AUTOMATON

Michael Scherer, David Adams, Jake Carr, Jonathan Mohlenhoff, Robin Adams

May 8, 2011

1 Introduction

The Robotics Club at the University of Central Florida is proud to present AUTOMATON, a brand new rugged and robust platform that builds on the collective experience of many minds. Construction began in fall 2010 by a small team of dedicated individuals of various ages and expertise who collaborated to produce a competitive platform. Team members were organized based on their respective fields: software, mechanical, and electrical engineering. The goal of this effort is to produce an extensible vehicle which can be built upon by future Robotics Club members in the years to come, by standardizing documentation and code, and by promoting open standards and open-source software.

2 Design Process

The Agile Development methodology is used for the ongoing creation and deployment of AUTOMATON. It was chosen for its iterative design process which permits the direction taken to be changed dynamically as new problems and design requirements arise. This paradigm encourages frequent releases of working software, shared between developers and customers using an open-source Subversion repository called Zebulon. The Agile design process focuses on direct communication, face to face when possible, to ensure clarity of design and system requirements. Specific attention is given towards the quality of work, where working iterations are used as a measure of progress.

Figure 1: Agile Development Cycle

Figure 1 demonstrates the multiple loops involved in the Agile Development strategy. Closer to the center are continuous iterations of development, analogous to the daily software releases of AUTOMATON's code. Further from the center are the more abstract goals. This multi-loop system enables the team to develop in a very dynamic way, reducing risks associated with challenges faced throughout the design process.

Team Member	David Adams	Jake Carr	Johnathan Mohlenhoff	Michael Scherer	Robin Adams	Total Man Hours
Mechanical Hours		300				1250
Electrical Hours			120	100		
Software Hours	350			250	80	

3 Mechanical

Automaton was designed with four, direct, drive wheels that support an aluminum frame. Four wheels give the calculated required torque, while the aluminum saves weight. All electronics are housed inside the frame except the camera, GPS receiver, and LIDAR. The camera and GPS receiver rise above the vehicle on the mast, while the Hokuyo laser sticks

out the front leading the vehicle. This central body design gives Automaton a clean look while maintaining high functionality and weather resistance.

3.1 Frame

Our overall frame size is a rectangular prism shape 1 ft. by 2 ft. by 3 ft. It is made mostly out of 1 in. x 1 in. aluminum tubing for weight saving. Four pieces on the bottom of 1 in. x 2 in. tubing provide a wide, flat surface for motor mounting and extra structural support. Also on the bottom is an 'X' pattern to provide extra stability for the two heavy car batteries. On the top of the frame, we incorporated cross braces that are spaced apart to exactly support the payload while allowing easy access to our batteries. Welded on the back of the frame is a sleeve for the mast to slide in which supporting the camera and GPS receiver.

3.2 Drive System

The team's calculations indicated that the four power chair motors used are powerful enough for the specifications of the course. The only modifications made to them were adding the encoders with a fixture plate in place of the brake. The fixture plate design allowed for four different mounting positions of the encoders that enabled us to run the wires easily for a clean look. To mount the motors to the frame, a fixture plate was fabricated that provided a flat surface for the motors to mount over the welds. For easy steering, the motors are located center on the frame in an exact square pattern measured from the apex of the center of the tires. The hubs used were made for the motor spindles, held in place by a key way. No two piece wheels were available that fit the hubs, so an interface plate was designed and manufactured. This made the two compatible while allowing extra clearance room between the tires and the frame.

3.3 Fabrication

The lengths of tubing used for the frame were saw cut on a horizontal band saw with active cooling, and then TIG welded together into the rectangular prism shape. The plates to mount the motors, the plates to interface the hubs to the wheels, and the 'L' shaped piece to mount the camera box were all manufactured on a precision three axis CNC machine with cooling system. The camera box is made of cast aluminum and purchased as a whole. Then the mounting holes, designed to align with the mounting piece, and lens hole were precision machined into it. The laser mount was designed to allow the laser the widest range of view without being unstable, and was precision CNC machined to match the laser mounting holes. The sheet metal exterior and trim pieces were cut on a vertical band saw and riveted to the frame, giving the vehicle a shiny, sleek look. The polycarbonate lid was cut to length on a vertical bandsaw and accented with the stylish trim. The tinted finish of the polycarbonate gives a peek inside the electronics.

3.4 Innovative Features

One innovative feature of the vehicle is the box that was designed to house the video camera that is being used. The design of the camera box aligns the center of vision with the center of the vehicle as well as allowing adjustability of the vantage angle of the camera. Being centered on the vehicle, the cone of vision produced can easily be implemented into vision algorithms. The adjustability of the camera allows the user to get exactly the range of vision desired. Inside the box, the camera is held in place with two plastic screws to

eliminate any interference between the camera and the box. Cables are run up the mast and quietly sneak into the box through the cut-out positioned right above the center of the mast. This increases weather resistivity while maintaining a clean look. Cushioning the camera from vibrations is a piece of sorbothane sandwiched between the box and the camera. On top of the box, which allows for the strongest signal, sits the GPS receiver, the highest component of the vehicle. It is centered towards the front of the box to allow for maximum height of the camera box. The whole mast and camera assembly is removable from the sleeve for easy transportation.

Another innovative feature is the lid system on the main frame. Separated into two pieces, the lid allows easy access to the on-board electronics. It also contains the chords of the light and resists the weather. The front section of the lid is completely removable and allows access to such electronic components as the batteries, compass, and GPS. The rear section is hinged from the frame and opens to reveal the on-board computer and motor controllers. The light on the back portion of the lid is conveniently flipped up while the wires stay out of the way. The lid is held on by six black latches that prevent the lid from coming off while navigating the course. The tinted look of the polycarbonate gives the spectator a glimpse of the magic inside.

The Hokuyo laser mount allows(I'm going to talk more about how we calculated the length of the plate and the angle vision we got from it.)

The air cooling system was designed utilizing two fans that push twelve cubic feet of air per minute through the vehicle. Exchanging the six cubic feet of air in the vehicle every thirty seconds, the electronics get needed ventilation. The two fans are placed on opposite corners of the vehicle to ensure the whole vehicle gets cooled.

4 Electrical

The electrical system in this vehicle was meant to be as strait forward as possible, predominantly using off-the-shelf parts to minimize cost and reduce human error on the part of our team. In the end, this simple philosophy proved to be very effective in making a platform which works consistently and effectively.

4.1 Power System

The entire platform runs on two marine batteries which are easily interchangeable with any other marine or car battery. These batteries are hooked up in parallel with an on-board charger with an accompanying built-in standard AC outlet, making it very easy to charge the vehicle or otherwise work on it while it is not moving without draining the batteries. A 24V and a 12V line can be accessed from the batteries for all components of the robot. The motors run off of the 24V line, while the sensors and computer run off of the 12V line. The computer is hooked up through an inverter which allowed so that a standard computer power supply could be used.

4.1.1 Safety

Safety is a primary concern of the team, and thus precautions are in place in the event of system failure. There are two main methods of disabling the vehicle: wireless ESTOP and hard ESTOP. The wireless ESTOP can be activated by the RC controller, which bypasses the computer system and forces the robot to stop all actuation. The hard ESTOP is activated by a button about 4 feet from the base of the vehicle. This function completely disables the robot's motion, requiring both a reset of the ESTOP button and for the motor

system to be power cycled. Additionally, the system is tied actively to the ESTOP button in such a way that if the line connecting the button were disabled, the system would go into ESTOP mode requiring a power cycle and reconnection of the button.

4.2 Electronics

4.2.1 Computer

AUTOMATON is equipped with a Quad-Core Hyper-threaded Intel Sandybridge i7 processor. The computer also has 8 GB of DDR3 RAM. In addition, it has support for connecting to devices which use 1394 FireWire, USB 2.0 and 3.0. RS232 communication is achieved using USB to Serial connectors external to the computer. It has a 300 Watt power supply in order to give it significant room for the machine's 95W processor.

4.2.2 Motor Controllers & Remote Control

Two Roboteq AX3500 motor controllers are employed to actuate the robot's motors. Each controller can output 60A per channel, and there is one motor hooked up to each channel. This allows the robot to have the capability to actuate each motor independently of the others. Commands to the controller are sent over RS232 serial. In order to be able to use both an RC controller and the computer to control the motors, a custom system was implemented which uses a microcontroller to read and interpret RC signals and also switch between remote and computer control dynamically. This system works well because it does not require the computer itself to be on in order to drive the robot remotely.

4.2.3 Sensors

AUTOMATON takes advantage of several different sensors in order to understand its own kinematics and also its surroundings. The robot employs a 3-axis digital compass, the Coral AHRS, along with 4 quadrature encoders (one for each wheel), and a differential GPS to interpret its motion. The compass makes use of gyroscopes and accelerometers in order to correct errors. The encoders, which run at 180 ticks per revolution before the gear box allow for sub-millimeter accuracy in the rotation of each of the individual wheels, which gives us good readings on speed and corrections for position. The Differential GPS on board gives sub-meter accuracy, typically within 2 to 8 decimeters.

Other sensors have been employed for detecting out surroundings: a Sony DCRHC1000 3 CCD Digital Camera and a Hokuyo Laser Rangefinder. The camera outputs full 720x480 resolution over FireWire directly to our computer. The LIDAR can scan objects up to 30 meters away, sweeping 270Â°, which grants a great visible range with which to work in.

5 Software

Over time, an enormous base of software for all of the robotic vehicles made at the Robotics Club at UCF has been built-up in a completely open-source online repository called Zebulon. This repository contains code for everything from visualizations to sensor input to embedded system interoperability. It has been extended with new code to implement a brand-new architecture on AUTOMATON that takes full advantage of JAUS in a manner which is clean and consistent with the standard. All code is commented in a manner which is compliant with Doxygen, which makes it clear and understandable for new and old team members alike.

5.1 Structure

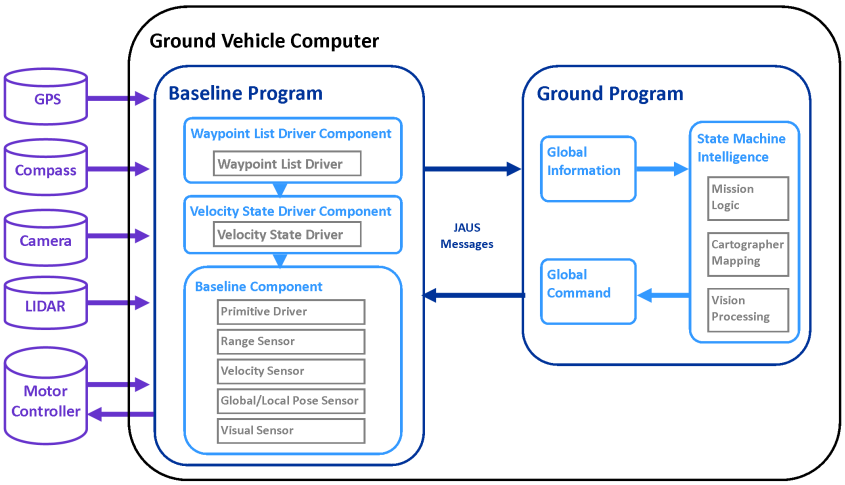

Figure 2: AUTOMATON Software structure

JAUS compatibility is one of the main goals for AUTOMATON. The JAUS standard itself is organized into several abstract units including, Subsystems, Components and Services. This existing architecture heavily influenced our final software design. The team also built on previous years experience and lessons learned for organizing software, and improved on those solutions.

JAUS (pronounced "jaws") is an emerging technology standard chartered by the Joint Robotics Program of the United States Department of Defense, currently developed by the Society of Automotive Engineers (SAE), and is designed to serve as the common interoperable framework which simplifies development for a broad spectrum of Unmanned Vehicles in military and civilian applications.

AUTOMATON runs the Baseline Program shown in Figure 1, which connects to all hardware and sensors. This program is composed of several JAUS Components which abstract hardware communication and translates all data in to JAUS messages. This program is standalone, and allows for any JAUS compatible program to then subscribe to data streams or request control, which is exactly what the second Ground Program does. This Ground Program contains all of AUTOMATON's intelligence, including vision processing, mapping and obstacle avoidance. These two programs work together in a master/slave relationship, the Ground Program analyzes data from Baseline, makes decisions and produces commands that are sent back to Baseline to execute.

The programs are currently using version 2.0 of the open source JAUS++ library available on sourceforge.net to provide JAUS compatibility.

5.2 Baseline Program

AUTOMATON contains many hardware devices that survey and communicate vital information about the outside world for our AI to process. Connecting to, taking in, and organizing all this data is the first step in creating autonomous robot software. This is one function of the Baseline Program. The second function is to create an interface to interact with the robot. To do this several JAUS services were added to allow direct thrust, velocity, and waypoint control. All of this functionality is wrapped up into the Baseline

Program.

5.2.1 Baseline Component

Each hardware device transmits data in different formats, speeds, sizes and in different time intervals. To manage these challenges, the Baseline Program uses an open-source project called Zebulon which is developed by and for UCF Robotics teams, and has been a staple in all UCF Robotics Club autonomous platforms. This project has been improved and upgraded over years and contains several libraries designed to connect to hardware devices commonly used in our robotic vehicles.

Once communication is established to all hardware devices, the data is taken and organized according to JAUS Service definitions and messages. For instance, GPS and Compass data is handled by the JAUS Global Pose Sensor Service, while the JAUS Primitive Driver Service is responsible for communicating with and controlling the motor controllers.

The Baseline Component resides within the Baseline Program and provides all these vital JAUS Services: Primitive Driver, Visual Sensor, Range Finder, Global/Local Pose Sensor, and Velocity State Sensor services.

5.2.2 Velocity State Driver Component

With low level motor control established through the Primitive Driver in the Baseline Component, more advanced vehicle drive capabilities were added with the Velocity State Driver Component. While the Primitive Driver allows control based on a percentage of thrust, from -100 to 100 percent, there is no speed or heading management.

The Velocity State Driver subscribes to data from Velocity State Sensor and Global Pose Sensor, and uses the information to maintain speed and heading. It accomplishes this task by way of Proportional Integral Derivative (PID) controllers, which continually measure error rates between actual and desired position/angle and calculate the appropriate response to correct the vehicles course.

5.2.3 Waypoint List Driver Component

The third component that makes up the Baseline Program handles high level waypoint navigation. The JAUS Challenge in this year's competition requires vehicles to drive to GPS coordinates through JAUS Set Waypoint Messages. This component was developed in order to meet that challenge. It is important to note that our AI program does not interface with this component, as there is no path planning or obstacle avoidance build into the Waypoint List Driver. It is strictly used to meet specifications provided by the JAUS Challenge.

Using information from the Global Pose Sensor, it calculates distances and heading to desired waypoints and relays that information to the Velocity State Driver Component to drive the vehicle. Waypoint List Driver continually recalculates distance and angle to target until it is reached, and as new Waypoint Messages are received they will be placed in a queued to be added as the next destination.

5.3 Ground Program

Having a robust and stable software platform is part of the challenge to creating a competitive autonomous vehicle. AUTOMOTON accomplishes this with its Baseline Program, but of course, full autonomy is the final goal. Through JAUS services in Baseline, there is a clean and reliable interface for receiving information about the environment as well

as controlling the robotic platform. The Ground Program uses this interface and adds intelligence by way of path planning, vision processing, and state machine logic.

5.4 Mapping and Obstacle Avoidance

All of the maps are built up in a framework called Cartographer. This framework is very flexible, allowing the system to build maps in whatever way is chosen to be best for the situation, and determine paths as well, abstracting away the underlying technicalities of a given type of map or navigation routine. Because of this, our team was able to prototype many different algorithms and determine what works best for a given situation. The map used is a 2 dimensional, top down view of the world. Information is stored in a Quadtree structure to take advantage of spacial locality of objects in order to minimize our search space for a given algorithm.

5.4.1 Autonomous Challenge

For the autonomous challenge, there are two types of objects which are added to the map. The first are points, generated from the LIDAR, which represent elements of a point cloud in our environment. These points come together to form the barrels and other obstacles. The other type is the lines, which are treated like walls in the map's representation. The computer vision system projects the lane lines it finds into the ground coordinate space, and then that information is added to the map as a line segment (or wall segment). All obstacle avoidance is then carried out to navigate around these dots and lines.

In order to navigate, a beam search algorithm was chosen. This type of algorithm attempts to find a set of connected beams which will reach the desired goal without hitting any obstacles. A depth of 3 beams which are considered to be slightly wider than the vehicle was chosen to give tolerance to the size of the vehicle and also because it gave a sufficient search space for the robot to find a path in most instances. The robot itself is given a heading from this algorithm and it conforms to this path with a PID controller.

Higher and lower speeds can be tolerated dynamically by adjusting the beam length and depth. For instance, if the robot travels at a faster speed, the length of the beams can be extended and the maximum angle which the beams are allowed to bend is reduced so that the robot can anticipate motion in advance and does not need to make as drastic of turns.

5.4.2 Navigation Challenge

For the navigation challenge a similar system to the autonomous challenge is employed, with the exception being that there are no lines added to the map. Additionally, maps maintain a longer persistence in this challenge, due to the higher tendency of traps which could be encountered repeatedly. Although this creates some problems with map data conflicting when something has been viewed at different times, showing up in different locations, these anomalies eventually clear up and the robot will be able to reassess its situation. Navigation is done using a best-first traversal algorithm, where goal points are set in a list given before run time. This algorithm was chosen because it is one of the fastest approaches to search. However, unlike Dijkstra's algorithm and A*, which take O(n) amount of memory, best-first uses memory O(n2). The system can tolerate this memory usage because it has 8 GB of on-board RAM, and therefore memory was traded in favor of speed.

5.5 Vision Processing

As humans, we take for granted how much information we gather and process through our eyes. We quickly and effortlessly classify countless objects, avoid hazards, navigate spaces, and coordinate with other dynamic objects all the time. What is a simple every day process for a human remains an immensely difficult problem for computers and AI. While difficult, the team felt that onboard vision processing was a necessary and critical part of creating a vehicle capable of navigating autonomously through this year's challenges.

Vision would be best applied to the problem identifying and locating boundaries lines of the course and providing supplemental cues to the AI about potential obstacles or objects of interest. The IGVC course boundaries are composed of white painted lines on grass that may be faded or non contiguous. The course also contains several colored obstacles which may contain white line segments, so classifying and filtering out these objects from the image is an important part of finding the lane line boundaries.

Our goal for AUTOMATON's vision system is to find, classify, and estimate distances and angles of lane boundary lines, as well as recognize and filter out potential obstacles and objects in its path. This is accomplished with help from the OpenCV library; used for its powerful and well documented computer vision tools. The problem is broken up into two logical steps, obstacle filtering and lane line detection.

5.5.1 Obstacle Filtering

The final IGVC course contains several types of objects that could be physical barriers or technical barriers to recognizing and staying within boundary lines. Using images captured from past years IGVC events, as well as this year's Washington D.C. event, the software team was able to do offline testing of actual conditions and create a system robust enough to recognize and filter out unwanted objects.

Color-space Conversion Images that come straight from the camera are in a red green blue format(RGB), but several other formats exist, including those that separate hue, saturation and lightness, which can be more reliable in computer vision applications. This allows easier color classification, since lightness and saturation channels can be ignored in favor of just hue.

Color Classification Rather than explicitly classify certain features in the images as obstacles, AUTOMOTON simply removes them from the image as to reduce interference with lane line finding. This is achieved through use of color classification, a method that, while primitive, has proven itself in past competitions for its speed and simplicity. Many objects in the IGVC challenge are made of distinct bright solid colors. The algorithm searches for these specific colors which must be sampled from example images; pixels falling within a certain threshold are activated and together with blob finding, a binary mask is created of areas to ignore. Some objects can be completely detected this way; others remain a bit more challenging.

Blob Finding Once colors of interest have been identified, "blobs" of colored areas can be compared; blobs of a certain size will be removed from the image, while others will be left alone. More complex objects can contain several small areas of color and white, and blob finding may detect a single object as several small objects. By taking advantage of regularities in the patterns of obstacles, AUTOMATON can confidently merge blobs after considerations to proximity, size, shape and angle. While this method is not a robust

solution for objects the real world, it is adequate for the controlled environment of the IGVC course.

5.5.2 Lane Line Detection

The most important function of AUTOMATON's vision processing is the ability to estimate position relative to discovered lane lines. Through a combination of filters, perspective correction, edge detection and line fitting does AUTOMATON accomplish this goal. The software team researched existing approaches to solving this problem, including solutions from previous IGVC events; ultimately a combination of methods proposed from several sources in years past were used: Princton's PAVE algorithm, Detroit Mercy's uCeratatops, and UCF's own Calculon.

Filtering Similar to color classification in object removal, a white filter is applied to the image after obstacles have been removed in order to identify bright areas in the image. Usually there are many areas in an image that can appear white or bright, particularly on sunny days, so more filtering steps must be taken. The image is passed through a Gaussian blur to smooth and a threshold to set dark areas to zero.

Edge Detection Edge detection is used to outline any contrasting areas of the image. If run on an image of a complex scene, thousands of edges could be found, if run on a fairly sparse image though, a much more reasonable picture emerges. Images being processed at this stage though are usually fairly free of any complex shapes and hard edges due to object removal and filtering stages. This makes edge detection ideal for identifying the last remaining features in the image: the lane lines. A Canny edge detector is used to process images at this stage which will primarily outline edges of white lines on green grass.

The process is not perfect though, brown patches in grass or other unanticipated objects can be picked up. So once edge detection is run blob detection is used to remove any edges smaller than some predefined level; this removes much of the random noise while maintaining line edges.

Line Fitting The image at this point is essentially black with white pixels outlining various objects that stand out against the background. They must now be fitted to a computable line. There are several common solutions to this problem, the software team chose to use the RANdom SAmple Consensus (RANSAC) algorithm for line fitting. Tests against popular Hough line transforms showed better consistency in fitting lines, and had the added benefit of returning a single best fit line rather than many lines which needed additional processing. RANSAC also deals well with gaps in lines, which can happen often through either course design or data loss from overzealous filters.

Since RANSAC only returns a single best fit line, the algorithm has to be run twice to detect both left and right lane lines. If on the first run a line is found, it is simply masked with black and RANSAC is run again to find a second line, if on exists.

Perspective Correction AUTOMATON's single camera is mounted about 6 feet above ground and aimed at a downward angle to see the ground directly in front of the vehicle. In a 2D image captured from the camera, objects and lines close to the vehicle appear large or far apart, while objects and lines further from the vehicle appear small or close together. This is the illusion of perspective, and because of this illusion an accurate mapping of

line angles, lengths and locations cannot be done without some transformation of the 2D image.

Correcting perspective requires an offline calibration phase in which a rectangular object of known size is placed on the ground plane in front of the vehicle. The corners are picked out, and a matrix is generated that transform relative pixel coordinates of the corners to the actual known relative corner positions. This matrix can then be applied to the entire image to correct each pixel for perspective distortion. This has the primary benefit of correcting angles that would be incorrect due to perspective, such as the illusion of parallel lines converging in the distance.

The secondary benefit of this correction is that pixels can also be mapped to an arbitrary unit of distance; in AUTOMATON's case each pixel is mapped to 1 centimeter. This mapping only applies to features on the ground(such as lane lines), and can fairly accurate depending on the quality of image and calibration.

5.6 Innovative Features

Writing software is an incremental process, each stage hopefully builds on previous work and research. The goal with AUTOMATON was not to reinvent the wheel, but to leverage the experience and expertise gained from previous competitions and add improvements where necessary.

5.6.1 Real-Time Configuration

Most important variables that tweak AUTOMATON's behavior have been added to XML files which are easily human readable and modifiable. These files contain information about the robots JAUS configuration, hardware devices, vehicle dimensions, mission, and so on. For situations that require input or training, such as vision processing or Proportional Integral Derivative (PID) controller tweaking, AUTOMATON can load changes in real-time so that effects can be observed immediately.

5.6.2 Logging and Playback

Recording data for analysis is an important part of improving performance of this vehicle. Through XML, logging of all sensors, including video and LIDAR can be turned on at any specified rate. Recording runs through courses will be valuable data for not only the current team but for future UCF teams that wish to attend this competition. Because of the flexibility and modular nature of AUTOMATON's software systems, logged data can be replayed through AUTOMATON's AI, this allows for a powerful way to understand the dynamics of AUTOMATON's decision making, as the system knows no difference between being fed logged data or real-time data.

5.6.3 Real-Time Mapping

The Cartographer mapping library has been used before in previous competitions, but not to the same extent as AUTOMATON. This year sees the addition of more efficient map storage and retrieval, improved path planning and navigation and for the first time cooperation with the vision processing subsystem. Lane lines found through vision are merged with LIDAR data to create a detailed and cohesive map of the environment around AUTOMATON.

6 Vehicle Performance & Analysis

6.1 Performance

AUTOMATON has been designed to meet or exceed the requirements of this competition now and years to come. It is rugged and robust, built to be weather resistant and terrain dominating.

Metric	Analysis
Speed	7.6 miles per hour
Battery Life	2-3 hours (approximately)
Max Tested Incline	50Â°
Reaction Time	.07 seconds (approximately)
Object Visible Range	30 meters
Trap, Dead End Compensation	Global map stored so that alternate routes can be found
Waypoint Accuracy	20-80 centimeter

6.2 Budget

Although several of these parts were inherited from previous projects at the lab, their cost has been included in order to better describe the price to duplicate the work.

Item	Unit Cost	Quantity	Total Cost
Computer	$800.00	1	$800.00
Aluminum (box and sheet)for Frame	$350.00	-	$350.00
Welding Work	$100.00	-	$100.00
Wheelchair Motors with Gearbox	$299.99	4	$1,199.96
Mini-MAX DGPS	$2,000.00	1	$2,000.00
180 PPR Quadrature Optical Encoders	$50.00	4	$200.00
Wheels, Inner tubes, Hubs	$78.04	4	$312.16
Hokuyo LIDAR	$5,000.00	1	$5,000.00
Coral AHRS Digital Compass and IMU	$1,245.00	1	$1,245.00
Sony DCRHC1000 3 CCD Digital Camera	$1,183.00	1	$1,183.00
Roboteq AX3500	$400.00	2	$800.00
Miscellaneous Electronics	$400.00	-	$400.00
Miscellaneous Mechanical	$200.00	-	$200.00
Total			$13,790.12