Georgia Tech 2014 RoboBoat Competition

Sinead O'Sullivan*, Michael Cormier*, Claudel Lessard-Therrien*, Samuel Seifert*, Dimitri Mavris*

*Georgia Institute of Technology, Atlanta, GA 30332 USA

{sc.osullivan, mcormier7, claudel.l.therrien, seifes1, d.mavris}@gatech.edu

June 13, 2014

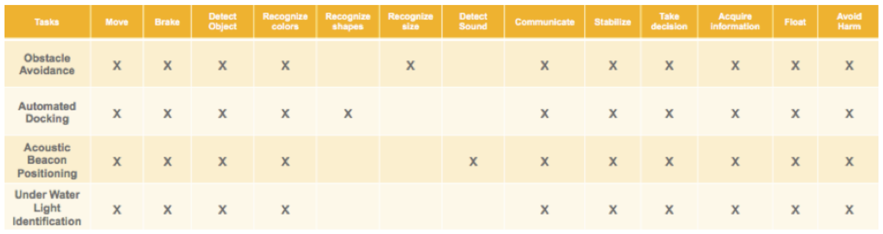
Abstract

Georgia Tech Marine Robotics Group (MRG) are taking an already built autonomous surface vehicle (ASV) for entry to the 2014 AUVSI RoboBoat Competition. The boat is a trimaran configuration, built for stability and speed, with a sonar system, underwater light sensors, a LIDAR, an IMU, a GPS and stereo cameras. This boat design, along with the sensors already listed, enable it to compete in each of the missions in the competition. The software architecture of the boat includes a hybrid reactive-deliberative behavior-based control, allowing ease of navigation, obstacle avoidance and overall task completion.

1 Introduction

The Aerospace Systems Design Lab (ASDL) at Georgia Tech has selected an abstraction based paradigm for robotics pioneered by Georgia Techs research on the Open Control Platform [1]. This modular paradigm allows for the reuse of sensors, drivers and code across multiple vehicles of different types. Our approach for RoboBoat 2014 is to continue this successful practice in adapting abstracted building blocks from previous projects. This abstraction based paradigm enables reuse of proven building blocks, allowing the MRG team to take an already built ASV and focus on rigorous simulated and physical testing of the integrated system.

A functional breakdown of the building blocks required for each mission is provided so that these building blocks can be described in greater detail. Figure 1 presents a matrix linking the building blocks into the missions in the RoboBoat rules description. A written description of the link between the missions and the autonomous functions is described in the following Section 2. The physical systems performing these functions are discussed in Section 2. The software and sensing architecture is described in Section 6.1. The functional elements are discussed in Section 1.5. Finally, the testing of the integrated system is discussed in Section 7.1.



Tasks	Move	Brake	Detect Object	Recognize colors	Recognize shapes	Recognize size	Detect Sound	Communicate	Stabilize	Take decision	Acquire information	Float	Avoid Harm
Obstacle Avoidance	X	X	X	X		X		X	X	X	X	X	X
Automated Docking	X	X	X	X	X			X	X	X	X	X	X
Acoustic Beacon Positioning	X	X	X	X			X	X	X	X	X	X	X
Under Water Light Identification	X	X	X	X				X	X	X	X	X	X

Figure 1: Building Block Matrix

2 Mission Breakdown

The competition missions will be completed by combining the abstracted autonomous functions described in Section 2. A tabular mapping between the missions and the autonomous functions required can be found in Figure 1. This section will describe how the competition tasks are completed using existing functions.

Mission One, Obstacle Detection and Avoidance (OD/OA), is one of the fundamental elements used in the navigation and control of Captain Hindsight. The boat will be provided with the specified exit gate and as a fundamental element of the navigation and control, the boat will by default perform Obstacle Avoidance as it is an always on behavior in the arbiter. This is completed using a Distributed Architecture for Mobile Navigation (DAMN) arbiter, described in Section 6.2.5.

Obstacle detection and avoidance is a reactive always on behavior in the DAMN arbiter and determines how far the boat can move in each direction without colliding with any obstacles. It relies on sensor data from the laser scanner and is augmented by the stereo computer vision. This behavior is augmented with knowledge of the vehicles overall width, length and maneuverability which enables the behavior to calculate the angle of the turn required to avoid an object. Closer objects thus require tighter turns to avoid. This behavior is always given the highest priority within the DAMN arbiter, since it is most important that the boat does not run into obstacles.

Mission Two, Symbol Recognition and ASV Docking, will utilize the multiple autonomous functions described in Table 1 with an emphasis on the template matching. Obstacle avoidance will be used to successfully enter the dock. To decipher the correct dock, Captain Hindsight will use the camera-aided object recognition with template matching to distinguish each of the docks. The docking itself relies on the object avoidance behavior, and the use of multiple LIDAR allows the Georgia Tech entry to focus on both near and far obstacles.

Mission Three, Acoustic Beacon Positioning, will be completed using hydrophone array described in Section 1.5. Once the ASV is within a specified range of the underwater pinger, it will then use both passive sonar and color detection to detect the GPS location of the buoy and it's associated color.

Mission Four, Underwater Light Identification, can once again be accomplished by using obstacle recognition to initially locate the light sequence. Once the vehicle has navigated to the sequence, the underwater light sensor will be used in conjunction with color detection software to establish the light sequence being illuminated.

3 Mechanical Design

As mentioned before, the RoboBoat 2014 team have utilized a previous RoboBoat entry as the mechanical design. This decision was made through a "make or buy" tradeoff. Reusing an already built RoboBoat allowed the team to make subtle mechanical changes to adapt to the 2014 competition, but most importantly it allowed maximum time to spend on rigorous virtual and physical testing of software for the missions, as described in Section 7.1. The mechanical layout will be described below.

3.1 Hull Design

A trimaran hull design, Figure 2, was selected as a trimaran has advantages over competing mono- and catamaran designs which includes reduced weight, simplicity, maneuverability, speed, and stability. A trimaran can be hollow and requires very little structure to connect the pontoons. The vehicle is three feet wide and just over four feet long, emphasizing a low weight.

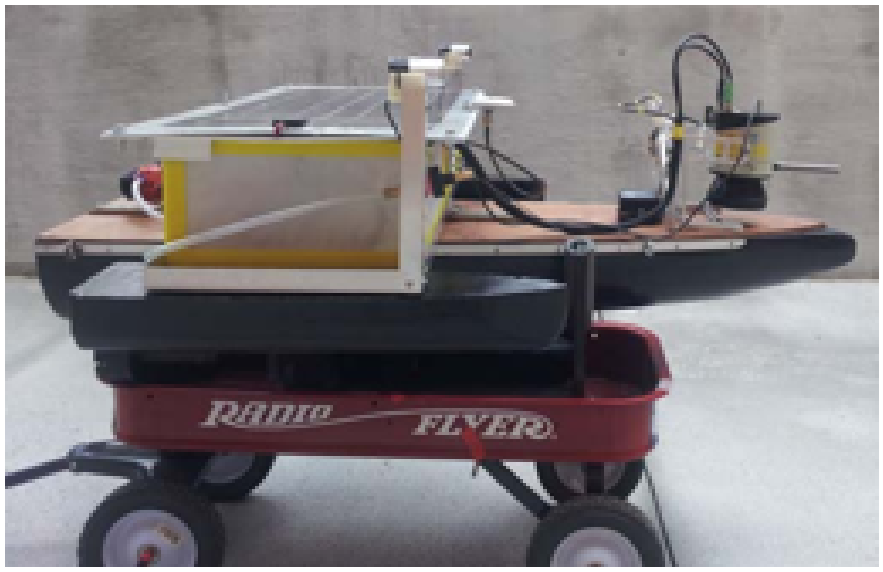


Figure 2: Trimaran assembled boat

3.2 Electrical Design

The boat electronics are divided into two separate systems, propulsion and sensors. Each system is powered from a dedicated battery to prevent power surges, and to allow R/C control over the vehicle even if the sensors or computer are offline. The propulsion system is powered from a six-cell, 5000 milliamp-hour lithium polymer (LiPo) battery, while the sensors are powered from a four-cell 5000 milliamp-hour LiPo battery. The use of LiPo batteries reduces the weight and size of the vehicle while providing the necessary power requirements. A simplified power diagram is shown in Figure 3.

3.2.1 System Assembly

The finished electronics box sits inside the frame that connects the pontoons to the main hull. This attachment provides additional stiffness to the frame, and allows the box to shift with the frame to adjust center of mass and pitch. Two cameras for stereo vision mount to the corners of the frame, maintaining a set distance and maximizing depth perception at farther distances. The 3D LIDAR mount sits at the nose of the vehicle to provide an unobstructed scanning. Finally, the lid of the box is actually a solar panel that provides all of the power to run the cooling fans, preventing the laptop from overheating. The solar panel generates enough power to also run both the IMU and GPS.

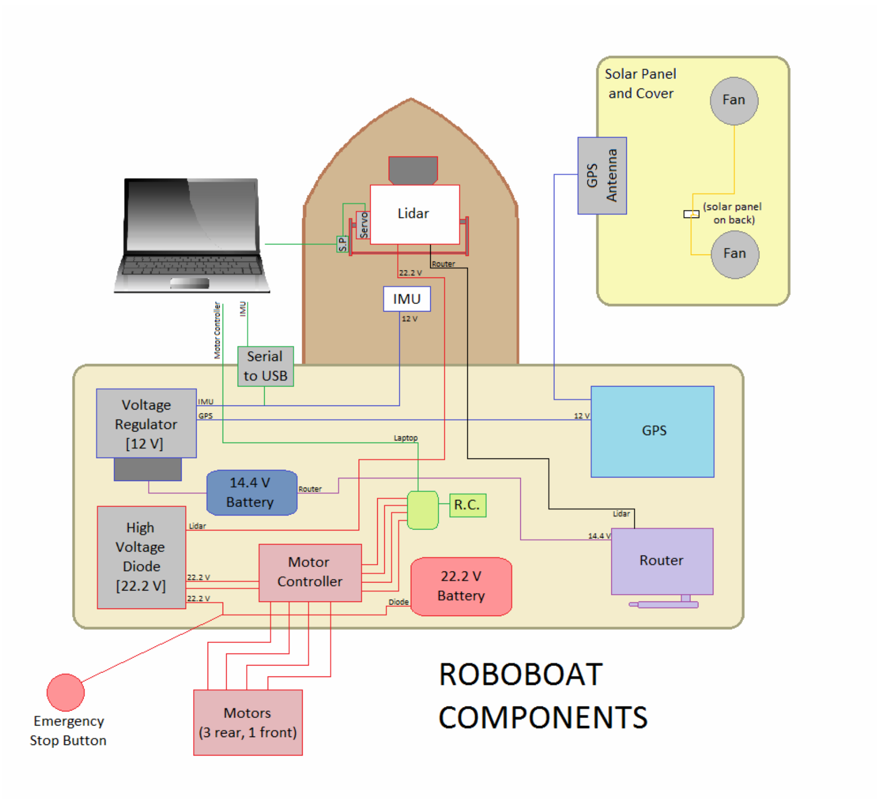


Figure 3: Electrical Design

4 System Identification and State Estimation

4.1 Identification of Boat Dynamics

Tests were created to get the relationship between the throttle of each motor and the linear and rotational velocities. While manually driving the boat on a lake, system inputs and outputs were recorded. Inputs consisted of throttle commands sent to the motors while outputs consisted in recorded data coming directly from IMU and GPS. Using that data, models were computed using Matlab Identification Toolbox to represent the dynamics of our boat.

For simulation purposes, a non-linear model was used to get a boat behavior as accurate as possible to test algorithms and code. This model was estimated using regressors and wavelet neural networks. The non-linear model was validated using data from another test to compare outputs of the model to actual data from the sensors as shown in Figures 4 and 5.

Notice in Figure 5 the simulated velocity is sometimes inverted from the actual data. This is due to the fact that the GPS gives positive velocity values even when the boat is going backward.

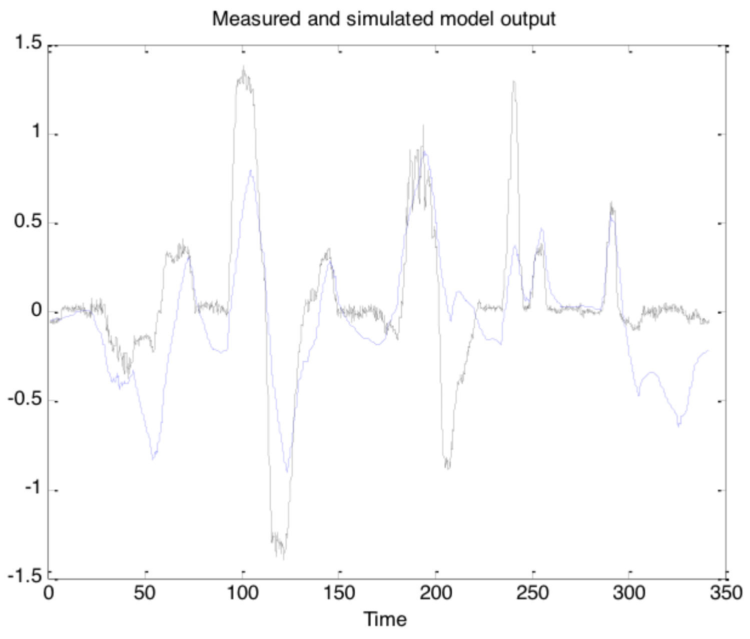


Figure 4: Validation of the model for the rotational velocity. Actual data shown in black, simulated data shown in blue.

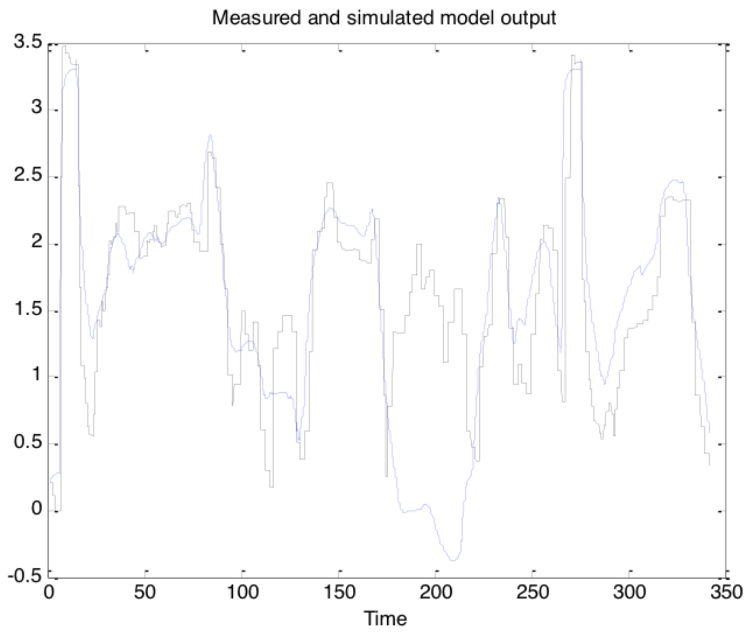


Figure 5: Validation of the model for the linear velocity. Actual data shown in black, simulated data shown in blue.

Using this dynamic model, we were able to simulate accurately the behavior of our boat. A linear state-space model was also computed using the System Identification Toolbox. This model was used to predict the states of the vessel. The same test data as for the non-linear model were used to get the standard A, B and C matrices of the state-space representation.

4.2 Velocity State Estimation

The measured velocity from the GPS tends to be noisy and is updated only once per second. Using this measurement directly to control the boat would cause some problems. Therefore, acceleration data was combined with the IMU and GPS to estimate the linear velocity.

Firstly, the state-space linear model of the boat was used to predict the linear acceleration using throttle inputs. This prediction was compared to the acceleration measured by the IMU and the difference was passed through a low-pass filter. This processing was done in order to get rid of the noise in the acceleration measurement, which was mostly caused by the motor vibrations.

Then, using the conditioned acceleration data, the velocity measured by the GPS, and the state-space model with a Kalman filter, a good estimation of the velocity was achieved. This method allowed elimination of the lag inherent in the linear velocity measurement.

5 Velocity State Estimation

5.1 Position Controller

The position controller sends a target heading and a target velocity to the velocity controller. It takes a target position coming from the DAMN arbiter as inputs as shown in Figure 6.

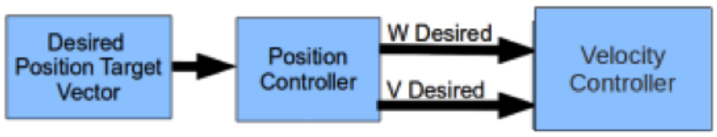


Figure 6: Position Control block diagram

This controller is open loop because no direct feedback loop was necessary since the target position is relative to the robot. Therefore, the linear velocity gains were scaled according to the relative distance of parallel to the heading of the boat (i.e. target position in x). The target heading was computed by adding a increment to the current heading of the boat based on the relative distance of the target perpendicular to the heading (i.e. target position in y). This algorithm gave better results than a PID algorithm in the simulation.

5.2 Velocity Controller

The velocity controller uses a basic PID algorithm. The linear and rotational velocity dynamics are assumed to be decoupled. One PID loop regulates the linear velocity and another regulates the rotational velocity (i.e. the heading). The velocity PID output is sent to each motor as a throttle command and the heading PID output is added to one side motor command and subtracted to the other side motor command to create a thrust differential as shown in Figure 7.

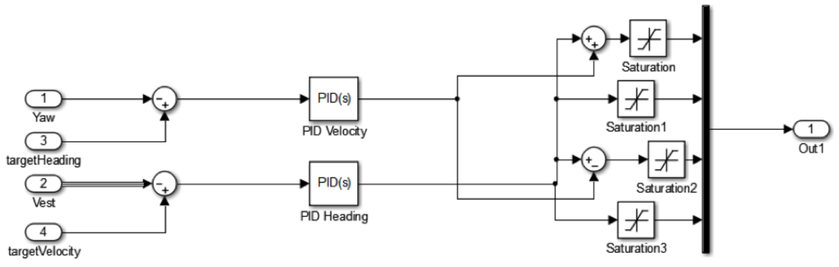


Figure 7: Velocity Control block diagram

Since the system is highly non-linear with the velocity, the derivative gain of the heading PID had to be high to reduce oscillation when the boat is trying to follow a heading. Because of the fact that the derivative action can make the system unstable, we used a low pass filter to get rid of the noise and smooth the response of the PID. The derivative was also computed by using a prediction of the future error instead of doing backward differencing to get a better behavior.

6 System Architecture

6.1 Vehicle Sensor and Software Architecture

The boat's system architecture has been successfully used on previous AUVSI RoboBoat entries. The software architecture resembles the Sense-Think-Act paradigm with an additional behavioral control aspect [2]. In the literature this is commonly referred to as Hybrid Reactive-Deliberative Control [3]. Figure 8 shows an overview schematic of the software architecture. In the Sense-Think-Act paradigm, each of these three fundamental stages feeds data into the next stage. In this architecture, sensor data acquisition, processing and actuation are done sequentially. The high level logic and decision stage (or thinking stage) rectifies multiple behavioral objectives though the use of a series of voting matrices. Each voting matrix, or behavior, (e.g. obstacle avoidance, hold heading, buoy navigation, etc.) outputs a local target position preference in the form of a vote matrix. All behaviors' votes are then fused in the DAMN arbiter, with the behavior with the highest votes being implemented. This is outlined in Figure 9. The output of the DAMN arbiter is then fed into the acting stage that consists of the position and velocity controller as well as the actual thruster motors.

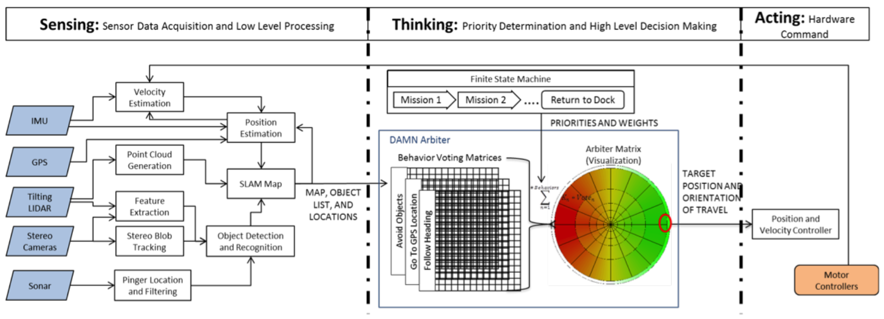


Figure 8: Vehicle Software Schematic

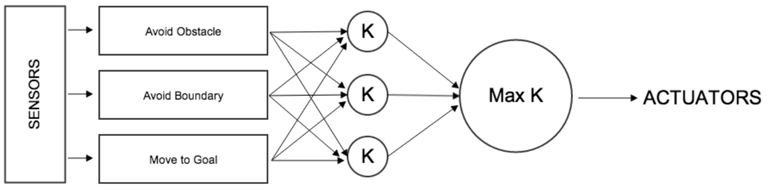


Figure 9: DAMN schema

6.2 Autonomous Functions

6.2.1 Environment Sensing

The primary source of environmental data will come from a single LIDAR mounted to tilting mechanisms, allowing for 3D point cloud generation.

Each LIDAR generates an array of ranges to the nearest object corresponding to a horizontal field of view from the LIDAR. The incorporation of a tilting mechanism provides a three dimensional array of distances constructed from these horizontal slices. Each ’slice’ of LIDAR data is translated into the boat’s coordinate frame and then into a fixed global frame using information from the IMU and the Kalman filters position estimator. The resulting data is a point cloud depicting the surrounding environment that compensates for the dynamics of the vehicle. This approach has been successfully used on other maritime platforms constructed by Georgia Tech.

6.2.2 Point Cloud Segmentation

Point clouds contain a large amount of data in the form of thousands of points from which we need to extract useful information, i.e. clusters, shapes, and features. This segmentation is done with the help of the Point Cloud Library (PCL). Points are clustered based on centroid and a matching algorithm identifies primitive shapes. All of these recognized primitive shapes and any clusters that could not be matched with a shape are then GPS-tagged. The tagged objects are stored which aids the vehicle in navigation and object avoidance.

6.2.3 Camera-aided Object Recognition

A stereo camera system using two Playstation Eye cameras are used. The setup of the boat allows for a wide baseline between the cameras giving a large field of view in which 3D information is available. In order to compute 3D information from a stereo vision system, pixels in the left image need to be associated to pixels in the right image. This is done with a block matching algorithm that is fast enough to run in real time. Two algorithms are used to extract 3D information from the cameras: firstly stereo blob-tracking and secondly template matching.

Stereo Blob Tracking: This is an algorithm based on the OpenCV computer vision library that is currently able to track colored objects to find the corresponding objects and buoys in the left and right camera frames. After the stereo blob tracker has found blobs in both frames, blobs are matched based on distance in image coordinates. A blob in the left image and the associated closest blob in the right image are assumed to be the same blob and allow us to compute a disparity value between the two images. The image coordinates together with the disparity enable us to project a blob into 3D space and compute the 3D location of each blob.

Template Matching: In addition to tracking colored blobs, the successful completion of missions requires a method of recognizing docking bays. The approach combines point cloud segmentation and template matching. Specifically:

1. Distances and sizes of objects retrieved from the point cloud segmentation are fed into the template matching module.
2. These objects are then projected into the current camera image to constrain the search space for the template matching algorithm.
3. The template matcher then extracts local feature descriptors from both the template and the current object in the image and computes a confidence level of the match.

6.2.4 Acoustic Triangulation

The Georgia Tech team plans on implementing a simple acoustic triangulation that will be designed and built by high school students as part of ASDL's STEM outreach program, STEP. Three hydrophones will be mounted to the hull of the boat. Knowing the fixed displacement between the three sensors and the speed of the underwater signal allows for the calculation of the estimated angle of arrival of the pulse.

Once the combinations have been evaluated, the angle data can be compiled and filtered to provide an overall average angle estimate to construct a dynamic bearing. These sensors will provide a heading and distance estimation to the acoustic source in the vehicles frame of reference, which can be turned into a global position using the vehicles GPS location.

6.2.5 Autonomous Navigation and Control

Autonomous functions combine all the sensory inputs through pre-specified behavioral routines or behaviors and generate motor commands through the DAMN arbiter which generates target positions based on the behavior weights, and high-level controls are realized as a finite state machine described below.

Finite State Machine: The high level control - or mission planner - is realized as a finite state machine that controls the weight of each behaviors vote and therefore its importance on the target position in the DAMN arbiter. By setting weights to zero, it can also dynamically deactivate behaviors. The transitions from one state to another are based on the detection of objects marking the start or end of any mission segment. Additionally, the mission planner ensures that the boat always remains in a defined state and never gets stuck in an obstacle or attempting a task.

7 Testing

7.1 Integrated Testing

Testing of the vehicle is critical to ensuring a robust design capable of competing at a high level. While each of the autonomy functions are being adapted from previous Georgia Tech Marine Robotics Projects, ensuring a robust and high level of performance from the integrated system will require extensive testing. The Georgia Tech team has had success in implementing a dual software and hardware testing approach.

7.2 Virtual Testing

The boat, its sensors, and dynamics in were simulated in the the 3D simulator Gazebo. Gazebo is integrated into Robot Operating System (ROS) and seamlessly hooks into ROS's message passing architecture allowing for simple interfacing of all codes. ROS is designed with the goal of minimizing the gap between simulation and real hardware. Therefore, the only modules that have to be changed are the drivers that feed sensor data into our ROS-based software architecture. Gazebo not only simulates physical properties of interconnected rigid bodies, such as friction, gravity, mass, etc. but also generates sensor data including GPS and IMU data and camera image streams. This setup allows testing of every module of the system architecture for the boat in simulation before deploying it on the boat. This includes code for position, velocity estimation and control and also all behaviors outlined in Section 2 including obstacle avoidance, global waypoints, blob tracking and even template matching and point cloud generation and segmentation. An example of a simulated buoy channel in Gazebo is shown below in Figure 10.

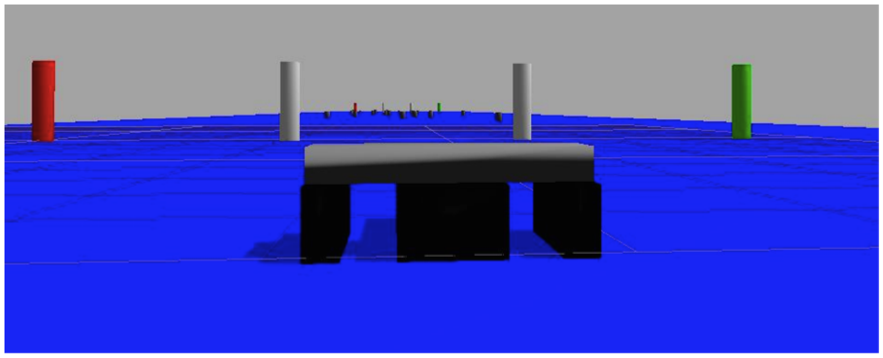


Figure 10: Gazebo simulation of Mission 1

7.3 Physical Testing

The purpose of physical testing is to ensure that the boat can robustly operate in the competition environment by reproducing the competition course. To replicate this environment, testing of the boat is performed at Sweetwater State Park. Access to this lake allowed the team to construct a mock competition course and test the vehicle in a large isolated space.

8 Conclusion

The Georgia Tech RoboBoat 2014 team utlized an already existing boat from a previous year's competition, as it allowed heavier emphasis to be placed on software development, as well as software integration and testing. ROS and Gazebo were used to test behavior algorithms and hardware/software interfacing before being tested physically. In addition, the team used an abstraction based paradigm which meant that sensors, drivers and code could be easily reused modularly across the system.

9 References

1. S. Clements, N. Kejriwal, L. Wills, B. Heck and G. Vachtsevanos, "Rapid prototyping of transition management code for reconfigurable control systems," in Rapid System Prototyping, 2002. Proceedings. 13th IEEE International Workshop on, Darmstadt, Germany, 2002.

2. R. Brooks, "A robust layered control system for a mobile robot," Journal of Robotics and Automation, vol. 2, no. 1, pp. 14-23, 1986.

3. G. A. Bekey, Autonomous Robots: From Biological Inspiration to Implementation and Control (Intelligent Robotics and Autonomous Agents), The MIT Press, June 2005.

4. J. Rosenblatt., "DAMN: A Distributed Architecture for Mobile Navigation," Pittsburg, 1997.

5. H. Bay, T. Tuytelaars and G. Luc Van, "Surf:Speeded-up robust features," in 9th European Conference on Computer Vision, Graz, Austria, 2006.

6. International Traffic in Arms Regulation (ITAR 22 CFR Section 121.1 et seq..

7. Export Administration Regulations (EAR) 15 Section 774, Supp.1, (Categories 0-9).

8. International Conference on Intelligent Robots and Systems, vol. 3, pp. 2421-2427, 2003.