USNA 2012 Autonomous Surface Vessel

ENS. Trentt James, ENS Matt Minkoff, ENS William Queen
MIDN. Emily Hughes, and MIDN David Jackson

Advisors: Assoc. Prof. M. Feemster and LCDR Kirk Volland
105 Maryland Ave. (Stop 14A)
Annapolis, MD 21402
Email: feemster@usna.edu

***Abstract*—In this paper, the design of the U.S. Naval Academy's 2012 entry to the Association for Autonomous Unmanned Vehicle Systems International (AUVSI) autonomous surface vessel challenge is presented. We will specifically cover the vehicle's mechanical/electrical construction, and our approach for solving the various challenges posed by the competition sponsors.**

I. Introduction

The United States Naval Academy has designed and constructed an autonomous surface vessel (ASV) for entry into the 5th annual Autonomous Unmanned Vehicle Systems International (AUVSI) competition held in Virginia Beach, VA in June 2012. Specifically, the specification set forth in the rules of [1] specify that the delivered vehicle exhibit the following requirements:

- Must possess autonomous and remote operation.
- Must exhibit remote/local safety disconnect of power.
- Must occupy a volume less than 6ft. (length) x 3 ft. (width) x 3 ft. (height).
- Must weigh less than 140 lbs.

The above list serves as the primary design specifications for the physical dimensions and basic operation of the vessel; however, the additional challenges offered by the 2012 competition impose the following additional design considerations:

- Must navigate a designated COREG-based buoy channel
- Must locate signs depicting typical playing cards
- Must launch a stream of water at select targets
- Must be capable of an amphibious landing to reclaim a hockey puck size object
- Must be able to depress a plunger type switch

The remainder of the paper presents our 2012 vehicle as designed according to the above primary and sub-objectives. In Section II, the vehicle's mechanical/electrical design is presented. In Section III, we discuss our proposed approach to autonomous navigation of a buoy channel. In Section IV, the image processing algorithm is discussed. Section V briefly describes our approach to select challenges of the competition.

II. The 2012 Vehicle Design

The current version of the USNA 2012 autonomous surface vessel was motivated from two issues: 1) problems experienced in the previous year's competition performance (magnetic interference and the accuracy of the global position measurements) and 2) the newly introduced challenges. With regards to 1, the 2011 vehicle experienced intermittent difficulty in tracking a commanded heading. The hard iron errors were addressed through proper calibration (offsetting) of the magnetic compass as depicted in Figure 1. Specifically, the vehicle was rotated slowly in circles while recording the raw field measurements from the MicroMag compass. Ideally, a circle of field readings in the x/y plane centered at the origin is desired; therefore, offset values were calculated to add/subtract to the measured field readings to promote this.

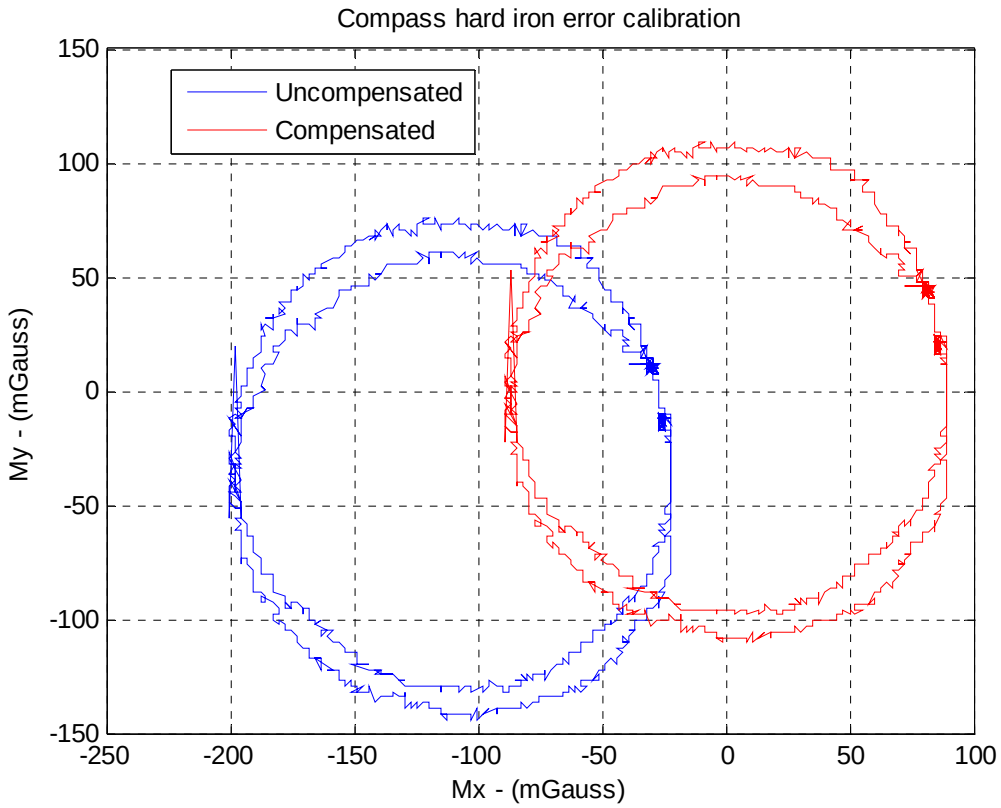


Figure 1: Hard iron errors in compass

Even with hard iron calibration effects addressed, the vehicle still experienced intermittent heading tracking performance. It was concluded that the magnetic compass was placed too close to the internal laptop's fan. While underway, the fan would disrupt the magnetic field causing erroneous heading measurements. Therefore, the primary change for 2012 is to isolate the magnetic compass by placing it in its own pelican case away from perceived magnetic disturbances as shown in Figure 2 (the compass is housed in the orange pelican case)

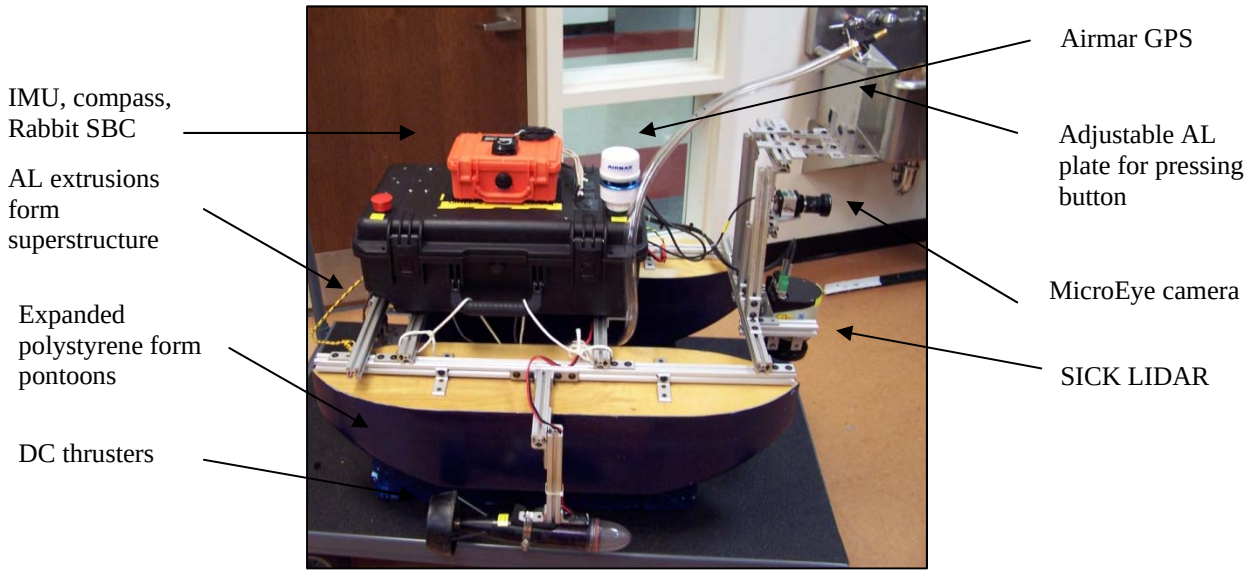


Figure 2: 2012 USNA ASV

Furthermore, concern was expressed over the precision of our current global position measurement sensor (the Airmar PB200). At competition, the boat's measured position would wander as much as $12(m)$ though the boat was stationary. This could possibly be due to poor alignment of the satellites at that time; however, the team decided to improve the our position measurement system with the addition of MicroStrain's 3DM-GX3® -45 inertial measurement unit (IMU) show in Figure 3

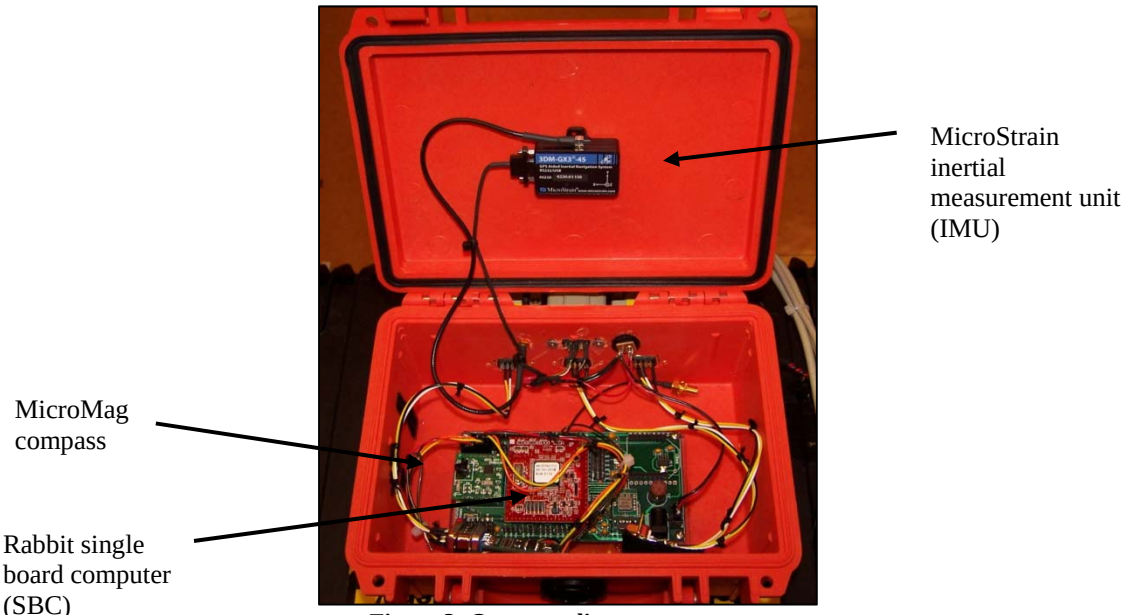


Figure 3: Orange pelican case contents

Specifically, the MicroStrain IMU contains a GPS sensor, 3 axis gyroscope, 3 axis accelerometers and implements a Kalman filter to fuse these measurements to provide an improved position measurement of the vehicle at a rate of up to 100 Hz. An experiment was conducted to verify the improved performance. The vehicle was fixed at a stationary location. Approximately 1500 position measurements were collected from both the Airmar GPS and the MicroStrain IMU. A histogram of the measured positions is shown in Figure 5. As observed in Figure 5, the Kalman filter is able to reduce the standard deviation of the position measurements from approximately $1.2(m)$ to $0.6(m)$. Though this reduction may not seem significant at a large scale, this reduction could prove significant when navigating the narrow buoy chain.

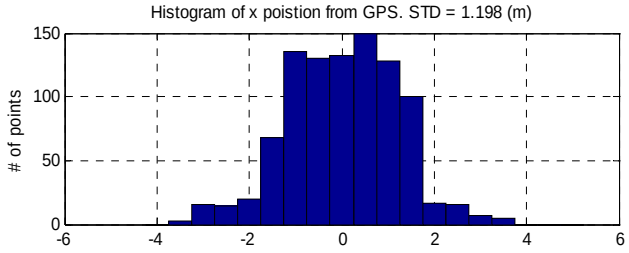


Figure 4: Histogram of x position measurements from Airmar GPS

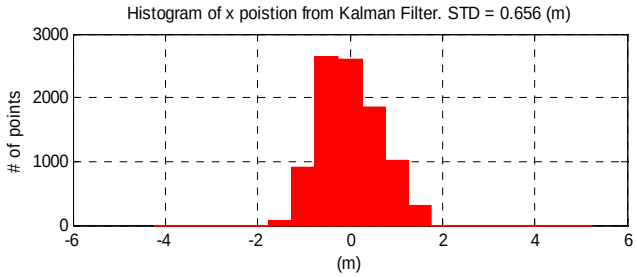


Figure 5: Position measurements from MicroStrain IMU sensor

The calculated (x, y) position of the vehicle from the Airmar GPS sensor and that of the MicroStrain IMU sensor can be observed in the following figure

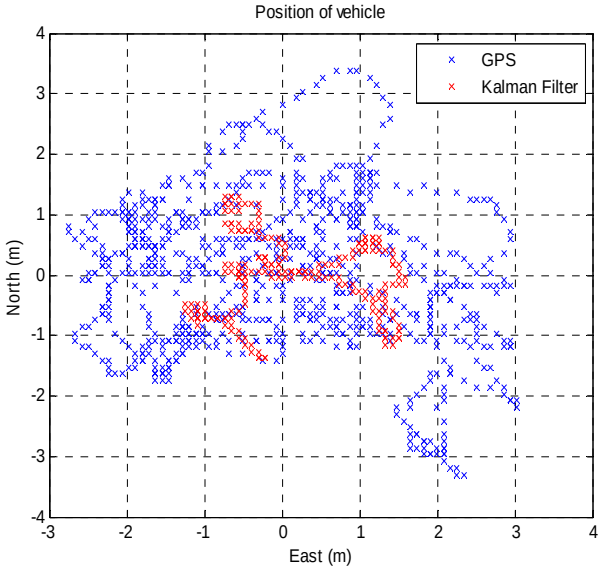


Figure 6: (x,y) position of ASV from (a) Airmar and (b) MicroStrain

From Figure 6, mapping efforts using only the Airmar (blue) would be strenuous at best due to the relatively large wandering of the position track (up to 3.5 m in the North direction).

A summary of the vehicle mechanical parameters are provided in the following table.

Length	Beam	Height	Draft	Weight
3.5 ft. (42 in)	2.6 ft. (31 in)	2.2 ft. (26 in)	0.8 ft. (9.5 in)	70 lbs.

Table 1: Summary of ASV mechanical characteristics

The vehicle pontoons were constructed from a non-absorbent polystyrene foam (from Dow Industries) which is commonly utilized for home insulation. The foam was covered with a thin fiber glass mesh and painted with a thin wood decking for mounting purposes. The superstructure that joins the two pontoons and serves as the skeleton backbone for mounting is 80-20 aluminum extrusion. Two pelican cases houses the all the required electronics. The SICK LMS111 laser range sensor was utilized to detect obstacle within a 270 (deg) field of view (± 135 (deg) on either side of the bow) out to a range of 16 (m).

In order to aid in the buoy channel navigation, a uEye USB camera (shown in Figure 7) was mounted in-line with the LMS111 to provide visual feedback information.

Figure 7: μ Eye USB camera

In order to distribute the processing load, a Beagle board processor was added to the ASV running the LINUX operating system. The acquired camera image is captured by the Beagle board and processed using Open CV libraries (more detail is provided below on the image processing algorithm).

During each control cycle, the IMU, compass, LIDAR and camera are polled for range measurements/images respectively. Two DC thrusters from VideoRay served as actuators for the vessel. Each thruster produces approximately 10 (lbf) at 24 (V). An alternative, more powerful BLDC thruster set from VideoRay were considered and acquired, but were not incorporated due to the readily available dual channel, DC power amplifiers by RoboteQ. These power amplifiers have proven reliable in numerous projects and have an established interface.

The electrical layout for the vehicle is shown in Figure 8.

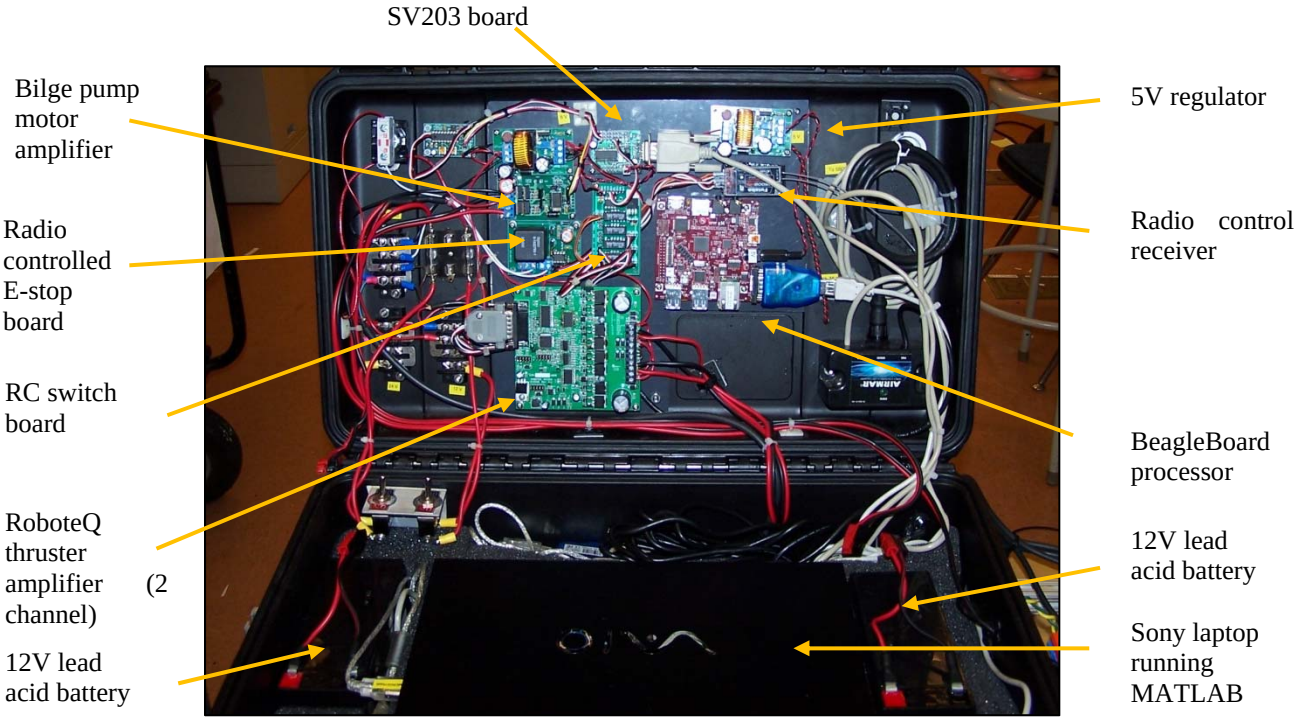


Figure 8: The electrical layout

Two 12 (V), 8 (Ah) batteries provide a 12 and 24 (V) bus for the electronics (two 5/6 V regulators are utilized for lower voltage electronics). The ability to switch between autonomous and remote controlled operation is achieved through a custom designed printed circuit board (denoted as RC switch board in Figure 8). This board receives an RC signal via the Futaba receiver that disconnects primary power to the RoboteQ power amplifier (in addition the

local E-stop button on the top of the pelican case). The low-level heading control is implemented on below custom navigation board (see Figure 9).

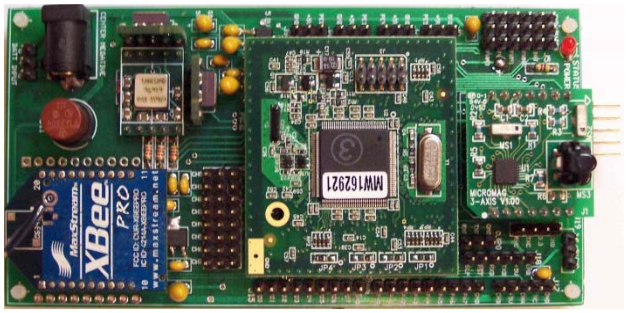


Figure 9: WSE navigation board

The above board was designed, developed, and constructed by the Technical Support Division of the Systems Engineering Department specifically for use in mobile vehicle applications (including ground, aerial, and underwater vehicles). The board is centered around the Rabbit 3000 microprocessor and is equipped with a MicroMag magnetic compass, three axis rate gyros and accelerometers, a Trimble global position system, and a MaxStream wireless serial modem. The Rabbit 3000 is programmed in Dynamic C to receive desired heading commands from the high-level navigation software (implemented in MATLAB on the Sony laptop) and then implement a low-level PID heading controller at a frequency of 35 Hz which generates the commanded thrust. The thrust command is sent serially to an SV203 board that then generates the corresponding RC pulse train to be received by the RoboteQ power amplifier.

The high level motion planning (such as the buoy channel navigation algorithm of Section III) is implemented (at an approximate frequency of 1.0 Hz) within the MATLAB software environment running on the Sony Viao laptop under the Windows XP operating system. The high-level software accepts the following feedback measurements: 1) global *(x,y)* vessel location from the MicroStrain IMU, 2) the ASV true heading from the MicroMag compass located on the Rabbit SBC, 3) range to obstacles or buoys from the SICK LMS111 LIDAR, and 4) the relative bearing of red/green buoys from the processed image from the BeagleBoard. Given these four feedback quantities, the output of the high-level navigation functions which are to be sent to the low-level navigation board are: 1) a desired heading and 2) a desired forward/reverse thrust bias.

III. The Buoy Channel Navigation

To determine the required heading to stay in the buoy channel, the LIDAR initially scans the area in front of the ASV and returns ranges of objects within a 270° field of view with respect to the bow of the ASV out to maximum scanning range of 16 meters of the ASV's current location. The returns from the LIDAR are sent to a clustering algorithm that determines which cluster of returns possibly represent buoy locations (we exclude collections of returns that are obviously too large to represent buoys). The location of these buoys is then sent to the designed buoy chain function that determines the desired heading of the ASV that will enable the vessel to navigate the buoy channel. This function receives the following information: 1) the current position and heading of the ASV, as well as 2) the global frame *(x,y)* coordinates of all detected buoys from LIDAR and cluster detection algorithm.

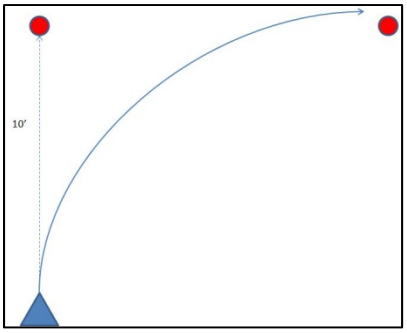


Figure 10: An approach from outside the channel

The channel navigation function first measures and stores the relative angle and distance from the ASV to each of the buoys. By calculating the standard deviation of these angles, the function then decides if the ASV is either inside or outside the buoy channel. If it determines the ASV is outside the buoy channel, it sets the desired heading in the direction of the closest buoy. Once the ASV gets within 10 feet of this buoy, it uses the angles to determine which side, left or right, has the most buoys, and gradually begins turning in the direction of the largest number of buoys so that as it enters the channel (note: if vision is successfully integrated and the color of the buoys are now known, then we will know the direction of the channel), its heading will be nearly parallel to that of the channel, allowing the LIDAR to detect the largest possible amount of buoys. An example of this path is shown in Figure 10. Before it has fully entered the buoy channel, it will detect that it is close enough and transition to a state that navigates within the buoy channel, which will guide the ASV on the appropriate course. In addition, this state will be used to find the entrance to the buoy channel (*i.e.*, locating the direction with largest number of buoys), and therefore the direction the ASV is intended to travel. Furthermore, this approach would also be used if there is a disturbance (such as wind) and the ASV ends up leaving the channel and needs to relocate it, in which case, for simplicity, the default assumption is that the direction with the most buoys is the appropriate one. However, for a scenario in which the general direction of the buoy channel is known, such as in the competition, the overall desired heading from the beginning to the end of the buoy channel (*i.e.*, the general bearing of the channel) will be calculated and made available for comparison to. With the general bearing of the channel known, instead of going in the direction of the most buoys, the ASV will go to whichever direction takes it closer to the overall desired heading, so even if it gets off course temporarily, when it corrects itself it will continue in the right direction instead of going back to the start of the buoy channel (we realize that if the buoy channel makes a “horse-shoe” turn then this approach is not viable).

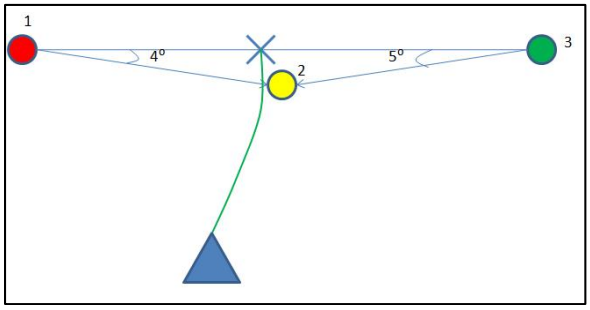


Figure 11: An example scenario within buoy chain navigation

Once the function determines the ASV is in the channel, it uses a different method to determine the desired heading. First, it finds the three closest buoys and compares the angles between them. In the competition, there will be decoy buoys in the middle of the channel that will look exactly like the real buoys to the LIDAR. An example of this situation is shown in Figure 11. If it determines that the middle one is between the other two, within 5 degrees of the line between them, it determines that the middle one is one of the decoy buoys and therefore uses the other two to determine the required heading. In the example in figure 4, for example, the function detects that buoy 2, the yellow one, is between buoys 1 and 3. Because the angle from buoy 1 to buoy 2 is less than 5 degrees off of the line between buoys 1 and 3, it identifies buoy 2 as a decoy. The function would then compute a heading to the point directly between buoys 1 and 3, as shown by the ‘x’.

If the function determines that the middle buoy is not directly between the other two, the function moves onto the next test. By comparing the distances to each buoy, the function determines which buoy is the outlier, meaning it is either significantly closer to or farther away from the ASV. Since the other two buoys are a similar distance from the ASV, this indicates that they are the corresponding red and green buoys, and therefore the ASV must travel

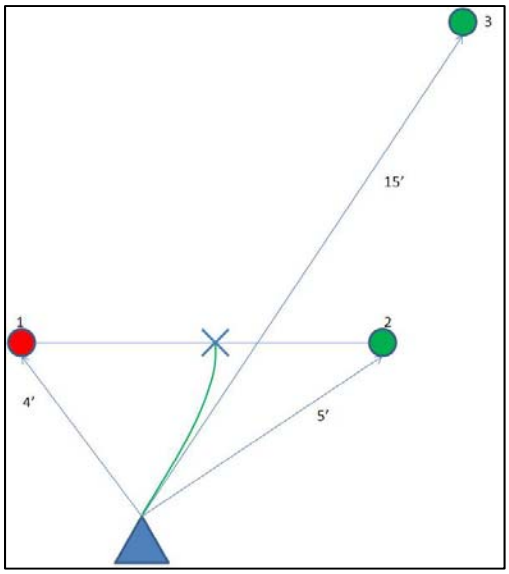


Figure 12: Another buoy channel encounter

directly between them. This aspect of the algorithm is demonstrated in Figure 12. In this example, the distances to buoys 1 and 2 are very similar, only 1 foot difference. However, the distance to buoy 3 is 10 to 11 feet larger, so buoy 3 is identified as the outlier. This indicates that the appropriate course through the channel should be between buoys 1 and 2, and the heading is calculated accordingly.

To approach this point, instead of heading directly at it, the function calculates a heading that will take the ASV towards the point while gradually changing the heading so when it is between the two buoys, the heading of the ASV is perpendicular to the line between them. This allows the LIDAR to scan the maximum amount of the buoy channel, making the next heading calculation more accurate. The heading that the ASV would follow is shown by the green line in Figure 11 and Figure 12. While this is more complicated than simply following a straight line to the desired point, it has proven to be much more effective at locating the entire buoy channel, which makes the algorithm that much more reliable.

A. Simulation Results

The algorithm was tested using a simulation (which include representative ship dynamics) within MATLAB that sends to the buoy channel navigation function the coordinates of the buoys in a channel. Initially the function had several issues, especially with determining if the ASV was in the channel or not. When there is a bend in the channel, because the ASV is perpendicular to the straight part of the channel before entering the turn, it would see more of one side of the channel than the other. When the standard deviation of the angles to the buoys was computed, it would return a value indicating it was outside the channel, and act accordingly. This resulted in the algorithm plotting a course that would take the ASV outside of the channel. By tuning the value of the standard deviation that indicates if the ASV is in the channel or not, this problem was largely eliminated. Repeated testing showed that the optimal standard deviation was .5 radians, or approximately 29 degrees. If the standard deviation is larger than this, the ASV is outside the channel; otherwise it is in the channel. This improved the accuracy of the algorithm, but it was still not ideal because occasionally even this limit was exceeded and the course calculated assuming the ASV was outside the channel was very different from what it would be if it was in the channel.

Figure 13: Simulated buoy channel navigation

This problem was solved by increasing the distance from the closest buoy when the ASV begins turning into the channel if it determines it is not in the channel. By increasing this to ten feet as previously described, this caused the ASV to begin turning almost immediately when it was actually in the channel, because the channel is only five to six feet wide. Because it starts turning so soon, the standard deviation would quickly decrease as the ASV turned with the bend in the channel and it could detect more buoys on the opposite side of the channel. At this point, it is not far enough off course to cause it to leave the buoy channel, and it quickly corrects this error and continues on an appropriate course within the buoy channel. These changes also caused the ASV to enter the channel more gradually when coming in from the outside, which increased the accuracy of this aspect of the algorithm as well.

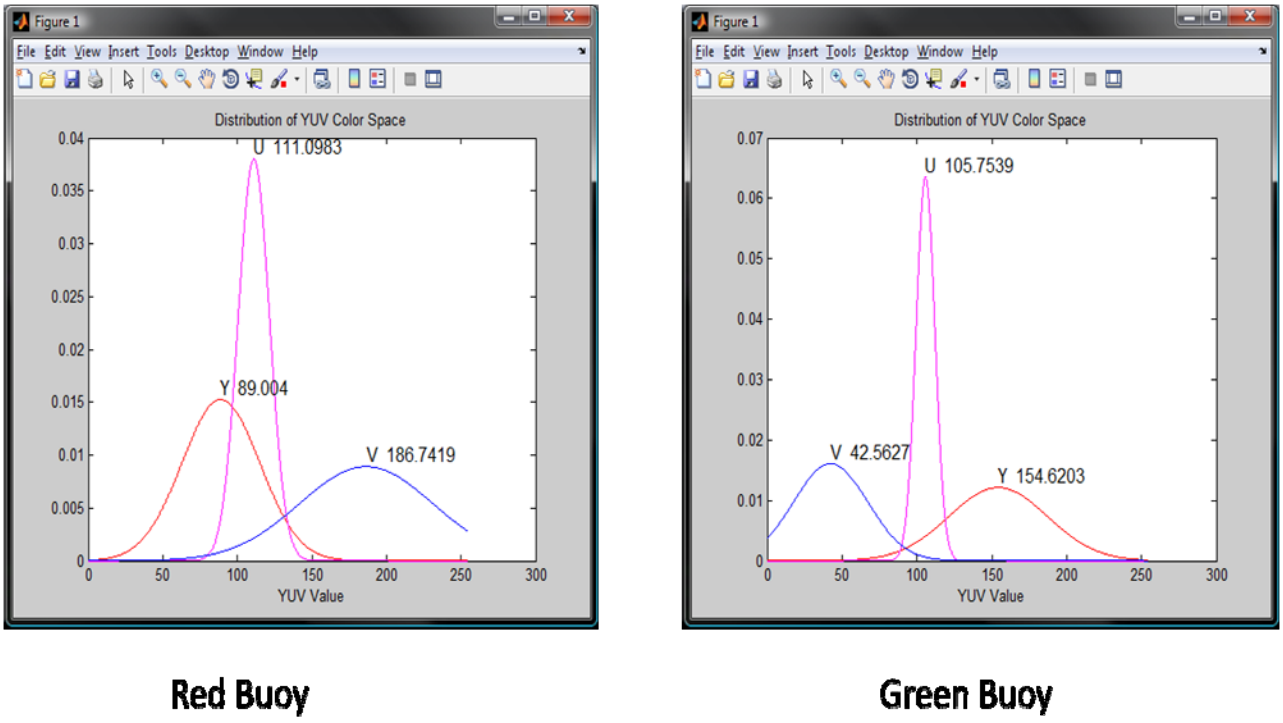
After some adjustments and additions to the algorithm, it was able to accurately return the necessary headings to guide the ASV through the channel. In the same way the function will be implemented with the ASV, the function constantly recalculated the desired heading as the simulation moved the ASV through the channel. The final results of the simulations are shown in Figure 13. This will be tested further on the water over the next month as the ASV program is prepared for the competition in June, and necessary adjustments will be made to the algorithm. The only problem with the algorithm at this point is that it is inconsistent when entering the buoy channel from a position away from the actual beginning of the channel. It will sometimes go through the channel without turning into it, so it simply passes out the far side of the channel. While this is an issue that needs to be addressed in the future, for the competition, we will be starting at the entrance to the channel, and from this starting point the function is very reliable.

IV. Image Processing

The primary objective of the vision system on the ASV is to locate and determine the color of buoys through the utilization of different color scales. Other than RGB, different color scales give different information that may be pertinent to classifying a buoy as red or green. For instance, we converted RGB into the YUV color space because there is such a great distinction between red and green buoys in the chroma channels (UV). In comparison, we use the Mahalonobis distance to determine colors of interest. To discriminate red from green, finding Mahalonobis distances of specific color channels worked very well. When looking at a buoy, it contains Y-U-V channels in a 3-D color map. For example, each pixel on an image of a red buoy contains values from 0-255. By taking the averages of all the pixels for all three YUV channels, we are then able to take those values and compare them to a defined set of values for that buoy. If the values are very small, we can conclude that there is a high probability that it is a red

buoy. However, as the value gets larger, it falls farther away from actually being a red buoy. Therefore, the chances it being a red buoy are decreased. The same process for Mahalonobis distance can be said for green as well when you compare the pixel averages to a set of defined YUV values of green.

Below are two distributions of the YUV channels of a red and green buoy. The two buoys seem to be alike and have similar U values, but the main difference lies in the V channel where can separate red from green. The Y channel can be referred to as the luminance channel where it describes the brightness of the image. Red and green buoys both behave differently by reflecting and absorbing rays of light under different types of light. Because of this, the Y channel can mostly be ignored.



Red Buoy Green Buoy

Figure 14: Distributions in YUV space for (a) red and (b) green buoys

In order to calculate these differences, there is a distinct vision processing pipeline that the ASV goes through. At the same time, before we process an image, we first need a processor that will allow us to reach our goals of determining red and green buoys. For this case, we selected the BeagleBoard xM. It uses an ARM processor for the image processing at competition. It contains 1GHz CPU speed and hardware acceleration for Single Instruction Multiple Data (SIMD). The BeagleBoard xM is Linux based with Angström distribution. When we first received the BeagleBoard xM, a package manager program was needed along with a GNU Compiler Collection (GCC). After these initial installs, our core tools needed were OpenCV (Open Computer Vision) along with its libraries for blob detection. Finally, specifically for the ARM processor, we achieved the uEye USB camera's Software Development Kit from the manufacturer. Upon completion of initializing the BeagleBoard xM, we were then able to use the camera by turning on its auto modes (shutter speed, brightness, etc…) and capturing images into program memory. The following functional block diagram explains the connections from the camera to the laptop inside the ASV.

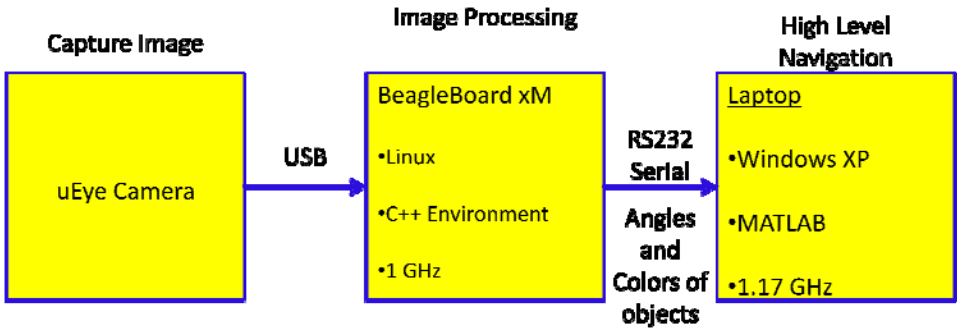


Figure 15: Flow chart for image processing to laptop

The flow chart of Figure 16 illustrates the tasks executed by the BeagleBoard on each newly acquired image.

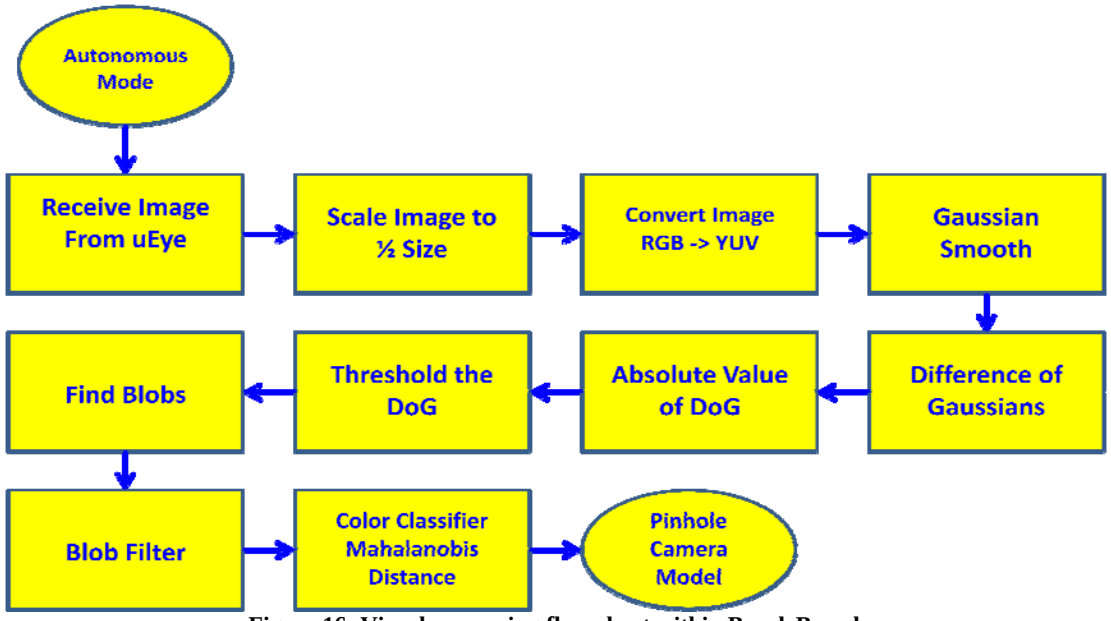


Figure 16: Visual processing flow chart within BeagleBoard

Specifically, we receive an image through the uEye camera and scale it to half its size. We scale the image because of clutter and data that is unwanted in the background of the buoys, such as, trees, grass, walls, signs, etc… As explained previously, we then convert the image from RGB to YUV and separate the 3 channels from each other. Gaussian smoothing is then applied to each of the channels which allow the images to be "blurred" to remove detail and noise. It can be compared to the mean filter, but it uses different kernel sizes that represent the shape of a Gaussian (bell-shaped) curve. By setting two defined kernel sizes, one larger than the other, we can subtract the two Gaussian kernels. From there, we threshold and take the absolute value of the difference. From the OpenCV blob library, we can locate blobs that we specifically care for. In our case, we want circular blobs so it would be in our best interest to filter the blobs with specifications that include circle-like properties. Finally, we incorporate our color classifier: the Mahalonobis distance. After this is complete, we can take a quick look at the pinhole camera model as depicted in the example image of Figure 17

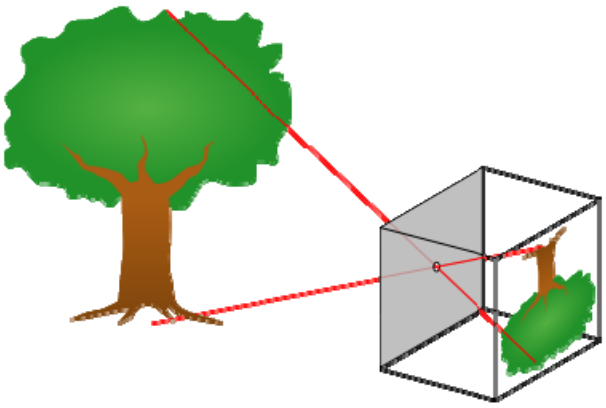


Figure 17: Pin-hole model example

In comparison to the above picture, the image has a field of view (in degrees) that it can actually see. The uEye camera, with its lens, can see a total of 51.4 degrees. And as stated before, we scale our image to half its size so the width of the image contains 378 pixels rather than 756. Therefore, an image contains 51.4 degrees for every 378 pixels. Determining location of the buoy (whether it's on the right or left side of the boat) can be described by locating its angle from the center line of the image, -25.7 degrees to the left and +25.7 degrees to the right. By multiplying the equation [(51.4^o/378 pixels) – 25.83] by the x position buoy's center, we can achieve the direction (left or right) of the buoy. This data can then be transmitted serially to the laptop along with the color. As stated before, this completely frees the laptop from large chunks of information and allows it to receive data rather than process it.

V. Competition Challenges

One of the most difficult challenges posed by the competition this year is the amphibious landing of a craft to recover a hockey sized object. In Keegan *et al.* [2], an amphibious vehicle (termed the AquaMonkey) was designed and constructed for the purposes of: 1) navigating land, 2) navigating on the water, and 3) climbing magnetic surfaces (such as select ship hulls).

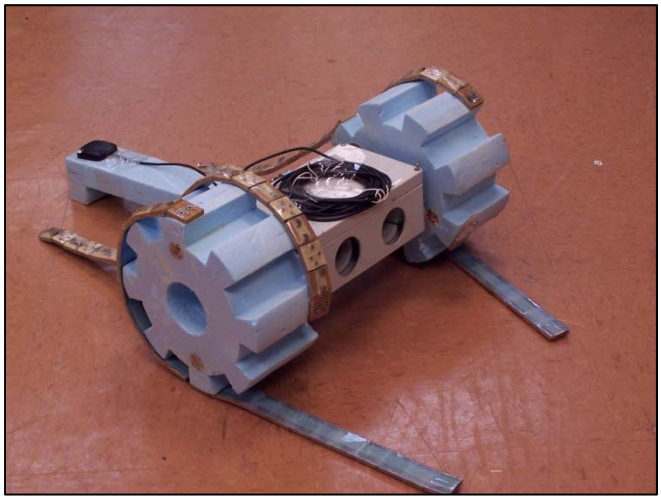


Figure 18: USNA's Aqua Monkey

The vehicle of Figure 18 is equipped with a GPS sensor, 3 axis gyro/accelerometer sensors, MircoMag compass, two ultrasonic range sensors, and a wireless serial communication link. Therefore, we are currently in the process of

modifying the Aqua Monkey by removing the magnet tentacles and sonar sensors and adding a USB camera and a BeagleBoard for image processing (for locating the object). The front of the remote vehicle will be covered with the appropriate Velcro to secure the object. Since the Aqua Monkey has its own GPS sensor, the ASV will broadcast its location to promote re-docking of the craft. Securing the amphibious vehicle to the ASV is being done through high strength neodymium magnets to a ferrous landing point (the Aqua Monkey produces enough force to detach the vehicle yet drag forces are not enough to pull the vehicle loose through normal operation).

In addition, a submersible pump has been installed to provide water ejection capabilities to address the cheater's hand challenge. Furthermore, an adjustable plate has been installed on the front of the vehicle to depress the buttons at the jackpot station.

CONCLUSION

Our primary focus this year has been to better navigate the buoy channel autonomously. LCDR Volland and Trentt James have supported this effort through the inclusion of a robust image processing system. We are currently underway with field trials on the Severn River. We suggest the reader visit our website at: https://sites.google.com/site/usnaasv/home for our most recent updates on our progress.

ACKNOWLEDGMENT

The 2012 USNA ASV team was funded for equipment and travel by a PEO Integrated Warfare System grant. In addition, the team would like to express their extreme appreciation to the Weapon and Systems Engineering technical support staff (Norm Tyson, Joe Bradshaw, Keith Groch, and Judy Raymond) for their unwavering support and to Tom Price of the U.S. Naval Academy machine shop. The team would also like to thank Professor Brad Bishop for loan (probably donation) of the Aqua Monkey. In addition, we would like to thank MicroStrain for aiding with the purchase of the inertial measurement unit.

REFERENCES

[1] AUVSI competition rules at http://www.auvsifoundation.org/AUVSI/Foundation/Competitions/RoboBoat/Default.aspx, last visited, June 6th, 2012.

[2] Keegan, C.M.; Bishop, B.E.; , "AquaMonkey: A novel multi-mode robotic vehicle," *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on* , vol., no., pp.2557-2558, Oct. 29 2007-Nov. 2 2007.