Autonomous Surface Vehicle

Senior Design Group #8

Date Submitted: December 11th, 2012

Faculty Technical Advisor:

Professor Yan Meng
Yan.meng@stevens.edu

Group Members:

Alex Cihanowyz  
Francis Garcia  
Charles Steiner

Rover Development  
Sensor Development  
System Integration

“We pledge our Honor that we have abided by the Stevens Honor System”
# Table of Contents

Abstract 1  
Acknowledgement 1  
Project Final Design Plan  
  Introduction 2  
  Design Requirements 3  
    Contest Tasks 4  
      Task 1 – Propulsion Test 4  
      Task 2 – Navigation Test 5  
      Task 3 – Speed Test 5  
      Task 4 - Channel Navigation Test 5  
      Task 5 – Poker Chip 6  
      Task 6 – Jackpot 6  
      Task 7 – Cheater’s Hand 7  
      Task 8 – Hot Suit 7  
      Task 9 – Card Exchange 8  
      Task 10 – Return to Dock 8  
Design Approaches 9  
  Sensors 9  
  Actuators 12  
  Computer 13  
  Software 14  
    Simulation Environment 15  
Competition Approaches 16  
  Task 1 – Propulsion Test 17  
  Task 2 – Navigation Test 17  
  Task 3 – Speed Test 17  
  Task 4 - Channel Navigation Test 18  
  Task 5 – Poker Chip 19  
  Task 6 – Jackpot 19  
  Task 7 – Cheater’s Hand 19  
  Task 8 – Hot Suit 19  
  Task 9 – Card Exchange 19  
  Task 10 – Return to Dock 19  
Summary 20  
References 21  
Appendices  
  Hot Suit Transmission Format A  
  Task Layout B  
  Scoring Sheet C  
  Sample OpenCV Program D  
  Financial Budget E  
  Project Schedule F
Abstract

There are myriad problems in the world that are too great in magnitude for humans to solve. One recent example of this is the Deepwater Horizon oil spill in 2010, where an explosion caused millions of barrels of oil to leak into the ocean through a hole in a pipe underwater. This caused a massive oil spill that required more manpower and time to clean as time passed. This need for a solution caused several research groups to develop robotic platforms to help clean the oil spill. Most familiar of these is a robot developed by MIT known as the SeaSwarm, which autonomously navigates through the oil slick and absorbs the oil with a nano-fiber net. Other recent problems include Hurricane Sandy, particularly in Hoboken and in the New Jersey shore area where towns were flooded and required boats or high water vehicles to traverse through flooded areas. With autonomous boats that can navigate to a specific point in an area, it can deliver a payload such as food, water or medicine while emergency responders can prioritize evacuating people rather than food delivery.

In order to foster the design process and develop skills in systems engineering by completing missions with autonomous robotic boats, the Association for Unmanned Vehicle Systems International (AUVSI) developed the RoboBoat Competition. This competition is comprised of ten tasks that simulate problems that an autonomous boat may encounter, from channel navigation to receiving a payload from a dock.

For the 3rd year in a row, Stevens Institute of Technology is submitting an entry to this competition in hopes to win the competition and allow its students to develop skills to design a solution to a series of problems. Unique to this year’s entry is the inclusion of a team of electrical engineers, who will be tasked with the programming and sensor integration of the boat. They will use sensors such as a Kinect sensing device, high definition cameras, sonar sensors, and infrared sensors. After completion of the project, the team will have an autonomous surface vehicle (ASV) that utilizes all these sensors to complete the tasks set out by the AUVSI.

Acknowledgement

The electrical team would like to acknowledge Professor Yan Meng for her advice and guidance throughout the semester, Professor Fisher from the mechanical engineering department for allowing a team of electrical engineers to join in on the AUVSI Robo-boat team, and the electrical engineering department for funding the project. The team as a whole would also like to thank Davidson Lab for allowing the use of their wave tanks for testing.
Introduction

The purpose of this project is to design and create an autonomous boat to compete in the AUSVI Roboboat competition. This competition is an obstacle course designed to be completed by using multiple sensors on the boat. Since the rules to the 2013 competition are not yet released, the team will use last year’s rules to begin testing equipment and start writing our base programming.

The competition is comprised of two mandatory sets of tasks and one optional set of tasks. The first set must be completed sequentially before the robot can proceed to the second set of tasks. The second set can be completed in any order, while the third set of tasks can be completed at the same time as the second set.

This is the third time that Stevens is submitting an entry for this competition. As of now the boat that will be used in the competition is already constructed and a majority of the sensors that will be used in to complete the tasks have been purchased by previous teams. However, many things can be improved over past entries. Last year’s entry had navigation problems, where the boat ended up following a duck around the course rather than navigating properly through the course. This prevented the team from completing the first set of tasks, which ultimate resulted in poor competition performance.

Testing and configuration for the sensors is currently being performed by the team, and the base program is being rewritten from LabView to C++. Other main tasks include converting a mini-rover from wired communication to wireless communication, implementing a Kinect sensor for better obstacle detection, and simplifying the circuitry and wiring contained in the control box. Once all the sensors are configured and attached, testing will be done in the Davidson Lab on the Stevens Institute of Technology’s campus.
Design Requirements

The AUVSI has several design requirements that need to be met with regards to the boat’s construction. As of December 10, 2012, the rules have not been released yet for the 2013 Competition. From recent discussions on the AUVSI Robo-boat Competition forum, there will be several changes made to the competition. Officials state that they will be adding two new challenge stations and removing some. Fortunately, the same sensor suite that was used in 2012 will be able to meet the need of the 2013 competition. However, the new challenge stations may require the addition of new actuators. When the updated rules are released, the team will make any necessary modifications to meet the requirements of the newer set, from selecting new actuators to developing solutions to solve the new problems posed at the new challenge stations. Additionally, the group will base their overall design to meet the requirements set in the document “5th RoboBoat Competition – Final Rules” from the 2012 Competition.

The general requirements for the boat are outlined in the following chart from the 2012 rule set:

<table>
<thead>
<tr>
<th>Rule</th>
<th>Requirement</th>
<th>Detailed Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>Autonomy</td>
<td>The vehicle must be fully autonomous and all decisions must be taken onboard the ASV</td>
</tr>
<tr>
<td>R2</td>
<td>Buoyancy</td>
<td>The vehicle must be positively buoyant and be buoyant for at least 30 minutes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>in the water</td>
</tr>
<tr>
<td>R3</td>
<td>Communication</td>
<td>The vehicle cannot send or receive any information while in autonomous mode, with</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the exception of communications for reporting the ‘Hot Suit’ target</td>
</tr>
<tr>
<td>R4</td>
<td>Deployable</td>
<td>The vehicle must have its own 3 or 4 points harness for crane deployment</td>
</tr>
<tr>
<td>R5</td>
<td>Energy Source</td>
<td>The vehicle must use self-contained electrical energy source. Sailboats are permitted</td>
</tr>
<tr>
<td>R6</td>
<td>Kill Switch</td>
<td>The vehicle must have at least one 1.5in diameter red button located on the vehicle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>that, when actuated, must disconnect power from all motors and actuators.</td>
</tr>
<tr>
<td>R7</td>
<td>e-Kill Switch</td>
<td>In addition to the physical kill-switch, the vehicle must have at least one</td>
</tr>
<tr>
<td></td>
<td></td>
<td>remote kill switch that provides the same functionality</td>
</tr>
<tr>
<td>R8</td>
<td>Payload</td>
<td>The vehicle must have a place to mount a payload up to a 60-inch cube weighing up to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 lbs</td>
</tr>
<tr>
<td>R9</td>
<td>Payload Location</td>
<td>The payload must have an unobstructed view of the sky and front of the vehicle</td>
</tr>
<tr>
<td>R10</td>
<td>Propulsion</td>
<td>Any propulsion system is fine (thruster, paddle, etc) but moving parts must have a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>shroud</td>
</tr>
<tr>
<td>R11</td>
<td>Remote Controllable</td>
<td>The vehicle must be remote-controllable to be brought back to the dock</td>
</tr>
<tr>
<td>R12</td>
<td>Safety</td>
<td>All sharp, pointy, moving, sensitive, etc. parts must be covered and clearly identified</td>
</tr>
<tr>
<td>R13</td>
<td>Size</td>
<td>The vehicle must fit within a six-foot long by three foot wide by three foot high “box”</td>
</tr>
<tr>
<td>R14</td>
<td>Surface</td>
<td>The vehicle must float or use ground effect of the water. Mostly submerged/flying is</td>
</tr>
<tr>
<td></td>
<td></td>
<td>forbidden</td>
</tr>
<tr>
<td>R15</td>
<td>Towable</td>
<td>The vehicle must have designated tow points and a tow harness installed at all times</td>
</tr>
<tr>
<td>R16</td>
<td>Waterproof</td>
<td>The vehicle must be rain/splash resistant. The competition is held “rain or shine”</td>
</tr>
<tr>
<td>R17</td>
<td>Weight</td>
<td>The vehicle must be 140 lbs or less.</td>
</tr>
</tbody>
</table>
Relevant to the group’s design interests are R1 (Autonomy), R3 (Communication), R5 (Energy Source), R6 and R7 (Kill switch and E-kill switch), R10 (Propulsion), and R11 (Remote-controllable). Since this is the third year that Stevens is competing in the AUVSI Robo-Boat Competition, there is already an existing boat chassis that meets the rest of the physical design requirements, with minor adjustments from the mechanical engineering team.

Meeting these general specifications will not require a significant amount of effort and engineering design due to their necessity in the physical construction of the boat, such as R2 (Buoyancy) and R13 (Size). The main requirements for the electrical engineering team to meet come in the form of three sets of tasks that comprise the final competition. The group will allocate the majority of its time and resources to completing these tasks successfully.

**Contest Tasks**

The competition is currently comprised of three sets of tasks that must be completed to obtain points for final scoring and allow for officials to measure the performance of each boat. The first set is made up of four sequential, mandatory tasks related to navigation and localization. Failure in these tasks means that the other sets of tasks cannot even be attempted. Because of this high risk, the team will devote most of its time testing boat functionality in these tasks so that the boat is able to complete them in any condition.

The second set of tasks consists of four challenge stations that can be attempted in any order. Each challenge is casino-themed and requires the use of image processing and other different sensors to complete them successfully. The third set consists of two optional challenges; one challenge is performed at the same time as the second set, while the other can only be attempted after completion of every other task.

**First Set:**

**Task 1 - Propulsion Test**

In this task, the team will demonstrate the amount of thrust the ASV can generate in ten seconds by attaching it to a strain gauge. This task does not have to be performed autonomously, so the group will be able to control the boat with a remote control.
Task 2 - Navigation Test:
In this task, the boat will demonstrate the ability to steer a steady course by navigating to the starting gate from the dock. A diagram of this task is shown in Figure 2.

Task 3 - Speed Test:
After the completion of task two, the vehicle will then be timed on how long it takes to transit between the starting gate and the speed gate. The starting gate and speed gate will be located such that it is possible to travel between the two in a straight line.

Task 4 - Channel Navigation Test:

After passing through the speed gate, the boat then will navigate a channel outlined by red and green buoys. The boat will be expected to use the 3R (red-right-return) navigational mantra, which means the boat will keep the red buoys on the port side when entering the channel and on the starboard side when exiting the channel. The boat will avoid the yellow buoys, which represent obstacles, and navigate through the channel until it reaches a blue buoy, which represents the end of the channel. From this buoy, the boat will be able to navigate to the challenge stations for the second set.
Second Set

Task 5 - “Poker Chip”

For this task, the boat locates a landing zone and makes contact with it by docking. Once it is there, it will deploy a rover that will navigate the landing zone and pick up a “poker chip,” which is a hockey puck covered in black Velcro. The rover then will navigate back to the boat and deposit the poker chip in the boat. Upon completion of the task, the boat will then un-dock and proceed to the next task.

![Figure 4: Poker Chip](image4)

Task 6 - Jackpot

In this tank, the robot will travel through a field of poles which each have a button attached to them. Next to one of these poles is a white buoy full submerged underwater, which signifies that the pole corresponds to the “jackpot.” If this button is pressed, the team will complete the task and earn the maximum amount of points for the task. If the robot presses any of the other buttons, the boat fails the task but will still earn half of the points for the task.

![Figure 5: Jackpot](image5)
Task 7 - Cheater’s Hand

In this task, there is a large printout of the “cheater’s hand” located on the shore, which consists of five images of playing cards. One card will have a blue square on it rather than a suit. The robot is required to locate this card and shoot it with a water jet to complete the task.

![Cheater's Hand Diagram](image)

Figure 6: Cheater’s Hand

Task 8 - Hot Suit

For the “Hot Suit” contest, there is a set of metal signs on the shore, each marked with one of the four suits of playing cards. One of the signs will be approximately 20°C warmer than the other signs. The boat is required to locate this sign and report which suit is hot (Clubs, Diamonds, Hearts, Spades) as well as its GPS coordinates.

![Hot Suit Diagram](image)

Figure 7: Hot Suit
Third Set:
Task 9 - Five Card Draw

In this optional task, each team is given a random set of 5 playing cards at the start of each run. The “dealer” will have a random hand of 5 playing cards as well, which will remain fixed and visible during the competition.

At each challenge station, there is a card exchange station. You can swap any number of cards in your hand to get the best possible hand that will beat the dealer (following poker rules). The card will be attached magnetically to a card exchange board and will require the boat to remove it with a stream of water.

![Figure 8: Five Card Draw]

Task 10 - Return to Dock

This task is simple. In order to successfully complete this task, the boat has to autonomously return to the dock. It will complete this by returning to the channel that it navigated in the fourth task. It will then navigate through the channel the opposite way by following 3R (red-right-return). It will then return to the dock.
Design Approaches

As previously mentioned, this is the third year that Stevens is participating in the competition. This means that there are pre-existing sensors, actuators, and computer components from previous years available for this year’s team to use in the construction of the robot and its systems. As part of the design process, the group determined whether or not the existing components are still suitable for the contest and whether or not they need to be replaced or upgraded. Based on a preliminary analysis of the datasheets, the group determined that each sensor can fulfill the needs required by the competition. If any component does not perform up to specification or cannot be used in a sub-contest, the group will then determine the appropriate sensors and purchase them.

Sensors

**Microsoft HD Web Camera 6000**

For simple vision applications, the group will be using Microsoft HD 6000 Web Cameras. These cameras are connected to a computer through USB 2.0. They have a resolution of 1280 x 720 pixels, much larger than required by the computer vision libraries the team will use. It is also able to capture video at 30 frames per second.

![Figure 9: Microsoft HD Web Camera](image)

**Xbox Kinect Sensor**

In addition to the HD Web Camera, the group will be using an Xbox Kinect Sensor to do advanced image processing. The depth sensor and RGB camera will be useful to help determine nearby obstacles. It has a range limit of 4-11 feet. This is useful so the boat will be able to detect any obstacles before they are near enough for the robot to collide with them. The sensor updates at 30 Hz and takes images with a 640 x 480 resolution.

![Figure 10: Microsoft Xbox Kinect Sensor](image)
GlobalSat BU-353 GPS Sensor

The GlobalSat BU-353 GPS Sensor will be necessary for navigation and certain tasks in the competition. The sensor has a frequency of 1575.42 MHz with -159 dBm sensitivity. It outputs NMEA 0183 code at a transfer rate of 4800 baud. It can be powered through USB with negligible current draw. It also has the benefit of a small form factor and low weight of 2.2 oz.

![GlobalSat BU-353 GPS Sensor](image)

Figure 11: GlobalSat BU-353 GPS Sensor

Raytek CM IR Sensor

The IR sensor chosen will be used in the hot suit competition. It outputs a signal from 0 – 5V or an output based on calculations similar to a J or K-type thermocouple. The sensor is made with a NEMA-4 stainless steel housing, which is designed for outdoor use to provide protection against rain, splashing water, and water in general. The sensor interfaces with the computer through RS232 with a 150 ms response time. It can detect a temperature with an accuracy of ±2°C and a resolution of 0.1°C. This is fine for the application, because the temperature difference between the signs will be far greater than this.

![Raytek CM IR Sensor](image)

Figure 12: Raytek CM IR Sensor
**Ocean Server OS 5000 US Kit**

The OceanServer OS5000 US Kit is a 3-Axis digital compass that uses three axis magnetic sensors, three axis accelerometers, and 24-bit A/D converters to determine heading, roll, and pitch. It is powered through USB and can be connected with RS232 Interface. It has a .5 degree compass accuracy with a 0.1 degree resolution. It has a small form factor and a small current draw of less than 30 mA. It operates from 4800 to 115000 Baud, and provides updates at 40 Hz.

![Ocean Server OS5000 US Kit](image)

**Figure 13: Ocean Server OS5000 US Kit**

**XBee Transmitter and Receiver**

X-Bee ZB is a wireless USB transceiver solution that will be newly implemented to the boat. The new rules will require the rover to no longer be tethered to the boat. It is imperative the wireless system meet two requirements. First, range must be long enough to ensure the rover will not be lost while performing its task. Second, reliability is important to ensure no time is wasted trying to communicate with the rover as it performs its task. X-Bee ZB fulfills both these requirements with a 120m outdoor range and reliable 2.4 GHz RF communication.

![XBee Transmitter and Receiver](image)

**Figure 14: XBee Transmitter and Receiver**
Humminbird Fish Finder 570 DI

The “fish finder” sensor is a type of underwater sonar that will return an image of what is below a water craft such as a boat. It produces a down image which shows structure and activity directly beneath the boat. The sensor features a 640 x 320 screen and can measure a depth of 250’. The down imaging processes at 455 kHz or 800 kHz.

Figure 15: Fish Finder 570 DI

Actuators
Seabotix BTD-150 Thrusters

The thrusters that the Stevens ASV teams have used in the past are Seabotix BTD-150 Thrusters. They run off 19.1V DC and a maximum power of 110W. At peak thrust, they draw 5.8A for 30 seconds; continuous thrust draws 4.25A. After performing a cost benefit analysis and operational test, the team decided that they will use these thrusters again. The team as whole is not getting funding from the Davidson Lab and the mechanical team is using their budget for their designs. With that in mind, each thruster currently costs $1000, and it would not provide a significant advantage upgrading two of them.

Figure 16: Seabotix BTD-150 Thrusters
Sabertooth 2x25 v2 Motor Controller and 2x10 Motor Controller

The motor controller chosen to run the thrusters is a Sabertooth 2x25 v2 Motor Controller. This drive is able to provide two motors with 25A each. It is also able to supply a peak current of 50A to each motor for a few seconds, and if more current is supplied than desired, it has thermal and overcurrent protection to protect the drives. Other features include the ability to control motors with analog voltage, radio control, or serial commands, and independent speed and direction operating modes for each motor. This makes it the ideal driver for a differential drive robot application like the boat.

Several more key features include regenerative motor drive, in which batteries will be recharged when the robot is commanded to slow down or reverse. This is highly desirable to prevent batteries from running out while the boat is in the middle of the competition. The drive also switches at 32 kHz, allowing it to make very fast stops and reverses. To control the driver, there are libraries for Arduino as well as PIC Microcontroller.

Figure 17: Sabertooth 2x25 v2 Motor Controller

The 2x10 Motor Controller has similar features to the 2x25 v2. The main difference is that the driver provides up to 10A to two DC brushed motors with peak currents of 15A. This controller will be used to control the mini-rover.

Computer

Arduino Uno

The Arduino Uno is a prototyping board based on the ATMega328 microcontroller. It has 14 digital input and output pins, 6 of which can be used as PWM outputs, and 6 analog outputs. It can be powered through USB and at any input between 7V to 12V. It has 32 kB of flash memory and operates at a 16 MHz clock speed. The board and programs are open source, which will provide the team with a lot of resources to allow for easy development, troubleshooting, and improvement on existing code. Additionally, several group members have experience programming with Arduino.

For simple control of the actuators, the group will use four Arduino Unos connected through USB. One Arduino will serve as an onboard processor for the rover. The other three will perform functions for the robot such as taking input from sensors and sending controls to the motor controller.
Since the Arduino is generally regarded as a good board for temporary prototyping, the group can replace it with a permanent solution, such as a PCB with an embedded PIC microcontroller or ATmega chip.

**Computer Components**

The computer uses the following components:

<table>
<thead>
<tr>
<th>Component</th>
<th>Model</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motherboard</td>
<td>Zotac H55ITX-C-E</td>
<td>Intel LGA 1156</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Mini-ITX Form Factor</td>
</tr>
<tr>
<td>CPU</td>
<td>Intel Core i3 550</td>
<td>3.2 GHz Dual-Core</td>
</tr>
<tr>
<td></td>
<td></td>
<td>73 W Intel HD Graphics</td>
</tr>
<tr>
<td>RAM</td>
<td>8GB G-Skill RipJaws</td>
<td>240-Pin DDR3 1333 (PC3 10666)</td>
</tr>
<tr>
<td>SSD</td>
<td>OCZ Agility 3 60 GB</td>
<td>OS</td>
</tr>
<tr>
<td>SSD</td>
<td>OCZ Onyx 30 GB</td>
<td>Data</td>
</tr>
<tr>
<td>Hard Drive</td>
<td>Seagate FreeAgent GoFlex 1TB</td>
<td>Removal USB 3.0</td>
</tr>
<tr>
<td>Power Supply</td>
<td>Mini-Box M4-ATX</td>
<td>250W</td>
</tr>
</tbody>
</table>

These components have been used with the boat for the past two years. After much discussion, the team decided to keep this computer configuration for the time being, due to cost restraints and projected effectiveness. If the computer seems to be operating below a useful range, there are several simple upgrades that can be made, such as increasing the RAM to 12 or 16GB or upgrading the processor to an Intel Core i5-2500K, which operates at 3.3GHz with a Quad-Core.

**Software**

**Operating System**

The computer will run Windows 7 operating system due to the operating system requirements of the Kinect sensor. There is recent discussion within the team about the possibility of upgrading to Windows 8, based on user experience. This can introduce incompatibility issues as well as bugs that would be detrimental to the robot’s development.

**Programming Environment**

     Visual Studio 2010 and the Arduino development environment will be installed on the robot to do necessary programming for the ASV. The robot will be programmed in C++ as advised by our advisor. Previous boats were programmed using LabView, which the group deemed to be too slow for the competition. This also allows the group to use powerful open source computer vision libraries.

**OpenCV**

     OpenCV is an open source computer vision library that provides functions to perform image processing and analysis on a single image or on a stream of images. A preliminary program was written
to test the functionality of OpenCV with C++. The program first takes an image converts it from RGB (red, green and blue) to HSV (hue, saturation, and value). It then calculates the HSV of each pixel. It then filters through to find pixels with a certain HSV value, which corresponds to a specific color, such as red. It then performs a Hough Transform, which is a feature extraction method that can find circular objects within an image. The algorithm for this process is to perform a circular sweep at each edge in an image and assign a “vote” to each point. The points with the most votes are most likely to be circles. The results of this program are shown below:

![Figure 18: OpenCV Program Result – First window shows the original stream with predicted Hough transformed circles; Second window displays the original stream in HSV color-space; Third window displays the stream after color filtering, with the pixels above the threshold appearing “white.”]

**OpenNI and OpenKinect**

OpenNI and OpenKinect are two open source Kinect libraries that the team is looking into using. There are different benefits and disadvantages to each library. The next phase of design will involve looking into both libraries to see which one is more suited to the application. It may be possible for neither to be suited to the application, as OpenCV also has functions for the Kinect as well.

**KinectSDK**

The KinectSDK consists of “official” Microsoft Kinect libraries. This is another Kinect library that the team is looking into using. So far, the team has used it to verify that the Kinect was working correctly, as last year’s Kinect broke.

**Simulation Environment**

One new addition to the project is the use of the WeBots simulation environment. WeBots gives the user the ability to model a physical robot with different sensors, and program a robot controller using different languages such as C or C++. This will allow the group to simulate different concepts for the controller of the boat without using the boat itself. This will speed up testing and development of programs, as it will not be necessary to use the physical boat to test new programs. WeBots also has the ability to simulate different lighting conditions, which was one reason why last year’s design failed to complete the navigational portion of the competition.
Figure 19: WeBots Simulation containing a modeled robot and different colored buoys.

**Competition Approaches**

Several design requirements listed by the AUVSI Foundation do not merit significant design attention by the group due to their necessity for the project, such as autonomy and propulsion. Several requirements have also been addressed in the pre-existing boat design, such as kill switches, remote control, and waterproofing. The main design focus that the group will be concerned with is the competition and completion of the three sets of tasks. However, this section of the report will briefly address some of the design requirements for the boat’s construction that will require modification to the existing chassis:

**Communication** – The vehicles cannot send or receive any information while in autonomous mode (with the exception of communications for reporting the Hot Suit target)

This particular requirement is expected to change slightly for this year’s competition. In past competition, the rover used to complete the Poker Chip task had been tethered to the boat by USB cable. For this coming competition, the change will be made so that the rover will communicate wirelessly. This won’t affect the boat design significantly as the wording of the rule will change due to technicality.

**Energy source** – Vehicle must use self-contained electrical energy sources

The past teams have used two 12V batteries in series to provide X W of power to the computer and propulsion system. The configuration previously chosen had the power to cause power dips and spikes in the computer, which could damage the system. Based on a preliminary system analysis, the
batteries appeared to be connected to both the computer and the motors at the same time. In order to fix this, the group will switch the power system over from two 12V batteries powering the whole system to one 24V battery powering the computer and one 24V battery or equivalent battery configuration powering the thrusters.

**Propulsion** - Any propulsion system is fine (thruster, paddle, etc) but moving parts must have a shroud

The team will not change the thrusters, but may change the configuration. Currently, the boat is expected to steer with differential steering. This method of propulsion may be preferred for the thruster task and the speed task, but will be slower to turn during the channel navigation. There is also concern that the thruster is unable to operate and produce thrust in reverse, which means that the boat would not be able to leave when it docks. Because of these reasons, the team will be working with the mechanical engineering team also involved with the project on creating a steering rudder actuated by a gear-motor that was previously used for the tether winch. This will allow for more precise steering during the navigation as well as the ability to reverse and direct the boat. It is also possible to keep the current configuration and just add a rudder.

**First Set**

**Task 1 - Propulsion Test**

The propulsion task is straightforward. The only requirement is that the boat demonstrates the maximum amount of thrust that it can produce over ten seconds. This can be accomplished simply by running the motors at full power for the ten seconds.

**Task 2 – Navigation Test**

This task will be completed by using the cameras and the Kinect on the boat to navigate to the first set of buoys. A discussion with previous group members about the competition may be necessary; if GPS coordinates of the buoys are known or if we are allowed to do test runs through the course, the boat could just use known or learned GPS coordinates to navigate to the start gate. The OceanServer digital compass will be used to ensure proper heading facing into the buoys. If a test run is allowed, the group will attempt to implement simultaneous localization and mapping (SLAM) so that the boat will know the environment and be able to navigate through it.

If neither of these is possible due to how the competition is run, the robot will then navigate to the starting gate by rotating in place at the dock until it registers the closest red and green buoys with the camera and the Kinect.

**Task 3 – Speed Test**

For this task, the robot will use the camera to orient the red buoys to the left and the green buoys to the right. It will then use the depth sensor on the Kinect to locate the nearest buoys. Once it locates the speed gate, the robot will then actuate the thrusters at full thrust to achieve the highest speed between the two gates.
Task 4 – Channel Navigation Test

This task is the most crucial task in the competition. If the boat is not able to navigate the channel, it cannot attempt any of the other tasks. Because of this, the team will implement a variety of ways to make sure the boat is able to complete the task successfully.

One solution is to go through the navigation channel using the Kinect’s depth sensor to find the closest set of buoys, use the camera to verify buoy color, and navigate through the barriers by keeping the center of the processed image between the closest obstacles.

Another solution which may prove to be effective is to square the robot off once it passes another set of obstacles. This will prevent situations where the robot will continuously switch between two states by seeing a close red buoy, rotating, seeing a close green buoy, and repeating.

A third solution which may be the most effective, but also the most complex, is using potential fields to navigate through the channel. This method of navigation assigns obstacles with “repulsive forces” or high values following a 3-dimensional Gaussian distribution and the destination with “attractive forces” or low values following a negative 3-dimensional Gaussian distribution, as pictured below. With these attractive and repulsive forces, the robot follows the path with the low values to the destination, which has the lowest values.

![Gaussian function for potential field navigation](image)

The implementation of this process would require the Kinect and the HD camera. First the Kinect depth sensor would find the closest buoys and use the value from the depth sensor and trigonometry to determine the buoy’s location in space relative to the robot. This will allow the robot to create a map with buoy locations and populate this map with the Gaussian distribution centered at the buoy’s predicted location. A Kalman filter will be required to ensure that the predicted location is close to the actual location with each movement step.
Second Set
Task 5 – Poker Chip

To complete this task, the rover will use a camera and perform image processing using color detection to follow along the marked perimeter and the Hough Transform to recognize the puck. When it recognizes the puck, it will pick it up and then return to the boat by the same route. Odometry will be incorporated if possible; it is not yet known if there is any feedback devices located on the rover or any places to install feedback devices. It may be beneficial to mark the boat with a colored marker so that once the robot locates the puck, it can just bring the puck back to the boat directly, taking care to avoid the edges of the platform so that it does not lose the puck or fall off the platform.

Task 6 - Jackpot

For this task, it is necessary to use a particular kind of ultrasonic sensor, known as a “fish finder sensor.” This sensor operates on the principle of SONAR and allows a user to locate schools of fish or, relevant to the competition, a large underwater object. This will allow the boat to locate the underwater buoy that corresponds to the correct button that will complete the task.

Task 7 - Cheater’s Hand

The cheater’s hand appears to be one of the simpler tasks to accomplish for the group. In this task, the robot will use a camera to detect which card has the blue tape outline on it using color and edge detection. Once the card is located, the robot will simply shoot the target with the water jet.

Task 8 – Hot Suit

The hot suit will be slightly complicated to complete. Detecting the hot suit with an infrared sensor will be simple, but transmitting the coordinates back will be challenging. The AUVSI rules require a particular format for this transmission. This format is documented in Appendix A.

Third Set

Task 9 – Card Exchange

This challenge will be more difficult than the others, which may be why it is a bonus competition. Not only will the robot have to detect the suit of a card, but it will also have to determine the value of a card (from 2 – Ace). In addition to this, the robot will need to know the dealer’s hand, its current hand, and what poker hands can be made from its hand that will beat the dealer hand. There are methods of shape recognition that can be done with OpenCV. The group will try to utilize this to complete the task, but as it is an optional challenge, it is very low on the list of priorities for the team.

Task 10 - Return to Dock

There are several ways for this task to be accomplished. One way is to have the robot rotate in place upon completion of the final task until it sees the blue buoy. The boat then navigates to the blue
buoy, locates the channel used in task 4, and navigates the channel as it did in task 4, this time keeping the red buoys on the starboard side.

Another alternative would be for the boat to log the GPS coordinates of the blue buoy once it completes task 4. It can then use these GPS coordinates to navigate back to the blue buoy when the robot attempts to return to the dock. It will then perform the same navigation as stated above by keeping the red buoys of the channel on the starboard side until it returns to the dock.

**Summary**

The AUVSI Roboboat will be designed to complete the tasks described in the above sections. In addition to programming the boat to complete each task successfully, the group will also program it to be robust enough to operate in different weather conditions. The scope of the project includes converting a rover from a tethered connection to a wireless connection with the boat, improving the performance by converting the program from LabView to C++, and integrating the Kinect sensor with HD cameras for improved obstacle detection and recognition. The majority of the work will be directed at completing the first set of tasks, which is a concern following last year’s competition. After this is accomplished, the group only needs to program the boat to navigate to a challenge station and complete the task. Once the updated rule set is released, the group will be able to finalize the requirements for this year’s competition and begin work on design approaches to meet them. With the effort put into the project from this past semester as well as the solutions developed, the team feels confident that it will place very high in the competition.
References

2012 AUVSI RoboBoat Competition Rules
http://www.auvsifoundation.org/foundation/competitions/roboboat/

2012 Stevens ASV Journal Paper
http://stevens.edu/asv

Open CV
http://opencv.willowgarage.com/wiki/

SLAM

ROBOT BOOK
Introduction to Autonomous Mobile Robots – Siegwart and Nourbakhsh

WEBOTS
http://www.cyberbotics.com/

Components
http://www.dimensionengineering.com/products/sabertooth2x25
http://www.robotshop.com/oceanserver-os5000-us-compass.html
http://www.arduino.cc/en/Main/arduinoBoardUno

http://www.raytek.com/Raytek/en-r0/ProductsAndAccessories/InfraredPointSensors/CompactSeries/RaytekCM/
Appendices

Appendix A: Hot Suit Transmission Format

5.2.5 Network Communication Protocol

All messages are TCP messages (UDP will not be supported). The protocol is ASCII based and each command is terminated with a Linefeed (LF) character (ASCII value: 0xA). The protocol description below will use the value “\n” to represent the Linefeed characters. The following commands are available:

Ping command:

Used for: Testing connectivity with reporting service
Sender sends: “PING\n”
Service response: “PONG\n”

Report command:

Used for: Reporting the ‘hot’ target and its position
Sender sends: “REPORT;<university>:<target name>@<gps>\n”
Service response: “SUCCESS\n” or “FAIL\n” or “ERROR\n”

Meaning of response:  SUCCESS = reported accurately the hot target.
FAIL = reported the wrong target or GPS
ERROR = the request is malformed

Possible values for <targetname>: “HEART”, “DIAMOND”, “CLUB”, “SPADE”
Format of values for <university>: regex: [A-Z]{2…5} A two to five upper letter code to identify your university/team. Ex: URI, UCF, GTASL, VT, etc. Teams will need to come “register” their acronym the first time they schedule a time in the water.
Format for GPS positions (in WGS84 datum): “<LATITUDE>,<LONGITUDE>”

Examples:

ASV: “PING\n”
Service: “PONG\n”
ASV: “REPORT;AUVSI:CLUB@40.688888,-74.045111\n”
Service: “FAIL\n”
ASV: “REPORT;AUVSI:DIAMOND@36.802327,-76.191379\n”
Service: “SUCCESS\n”
ASV: “TIMELEFT\n”
Service: “ERROR\n”

Any other command issued will be answered with “ERROR\n”
Appendix B) Task Layout
### Appendix C) Scoring Sheet

#### Table 1: Static judging scoring sheet

<table>
<thead>
<tr>
<th>Subjective Measures</th>
<th>Max. Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utility of team website</td>
<td>50</td>
</tr>
<tr>
<td>Technical merit (from journal paper)</td>
<td>50</td>
</tr>
<tr>
<td>Written style (from journal paper)</td>
<td>50</td>
</tr>
<tr>
<td>Technical accomplishment (from static judging)</td>
<td>75</td>
</tr>
<tr>
<td>Craftsmanship (from static judging)</td>
<td>75</td>
</tr>
<tr>
<td>Team uniform (from static judging)</td>
<td>10</td>
</tr>
<tr>
<td>Video quality (from introduction video)</td>
<td>25</td>
</tr>
<tr>
<td>Discretionary static points (awarded after static judging)</td>
<td>40</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>375</strong></td>
</tr>
</tbody>
</table>

#### Table 2: Performance judging scoring sheet

<table>
<thead>
<tr>
<th>Performance Measures</th>
<th>Max. Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>See Table 3</td>
</tr>
<tr>
<td>Generate F pounds of thrust (thrust measurement lbs)</td>
<td>(F / weight) * 100</td>
</tr>
<tr>
<td>Pass through the starting gate</td>
<td>100</td>
</tr>
<tr>
<td>Navigate from Starting to the Speed gate in T seconds</td>
<td>250 - T</td>
</tr>
<tr>
<td>Enter navigation channel</td>
<td>50</td>
</tr>
<tr>
<td>Navigate through X buoy set in the channel</td>
<td>X x 50</td>
</tr>
<tr>
<td>Avoid N obstacles in the navigation channel</td>
<td>N x 100</td>
</tr>
<tr>
<td>Successful dock-deployment or go up the ramp at Poker chip station</td>
<td>1000</td>
</tr>
<tr>
<td>Return back to water with poker chip</td>
<td>2000</td>
</tr>
<tr>
<td>Jackpot station</td>
<td>1000</td>
</tr>
<tr>
<td>Cheater's Hand station</td>
<td>1000</td>
</tr>
<tr>
<td>Hot Suit station</td>
<td>1000</td>
</tr>
<tr>
<td>Make at least one beneficial card swap</td>
<td>500</td>
</tr>
<tr>
<td>For each additional beneficial card swap</td>
<td>100</td>
</tr>
<tr>
<td>Beat the dealer at 5-card draw</td>
<td>250</td>
</tr>
<tr>
<td>Be the only team/vehicle to attempt a challenge station (bonus) *</td>
<td>500</td>
</tr>
<tr>
<td>Return to dock</td>
<td>500</td>
</tr>
<tr>
<td><strong>Finish All Tasks with T minutes Left on Clock (whole + fractional)</strong></td>
<td>T x 100</td>
</tr>
</tbody>
</table>

* = If a team is the only team to have attempted a specific challenge station during a stage of the competition (qualification or final), they will be awarded 500 bonus points for that station irrespective of whether or not they succeeded.
Appendix D: Sample OpenCV Program

```c
#include <opencv/cvaux.h>
#include <opencv/highgui.h>
#include <opencv/cxcore.h>
#include <stdio.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <assert.h>
#include <math.h>
#include <float.h>
#include <limits.h>
#include <time.h>
#include <ctype.h>

int main(int argc, char* argv[]) {
    // Default capture size - 640x480
    CvSize size = cvSize(640,480);
    // Open capture device. 0 is /dev/video0, 1 is /dev/video1, etc.
    CvCapture* capture = cvCaptureFromCAM( 0 );
    if( !capture ) {
        fprintf( stderr, "ERROR: capture is NULL \n" );
        getchar();
        return -1;
    }
    // Create a window in which the captured images will be presented
    cvNamedWindow( "Camera", CV_WINDOW_AUTOSIZE );
    cvNamedWindow( "HSV", CV_WINDOW_AUTOSIZE );
    cvNamedWindow( "EdgeDetection", CV_WINDOW_AUTOSIZE );
    // Detect a red ball
    CvScalar hsv_min = cvScalar(0, 151, 133, 0);
    CvScalar hsv_max = cvScalar(180, 256, 256, 0);
    IplImage * hsv_frame    = cvCreateImage(size, IPL_DEPTH_8U, 3);
    IplImage* thresholded   = cvCreateImage(size, IPL_DEPTH_8U, 1);
    while(1) {
        // Get one frame
        IplImage* frame = cvQueryFrame( capture );
        if( !frame ) {
            fprintf( stderr, "ERROR: frame is null...\n" );
            getchar();
            break;
        }
        cvSmooth(frame,frame, CV_GAUSSIAN, 9, 9 );
        // Covert color space to HSV as it is much easier to filter colors in the HSV color-space.
        cvCvtColor(frame, hsv_frame, CV_BGR2HSV);
        // Filter out colors which are out of range.
        cvInRangeS(hsv_frame, hsv_min, hsv_max, thresholded);
        // Memory for hough circles
        CvMemStorage* storage = cvCreateMemStorage(0);
        // hough detector works better with some smoothing of the image
        cvSmooth(thresholded,thresholded, CV_GAUSSIAN, 9, 9 );
        CvMemStorage* storage = cvCreateMemStorage(0);
        // hough detector works better with some smoothing of the image
        cvSmooth(thresholded,thresholded, CV_GAUSSIAN, 9, 9 );
        CvSeq* circles = cvHoughCircles(thresholded,storage, CV_HOUGH_GRADIENT, 2,
                                        thresholded->height/4, 100, 50, 10, 400);
        for (int i = 0; i < circles->total; i++)
            ...

```


```c
{
    float* p = (float*)cvGetSeqElem( circles, i );
    printf("Ball! x=%.f y=%.f r=%.f\n",p[0],p[1],p[2]);
    cvCircle( frame, cvPoint(cvRound(p[0]),cvRound(p[1])),
              3, CV_RGB(0,255,0), -1, 8, 0 );
    cvCircle( frame, cvPoint(cvRound(p[0]),cvRound(p[1])),
              cvRound(p[2]), CV_RGB(255,0,0), 3, 8, 0 );
}

cvShowImage( "Camera", frame ); // Original stream with detected ball overlay
cvShowImage( "HSV", hsv_frame); // Original stream in the HSV color space
cvShowImage( "After Color Filtering", thresholded ); // The stream after color filtering

// Do not release the frame!
// If ESC key pressed, Key=0x10001B under OpenCV 0.9.7(linux version),
// remove higher bits using AND operator
if( (cvWaitKey(10) & 255) == 27 ) break;

// Release the capture device housekeeping
cvReleaseCapture( &capture);
cvDestroyWindow( "mywindow" );
return 0;
}
```