PARADROID

Presented by the
IEEE Robotics Team
University of Wisconsin-Madison

Faculty Advisor Statement

I certify that the engineering design of the robotic vehicle described in this report, Paradroid, has been significant and equivalent to what might be awarded credit in a senior design course.

Professor Michael Zinn
Department of Mechanical Engineering

Table of Contents

1 Introduction

The University of Wisconsin-Madison IEEE Robot Team is pleased to re-introduce Paradroid to the 18th Annual Intelligent Ground Vehicle Competition. After our most successful competition to date, we have spent an entire year modifying and improving Paradroid, rebuilding several systems from the ground up. The goal of this project is to go beyond the challenge of the competition and design a versatile and adaptable platform that is useful for other applications as well.

The UW-Madison IEEE Robot Team is composed entirely of undergraduate students studying engineering and computer science. Each sub-team meets several times a week outside of class to work on their projects. All of our 30 members are volunteers, and no one receives course credit for this project.

2 Innovations

This year Paradroid was significantly upgraded from last year's design to address several shortcomings observed at IGVC 2009. Upgrades to the mechanical, electrical, and software systems were made.

2.1 Mechanical Innovations

Paradroid's footprint was decreased from 44 inches to 39 inches long by recessing the laser range finder under the payload tray. In addition to making the robot easier to maneuver, this change protects the laser range finder in the unlikely event of a front end collision and does not obstruct the view of the imaging system.

2.2 Electrical Innovations

Paradroid's previous electrical systems had a several shortcomings, including limited processing power and expandability. To correct these issues, the embedded control system has been upgraded to an ARM9 single board computer to provide more computational power and connectivity. The motor control system has been replaced with a team designed system that allows finer-grained control of the vehicle's motion. The emergency stop system has been upgraded to use an encrypted ZigBee wireless link and integrated with an operator control unit. Sensor upgrades include a new accelerometer and an improved camera system.

2.3 Software Innovations

Software changes from last year include improvements to our team-developed software framework, the Robot Simulation and Control Lab (RSCL) and several innovations in autonomous navigation. The internal data handling in RSCL has been redesigned to improve scalability and reaction time. The machine vision system has been upgraded and is much more robust under difficult lighting conditions. Paradroid also features rigorous simultaneous localization and mapping using an Unscented Kalman Filter and high resolution map, and uses a new portable implementation of the JAUS standard for its control interface.

3 Design Process

The development process for Paradroid began in September of 2007. The original mechanical and electrical designs were completed in spring of 2008, but the software system was not ready in time to qualify for the performance events at IGVC 2008. Following the successful return to the team-developed software platform in 2009, this year was spent streamlining and optimizing it to handle more complex decision-making algorithms. The mechanical and electrical systems have also been improved to make them even more versatile and reliable. The team spent an estimated 5000 hours over a period of nine months redesigning and improving Paradroid for IGVC 2010.

3.1 Team Structure

The UW-Madison IEEE Robot Team is a student organization comprised entirely of volunteers. The team consists of undergraduates from various engineering disciplines and computer science. The team is broken up into three sub-teams: mechanical, electrical, and software as shown in Figure 3.1. Each sub-team leader is selected based on past involvement and level of experience. An all-team meeting is held monthly to facilitate communication between the sub-teams. In addition, the software and electrical teams collaborate on a weekly basis to better meet each other's needs. Most major design decisions are made by a consensus of team members. When inevitable disagreements occur, all options are further researched and each party involved presents the advantages and disadvantages of their proposals to their sub-groups. If a consensus still cannot be reached, the decision is put to a vote.

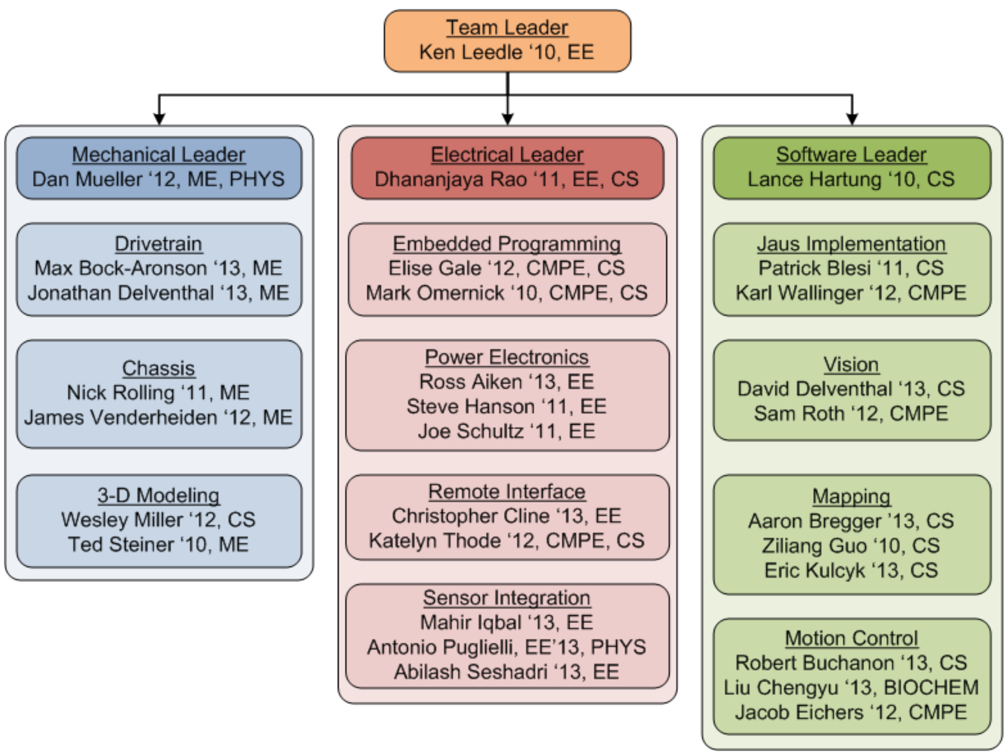

Figure 3.1: Team Structure

3.2 Team Development

Continuing recruitment efforts brought the team size to about 30 members, about half of whom have been on the team for a year or more. This one-to-one ratio made it easier to integrate new members, many of whom had little to no experience in engineering or software development. New members explored their talents and interests through hands-on training and guided group projects. In the spring, many of our new members took charge of their own projects, including the operator control unit, motor control system, and mapping.

3.3 Project Planning

The planning process began at the first meeting of the year, when Paradroid was compared to other top performing robots from IGVC 2009. Several new features for Paradroid were discussed, and the approved features were broken down into smaller tasks for each of the sub teams. Tasks and progress were tracked using an online project manager, which allowed the team to document design decisions and their justification, post important documents for review, and set team priorities. This system made interdependence of different features easy to understand, and allowed the team to adjust its priorities based on which tasks would add the most value to the robot given the time remaining until competition. This system was easy to use and update, and will provide crucial documentation for future team projects.

3.4 Development

The mechanical, electrical, and software sub-teams each used their own development processes suited to their task requirements. The mechanical team, whose work consisted of mostly hardware design, used a stricter phase-based development process. Conversely, the software team used the agile methodologies to allow for easier adaptation to the changing scope of their projects. The electrical team, whose projects involved both hardware and software design, used a combination of both development processes.

The mechanical team's development cycle consisted of computer-aided design, prototyping, production, and testing phases. SolidWorks, a computer aided design program widely used in industry, was used to model each component in the vehicle. By using SolidWorks, many ideas could be visualized very quickly without cost, and components could be tested for interference and proper interaction before they were built. From time to time, experts on campus were contacted about how to best solve a specific design issue in the most efficient and effective way. After designs were completed and tested on a computer, prototypes were often built for proof-of-concept testing. If the prototypes worked, the designs were finalized and parts were manufactured in house.

The electrical team followed a similar development process for their hardware design, using computer aided design and prototyping whenever possible. Custom boards were designed using EAGLE, a computer aided design printed circuit board layout tool. The embedded boards were prototyped when possible using breadboards before committing the design to a printed circuit board.

The software team carried out much of its development using pair programming techniques. This reduced the amount of debugging needed and resulted more legible code. Pairs worked on individual components and unit tests for the components. When unit tests passed, they moved on to testing their component in conjunction with other components. The software team also focused on producing working revisions of software whenever possible. The use of a modular software framework made this relatively easy, because nonfunctioning components could be kept in the root of the versioning repository without being included in a build.

4 Mechanical Design

Paradroid is designed to be a reliable, safe, accessible, power and space efficient vehicle. The goal of the mechanical design is to provide a versatile platform that software and embedded systems can be easily tested upon and interfaced with. It embraces a modular design for reusability and upgradeability. The design structure consists of a drive base and separate modules for the embedded system, custom built computer, laser range finder and main sensor module. The drive base features independent shock absorption for each wheel to handle rough terrain. The entire robot is weather resistant from the outside but the compartments are open inside for ventilation and air circulation. Each compartment can be accessed simultaneously for maximum serviceability. The vehicle was designed and manufactured completely from scratch by team members.

4.1 Drivetrain and Suspension

Paradroid is a differentially steered, four wheel drive vehicle with a fully independent suspension. The left-side and right-side wheels are powered by 24V DC wheelchair motors rated at 0.9HP continuous duty. These motors provide ample power for turning and climbing ramps as steep as 21.4°, as well as curbs and small stairs. Although the vehicle's top speed is 5.7mph, it is limited to 5 mph by the control software. This allows Paradroid to maintain a speed of 5 mph when climbing ramps and traversing rough terrain. Differential steering was chosen because it requires few moving parts and provides a zero turning radius for maneuverability. Paradroid has 16 inch five-spoked pneumatic tires in the front and 16 inch custom built omni-directional wheels in the rear (Figure 4.1).

Figure 4.1: Paradroid's Omni-wheels

The combination of pneumatic tires and omni-directional wheels gives Paradroid the traction and terrain handling capabilities of a four-wheel drive vehicle and the turning efficiency of a two wheel drive vehicle. Each omni-wheel has 10 circumferential lateral rollers which allow the wheels to transmit forward or reverse torque as well as roll side to side with little resistance. These wheels reduce power consumption by decreasing lateral friction while turning compared to standard four wheel drive differential steering (Table 4.1). The large 1.4" to 3" diameter of these rollers allows them to roll over most small obstacles without difficulty or reduction in ride quality.

Table 4.1: Comparison of Omni Wheel and Standard Wheel Power Requirements

Wheel	Movement Type	Torque	Power
Omni-Wheel	Zero Radius Turn	38 N-m	480 W
Standard Wheel	Zero Radius Turn	143 N-m	1810 W

The suspension system is designed to allow Paradroid to traverse a wide variety of terrain conditions. The suspension has been optimized for a 225 pound vehicle, which is Paradroid's operating weight with a 20 lb. payload. Used in conjunction with the pneumatic tires, the suspension cushions the ride for the electronic and mechanical components, reducing vibrations and fatigue. The unique coaxial drive sprocket and suspension pivot allows the drive modules to maintain chain tension throughout the range of travel.

The suspension arms use a single pivot design for simplicity and reduced cost, and they are long enough to have an effectively vertical path of travel. Each side of the drivetrain is a self-contained module, allowing it to be individually tuned, as well as mounted on virtually any platform with 5 mounting holes. The drivetrain incorporates parking brakes controlled by the electrical system that are automatically activated in the event of an electrical failure.

4.2 Chassis

Paradroid's chassis consists of a minimalistic steel skeleton to support all modules. The core of the frame is a simple rectangle on the top and bottom; everything else on the robot is modular. This design allows any individual component to be augmented independently of any other component, while securing all components of the robot to the frame. The steel frame provides superior strength and durability, while reducing vibrations. Low cost sheet metal covers the outside and protects against the environment and minor impacts. The chassis has a minimum of 7 inches of ground clearance, but under normal operation 8 inches of ground clearance is maintained. The battery compartment is located in the bottom of the vehicle to provide a low center of gravity and improve wheel traction. Each battery can be exchanged with a fully charged battery in less than one minute to maximize run time. Alternatively, the batteries can be charged inside the vehicle via a charging port conveniently integrated into the back of the robot.

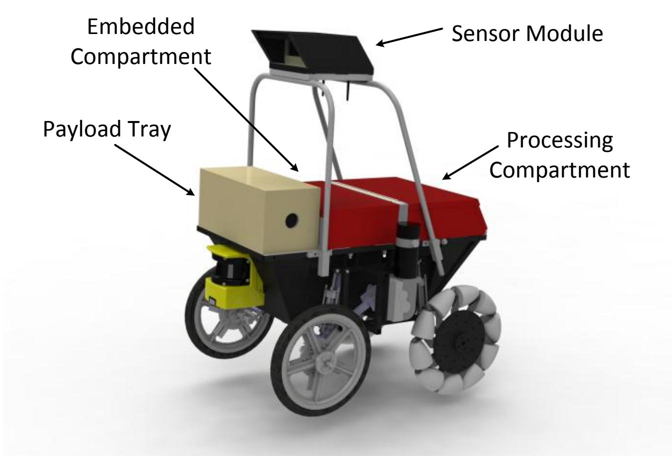

Figure 4.3: Paradroid's Compartments

4.3 Main Platform

The main platform consists of three compartments: embedded, processing, and payload (Figure 4.3). The embedded compartment is located in the middle of the platform on top of the chassis. This compartment houses the motor controller, power supplies and boards for interfacing with all of the robot's sensors. The open bottom allows air to circulate between this compartment and the battery compartment and facilitates routing wires and cables. All of the components are located on a subpanel which can be easily removed for service. Additionally, the entire compartment can be easily removed via four screws.

The processing compartment is situated on the rear of the platform for easy access. It is designed to house either a laptop or a custom built computer. Similar to the embedded compartment, it provides protection from impacts and the environment. This compartment has two doors that allow unobstructed access to the computer. In addition, the flat lid provides an ideal location for a laptop to be placed to interface with the internal computer during testing.

At the front of the platform is the payload tray. This versatile tray can accommodate payloads of various shapes and sizes weighing up to 50 lbs. Because the payload is secured to the front of the robot, it applies more weight to the pneumatic tires to reduce vibration and increase traction. This is an ideal location for a sensory payload which, in a military or commercial application, might be the primary purpose of the robot.

4.4 Sensor Module

The main sensor module is mounted directly above the embedded and processing compartments and houses the majority of the sensors on the robot. It is designed with the same frame and covering style as the embedded and processing compartments. The camera, GPS sensor, digital compass, emergency stop and wireless router are housed in the sensor module. This compartment is directly above the pivoting axis of the robot in order to keep the camera and sensors in a fixed reference during turns. The entire outer shell of the sensor module is attached with a hinge and can be easily lifted out of the way, allowing unobstructed access to all components housed within. The module stands 4 feet off the ground, which keeps the profile of the robot low.

Paradroid also has the option of using a laser range finder which can be mounted underneath the payload tray for a 180-degree view of obstacles in front of the robot. In this position, the laser range finder is protected from accidental collisions and does not obstruct the view of the cameras.

5 Electronics Design

Paradroid's electrical systems are designed to provide a simple interface between the software and the mechanical systems, delivering efficiency and expandability without compromising features or safety. The electrical systems handle all low-level sensor interfacing and provide power to all system components.

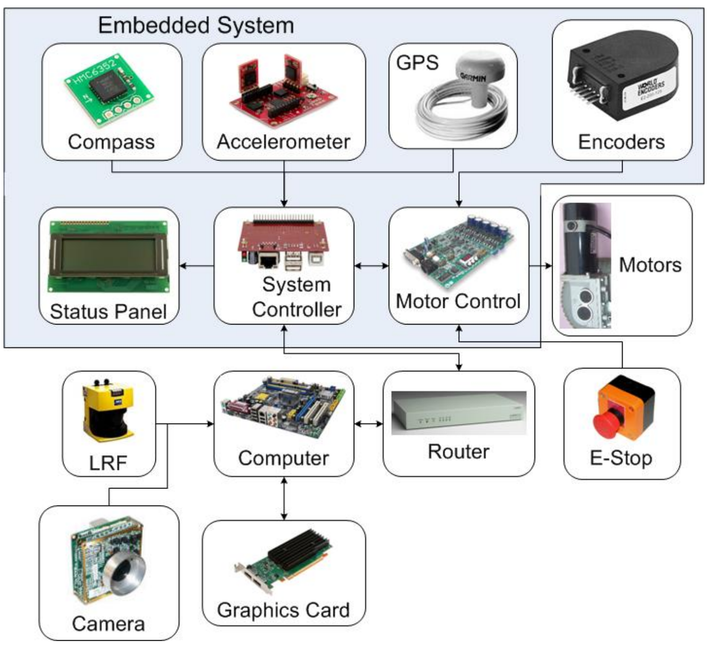

Figure 5.1: Electrical System Diagram

5.1 Embedded Control System

The core of the embedded system is a Technologic Systems TS-7500 ARM-based single board computer (SBC) running Linux. The primary purpose of this system is to relay commands and data between the main computer and peripheral boards. It interfaces with the main computer via a 100BASE-TX Ethernet connection, which enables simple and reliable high-speed data transfer.

In addition, the SBC provides a number of other interfaces, including I2C, SPI, RS-232, and digital I/O. A team-built expansion board routes these interfaces to generic modular connectors according to appropriate standards or manufacturer guidelines for maximum flexibility and compatibility with other systems.

5.2 Remote Control and Communications

The robot uses a Linksys WRT54GL wireless router for communications between internal systems and with the outside world. This enables the robot to interface with any computer or mobile device supporting 802.11b/g for remote control or data logging. Several command/control protocols are supported, including JAUS, a custom-designed software interface, and low-level SSH access.

We have also built a dedicated Operator Control Unit (OCU) to allow manual control of the robot and monitor vital statistics during operation. The OCU communicates with the robot using 900MHz ZigBee modules which offer excellent reliability and performance. The system supports automatic channel hopping, power modulation, and 128-bit AES encryption of the data link, making it secure and extremely resilient to electromagnetic interference potentially encountered during operation. Additionally, the modules provide a line-of-sight range of up to 6 miles which is far greater than required in any reasonable usage scenario for this robot.

5.3 Motorcontroller

Paradroid's motor control system has been upgraded for 2010 as well. Paradroid now features a team-designed motorcontroller to replace the Roboteq AX3500, allowing motor speed commands and odometry readings to be updated much more frequently. This enables the robot to perform more accurate path execution and localization. The new motorcontroller is passively cooled and can easily handle the power requirements of Paradroid's motors.

5.4 Main Processor

Paradroid uses a custom-built micro-ATX computer as its main computational platform. It provides substantially more processing power than an equivalently priced laptop, and is tightly integrated with the rest of the robot, which reduces the risk of damage to the computer and peripherals during operation in tough environments. This allows Paradroid to quickly analyze sensor data and react to changes in its environment. The computer is outfitted with a 2.5GHz Intel Core 2 Quad Processor, 4GB DDR2 RAM, and 4GB of solid-state permanent storage. A 23W passively cooled nVidia Quadro graphics card is also installed for parallel data processing (GPGPU). The computer has a removable 15" LCD monitor and mini-keyboard for adjustment in the field.

5.5 Sensors

Paradroid's camera is a HEI-CCD-USB from Howard Electronic Instruments. This camera provides a color 1024 x 768 picture at 30 FPS over USB. The camera also has a zoom lens set to 80° field of view and an adjustable iris so the camera can adapt to various light conditions to prevent over or under exposure. Paradroid also has a SICK PLS101-112 laser range finder to provide 180° single-plane sweeps of the area in front of the robot with 0.5° angular and approximately 70 mm radial resolution. The range of the laser range finder is restricted to eight meters. To supplement the limited view of the laser range finder, rear-facing infrared proximity sensors are used to detect obstacles during sharp turns and backing up. Other sensors include a Garmin GPS17HVS Global Positioning System receiver with one meter accuracy, Honeywell HMC6352 digital compass, an inertial measurement unit (IMU) with six degrees of freedom, and quadrature wheel encoders.

5.6 Electrical Safety Features

The new wireless emergency stop system is integrated into the OCU outlined in Section 5.2. The system uses a side-channel provided by the ZigBee link to communicate with the robot independently of the OCU data stream. This dedicated data link is more reliable due to the low bandwidth requirements. To ensure maximum safety, the emergency stop on the robot is triggered if this link is broken. This system is augmented by a physical emergency stop switch on the frame of the robot. The emergency stop directly cuts power to motors, bringing the vehicle to a complete stop within 18 inches of travel. In the unlikely case of complete power failure, electromagnetic parking brakes engage, bringing the vehicle to a complete stop in less than one foot.

Paradroid also includes a warning light and an optional 110dB air horn to provide visual and auditory warnings during operation. All electrical systems on Paradroid are fuse protected to minimize risk of damage in the event of an electrical failure. In addition, the power converters incorporate over-voltage, under-voltage, short-circuit, and electrostatic discharge protection, making the power system robust under a wide variety of conditions.

5.7 Power System

Paradroid's power system is designed to maximize vehicle run time and make software development and testing as easy as possible. Power is derived from two 12V deep-cycle lead acid batteries that form a 24V nominal battery pack with 75AH capacity. This battery system provides power for up to three hours of operation under normal conditions and up to ten hours in standby mode. The long battery life and integrated charging port allow Paradroid to run nearly continuously. In addition, depleted batteries can be replaced in less than one minute to maximize run time in the field. Power conversion using team-built DC-DC converters provides 24V, 12V, 5V, and 3.3V power to the various systems on the robot at 85-90% efficiency. Paradroid also features a 24VDC to 120VAC inverter for powering monitors and laptops during testing. There is also an optional light for the embedded compartment and a 40W headlight for operating the vehicle in low-light conditions.

Table 5.1: Power System Requirements

	Normal Operating Conditions			Worst-Case Conditions		
Device	Volts	Amps	Watts	Volts	Amps	Watts
TS-7500 Single Board Computer	5	0.4	2	5	0.4	2
Linksys WRT54G Router	12	0.4	4.8	12	0.4	4.8

Garmin HVS17 GPS	12	0.1	1.2	12	0.1	1.2
HEI-CCD-USB Camera	12	0.25	3	12	0.25	3
Misc Electronics	5	0.5	2.5	5	1	5
Sick PLS101 LRF	24	0.8	19.2	24	1	24
Main Processor DC-DC supply	24	3	72	24	4.5	108
Wheelchair Motors (2)	24	10	240	24	120	2880
			Total Watts			Total Watts
			344.7			**3028**

6 Software Design

Paradroid's software system is an evolved version of the Robotics Simulation and Control Lab (RSCL), a framework originally designed by the team in 2005. The primary goals of the software are to be a training platform for new team members and to meet performance requirements. As such, this general approach emphasizes modularity, with each aspect of autonomous operation being handled by discrete components dedicated to performing a specific task. These components are tied together in a framework that is designed to be efficient and flexible.

6.1 Structure

The software stack comprises several layers, with the lowest level containing servers that relay data from individual devices. A common design feature throughout our software is an event-based subscriber model. This is a marked difference from the previous incarnation of RSCL, where data recipients were run at set frequencies, and data sources only supported one receiver. This purely synchronous model had certain advantages in predictability, but because of the power of modern computers it resulted in an overall decrease in performance. With an asynchronous model, various data consumers run on demand and ultimately keep the robot more up to date on its surroundings.

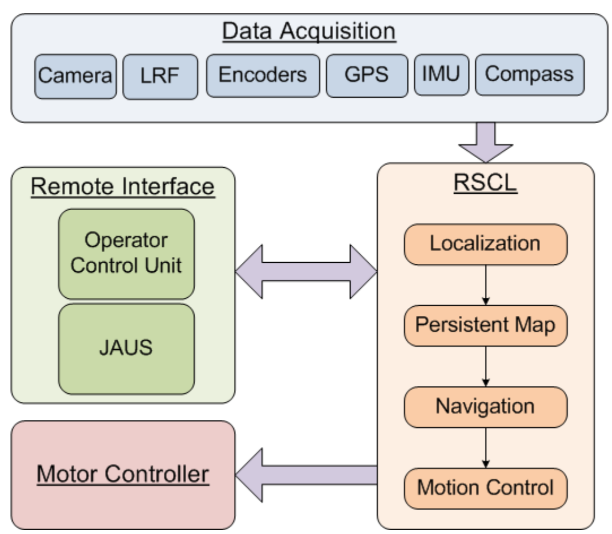

Figure 6.1: Software Structure

The primary recipients of this data are the sensor fusion and mapping/pathfinding modules. The sensor fusion module refines the data received from the various sensors to create a best guess for the current location of the robot, which is then additionally passed on to the mapping/pathfinding module. With the location data, the mapping module can correctly orient the data from the laser range finder and camera to mark obstacles in the world. This map is used by the pathfinding code to search for a clear path, which is translated to driving instructions for the robot.

6.2 Vision

The primary focus of the vision code is line detection, as this is the most important data that cannot be derived from the measurements obtained by the laser range-finder unit. The procedure for finding lines is split into three portions: identifying likely points where lines exist, fitting splines to these points, and inferring their positions in real space. To identify the positions of points that likely make up lines in an image, the image is scanned horizontally and then vertically, searching for relative maxima in the image brightness. These points are then grouped in accordance with their location. These groups are then geometrically analyzed and the groups that do not resemble lines are culled. The remaining groups continue to the spline-fitting module. The splines are then transformed so that they can be located in real space as opposed to the camera plane.

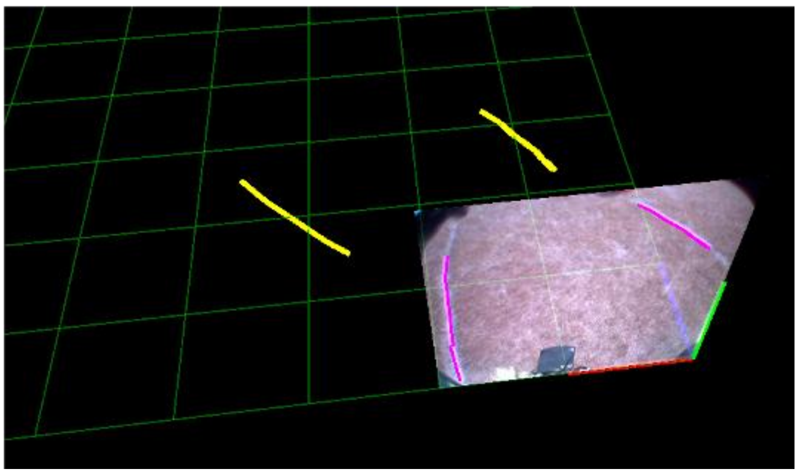

Figure 6.2: Robot's view of the world. The lines as seen by the camera are shown in magenta, while the lines in actual space are shown in yellow. The grid is represented using global coordinates.

6.3 Localization and Mapping

6.3.1 Localization

Successful mapping relies on having highly accurate information about the robot's pose and velocity in the environment. Robot localization systems typically accomplish this through sensor fusion via Kalman filtering, which is a rigorous, probabilistic technique for incorporating multiple noisy sensor readings into a state estimate. This was implemented using an Unscented Kalman Filter (UKF), which outperforms the ubiquitous Extended Kalman Filter (EKF) for highly non-linear transformations and does not require computing Jacobian matrices. These benefits come at negligible computational cost over the EKF. A combination of sensors, including quadrature encoders, GPS, three-axis accelerometer, gyroscope, and compass, provide complementary observations of the robot's current state. The UKF is dynamically configured based on sensor availability and uses dead reckoning when GPS is unavailable. By necessity, the localization module operates on a strict real-time deadline. It uses a physical motion model based on velocity and acceleration to provide state estimates between sensor readings.

6.3.2 Laser Range Finder Scan Matching

The laser range finder (LRF) serves both to identify obstacles around the robot and as a localization mechanism. A brute force scan matching technique that relies on the highly parallel nature of graphical processors is used to refine odometry. First, a single LRF scan is used to create a 2D lookup table. Each detection point is added to the probability map according to a two-dimensional Gaussian, which implies nonzero but decreasing likelihood with distance. A subsequent scan is then taken, rotated and displaced within a specified range and the resulting position of each LRF point is used as an index into the lookup table. The probabilities from each lookup are summed and the rotation and displacement that results in the highest sum is treated as the most probable change in the robot's position. The estimate of velocity and turning rate obtained from the encoders and IMU to narrow the search space for scan matching and subsequently refine the odometry. The scan matching algorithm runs approximately twice a second, which in conjunction with the UKF, provides adequate performance for navigation.

6.3.3 Mapping

The mapmaker module maintains a persistent map of the world, updating it with respect to objects found by the laser range finder. There are two maps - an occupancy grid in the form of a probability map and a navigation mesh. The occupancy grid maintains the persistent view of the world and can handle both static and dynamic objects surrounding the robot. The resolution of the occupancy grid is one square centimeter, chosen based on the resolution of the LRF. The algorithm updates the map according to a Gaussian mixture with means centered on the LRF detection points. This mixture is approximated to minimize computation by operating on the segment of the map between the obstacle and the robot, assigning decreasing occupancy probabilities to each cell beginning from the obstacle. This approach allows the map to account for moving objects, since the probabilities will decrease toward zero as the LRF scans cleared space. To allow for a dynamically scaling map, the global occupancy grid is composed of 15m by 15m blocks. As the robot moves beyond the range of a single block, a new one is created and connected to the existing grid.

Once the global occupancy grid has been updated, two algorithms are run over it to generate a navigation mesh. Navigation meshes are an idea taken from the video game industry, but for the most part meshes are not automatically generated in video games. However, the ease with which pathfinding algorithms can be run over meshes made them very enticing. The first algorithm works on the premise that the robot cannot drive through any two points on the map that are closer together than the width of the robot. These points can be connected together into lines which the robot cannot drive through. An assumption is made that any point behind the line is unreachable and thus ignored until the robot can directly observe it from a new position. The second algorithm uses this output to dynamically generate navigation meshes in the form of Voronoi diagrams using an incremental Delaunay triangulation. The set of points used to form triangles is the set of endpoints of the lines mentioned above.

6.4 Navigation

Navigation involves optimizing a path through the map and requires a series of destinations in the global coordinate system. For the navigation challenge, these are given in the form of GPS waypoints. For the autonomous challenge, waypoints are set seven meters ahead of the robot at the center of the lane acquired through vision.

Again borrowing from video games, the pathfinding operation is an A* search run over the navigation mesh. The center points of each line on each triangle in the mesh are used as the nodes for pathfinding. This allows for maximum distance between the robot and an object. However, the downside of arbitrarily choosing the center point of the line is that the path can become too jagged or the lines too short for the robot to realistically follow. A path optimization and smoothing algorithm was devised to solve this problem. The sharp turns in the path were incrementally smoothed in the flavor of gradient descent. To do this, a coefficient is calculated for each edge which represents how jagged it is. By moving the endpoints of an edge away from the center of the triangle's line, the path becomes less jagged. The optimum driving path for the robot results from incrementally changing the most jagged edge at each step.

An algorithm from reinforcement learning discovers the optimal driving function for a precise path using a lookup table containing the state as distance to and angular difference from the target point and the possible actions as motor speeds. The motor commands appropriate for any situation are discovered by applying Q-learning with a punishment proportional to distance and angular correction required to reach the target.

7 JAUS Integration

A single, light-weight framework acts as a code-base for our client-side and server-side JAUS implementation. This framework provides all message encoding, decoding, sending and receiving functionality. This code-base was developed with the following specific features in mind: ease of implementation, full JAUS compliance, and clear separation of robot integration code and JAUS specification code. This implementation allows for complete custom robotics functionality and behavior without the need to modify a single file from the original code-base. This provides the robot with complete JAUS compliance, a fully functional GUI application for controlling the robot, and an Android application that can control the robot. In addition, this reusable code base will allow JAUS functionality to be easily ported to future robots, as well as distributed as a functional solution for others to make their robots JAUS compliant.

8 Cost Summary

Ideally, the team would design and manufacture all components on the robot for the experience it would provide. However, several components are too expensive to make in small quantities, require access to specialized equipment, or are simply beyond the level of undergraduate work. These components, such as motherboards, motors, the GPS, and others, were purchased, saving both time and money. The vast majority of the components on Paradroid were designed and manufactured by the team, including the suspension, frame, power supplies, and operator control unit. Most of the software is written entirely by team members. In many cases, code originates from various open source projects and is updated or improved upon. After the competition, these improvements will be returned to their respective projects so that others may benefit from these improvements as well.

Table 8.1 Estimated part costs for the project, including those purchased prior to this academic year.

System	Item	Qty	Cost	Our Cost
Mechanical	1008 Low Carbon Steel / 6061 Aluminum	1	$550	$0
	Misc Hardware	1	$200	$0
	UHMW Polyethylene (6 ft)	-	$98	$0
	Pneumatic Tires / Bearings	4	$105	$0
	Wheelchair Motors	2	$210	$0
	#35 Roller Chain / Sprockets	4	$92	$0
Computer	Main Board - Foxconn G33M-S Micro-ATX	1	$95	$0
	Processor - Intel Q9300 Quad Core	1	$280	$0
	Graphics Card - Quadro NVS 290	1	$120	$120
	Solid State Memory - 4GB	1	$65	$0
	Memory - 4GB DDR2 800	1	$61	$0
Vehicle Control	TS-7500 Single Board Computer	1	$130	$130
	Interface Board – PCB & Parts	1	$50	$50
	Motor Controllers - Team Designed	1	$300	$300
	Wireless Router - Linksys WRT54G	1	$50	$0
	Operator Control Unit - Team Designed	1	$300	$300
	Wire and Interface Hardware	-	$100	$45
Sensors	HEI-CCD-USB Camera	1	$799	$799
	Quadrature Shaft Encoders - HEDS-5600	2	$120	$0
	GPS - Garmin 17HVS	1	$112	$0
	SICK PLS101-112 Laser Range Finder	1	$4,500	$0
	HMC6352 Compass Module	1	$35	$0
	Accelerometer	1	$125	$125
Power	Embedded Power Supply – PCB & Parts	1	$30	$20
	ATX Power Supply - M4-ATX 250W	1	$100	$0
	Batteries – 75Ah 12V Deep Cycle Lead-Acid	2	$120	$0
	Battery Monitor/Status Panel	1	$80	$0
Total			**$8,827**	**$1,889**

9 Performance and Conclusion

The relatively high top speed and improved independent suspension system allow Paradroid to move fluidly over a variety of terrain at any speed up to 5mph. The combination of high-power motors and 4-wheel drive allows Paradroid to climb slopes and curbs with ease, while the omni-directional wheels make it easy and efficient to maneuver. Paradroid quickly reacts to obstacles using its powerful on-board computer and optimized localization and path planning algorithms. The large lead-acid batteries provide ample run time for testing and can last all day under intermittent use.

Performance Parameter	Prediction	Result
Top Speed	5.0 mph	5.7 mph
Ramp Climbing	15°	21.4°
Curb Climbing	3"	6"
Reaction Time	0.1 sec	0.2 sec
Battery Life	2 hours	3 hours
Detection Distance	15 feet	17 feet
Waypoint Accuracy	1.0m	1.0m

Paradroid was originally designed with both military and commercial applications in mind, and with the hope of advancing the field of unmanned ground vehicles. In that same spirit, Paradroid was redesigned and improved this year to be a strong competitor in the 2010 Intelligent Ground Vehicle Competition. Paradroid's modularity, versatility, and efficiency have proven it to be an ideal platform for autonomous vehicle research.