Ground Vehicle Competition. 01 OCT 2006. Virginia Tech. 09 OCT 2006. Available from: http://www.igvc.org/design/reports/dr45.pdf.

Ryu, Jae Heon, Yu Shin Chang, Gun Hong Pak, Chun Han Lee, Seok Hyun Park, and Man Hyung Lee. Research on navigation system of an autonomous vehicle using RTK-DGPS. 30th Annual Conference of IEEE on Industrial Electronics Society, 2004.

Slominski, Eric, Joong-Kyoo Park, Christopher Terwelp, Patrick Forman, Ian Hovey, Jared Mach, Joseph Roan and Merritt Draney. "Biplanar Bicycle 2002." Intelligent Ground Vehicle Competition. 01 OCT 2006. Virginia Tech. 09 OCT 2006. Available from: http://www.igvc.org/design/reports/dr9.pdf .

Some Sensing and perception techniques for an omni-directional ground vehicle with a laser scanner. IEEE International Symposium on Intelligent Control, 2002.

Shu, Yuan and Zheng Tan. Vision Based Lane Detection in Autonomous Vehicle. Hangzhou, PR China: Proceedings of the 5th World Conference on Intelligent Control and Automation, 2004.

Taropa, E. Srini, V.P., Wong-Jong Lee and Tack-Don Han. Data fusion applied on autonomous ground vehicles. The 8th International Conference on Advanced Communication Technology, 2006.

V Murthy, Peter Cao, Masoud G, Sharad Kansara, Sherry Liao, Shince Francis, Manoj Chawla, Maurice Tedder, Dinesh Kumar, Parthiban Swami , Pravin C., Balaji S, Arunkumar C , Ismail Raja and Sousma Allahaja. "BearCart III." Intelligent Ground Vehicle Competition. 01 OCT 2006. University of Cincinnati. 09 OCT 2006. Available from: http://www.igvc.org/design/reports/dr8.pdf.

Appendices

[A] Requirements Specifications

[B] Project Plan

[C] MATLAB Code

Appendix A- Project Plan

Autonomous Ground Vehicle
PROJECT PLAN
09 OCTOBER, 2006

Purpose: The purpose of this project is to design and build an autonomous ground vehicle to compete in the Intelligent Ground Vehicle Competition. A fully autonomous vehicle will compete in three areas: navigation, obstacle avoidance, and inspection. The navigation competition requires the vehicle to maneuver around obstacles while moving from waypoint to waypoint, then back to the start area. The obstacle avoidance competition has the vehicle maneuver around obstacles while staying on the "road," white lines in the grass. The final competition, the design, is an inspection to ensure the vehicle meets specific design requirements.

Organization

Project Sponsor: C2CEN, TISCOM, LT Everett

Project Advisor(s): CAPT Hartnett, LCDR Godfrey

Team Member(s): 1/C Tomas Laucys, 1/C Adrian Ulanoff, 1/C Nathan Morello

Goal

The goal of the Autonomous Ground Vehicle project is to have a fully functional autonomous ground vehicle to compete in the 15th annual Intelligent Ground Vehicle Competition in Rochester, Michigan.

Technical Challenges

Challenge 1: System Integration

- The challenge of integrating the GPS, laser rangefinder, and video feeds will be handled mostly by LT Everett. It will include sending data from the sensors to the laptop, then interpreting the data, then sending back the results of the interpretation.

Challenge 2: Autonomous Control

- This challenge will require a program that will run the vehicle depending on the surrounding area. It will be based upon the work done last year by ENS White on controlling the autonomous blimp.

Challenge 3: Robot Build/Design

- This challenge will require the team to design and build a robot that can be used during testing phases first semester. It will be based upon the designs found from past projects that worked well.

Action Items/Responsibilities

This should be a specific list of all the tasks that need to be done this year to accomplish your goal(s). Each item should include who is responsible for the item whether a specific team member, the entire team, a contractor, project advisor or the sponsor, etc. Don't forget to include required class submissions.

1. Interface all sensors with LT Everett's program. Entire Team.
2. Purchase Image Acquisiton Toolbox for Matalb. 1/C Morello
3. View live video in Matlab and perform Hough Transform for edge detection. 1/C Morello
4. Build a temporary robot for testing. 1/C Morello
5. Test line detection on football field. 1/C Morello
6. Perform obstacle detection in Matlab using SICK laser range finder. 1/C Laucys
7. Design and construct portable power source for the robot components. 1/C Laucys
8. Design robot motion control software. 1/C Laucys

Timeline

This should be a specific list that states the date when each of the above items will be completed. Some items are included below.

-
- Nov ___: Draft Fall Paper due to Project Advisor.
- Dec ___: Fall Presentation
- Dec ___: Fall Paper/Project Notebooks due.
- May ___: Spring Project Presentation
- May ___: Spring Paper/Project Notebooks due.

Project Members:

_____ Date: _____

_____ Date: _____

_____ Date: _____

Project Advisor:

_____ Date: _____

Appendix B- Requirements Specifications

Autonomous Ground Vehicle
Requirements Specification
22 October, 2006

Project Sponsor: C2CEN, NAVCEN, LT Everett
Project Advisor(s): CAPT Hartnett, LCDR Godfrey
Team Member(s): 1/C Tomas Laucys, 1/C Adrian Ulanoff, 1/C Nathan Morello

Background: The Intelligent Ground Vehicle Competition has existed since 1992. Since then, numerous schools and universities have participated in the competition and have allowed their students to learn, design and implement the skills they have gained in their respective undergraduate and graduate programs. The Unites States Coast Guard Academy has decided to participate in the competition for the first time.

This project is of great interest because the education and technology it presents relates to a wide variety of fields, particularly electrical engineering. In addition, not only does this project arise great interest in an education manner, but its practicality for society is what stimulates a curiosity in all minds.

The technology of an autonomous ground vehicle will, in the future, create all modes of transportation in a safer fashion by removing an actual human being from being the operator of every machine. In addition, this can prevent the loss of lives by creating all kinds of military crafts that do not need humans to control them. The autonomous ground vehicle shows a very promising future. In the previous competitions, teams have achieved remarkable results. However, every team strives to achieve better results than previous years.

Problem Statement: The objective of our project is to build an autonomous ground vehicle to compete in the 15th Annual Intelligent Ground Vehicle Competition (IGVC) in Rochester, Michigan.

Requirements:

Level	Sponsor Need	Engineering Requirement	Justification
1	Design	Must be a ground vehicle propelled by direct mechanical contact to the ground.	It is required to qualify to participate in the competition.
2	Vehicle Dimensions		They are required to

			qualify to participate in the competition.
2a	Length	3ft to 7ft inclusive	
2b	Width	2ft to 5ft inclusive	
2c	Height	Less than or equal to 6ft	
3	Propulsion	Vehicle power must be generated onboard.	If the propulsion system wasn't independent, the ground vehicle wouldn't be autonomous.
4	Mechanical E-stop		Besides being a criterion to qualify for the competition, the vehicle would need this if the wireless e-stop fails.
4a	Construction	The E-stop button must be a push to stop.	
4b	Color	Red	
4c	Diameter	1"	
4d	Location	Center rear of vehicle	
4e	Height	At least 2 ft at most 4ft above the ground	
4f	Implementation	Vehicle E-stops must be hardware based and not controlled through software	
4g	Operation	Activating the E-Stop must bring the vehicle to a complete stop.	
5	Wireless E-stop	Must be wireless	If the vehicle is not within reach of any human contact, the vehicle must have the ability to be stopped at any time.
5a	Range	At least 50 ft	
5b	Implementation	Vehicle E-stops must be hardware based and not controlled through software.	
5d		Activating the E-Stop must bring the vehicle	

		to a complete stop.	
6	Max Speed	For safety, a maximum vehicle speed of five miles per hour will be enforced.	Besides being a criterion to qualify for the competition, this speed ensures the safety for the vehicle and its surroundings.
6a	Implementation	All vehicles must be hardware governed not to exceed this maximum speed.	
7	Payload		Each vehicle will be required to carry a payload to demonstrate the vehicle's design can withstand the extra load and still perform its tasks.
7a	Weight	20 lb	
7b	Length	18"	
7c	Width	8"	
7d	Height	8"	
8	Lane Following	The vehicle must demonstrate that it can detect and follow lanes.	Since the course will consist of a marked path of white chalk on grass, the vehicle must follow the prescribed path to reach the target (end) point of the course.
9	Obstacle Avoidance	The vehicle must demonstrate that it can detect and avoid obstacles.	Since the course will have several different objects along the path of the course, the vehicle must have some means of finding a path around those obstacles.
10	Waypoint Navigation	Vehicle must prove it can find a path to a single 2 meter navigation waypoint.	In order for the vehicle to be autonomous, it must be able to navigate to any prescribed point in a course within a specified degree of

			accuracy.
11	Weather Proof	The competition will take place in the event of light rain or drizzle but not in heavy rain or lightning.	Since the competition will be held in an environment where weather can create a hindrance on electrical and mechanical systems, the vehicle must be able to endure the elements and complete all of its tasks simultaneously.

References: The official IGVC rules can be found at http://www.igvc.org/rules.html

Appendix C- Support Plan

Autonomous Ground Vehicle
SUPPORT PLAN
November 22, 2006

Purpose: The objective of our project is to build an autonomous ground vehicle to compete in the 15th Annual Intelligent Ground Vehicle Competition (IGVC) in Rochester, Michigan. In order to qualify and participate, the autonomous ground vehicle must follow the design and functionality requirements specified in the rules for the competition. Once the judges have determined the autonomous ground vehicle has met the requirements, the group may commence with the rest of the competition which consists of an autonomous challenge, a navigation challenge, and a design challenge.

Design Solution: The basic composition of design solution consists of the construction of the vehicle, the implementation of the sensors, and the autonomous control system.

For the construction of the vehicle, the objective is to ensure it is built according to the requirement specifications laid out in the competition rules. For the power supply of the vehicle, a “black box” type unit with a modular wiring system will be constructed. The wiring system will ultimately lead to a single weather-proof outlet on the outside of the robot. The purpose of this is to provide the capability of recharging the batteries necessary for fueling the laptop, different sensors, and motors of the vehicle. Another aspect for the construction of the vehicle is the mechanical and wireless e-stops. The mechanical e-stop will be constructed according to the requirements specifications mentioned in the competition rules. For the wireless e-stop, a remote control having a single toggle switch will be constructed. The wireless e-stop system will be implementing HF receivers. When the toggle switch is manipulated, a relay switch on the vehicle will be energized to either connect or disconnect depending on the state it was previously at. This action will either provide or cease power to the motors. The next concern with respect to constructing the vehicle is satisfying the max speed requirement. To accomplish this, the diameters of the gears will be adjusted to allow the vehicle to accelerate at a maximum speed of 5 mph given that the vehicle is completely built and is able to carry the required payload.

The next section of the design composition is the implementation of the sensors. The first objective of this section is to incorporate all the sensors into MATLAB. The sensors that will be utilized for this vehicle are DGPS, electronic compasses (e-compasses), digital cameras, and a RTK laser. Two cameras will be placed on the left and right side of the vehicle and used for lane detection. One camera will be placed in the middle–front of the vehicle and used for observing obstacles. The data retrieved from the cameras on the sides of the vehicle will be filtered by the Hough transform (in MATLAB) to detect the necessary lines for the vehicle to follow. With the RTK laser, the data retrieved from this sensor will be entered into MATLAB and used to detect obstacles within a specified range and act accordingly if obstacles are too close or are along the

path to a targeted destination (waypoint). For the rest of the sensors, the DGPS and e-compasses, the data retrieved from these sensors will be entered into MATLAB to identify and compare location (latitude and longitude), heading, and speed of the vehicle.

The last component of the design solution is the autonomous control system. The autonomous system requires the development and implementation of intelligent and stable fuzzy logic controllers for the purpose of steering and speed control of the ground vehicle. The autonomous control system will be comprised of four distinct fuzzy logic controllers. There will be a controller for driving, braking, steering, and simultaneous operation of steering and driving. If the autonomous system is developed correctly and as intended, the results should verify that implementing fuzzy logic controllers not only provide acceptable lateral and longitudinal control of the autonomous ground vehicle, but also excel in tracking accuracy, steady-state error, control chatter, and robustness of this control system.

The manufacturability criteria factored the most into the decision because it looks at the design of the solution. It enabled the team to be creative, discover any potential conflicts, and come up with an optimal design solution. Due to the manufacturability criteria, it supports the importance of implementing fuzzy logic controllers for the autonomous ground vehicle. Through extensive research, it has been noted that fuzzy logic controllers guarantee stability, uncouple steering control from speed control, and provide easy incorporation of braking while facilitating the incorporation of control heuristics.

Support Considerations:

Economic:

The project's funding source is C2CEN and NAVCEN. The project's budget consists of $10,000 provided from distributed funds from C2CEN and NAVCEN. Each sponsor provided $5,000 toward the project. In addition to these funds, the project has a smaller but unknown amount of funds provided by USCGA. So far, these funds have provided for the acquirement of the MATLAB image acquisition toolbox, the MATLAB fuzzy logic control toolbox, and additional batteries. The money must be spent by the end of the project or completion of it which is around April 2007. The project does not have to compete for funding.

Manufacturability:

The design will be built by constructing the multiple systems comprising the autonomous ground vehicle, followed by integrating all the systems into the final product. The autonomous ground vehicle's systems will consist of the robot itself, the sensors, power supply unit, and the software (the brain). The wireless e-stop is not another system, but it is important to note, since the judges will have this remote control device to use at their disposal. Pertaining to the robot, the platform was obtained from the mechanical engineering department of USCGA. This platform consisted of the main chassis that is comprised of the four wheels, the two motors, a battery, gears, and connections between the power supply (motors and battery) and the wheels. Ultimately, the robot will be comprised of three dependent systems. Among the software system of the project, there will be four subsystems (four distinct controllers) which autonomously control the vehicle's motion. There are no specific concerns associated with the building design as our group is building the project ourselves.

Sustainability and Reliability:

The sponsor's sustainability and reliability requirement is to build an autonomous ground vehicle that will be able to participate in future competitions and allow for other teams who might want to enter the competition to modify and improve upon the design. The type of spare parts required to maintain this reliability is batteries, wheels, and gears. As far as the number of spare parts to have necessary for emergencies, it is recommended the following are on hand: three 12 volt batteries, six 9 volt batteries, two sets of linkage, and four wheels. These spare parts will be stored in the Electrical Engineering lab.

Life-Cycle Costs:

Life-Cycle costs are not applicable to the project since our project is not intended for use by the Coast Guard. The sole purpose of our project is to have a complete product that will qualify and participate at a competition.

External Considerations:

Environmental:

The design has a minimal impact on the environment. The batteries utilized for the vehicle during the design and testing phases will be discarded appropriately in

accordance with Coast Guard HAZMAT regulations. In addition, rechargeable batteries are implemented as another source of power. There is no need to dispose of our equipment at the end of its serviceable life since our vehicle should be able to last indefinitely. To be specific, its life expectancy should exceed the duration of the competition. However, if one wishes to disassemble the autonomous ground vehicle, all the parts of the robot are either recyclable or reusable. While carrying out all of these cautions mentioned above, these actions are in accordance with the Federal and State regulations (EPA) that preserve the environment.

Health and Safety:

The design has a minimal to no impact on the user and public's health and safety. In regards to health and safety, the vehicle has a mechanical and a wireless e-stop, a max speed, and a weather proof requirement. The wireless e-stop would be used if the vehicle is not within reach of any human contact. The justification for having the e-stops is so that the vehicle has the ability to be stopped at any time. The vehicle would need the mechanical e-stop if the wireless e-stop fails. The max speed of 5 mph ensures safety for the vehicle and its surroundings. Lastly, the weather proof requirement is necessary for the user and the public's health and safety as the vehicle may be operating in weather conditions of light rain or drizzle. Water could have an adverse and dangerous effect on the mechanical and electrical systems of the vehicle. Our design complies with Federal and State regulations by requirement specifications governed by the competition rules which ensure the health and safety for the user and the public. There is no special training required for users to understand our solution's health and safety impact.

Ethical, Social & Political:

Future application or use of this vehicle may have a purpose for use in the military fields and in civilian transportation. Therefore, it is important to consider the possibilities of the social, ethical, and political impacts that our design will have on everyone's lives.

Software Considerations:

Software Considerations is not applicable to our project because there is no private or personal information that is disclosed when entered into our autonomous ground vehicle. The only information required by the autonomous ground vehicle are destinations (navigation waypoints) which it must know in order to attempt to reach those prescribed target locations.

Appendix D- MATLAB Code

SICK LASER CODE—

start_sick.m

```
commport = 'COM1';
baud = 9600;

% This function starts the SICK laser and places it into continous
% scanning mode.

% Construct telegram to request an acquisition
startmsg = [02 00 02 00 32 36 52 08];

sick_port = serial(commport,'BaudRate', baud);
set(sick_port,'InputBufferSize',900);
fopen(sick_port);

% Flush the serial buffer on the sick port
if sick_port.BytesAvailable
    fread(sick_port,len,'uint8');
end

% Write start telegram to Sick laser to start continous scanning
fwrite(sick_port,startmsg,'uint8','async');
delay(sick_port,0);

hold on;
while (2 < 3)
    scandata = get_scan(sick_port);
    clf;
    plotscan2(scandata);
    pause(.75);
    drawnow;
end
```

get_scan.m

```
function scandata = get_scan(sick_port)

% This function dumps the current serial buffer to a variable, parses
% a scan output, converts measurement data, and outputs a vector of
% 181 ranges.

clear scandata;
clear msg;

% Read the contents of the serial buffer
delay(sick_port,0);
if sick_port.BytesAvailable
    msg = fread(sick_port,sick_port.BytesAvailable,'uint8');
else
    msg = [];
end
flushinput(sick_port);
```

```
msg = msg';
bytecount = 369;
for n = 6:(900-3);
    if (msg(1,n) == 2) && (msg(1,n+1) == 128) && (msg(1,n+2) == 110)
        if (n+bytecount) <= (900-3);
            scan(1,:) = msg(1,(n+7):(n+7+bytecount));
            break;
        else
            scandata = [];
            break;
        end
    end
end
```

plotscan_buffer.m

```
degree = 1;
for n = 1:2:361
        scandata(1,degree) = scan(1,n) + (scan(1,n+1) * 256);
    degree = degree+1;
end
scandata = scandata';

function plotscan_buffer(scandata)

% This function plots a single scan of data from the Sick Laser
hold on;
angle = 1:181;
angle = angle';
rdir = angle * pi/180;
for n = 1:327
    [x y] = pol2cart(rdir, scandata(:,n));
    fill(x,y,'g');
    axis([-8500 8500 -500 8500]);
    k = strcat('Plot of Sick Data - Scan: ', num2str(n));
    title(k);
    xlabel('Distance - mm');
    ylabel('Distance - mm');
    pause(.25);
    drawnow;
    clf;
end
hold off;
end
```

GPS CODE

GPS.m

```
clear;

disp('Trying to connect to sensor...');
h = serial('COM1');
h.baudrate = 38400;
set(h, 'Parity', 'odd');
h.terminator = 'CR/LF';
```

```
fopen(h);
disp('Connected to Heading Sensor...');

%fwrite(h, '*S');

headingdata = {};
i = 1;

disp('Recieving data:');

while (1)
    tic;

    flushinput(h);
   % fwrite(h, '*Q');
    [data,null] = fscanf(h, '%s');

    headingdata{i,1} = data;
    disp(headingdata{i,1});
    a = toc;

    headingdata{i,2} = a;
    i = i + 1;
end;
```

ROBOT CONTROL CODE

robot_turn.m

```
function varargout = robot_turn(varargin)
% ROBOT_TURN M-file for robot_turn.fig
%      ROBOT_TURN, by itself, creates a new ROBOT_TURN or raises the existing
%      singleton*.
%
%      H = ROBOT_TURN returns the handle to a new ROBOT_TURN or the handle to
%      the existing singleton*.
%
%      ROBOT_TURN('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in ROBOT_TURN.M with the given input
arguments.
%
%      ROBOT_TURN('Property','Value',...) creates a new ROBOT_TURN or raises
the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before robot_turn_OpeningFunction gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to robot_turn_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES
```

```
% Edit the above text to modify the response to help robot_turn

% Last Modified by GUIDE v2.5 09-Dec-2006 10:01:50

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @robot_turn_OpeningFcn, ...
                   'gui_OutputFcn',  @robot_turn_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before robot_turn is made visible.
function robot_turn_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to robot_turn (see VARARGIN)

% Choose default command line output for robot_turn
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes robot_turn wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = robot_turn_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
```

```
function act_head_Callback(hObject, eventdata, handles)
% hObject    handle to act_head (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of act_head as text
%        str2double(get(hObject,'String')) returns contents of act_head as a
double


% --- Executes during object creation, after setting all properties.
function act_head_CreateFcn(hObject, eventdata, handles)
% hObject    handle to act_head (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function des_head_Callback(hObject, eventdata, handles)
% hObject    handle to des_head (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of des_head as text
%        str2double(get(hObject,'String')) returns contents of des_head as a
double


% --- Executes during object creation, after setting all properties.
function des_head_CreateFcn(hObject, eventdata, handles)
% hObject    handle to des_head (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
function difference_Callback(hObject, eventdata, handles)
% hObject    handle to difference (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of difference as text
%        str2double(get(hObject,'String')) returns contents of difference as
a double


% --- Executes during object creation, after setting all properties.
function difference_CreateFcn(hObject, eventdata, handles)
% hObject    handle to difference (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

robot_turn_control.m

```
function robot_turn_control

%Enter robot_turn_control to run this equation.
%
%Function Name:     robot_turn_control
%
%Parameters:        none
%
%Returns:           none
%
%Description:       This function is the callback for the robot_turn GUI.
%                   It reads in the desired heading from the GUI, the
%                   actual heading from the compass, and will turn the
%                   vehicle until the headings are less than 1 degree
%                   apart.  As it runs it continuously updates the GUI
%                   display with the actual heading and difference between
%                   the actual and desired headings.
%


% Initialize the output to the D/A converter
ao = analogoutput('mcc',0);
chans = addchannel(ao,0:1);
set(ao,'SampleRate',500);
```

```
% Put the value of 2.9 volts on each motor, stopping each from running
putdata(ao,[2.9 2.9]);
start(ao);

% Read in the compass heading
act_head=heading_nate

% Set the handles for the text boxes in the GUI
h_des=findobj('Tag','des_head');
h_act=findobj('Tag','act_head');
h_diff=findobj('Tag','difference');

% Set the actual heading text box to the value read
set(h_act,'String',act_head);

% Get the actual value from the string in the text box
actual=get(h_act,'String');
% Get the desired value from the string and change it to a number
desired=str2num(get(h_des, 'String'));
% Change the actual value from cell format to matrix format, then change
%   the string value in the matrix to a number.
act=cell2mat(actual);
actu=str2num(act);

% Solve for the difference between the actual and desired headings. Set the
%   string in the GUI for the difference to the correct value
difference=actu-desired;
diff=num2str(difference);
set(h_diff,'String',diff);

% Find which direction the robot should turn depending on the difference
%   between the headings.
if (difference< 180) & (difference > -180)
    if difference >0
        turn = 'right';
        turn_val=difference;
    else
        turn = 'left';
        turn_val=abs(difference);
    end
else
    if difference> 0
        turn = 'left';
        turn_val=360-difference;
    else
        turn = 'right';
        turn_val=360+difference;
    end
end

% Execute the while loop when the difference in headings is greater than 1
%   degree.
while (abs(difference)>1)
    % Read in the heading and display in the GUI.
    actual=heading_nate();
```

```
    set(h_act,'String',actual);
    % Manipulate the heading so it is a number
    act=cell2mat(actual);
    actu=str2num(act);
    % Find the difference and display it in the GUI.
    difference=actu-desired;
    diff=num2str(difference);
    set(h_diff,'String',diff);
    % Put the correct voltages onto each motor depending on which way the
    %   robot should turn
    if (strcmp(turn,'right'))
        putdata(ao,[2.5 3.3])
        start(ao);
    else
        putdata(ao,[3.3 2.5])
        start(ao);
    end
end

% Stop the robot from moving.
putdata(ao,[2.9 2.9]);
start(ao);

%close the analog output handle.
stop(ao);
```

heading_nate.m

```
function [actualhead] = heading_nate

%Enter heading_nate to run this equation.
%
%Function Name:     heading_nate
%
%Parameters:        none
%
%Returns:           actualhead
%
%Description:       Connects to the compass in the serial port and reads
%                   the NEMA string.  Returns the second value in the
%                   string, which is the heading in degrees.
%

clear;

% set handle to the serial port and set initial port conditions
 h = serial('COM1');
 h.baudrate = 19200;
 h.terminator = 'CR/LF';

% open the port
 fopen(h);

% flush the data to remove errors
```

```
 flushinput(h);

% read in the NEMA string from the compass
 [data,null] = fscanf(h, '%s');

% read the result of the NEMA string and return the value in the second cell
result = strread(data,'%s',-1,'delimiter',',');
actualhead = result(2);

% closes the port
fclose(h);
```

runrobot.m

```
function varargout = runrobot(varargin)
% RUNROBOT M-file for runrobot.fig
%      RUNROBOT, by itself, creates a new RUNROBOT or raises the existing
%      singleton*.
%
%      H = RUNROBOT returns the handle to a new RUNROBOT or the handle to
%      the existing singleton*.
%
%      RUNROBOT('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in RUNROBOT.M with the given input arguments.
%
%      RUNROBOT('Property','Value',...) creates a new RUNROBOT or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before runrobot_OpeningFunction gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to runrobot_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help runrobot

% Last Modified by GUIDE v2.5 15-Dec-2006 13:05:46

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @runrobot_OpeningFcn, ...
                   'gui_OutputFcn',  @runrobot_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
```

```
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before runrobot is made visible.
function runrobot_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to runrobot (see VARARGIN)

% Choose default command line output for runrobot
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes runrobot wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = runrobot_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of
slider


% --- Executes during object creation, after setting all properties.
function slider1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
```

```
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes on slider movement.
function slider2_Callback(hObject, eventdata, handles)
% hObject    handle to slider2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of
slider


% --- Executes during object creation, after setting all properties.
function slider2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end



function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a
double


% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a
double


% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in stop.
function stop_Callback(hObject, eventdata, handles)
% hObject    handle to stop (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3 as a
double


% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in turn.
function turn_Callback(hObject, eventdata, handles)
% hObject    handle to turn (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --------------------------------------------------------------------
function Untitled_1_Callback(hObject, eventdata, handles)
% hObject    handle to Untitled_1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

run_robot.m

```
function run_robot
%Enter run_robot to run this equation.
%
%Function Name:     run_robot
%
%Parameters:        none
%
%Returns:           none
%
%Description:       The callback function for the GUI to control the
%                   movement of the robot. Connects to the D/A card, then
%                   connects to the objects in the GUI, and outputs the
%                   correct voltages based on the level of the slide bars.
%

% Connect to the D/A card, and set parameters
ao = analogoutput('mcc',0);
chans = addchannel(ao,0:1);
set(ao,'SampleRate',500);

% Set the handles for all the objects in the GUI the function will modify
%   or use.
h_left=findobj('Tag','slider1');
h_right=findobj('Tag','slider2');
h_text1=findobj('Tag','edit1');
h_text2=findobj('Tag','edit2');

% Read the value of the left and right slide bars, representing the voltage
%   to the left and right motors.
left_val=get(h_left,'Value');
right_val=get(h_right,'Value');

% Add 1.9 volts to the values to output the correct voltage to the motor to
%   run within the correct speed constraints
left_val = left_val + 1.9;
```

```
right_val = right_val + 1.9;

% Output the voltages to the motors
putdata(ao,[left_val right_val ])
start(ao);

% Display the voltages, level for each slidebar, in the GUI
set(h_text1,'String',left_val);
set(h_text2,'String',right_val);

stop(ao);
```

stop_movement.m

```
function stop_movement
%Enter stop_movement to run this equation.
%
%Function Name:     stop_movement
%
%Parameters:        none
%
%Returns:           none
%
%Description:       Connects to the robot via the D/A converter and puts
%                   voltages equating to stop values on each motor.
%

%set the handles, from the GUI, to the sliders
h_left=findobj('Tag','slider1');
h_right=findobj('Tag','slider2');

%set the value that centers the sliders
middle_val_left=1;
middle_val_right=1;

%put the value onto the sliders
set(h_left,'Value',middle_val_left);
set(h_right,'Value',middle_val_right);

%change the text showing the value of the sliders
h_text1=findobj('Tag','edit1');
h_text2=findobj('Tag','edit2');

show_left=2.9;
show_right=2.9;

set(h_text1,'String',show_left);
set(h_text2,'String',show_right);

% call the run_robot function to output the voltages to the motors.
run_robot;
```

turn_robot.m

```
function turn_robot
%Enter heading_nate to run this equation.
%
%Function Name:     heading_nate
%
%Parameters:        none
%
%Returns:           actualhead
%
%Description:       Connects to the compass in the serial port and reads
%                   the NEMA string.  Returns the second value in the
%                   string, which is the heading in degrees.
%

%set the handles, from the GUI, to the sliders
h_left=findobj('Tag','slider1');
h_right=findobj('Tag','slider2');

%set the value that will turn the robot
middle_val_left=.50;
middle_val_right=1.35;

%put the value onto the sliders
set(h_left,'Value',middle_val_left);
set(h_right,'Value',middle_val_right);

%change the text showing the value of the sliders
h_text1=findobj('Tag','edit1');
h_text2=findobj('Tag','edit2');

show_left=2.3;
show_right=3.4;

set(h_text1,'String',show_left);
set(h_text2,'String',show_right);

% call the run_robot function to output the voltages to the motors.
run_robot;
```

Autonomous Ground Vehicle
Test Plan

Synopsis.

Acceptance tests will be conducted in order to verify that the system meets the requirements specification in order to participate in the Intelligent Ground Vehicle Competition (IGVC). The design, the vehicle dimensions, the mechanical E-stop, and the wireless E-stop will be tested.

Personnel.

All of the tests conducted will require only one person.

Procedure.

The only equipment needed to conduct these tests are the vehicle itself, the remote controller, and a measuring tape. In order to test the mechanical and wireless E-stops for the vehicle, the tests must be conducted in an unobstructed environment with a distance of least greater than 50 ft. The preferred testing field would be on the football field since it's free of obstacles and has measurements on the field. It should not take any longer than 15 minutes to conduct the four tests.

Equipment Required:

1. Vehicle-laptop, mechanical propulsion system and a power system (generated onboard) for it
2. Remote controller and battery to supply power for it
3. Measuring tape

Initial Set-up

1. Push the red switch button that is located on the base of vehicle toward the left back corner wheel. This turns the power for the motors on for the vehicle.
2. Push the top middle button to turn the laptop on.
3. Double click on MATLAB icon to open up the MATLAB programming language.
4. Go to the directory of C:\MyDocuments\PECE\RobotControl\
5. Double click on the m-file named ForwardControl.m to open the program which allows the vehicle to drive forward.
6. If user is testing the mechanical or wireless E-stop, to enable the vehicle to drive forward, go to the command window of MATLAB and type "run ForwardControl" then hit enter.

Parameter Required Value

N/A

Test # 1 – Design

Objective:

Verify if vehicle's design is ground based and is solely propelled by a mechanical propulsion system.

Requirements:

Must be a ground vehicle propelled by direct mechanical contact to the ground. The requirement I.D. level is 1.

Test Overview:

The vehicle will be examined first to see identify if the vehicle is powered by a mechanical propulsion system. Then the vehicle will be operated in the forward and backwards directions to verify if vehicle maintains constant and direct contact with the ground at all times. The expected result of this test is the vehicle will successfully move in all directions while remaining on the ground.

Test Procedure:

1. Ensure the vehicle is on an unobstructed field.
2. Lay out a measuring tape for 15 ft.
3. Ensure the vehicle is lined up in parallel with the measuring tape and placed on one end of the tape.
4. Push the red switch button that is located on the base of vehicle toward the left back corner wheel. This turns the power for the motors on for the vehicle.
5. Push the top middle button to turn the laptop on.
6. Double click on MATLAB icon to open up the MATLAB programming language.
7. Go to the directory of C:\MyDocuments\PECE\RobotControl\
8. Double click on the m-file named ForwardControl.m to open the program which allows the vehicle to drive forward.
9. Go to the command window of MATLAB and type "run ForwardControl". Then hit enter. (This enables the vehicle to drive forward.)
10. Once the vehicle is moving forward, wait until the vehicle the end of the measuring tape and then type CTRL + C in the command window of MATLAB to stop the code from running. (This stops the vehicle from moving forward.)

Results:

For successful results, the expectation is that by inspection the vehicle's propulsion system is composed of electromechanical parts and the vehicle remains on ground the entire time it is moving or remaining still.

Test # 2 – Vehicle Dimensions

Objective:

Verify if vehicle's design regarding the length, width, and height requirements are met.

Requirements:

1. Length – 3 ft to 7 ft inclusive.
2. Width – 2 ft to 5 ft inclusive.
3. Height – less than or equal to 6 ft.

The requirement I.D. level is 2.

Test Overview:

The vehicle will be measured with a measuring tape to verify if vehicle dimensions meet the requirements. The expected result of this test is the vehicle will successfully meet the dimension requirements.

Test Procedure:

1. For length, measure one end of the base to the other end (the same side) with a measuring tape. Measured Length = _____
2. For width, measure with a measuring tape the perpendicular distance across the base from one end of the base to the same point on the opposite side of the base.
Measured Width = _____
3. For height, measure with a measuring tape the perpendicular distance from any point on the base to the highest point of the vehicle. Measured Height = _____

Results:

For successful results, the expectation is that after measuring the vehicle's dimensions that these values are within the bounds specified in the requirements.

Test # 3 – Mechanical E-stop

Objective:

Verify if vehicle's mechanical E-stop meets design, implementation and operation requirements.

Requirements:

1. The mechanical E-stop must be a push button
2. Button must be red.
3. Button must be one inch diameter.
4. Button must be located in center rear of vehicle
5. Button must be at a height of at least 2 ft but no more than 4 ft above the ground.
6. The implementation of this E-stop must be hardware based and not controlled through software.
7. Activating the mechanical E-Stop must bring the vehicle to a complete stop.

The requirement I.D. level is 4.

Test Overview:

What will occur in this test is the vehicle will be operated in the forward direction. The tester will keep up with vehicle at a moderate speed of no greater than 5 mph and push the constructed mechanical E-stop button when the vehicle has reached the end 15 ft straight line path. After the button has been pushed, the tester will examine if the button completely stopped the vehicle. The initial conditions will be that the power to the motors is turned on and the laptop is on with the appropriate MATLAB code running to move the vehicle. The expected result of this test is the mechanical E-stop will completely stop the vehicle by shutting off the power to the motors.

Test Procedure:

1. Is the mechanical E-stop a push button? _____
2. Is the button red? _____
3. Is the button one inch in diameter? _____
4. Is the button located in center rear of vehicle? _____
5. Is the button at a height of at least 2 ft but no more than 4 ft above the ground? _____
6. Is the mechanical E-stop solely hardware based? _____
7. Ensure the vehicle is on an unobstructed field.
8. Lay out a measuring tape for 15 ft.
9. Ensure the vehicle is lined up in parallel with the measuring tape and placed on one end of the tape.
10. Push the red switch button that is located on the base of vehicle toward the left back corner wheel. This turns the power for the motors on for the vehicle.
11. Push the top middle button to turn the laptop on.
12. Double click on MATLAB icon to open up the MATLAB programming language.
13. Go to the directory of C:\MyDocuments\PECE\RobotControl\
14. Double click on the m-file named ForwardControl.m to open the program which allows the vehicle to drive forward.
15. Go to the command window of MATLAB and type "run ForwardControl". Then hit enter. (This enables the vehicle to drive forward.)

16. Once the vehicle is moving forward, follow the vehicle and push the red circular push button that is located at the center rear of the vehicle 3 ft from the bottom when the vehicle has reached the end of the measuring tape.
17. When activating the mechanical E-stop, does the vehicle come to a complete stop?
_____

Results:

For successful results, the expectation is the mechanical E-stop button meets the construction requirements and when activated it completely stops the power supply for the motor bringing the vehicle to a complete stop.

Test # 4 – Wireless E-stop

Objective:

Verify if vehicle's wireless E-stop functions at least 50 ft away. It must be solely hardware based and bring the vehicle to a complete stop.

Requirements:

1. It must be wireless.
2. It must function at least 50 ft away.
3. It must be hardware based and not controlled through software.
4. It must bring the vehicle to a complete stop.

The requirement I.D. level is 5.

Test Overview:

What will occur in this test is the vehicle will be operated in the forward direction alongside by a measuring tape set out to 60 ft. The tester will remain at the starting point of where the vehicle began moving forward and operate the remote control to stop the vehicle after it has reached a distance of at least 50 ft. After the remote control switch has been flipped, the tester will examine if the switch completely stopped the vehicle. The initial conditions will be that the power to the motors is turned on and the laptop is on with the appropriate MATLAB code running to move the vehicle. The expected result of this test is the wireless E-stop will completely stop the vehicle by shutting off the power to the motors.

Test Procedure:

1. Is remote controller used for an E-stop wireless? _____
2. Is the wireless E-stop solely hardware based? _____
3. Ensure the vehicle is on an unobstructed field.
4. Lay out a measuring tape for 60 ft.
5. Ensure the vehicle is lined up in parallel with the measuring tape and placed on one end of the tape.
6. Push the red switch button that is located on the base of vehicle toward the left back corner wheel. This turns the power for the motors on for the vehicle.
7. Push the top middle button to turn the laptop on.
8. Double click on MATLAB icon to open up the MATLAB programming language.
9. Go to the directory of C:\MyDocuments\PECE\RobotControl\
10. Double click on the m-file named ForwardControl.m to open the program which allows the vehicle to drive forward.
11. Pull out the antenna on the remote controller which is located on the top-middle of the device.
12. Flip the top-left switch to turn on the remote controller.
13. Go to the command window of MATLAB and type "run ForwardControl". Then hit enter. (This enables the vehicle to drive forward.)
14. Once the vehicle is moving forward, remain at the beginning of the measuring tape where the vehicle started moving.
15. When the vehicle has reached the end of measuring tape (laid out at 60 ft), flip the upper left corner switch, which is directly below the power switch for remote controller.
16. When activating the wireless E-stop, does the vehicle come to a complete stop? _____

Results:

For successful results, the expectation is the remote controller (wireless E-stop) when activated at a distance of at least 50 ft, stops the power supply for the motor bringing the vehicle to a complete stop.

Autonomous Ground Vehicle
Design Specification
April 20, 2007

Purpose:

The purpose of our project is to design and build an autonomous ground vehicle capable of participating in the Intelligent Ground Vehicle Competition (IGVC). In order to qualify for the IGVC, the vehicle must meet requirements in eight criteria: length, width, height, mechanical E-stop, wireless E-stop, max speed, lane following, obstacle avoidance, and waypoint navigation.

Design Solution:

Constructing an autonomous ground vehicle to meet qualification requirements can be divided into three sections: mechanical and propulsion system (vehicle design), autonomous navigation system (sensors), and power systems. The mechanical and propulsion system will consist of an aluminium frame, 4 ATV wheels, and a tank drive system. The power system will consist of 3-12 volt batteries, 1-9 volt battery, 2-5 volt batteries and laptop power. Lastly, the autonomous navigation system will consist of a Dell laptop, a Creative webcam, SICK laser range-finder, Honeywell 3000 electronic compass, shaft encoders, and a Trimble DGPS system.

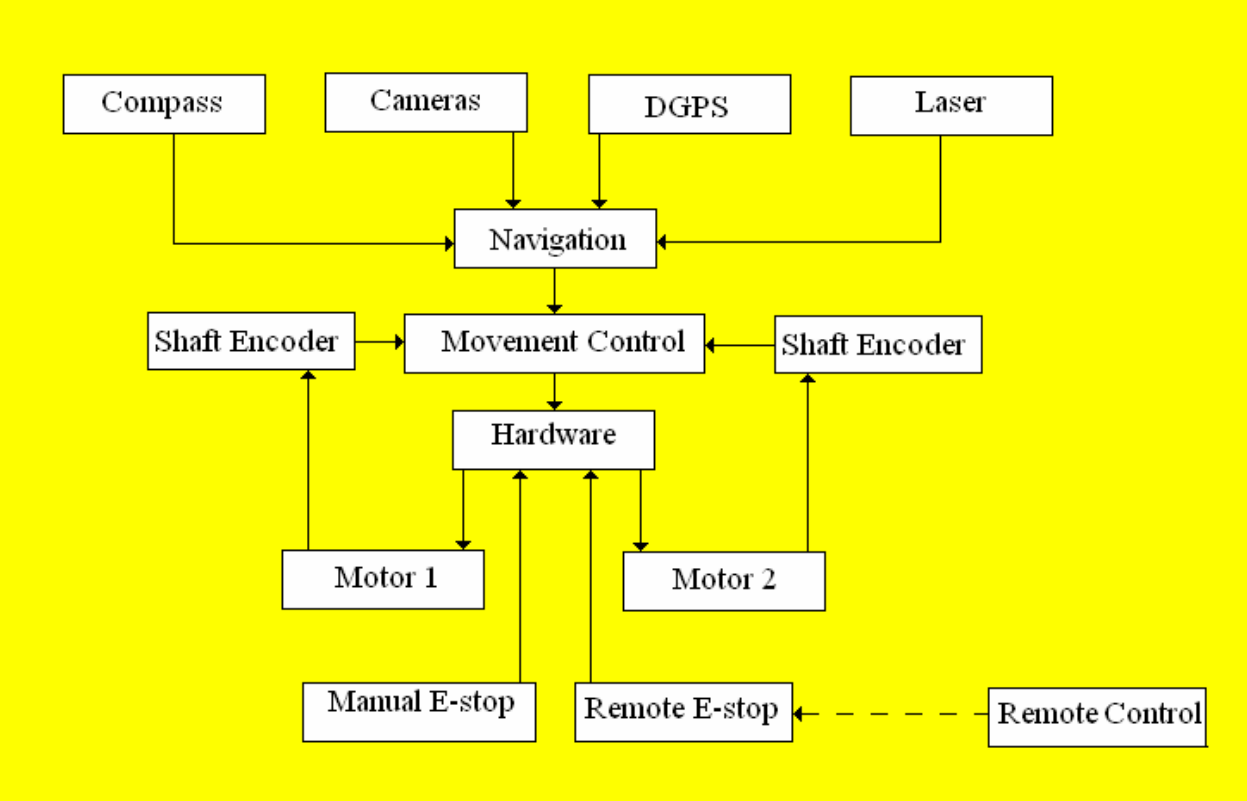

Figure 1- Interaction between the three sections

The electronic compass, cameras, DGPS, and laser are the sensors that aid in the autonomous navigation of the vehicle.

Sensors

For DGPS, the DGPS receiver used for determining the vehicle's position is the Trimble 214 DGPS. MATLAB was used to retrieve the position (longitude and latitude) and the time stamp from its data string. The DGPS outputs its data in the National Marine Electronics Association (NMEA) string. The data type chosen for the NMEA sentence is recommended minimum (RMC).

For the electronic compass, the Honeywell HMR3000 is being used to obtain the robot's heading. This particular electronic compass provides NMEA sentences in three different formats: HDG, HDT, and HPR. The NMEA sentence chosen is the HPR. This data type represents heading, pitch and roll. Furthermore, MATLAB was used to obtain the heading from the HPR NMEA string to provide to the robot so that it could calculate the difference between the actual heading and the desired heading and make the appropriate turn to achieve that desired heading.

Another sensor utilized is a digital camera. One camera will cover a 180° sweep of the robot's front view. The camera will be placed on the center of the vehicle's frame (behind the laser) elevated 2.17 ft above the ground to capture images of vehicle's front view. The data retrieved from the camera is then filtered by the Hough transform (in MATLAB).

With the laser rangefinder, the particular laser utilized is the SICK Laser LMS221. The data retrieved from this sensor is obtained via MATLAB. The data retrieved is a continuous real-time scan of the laser making a 180° sweep of the laser's front view. This data essentially looks like an upper semi-circle that is uniquely deformed depending on the environment the laser is viewing. The factors that affect the laser' scan are an obstacles size, distance from laser, and location.

To gather the wheel rotational speed and direction of movement, two simple shaft encoders were built and programmed using MATLAB. Three magnetic sensors were placed in the side of the vehicle next to each of the front wheels. The permanent magnet is mounted in the sprocket of the wheel. All sensors were connected to digital I/O pins on the A/D converter (PC board). The three sensors were labeled as a distinct variable with one sensor used to monitor the distance, speed, and rpm by calculating the time it took the magnet to come close in contact with it. The other two sensors were used only for determining which direction the shaft is rotating to display if the vehicle was moving forwards or backwards. MATLAB code interprets the data received from the A/D converter and converts it into speed over ground, distance traveled and direction. The shaft encoder sensor assembly can be seen in figure 2.

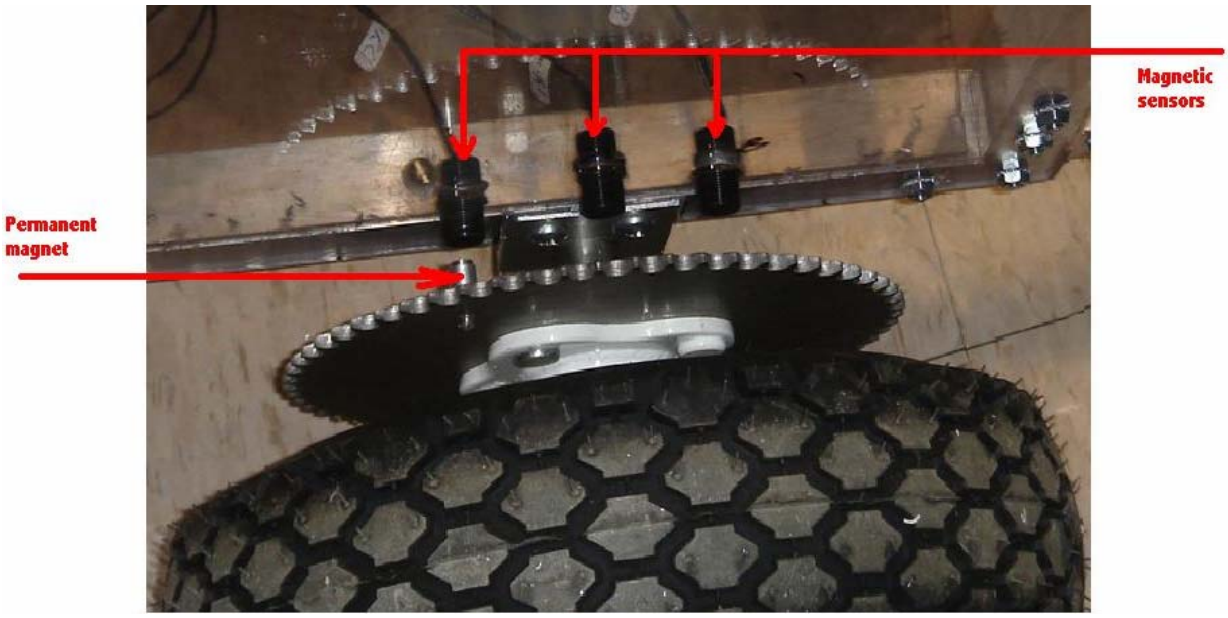

Figure 2- Shaft encoder assembly

Vehicle Design

The main consideration with regards to the vehicle was that it should be easy to install equipment, and have compartments for all the equipment that we need. The first step was to get a rough drawing of the vehicle, to see what it might eventually look like. After this was done, dimensions were added based on the IGVC requirements and what we required for size. Once dimensions were added, the vehicle was built in Solidworks using 1”x1” square tubing. The dimensions used were 3’x4’. This would provide a vehicle that is within the size constraints, but also give ample space for equipment. After designing this vehicle, a second was designed that was 2’x3’. This vehicle also meets the requirements and gives us space for all equipment, but it uses less material. Figure 3 - IGVC Vehicle Design shows the 2’x3’ vehicle frame from Solidworks.

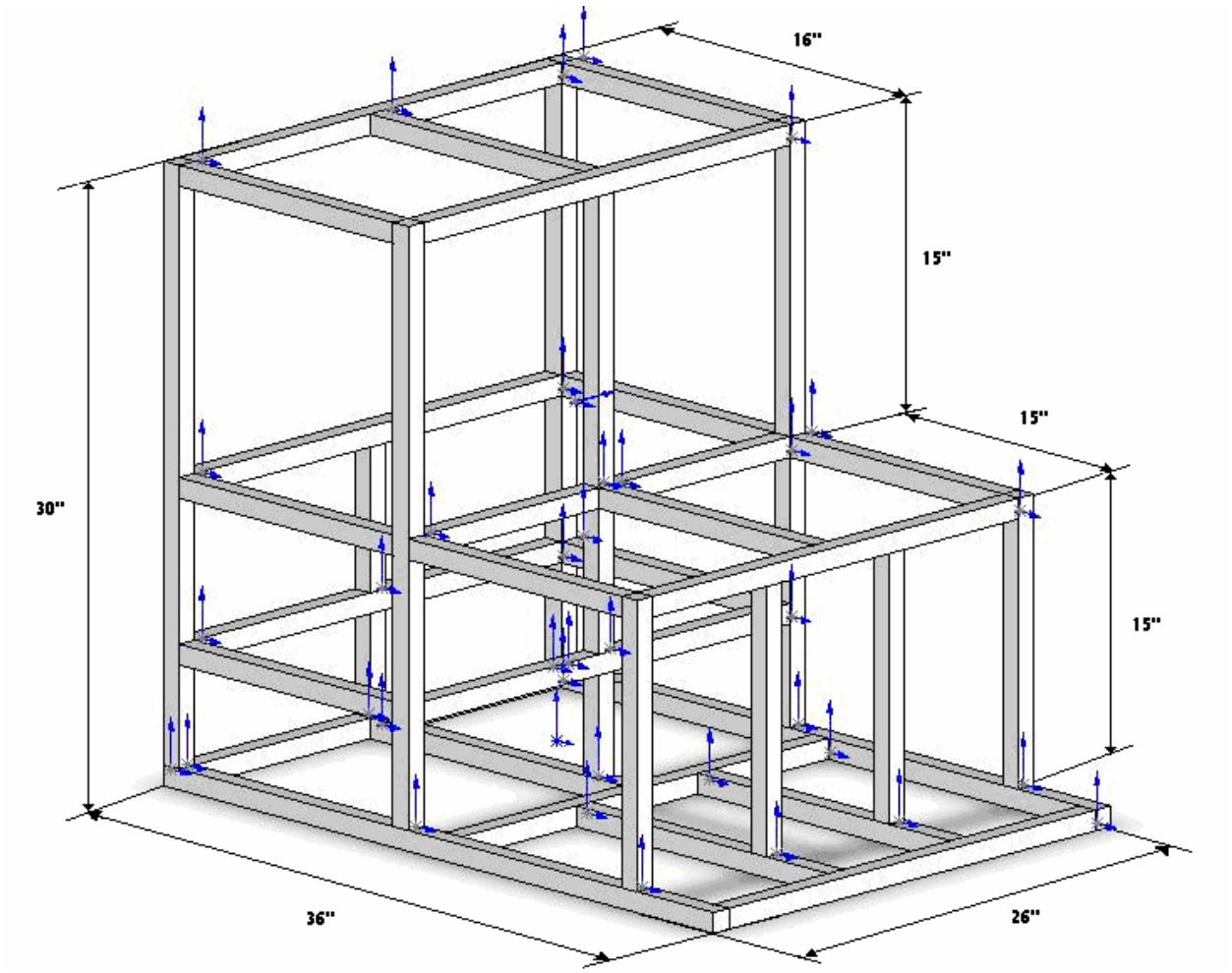

Figure 3 - IGVC Vehicle Design

This design has several advantages for IGVC use. On the front, there is a lip for the laser to mount on, with two vertical tubes in the center for the laser to be bolted to. The compartment behind the laser mount is for batteries, and the charging system. On the back of the vehicle there are 3 compartments on top of each other. The lowest is for the motors, and the First Robotics controllers and systems. The second is for the computer to motor and sensors interface, DGPS receiver, serial connection box, and other equipment that the operator does not have to access to get the machine running. The top compartment is for the laptop computer and switch panel. Here the operator will be able to make changes as necessary and start the operation of the vehicle. The compartments were designed this way to simplify equipment installation as well as make the building of the vehicle simpler.

After designing the frame, the wheel mounts, and antenna mounts had to be designed. Figure 4 shows a completed vehicle.

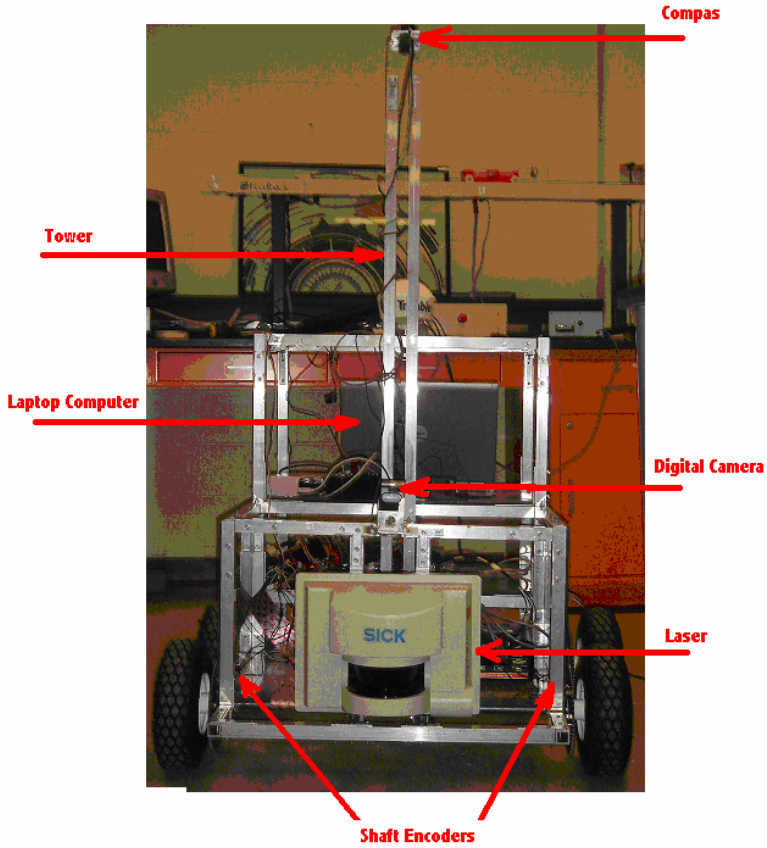

Figure 4 – Front View of the Robot

The electronic compass had to be mounted as far from motors and shafts as possible due to the interference they created. This led to a tower being designed that is secured to the frame in several spots but is screw mounted so it can be removed if necessary. The tower can be seen in the front view of the robot in figure 4.

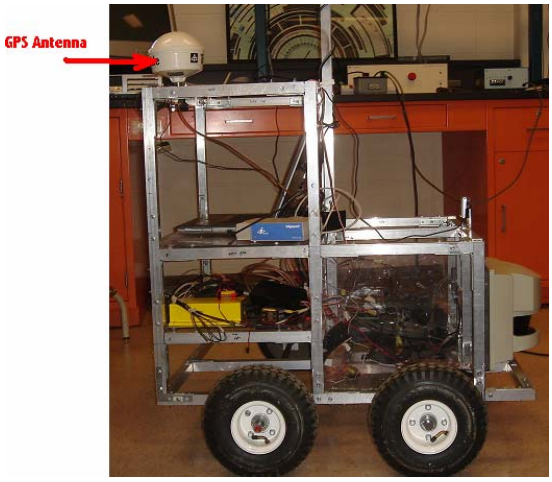

Figure 5 – Side View of the Robot

The DGPS antenna is mounted on the back part of the vehicle and can be seen in figure 5. The wheels are mounted so as to provide the ability to have a zero-turning radius, in a tank drive style system. They can be seen in side picture of the robot in figure 5.
The propulsion system for the vehicle is a simple one. It is a tank dive system, with two motors coupled together on each side. The two motors per side give more torque so the vehicle should be able to move quicker. The system is set up so that the motors deliver power to all the wheels. This will minimize the turning radius of the vehicle, thus providing better navigation. The motors are situated behind the back wheel so the chain runs forward to the back wheel sprocket, then forward to the front wheel sprocket.

Power Systems

Since the sensors, devices, and motors required different voltages, two power systems were created. The decision was made to split the power system into two was based off a couple of factors. First, the electrical or mechanical disturbance from the power grid connected to the motors could interfere with the continuous operation of the sensors. For example, the start of the motors will cause a voltage drop which could introduce an anomaly to the system. We also anticipated an implementation of the emergency stops and predicted that a simultaneous cutoff of power for the motors and sensors would result in a lengthy reset of the laptop and all of the sensors. Two separate power grids eliminate this problem and allow us to stop the robot without turning of the sensors. There are no concerns with the power budget since all of the distinct batteries are adequately provided by the electrical engineering and mechanical engineering departments.
The power system for the sensors uses two 12 volt batteries and three 9 volt batteries. The laser rangefinder requires 24 volts to operate while the DGPS used 12 volts to function. In addition, one 9 volt battery is used for each e-compass and the wireless camera while the web cameras are powered by the laptop power. The second power system is for the drive train and control of the robot. This system requires a 12 volt battery to power two motors and a motor controller. Figure 6 shows the power supply we will build for the robot.

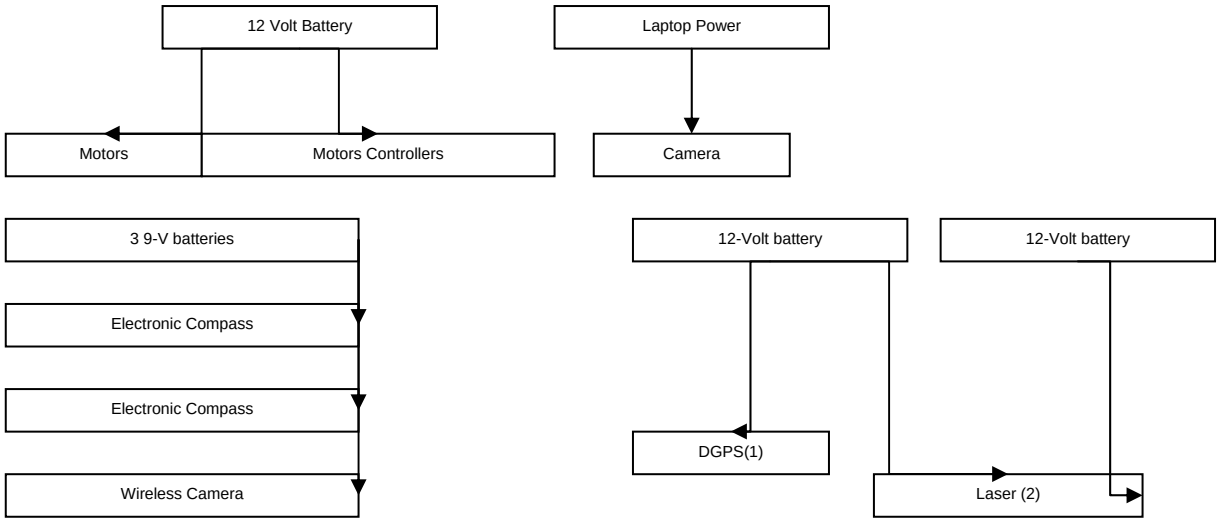

Figure 6 - Power System for the Robot

The power supply of the vehicle, we eventually consist of a “black box” type unit with a modular wiring system. The wiring system will ultimately lead to a single weather-proof outlet on the outside of the robot. The purpose of this is to provide the capability of recharging the batteries necessary for powering the laptop, different sensors, and motors of the vehicle.

Wireless E-stop

The key component in the emergency e-stop system is an electric relay that cuts the power for the motors. When the motors are of the resistance of the gears stops the movement of the vehicle. The electric relay is controlled by an electrical circuitry (figure 7) that consists of the two transistors and an operational amplifier. The 12 volt sensor battery provides power for the e-stop circuitry. The circuitry receives a logic signal from the Futaba analog servo circuitry (0.04-5V). A wireless link is established using the Futaba remote control unit and the Futaba FM receiver that work for the distances greater than 50 feet and thus meet a design requirement. The receiver is powered by 5 volt rechargeable battery. The remote control unit has an internal rechargeable 12 volt battery.

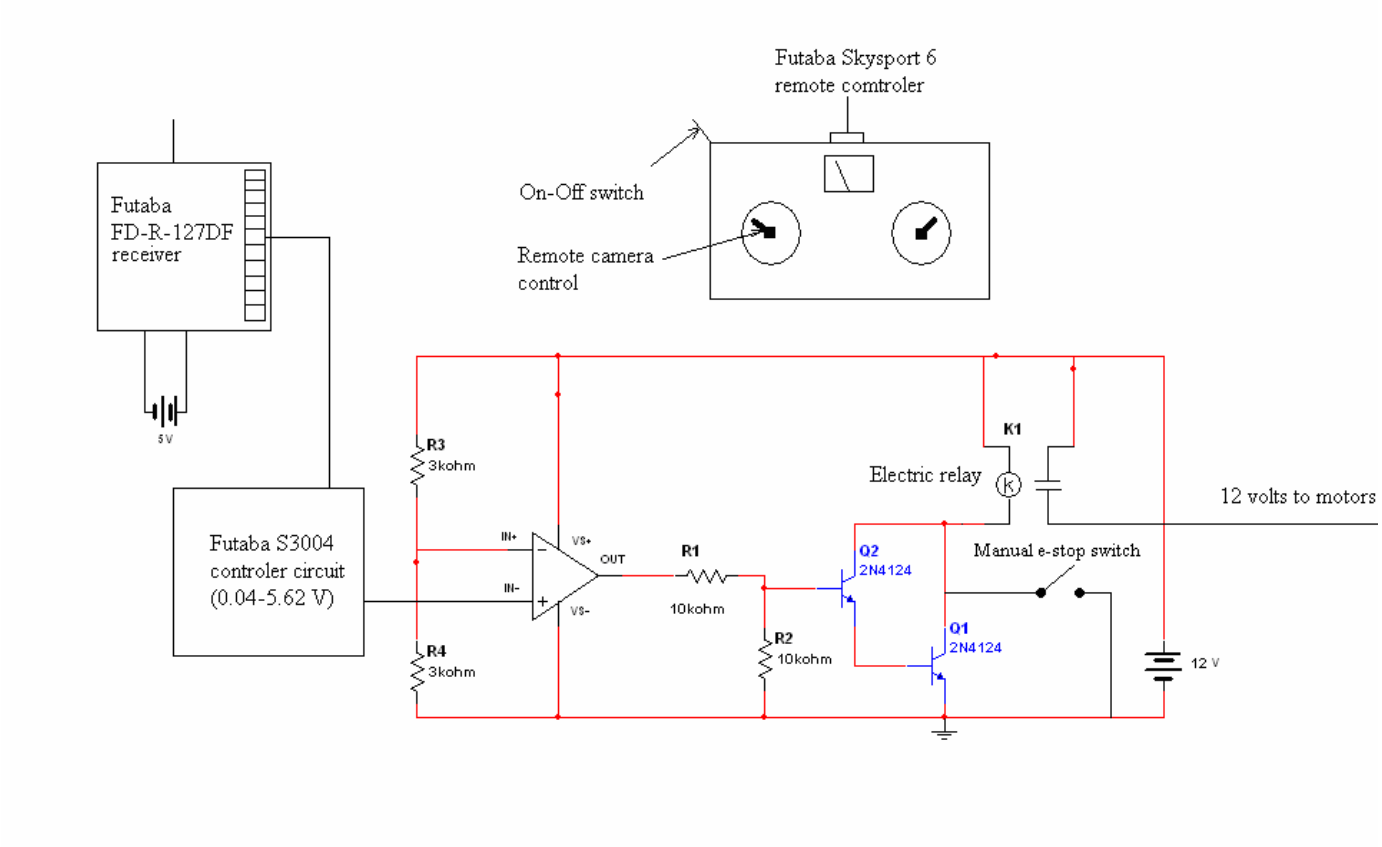

Figure 7 - Emergency stop (wireless and manual)

Mechanical E-stop

A manual emergency stop is simply a pushbutton which is pressed grounds a negative terminal of the electric relay bypassing the electric circuitry of the wireless emergency stop. If the manual stop is energized the power of the motors can not be restored remotely. The manual e-stop switch can be seen in the lower right corner of figure 7.

Control System

The last component of the design solution is the control system. In this component, the development and implementation of a single PD controller for steering and distance control of the autonomous ground vehicle was created in order to enable the vehicle to travel from one location to another. The controller will control all of the robot's movements and speed. In addition, it will work in unison with the data retrieved from the sensors and LT Everette's code to safely and accurately navigate on a prescribed path or to targeted waypoints. The autonomous control system will be comprised of four distinct state-space controllers. The results should verify that implementing this PD controller will provide acceptable lateral and longitudinal control of the autonomous ground vehicle as well as excel in tracking accuracy and steady-state error.

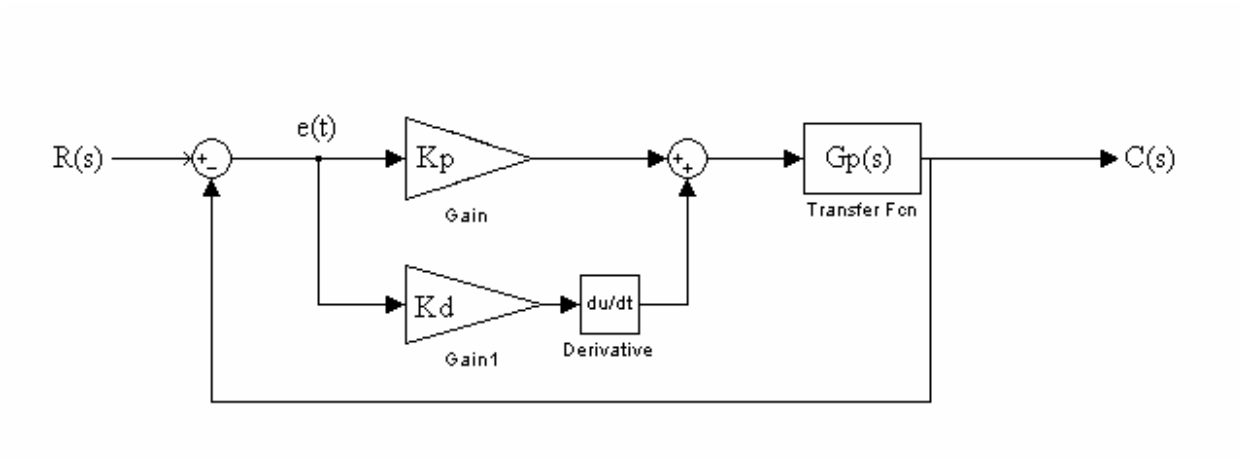

Figure 8 - PD controller devised for the robot.

Figure 8 represents the PD controller devised for the robot. The coefficients have not been determined yet and are pending since the vehicle has not been completely assembled. When the vehicle is completely built, the system I.D. can be calculated and thus the Kp and Kd values can be found which depend on the plant Gp (also known as the system I.D.). The Kp value is proportional to the error while the Kd value monitors the rate of change. The controller designed for the vehicle is simplified since it is only concerned with turning to a specified angle and driving a certain distance. The controller was designed this way since the input that would be provided to the vehicle would be the desired heading and range to a particular location. Lastly, the R(s) represents the voltage input for both left and right motors necessary for the vehicle to reach the desired location.

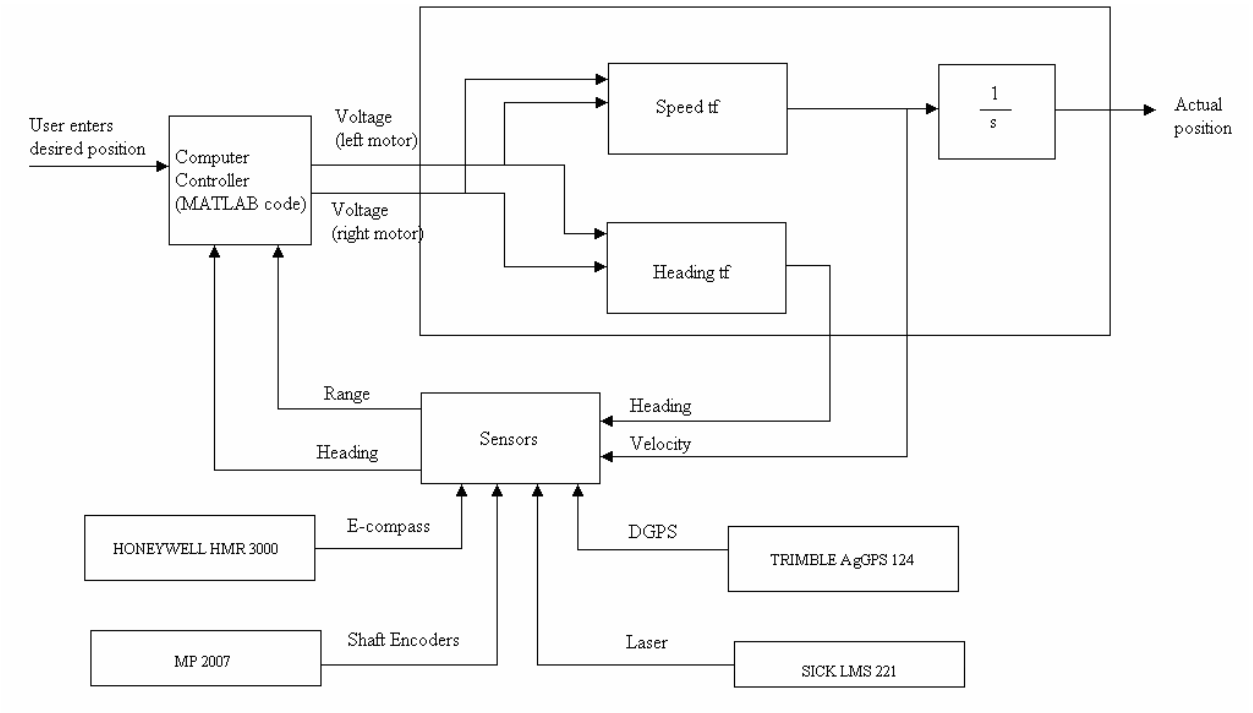

Figure 9 - Vehicle mechanical control scheme

The figure 9 presents a better illustration of how the entire vehicle control system works and the integration of the sensors. The system I.D. is modeled through the speed and heading transfer functions. Since the vehicle is not built yet, the equations can not be presented since they are not accurate. Basically, what should be drawn from this figure is the user enters the desired position, the software (MATLAB code) controls the vehicle's movements (i.e. voltage to the motors) and acquires data continuously from the sensors and feeds the data back to the computer. The controller, which is essentially the big box encompassing the speed and heading transfer functions, makes sure the vehicle reaches the desired position by comparing it with the actual position and makes suitable corrections when necessary to ensure the vehicle is facing the precise heading and has traveled the correct distance.

Design Decisions:

<u>Vehicle Design</u>

During the building of the vehicle we made several decisions regarding the vehicle. We decided to build the 2'x3' vehicle because it is more cost efficient to our project. A larger vehicle provides more room for equipment and a wider base for more stability, but comes at a higher cost and makes navigation in tight places more difficult. A smaller vehicle is cheaper, navigates better, and still has enough room for the equipment needed. The smaller vehicle is the best choice for this project.

The second decision, after deciding on the vehicle size, was the materials to use. There were several choices for what could be used. The first material was the aluminum used on the first robotics vehicle, which has predrilled holes for easily putting pieces together. It costs $39.95 per 6 foot segment. The second was aluminum bars, which are very strong and can support large amounts of weight but cost $56.72 per 6 feet. The third was aluminum tubing, which is strong,

but is lighter weight because it is hollow. Aluminum tubing costs $20.78 per 6 feet. We decided to use the aluminum tubing for several reasons. The first is that it provides the strength needed for our vehicle. Secondly, it is not hard to drill through aluminum. Finally, it was much cheaper than the solid aluminum bars, or the predrilled first robotics aluminum.
The third decision we had to make was the wheels to use. The competition requires a wheel that is capable to driving on pavement or grass. This means we need a wheel with tread and one that is sturdy. We decided to use the wheel shown in Figure 10 - Wheel used on vehicle because it is a large wheel, giving clearance for the bottom of the vehicle, it has tread, which will help with driving off road, and it has a sprocket mounted that is the correct tooth size, which will help with connecting the wheels to the power system. The wheels have a shaft size of ¾", which is larger than what was used on the first vehicle we tested on. Having a wheel with a sprocket mounted eliminated the need for the purchase of a sprocket with a shaft size of ¾".

Figure 10 - Wheel used on vehicle

The final decision, after all parts were decided upon, was how to build the vehicle. The strongest, most permanent solution would be to weld all the parts together. Another solution is to drill holes, and mount all the pieces together using bolts and brackets. We decided to use bolts and brackets to put the vehicle together because welding requires practice and knowledge, which nobody in the group has. By bolting the pieces together, we will be able to build the vehicle ourselves.
The decisions on the propulsion system were driven by what parts were available at the Academy. We did not want to purchase parts for the propulsion system since we had access to a wide variety of equipment from the FIRST Robotics competition teams. The motors and the coupling were used on the 2006 FIRST Robot. The rest of the propulsion system was taken from the 2003 vehicle that was originally being used.

Wireless and Mechanical E-stop

The Futaba transceiver and the receiver were chosen for the wireless link of the remote e-stop because of the economic considerations. The academy already possessed this equipment.
The control circuitry for the electric relay was necessary because a sufficient current supply was necessary in order to power the electromagnet of the relay.

Sensors

The team decided to build shaft encoders using three magnetic sensors, magnet and a MATLAB program because it was a cheapest option available. It was also faster to build the shaft encoder

rather than ordering them. While there are numerous shaft encoders both coupled with motors and separate ones that are available on the market they tend to be expensive and in many cases incompatible with our vehicle design because of the motors that were already available and the wheel that are mounted on the stationary shaft. The shaft encoders were placed on the two front wheels of the vehicle since the vehicle is operating a left and a right motor to emulate a tank drive system. The shaft encoders are vital because they provide the distance traveled by the vehicle which is one of the parameters required by the PD controller in order to travel to a prescribed location. This sensor provides constant feedback as to whether the vehicle has achieved that prescribed distance.

The Trimble 214 DGPS receiver is being used since this was provided for us from the beginning of the project. MATLAB was the chosen as programming language because not only is it a powerful and relatively easy-to-use computational tool, but it interacts very well with other sensors and devices. The RMC sentence was chosen because it provides the essential position, velocity and time data critical for autonomous ground vehicle to accurately navigate. In addition, the DGPS was chosen because it achieves an accurate position varying between 0.5 and 1 meter. Lastly, MATLAB will be used for integrating all other sensors and devices because it will be less confusing and difficult for the user if all programs are written in one common language. This also creates more ease for the user to identify any possible errors in the code.

The Honeywell HMR3000 is used since this sensor was provided for us from the beginning of the project. Even though, the only concerned information from HPR NMEA sentence is the heading, pitch and roll might be used later for determining if the robot is approaching a declining or inclining hill and how much speed to apply to the motors to ensure the robot can make way while maintaining the 5 mph limit.

A digital camera is used since the pictures it takes are digitized which makes possible for digital image processing techniques. Initially, three cameras were intended to be used to get an accurate 180° sweep of the robot's front view. However, due to the lack of processing capabilities of the digital cameras presented to the team, the digital cameras were not able to take pictures simultaneously and thus fail to work afterwards. So the plan reverted to just using one digital camera. The purpose of using the digital camera was to capture several images in order to detect the yellow or white lanes which the autonomous ground vehicle must follow. The Hough transform was chosen because it identifies lines in an image. The Hough transform applies an edge detection operator to a raw image (picture) and determines which lines fit most closely to the data in the image. To serve as an additional tool, the team decided to incorporate a wireless camera on the robot which displays on a remote laptop what the robot is viewing.

The SICK Laser LMS221 was the particular laser rangefinder chosen since this sensor was purchased before the team started working on the project. The purpose of the laser rangefinder to detect obstacles within 8 meters and safely navigate around those obstacles that may be too close or are along the path to a targeted destination (waypoint).

Requirements:

- The vehicle’s length is 3 ft which is between 3 and 7 ft.
- The vehicle’s width is 2.17 ft which is between 2 and 5 ft.
- The vehicle’s height is 5.17 ft which is less than 6 ft.
- The vehicle’s design meets requirement criteria since it has 4 ATV wheels and a mechanical propulsion system which enables the vehicle to maintain constant contact with the ground.
- The vehicle’s propulsion system is independent since its power is generated onboard.
- The mechanical E-stop is a push button.
- The mechanical E-stop button is red in color and is 1 inch in diameter.
- The mechanical E-stop buttons location is in the center rear of the vehicle.
- The mechanical E-stop is located 2.5 ft high (above the ground) which is between 2 ft and 4 ft.
- The mechanical E-stop is solely hardware based and completely stops the vehicle when activated.
- The wireless E-stop is a remote controller that is solely hardware based and completely stops the vehicle when activated.
- The wireless E-stop functions at distances of 50 ft away and greater from the vehicle.
- The vehicle is weather proof since the design of the vehicle is constructed with Plexiglas all around the vehicle and sealant for any small opening in the frame to prevent any rain from getting into the vehicle’s frame, particularly the compartments which house any electrical and mechanical components such as the laptops, sensors, circuitry, and the motors.
- The vehicle’s design has a compartment built in it with the dimensions 26” x 15” x 15” which is able carry a payload of 2 lbs with the dimensions of 18’’x 8’’x 8’’.
- All other requirements specifications are still pending.