LOUSIANA STATE UNIVERSITY
MIKEROBOT

INTELLIGENT GROUND VEHICLE COMPETITION
TEAM 2008

Abstract: The Intelligent Ground Vehicle Competition challenges university students around the world to design and manufacture an autonomous robot. The vehicles are to participate in a series of events competing towards the grand prize. The LSU team is comprised of both mechanical and electrical engineers that have developed the autonomous vehicle, MikeRobot. The report presented provides information of the design process followed by the team and gives justification for all the decisions that were made.

ME Group Members:	ECE Group Members:	Advisors:
David Mustain	Alex Ray	Bryan Audiffred
Diego Gonzalez	William Burke	Dr. M. S. de Queiroz
	Geoff Donaldson	

I, Bryan Audiffred, faculty advisor of the 2008 Louisiana State University IGVC team certify that the engineering design in the vehicle by the current student team has been significant and equivalent to what might be awarded credit in a senior design course.

Signature: _____ Date: _____

(1) Introduction & Motivation

Louisiana State University is planning to participate in the upcoming IGVC for the first time with a team of CAPSTONE design seniors. The CAPSTONE program at LSU is a year-long process where teams of students are challenged to complete a specific project. The first part of the program involves a conceptual design were the team is required to present a ready-to-build prototype in front of a panel of industry expert judges. After the concept is approved, the second stage of the program begins which involves fabrication and testing of the design. Throughout the course of the 2007-2008 academic year, the LSU IGVC team has successfully designed and fabricated MikeRobot; an IGV that will successfully compete in Rochester, Michigan on May 30, 2008.

The team is comprised of two Mechanical Engineers (David Mustain and Diego Gonzalez) and three Electrical and Computer Engineers (Alex Ray, Geoff Donaldson and William Burke). The motivation for this project comes from desire of each team member to represent LSU in a national competition with a multi-disciplinary engineering team. Although the Mechanical Engineering Department has CAPSTONE design teams to national competitions, the Electrical Engineering Department has only recently joined the effort. The Electrical Engineering students voluntarily enrolled in a CAPSTONE course in order to showcase their skills and motivation to excel. This project requires significant integration of two very different areas of study and will thus prepare each group member for industry engineering problems, which are rarely discipline specific.

(2) Design Process

The first-stage of the design process involved the establishment of constraints and engineering specifications for the project. This involved a detailed review of the competition rules as well as an in-depth literature survey of vehicles that had previously succeeded in the competition. From this, a set of requirements were devised that included physical, performance, environmental, power and safety requirements.

The design of the 2008 LSU Intelligent Ground Vehicle was then divided between Mechanical (ME) and Electrical (ECE) teams. Obstacle detection, line detection, waypoint navigation, evaluation of best routes, displacement tracking, and power distribution are some examples of tasks that were addressed by the ECE team. Although the electrical focus varied from the mechanical one, it was important that the design of the vehicle operated as one unit. For this reason, the mechanical design was molded around the needs of the electrical design. This included vibration reduction, maneuverability, weight distribution, structural integrity and weatherproofing. The final concept is presented in this section with proper justification of how the requirements of the project were met.

(3) Chassis Design

The chassis is one of the most important parts of the design of the robot since its structural integrity ensures that the vehicle effectively participates in the competition. To satisfy the engineering specifications, careful consideration had to be given to the structural design and material selection of the chassis.

The chassis for the robot is made out of aluminum 6063(1/8 inch thick) square tubing due to its low-weight and high strength. It consists of two levels, which hold different components. The first level holds the heavy equipment in order to lower the center of gravity of the vehicle. The second level holds the computer, motor controller and other electronic components. MikeRobot features a removable top to allow easy and quick access to the electronics. Finally, the robot is weatherproofed using acrylic sheets and adhesive weather-stripping.

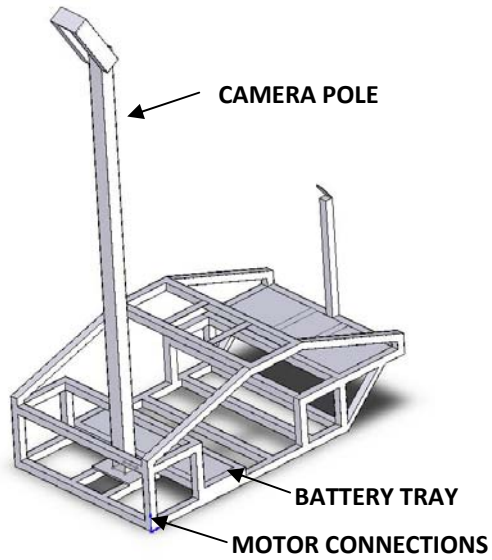

Figure 1: *Robot Chassis*

The camera pole was designed to be a 2 X 2 X ¼ inch square tube. This design was chosen instead of a previously conceptualized tripod for several reasons. The first reason was that the removable top did not make it viable to manufacture a tripod that can be quickly removed. Shipping was also a concern since it would have been difficult and costly to ship a rigid tripod connection. The current pole design uses bolts that can be quickly removed which allows for quick separation of the camera pole from the frame.

SolidWorks was used to both determine the total weight of the frame and to perform stress analysis to ensure that the vehicle would be able to support all operating loads. The total weight of the chassis was determined to be 28.1 lbs. Stress, displacement and natural frequency analyses were performed using CosmoWorks. In loading conditions around 4 times larger than actual operating loads, the vehicle had high strength points of around 0.86 ksi which was well below the yield strength of aluminum 7.99 ksi. From the different analyses performed, it was determined that the robot will be structurally sound and durable.

(4) Motor Sizing

The condition where the vehicle will need the highest amount of power is when it will be going up the 15° incline in the competition. The desired travel speed thru the incline is 5 mph (limited by competition rules). Using dynamic equations, a relation between the weight of the vehicle and the horsepower required can be found.

The estimated final weight of the vehicle (calculated in SolidWorks) is around 110 kg (243 lbs), which will therefore require a total HP of around 1.0. To allow for some uncertainty in the calculations, the motors chosen were required to provide at least twice this amount of power. The motors chosen were two NPC-64038 4-pole motors weighing 13 lbs each. They are each rated at a maximum of 24V, 230 rpm and 1.3 HP with a 20:1 gear reduction. Also, the stall torque for each motors is 68.75 ft lbs.

Torque calculations where performed to ensure that the torque required to move the vehicle would not exceed the stall torque. For the 12-inch diameter tires chosen the total torque to maintain a speed of 5 mph up a 15° incline was calculated to be 38.4 ft lbs. Also, the torque required to begin motion from rest is 14.8 ft lbs on level ground and 52.1 ft lbs at a 15° incline. All calculated values are well below the net torque that the motors can provide (137.5 ft lbs).

(5) Electrical System

The electrical system of the robot is charged with the task of intelligently governing the behavior of the robot. Because the team has no prior knowledge of the course, the robot must piece together an internal model of the course as it moves. It accomplishes this by gathering data from a combination of several sensors, and combining all the resulting knowledge in a unified algorithm. This task can be broken into two main subsections: hardware and software.

The hardware includes the following components: camera, laser range finder, distance sensors, GPS, compass and motor controller.

The software requires much more in depth explanation, as it is the more open ended portion of the project. After a high level overview of the obstacle detection and avoidance system, the low level interfaces to the sensors are described.

(6) Power Distribution

Since the robot has no connection to external power six batteries are carried onboard, four of which supply the motor controller and power distribution circuit. These are wired in a series/parallel configuration to generate 24VDC. This combination has enough capacity for about 4 hours of motor run time, meaning that the motors are actually turning the entire time. The other two are used to power the computer through an inverter. Although the computer runs on 19VDC and 4.75A, the decision was made to use the stock power supply to avoid any problems that might come with attempting to make a custom circuit. Safety of the equipment took precedence over efficiency. The two batteries are wired in parallel to increase capacity. They result is a battery life of about 8 hours for the computer.

The peripherals are connected to a custom designed power distribution circuit to allow for more efficient power consumption. The power distribution circuit contains three voltage regulators that convert the twenty-four volts coming in from the batteries into twelve volts, nine volts, and five volts. Each regulator has four voltage outputs and supplies two amps of current.

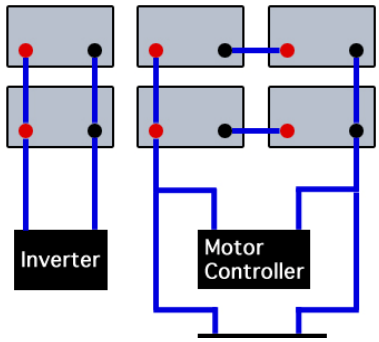

Figure 2: *Battery Usage*

(7) Software Overview

The software algorithm begins by gathering data from the sensors through the data acquisition layer. This layer will simplify interfacing to the hardware by providing high-level functions to access the desired data during program execution. Once the data has been gathered, several operations are performed. The laser range finder data is grouped, and the resultant centroids are transformed into image coordinates; image colors are identified and barrel distortion is corrected; and current heading is calculated from a combination of compass data and encoder data. Next, the laser range finder data is used to construct a ground plane map.

To extend the obstacle placement capability, a color ground plane map is also created. Meanwhile, the colors in the image are segmented to remove the green grass, which is of no interest. The two ground plane maps are combined into a single map, and the ground plane is separated from the rest of the image. This results in two images: obstacles and ground plane.

These two images are passed to their respective processing algorithms. Both begin with a black and white region properties analysis. Extra laser range finder group analysis is used to identify A-frames on the obstacles side. Then, the obstacles' depths are estimated, and they are placed on the final obstacle map. On the ground plane, a morphological process is used to generate two images: one including only small objects and lines, and the other depicting all other obstacles. Region properties are used to distinguish between the various types of ground features, including potholes, lines, and sand pits. The main purpose of the ground algorithm is to identify sandpits or other large passable obstacles, and remove them from the image. All other obstacles are put directly into the map since they should be avoided.

While the construction of the map is taking place, the algorithm will also parameterize the lines in the image. This information will subsequently be used to extract the course direction from the image. The course direction enables a final rotation of the generated map so that the desired direction will always go from south to north in the map. The robot's position and heading in the map will also be noted.

Once the map has been rotated to account for the desired direction, the A-star path planning algorithm will execute. Once the path has been calculated, it will be quantified into real world coordinates so that the robot will be able to follow the A-star's instructions.

Once the path correction module has executed, the result will be the path that the robot should follow. This information will be a list of X,Y points for the motor control tracking to visit. The robot will make its goal several points ahead of the current position to smooth out motion profiles. The motion can be interrupted at any time due to the detection of an obstacle by the emergency obstacle sensors.

The algorithm takes emergency distance sensors into account because the robot will be in motion during execution. These sensors are monitored for the presence of an obstacle by a dedicated PIC microcontroller. If the sensors detect an impending collision, the monitoring module will generate an interrupt for the motor control output layer so that the collision can be avoided. A preset routine will be executed to increase distance from the obstacles.

The entire algorithm will execute at once, with several threads tending to their own responsibilities. This parallelism will take advantage of true multitasking on a dual core system and also blocking execution while waiting for data or signals from other threads.

In order to increase the accessibility of multiple classes, an upper-level class structure has been implemented. The motor controller (Roboteq), Compass, and GPS classes were combined into the CurrentParams module which contains the functionality for keeping track of the current position of the robot. The Camera and LRF classes were combined into the Vision module along with a Matlab compiled library and many other C++ functions in order to perform some of the advanced obstacle detection features of the robot. The vision module sends a weighted grid to the path planning module, which in turn sends a list of points to the tracking module. The tracking module generates all motion and includes an instance of the CurrentParams class. From there it gets this list of points from the path planning and traverses them with its built-in functions. The class hierarchy is displayed below:

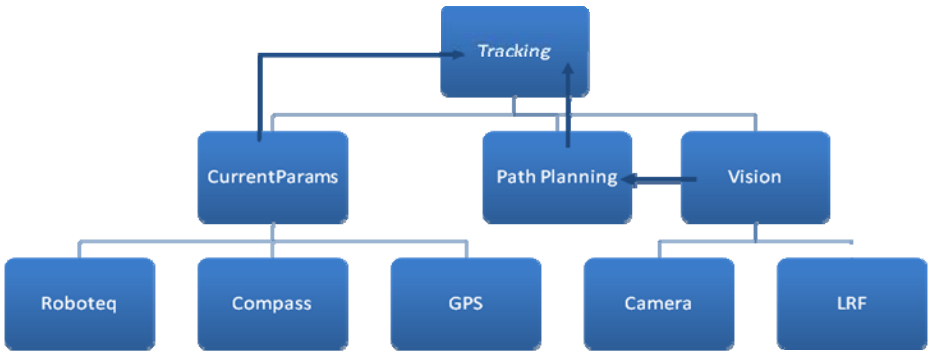

Figure 3: *Software Class Hierarchy*

(8) GPS NAVIGATION

To effectively navigate, the robot must have a sense of its location in relation to an origin at all times. This is accomplished by creating a thread that updates the x and y displacement of the robot. This thread has been named the odometer thread. The inputs to this thread are the encoder values from the motor controller, the compass, and the GPS as shown below:

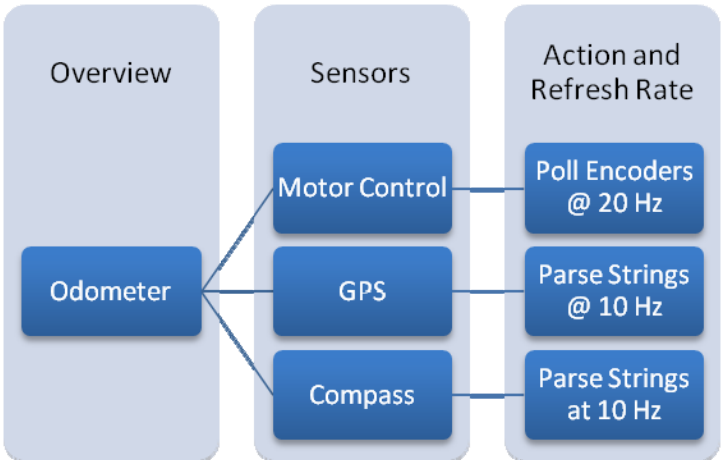

Figure 4: *Tracking Overview*

Due to the fact that the data comes in at different rates, the algorithm needs to be adjusted to compensate at different cycle times. For even cycles, the heading is corrected with the current heading data from the compass. Every four cycles, the position data is corrected with current position data from the GPS. Also, in every cycle, the X and Y values are updated based on the values that the encoder registers.

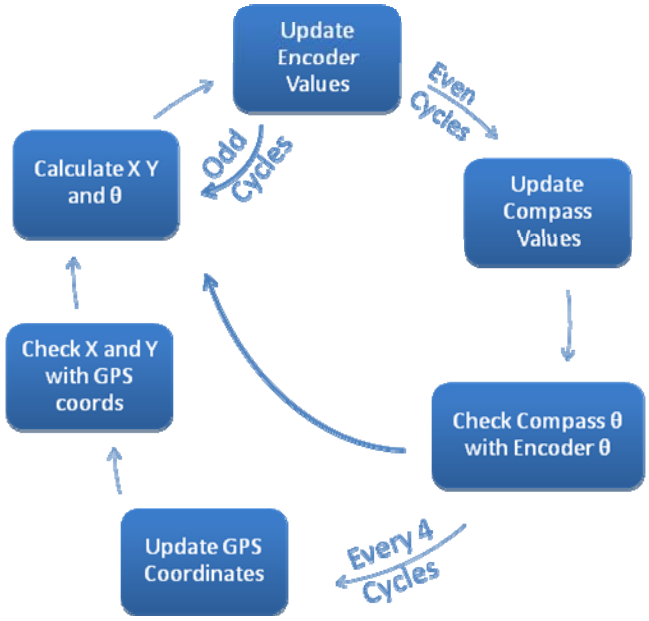

Figure 5:*Flow Diagram for Tracking*

Due to the fact that dead reckoning with encoders accumulates a large amount of error due to wheel slippage, additional sensors must be added to the equation. The primary error is accumulated when one wheel slips without the other slipping. This causes a large error in the calculated angle of turn. Since the compass is accurate within .3°, the angular displacement from the encoders is compared to the angular displacement

measured by the compass. If the encoder's heading data is off by more than .3°, the angle from the compass is used as seen in Figure 6.

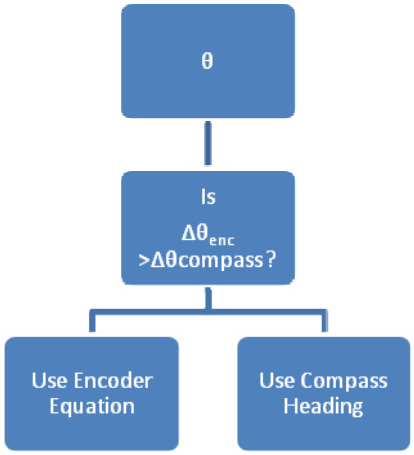

Figure 6: *Angle correction*

The GPS is accurate within 60 cm with differential correction. If the error in x or y is greater than 60 cm, the GPS's value will be used. This process is shown in Figure 7.Ideally a Kalman filter or an Alpha-Beta filter would be used with this data, but due to the complexity of the filter design, it was not implemented this year.

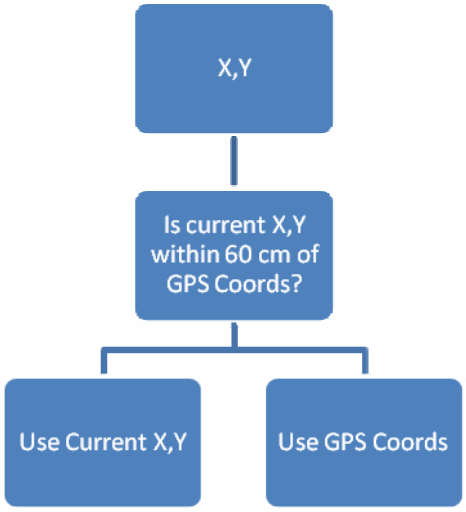

Figure 7: *Position Correction*

The tracking functionality of the robot uses an algorithm to navigate displacement in XY terms. It updates the robots speed and rate of turn each time the odometer function is run. This ensures that the position data is current when the function is called. The algorithm first converts XY coordinates to ρθ coordinates. From there it calculates the desired distance per wheel. The desired vector angle is then checked to see if it is greater than 45°. If it is larger than 45°, the turn first method is used. This method stops the robot and makes it turn past the 45° mark. It ensures that there is a 15° overshoot so the robot does not continuously

stop and turn a small number of degrees while taking its path. The speed and correction are the calculated and the robot begins its motion. From here, the speed and correction are checked to ensure that they do not exceed a user defined speed limit.

(9) Obstacle and Boundary Detection

The tasks of obstacle detection and boundary detection are accomplished in tandem in the design. The method uses the combination of a camera, laser range finder, and knowledge of the course and obstacles to reliably create a model of the course to pass to the path planning algorithm. The method will operate as follows:

(9.1) Image Color Identification

Since color images are three dimensional, it is difficult to perform many image processing algorithms on them. Most image processing is done with gray scale and black and white images. However, it is necessary to preserve the color information contained in the image so that grouping and color segmentation can be performed.

The color naming algorithm, which is described in *Learning Color Names from Real-World Images*[1], uses a trained matrix that was constructed from image sets from Google Image Search and Ebay to easily identify the color of a pixel in question. The algorithm converts the RGB values to L*a*b colorspace and finds the appropriate name for the color using the lookup table. The code used was ported from Matlab code supplied by the authors of the paper on their website [2]. The code can produce either a colorized image or an indexed image as an output. The indexed image is much faster to execute and more suitable for the intelligent vehicle's purposes. Figure 8 shows an example original image and color-identified image. Image courtesy of www.igvc.org.

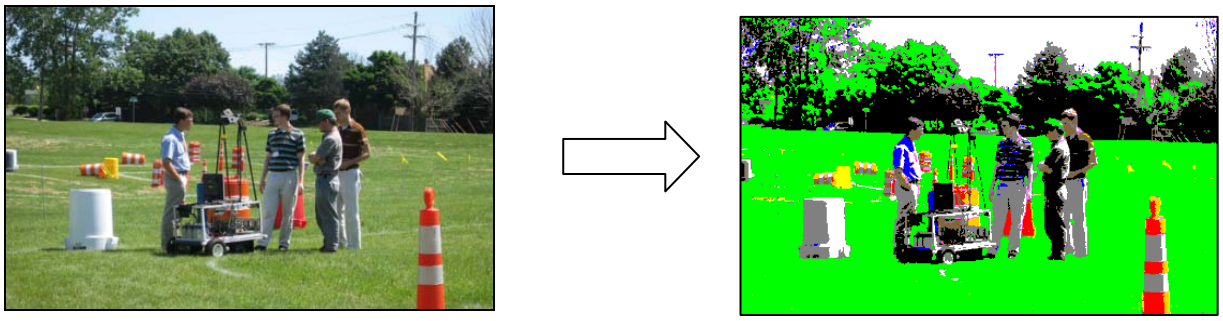

Figure 8: *Image Color Naming*

(9.2) Camera Distortion Correction

Most digital cameras are equipped with lenses that cause a significant amount of distortion to the image. When the image is used for computer vision purposes, the effect of this distortion is especially relevant. A model to account for the distortion was implemented and includes both radial and tangential distortion. The radial distortion accounts for the barrel shape of the image, and tangential distortion accounts for the centering of the radial distortion. Results of image correction using the values obtained by the Matlab Calibration Toolbox are shown in Figure 9. The corrected image is noticeably cropped by the toolbox. The final algorithm was written for C++, and it preserves all image data.

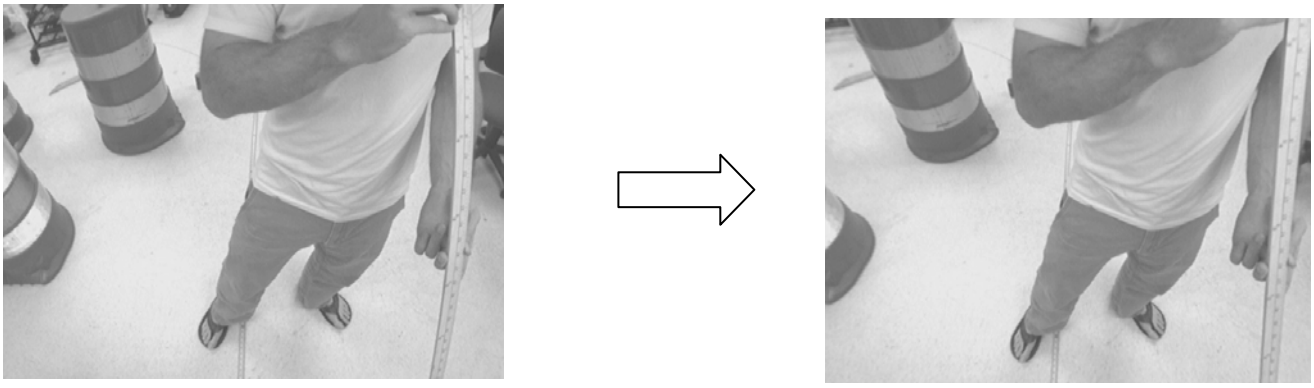

Figure 9: *Image Distortion Correction*

(9.3) Image Color Segmentation

The main reason to name the colors in the image is to facilitate partitioning of the image based on color. Some obstacles in the course are known to be white, orange, yellow, and blue, but more importantly, the majority of the course will be run on grass, which is obviously green. Therefore, a mask can be made to remove all of the green-colored pixels from the image. This not only generates excellent segmentation between obstacles, but also reduces the information of the image so that the processing will be less complicated. An example of the result of subtracting all green pixels is shown in Figure 10. The purpose of this is to generate a binary image that includes both ground features, such as boundary lines and potholes, and obstacles. Image courtesy of www.igvc.org.

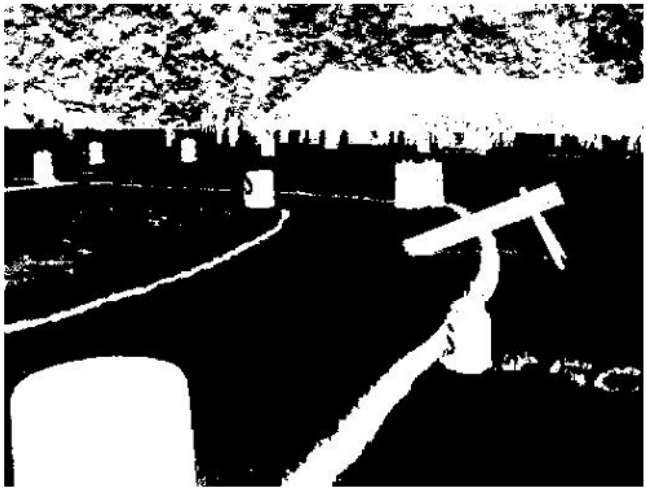

Figure 10: *Binary image showing pixels that are not green*

(9.4) Obstacle Placement using Laser Range Finder Data

Once the obstacles and ground features have been segmented from the background of the image, the position of the obstacles in the X-Y plane should be determined. There are two coordinate systems in question at this point: world coordinates and image coordinates. The former refers to the global position of objects in three dimensions. The origin can be placed anywhere, and the most logical place is directly beneath the laser range finder. Image coordinates refers to a location in pixels on the image that corresponds to a point in world coordinates. Using standard camera geometry methods and the pinhole model for a digital camera, the mapping from world coordinates to image coordinates is trivial.

To translate from image coordinates to exact world coordinates, on the other hand, is impossible, because any point on a specific line in world coordinates can be projected to the same point in image coordinates. To achieve a decisive placement of the object in the X-Y plane requires more information.

The extra information comes from the laser range finder. Because the location of the laser range finder is fixed compared to the world coordinates origin, it's X, Y, and Z world coordinates are known. If the laser range finder is taken to be at <0,Y1,0>, then the values it reports will actually be exact three dimensional world coordinates. To reduce the number of points that need to mapped using the camera matrix, the laser range finder points are grouped into expected obstacles. This not only reduces the amount of math that needs to be done in this step, but also helps reduce the effect of inaccuracy in the camera matrix. Only the center points of the laser range finder obstacles are mapped into the image to identify the location of the obstacle's leading edge. A sample result of laser range finder point grouping is shown inFigure11.

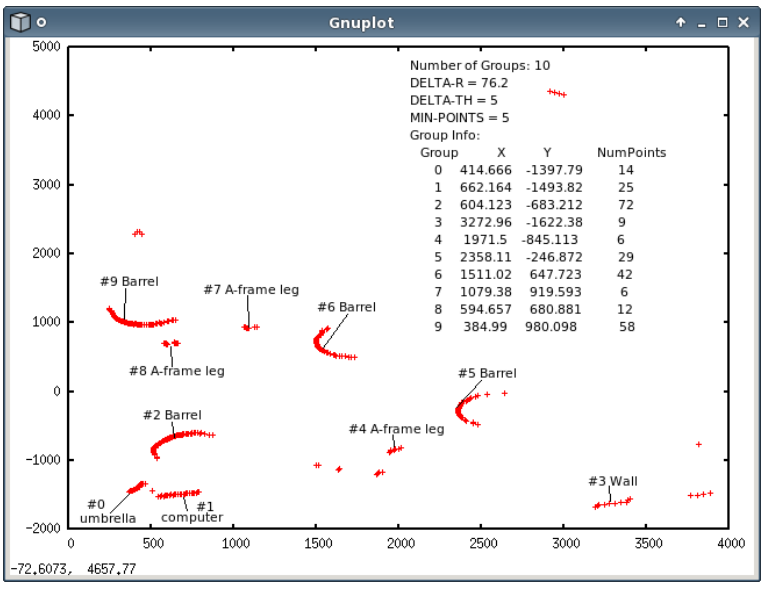

Figure 11: *Laser range finder data after grouping*

(9.5) Perspective Transformation

For the purposes of examining the ground plane, the image needs to be artificially rotated so that the camera view is looking straight down on the scene. This is done by a perspective transform that is based on the camera matrix.

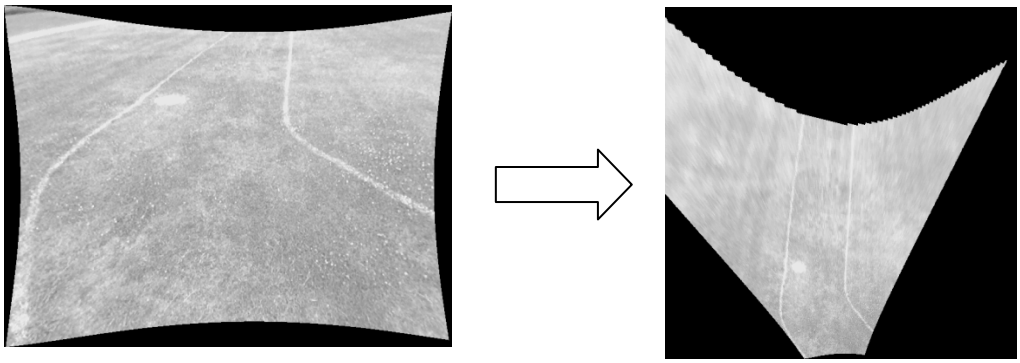

Figure 12: *Perspective Transformation*

(9.6) Ground Feature Identification in the Ground Plane

From laser range finder data and color data, the ground plane can be segmented in an image. Once the ground plane is segmented, it is much easier to detect terrain such as sand pits and hazards like potholes. The image that will be analyzed will be a binary image. The algorithm for differentiating between the three will depend on morphological characterization of connected areas. Once the green is masked out of the image, the separation of ground plane features is excellent. Pixels will be grouped into contiguous regions. Then, the regions will be analyzed to provide relevant information such as area, eccentricity, and orientation. An example of identifying contiguous circles and generating their region properties is shown in Figure 13. The properties have been overlaid onto the image.

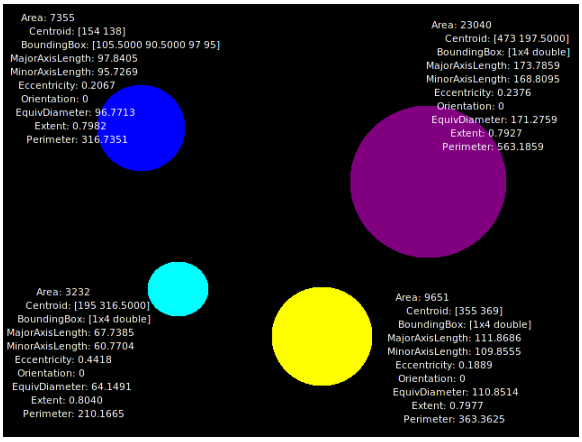

Figure 13: *Results of obtaining region properties on a sample image*

(9.7) Ground Feature Identification in the Ground Plane

After the characterization of the other features on the ground plane, the remaining pixels will represent both noise and the visible boundary lines. It is necessary to remove every possible object from the image so that the lines dominate the image as possible. The Hough transform is a method of finding and parameterizing straight lines in an image. It maps each point to a sinusoid in Hough space. The sinusoid corresponds to all of the possible lines that can pass through the original point. Therefore, when all of the points in an image are transformed into Hough space, sinusoids that correspond to collinear points in the original image will intersect at a point in Hough space. If there are many collinear points in the image, their Hough space sinusoids will intersect to create a maximum, as shown in Figures 14 and 15.

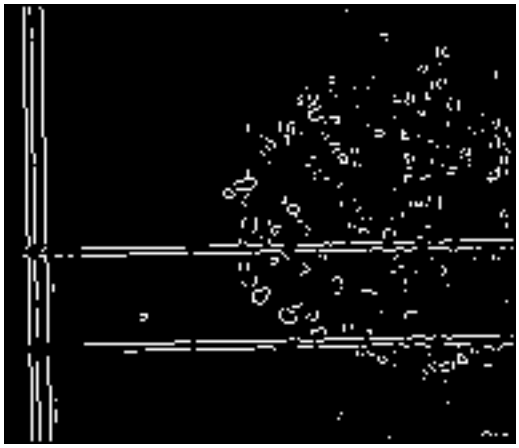

Figure 14: *Sample MATLAB Line Detection Image*

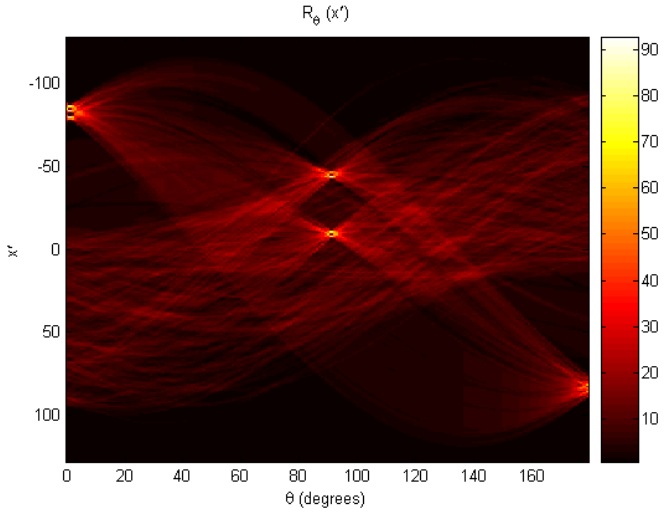

Figure 15: *Radon/Hough Transform of Figure 14*

The angle of the line as reported by the Hough transform will correspond exactly to the difference between the robot's heading and the heading of the course. Lines should be parallel, so the function should find at least two lines at the same angle.

(9.8) Final Map Construction

Once all of the previous steps have been completed, the information to make the final snapshot of the course is available. Nearly all of the image processing algorithms are dedicated to building a snapshot of the course as the robot sees it at a particular instant in time. However, it is necessary to keep the information from previous snapshots and merge it with new ones so that the robot has as full of a map of the course as possible to make its routing decisions. This requires a method of overlaying multiple maps. A sample course is shown in Figure 16. The orange line is a possible path that the robot could take. The boxes represent possible locations where a map fragment could be generated. Since each fragment will be rotated to the current heading of the robot. These rotated images are shown in Figure 17

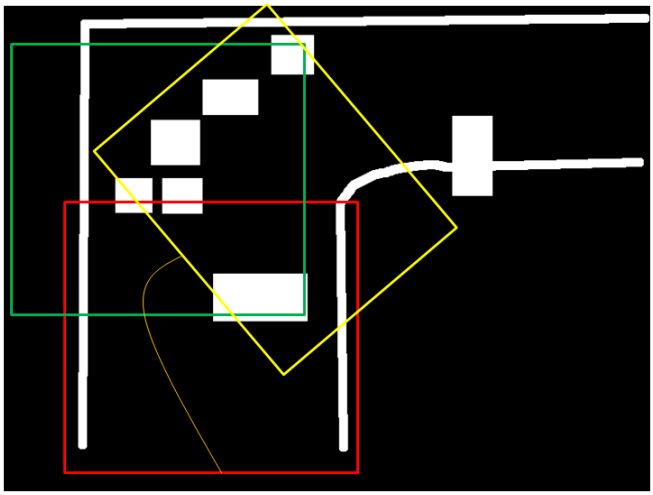

Figure16: *Example simulation*

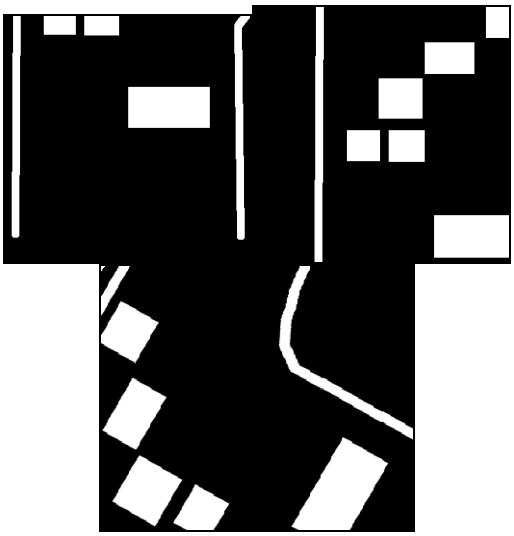

Figure 17: *Three images taken along the simulated course*

The overlaying of maps is made possible because the location at which the map was generated is saved along with the map itself. Therefore, it is a matter of rotating and translating all maps into the same coordinate system and expanding the full matrix to accommodate the information from all the maps.

(10) Path Planning

The path planning is all computed using the A-Star (A*) algorithm using the result of the overlay of all fragments. The plan is to periodically write the entire map to disk as it becomes too large. Then the map that resides in memory can remain as small as possible, but the full map can be reconstructed after the run for analysis. The A-star algorithm will compute the lowest risk path from a designated starting point and a designated ending point in a weighted grid as shown in Figure 18.

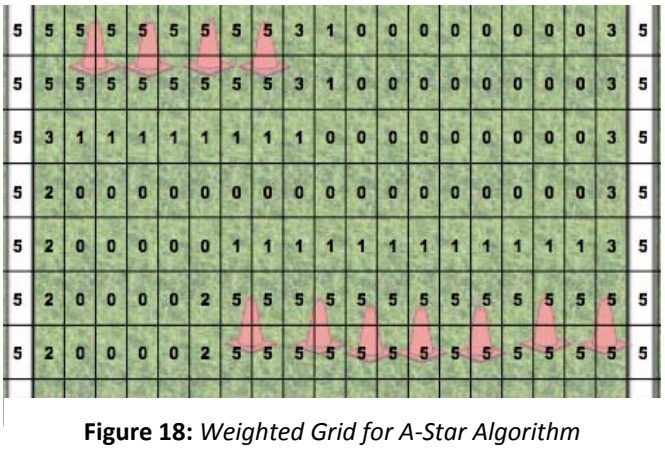

Figure 18: *Weighted Grid for A-Star Algorithm*

(11) Safety

The speed of the robot will be hardware limited by the motor controller at 5 mph during competition events. However, the speed limitation will be reduced during any tests conducted within the direct vicinity of people or property that could be damaged. The speed limitation reduces the risk of collision and it also serves as a means to reduce the severity of any potential collisions. The computer vision algorithms for detecting obstacles with the camera and laser rangefinder in combination with the obstacle avoidance algorithms will serve as the primary means of preventing collision. In the event that this system fails to avoid an obstacle, the emergency sensor system along the perimeter should prevent collision. If both systems fail, an emergency stop signal can be sent via the on board mechanical stop button or the wireless emergency stop button. The large mechanical stop button will be centered in the top panel at the front of the robot for easy access. The wireless stop will be achieved using a radio frequency module with a working range of 50 meters.

(12) Cost

The following table summarizes the overall cost for this project:

Description	Estimated Cost

Laser Rangefinder	$2700
GPS	$3000
Motors	$600
Camera	$300
Batteries & Chargers	$500
Motor Controllers & Encoders	$700
Testing Equipment	$1000
Computer	$1000
Frame Material, Wheels, Casters, Tires	$550
Miscellaneous Supplies	$100
TOTAL:	**$12,250**

(13) Conclusion

The 2008 Louisiana State University Intelligent Ground Vehicle introduced in this report provides a functional design that is expected to successfully attend the next IGVC. From its unique path planning methods to its practical chassis, the design of MikeRobot is a collaboration of two teams: mechanical and electrical. As it was discussed in this design report, the electrical team tackled issues such as obstacle detection, path planning and hardware/software required. The mechanical team successfully addressed the design of a reliable and lightweight chassis, while the electrical team combined innovative and known methods to develop this Intelligent Vehicle.

(14) References

[1] van de Weijer, Joost; Schmid, Cordelia; Verbeek, Jakob, "Learning Color Names from Real-World Images," Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference o*n* , vol., no., pp.1-8, 17-22 June 2007

[2] http://lear.inrialpes.fr/people/vandeweijer/color_names.html