Design and Control of an Autonomous Underwater Vehicle for the RoboSub Competition

Coline Ramée, Raphaël Gautier, Pierre Valdez, Samuel Seifert,
Samuel Dubin, Jesse Chen, Henry Chen, Mary Catherine Martin,
Kevyn Tran, Sarah Panitz, Jared Mehnert, Michael Keller, Daniel Cooksey

Georgia Institute of Technology, Atlanta

***Abstract*— An Autonomous Underwater Vehicle (AUV) is under development at the Aerospace Systems Design Lab (ASDL) since September 2015 to compete for the first time in the RoboSub competition. The *Why yes, that is PVC* team is a subgroup of the ASDL Marine Robotics Team (MRT) specialized in the design of autonomous maritime vehicles. While the MRT conducts research sponsored by the Navy's NEEC program in the field of vehicles collaboration and communications-based path planning, it also attaches great value to student competitions such as RoboBoat, RobotX, and RoboSub for the new challenges they rise as well as the team spirit that they contribute to strengthen. After introducing the context of the AUV development, the present paper presents an overview of the adopted design strategy and then dives more into details of the vehicle design at the components level. The resulting design is the outcome of numerous trade-offs performed to tackle challenges raised by the limited human and material resources, the lack of any pre-existing hardware platform and the limited initial knowledge of the team in the field of AUV design. Emphasis was put on designing a long-lasting, easily maintainable and expandable platform to be used in the years to come.**

I. Introduction

The 19th International RoboSub competition will be held in July in the TRANSDEC pool in San Diego. For the first time since 2008 a team of students from Georgia Tech decided to compete. The team is composed of graduate and undergraduate students majoring in Aerospace Engineering, Mechanical Engineering or Computer Science.

Since September 2015 the team has been working on designing and manufacturing the AUV Suby McSubface, as well as developing a simulation environment to implement and test the control and autonomy algorithms.

As it is the first year the team participates in the competition the design emphasis was put on the control and navigation rather than on the interaction with the environment, and on the adaptability of the vehicle for future competitions and research.

II. Design Strategy

A. Requirements and constraints analysis

An analysis of the different missions over the past 3 years showed that the RoboSub competitions rules remain similar over the years, hence the vehicle requirements could be derived initially from the 2015 competition rules, as the 2016 rules were not available at the beginning of the school year. The vehicle must navigate at a maximum depth of 16 feet for 15 minutes which means that it should have an Ingress Protection rating of IP68. The tasks require precise station keeping and maneuverability which calls for a stable vehicle with six degrees of freedom controllability, i.e. a fully actuated and stable vehicle. In addition, even if the rules are similar one year from another, there are still changes, which means that the vehicles needs to be adaptable to additional systems. This modularity requirement would also enable the vehicle to be used for other research purposes.

The first set of design constraints came from manufacturing. The ASDL owns a 3D printer and a laser cutter, and Georgia Tech is well equipped with rapid prototyping equipment thanks to its Invention Studio (water jet, CNC mill …) and Aerospace Machine Shop (metal cutting band saw). The team members learned to use the machines and designed the vehicles in a way it could be manufactured in-house. The choice of material was dictated by the machines and the properties of the material. For example it is very easy to bond PVC to PVC rather than PVC to metal or to acrylic, and this is taken into account in the design.

The second type of constraints came from the fact that it was the first year since a very long time a Georgia Tech team would be competing in RoboSub. A lot of research had to be done to acquire the required knowledge since there was no previous team to inherit the knowledge from. The team was relatively small: it started in September with only 3 members. With members joining the team in the middle of the semester and members leaving due to internships or co-ops, there were at most 6 students working at the same time.

The team had limited funding. The vehicle manufacturing takes time, and there was no hardware platform to develop and test the autonomy algorithm.

B. Design approach

Since no previous hardware platform was available, developing a new vehicle was required. While starting from scratch was initially challenging, it also provided the team with complete freedom regarding design philosophy. The conclusion was quickly reached, that aiming to succeed at all tasks was not realistic on the team's first participation to the competition. Instead, focus was drawn on developing a long-lasting platform that could be reused for future editions of the RoboSub competition and more generally in the context of Maritime Research Projects conducted at ASDL.

This reasoning led to the design philosophy for the underwater vehicle: opt for a design simple enough for it to be manufactured and tested in two semesters – while leaving time to work on the autonomy algorithms in parallel – but flexible enough so that its capabilities could be easily expanded in the future. The goal for the first year was set at performing the competition tasks that do not require interaction with the environment but only vehicle control and navigation. However, the design should obviously allow to easily add the capabilities required to perform these more complex tasks in the future.

Seeking both simplicity and expandability, a two-part design was chosen, consisting of a pressure vessel used to safely store the core electronic components of the vehicle in a waterproof environment and a frame used to easily attach new sensors and actuators to the vehicle.

In order to ensure the expandability of the pressure vessel, it should present two main features: 1) the components placement should be organized and optimized to allow the addition of new modules inside the core and 2) electronic interfaces should be available to connect new exterior sensors and actuators to the core components. Also, in order for the vehicle to be maintainable and easy to debug in early test phases ease of access to the electronic components should be maximized while always guaranteeing the waterproof nature of the pressure vessel.

In order to ensure the expandability of the frame, it should present a convenient way to fasten any kind of actuator or sensor while leaving sufficient freedom regarding the spatial placement and orientation of the component. For instance, the thrusters are the first components being installed and their positioning should be minimally constrained by the frame but rather by requirements in terms of control capabilities.

Finally, in order to allow a parallel development of the control and autonomy algorithms along with the hardware development, resort to an in-house simulation environment was decided. The simulation environment should allow to test the performance of algorithms by simulating the underwater environment, the vehicle physics and the vehicle sensors. Parallel software development should allow to start the software test and validation phases as soon as the vehicle is manufactured.

III. Vehicle Design

A. Hardware

1) Pressure Vessel

The team first tried to use a Pelican Case, to avoid the manufacturing complexity and for ease of use (latching opening). The first test of the case was done at a low depth (1 feet) in the lab and no leak was detected. However once the test was performed at 16 feet a small leak was detected. A good lesson learned from this experience was to always test at the right depth/pressure, and that watertight is not enough at 16 feet. Rather than trying to fix the case we decided to change the design completely and to use a cylindrical hull.

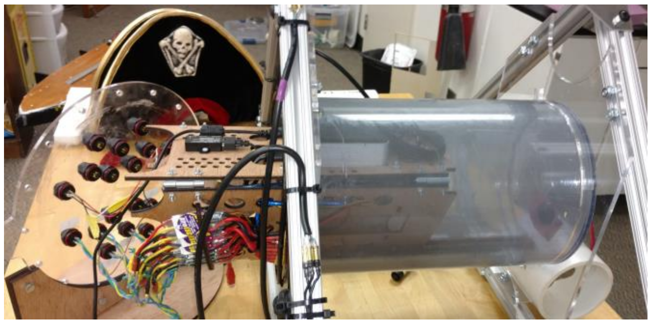

Figure 1: Pressure vessel, connector plate and electronic stack

The hull is made of a foot-long PVC pipe with an inside diameter of 8 inches. This is actually the same pipe that was used by Georgia Tech RoboSub Team in 2008 and it had been laying in a corner of the lab for 8 years. A clear PVC disk was cut using the water jet and glued using PVC cement to one end of the tube. The PVC is clear enough for the camera to see through. On the other end of the pipe a flange was made using two sheets of PVC. The flange has an O-Ring groove which was designed following the Parker O-Ring Handbook, and manufactured using a CNC mill. A connector panel is then screwed to the flange. All the wires that needs to go into the pressure vessel are connected to this acrylic panel.

The electronics must be attached to a platform rather than to the tube itself so that every elements can be easily inspected and replaced. In the first design all the electronics were attached to one platform which was attached to the connector panel. The design was improved by using 3 platforms attached by hinges that can be folded in a triangular shape. This was inspired by the Apple MacBook Pro design. In this more efficient configuration all the wires are routed in the middle, which produces a much more compact stack.

2) Frame

Early on a frame made of extruded aluminum was selected as it allows to easily attach or remove components anywhere on the structure. The most common type found is 80/20 1 inch. The rods were cut at the right dimensions using a metal cutting band saw. The core of the extruded aluminum piece is hollow and we first tried to do the assembly by using cubic connectors, but threading without a specialized machine is a tedious process, and the team decided to rather design and manufacture aluminum connector plates using the water jet.

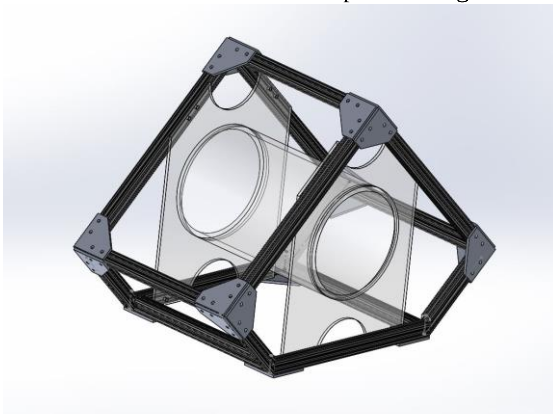

Figure 2: Frame

The frame evolved during the design process. It started as a cube rotated 45 degrees, this way the thrusters could be attached to the frame edges and be symmetric about the center of gravity. It was then modified to allow for a clear field of view for the camera and an easier handling by replacing the front and back face by a pentagon instead of a square. In the initial design the pressure vessel was attached to a platform attached to the frame, however on the first prototype the platform was bending and a lot of space was lost inside the frame. The team found a more efficient design by sliding the pressure vessel into two acrylic panels attached to the frame. The length of the frame was then modified to match the length of the tube.

3) *Thrusters*

To ensure control over the 6 degree of freedom seven thrusters are mounted on the vehicle structure. The positions and number of thrusters have been chosen to ensure controllability in the 6 degree of freedom without blocking the downward camera field of view. Three forward pointing thrusters control pitch and surge, two downward pointing thrusters control roll and heave (or depth), and two side thrusters control yaw and sway.

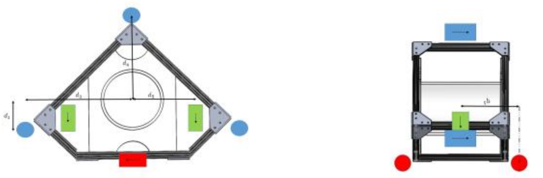

Figure 3: Thrusters position

With 7 thrusters the cost of most off-the-shelf thrusters was prohibitive. Hence it was decided to design and manufacture the thrusters at the lab.

Brushless motors naturally work under water since they consist of magnets and coiled wires, which means the wires are insulated.

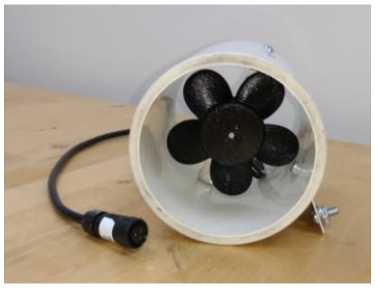

Figure 4: Thruster

The thrusters must be ducted for safety reasons.

No propeller was found on the market that fitted inside the duct and around the motor shaft, so it was decided to design and manufacture our own. An open source software called OpenProp was used to design the propellers. The resulting design were exported to SolidWorks and 3D printed. The propellers were then sanded and painted with epoxy paint.

B. *Electronics*

1) *Power supply*

The power source for the electronics and the thrusters are separated. Indeed the current drawn by the motors can change very quickly which can create fluctuations in the battery voltage. The computer is sensitive to these fluctuations. Moreover for safety reasons the computer should be on at all time when the motors are running, hence the computer battery should last longer than the thrusters battery.

2) *Computational Units*

The tasks to be completed in the competition rely heavily on vision, which is usually computationally expensive. The Intel NUC was chosen as the main on-board computer as it offers a good trade-off between size and computational power. Moreover since it is an actual desktop PC it is very easy to use compared to low-cost computer platforms such as RaspberryPi or BeagleBone. To compensate for the lack of digital Input/Output interface (IO) two Arduino Mega boards are used and communicate with the computer through serial USB.

3) *Sensors*

The vehicle relies mainly on three types of sensors to navigate: an Inertial Measurement Unit (IMU), two cameras, and a pressure sensor.

The IMU is a Microstrain GX3. It is a high-performance Attitude Heading Reference System that has been used with success on other vehicles in the lab.

The cameras are Sony PSEye webcams. They provide a sufficiently good resolution with a wide enough field of view for a very low cost. High definition cameras are not desirable as they increase the need for processing power. Code Laboratories driver and SDK are used to make the webcams compatible with the environment. One camera is pointing forward and the other one downward.

The pressure sensor is used to estimate the depth of the vehicle. A pressure sensor breakout board was used. To make it waterproof the board (except the pressure sensor itself) was potted in hot glue and fixed in a small acrylic enclosure.

Additional sensors such as a thermometer and a "rain" sensor are used to monitor the inside of the hull and make sure it is not overheating or leaking. A reed switch is used to activate the kill switch from outside the hull.

4) *Connectors*

Wires need to go in and out of the hull to power the motors and receive data from the pressure sensor and the camera. During tests the vehicle is tethered and it is monitored through Ethernet. The team decided to use connectors rather than potting the cable through the connector plate with epoxy to make the design more modular and each component easy to change.

5) *Actuators*

The actuator systems consists of one marker droppers, two torpedo launchers, and an active grabber mechanical system. The actuator systems are powered by a pneumatic system. Air supply from a 3000 psi paintball air tank is regulated down to hundreds of psi using off-the-shelf paintball regulators and on/off valves. The air is distributed via a manifold to 5 lines, one for the marker dropper, one for each torpedo launcher, and two for the active grabber. Switches are used to drive the air to one-way and two-way piston cylinders for each actuator system. This allows each actuator system to be controlled independently, and apply further pressure regulation for the systems that need less or more pressure. Pressure regulation was done carefully for each actuator system to make sure pressure losses from tubing reductions/extensions and fittings were accounted for.

C. *Software*

1) *AAVS*

The Adept Autonomous Vehicle Simulation (AAVS) is a simulation environment developed in-house at ASDL. It is coded in the C# programming language and runs on the Microsoft Windows operating system. AAVS is used to both simulate and autonomously control different classes of maritime surface and underwater vehicles (RoboBoat, RobotX and RoboSub).

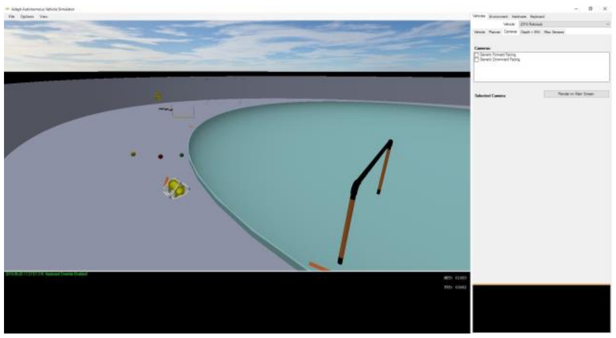
Figure 5: Simulation Environment

In simulation mode, AAVS can be run on any desktop computer to test the performance of autonomy and control algorithms. In this mode, the environment, the vehicle physics and the vehicle sensors are simulated and used as inputs to the control algorithms. The simulated vehicle moves about and receives feedback from a virtual world.

Instead of receiving data from simulated sensors, AAVS can be configured to receive feedback from actual hardware sensors. In this mode, AAVS is set up on the vehicle computer and the same autonomy and control algorithms developed and tested in the simulation mode can be applied to control the actual vehicle in the real maritime environment.

The resort to the C# programming language and the Microsoft Windows development platform was decided in order to minimize the learning curve that potential new team members would have to face. Both the hardware and software sides of this projects were designed to produce long-lasting platforms that could be reused by a wide variety of students and researchers with different backgrounds. In order to allow collaboration at the software development level, the AAVS project is stored in a private Git repository.

2) *Hardware/Software Interfaces*

A simple lightweight communication protocol was implemented for the Arduino to send to and receive data from the computer. Messages are separated by a zero byte. Messages consist of one identification byte between 1 and 254 that states the type of message (motor command, pressure sensor value…), if the message can contain a zero (e.g. when it is a float) it is coded using the Consistent Overhead Byte Stuffing (COBS) to remove zeros. Hence the only zeros that appear are the ones that separate two messages. It proved to be easy to implement, efficient and reliable.

AAVS automatically detects when a sensor is plugged and it identifies the type of sensor.

3) *Sensors Simulation*

When used in simulation mode, AAVS simulates the vehicle's sensors. The simulated sensor data is used as input to the control and autonomy algorithms that will then be used on the actual vehicle in the hardware-in-the-loop mode. AAVS can simulate sensor input to test the algorithm in the lab.

Inertial Measurement Unit (IMU) and pressure sensor

Noise and bias can be added to the simulated sensor data to see how the algorithm behave when the measurements are not perfect.

Vision

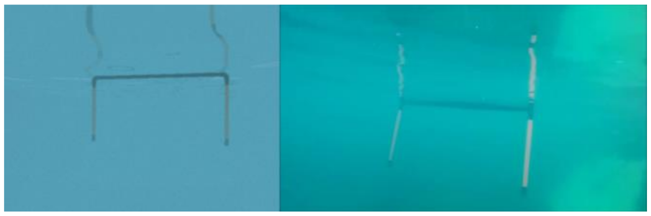
Figure 6: Simulated image (left) vs. real image (right)

The virtual environment used in simulation is rendered using the OpenGL API. 3D models of the environment and the objects encountered in the competition pool are designed using CAD tools, imported and placed in the simulation. Custom OpenGL Shader Programs were developed in order to recreate the conditions encountered by a visual sensor (such as a video camera) when placed underwater. The result can be seen in Figure 6: the simulated image resembles the real image with high enough fidelity to be used as a basis for the development of Machine Vision algorithms.

4) *Dynamics modeling*

The dynamics of an underwater vehicle have been extensively described and modeled by T. Fossen [2], and this model is widely used for the modeling and control of both surface and underwater vehicles.

$$M\dot{\nu} + C(\nu) * \nu + D(\nu) * \nu + g(\eta) = \tau \qquad (1)$$

Where M is the mass matrix (inertia matrix and added mass), C is the Coriolis and Centripetal matrix, D is the drag matrix, g(η) is the gravitational and buoyancy matrix, and τ is the force and torque vector given by the thrusters. Underwater currents can act as a disturbance on this model.

A complete derivation of the mass matrix M, and of the Coriolis and centripetal matrix C can be found in the work by T. Fossen and will not be detailed here. The dynamics of an underwater vehicle are highly non-linear due to damping, coupling and added mass effects. However, since the AUV described in this paper will travel at low speed and shows many symmetry, simplifying assumptions can be made. The Coriolis and centripetal matrix can be neglected as well as the coupling between the degree of freedom. The mass and drag matrix is diagonal. Drag is developed and identified to second order.

Since the vehicle is manufactured in parallel to the control development the dynamics of the underwater vehicle are modeled based on the planned characteristics of the vehicle and on data from another submarine with similar size and actuation.[3]. The parameters will be compared and updated against test data, and the impact of an incorrect model will be assessed.

5) *Control*

The control problem has been divided in two parts. An inner control loop stabilizes the system around the required angle and depth in the world coordinates. This inner loop consists of the IMU and depth sensor, a state estimator (such as an extended Kalman filter) and a Proportional Integral Derivative (PID) controller.

Four PIDs were implemented, one for each of the position on which there is direct measurements: pitch, roll, yaw, and depth. The PIDs parameters were selected to minimize the convergence time with limited excess of the target value. The PIDs were tuned using a test mission in the simulation. Position cannot be measured directly and the precision of the IMU is not good enough to perform dead reckoning (finding velocity and position by integrating the accelerometer). Dead reckoning is very sensitive to perturbations. Hence there is no feedback control on velocity or world frame position.

The extended Kalman filter is a very common state estimation techniques. It uses measurement and the local linearization of the dynamics model to estimate the state based on their relative confidence.

The outer loop uses image-based navigation. It extracts known features from the camera images to set the desired position and angle. Separating control in different loops enable the controllers to work at different frequencies. The outer loop is slowed down by the image processing time, whereas a fast inner loop ensures a good stability and minimizes drift.

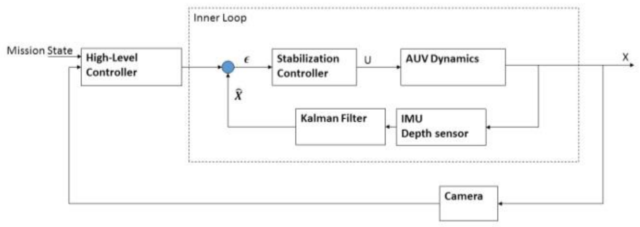

Figure 7: Nested control loops

6) *Machine Vision*

The Machine Vision algorithms developed to perform object recognition and tracking build upon the widely used and open-source OpenCV software. Wrappers for this native C++ library are available for most programming languages. Its C# wrapper - Emgu CV – proved to be particularly easy-to-use compared to other wrappers thanks to its clear documentation and its safe implementation of image types.

The Vision algorithms developed by the team can first be tested using the high-fidelity images rendered through the simulated environment. As with any other sensor, the simulated image data can then be replaced with actual video footage when AAVS is used in hardware-in-the-loop mode.

The general methodology that was used to perform object recognition and tracking consists of four steps: image processing, blob identification, object characterization and object tracking. These four steps are now described in greater details.

Image Processing

An initial image processing step is performed to obtain a single-channel binary image on which blob recognition can then be performed. First, the color space is changed from Red Green Blue (RGB) to Hue Saturation Value (HSV). Then, the value channel is extracted because it was empirically determined that it contained enough information to perform blob identification. Random noise is removed using morphological transformations. Finally, an adaptive thresholding is performed to obtain a binary image.

Blob Identification

Once a binary image has been obtained, contour detection is used in order to detect blobs, or regions of interest that can then be further characterized.

Objects Identification

The purpose of this step is to identify given objects among the blobs that were identified in the previous step. Whereas the two previous steps are very similar for every competition task, the object characterization step is highly dependent on the nature of the objects that need to be detected during a given task. In order to make an object's identification possible, a set of features is associated to it. This set of features is chosen so that it characterizes the object as uniquely as possible.

For instance, the starting gate was characterized as an object presenting two clusters of vertical lines (the two sides of the gate) and a single cluster of horizontal lines (the top bar of the gate). From the intersections of the horizontal and vertical lines and the bottom of the vertical lines, it is then possible to extract the position of the four corners of the gate and define a quadrilateral shape through which the vehicle should navigate.

Object Tracking

The purpose of this step is to use temporal information provided by the video stream (as opposed to still pictures) in order to improve the accuracy obtained with object identification. Because of noise present in the camera feed, and other factors such as reflections at the water surface distorted by waves, the results obtained from object identification may present variability from one image to the next. The actual movement of objects that we are tracking depends on the movement of the camera and the underwater currents, and since several images are recorded at a rather high framerate (several images per second), it is expected that the position of the identified objects does not vary too much from one frame to the next one. In order to take these effects into account, a particle filter was implemented. The particle population is propagated from one time step to the next by adding a random noise to allow for a slight movement of the tracked object. Then, particles are affected a weight based on their distance to the result from the object recognition. Finally, the particles are resampled and the weights are reset. The particle filter approach effectively results in a smoother and more reliable object-tracking.

IV. Experimental Results

A. Simulation Environment

1) Navigate through Validation Gate

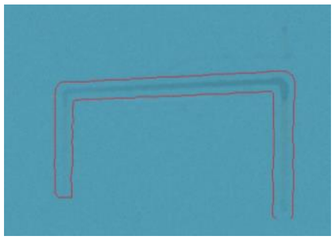
Figure 8: Gate detection

The starting gate is successfully detected in simulation using the methodology described in the previous section. In Figure 8, the contour of the starting gate was superimposed on the input image after the gate had been successfully detected. The robustness of the detection algorithm was assessed by adding random noises to the image. Once the gate is detected, the vehicle rotates to align with the center of the gate and them moves forward.

2) Follow the guide

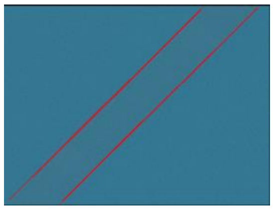
Figure 9: Guide detection

The vehicle must find the orange guide on the floor and align with it. The approach selected was to extract the guide position and angle relative to the vehicle. The normalized distance from the centroid to the center of the image and the angle between the vehicle and the guide is then passed to the controller which estimates the path heading by using the vehicle state estimation and the angle detected, and filter the result with a low-pass filter to set the target heading.

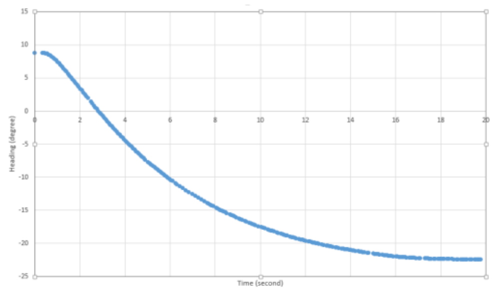

Figure 10: Evolution of heading during path alignment

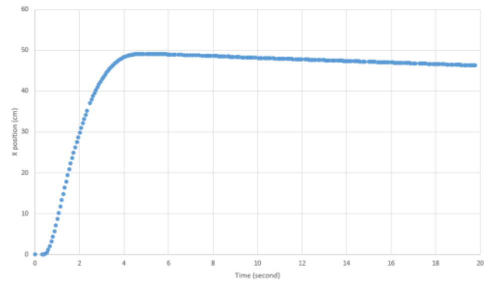

Figure 11: Evolution of X position during path alignment

3) Scuttle Ship

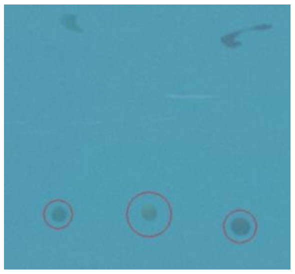

Figure 12: Buoys detection

The Scuttle Ship task requires to navigate towards and touch three potential target buoys with the vehicle. The treatment of this task slightly differs from the two previous tasks because some of the objects being tracked escape the camera field of view during the task completion. Therefore, the algorithm used in this particular task needs to be able to uniquely identify each of the buoys, and that even after they may have escaped the field of vision and reappeared. In order to easily identify the three buoys, it was decided that the vehicle would always keep the same point of view: after touching a particular buoy, it would navigate away from the buoys until all three buoys are back in the field of view.

B. Pool tests

The first pool test was performed on June 15th, 2016 at the Georgia Tech diving well. The objectives of this first testing were the validation of the waterproofing and the calibration of the IMU. Both tasks were successfully conducted. No mechanical issue nor leak occurred over the thirty minutes the vehicle was underwater. The IMU was successfully calibrated and all the thrusters were functional. The vehicle was initially slightly too buoyant but this was easily fixed by adding additional weights to the structure. The frame design proved to be particularly convenient to securely fasten the additional weight. More pool tests will be conducted to validate the results obtained in simulation.

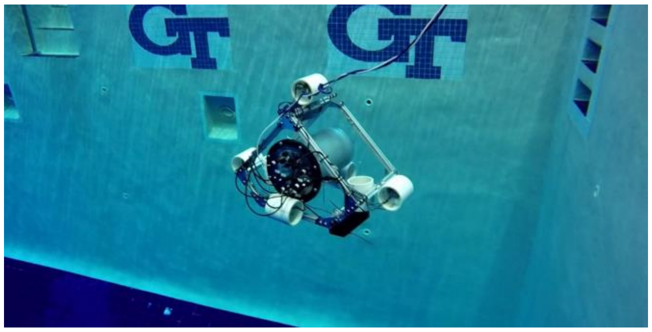

Figure 13: Pool Test

V. Conclusion

A new AUV has been designed and manufactured. A model of the vehicle was implemented in a simulation environment containing the RoboSub course. Control and autonomy algorithms have been implemented and tested in simulation. Future work before the competition includes validating the simulation results in the pool, developing the algorithms for the next tasks and integrating the actuator and hydrophone system if time allows it.

VI. References

[1] AUVSI, RoboSub team central, Available: http://www.auvsifoundation.org/competitions/competition-central/robosub/robosub-team-central

[2] Fossen T., “Guidance and control of ocean vehicles”, Wiley, 1994

[3] J.H.A.M. Vervoort “Modeling and Control of an AUV” 2009, Available: http://www.mate.tue.nl/mate/pdfs/10894.pdf.

[4] Triantafyllou and Hover, “Maneuvering and Control of Marine Vehicles”, MIT OPENCOURSEWARE, 2014

[5] Bonin-Font, Ortiz, and Oliver, “Visual Navigation for Mobile Robots: A Survey", *Journal of Robotic and Intelligent Systems*, Vol. 53, 2008, pp. 263{296.

[6] Krupinski, S., Allibert, G., Hua, M.-D., and Hamel, T., “Pipeline tracking for fully-actuated autonomous underwater vehicle using visual servo control," *ACC-IEEE*, 2012, pp. 6196-6202.