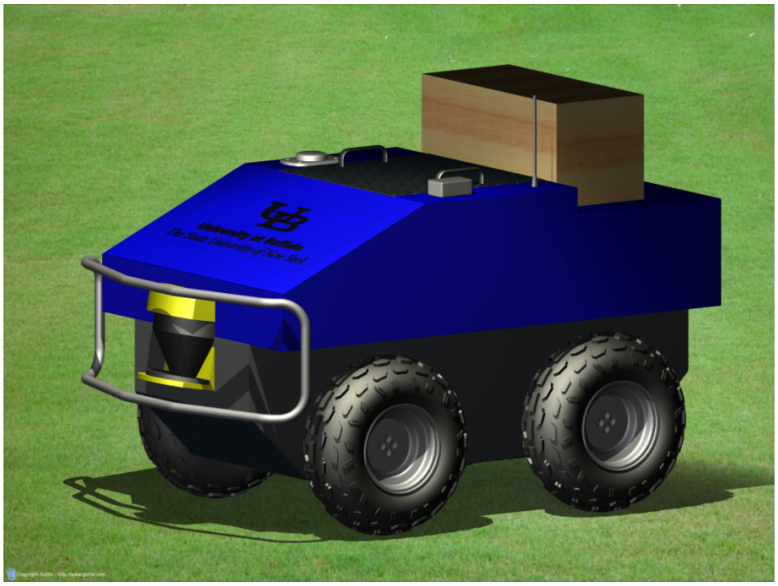
UB Robotics

An Undergraduate Club of the University at Buffalo

Big Blue



2009 Intelligent Ground Vehicle Competition

Daniel Muffoletto, Mark Tjersland, Tim Montgomery, Colin Lea, Mike DiSanto, Ben Deuell, Chris Nugent, Matt Pivarunas, Chih Yong Lee, Darwin Yip, Jake Joyce, Doug Calderon, Pradeep Gollakota, Dominic Baratta, David Berquist, Ashish Kulkarni, Matt Mott, John-Paul Sitarski, Andrew Puleo, Don Monheim, Oluwatlbi Busari

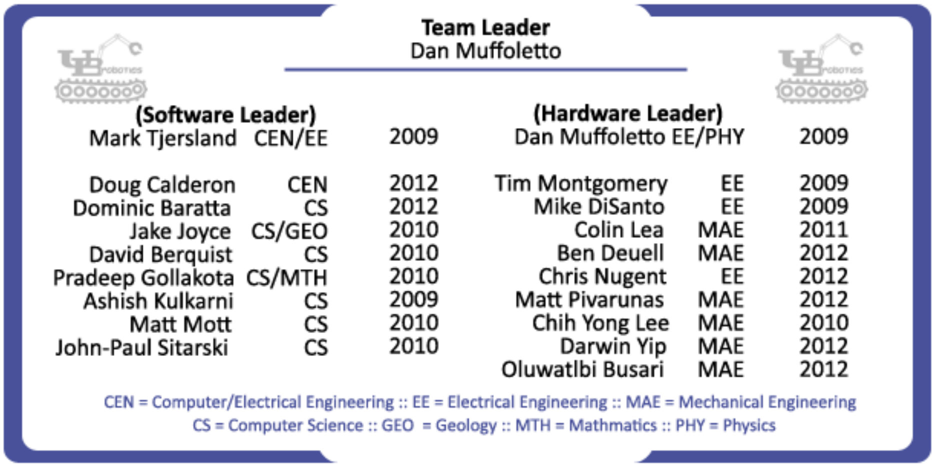
Advisor: Dr. Jennifer Zirnheld

I certify that the engineering design of the vehicle described in this report was done by the current student team and has been significant and equivalent to what might be awarded in a senior design class.

Dr. Zirnheld
Department of Electrical Engineering
University at Buffalo

I. Overview

The University at Buffalo Robotics Club (UBR) is competing for its second year in the 16th annual Intelligent Ground Vehicle Competition (IGVC). The team of undergraduate students used their knowledge from the previous contest to build a completely new vehicle that is significantly more capable than it's previous entry. Extensive use of Computer Aided Engineering (CAE) tools and simulation resulted in an outcome that met their original goals. Despite the team's diverse mix in education level, this year we have succeeded in bringing younger students up to speed and implementing innovative ideas. The team breakdown is as follows:



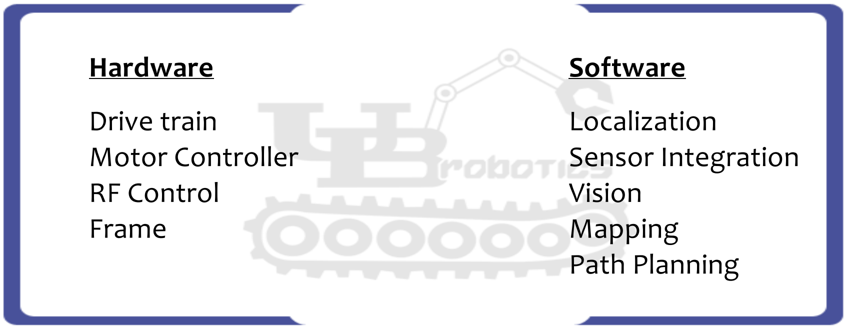
Team Leader
Dan Muffoletto

(Software Leader)			**(Hardware Leader)**		
Mark Tjersland	CEN/EE	2009	Dan Muffoletto	EE/PHY	2009
Doug Calderon	CEN	2012	Tim Montgomery	EE	2009
Dominic Baratta	CS	2012	Mike DiSanto	EE	2009
Jake Joyce	CS/GEO	2010	Colin Lea	MAE	2011
David Berquist	CS	2010	Ben Deuell	MAE	2012
Pradeep Gollakota	CS/MTH	2010	Chris Nugent	EE	2012
Ashish Kulkarni	CS	2009	Matt Pivarunas	MAE	2012
Matt Mott	CS	2010	Chih Yong Lee	MAE	2010
John-Paul Sitarski	CS	2010	Darwin Yip	MAE	2012
			Oluwatlbi Busari	MAE	2012

CEN = Computer/Electrical Engineering :: EE = Electrical Engineering :: MAE = Mechanical Engineering
CS = Computer Science :: GEO = Geology :: MTH = Mathmatics :: PHY = Physics

Design Process

This year's robot is the second iteration for UB Robotics. Last year's version was crucial in our understanding of the steps needed to create an autonomous robot of this scale. This year we spent less time figuring out what needed to be done and more time researching algorithms for navigation and learning about more intricate circuit design. Based on group discussion and feedback from professionals we planned to focus more time in the following areas:



Hardware

Drive train
Motor Controller
RF Control
Frame

Software

Localization
Sensor Integration
Vision
Mapping
Path Planning

Foremost, our goal was to create a robust robot capable of completing the navigation and autonomous challenges for the IGVC. Secondary goals include building a vehicle that can handle a variety of rugged terrains and have a platform that was robust enough to be used in future years. We had a limited budget and had to plan accordingly; we built as much as possible from scratch including the circuits and vehicle manufacturing.

II. Hardware

Our club utilized computer aided engineering (CAE) tools whenever possible in order to minimize wasted time and effort in later phases of development. For CAD we used Autodesk Inventor 2009. We chose this over other graphics packages because it is free for our members and it is comparable to other professional CAD software.

II.1 Mechanical Design

On our previous design, our rugged platform gained a lot of attention from professionals in industry and academia. This attention prompted us to continue with this style rather than being minimalist.

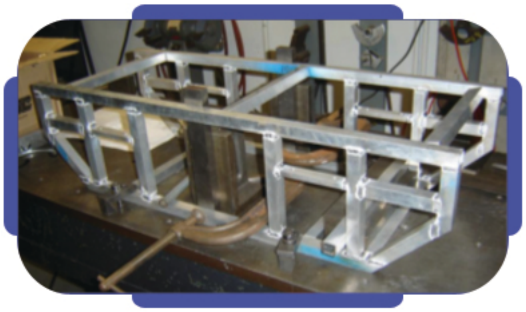


Figure 1 Lower Frame

The chassis is constructed of 1" square tubing with 1/8" sidewalls. The tubing was welded into a lower and upper half. The lower half houses the motors, batteries, and encoders. The upper half contains electronics and the computer. On the outside are mounts for sensors.

Mechanical Innovations

Batteries are heavy and inconvenient to regularly take out. Significant time was spent in designing a set of battery packs that can be handled easily. Two sealed lead acid batteries are packed into a single casing that can be swapped out of a compartment in the robot.

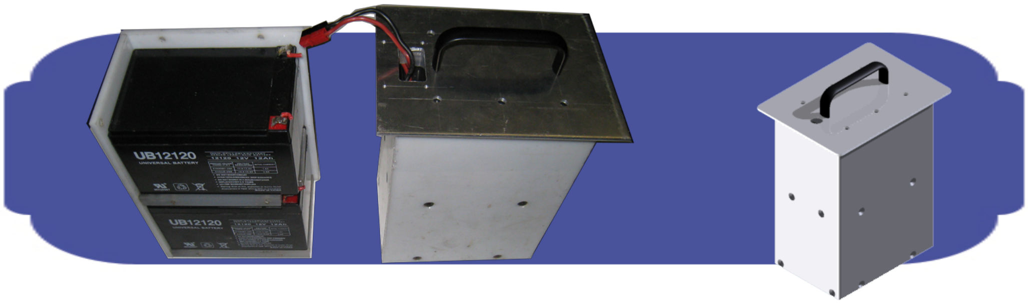


Figure 2 Battery boxes (Open, Final, CAD)

Big Blue has four motors that directly drive the wheels. While having four motors adds weight and cost to the vehicle, this greatly increases control and maneuverability. It eliminated rocking and vibration problems we had from our previous two-motor system and more importantly has a zero turning radius. The wheelbase width-to-length ratio is 1.1 which allows for greater stability and smoother turning.

A low center of gravity makes the vehicle more stable, distributes weight on the tires more evenly during accelerations, and prevents rollovers and excessive rocking. For this reason we kept the batteries and motors (the heaviest parts) as low as possible. Based on information from our CAD program, the center of gravity is 13 inches above the ground.

In order to secure the robot in case of a crash there are bumpers on the front. This avoids damage to the laser rangefinder and decreases damage to the body.

FEA Analysis

Finite Element Analysis was done in *Autodesk Inventor* to find out where the weak points on the lower frame are located. Many simulations were done with forces distributed over the top, pulling on the hitch, coming from the motors, and distributed over various points. Using forces totaling over 500 lb, the factor of safety never went below 3. Pictures below are using extreme forces to show where the weakest points are located.

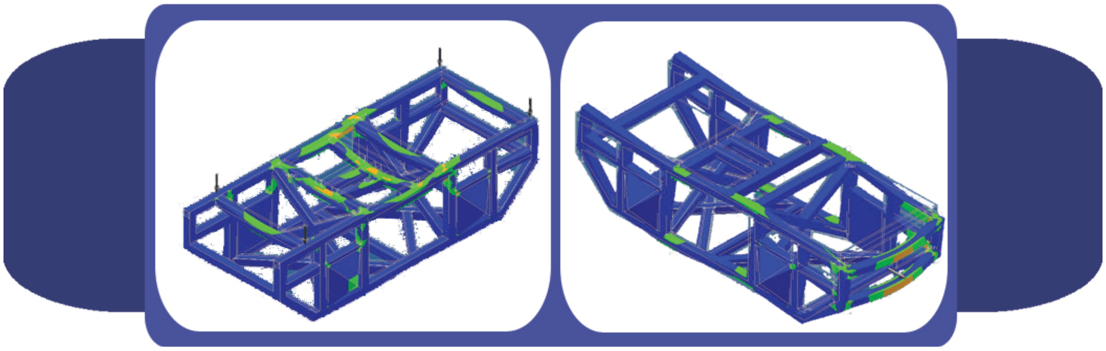


Figure 3 FEA (Left: Force on each corner; Right: Force pulling on hitch)

Our factor of safety is based on ideal conditions. Welds were done by our members and are likely not as strong as those simulated. The factor of safety was high enough in our simulations that we believe the frame will not break except under extreme circumstances.

Sensors

The need for sensors can be classified into those that can find obstacles and those that help determine where the robot is located. The first problem is for determining objects like cones, barrels and fences as well as for locating boundaries such as the white line in the autonomous challenge. The second problem is figuring out where we are in relation to where we were before so that we don’t run into previously discovered obstacles. Due to error, redundant sensors are necessary to get a more accurate depiction of the real world.

Sight. For determining obstacles we used a Panasonic 3CCD video camera with a 37mm wide angle lens. The lens creates minor distortion but provides us with a broader set of data. It is mounted 25 inches off the ground. These measurements were chosen based on empirical data from a field test, and it was determined that it would have a field of view equivalent to a camera mounted to a tall mast.

Rangefinding. To find distances to objects we decided to use a SICK PLS101 laser rangefinder. Ideally, it is capable of finding objects within a radius of 50 meters and field of view of 180 degrees. We chose this model, which is regularly used for industrial safeguarding, was significantly cheaper for us to acquire ($215 vs. $3600+ for the LMS series) and provides a level of accuracy that we believe is sufficient.

Localization. A Novatel Propak-V3 DGPS was donated to us last year. It provides accuracy of up to 0.1 meters. It records GPS coordinates and orientation. It is a differential GPS with an Omnistar HP subscription, which means it gets corrections from base stations therefore making it more accurate.

We are using a PNI 3-axis digital compass with pitch/roll compensation to find the vehicle's heading

Odometry. US Digital E4 Wheel encoders are used to give more accurate velocity data.

Each sensor has an independent accuracy, resolution and refresh rate. Because these are not perfect the localization data can be combined using an Extended Kalman Filter to get a more accurate output of where we are.

Component	Accuracy/Resolution	Refresh Rate
Propack GPS	0.1 meters	20 Hz
SICK LIDAR	7 cm at 4 meters	10 Hz
Camera	720 x 480 pixels	30 FPS
Digital Compass	Heading: 0.1 degrees	8 Hz
Wheel encoders	2560 CPR at wheel	10 Hz

II.2 Electrical Design

The UB Robotics club designed a significant portion of custom electronics for Big Blue.

Batteries and Power Supply

A custom power supply board was designed to regulate the battery voltage for the other electronics onboard the robot. Aside from the unregulated 24V rail for supplying the LIDAR and motor controller, a 12V rail (capable of 7A) was needed for the GPS, and wireless router, a 5V rail (capable of 5A) was needed for the digital compass and USB hub, and a 7V supply was needed to power the camcorder (in place of its battery). All rails are underutilized to allow for future expansion. An overview of this system is shown below.

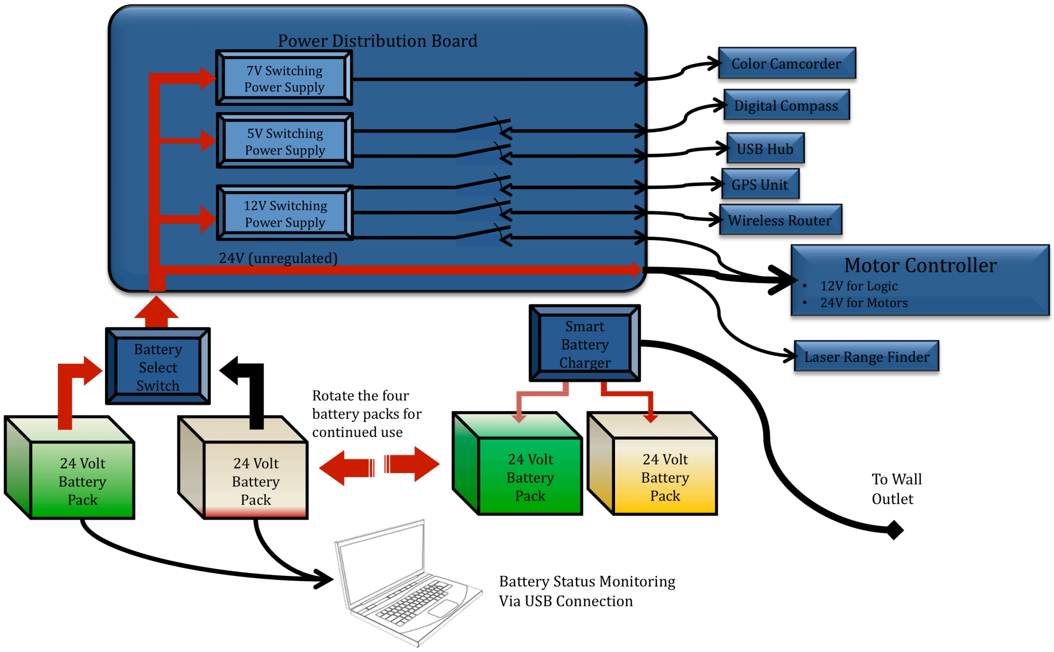


Figure 4 Power Overview

All power supplies were designed using the Simple Switcher series of step-down regulators from National Semiconductor. A picture of the power supply board is shown below.

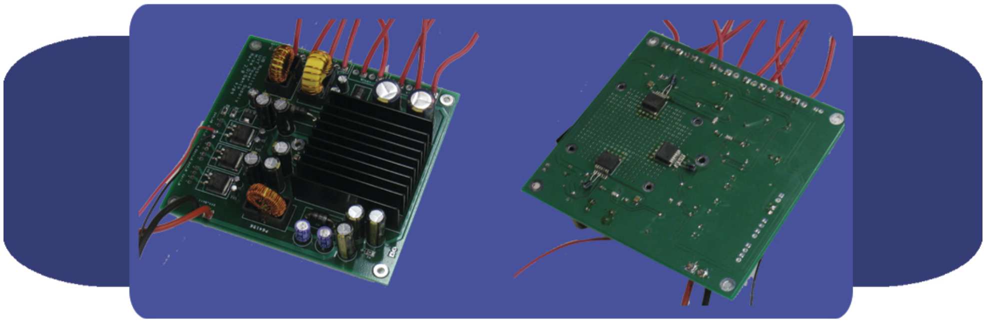


Figure 5 Power Supply Board

Big Blue was designed to carry two 24V battery packs to allow for a seamless transition when one battery pack is running low. With the flip of a switch on the power supply, the robot will draw its power from the second battery, and the first can be replaced. This promotes a

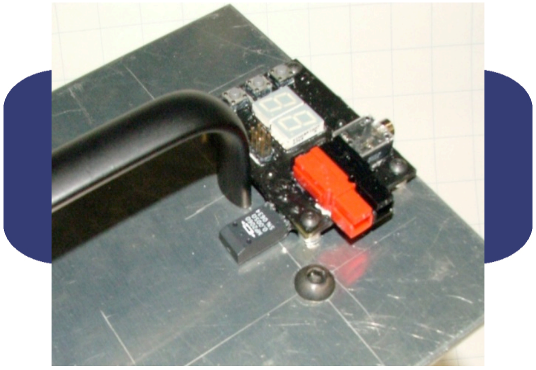


Figure 6 Battery Monitoring Board

healthy rotation of the batteries so that they will all age at the same rate.

In an effort to be able to intelligently know when to replace the batteries on the robot, a battery monitor was designed and built in to the battery pack to log the battery voltage and output current. It can output its readings to a seven segment display or over a USB interface to the laptop onboard the robot. The completed battery monitor is shown to the right.

Motors

The vehicle is propelled by four NPC Robotics T64 brushed DC motors, with each motor directly driving one of the robot's four wheels. They are run at 24 volts and have an output power of approximately 0.7 horsepower. Under low load at a slow speed, the motors draw about 5 amps. With heavier loads or while turning in place, they can take up to 15 amps. Without speed limiting, the robot can reach speeds of almost 10 mph. For the competition, this is limited in firmware by the motor controller down to the required 5 mph max speed.

Motor Controller

After experiencing a near-catastrophic failure of an off-the-shelf motor controller a few weeks before last year's competition, and noticing that its emergency stop control was a flash-configurable setting that occasionally reset itself, UB Robotics decided this year to build a custom motor controller this year. In addition, it was found that there are very few motor controllers that are intended to drive four independent motors of the size we are using. In going with a custom design, the emergency stop and remote control capabilities were able to be integrated into the design. A picture of the partially assembled controller is below.

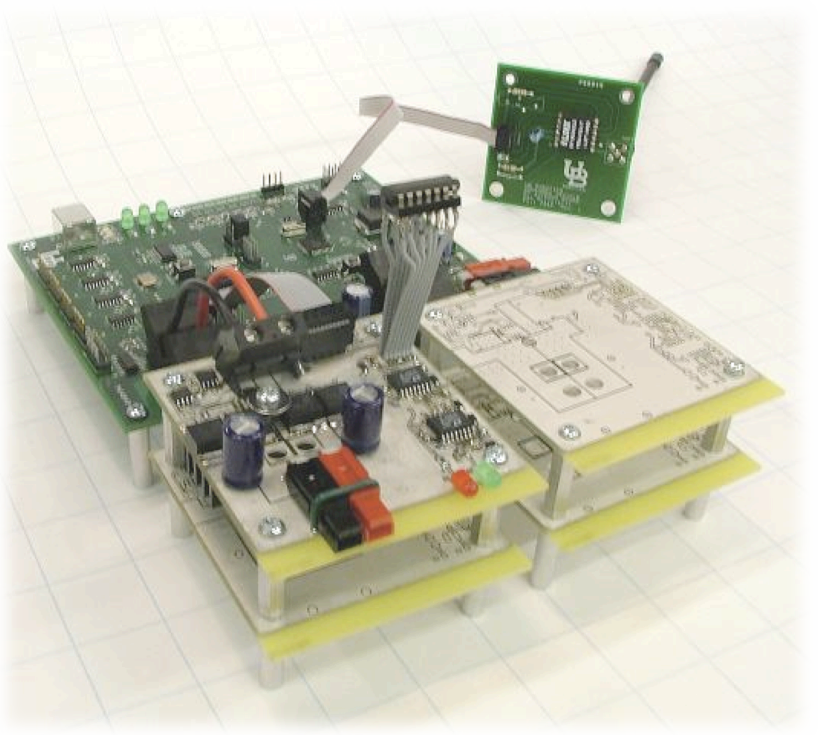


Figure 7 Motor Controller Board

Four interchangeable H-bridges, each capable of driving a motor at up to 50V and 30A interface with the main motor driver. The system is able to read the motor current, MOSFET temperature, and wheel encoder speeds and relay that information to the computer over a USB interface.

Unlike in many commercial designs, the hardware emergency stop will turn off all of the FETs through logic gates instead of through firmware. This is a much safer design, in that if the latching emergency stop button is pressed, the motors cannot run, even if the microcontroller was reset or had and experienced an error.

Lastly, the motor controller has a UART interface to a Linx Technologies 418MHz RF Transceiver through which it communicates with the remote control and wireless emergency stop.

Remote Control

A custom remote control and wireless emergency stop was designed for this project. Instead of using an analog hobby remote control, the system sends the emergency stop signal and joystick positions through digital packets that are checked for errors. This is a much safer way for a human to control the robot and a more reliable interface for emergency stop, as it removes the possibility for a stray servo signal to take control of the vehicle. The remote interface's connection to the computer (through the motor controller board) also allows us to load and start algorithms from the remote interface. To accommodate the custom electronics and controls, a rapid prototyped case was designed, which is shown to the right.

Figure 8 Remote Control

III. Software

Software was one weak point last year. In order to operate better autonomously, emphasis was put on mapping, navigation, and vision. The whole year was spent writing an entirely new robotic system.

Platform

Software was developed targeting the Java SE 6 development kit, with vision processing code being written in C++ using the OpenCV computer vision library. The computer used was a Dell Latitude D830 with a dual core processor and 2GB of memory.

Architecture

In the lowest level of software, raw data is read from the sensors and processed into a usable form. At this level, the vision processor takes camera frames and extracts line positions. The lines and LIDAR data are merged to give a single estimate of the position of obstacles around the robot. The Extended Kalman Filter (EKF) receives data from the differential GPS (DGPS), digital compass, and wheel encoders. Using state estimation algorithms, a better guess of the actual position of the robot in the world is generated. The obstacle data and position estimation are fed into the mapping module, which stores this data over time. The path planner then takes the position estimate, the map, and the goal and finds an optimal path between its location and the destination. The final path is then used to drive the motors to navigate the robot along the path.

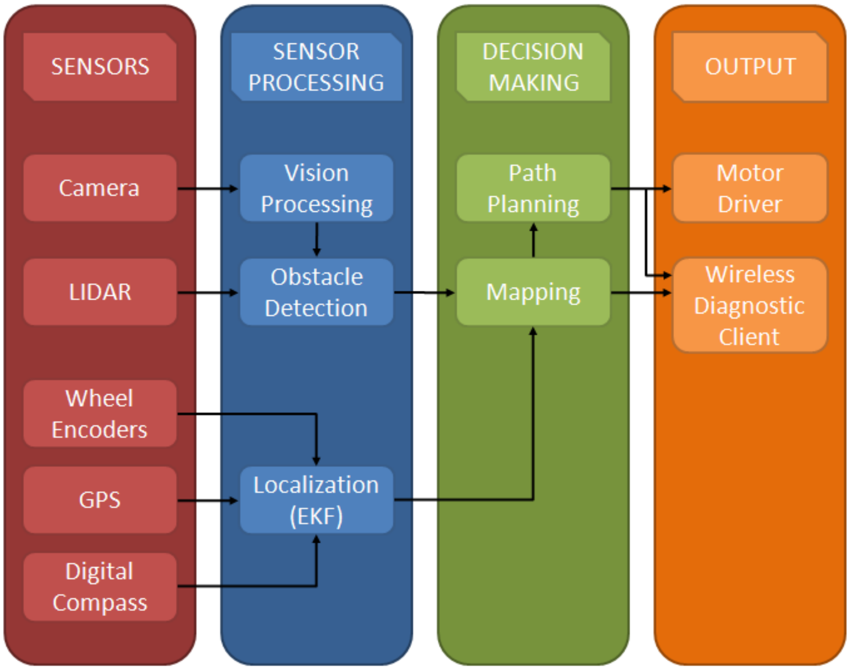


Figure 9 Software Architecture

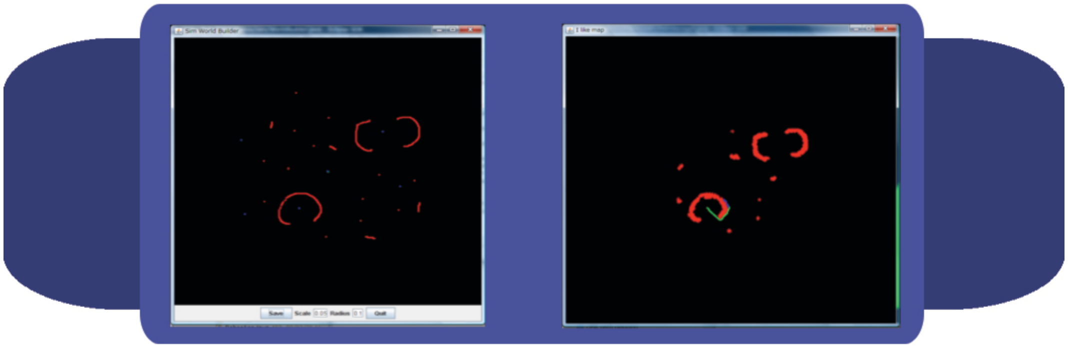
Portability

The software was designed to run on a variety of vehicle and computer platforms as the final vehicle would not be ready until shortly before competition. For safe indoor testing of path planning algorithms, an iRobot Create with an Asus Eee PC and SICK PLS 101 was used to navigate through simulated courses in a hallway. Last year's robot was used for testing outdoor path planning, vision processing, and hardware integration.

Figure 10 Left: Cornelius avoiding buckets; Right: Testing on last year's robot

Simulator

The simulator allows users to add variously sized circular obstacles of varying radii and waypoints though a GUI. This information is then fed into the software which then generates simulated LIDAR and localization data as if there was actually a robot moving through the world. Error is added to the data before it is handed to the robot making it more realistic. This allowed developers to test both localization and path planning code without needing the actual robot.



Scale 0.05 Radius 0.1

Figure 11 Left: Building a world; Right: The world the robot has seen

Mapping

A major feature that was lacking in last year's design was the mapping of obstacles that the robot discovered throughout the course. A common failure was seen when an object moved out of the field of view of the camera and the vehicle steered into it, terminating the run. The mapping system, which we called the Dual Map, maintained two levels of data. The global level, which included all data not within the robot's current field of view, was static and did not lose data over time. The local map, which included everything currently visible, was updated on each LIDAR cycle. Points that were reported as obstacles in previous scans but now appear clear in the current scan are removed. This helps to prevent map smearing caused by localization thereby increasing success of our path planning algorithm.

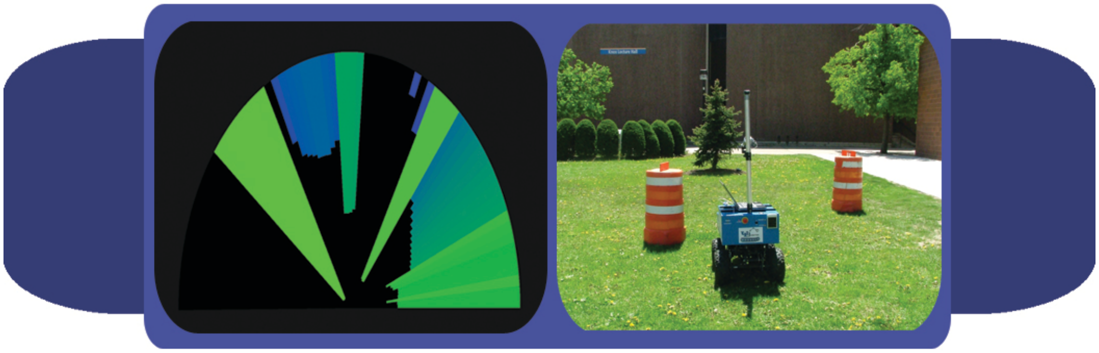


Figure 12 Left: Corresponding LIDAR output; Right: Actual scenario

Path Planning

Using the map built from LIDAR and camera data, the position of the robot from the localization module, and the goal, either a waypoint in the navigation challenge or some forward progress in the autonomous challenge, a path is planned using the A* graph search algorithm. Since the algorithm generates the shortest path, which may brush against objects, extra space is added around obstacles on the map to keep the robot at a safe distance.

Waypoint Navigation

Waypoints are added into the mapping system by transforming the latitude and longitude given into the Cartesian coordinate system using the World Geodetic System 84 (WGS84) and trigonometry. Waypoints are visited in the order specified by the use and the robot reaches them within a 30cm radius.

Vision

Vision grabs each frame from the camera feed and applies a series of stock OpenCV and custom algorithms in order to extract coordinates of lines from the frame. First, the frame is converted from color to grayscale (Fig. 13 Image 2), and then a histogram operation is applied to

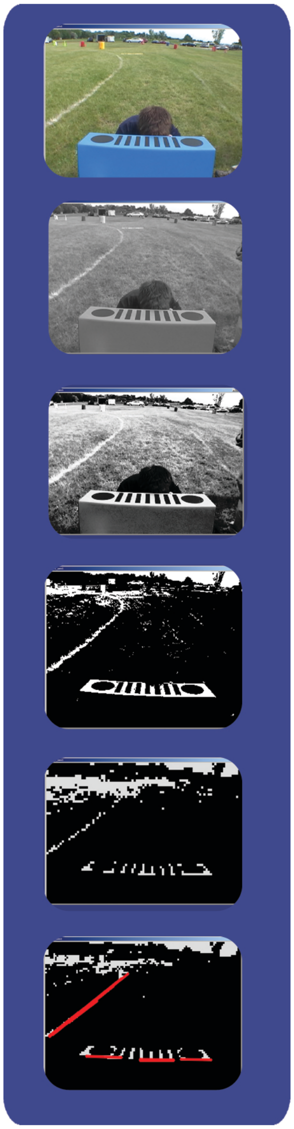


Figure 13 Image at each stage

enhance the colors (Fig. 13 Image 3). A threshold operation is then applied to the frame, removing channel intensities outside of a specified range (Fig. 13 Image 4). Noise is then filtered out by removing contiguous blobs less than a specified width and height (Fig. 13 Image 5). The only major features remaining in the frame are lines and obstacles such as cones and barrels. Our primary objective is to determine coordinates of lines, so by using the same noise masking operation we are able to filter out blobs greater than a specified width and height. A probabilistic Hough transform operation provided by OpenCV is then used to find lines within a specified range of pixel coordinates (Fig. 13 Image 6). This data is loaded into a packet and sent via TCP to the rest of the system.

Extended Kalman Filter

For good mapping and navigation a close estimate of your location is necessary. As detailed earlier, sensors are not perfectly accurate. In order to compensate, an Extended Kalman Filter (EKF) merges data and outputs a refined location. Essentially, it works by figuring out which sets of data are more accurate over time. It dynamically assigns weights to each sensor and averages the data. An EKF is used over other methods because of its ability to handle nonlinear equations. It is considered a standard for localization. More detailed explanations can be found in Fredrik Ordernud's paper *Comparison of Kalman Filter Estimation Approaches for State Space Models with Nonlinear Measurements.*

An EKF has two stages: Predict and Propagate. Predict estimates a new location based on new input data and creates a new covariance matrix. Propagate is a set of functions that updates our estimation.

IV. Performance

Based on early testing Big Blue is quick, agile, and all that we hoped it would be. It accelerates to max speed very quickly and climbed up every incline we tried; It ascended an approximately 55° hill without hesitation. We credit the tight, responsive controls to using four motors and having a low center of gravity.

Unfortunately, this power comes with a consequence. Our battery life is only 20 minutes per pack. We have two battery packs onboard bringing the total to 40 minutes of battery life.

Due to restrictions of the competition, the speed is limited to 5 mph, but we are capable of about 10 mph. The motor controller governs the max speed by monitoring the encoders.

Performance Results

Speed	5 mph
Reaction Time	Near Instant
Battery Life	20 minutes/pack (2 packs onboard)
Ramp climbing	55° +
Object Detection Distance	5 meters for lines/20 meters for objects
Waypoint accuracy	30 cm

V. Vehicle Costs

Component	**Retail Cost**	**Team Cost**
Dell Latitude D830 Laptop	\$1,200	\$0
Novatel Propak V3 DGPS	\$8,000	\$3,900
SICK PLS-101	\$5,000	\$215
NPC Motors	\$1,144	\$572
Batteries	\$250	\$250
PNI TCM-2.6 Digital Compass	\$850	\$0
Panasonic 3CCD color camera	\$800	\$0
Custom Electronics		
Motor Controller	\$725	\$525
Remote Board	\$250	\$250
Power Supply	\$260	\$260
US Digital E4 optical encoders	\$150	\$150
Mechanical Parts (Metal, hardware)	\$1,250	\$1,250
Anodizing	\$100	\$100
Total	\$19,980	\$7,472

VI. Conclusion

We believe out vehicle has been created to the best of our abilities. We are proud of what has been built and see it as a major accomplishment over our vehicle from last year. In future years we would like to put more effort on vision and navigation algorithms. Cheaper solutions for object detection can be obtained using multiple cameras, however it requires a significant amount of extra work. We plan to use the same mechanical design for at least one more year as we are very happy with how ours turned out.

UB Robotics would like to complete JAUS level 3 if time permits.

Acknowledgments

We would like to thank all of our sponsors: Novatel, Omnistar, PNI, Advanced Circuits, Sunstone Circuits, UB Student Association, Sub Board-I and the Energy Systems Institute (ESI) at the University at Buffalo for their product and money donations. A special thanks goes to our club adviser, Dr. Jennifer Zirnheld, for her continued support, as well as Kevin Burke, Jon McMahon and the rest of the ESI staff for all of their help.