Ohio State University Underwater Robotics: *Riptide* AUV Design and Implementation

Chris Bennett, Conor W. Cary, Elizabeth Clapp, Lisa A. D'Lima, Peter A. Edin, Charles C. Fletcher, Benji W. Justice, Erika C. Klek, Nathan A. Lackey, Daichi Mae, Sara E. Mahaffey, August R. Mason, Sam McTurner, Luke Newton, Marshall J. Sayre, Achal S. Singhal, Cameron J. Spicer, and Eric Wolfe

***Abstract*— The Underwater Robotics Team (UWRT) is a student led and student driven engineering project team at The Ohio State University. Currently consisting of eighteen active members, the team has been involved in the design of both remotely operated vehicles (ROV) and autonomous underwater vehicles (AUV), the latter being its current focus. Autonomous underwater vehicles are underwater vessels often used in the exploration of the oceans. This includes surveying sea floors for the oil and gas sector, mine detection for military applications, and research applications such as recording carbon dioxide levels and microscopic life activity. This year, the Underwater Robotics Team has designed, manufactured, and programmed an autonomous underwater vehicle (AUV), *Riptide*, to compete in the 2016 AUVSI RoboSub Competition in San Diego, California.**

***Riptide* consists of ten underwater thrusters providing incredible maneuverability. Various actuators are mounted to the vehicle to allow it to manipulate the environment. Two forward facing cameras are used to construct detailed maps of its surroundings, while a third camera monitors the pool floor. The vehicle maintains its orientation by utilizing two inertial navigation systems. Computational power is provided by an i7 CPU, and a modular printed circuit board assembly was designed to provide power distribution and a communications hub.**

I. INTRODUCTION

The Underwater Robotics Team is a student project team at the Center for Automotive Research (CAR) at The Ohio State University. Previously, the team has primarily focused on the design and construction of remotely operated vehicles for the MATE International ROV Competition. Initially a small team of mechanical engineering students, the objectives of the team matched the abilities of its members. Beginning from a PVC-frame vehicle, the manufacturing and design knowledge quickly accumulated to include mold design, mill and lathe use, and design for manufacturability.

Furthermore, with each passing year, the team size grew and other engineering disciplines were incorporated into the team. This allowed for the design of increasingly complicated, yet sophisticated underwater vehicles.

This diversification has provided the team with a multi-discipline environment, and thus a more interesting challenge was needed. The team has shifted its focus to the design of autonomous underwater vehicles.

As the team grew, it became more involved with the community – leading workshops for high school students interested in underwater vehicle design, as well as a local middle school engineering club. More information is given in the Appendix.

II. DESIGN STRATEGY

The design of *Riptide* began by drawing on the past experiences of the team with remotely operated vehicles. Previously, the team experimented with thrust vectoring on a neutrally stable, under-actuated vehicle, which proved to be a difficult challenge.

Given additional intricacies introduced in the team's first experience with autonomous underwater vehicles, the vehicle was designed to be stable and fault tolerant. Stability of the vehicle involves the presence of a restoring torque that will reorient the vehicle in a known orientation when a perturbation is applied. Fault tolerance is implemented by two methods – over-actuation that allows for the failure of multiple thrusters, and system redundancy. On a higher level this involves placing multiple actuators and battery packs. Low level system redundancy includes monitoring battery pack status with temperature, pressure, and current sensors, as well as using two inertial navigation systems.

The physical design of the vehicle began as a sketch on paper and evolved to a complete design using SolidWorks. The goal was to create a vehicle that could be easily expanded or modified, allowing for a longer testing period in future years. The final rendering of the vehicle is shown in Fig 1. With the design and drawings completed, jigs and fixtures were created to aid in the manufacturing of the vehicle as shown in Fig 2, using a computer numerical control (CNC) vertical mill.

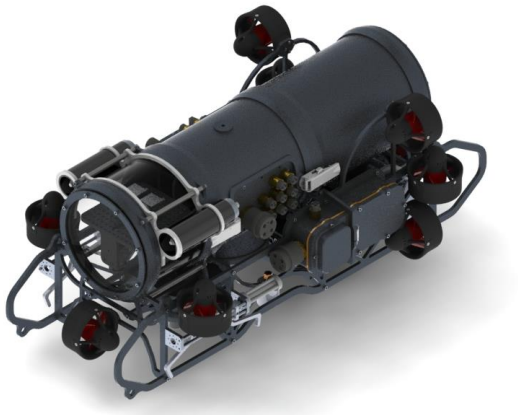

Fig 1. CAD Rendering of *Riptide*.

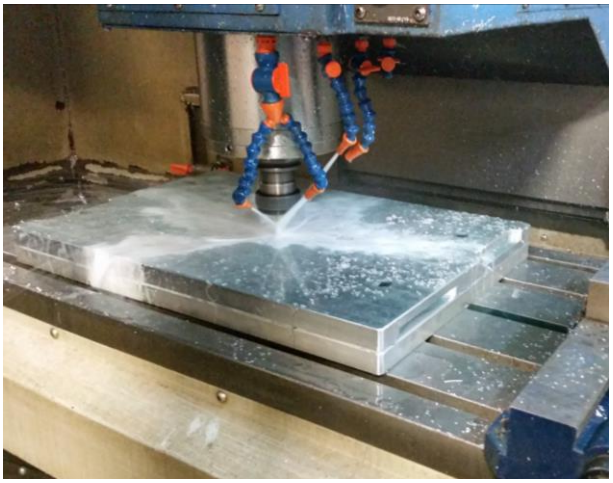

Fig 2. Milling of fixtures for the vehicle.

While the mechanical sub-team was focused on the design and manufacturing of the vehicle, the electronics sub-team concentrated on using computer aided design packages, namely CadSoft EAGLE, to construct printed circuit boards (PCBs). Modularity was heavily considered in the design to provide a base for future electrical systems. General circuit board design was a focus for team members so that modifications could be made easily between iterations and to provide a sufficient base for training new team members next year. Temperature and current sensing capabilities were included in the boards to be able to monitor the vehicle's state during a mission run. The manufacturing of the boards can be seen in Fig 3.

Fig 3. Soldering of the PCBs.

In the yearlong design cycle, a majority of the time was spent on the physical implementation of the vehicle. To aid in the limited pool testing time, a simulator was designed to provide a debugging environment for written code by the software team. This allows for the logic of the code to be tested without the vehicle being in the pool. The finalized specifications of the vehicle are provided in Table 1.

Table 1: Overview of *Riptide*.

Specification	**Dimension [*Units*]**
Length	36 [*Inches*]
Width	19 [*Inches*]
Height	14 [*Inches*]
Dry Weight	88 [*Pounds-Force*]
Max Depth (Tested)	17 [*Feet*]
Thrusters	10 x Blue Robotics T200
Cameras	3 x PointGrey Blackfly
Inertial Navigation System	1 x Microstrain 3DM-GX4-25
	1 x 9 DoF Razor IMU
Operating Framework	Robot Operating System (ROS)

III. Mechanical System Design

A. Primary Housing Module

The main housing is designed to provide easy access to the electronics for installation, maintenance, and diagnostics while minimizing disassembly. The vessel is made up of three main components, the central cylinder, and the forward and aft compartments. The central cylinder (Fig 4) secures the main housing to the chassis, and is not removed for regular maintenance. SubConn bulkhead connectors are able to pass through flat surfaces which were machined into the side of the housing. This allows for access to electrical components without disconnecting any connectors. To ensure the housing has the ability to be utilized in future years, a port on the underside of the central cylinder was machined to support a Doppler Velocity Logger.

Fig 4. CAD rendering of central cylinder of primary module.

The forward and aft compartments are fully removable. The assembly is sealed on both sides by double x-profile radial O-rings and secured by locking stainless steel latches. This allows for quick disassembly without risking incorrect reassembly, which could result in a leak. The forward compartment is made of an acrylic tube to allow an observer clear view of the main electronics boards for diagnostic purposes, as well as a downward facing camera view. This housing can be easily expanded and re-used in the future if the electronics volume changes. Both compartments have clear acrylic end caps to support stereoscopic cameras and a

diagnostic screen. A CAD rendering of the primary module can be seen in Fig 5.

The main housing was entirely custom made for this vehicle. The sealing rings on the front and rear of the three main cylinders, the bulkhead connector inserts, and the DVL port were machined in-house on a CNC machine. The parts were welded to reduce the machining complexity, and then post-weld machining was completed to bring the parts to final dimension.

Fig 5. CAD rendering of primary module with latches engaged.

B. Auxiliary Housing Modules

The auxiliary external housings contain the batteries, pneumatic valves and switches, and acoustic sensors. These were designed to be able to be moved around on the vehicle during the final design phase to evenly balance the vehicle, reducing the need for additional ballast weight or buoyant material. The auxiliary housing locations on *Riptide* can be seen in Fig 6.

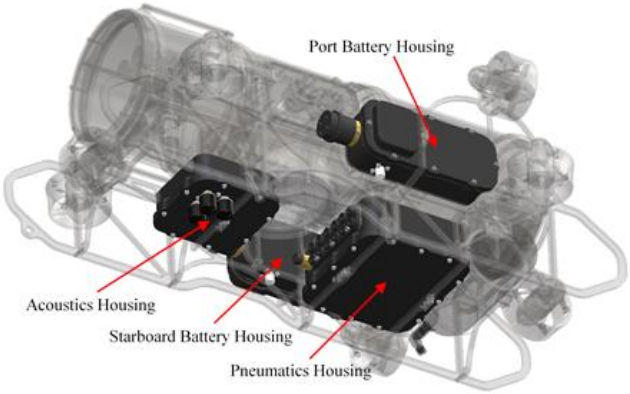

Fig 6. CAD rendering showing *Riptide*'s secondary modules.

The four housings have similar sealing methods that have been implemented and tested on previous vehicles. A lid is secured against a sealing flange using evenly spaced fasteners that are tightened to a specified torque, as shown in Fig 7. Housings that require a pass-through connection to the external environment were designed to have a reinforced bulkhead in the region that the connector will pass through. All structural features of the external housings were designed specifically for *Riptide* and were machined from raw aluminum alloy stock in-house using CNC and manual machining processes.

To make *Riptide* start and stop a run, or stop in an emergency, an external switch was designed. Since there were no good options for a commercial watertight button, a housing was designed to hold the buttons. The base consists of a Delrin block with holes for the two buttons as well for the SubConn bulkhead connector. On top of the base is a series of gaskets and aluminum plates to hold the gasket and buttons in place, as shown in Fig 8. There is a momentary switch for starting and a maintained switch for the emergency shutoff. To ensure the kill switch can be pressed in an emergency an additional shaft and large button head was machined and placed on top of the design base.

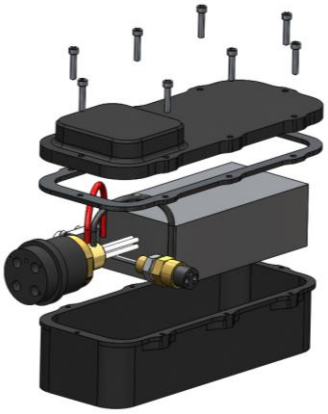

Fig 7. Exploded CAD rendering of battery housing.

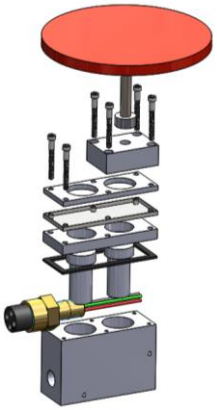

Fig 8. Exploded CAD rendering of kill switch.

C. Chassis

The vehicle features a modular design that is based around a central chassis, shown in Fig 9. The chassis is made from water-jet aluminum sheets and joined with stainless steel fasteners. There are hard-points for securing the tool-packages and reinforced regions to handle the vehicle. The vehicle may be carried by handles on the front and rear or hung from a crane to allow easy deployment.

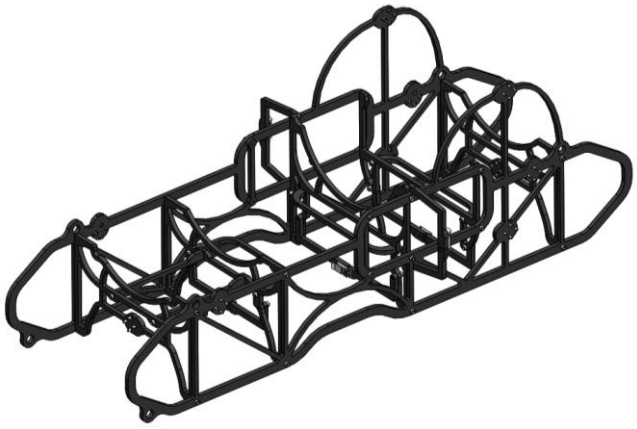

Fig 9. CAD rendering of *Riptide's* chassis.

D. Pneumatics

The pneumatics system is controlled by eight 3-way electronic solenoid valves. The valves are part of a manifold to reduce the number of connections as well as make the system more space efficient. The pneumatics system is kept isolated in the pneumatics auxiliary housing located near the bottom of the vehicle.

Air is supplied by an onboard, externally mounted, air cylinder. The pressure is regulated down to the working pressure of 38 psi to allow a constant working pressure until the tank is nearly depleted. Air lines pass through a custom reinforced bulkhead that supports the use of off the shelf fittings.

E. Actuators

Riptide is able to move and complete the various tasks of the RoboSub competition through the use of several actuators mounted to the chassis of the vehicle.

1) Thrusters

Ten T200 Blue Robotics thrusters are utilized on *Riptide* and give it the ability to move with six degrees of freedom. The thrusters are fixed to the water-jet chassis, four horizontal, four vertical, and two sideways. The thruster wires were cut to length, soldered, and resin was injected in molds to seal the electrical connections.

2) Grippers

There are two manipulators on *Riptide*, located on either side of the vehicle along the forward housing compartment. The claws were designed to reach as far as possible while still maintaining a low profile when not in use. To do this, a two way linear actuator was used. The claw arms were water-jet and Delrin inserts were used around the screws to reduce friction during actuation. Additionally, stainless steel rods with a slider were used to reduce any side loading. A CAD rendering of one of the claws can be seen in Fig 10.

Fig 10. CAD rendering of *Riptide*'s claw.

3) Marker Droppers

There are two marker droppers aboard *Riptide*, located on either side underneath the central cylinder of the housing. Each marker dropper is operated by a permanent magnet, a steel 1 in diameter ball, and compressed air. The steel marker ball rests in a cylindrical chamber and is separated from the magnet by a 1/8" Delrin plate. The plate is thin enough that the magnet can keep the ball still, yet it is thick enough that firing compressed air through a check-valve at the top of the chamber can overcome the magnetic force and release the ball. Gaskets are used to water-tight seal the magnet. A gasket is placed between the magnet and the top plate and between the magnet and the magnet face plate.

Other designs were considered, but this design was ultimately chosen due to simplicity. Another design required pneumatic controls which would have worked, yet required additional hardware for the AUV. The final physical design was chosen based on machinability. Other previous designs were found to be too difficult to machine or too difficult to replicate more than once if something went wrong with the first part. The use of Delrin as the main component material kept the part lightweight, easy to machine, and inexpensive.

4) Torpedo Launchers

The torpedo launchers consist of two Delrin barrels held in place by two Delrin support structures, shown in Fig 11. The launchers are attached at the top of either side of *Riptide*'s forward housing compartment. Two torpedoes are launched from the barrels through the internal pressurization of the torpedoes themselves. At the pneumatic pressure's peak, the torpedo slips passed the O-ring keeping it attached to the pneumatic valve and speeds forward through the water. The torpedoes consists of a 1.5-inch diameter head attached to a 0.75-inch diameter shaft that has 3 stability fins attached at its base. The 5-inch long torpedo fits snugly in the 5.5-inch long barrel to assure that all air pressure is devoted to pushing the torpedo out of the barrel and so the torpedo does not wobble prior to exit.

Fig 11. CAD rendering of *Riptide*'s torpedo launcher system, with one torpedo in resting position and one in the launched position.

The initial stages of the torpedo launcher's development began with extensive research into approaches of past successful teams. It was found that pneumatics seemed to be the most feasible option for success.

After finalizing the pursuance of a pneumatic launcher system, the next step was to determine how to pressurize the mechanism. Several methods were proposed including pressurizing the barrel external to the torpedo. Finally, the decision was made to pressurize the inside of the torpedo itself to eliminate the friction vs. effective sealing decision of the torpedo-barrel contact.

Most of the machining was done on a lathe including the barrels (turned and bored), the torpedo head (turned and bored), and the torpedo shaft (faced). The support structures for the barrels were water-jet by a third party.

IV. ELECTRICAL SYSTEM DESIGN

A. External Electronics

Within the secondary modules of *Riptide* there are several electrical systems in place to provide power to the vehicle and to control several external devices.

1) Battery Monitoring

Riptide is powered by two MaxAmps 8000 mAh 18.5 V Dual Core battery packs, allowing for two to three hours of testing depending on intensity.

There are battery monitor boards located in each battery housing. They monitor the temperature and pressure within the housing to ensure the battery conditions are safe and allow efficient operation. This status data is relayed to the Power Distribution board over a RS-232 serial connection.

2) Pneumatics Controller

The Pneumatic control board activates the solenoid valves that direct air to the pneumatic actuators on the AUV. An atmega328 microcontroller receives commands from the computer through an RS-232 to UART interface to control eight separate outputs. Darlington transistors are used to amplify the logic signal to a higher current, 12 volt signal used to drive the solenoid valves.

3) Acoustics Processing

Three hydrophones mounted below the vehicle provide data for determining the pinger's position. The hydrophones are separated such that the underwater pinger's transmitting wave cannot travel one whole wavelength in between the hydrophones.

The acoustics processing is currently being completed with a Codec shield connected to an Arduino Uno. The shield has a WM8731S Codec for capturing the hydrophone data, which is then transferred to the main computer for processing.

B. Internal Electronics

The bulk of the electronics for *Riptide* are located inside the main housing of the vehicle and provide capabilities such as high-level processing, sensor data collection, power conversion, and thruster control. The electrical components are mounted on an aluminum frame that can slide out of the vehicle if required.

1) Onboard Computer

The computer onboard *Riptide* for running all software packages and sensors consists of a BCM Advanced Research MX87QD motherboard with an Intel Core i7-4790S CPU.

2) Sensors

The sensor suite that is utilized on *Riptide* consists of two inertial measurement units (IMU), one LORD MicroStrain 3DM-GX4-25 and one 9-DOF RAZOR IMU, for acceleration and orientation information, three PointGrey Blackfly USB3 Vision cameras (BFLY-U3-13S2C-CS) for vision processing and visual odometry, one Blue Robotics Bar30 pressure sensor for measuring the depth of the vehicle, and three Aquarian Audio H1C hydrophones for acoustic signal processing.

3) Backplane

The backplane is a circular printed circuit board with 11 female connectors. All of the boards in the inside housing mount to this board, creating a wireless system for all the boards to communicate and source power from one another, as shown in Fig 12.

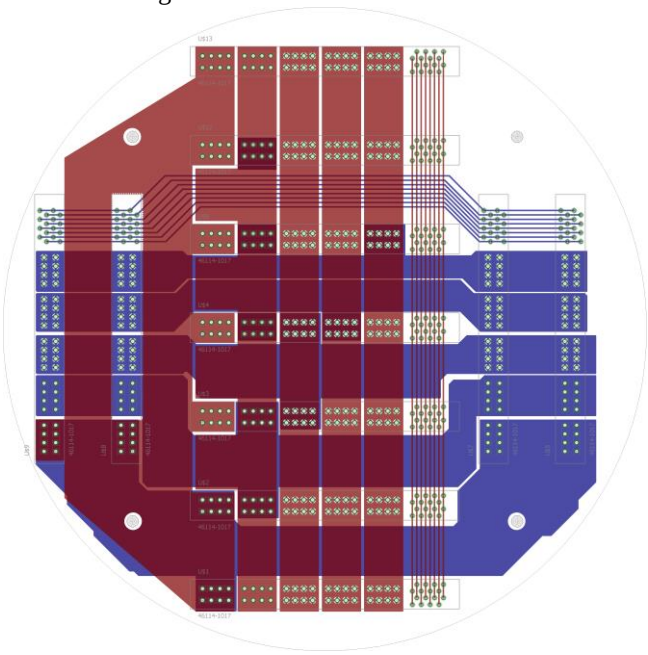

Fig 12. Board layout of backplane.

The power system is mounted on the four edge connectors from either side, while the thruster controller board and five electronic speed controller (ESC) boards are mounted down the center, as the ESC boards must receive signals from the master thruster board. Five and twelve volts are converted from the balanced load sourced from the power distribution board and then distributed to each board through the designated pins. However, the boards may communicate only within their system; e.g. the thruster controller board may only talk to the ESC boards, but not the 12 V board. By separating the communication signals, the amount of noise is reduced. Finally, the backplane has four holes in each corner. This is where the board is mounted in the main housing. An image of the assembled electronics mounted on the backplane can be seen in Fig 13.

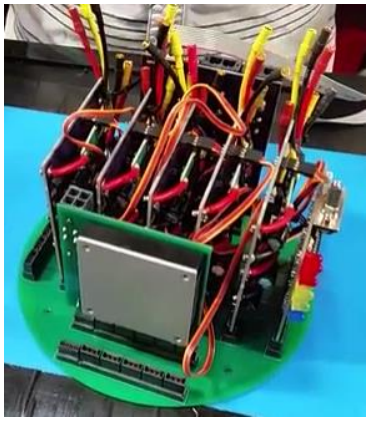

Fig 13. Custom Electrical Assembly.

4) Power Conversion

Power conversion is completed with a 12 V and a 5 V board. The purpose of the 5 V converter board is to take the raw voltage from both batteries and alter the 18.5 V to the desired 5 V. First, the raw voltage flows from the power distribution board to the edge connector. This then follows large traces to a Murata isolated 5 V DC-DC board. The newly converted voltage is then fed into a current sensor, so that the team may monitor the flow of power and potentially cut power from the robot if there is significant overcurrent. After the current has been measured, the 5 V is then sourced into a specified pin on the edge connector, which all other boards use to obtain 5 V. This board also has board-to-wire connectors for fans that will help circulate air throughout the main housing.

The 12 V converter board simply converts the raw 18.5 V into 12 V in a similar process to the 5 V converter board. However, this board features a trim pot that is connected to the sense pins in order to adjust the output voltage. Again, the output of the 12 V converter board is fed into a current sensor to monitor if the current flow is correct and stable, and then flows into the designated 12 V pin. All other boards will source from this pin for 12 V if needed. This board directly powers the computer to ensure that there are no significant drops in current in order to prevent the system from rebooting.

5) Power Distribution

The power distribution board manages the load from both batteries and communicates the status of all the power system boards to the computer. First, both currents travel through a 30 amp fuse to prevent an overcurrent in the system. Current sensors individually monitor both batteries, and then the current paths are merged using a low-loss power path controller. The balanced voltage then enters the edge connector. This board has two RS-232 connections, as well as three small Molex connectors. Because there is a large current flow through this board, there are several decoupling capacitors around the microcontroller to remove noise. The board layout for the power distribution board can be seen in Fig 14.

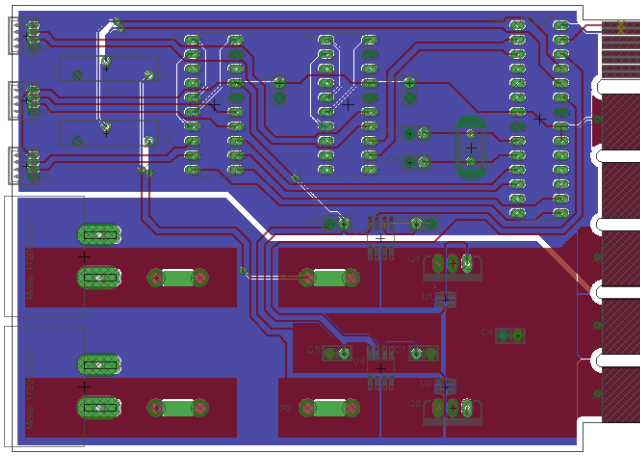

Fig 14. Board layout of power distribution board.

6) Thruster Controller

The thruster control board relays commands between the computer and the individual motor ESC boards, as well as monitoring the status of each ESC and the state of the kill switch. An atmega328 microcontroller communicates with the computer via a RS-232 serial signal. The computer sends the power level for each thruster through this connection then the microcontroller relays these power levels over an I2C port to the ESC boards. Through the same I2C connection each ESC board returns current and controller temperature data. The kill switch is connected through this board, allowing it to activate the thrusters.

7) ESC Controllers

The ESC board is designed to send a PWM signal to each motor controller, as well as monitoring the status of each thruster and controller. Each individual board has two motor controllers. The battery power comes in from the backplane. The power then goes through two relays that are hardwire-controlled by the kill switch. The relays are fail open, or in other words the kill switch has to be providing 5 V to each relay in order for the thrusters to be powered. There are four sensors on each ESC board: a current sensor and temperature sensor for each motor controller. An attiny1634 microcontroller communicates with the thruster control board over an I2C bus, receiving the desired thrust level and sending the sensor data for each thruster. An image of a complete ESC board can be seen in Fig 15.

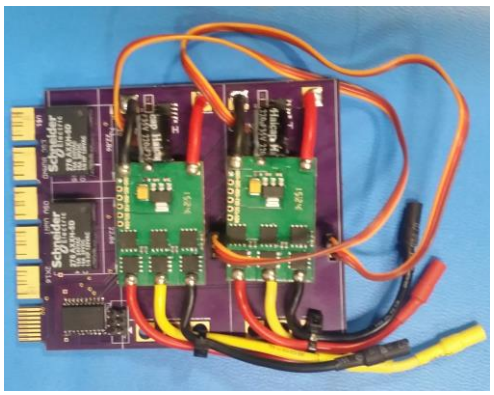

Fig 15. ESC circuit board.

V. SOFTWARE SYSTEM DESIGN

A. Robot Operating System

The team has elected to build most of its software using Robot Operating System (ROS), a modular framework for collaborate robotic software development. The modularity of ROS eases efforts to break the software system into manageable projects throughout the school year and then integrate them into the full system as they are completed. The collaborative aspect of ROS encourages many researchers and corporations to provide open source modules (called packages in ROS) which implement a variety of algorithms and device drivers. By taking advantage of this pre-existing software, the team saves significant development time and

gains the ability to learn from the work of experts.

The team has created a number of its own packages to operate its AUV. The most significant packages are Autonomy, Estimation, Navigation and Vision, as seen in Fig 16.

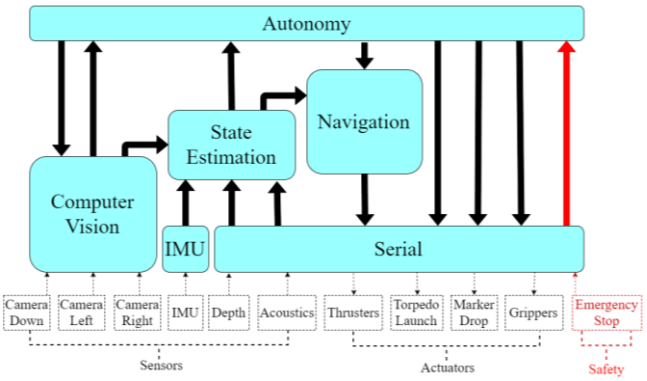

Fig 16. Software package stack.

The Autonomy package runs the mission by creating a sequence of desired states throughout the competition. The Estimation package continuously analyzes feedback to determine the vehicles current state. To travel throughout the course, the Navigation package resolves discrepancies between the desired and current states to create thruster commands. The Vision package handles the vehicle's primary sensors, the forward-facing stereo camera system and the downward-facing monocular camera system.

A number of open source packages were used in conjunction with those created by the team, such as the imu_3dm_gx4 packages [1].

A number of smaller packages were also created to support those listed above. A serial interface package enables communication with the team's custom electronics system and, by extension, the remaining sensors and actuators. A Teleoperation package was created for initial control testing and vehicle demonstration. The Description package provides a vehicle model which can be accessed by other software to determine the vehicles properties when making dynamic calculations.

B. Autonomy

The State Machine is the decision-making tool that allows the computer to control its behavior autonomously. SMACH is a package in ROS that provides these functions to be used in easier form [2]. In SMACH each state defines the actions or tasks to be performed. This allows *Riptide* to execute each task and make transitions smoothly, and it also allows team members to monitor the vehicle's state. One of the classes in SMACH that was particularly useful was the SimpleActionState, which allows the state machine to access nodes through action files. This feature was necessary since the state machine manages and operates specific nodes accordingly.

C. Model

The vehicle is modeled using the Universal Robot Description Format (URDF) to create a single point of control for many of the vehicle's physical and operating parameters. All packages can access the model as necessary. For example, the subsystem which maps the commanded acceleration on *Riptide*'s body to forces at each thruster references the model each time it is initialized. Referencing the model allows this subsystem to update the equations it uses for mapping accelerations to forces for the current vehicle configuration. Should a thruster or massive component be moved, all packages are effectively updated whenever the model is updated.

The URDF models the vehicle as a graph. This graph takes the form of a tree, where each node represents a physical section of the vehicle (link) and each edge represents how those links are connected (joints). Storing vehicle parameters in this way allows graph-based algorithms to easily determine how various portions of the vehicle are related to others. For example, determining the transformation between two links by identifying all intermediate links and combining their transformations.

The team is working to utilize its models in the future to add significant simulation capabilities to the development cycle. This addition would allow the software team to better prepare for vehicle testing before construction has been completed. It is hoped that this will eventually lead to the ability to simulate designs before they are finalized, further enhancing the design cycle. The progress of the simulator can be seen in Fig 17.

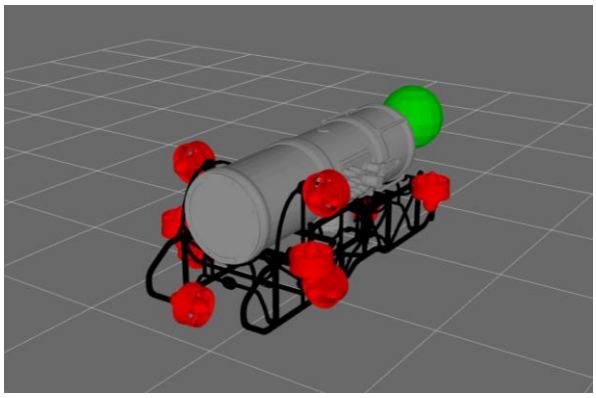

Fig 17. Image of simulator environment.

D. Estimation

The outputs from the two IMUs, mono visual odometry from the bottom-facing camera, absolute position estimates from the vision system, and depth measurements from the pressure sensor are inputted into an extended Kalman filter, specifically the EKF Localization node [3]. This filter then produces estimates for the absolute linear and angular positions and velocities of *Riptide*.

For consistency between ROS nodes and simplicity in setting the desired states of *Riptide*, the world frame is utilized for all calculations. Since several sensors output in

the body frame, for example the IMUs, transformations are necessary. This is completed using the tf package in ROS [4].

E. Vision

The *Riptide* Vision System revolves around the three cameras onboard the vehicle. Two of the three are mounted in the front side-by-side and provide a depth map via OpenCV stereo camera algorithms. Additionally, each camera outputs both a color and monocular rectified image for general color and contour matching algorithms.

As a team in its first year competing at AUVSI, algorithm design was kept relatively simple. Each algorithm is stored in a general purpose library which can be called on with only an image as an argument. Having simple function contracts allows those working on mission design logic to utilize the vision library to its full effectiveness by not having to understand every detail of the vision system, only needing to understand the concept of passing in an image, and getting a heading or distance in return.

This is the team's first attempt at a computer vision system, as such, a lot was learned very quickly. This includes the distortion of colors based on location, time, and depth of submersion. Additionally, there are a variety of ways to go about doing each task with no clear winner. Up to four versions for each algorithm were developed, each shined in certain circumstances and flopped in others. For example, the algorithm which determines the direction of the orange marker went through three iterations, one which detected the rectangular shape and drew conclusions about the heading via the corners, one which found the middle point of the object and then the middle point of the top half and bottom half, and the heading from those two points. Lastly, the winner, operates by taking the threshold of the color orange, yielding a black and white image, with white points being where orange is in the original, and an average line is calculated based on those points. The theory is that the distribution of points will always form a best fit line in the direction of the marker, no matter how cloudy the water or the small differences in color which are not let through the threshold. This proved to be simple to implement, and very effective during trials.

For more complicated tasks, images go through a series of stages which work on a guess-and-check basis. For example, when buoys are being located, the image is first separated by color and potential circles on the threshold image and are added to a list. This list is then passed to the next stage where a circle contour algorithm checks if, in that area of the image, a circle is present. A similar next stage happens with the depth image. This approach is very useful because taking the threshold of an image is computationally cheap. This allows us to dramatically simplify the image for the more computationally expensive algorithms to take place. For example, rather than finding circles in the raw 1024 x 768 image, the complex circle algorithm only runs through a few 100 x 100 images, 1/70 the size; with a runtime complexity of O(n3), this simplifies the computational task to relatively nothing, reducing heat production in the housing, and freeing up computer resources for other algorithms.

F. Acoustics

The acoustics processing software was programmed using Code Composer Studio in the language of C++. After receiving a signal from the hydrophone array, unwanted frequency ranges are filtered out using a digital bandpass filter. The specified frequency range is 25-40 kHz.

Once the signal is received by *Riptide*'s three hydrophones, the angle of arrival of the emitted wave is calculated using phase difference between the three received signals. With the hydrophones positioned so that one whole wavelength is larger than their separation, the angle of arrival of the wave is calculated using the phase difference between each of the hydrophones and simple trigonometry. If the hydrophones were separated further than one wavelength then the phase difference could not be used since the system would not be able to determine which hydrophone received the signal first.

The heading information containing the direction that the vehicle needs to travel in order to reach the pinger is then sent to the central processing computer continuously until the breeching sequence activates.

G. Navigation

The current state of the vehicle, determined from the Kalman filter, is used in conjunction with the desired state provided by the state machine and several PID controllers to ascertain required x, y, and z acceleration values. The PIDs are configured and outputs calculated by leveraging the Control Toolbox class in ROS [5]. The PID controllers are implemented separately for linear and angular accelerations, as shown in Fig 18, and for each axis in order to provide the most accurate PID parameters for each case. The PID parameters can be updated real time by using Dynamic Reconfigure [6]. The desired state can also be provided by a joystick from the surface in the case of tethered movement during the testing phase. This capability allows for easy tuning of the PID controllers and stabilization assessment.

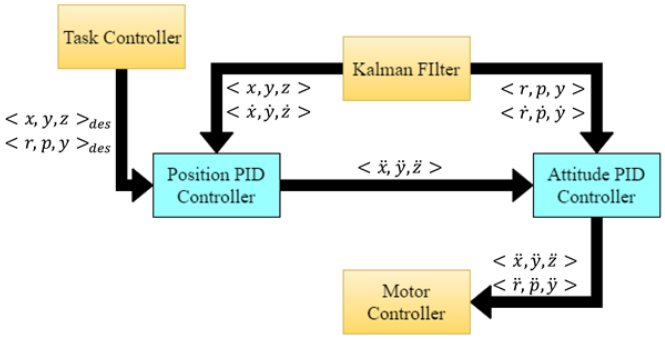

Fig 18. PID diagram.

H. Controls

The vehicle control stack is comprised of a thrust mapper and a thrust calibrator. These components begin with an acceleration vector, provided by the navigation stack, and ultimately activate the thrusters in any combination necessary to achieve said acceleration. The modular form of this system allows for control by either high-level mission planning software or direct user input, which has proven useful for testing purposes.

The thrust mapper solves the equations of motion for the vehicle using a non-linear solver called Ceres [7]. This determines what force each thruster must produce in order to yield the required acceleration. Success of the mapper relies heavily on the accuracy of the error calculated by the PID controllers as well as the mathematical representation of the vehicle as a physical body, provided by the model. The mapper succeeds when it finds a solution for the thrust required by each thruster within certain constraints, such as maximum thrust possible. If no solution exists, the mapper outputs an error which is then handled by the rest of the system as needed.

Using the forces provided by the thrust mapper, the thrust calibrator transmits the PWM signals activating the thrusters as required. The challenge in developing this component lied in finding which PWM signals actually yielded the desired thrust. Being the only component in the stack with a physical output, testing its real values required significantly more effort than the components with software outputs.

VI. EXPERIMENTAL RESULTS

Each thruster was tested with a thruster testing stand that consists of a 100 gallon tank, aluminum t-slot, Arduino microcontroller, and a 50 lb. load cell to measure power, rpm, current, voltage, and thrust output. When the thruster applies forward thrust, the L shaped t-slot pivots and the force exerted on the load cell is recorded. In future AUV designs, the thruster testing stand will be used prior to thruster selection to allow the most efficient thrust to be chosen.

Fig 19. Image of *Riptide* during pool testing.

The method of testing for *Riptide* consists of pool testing as well as simulations. Thus far, 18 hours of pool testing has been completed, shown in Fig 19, and the simulator will begin getting utilized soon. The buoyancy of the vehicle has been finalized and initial thruster calibration, and camera processing has been completed.

VII. ACKNOWLEDGEMENTS

The Ohio State Underwater Robotics Team's vehicle, *Riptide*, would not have been possible without the many devoted team members as well as the Center for Automotive Research, the Ohio State University College of Engineering, and the team's corporate sponsors who provided services, products, and funding.

Brutus Level Sponsors:

- College of Engineering
- Ohio State Center for Automotive Research
- Honda OSU Partnership
- CadSoft Eagle
- Ohio Space Grant Consortium
- Battelle
- HSMWorks
- SolidWorks

Buckeye Level Sponsors:

- Shell
- Ometek Incorporated
- EWI
- LORD Microstrain
- Danco Anodizing
- B&G Tool Company

Block-O Level Sponsors:

- Rimrock
- E&F Plastics
- John Deere

REFERENCES

[1]"imu_3dm_gx4 - ROS Wiki", *Wiki.ros.org*, 2016. [Online]. Available: http://wiki.ros.org/imu_3dm_gx4. [Accessed: 20- Jun- 2016].

[2]"smach - ROS Wiki", *Wiki.ros.org*, 2016. [Online]. Available: http://wiki.ros.org/smach. [Accessed: 16- Jun- 2016].

[3]"robot_localization - ROS Wiki", *Wiki.ros.org*, 2016. [Online]. Available: http://wiki.ros.org/robot_localization. [Accessed: 16- Jun- 2016].

[4]"tf - ROS Wiki", *Wiki.ros.org*, 2016. [Online]. Available: http://wiki.ros.org/tf. [Accessed: 16- Jun- 2016].

[5]"control_toolbox - ROS Wiki", *Wiki.ros.org*, 2016. [Online]. Available: http://wiki.ros.org/control_toolbox. [Accessed: 16- Jun- 2016].

[6]"dynamic_reconfigure - ROS Wiki", *Wiki.ros.org*, 2016. [Online]. Available: http://wiki.ros.org/dynamic_reconfigure. [Accessed: 16- Jun- 2016].

[7]"Ceres Solver — A Nonlinear Least Squares Minimizer", *Ceres-solver.org*, 2016. [Online]. Available: http://ceres-solver.org/. [Accessed: 16- Jun- 2016].

APPENDIX: COMMUNITY OUTREACH

The Ohio State Underwater Robotics Team represents a unique niche in the local community. The team engages in the community by participating in events such as the Ohio State Fair. As a participant in the Ohio State Fair, the Underwater Robotics Team designs and builds an exhibit to educate the local community about marine engineering. This involves small remotely operated vehicles (ROV) which guests can actively control and engage within a small pool. The past year, the team moved from using the classic PVC and bilge pump ROV to a more innovative vehicle. This new vehicle, the STEMbot (shown in Fig A1), replaces PVC pipe construction with an aluminum housing, swaps the bilge pumps out for brushless hobby motors and is controlled using a Raspberry Pi and PlayStation controller instead of the typical switch box. The STEMbot is more maneuverable and its controls are more intuitive than previous PVC based vehicles. This enhances the entertainment value for many guests; both young and old, while providing more thought provoking and relevant discussions with young persons interested in STEM.

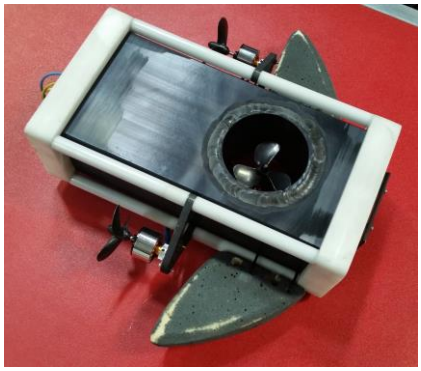

Fig A1. Photo of STEMbot

Although not a quantifiable metric, the joy the exhibit brought to everyone was a rewarding experience, engaging teachers and encouraging them to incorporate the aspects of underwater vehicular design to their classroom. The team has even engaged younger interested individuals with special needs to encourage them to pursue their underwater interests. The Ohio State Underwater Robotics Team has reached out to the PAST Foundation and Metro Early College High School to plan and create sub-sea activities for students. With the PAST Foundation, for example, the team has aided in two ROV competitions as team mentors and competition judges. This year, the team has been helping the PAST Foundation transition their events to MATE regional qualifiers. Having competed at MATE the team was able to provide unique guidance to the PAST Foundation and local teams as they prepare to get involved with MATE. With the students involved having little knowledge on the tricks of the trade, workshops were created in order to guide them in areas such as buoyancy, technical communication, career opportunities, and mechanical design. These workshops reached over 50 students at two different local high schools. An image of a volunteer discussing ROV stability can be seen in Fig A2.

Fig A2. Photo of volunteer with students at workshop event.

The Underwater Robotics Team has also engaged younger students at the Metro Middle School. As both mentors and lesson planners, volunteers taught 15 grade schoolers how to construct and program a line following robot. This year there are plans to incorporate more of the design process, by having students design and build their own speaker.