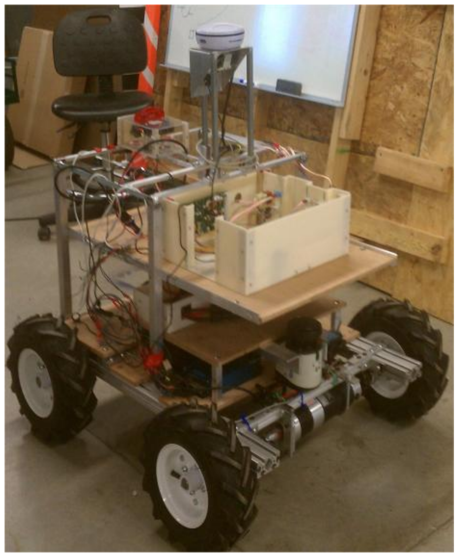
Proudly Presents: Robot Downey Jr.



Team Members:

Aaron Carteaux	Nolan Keim
Cody Feltch	Ryan Glavin
Joseph Maloney	Jade Emig
Thomas Powell	Daniel Bosenbark
Michael Shaffer	Robert Bruder
Derrike Hill	Joshua Heltzel
Troy Billups	

Faculty Advisors: Dr. Patrick Martin, Dr. Kala Meah

1. Introduction

The senior engineering class of York College of Pennsylvania is proud to present Robot Downey Jr. for the 2013 Intelligent Ground Vehicle Competition. This vehicle was designed as a capstone project by computer, electrical, and mechanical engineering students as a culmination of all the knowledge gained in classes as well as the previous team's experience. This year our team focused on improvement of last year's robot emphasizing simpler maintenance, modularity, and functionality. The team is now in its fourth year and significant improvement has been made since then as well as bugs have been worked out. Our improvements include an upgraded auto-switching umbilical power system, an actor-based code architecture implemented in Java, an improved potential fields navigation algorithm introduced last year, the team's first implementation of JAUS, a modular frame, and a simpler drivetrain.

1.1 Team Organization

Our team has two semesters to work on our capstone project. The first semester was summer of 2012 in which the team designs the robot as well as early prototyping and programming. The second semester was spring of 2013 in which the robot actually gets built and final programming is done. The team is composed of four subteams: mechanical, power, sensors, and navigation. Mechanical subteam is responsible for designing and building the frame, suspension, drivetrain, and enclosures for electrical components. Power subteam is responsible for supplying electricity to the motors, sensors, and the onboard desktop computer. Sensor team uses sensors and programming to provide the navigation subteam information about where the robot is and details about the surrounding environment. Navigation team takes the information provided by the sensors and finds a safe and efficient path to the waypoints and gives commands to the motor controller to move the robot.

1.2 Team Budget

YCP's IGVC team has been around for four years. In that time, some equipment has been carried over. Approximately $9,000 of equipment has been carried over through the past few years as seen in Table 1. The biggest expenditure this year that can be carried over in future years is the motors which cost roughly $1750. The variety of parts that make up the frame and electrical subsystems is where the team spent another $1850 and $650 respectively. In total

$4,581.02 was spent this year, $9,000.32 in carried over equipment from last year for the robot. The total cost of the robot as it sits is $13,583.34.

Component	Spent	Carried Over
Frame/Wheels	$1854.52	-
Motors	$1751.00	-
Electrical	$647.66	-
Computer Components	$217.84	$827.00
Wireless Transceiver	-	$90.00
Motor Controller	-	$675.00
AC/DC converter	-	$290.00
GPS	-	$2000.00
IMU	-	$120.00
LIDAR	-	$5000.00
Camera	$55.00	-
Total	$4581.02	$9002.32
Grand Total	$13583.34	

Table 1: Team Budget

1.3 Design Process and Integration

Summer of 2012 was spent designing the robot. Twice a week during the assigned class time the team met to discuss and collaborate on design and create design specifications. Meetings started with “stand-ups”, a 2 minute briefing of progress, if issues were brought up, further discussion was held after the “stand-ups”. This setup enabled organized and freely flowing communication to make sure all members were up to date with the design process. As part of the Capstone course, there were several presentations of progress that was presented to the professors and the rest of the team. This gives the team detailed progress updates that were not covered in the standard meetings. Sub-teams and individuals discussed heavily outside of the meetings and presentations furthering the communication between members as well as the use shared documents and emails.

1.4 **Design Specifications**

The team created specifications to guide the design process. The goal was to have the robot weigh 175lbs and achieve as close to minimum dimensions as possible. The robot's overall weight is 190lbs without payload and the actual size is 36'' x 39'' x 42''. The team calculated that the max speed should be at 1.7882 m/s, acceleration at 0.4470 m/s^2, and deceleration of 1.7882 m/s^2. We specified the low power (computer and sensors) battery life to be 5 hours, high power (motors) to have 1 hour battery life with both systems at 24 volts. We specified the navigation cycle time to be <100ms, the tested navigation cycle time is 110ms on a laptop with a third of the specifications as the robot's onboard computer. We specified to use a 4-wheeled skid-steer robot to replace the expensive tread system the team traditionally used but maintain the zero-point turn.

1.5 Design Innovations

Significant emphasis was placed on improvement of last year's robot. Some of the innovations include an upgraded auto-switching umbilical power system, improvement on the navigation algorithms, implementation of JAUS, modular frame, and simpler drivetrain.

Last year, the team created an umbilical power system to allow the onboard desktop computer and other sensors to remain on while the Lithium Polymer (Li-Po) battery charges. This allows the robot to remain on for debugging purposes. The umbilical power system required manual switching from one power source to the other via switches. The innovation and improvement over the system last year is the ability to automatically switch the power sources once the physical cord is plugged into the robot. A printed circuit board based on a custom circuit was designed for the automatic switching.

Significant time was spent debugging and improving the navigation algorithms that were implemented last year. Last year an actor-based architecture with potential fields algorithm for navigation was programmed in C# and run on a Windows 7 desktop computer. This year the code was reprogrammed into Java and run on Windows 7 desktop computer. The move to Java was for future teams to move to Linux to reduce the operating system overhead processes.

1.6 Safety

The team built in the required safety features for the IGVC competition. When the robot turns on a safety light illuminates and flashes when the robot is in autonomous mode. A physical emergency stop button is mounted near the safety light. A garage door opener transmitter and receiver make up the wireless emergency stop and meet the minimum 100 feet transmission distance. The physical and wireless emergency stops are wired to a hardware port on the motor controller which immediately cuts power to the motors, stopping the robot.

The base of the frame has two wide 80/20 pieces that the rest of the frame is based around. These two pieces offer ample safe gripping area to lift the robot should the need arise. A stand was made to lift the robot off the ground so the wheels are not in contact with the ground in testing and maintenance circumstances. All the subsystems of the robot contain fuses to protect the components from electrical problems.

2. Mechanical Designs

2.1 Introduction

The mechanical portion of this project consisted of 3 main areas: drivetrain, robot frame, and the camera tower. The main job of this subteam was to create a vehicle with specifications derived from the IGVC guidelines and other subteams. After assessing the pros and cons of last year's robot, many modifications were made to the design to improve maintenance, modularity, and design simplicity. The robot CAD drawing can be seen in figure 1.

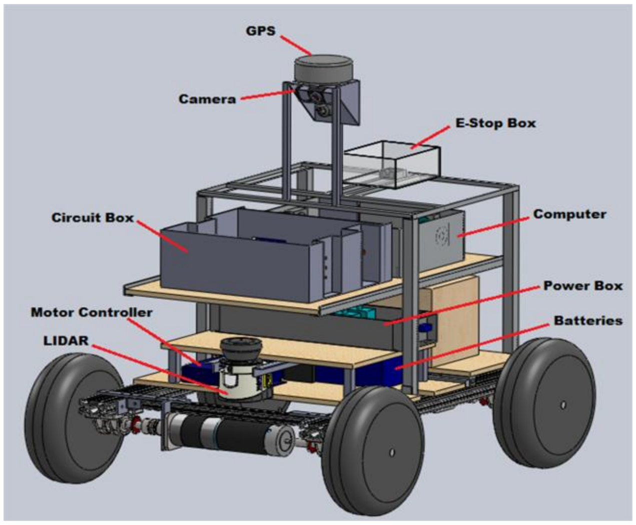


Figure 1: Robot Cad Layout of the robot 1

2.2 Frame

The frame for this year's project was designed with the intent to make the vehicle as modular as possible. One of the biggest issues last year's team faced was taking components out of the robot without disassembling a large portion of it to do so. The frame was redesigned to resemble a bookshelf, where the power supply and motor controller lay on the bottom shelf, some power components on the middle shelf, and the computer and component box lay on the top shelf as seen in figure 2. The frame is designed for easy maintenance and transport of the robot with simple sliding bolt attachments to the two 80/20 main beams. Another crucial design aspect for the frame was to keep it as small as possible within the IGVC guidelines. Doing this gives the robot the best condition to maneuver and progress on the field during the competition. It also makes the robot as light as possible so the power team could get efficient motors with ample power to move the robot. Aluminum (specifically 6061-T6 aluminum) was used to make the frame because it was lightweight, cheaper, weldable, and stronger than other common aluminum alloys. The majority of the frame was welded together because it was determined to be the most efficient way to construct the frame.

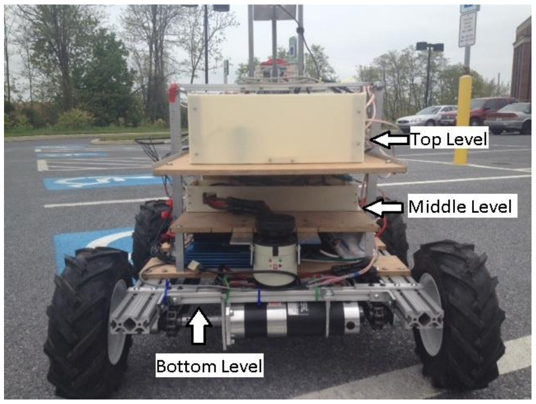


Figure 2: Robot shelving

2.3 Drivetrain

For the drivetrain the team worked on improving the design from last year. For example, the biggest change was switching to a wheel based driving system instead of a tread system. This was chosen because the IGVC competition does not have any sand or loose dirt areas in the field of competition which a tread system is ideal for. The tread based system is a very expensive system. The treads from last year's design cost over two thousand dollars, this year the wheels collectively cost three hundred dollars. The treads allowed the robot to have a zero-point turn capability.

In order to preserve the zero-point turn ability for the 2013 robot, a skid steer design was implemented. The robot's skid steer design turns the two wheels on one side of the robot at the same time. The wheels are connected using roller chain and sprockets that are mounted on each of the axles. For mounting the drivetrain on the robot, 80/20 extrude was chosen for simple mounting as well as changing any wheelbase specs the other subteams may need for navigation. The skid steer drivetrain is easy to design, assemble, adjust, and maintain.

2.4 Camera Tower and Sensors

The camera tower and sensor mounting was redesigned this year to improve the issues that last year experienced involving excessive vibration and top heaviness. This year, we lowered the height of the camera tower to reduce the camera vibration. A natural frequency analysis was performed on the camera tower to ensure the natural frequency of the camera tower would not coincide with a frequency experienced by the robot motors and frame.

All sensors on the robot were placed in enclosures so that they would remain dry in a light rain and protected from the surrounding environment. The use of enclosures emphasizes the modularity of the robot since enclosures can be placed and rearranged better than exposed and different sized components. The computer enclosure has also been rebuilt to be more modular with flush mounted I/O ports so that the computer could easily be removed when needed. The material used for the computer enclosure was also changed to ABS plastic to lighten the weight of the robot and to decrease the chance of a static discharge causing damage to the computer.

3. Electrical Designs

3.1 Introduction

The electrical subsystem is divided into two parts: the high power and the low power. The high power system is aptly named for powering the motors which requires relatively large amounts of current that flows in the motors. The low power system is named for powering the more sensitive equipment such as the onboard computer and sensors that cannot stand large power fluctuations produced during motor usage. The two power systems are powered by their own battery source both at 24Vdc with the grounds tied together as per convention and safety. When the robot is not actively participating in tests or challenges, an umbilical power system keeps the low power system going over an external power source to allow the low power battery to last longer. The umbilical is particularly useful during debugging of the code.

3.2 High Power

The high power sub-system of the robot consists of the power needed to drive the robot. This included the motors, motor controller, and the batteries that supply power to these components. All the high power components are heavy, so to keep the robot’s center of mass down, all the high power components are on the lowest “shelf” of the robot as seen in figure 2. The high power system is powered by two, 12V Sealed Lead Acid batteries in series providing 24V. To follow through with the goal of ease of maintenance, the team added a switch to take the batteries from a series configuration for running the robot to a parallel configuration to be charged at the same time by a charge controller. This configuration makes charging easier without needing to disconnect wires.

The team is reusing the Roboteq HDC2450 motor controller from the 2012 YCP robot. We chose to use this controller over again because it features dual forward and reverse 150A channels with USB and encoder inputs. The dual channel motor controller allows both motors to be operated in one device, simplifying the design over using two separate controllers. The controller has a flyback converter inside of it which regulates and observes the voltage and current supplied to the motors, useful for monitoring purposes.

Our team decided to get new motors because the motors used last year were outdated and technical specifications were unknown. Last year’s team had difficulty with the motor encoders they mounted on the motors. The team decided on brushed, inline, planetary gear motors from Midwest Motion with the option of professionally mounted encoders. Midwest Motion offered a variety of motors with varying RPM, torque, voltage, and power. The motor chosen from Midwest Motion provides a final output speed of 138 RPM, 233 in-lbs of continuous torque, 24 Vdc, continuous power of 394 Watts, and 1024 points per revolution on the encoders.

3.3 Low Power System and Umbilical Power System

The low power system includes the onboard computer, LIDAR, GPS, cameras, and anything not powered on the high power system with the motors. The umbilical power system was developed by YCP’s 2012 team for keeping the low power system operational when the robot is not being tested and keeping the onboard computer online. The umbilical power system removes all the low power components from being powered by the onboard Li-Po batteries to an external power source while simultaneously allowing the Li-Po batteries to be charged. Originally, the switching from one power source to another required manual intervention via two switches. The innovation this year is the automatic switching from the batteries to the external power once the umbilical is plugged in. The benefit of the auto-switching circuit was to make the switching safer and more intuitive. A custom printed circuit board was made for the new circuit as seen in figure 3.

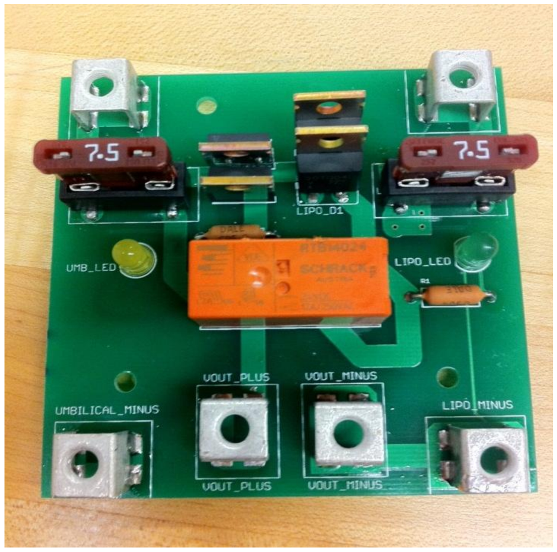


Figure 3: Auto-switching Umbilical Circuit

A high-level, low power distribution circuit can be found in figure 4. This circuit was the implementation used by the 2012 team. Since the Li-Po battery outputs a voltage of 25.9VDC over the umbilical's 24 VDC, the DC/DC converter will continue to draw from it even when the umbilical was connected. To overcome this, a set of simple switches was added in before the two diodes. Figure 5 shows the final version of the auto-switching circuit as well as where it connects to the high-level low power distribution circuit from figure 4. The original circuit design featured MOSFETs to do the automatic switching. The final version utilizes a relay to handle the power switching.

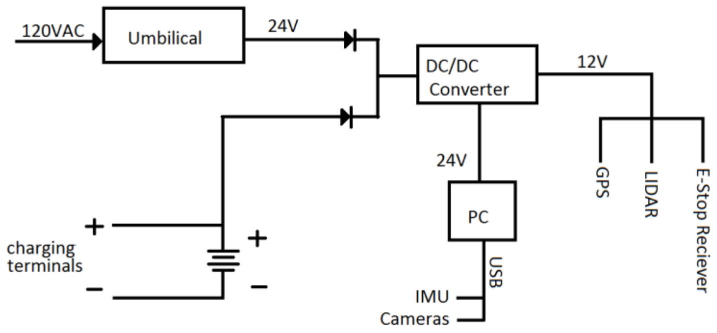


Figure 4: High Level Schematic for 2012

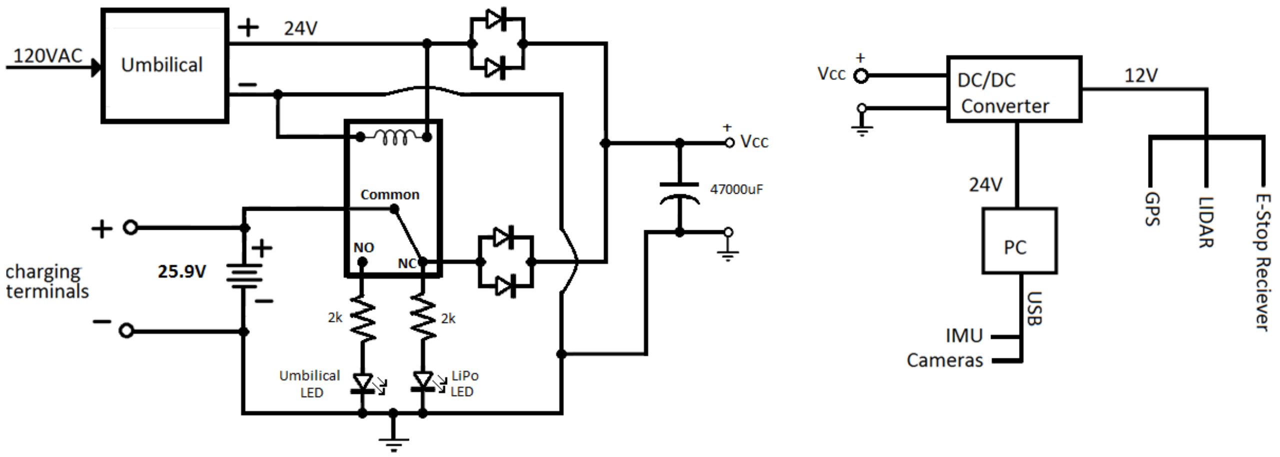


Figure 5: Umbilical circuit and high level schematic

The circuit status can be broken down into one of 4 cases:

1. Supply power from the Li-Po battery
2. Supply power from the Umbilical
3. Switch power from the Li-Po to Umbilical
4. Switch power from the Umbilical to Li-Po

In case 1, the umbilical is not connected and the relay contact rests on the normally connected (NC) terminal, powering the Green LED. In case 2, the relay contact rests on the normally open (NO) terminal, disconnecting the Li-Po from the DC/DC converter and powering the yellow LED while the umbilical to supplies the DC/DC converter. During case 3, the umbilical is connected to the circuit creating 24V across the coil, causing the switch inside the relay to move from the NC terminal to the NO terminal, thus transitioning to case 2. In case 4 the umbilical is removed from the circuit, de-powering the coil and allowing the switching inside to relay to return to its NC position (returns to case 1).

4. Sensor Integration

4.1 Camera and Line Detection

The vision system is responsible for identifying the white lines and colored flags appearing throughout the course. In order to identify these objects, a camera is mounted to receive images of the robot's environment and filter the image for specific colors and shapes. The object's locations are then translated to physical distances from the robot.

The number of cameras was reduced from two on the previous year's robot to just one. Adding a new camera with 90 degree viewing angle increased the viewing area enough to eliminate the need for a 2nd camera. This reduces the processing time and simplifies the vision algorithm.

The image filtering algorithm utilizes the OpenCV library that provides many useful computer vision functions. Figure 6 shows a block diagram of how the image is processed. The key

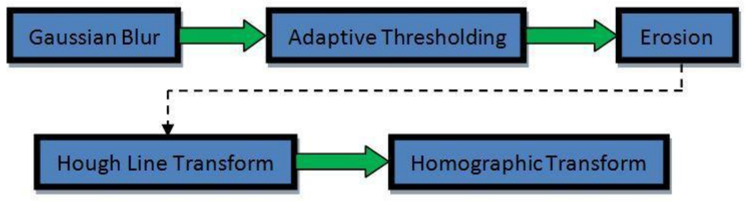


Figure 6: Image Processing Flow Chart

innovation this year for the vision system is the adaptive thresholding method provided by OpenCV. It works by gathering statistical data about the image and uses the data to determine the best thresholds for detecting white lines. All pixels below this level are considered non-white lines and are ignored. Other methods are then applied to clean up the image.

Figure 7 shows the original image featuring a line is on the left side and the resulting image from the algorithm on the right. A homographic transform takes pixel locations using a matrix that maps the images pixels to distances from the robot. These are then passed to the navigation team to allow the robot to avoid running over the white lines.

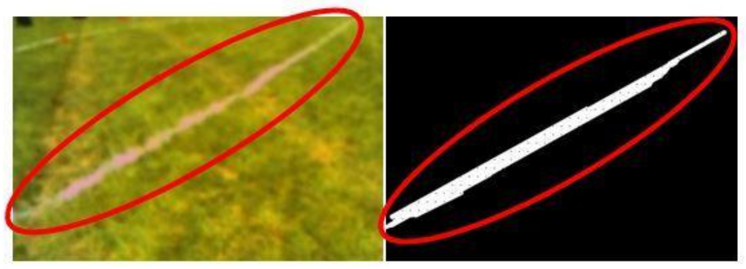


Figure 7: Original picture and line detection

4.2 Kalman Filter

One goal of the 2013 York College team was to improve sensor fusion for pose estimation. Similarly to last year's team, the robot is outfitted with three types of sensors that provide pose information about the robot: motor encoders, IMU, and GPS. The 2012 team used the encoder information to provide position and a weighted averaging of the compass and gyroscope of the IMU to provide robot heading. The team's innovation in sensor fusion is the use of an Extended Kalman Filter (EKF) to integrate the sensor information to providing a more accurate pose estimation to meet the design goal. Sensor information goes to the EKF and the resulting pose information is sent to the world scene graph (WSG) for use in the navigation algorithm.

The EKF uses a series of prediction and update procedures to calculate the robot position and heading. The EKF predicts the next state based off control information, and then updates the states based on the data measured from each sensor when it is ready to be sampled. The variation of the EKF the team used treats the encoder data as control information instead of sensor information. This eliminates the accumulation of encoder data error that is introduced by wheel slippage throughout the course, by the update procedure from the other sensors.

After testing, it was determined that the accelerometer data was being aliased due to the low sampling rate, causing large errors when integrating the data to position information. Due to the aliasing and the large vibration when turning the accelerometer data was chosen to be ignored. Additionally, testing showed the GPS data resulted in a large variance when not time averaging the data. However, time averaging GPS data while moving resulted in skewed data, causing the decision of excluding GPS from pose estimation. The resulting filter uses a combination of encoder, compass, and gyroscope data with the EKF to provide accurate local pose. Testing shows the pose estimation with the EKF is consistently within 5 inches of the actual distance travelled.

Another task of the EKF is the acquisition of data required for conversion of GPS waypoints to local waypoints. Prior to autonomous mode the EKF will collect GPS data and compass data while the robot is stationary. The EKF will then transmit the initial GPS coordinates and initial compass heading from North to the WSG. The WSG will then use the haversine formula to find a distance and angle from the initial coordinate to a GPS waypoint on the local coordinate system. The initial compass angle is then used to rotate that waypoint on the local coordinate system based on the robots current heading.

5. Autonomous Navigation

5.1 Introduction and Motivation

Emulating the same code architecture as last year's team, the 2013 team has refined an actor-oriented architecture written in the Java programming language as it moved from last year's C# code. A motivation to use Java was to use the Simbad testing environment which was very useful in debugging the algorithm from last year. Figure 8 shows the robot running in the Simbad simulator. This architecture is an abstraction that makes parallel programming easier: the actor is the fundamental unit of parallel computation, allowing all actors to perform their tasks concurrently, and communication between actors occurs by passing messages that contain the data actors need to complete their tasks. The figure 9 is a flow chart of the actor architecture communication process.

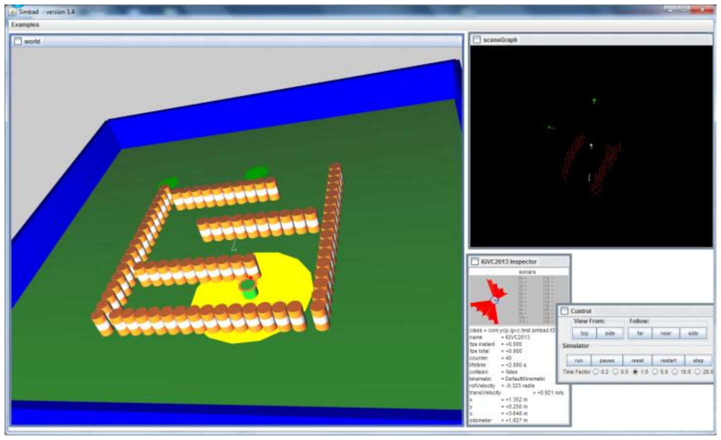


Figure 8: Robot running in the Simbad simulator

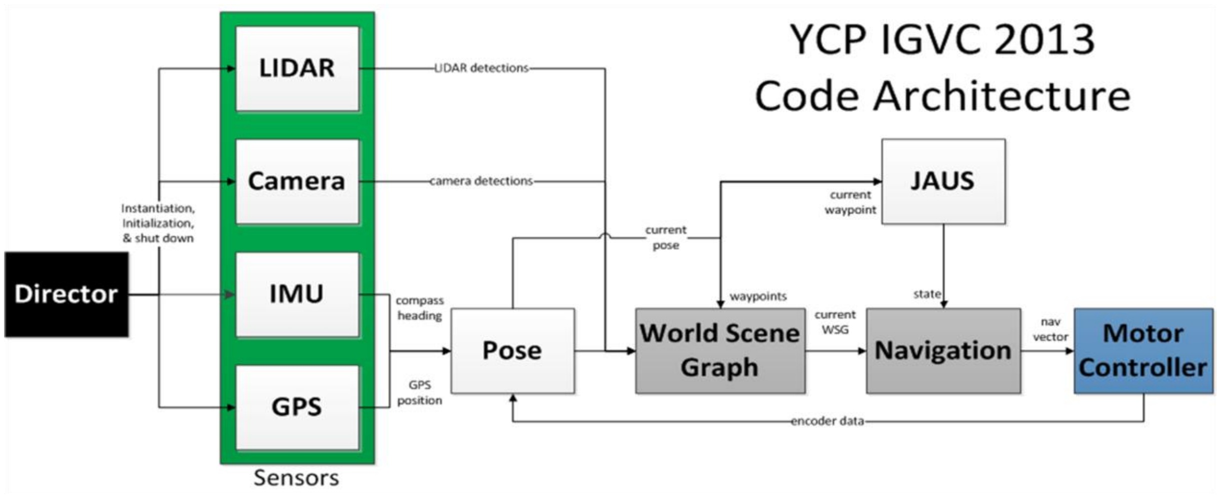


Figure 9: Actor based Architecture Flow chart

5.2 Computation Hardware

The computer hardware starts at the 24VDC low power system. A DCX6 DC-DC convertor reduces and regulates this input voltage down to +12VDC for the rest of the coputer hardware to use. At the heart of the computer is a 3.5GHz quad core Intel i7-3770K CPU. The motherboard is a ASUS Maximus V GENE with 8 USB ports on the back panel for the sensors. There is two 4GB of DDR3-1866 Corsair Vengeance RAM plugged into the motherboard. The Computer will run Windows 7 for the operation System.

5.3 Navigation Algorithm

The previous year's code used a Potential Fields Algorithm (PFA) for the heart of its navigation. The PFA treats all detections (obstacles and lines) as similarly charged particles and the goals (GPS and local waypoints) as oppositely charged particles. When the robot starts to sense an obstacle, because they are treated as similarly charged particles, a repulsion vector occurs between them. This gradually grows stronger, as the robot gets closer, forcing the robot gently around the objects. Conversely a goal, being of opposite charge, creates an attraction vector pulling the robot towards it. Graphically the vectors can be seen in figure 10.

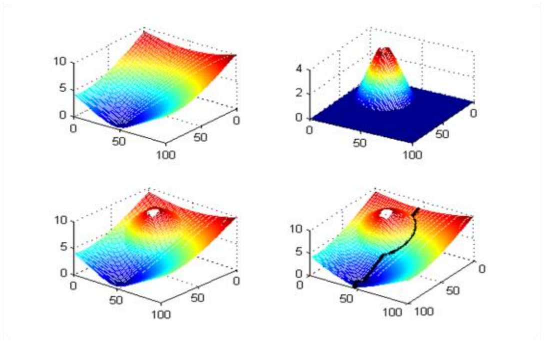


Figure 10: Potential Fields Graphical example

This year's team improved the navigation process by implementing a path planning algorithm (PPA) that would set up minor goals to help the PFA to choose a path through the course. The path planning also takes into account where it has already been to try exploring all possible paths till the goal is reached. This memory is able to exist thanks to the position estimation done by the Kalman Filter. Figure 11 is a graphical representation of the scene graph where red dots are detections, yellow dots are where it has been, the green arrow is the direction to the waypoint off the detectable area, the orange arrow is the short term goal direction.

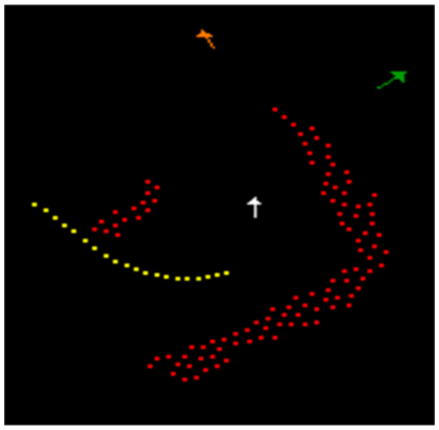


Figure 11: Graphical Scene graph

The Kalman Filter helps to accurately estimate how far and in what direction the robot has moved with respect to a global coordinate system. This allows the system to have memory by updating the robot current position relative to the coordinate system. The Filters accuracy comes from the combination of position estimates from Inertial motion unit (IMU) , motor encoders, and the GPS.

Our intelligent robot perceives the world around it using Camera and LIDAR sensors. To expedite the process of sensing data, each sensor is distributed into its own actor. Each sensor actor is able to collect the data, preprocess it, and generate a list of points. These detections are then sent to the WSG which uses occlusion to filter out repeated detections to help limit the size of the data set for the PFA and PPA to operate on.

6. Joint Architecture for Unmanned Systems (JAUS)

For the 2013 IGVC competition we have developed our first attempt at a fully competition compliant JAUS implementation. The current design goal was for a minimalistic implementation, containing only the services and functionality necessary to satisfy the 2013 competition guidelines (i.e. handle the required input messages and output the necessary responses). It was derived that the Discovery, Access Control, Management, Events, List Manager and Transport services from the JAUS Core Service Set, and the Velocity State Sensor, Local Pose Sensor, and Local Waypoint List Driver services from the JAUS Mobility Service

Set, needed to be, and were, at least partially implemented. An Operator Control Unit was developed to simulate the competition’s Common Operating Picture for testing.

The JAUS Tool Set (JTS), a set of open source development tools designed for rapid prototyping of JAUS systems and services, was used to supply and construct much of the background code for the implementations. The majority of the developed JAUS code consists of Java methods for handling protocol, instantiating/filing the output messages and sending messages to the appropriate source, as well as graphical user interfaces for test and debug. The handling on JAUS messages “on-the-wire” is done by the node manager supplied by the JTS; this manager makes use of the JAUS Router (JR) middleware, designed for managing the UDP incoming and outgoing JAUS traffic. Our JAUS implementation runs as an independent, separately compiled, application, which interfaces to our robotic framework via a simple port communication protocol.

7. Conclusion

York College of Pennsylvania would like to thank the judges for their time and consideration. We look forward to going to the competition.