ROSE-HULMAN INSTITUTE OF TECHNOLOGY

2013 IGVC DESIGN REPORT

GALATEA

ROSE-HULMAN ROBOTICS TEAM

ROSE HULMAN ROBOTICS TEAM CM 5000
5500 WABASH AVENUE
TERRE HAUTE, IN 47803

This report and the described vehicle were designed and constructed by the Rose-Hulman Robotics Team during the 2012-2013 school year and it is compliant with the Senior Design report standards.

David M. Mutchler Ph.D. (Advisor) Date

Dave Fisher Ph.D. (Co-Advisor)

I. Introduction

The RHIT Robotics Team comprises of undergraduate students studying engineering and sciences at Rose-Hulman Institute of Technology in Terre Haute, Indiana. This year, the team constructed an entirely new and original robotics platform for the 2013 Intelligent Ground Vehicle Competition (IGVC). Based on Greek mythology, the team named the new platform Galatea after the ivory statue that comes to life carved by Pygmalion of Cyrus. The goal of Galatea is to intelligently navigate through an outdoor obstacle course, avoid collisions with obstacles, stay between white lines, and travel to several designated GPS waypoints. After recognizing some major drawbacks and disadvantages from previous platforms, another goal of Galatea is to improve the team's performance in the competition. This report documents the design process and Rose-Hulman Robotics Team's rationale for this year's new robotic platform, Galatea.

Figure 1. **Rose-Hulman Robotics Team picture.** The Rose-Hulman Robotics Team is made up of two main competition teams, IGVC and MATE, that compete each yearly in international and national competitions.

II. Team Structure and Design Planning Process

2.1. Describing the team's hierarchal structure and organization

The IGVC design team is structured into two sub-teams, the hardware team and the software team. The hardware team is responsible for designing and constructing the frame of the robot that meets the requirements of the competition and software team, installing the electronic systems on the platform, and servicing the platform's hardware components. The software team is responsible for developing and implementing the software strategy, integrating the sensors of the platform, developing the mapping techniques, and creating an original yet innovative algorithm that allows the robot to accomplish the required goals for this competition. The sub-teams maintained constant communication to avoid unexpected problems that we experienced in past competitions. Both of the sub-teams met for seven hours a week with an additional hour for the all-team meeting where the sub-teams discussed updates and the status of the robot. This resulted in roughly 3,500 accumulated man hours over a period of thirty weeks. Aside from the sub-team meetings, the officers and the team advisor held a weekly one-hour meeting to plan the necessary course of action and handle any administrative actions.

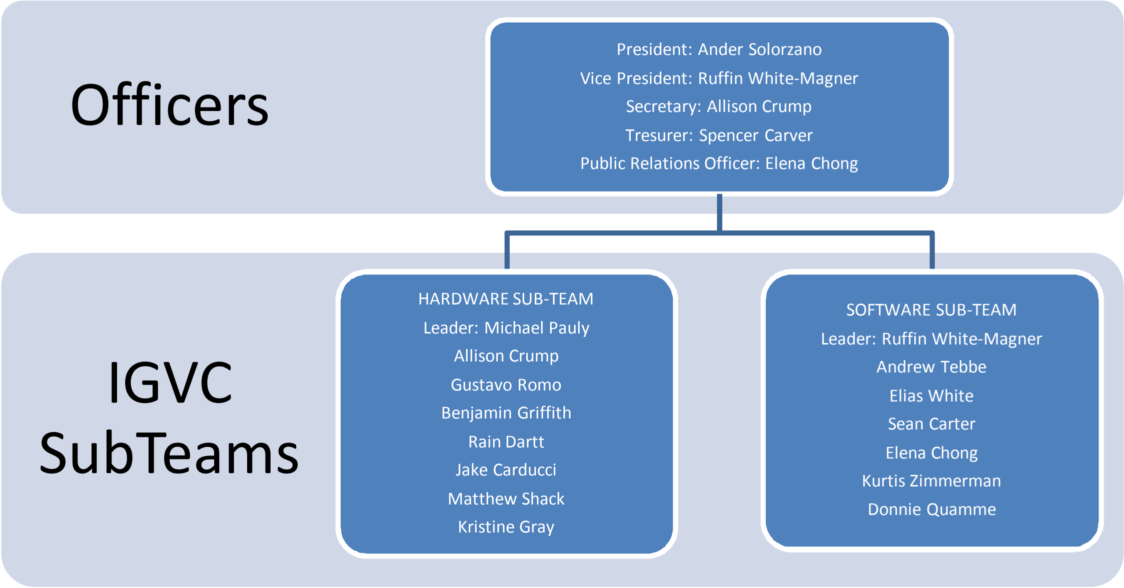


Figure 2. **IGVC Team breakdown**. The figure above demonstrates how the team is structure and organized into its two IGVC sub-teams with a team leader assigned to each sub-team and the administrative branches of the officers.

2.2. Envisioning and creating the overall design

Prior to designing and construction of Galatea, the team gathered information on the past successes and failures of previous robotic platforms. The team was then able to come up with a list of requirements and goals that Galatea must be able to accomplish. Some of the design requirements include turning easily on expected competition terrains, driving at an adequate pace, lightweight yet sturdy mechanical frame, allow easy servicing, and consisting of an effective power distribution system.

After determining the goals for our robot, the team began to determine how to best meet those requirements. To begin, numerous brainstorming sessions were held in which members each created a rough chassis design and then presented it to the rest of the team for a discussion on the benefits and drawbacks. Some of these designs were eliminated while others were elaborated upon. Eventually, the team narrowed down the options to our current design. The next step in our process was the creation of a CAD model. This model allowed the team to determine the location of key components such as the battery, payload, and key electronics. Fleshing out the design before construction allowed for careful analysis leading to a lower center of gravity, a more compact frame, and an overall better design.

III. Mechanical Systems Report

3.1. Choosing the physical design constraints

By drawing from lessons learned from both the competition and the previous robot, the team set out to design a compact, effective, and user-friendly platform capable of competing successfully in this year's competition. When designing this new robot, the hardware team strove to provide a safe, lightweight, and durable platform from which the software team could efficiently execute their tasks.

Though the option of using a commercially manufactured robot or wheelbase was offered during early brainstorming, the team elected to design and construct Galatea from scratch. Doing so afforded greater freedom in design, allowing the team to focus on three key design requirements: maneuverability, durability, and serviceability. These criteria were chosen to ensure that Galatea could perform at a competitive level while maintaining ease-of-use.

3.2. Modeling the new robotic platform based on design constraints

Taking a design from a simple sketch to a CAD model was a key phase of designing our vehicle. The team started with only a sketch and divided it into many smaller components. Members of the team were tasked design a CAD model of one or two of these different components. Once this was finished, the team compiled the files and assembled them into the 1st draft of a CAD model. The team worked on this model to improve upon it until a final decision of the design was reached.

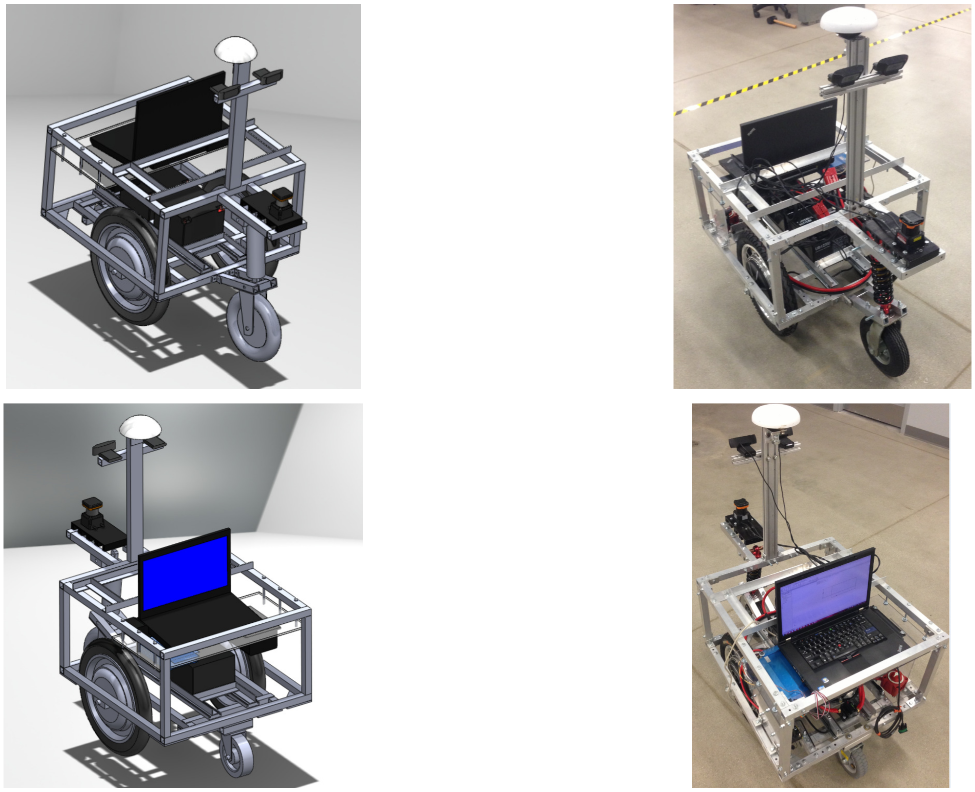


Figure 3. **Comparing the pre-construction CAD model to the constructed mechanical frame of the robot.** Both figures on the right only show the robot without the panels and covers to allow for easier comparison of the frames.

3.3. Building a chassis focused on maneuverability and increasing performance

When designing the chassis, the team favored a hollow rectangular prism shape with a rectangular protrusion on the top plane for attaching the front caster. Two materials were considered for the frame of the robot: 80/20 10 series T-slotted aluminum extrusion or plain aluminum square tubing $\frac{1}{8}$ inch thick. For one square inch of material, the 80/20 was about 0.5072 lbs. per foot while the tubing was much less at 0.3638 lbs. per foot. With heavy weight being a significant drawback of previous robot designs, the team chose the lighter square tubing over the conveniently slotted 80/20. This choice meant more work drilling holes for bolts, but the payoff was worth it. The heavier components were positioned in the middle of the bottom plane between the hub motors. This lowered the center of gravity from previous robot designs, and the hub motors helped achieve a near zero turning radius for Galatea.

3.4. Choosing new technologies for motors

The choice of motors was a driving force in the overall design. The team chose to use three-phase dual channel brushless controller with hub motors. These hub motors are very compact in design and greatly simplify mounting and installation. Rather than having to deal with the separate mounting of a wheel, motor and weight baring axel separately, the hub motors integrate a pneumatic wheel with a motor seamlessly. This was important because the design our chassis was greatly based around how the hub motors would need to be mounted. The hub motors even contain built in Hall Effect sensors to provide feedback. Despite their compact design, the hub motors had high torque, even at low voltages and low speeds. These hub motors used were originally designed for converting bicycles into electric mopeds in China, meaning they were designed for outdoor use with heavy loads. The hub motors design ensured that the robot platform could remain simple and compact, could handle rough terrain, and provide high torque for good control.

To stabilize the robot, two pneumatic castors were used. A rear castor was mounted direct to the frame for balance, while the front castor was mounted to a vertically pivoting arm. This arm was then connected to the top level of the frame by a shock absorbing spring-dampener system, intended to reduce the jolts and shaking caused by driving on uneven terrain.

3.5. Ensuring serviceability and ease-of-use

One of the primary goals for this year's design is to create a serviceable robot to minimize downtime. Easy internal access to the robot was deemed essential for quick repairs to the electronics hardware, and a laptop dock allows instant modification and monitoring of the navigation software as well as ergonomic interaction for programmer coding on the fly.

Mounted a few inches below the top surface of the robot is a thick sheet of acrylic, on which currently a Lenovo W520 laptop dock is mounted. This is another key aspect of the modularity of Galatea; any compatible Lenovo W-series laptop with the appropriate software and drivers can be placed in the dock and be used to operate the robot. A simple press of one eject button disconnects every cable from the laptop for removal. It also allows for more flexible debugging, since the laptop can be used while attached to the robot or it can be quickly removed for working elsewhere.

To allow for easy battery swapping, the batteries are mounted on a sliding track in the center of the robot. This allows for quick access to the batteries from the rear of the robot, while keeping them stored as close as possible the rest of the electronics to minimize wire clutter. Positioning the track low to the bottom of the robot also helped in keeping the robot's center of mass as low as possible to maintain stability.

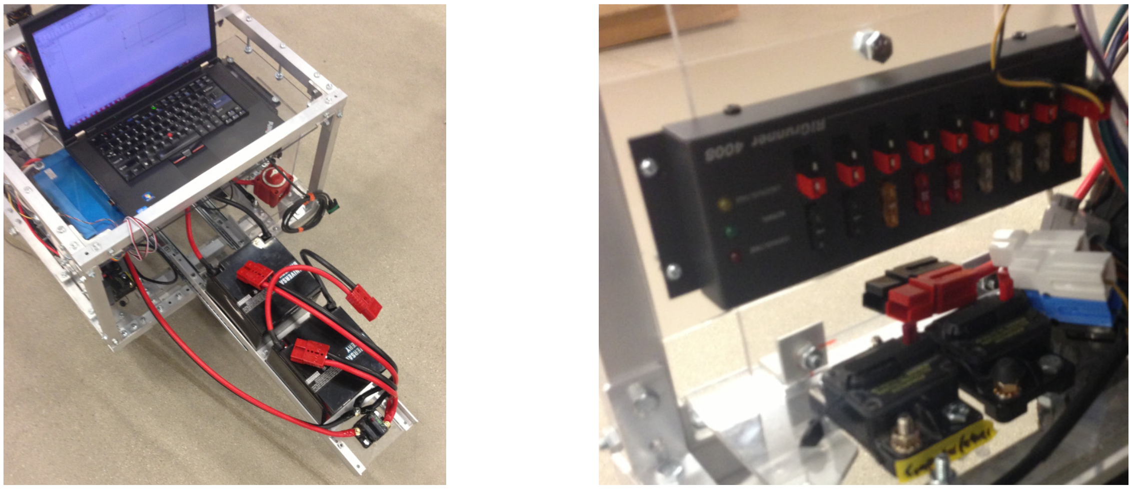


Figure 4. **Demonstrating the serviceability aspects for ease-of-use operations.** The figure on the left shows the battery compartment tray sliding out of the robot for easier serviceability. The figure on the right shows the breakers of the motor controller and the power distribution (PD) board that routes power to all the sensors.

Compressed PVC panels are attached to the top and all sides of the robot to protect Galatea from rain, snow, or other weather conditions. These are attached with Velcro so that they can be quickly removed for repairs, yet stay in place when the robot is in use. There is a hinged panel on top for quick access to the laptop. Another important aspect of the modularity of Galatea is hardware consistency. The team settled on a standard of ¼"-20 bolts for connecting parts. This ensures that only one type of wrench is needed to make any repairs.

IV. Electrical Systems Report

The electrical team made significant adjustments to last year's design to improve the overall performance, power distribution, reliability, safety, and integration of the electrical systems. The robot's electrical structure must be modular to allow the team to replace components or reposition some components with ease. The robot's electrical components must be robust, sturdy, and capable of handling off-road terrain. The components must be reliable, user-friendly, and rarely experience serious or minor malfunctions, experience low wear over time, and contain safety measures to protect it against a current spike.

4.1. Improved requirements

To increase the safety of the robot, the team decided that the robot must contain circuit protectors to prevent unexpected damage to the components. The use of 180 A, 150 A, 50 A, and 40 A snap action breakers protect major components, such as the motor controllers, the AC/DC charger, the PD board that routes power to the sensors, and the batteries, from an unexpected current spike. The power supply and switching network delivers power to the entire system while isolating the batteries from the sensitive electronics while trickle charging the solid sealed lead acid batteries.

The robot must is able to be safely shut off by pressing the on-board emergency stop (E-stop) or by wireless communication in case any unexpected problems arise. To improve the robustness of the safety mechanism, absolutely no software-dependent signal controls the E-stop switch or wireless communication. Instead of using microcontrollers and RF garage door openers like in previous designs, Galatea uses a sequential structure of hobbyist grade RF transceivers, mechanical E stops, pulse width modulated relays and high-power solid-state relays (SSR). This allows for better performance in range testing as well as hierarchy of software and hardware fail safes.

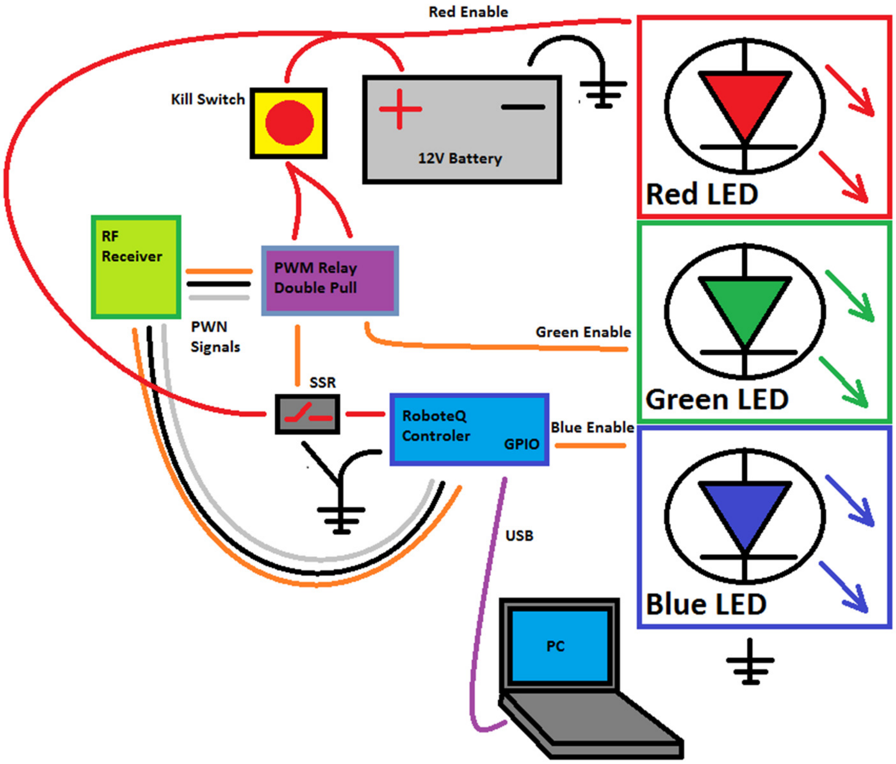


Figure 5. **Circuit diagram logic for the emergency shut-off switch states.**

Figure 5 shows a simplistic schematic of the E-stop out mechanism and status indication system. When the red LED system indicator is enabled, this means that the robot power subsystem is live and electronics are under mobile battery power. Upon power up, the RF receiver will use spectrum hopping in tandem with the intended transmitter to find a suitable open channel within the 2.4 GHz public spectrum. As the receiver receives commands from the transmission line through the relay, the PWM operated double pole and the will throw relay will remain normally open preventing a closed-circuit across the input of the solid-state relay that is used to open and close a path for current to the high-power motor controllers. Once radio communication is established and the designated channel for each stop operation is disengaged, the PWM controlled relay will change to the closed state, thereby simultaneously enabling the green LED status indicator. This means that the robots motor controllers are under power and operational using the manual override of RC transmitter. This manual override is helpful when low-level control over the drivetrain would be convenient. Such is the case where a potential software malfunction during testing prevents the vehicle from being operated through the software-user-interface. Using the remote control override, the robot can still be driven quickly back to the workshop where further software debugging can be conducted in a more convenient location. Finally, using the GPIO on the motor controller, the computer can simultaneously enable digital outputs of the controller while specifying motor velocity commands. Thus the flashing of blue LED status indicator is used as a method to determine whether the motor controllers are within the command of the onboard artificial intelligence.

4.2. Redesigning the power distribution for increased safety, serviceability, and performance

Galatea draws its power from two +12 V sealed lead acid batteries connected in parallel. The batteries are connected to a 180 A main breaker that distributes the power to all the electronics components, motors, and laptop.

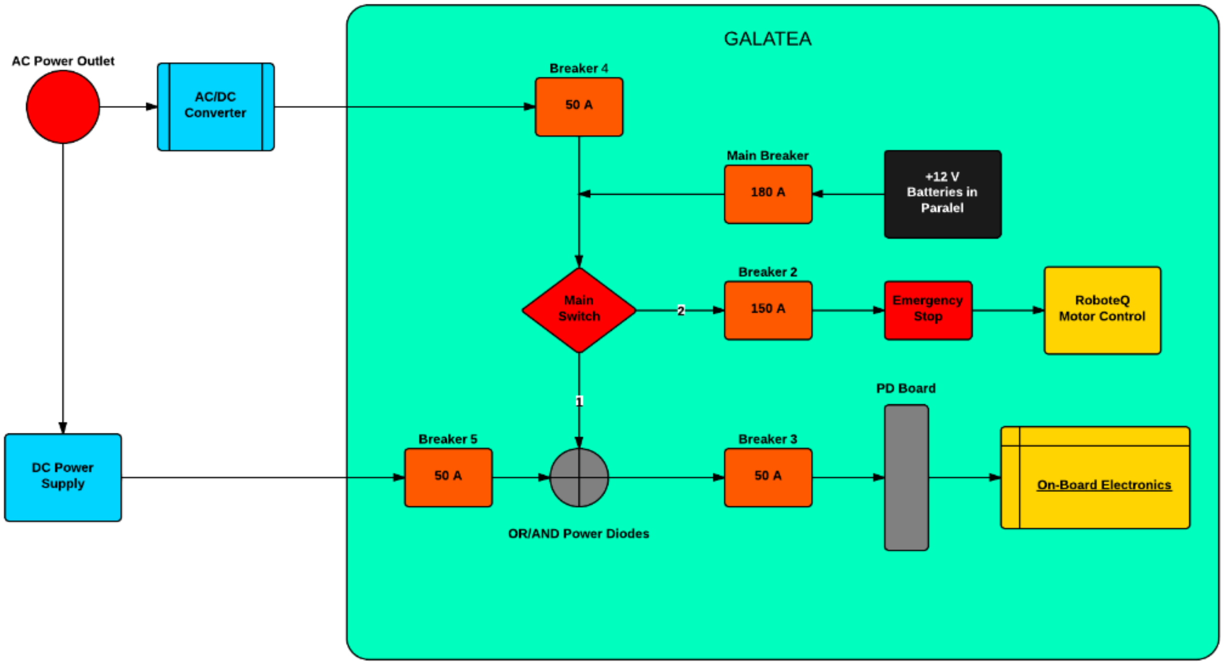
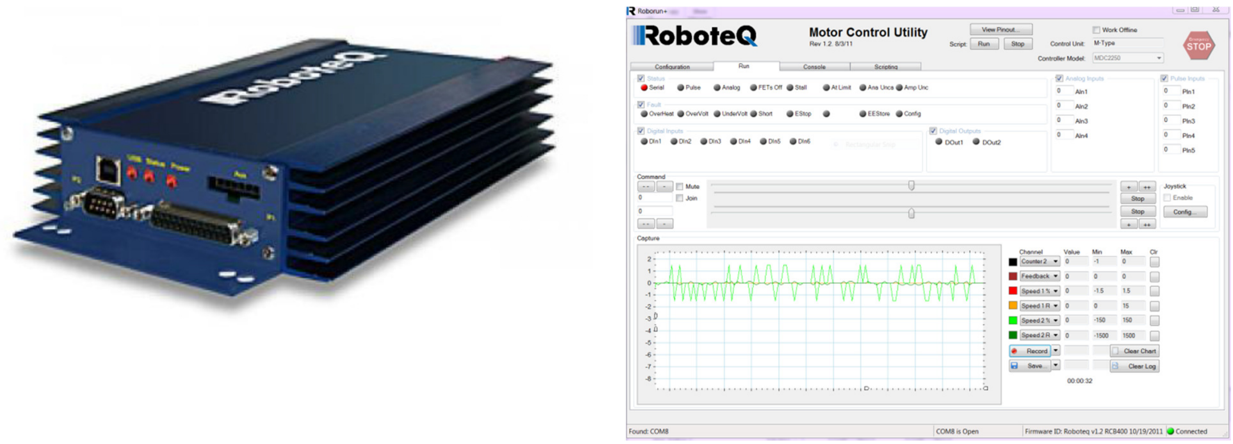


Figure 6. **Circuit diagram flowchart for Galatea.** The power distribution to the Galatea is controlled by a main switch and main breaker that diverts power to the whole system.

4.3. Setting design requirements for the electronics

The team performed several tests run on the original design to observe the performance of the robot design. Critical areas of concern such as turning ability, turning speed, power consumption, average travel speed, responsiveness to the kill switch, overall performance of electrical components, and structural stability served as the basis for our tests.

Improving the motor controller performance



□ Work Offline
Control Unit: M-Type
Controller Model: MDC2250

Status: ● Serial ● Pulse ● Analog ● FETs Off ● Stall ● At Limit ● Ana Unca ● Amp Unc
Fault: ● OverHeat ● OverVolt ● UnderVolt ● Short ● EStop ● ● EEStore ● Config
Digital Inputs: ● DIn1 ● DIn2 ● DIn3 ● DIn4 ● DIn5 ● DIn6
Digital Outputs: ● DOut1 ● DOut2
Analog Inputs: 0 AIn1, 0 AIn2, 0 AIn3, 0 AIn4
Pulse Inputs: 0 PIn1, 0 PIn2, 0 PIn3, 0 PIn4, 0 PIn5

Channel	Value	Min	Max	Clr
Counter 2	0	-1	0	□
Feedback	0	0	0	□
Speed 1 %	0	-1.5	1.5	□
Speed 1 R	0	0	15	□
Speed 2 %	0	-150	150	□
Speed 2 R	0	-1500	1500	□

00:00:32

Found: COM8 | COM8 is Open | Firmware ID: Roboteq v1.2 RCB400 10/19/2011 ● Connected

Figure 7. **The dual channel 3-phase RoboteQ motor controller and the calibration UI used for tuning the controller output and response.** Motor controller tests showing wheels changing direction due to the direction of current flow.

This year the team upgraded the motor controller from the RoboteQ Motor Controller MDC 2250 to the RoboteQ Motor Controller HBL 2350 to comply with the new brushless motors being used. This model was chosen because control of two motors was all that was needed. It also contains many advanced control options and allows for USB communication via RS232 to a computer meaning that a RS232 to USB adapter is not required making more reliable and simpler interface. For power efficiency, the motors also allow for regenerative braking which help to return power to the batteries as the motors slow by use of the Hall Effect.

The motor control also allows for interfacing with a RC controller using pulse-width modulation (PWM). The motor control has been configured in such a way that when the RC controller is connected to the motor control it overrides the USB command line and stops the robot. The wireless emergency stop is integrated into the RC controller with a switch on the controller used to override the motor controller to stop.

Improving the vision and object detection capabilities

Previous generation robots from the robotics team extended an image perception platform based on a singular camera approach. Many issues can arise when using a mono vision approach such as a limited field of view. To compensate for this drawback, stereoscopic vision was implemented to improve the field of view. Using two Logitech HD 1080p webcam that connects via USB for a coplanar stereoscopic set up, the team is practicing and implementing the use of stereoscopic perception to aid in obstacle avoidance and line detection. A precisely machined mounting set up for the cameras is in place to aid in the process of geometrical calibration as well as some flexibility when testing various configurations.

Lastly, the team also swapped the 24-volt SICK LIDAR sensor with a 12-volt Hokuyo UTM30LX LIDAR sensor. Aside from decreasing the power consumption, the new LIDAR also increased the robot's scan angle to from 180^o to 270^o with a 30 meter range. The team decided that more scan angle rather than range was crucial since it would allow the robot to detect obstacles that are 90^o left or right and on the blind spots of the robot.

Sensors to improve localization and communication

Due to issues with interfacing with the previous year's IMU the team decided to replace the IMU with a Yost Engineering 3-Space USB/RS232 IMU. This model allowed us to interface and power the device directly over a USB connection removing the potential failure point of a RS232 to USB converter. The sensor contains a magnetometer, gyroscope, and accelerometer which combine to provide accurate data about the orientation of the sensor. The accuracy of data is augmented by an internal Kalman filters which reduce noise and magnetic interference. This data is returned to the path planning program to provide heading information. An additional feature of this sensor is a RGB light as a status indicator and a user feedback indicator.

In addition, the team also installed the NAVCOM SF-2050 GPS that allows us to acquire the robot's position within a 10-cm radius as long as the robot is in an open-field and the WAAS feature is enabled. The WAAS feature allows the GPS to improve accuracy, integrity, and availability by measuring small variations due to the geometry of earth, the ionospheric conditions, and electromagnetic disturbances.

V. Software Systems Report

Using National Instruments' LabVIEW programming environment, the team developed a robotic platform implementing the basic principles of sense, think, and act. The general implementation of these principles involved performing every process of sense-think-act in parallel. Using this method proved more advantageous since many robotic applications involve small and mobile computational architecture, relatively lengthy computation times, slow sensor refresh rates, and mission-critical safety requirements. By observing the everyday behavior of living creatures, the characteristics of sense-think-act are performed in parallel instead of sequentially.

The team reached a consensus that the overall software design must contain a top-level module that instantiates shared-variables between Virtual Instrument called VI's, initializes all the sensors at once, reads and manipulates the data from the sensors, performs the necessary operator loops, and correctly closes all sensors when needed. This increased the system integration between all the electrical sensors and peripherals.

5.1. Top-level module to improve system integration

Using a top level module to encapsulate the subsequent sub-processes, we are able to structure the startup initialization, continuous operation, and shutdown procedures. Upon startup, our code first initializes all the sensor acquisition loops, which include the GPS, LIDAR, IMU, and motor telemetry acquisition loops. Every sensor loop runs independently and continuously while acquiring and writing sensor data to share global variables to improve the system's integration. This allows for different acquisition rates that are dependent on each sensor type. For instance, the GPS updates once every second, while the LIDAR data updates at 500 Hz. This enables each sensor to operate at its maximum frequency, allowing the environmental sensor database to update as periodically possible. All sensor interfaces communicate with computer platform through virtual serial RS-232 ports emulated by USB adapters. These ports are opened and configured using LabVIEW's VISA tool.

5.2. Describing the "sense" loop logic

The LIDAR continuously broadcasts polar coordinates that contain the angle magnitude of object surfaces that intersect the LIDAR's plane of sight. The GPS sensor loop acquires a longitude and latitude along with additional information such as altitude and absolute time. Using a known location approximation, the startup time for the GPS can be improved by setting a close estimate of the GPS's physical location. Using Kalman filters, the IMU reports the robot's orientation in space with respect to the Earth's geometric north pole. This is used as a magnetic North reference for the compass in order to guide our robot to a command heading.

Once the entire sense network is initialized, the top level module continues to start up the logic VI, or "think" portion of the code. This VI takes it the current sensor data along with the current set mode that defines the robot's behavior, which include a "tele-operated" mode for remote operation and drive control with external RC transmitter or any other handheld remote and an autonomous mode for intelligent course navigation.

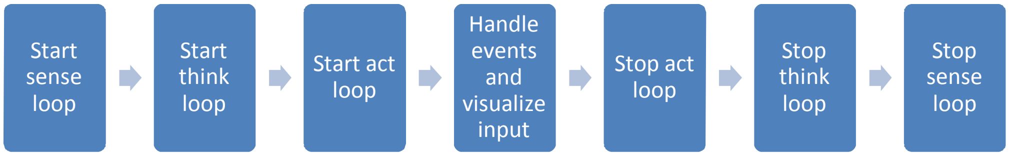


Figure 8. **Implementation of the deliberative control schema for complex tasks.** This flowchart shows the start-up and shut-down of the sense-think-act method. The order is reversed so that the motors are shut down first, followed by the computer, and the sensors. It is important that all sensors close down correctly at the same time before rerunning the code once again.

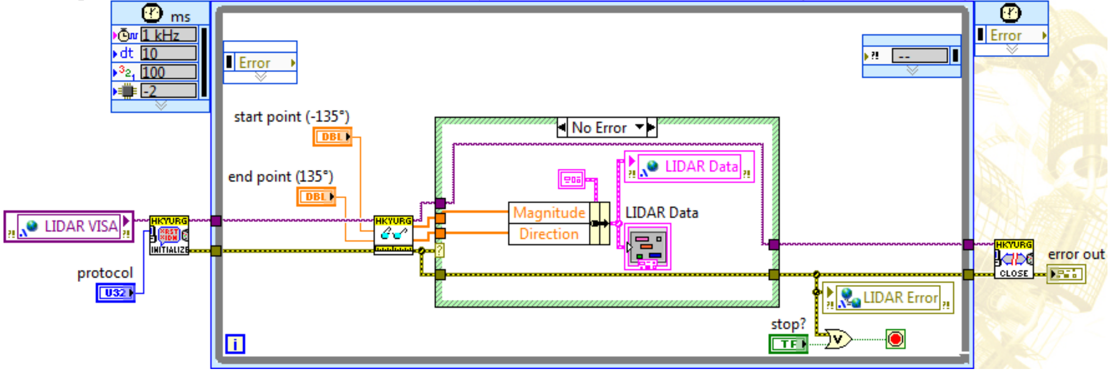


Figure 9. **Example of sensor acquisition loop.** This figure shows the LIDAR acquisition loop used for mapping out and detecting the obstacles along the path.

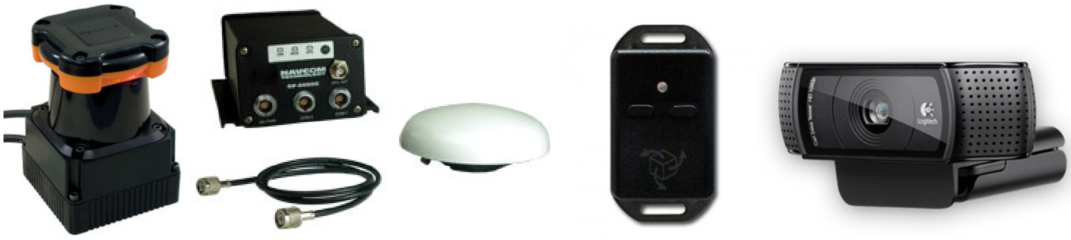


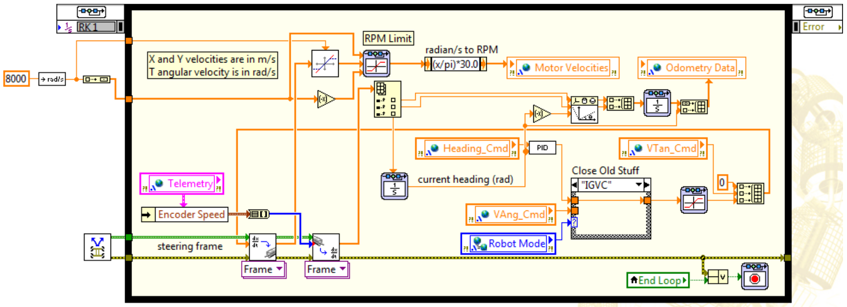
Figure 10. **The main sensors used to provide feedback and perception of the world to the robot.** The sensors are, from left to right, the Hokuyo LIDAR, the NAVCOM GPS, the IMU, and the Logitech webcam.

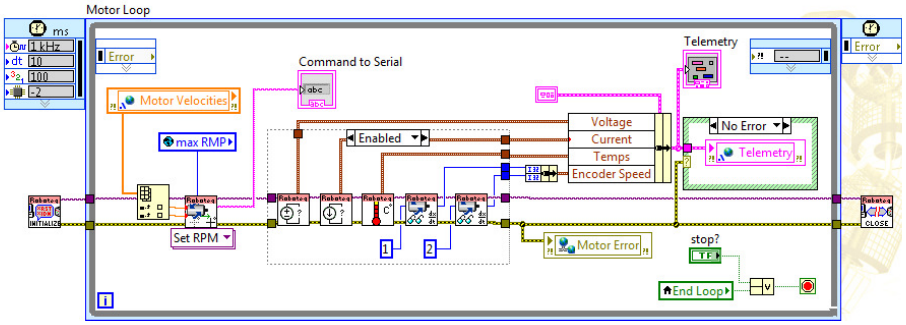
5.3. Describing the “think” loop logic and sensor integration

While the LIDAR data is operating, the vision loop captures the white lines and stores an array of several line vectors that contain start and end points. Transposing the line vectors on top of the LIDAR distance readings help create a new obstacle histogram. Using our current GPS location and the location of the next GPS waypoint, the team uses spherical coordinates to calculate the current distance and directional bearing to the waypoint. Using the obstacle histogram and desired bearing, the team constructed a simple bearing controlled algorithm that attempts to align the robot towards a suitable opening within the histogram field. An opening is determined “suitable” when the distance and angle of the opening produce an absolute opening wider than the width of the robot’s wheelbase plus tolerances. While searching for suitable openings the robot will also implement a cost function based on its current difference in sub-goal heading and current heading along with a current distance. If the robot veers too far from the sub-goal heading or the current distance exceeds the specified maximum tolerance, the robot will find alternative paths that might provide a more direct path.

5.4. Describing the “act” loop logic

Lastly, the motor control loops are initialized, thus implementing the "act" portion of the code. From the command bearing and command velocity specified by the logic module, the control loop attempts to drive the robot in the set direction. The specific motor velocities or RPM values are calculated by defining the drivetrain model matching the robot's own differential drivetrain. All relevant physical dimensions are specified within the model such as wheel radius, wheelbase width, and gear ratios along with clockwise and counterclockwise motor orientation. By using the system controller and implementing an integrator feedback loop, the command heading serves as a set point and the current heading from the IMU serves as the control system output. Using a basic integrator method the robot adjusts the angular velocity to orientate the robot towards its command heading.





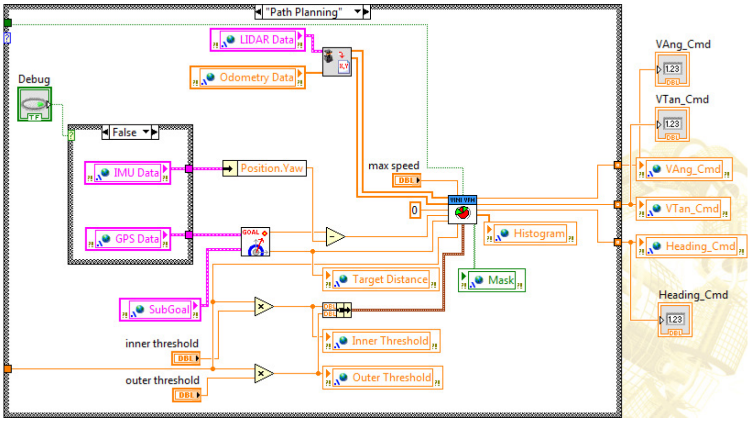


Figure 11. **Implementation of the Act loop to adjust the turning speeds of the wheel.** The motor loop showing the telemetry data that user can use to monitor the state of the robot (previous page). The path following VI that allows the robot to intelligently navigate the course by transposing the IMU data, the LIDAR data, and the GPS data.

5.5. Improving the vision perception and obstacle detection

One of the goals for this year was to implement vision based detection algorithms for autonomies robotic navigation; including planar perspective correction, line detection, grass filtering, and obstacle recognition. Using previously published [1][2][3] procedures for automotive road navigation, the team adapted solutions proved to be advantageous when traversing a smaller outdoor obstacle course at slower speeds. The importance of this project is that it can lead to potential breakthroughs and future applications in autonomous navigation and it is currently an on-going research for developing automated driving assistance.

Implementation of stereoscopic imaging for increase field of view

In order for the line detection algorithm to detect relevant lines with respect to the reference of the ground plane, the acquired image from the robot's perspective must be corrected for the planar distortion when observing the IGVC field from low elevation. A more suitable representation of lines and its locally geographical figures would be from a bird's eye (top) view. Logically being within the same dimension as the robot will inevitably express its own occupancy grid, this will allow for easy conversion between raw data and geographical features such as describing avoidable obstacles and potential paths to sub goals.

After a sample calibration images has been acquired, the tedious process of point-based calibration and is performed by marking and identifying specific points and coordinates within the image that represent real space dimensions. In this instance, four ends of the pizza boxes were selected as identifiable uniform objects as switch to serve for the point-based calibration routine. When generating the point-based calibration file, it is important to take detailed and consistent measurements, regardless of the calibration tool; a small deviation in error will result in a large skewing effect, further distorting upper background regions within the image. After sequentially specifying each marked point of interest and its respective real space coordinates, the distortion model is calculated. As a real-time operated navigational robotic system requires a high refresh rate and its perceptual environmental awareness, faster sensor acquisition and processing is a direct constraint for system response times. It was important to maintain processing structure that is minimalistic in computational intensity but still relatively accurate.

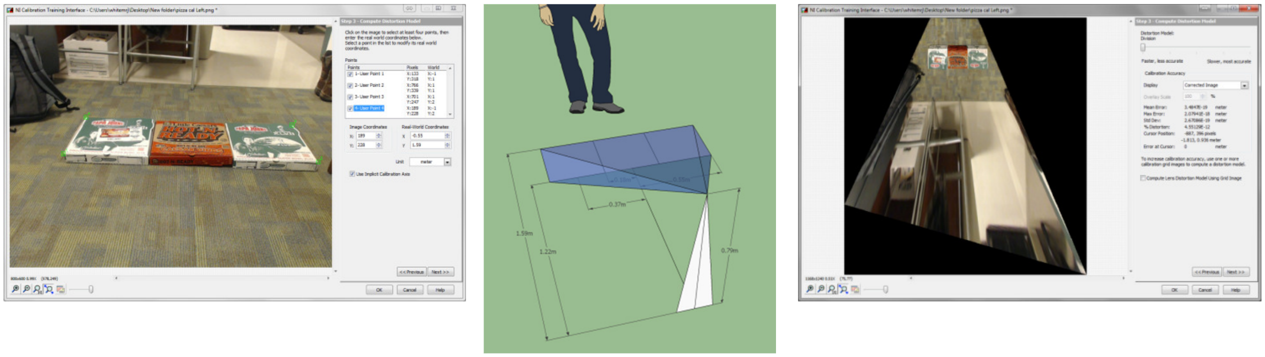


Figure 12. **Procedure for distorting the acquired image by means of point-based calibration.** The figure on the right represents the input image taken by the webcam. Using a design tool to accurately measure a set of known distances, the calibration reference was created (middle). The image on the right shows the output of the distortion model.

Implementation of grass filters to aid in line and obstacle recognition

Grass filtering was an important part of this project. If grass can be filtered out well object and line detection becomes much simpler. Thus a functional grass filter is critical to the success of the vision system. To solve this problem, the use of a SVM classifier was implemented. For this many chunks of known grass or non-grass pixels were extracted for a series of sample images. From these chunks features were extracted, including color, color moment, and variance. Using these values an SVM was trained. After several trials it was determined the best combination of accuracy and speed was generated using an SVM with kernel width of 5.

The SVM worked fairly well. It identified the majority of clearly defined lines, and a significant portion of lines that were not clearly defined. There was, however, more noise introduced into the image than the manual filtered provided. This was deemed not problematic because this is a preprocessing step, and future steps are well equipped to deal with small amounts of noise.

Figure 13. **Implementation of the SVM classifier to filter out grass from lines.**

Implementation of line detection algorithms in conjunction with the grass filter

An integral portion of the robot's navigation is identifying and remaining within the bounds constrained by white lines. Based on a lane line detection algorithm developed in the GOLD report [1], the team developed a process for detecting white lines on a grassy field. To begin setting up, the image in RGB space is converted to HSV space, using the value band to represent brightness. While not perfectly consistent, HSV space provided the most consistent brightness indicator, as compared to two other calculations found in [1], distance in RGB space as well as a more refined weighted distance measure.

Then, instead of doing two iterative passes over each image, the team applied a filter which accounts for the same calculation that the GOLD Report computes, accounting now for the vertical pass as well. The matrix can be expanded with more zeros to account for the expected line width. Applying the filter will amplify all pixels which are candidate line pixels, while grass field pixels will be reduced to lower intensities. Because of dead grass and patches of dirt, some grass pixels are misclassified as line pixels, the team utilizes the power of the grass filter to remove as much of the grass background as possible. Lastly, applying a simple Hough line transform to the remaining binary image, the output image turns out to be a binary image marking all remaining strong candidate lines.

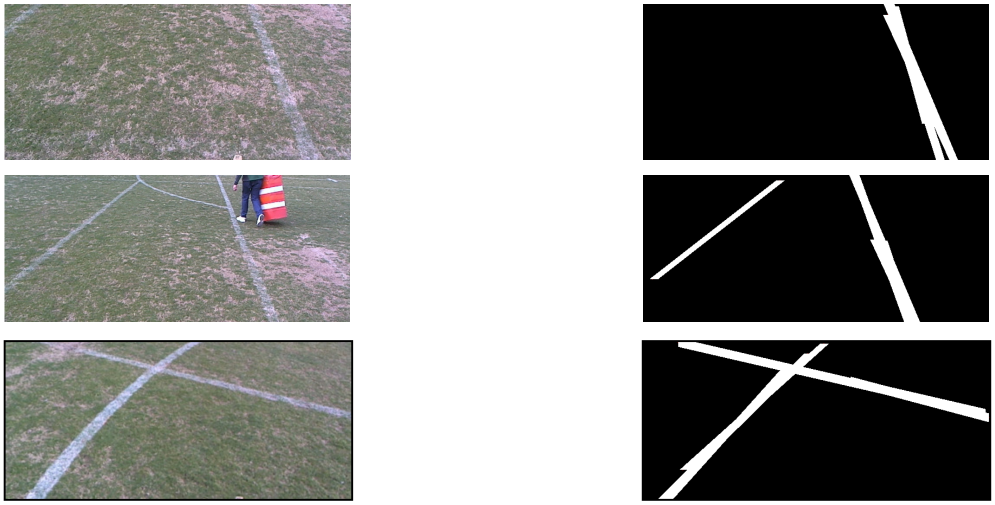


Figure 14. **Final results of the line filter used along with the grass filter.**

Implementation of the barrel detection techniques

In the past competitions, construction barrels have proven to be a rather tricky obstacle due to their pattern, reflective properties, and white-line features that tend to get mixed with white lines and thus detected as white lines and not barrels. Another obstacle that we had to overcome in this particular case consisted of dealing with barrels in the background and barrels that appear to overlap. Using LabVIEW as a programming language, the team explored barrel detection techniques using the Vision Assistant tools. This barrel detection algorithm is done in three major steps.

First, the vision assistant acquires the image from an image acquisition block. In our case this would be the direct output of the perspective correction with the image already distorted. Then, the algorithm will search in the entire picture for regions or matches whose parameters give it a score that is high enough to be detected. Such parameters include setting a color score weight, a minimum contrast, rotation invariance, feature types (i.e. color and/or shape), pixel accuracy, color sensitivity, and search strategy (i.e. conservative vs. aggressive). All these parameters can be tuned in LabVIEW to increase the accuracy of barrel detection but we must keep in mind that some parameters, like the conservative search, will increase computation time which will affect the speed of the robot. The final step of the color classifier algorithm searches everywhere in the image until a region with a high enough score based its parameters is found. Such regions will have bounding box drawn around them.

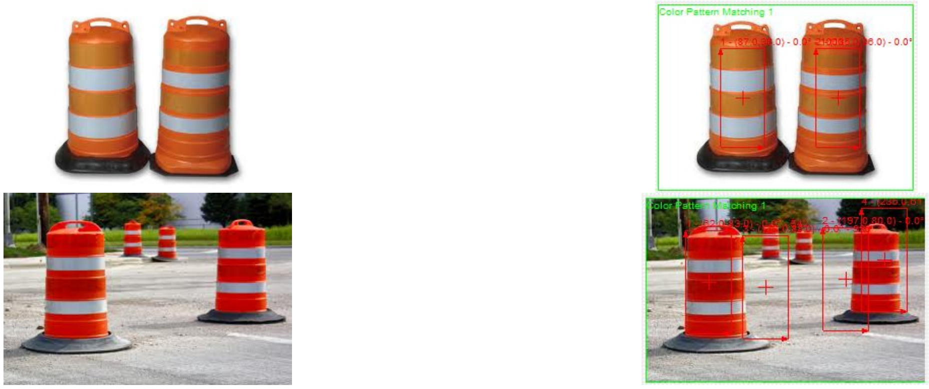


Figure 15. **Results of some barrel detection via color classification.** The top row shows the ideal case with the barrels being correctly classified due their features, sizes, pattern, and contrast. The bottom row shows a more realistic case where although the barrels were detected, the algorithm also detected some noise.

To improve the barrel detection techniques, the team decided to use monochromatic bands for the templates used in template matching. The idea behind this process was to reduce the variations given by natural causes such as lighting and shading. This resulted in matches that were closely packed together and closer to the barrel. The team also decided to apply a "skyscraper barrel mask" to the image wherever a barrel was detected. The "skyscraper barrel mask" would start from the point that the barrel was detected and then apply several masks (of similar size and shape) from the detected barrel to the top of the image. This would create blind spots past the barrel but since the robot cannot jump over the barrels or fly and since the robot should only be aware of its close immediate environment, the robot doesn't need to know what's beyond the first obstacle until after it passes the first obstacle. This would reduce computation time in our algorithm and provide faster reaction time for our robot. Although this newly thought part of the project has not yet been implemented, the figure below shows the idea of what the team is trying to pursue.

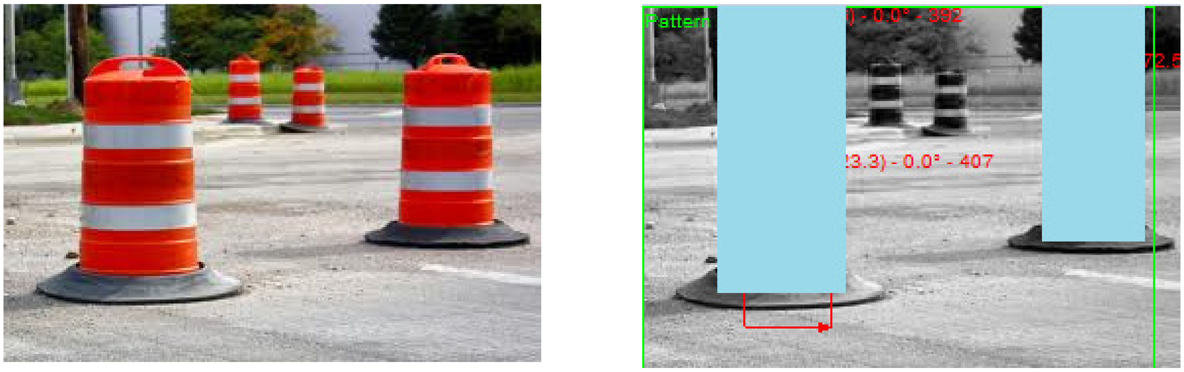


Figure 16. **Monochromatic band detection with "skyscraper" barrel approach.** This figure demonstrates the idea of using a "skyscraper barrel mask" that can potentially block out overlapping barrels in the background and random noise. *Though it was not implemented in this project, the goal of the team will be to have some major aspects of the solution implemented for this year.*

Appendix I: References

[1] M. Bertozzi and A. Broggi, “Gold: A Parallel Real-Time Stereo Vision System for Generic Obstacle and Lane Detection,” IEEE Trans. Image Processing, vol. 7, pp. 62-81, 1998.

[2] Anikstein, Alex. “IGVC Red Raven 2.0.” *California State University,* pp. 9 – 11. June 2012.

[3] Randy Breingan, et. al., “Reagle V.” *Embry-Riddle Aeronautical University*.