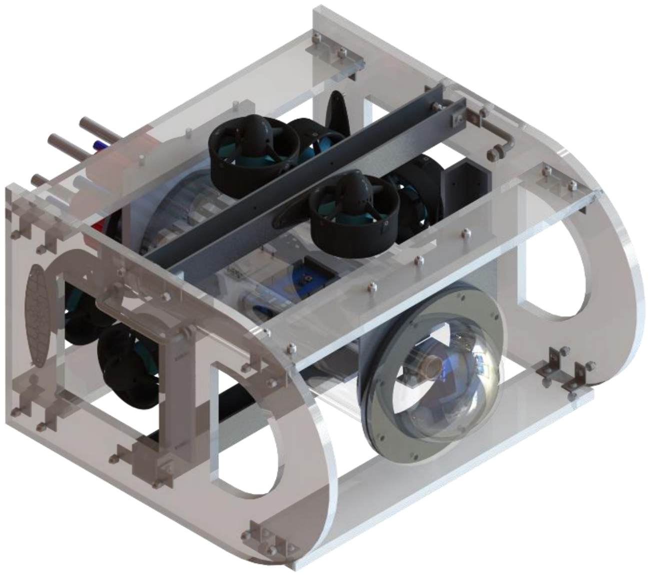
UNH AquaCats

University of New Hampshire (UNH)



Team Advisors

Professor Dr. May-Win Thein
Professor Dr. M. Robinson Swift
Dr. Firat Eren

Team Members

Jarrett Linowes (CEO, ME)
Nathanial Cordova (CTO, ME)
Shaun Hespelein (Chassis, ME)
Sayward Allen (Treasurer, ME)
Gerald Rosati (Controls, CE)
Alex Sarasin (Controls, EE)
Sathya Muthukkumar (Controls, ME)
Alex Dzengeleski (Chassis, ME)
Zhuo Xu (Controls, ME)
Jianqing Ye (Controls, ME)

Table of Contents

Abstract

UNH ROV is an interdisciplinary engineering company focused on designing, fabricating, and competing with an underwater Remotely Operated Vehicle (ROV). The design incorporates a high degree of modularity by accounting for constraints defined by the international MATE ROV Competition as well as research specifications involving autonomous Unmanned Underwater Vehicle (UUV) control. The MATE ROV Competition features 3 missions which have unique objectives relating to an arctic theme, while the research involves using an optical feedback system for pose control of multiple UUVs. Each part of the design is conceived using 3D modeling software, analyzed using finite element simulation packages, and verified through a prototyping and testing process. UNH ROV also focuses on maintaining a diverse team of students to maximize innovative and creative design. By using cutting-edge methods in team dynamics, analytic tools, and engineering design, UNH ROV is able to maintain continued success in developing underwater robotic systems used for multiple different platforms for research and competition.

Final Design

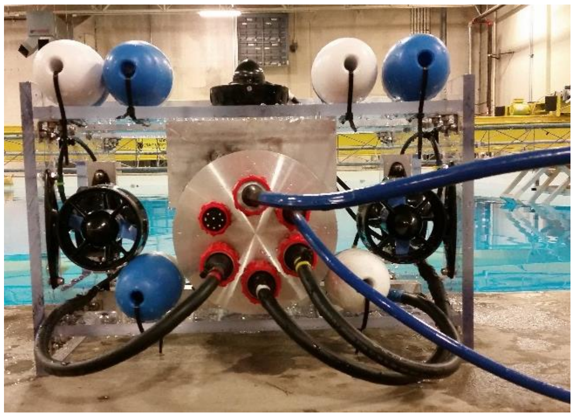


Figure 1: The back view of the final design of the UNH ROV, code named Viper, showing the connections to and from the electronics housing tube as well as the forward/backward thrusters.

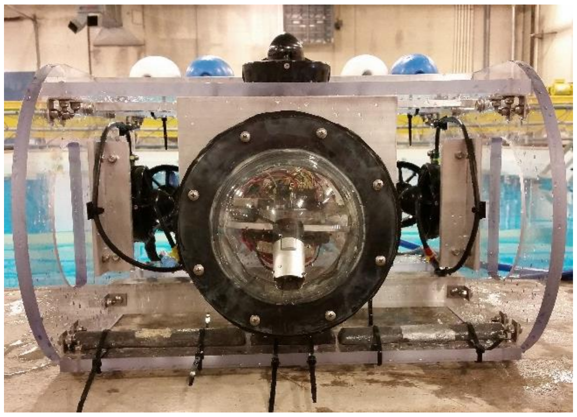


Figure 2: The front view of the final design of the UNH ROV, showing the camera dome as well as the frontal support beams for the mounting of sensors, lighting, and various manipulators.

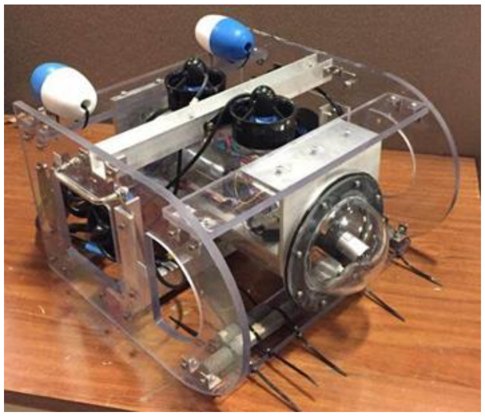


Figure 3: Full isometric view of the final design of UNH ROV.

Tentatively Proposed Budget

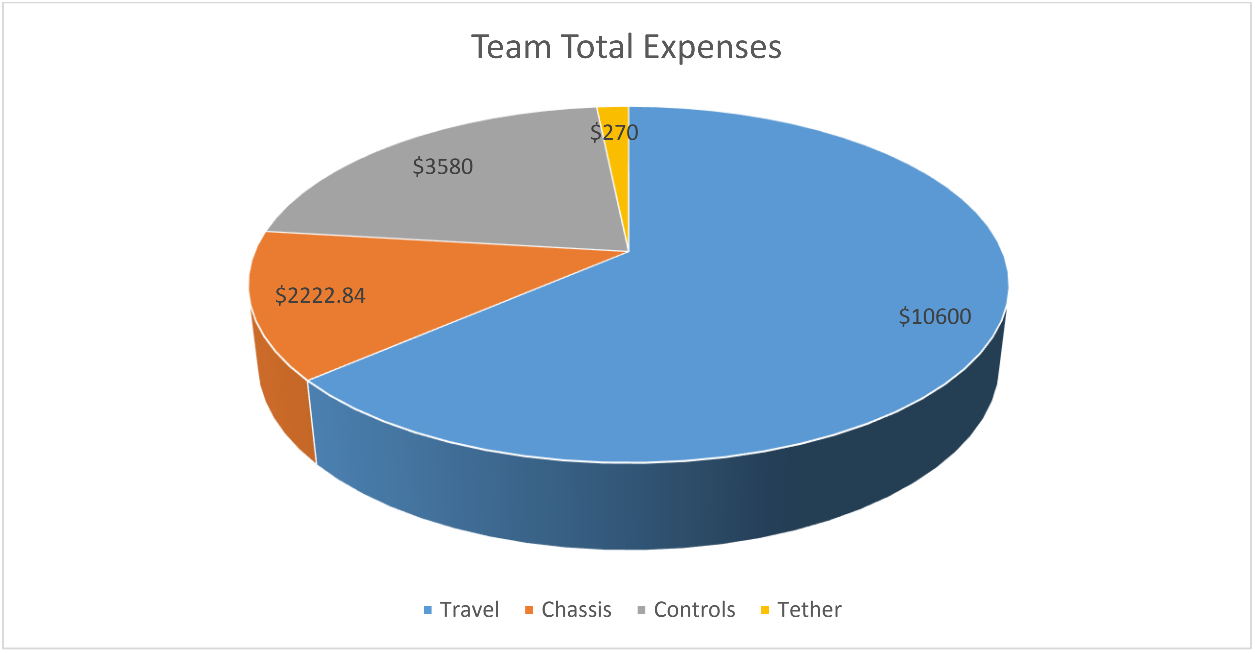


Figure 4: Budget initially proposed for travel to and from competition for 10 members, as well as the cost of the build for the chassis, controls, and tether team.

Project Cost Analysis

The team's income came from various sources for various reasons. Designing and building an entirely new product meant that a lot of money would have to be raised in order to do so. The team received funding from a few school departments in recognition for students efforts to aid in the senior capstone project. Also, UNH Aquacats received some funding from a NEEC and NAVSEA grant that tailored to the use of the product after the MATE competition. In order to further raise funds for travel to and from the competition location as well as for the ROV build, it was necessary to design a sponsorship package. The package was sent to various companies, local and not, with requests for funds in exchange for advertising and tax breaks for donating to the university.

The initial budget and funding requirements were split between travel, the chassis team, the controls team, and the tether. Travel, the largest cost burden, ultimately ended up costing less than initially budgeted for due to some members not attending competition. Due to our ability to machine more parts than expected in house, the chassis spending was also less than initially budgeted for. Because the thrusters and speed controllers were suggested to be the most expensive components of the build (due to our decision to purchase rather than design and build), we initially budgeted the controls as a large cost. Ultimately, the Blue Robotics T100 thrusters purchased were more reasonably priced than expected, therefore the total proposed budget for the UNH ROV was larger than the total expenses of the project.

Table 1: Table displaying the funds obtained and purchases with regard to each engineering team.

Column1	Price	Material	Purchase/Donated
Finances Gained	1600	Monetary funds	Supplied by Department of Mechanical Engineering
	1000	Monetary funds	Supplied by Undergraduate Ocean Research Project Class
	150	Monetary funds	Supplied by Department of Electrical and ComputerEngineering
	200	Monetary funds	Earned through presentation
	2000	Monetary funds	Earned by winning grant money
	532	Monetary funds	Donated
	1000	Monetary funds	Green Pages Technology Solutions Sponsorship
	500	Monetary funds	EdgeTech Sponsorship
	1000	Monetary funds	Kepware Technologies Sponsorship
	500	Monetary funds	Donated
		Monetary funds	Supplied by NEEC and NAVSEA grant)
Total	**8482**		
Chassis Purchases	191.2	Electronics tube raw material	Purchased
	34.03	Aluminum plate and polycarbonate raw stock	Purchased
	21.89	O-rings for endcaps	Purchased
	617.84	Aluminum raw material and hardware	Purchased
	459.06	2 36”x36” Polycarbonate sheets	Purchased
	48.96	Aluminum angle raw material	Purchased
	70.86	Bracket, nuts, bolts, handles	Purchased
	61.45	Acrylic Dome	Purchased
	29.16	Gloves, knives, tape, calipers	Purchased
	107.84	Nylon nuts and brackets for frame	Purchased
	300	Outsourced machining	Purchased

Total	**1942.29**		
Controls Purchases	48.83	Arduino Mega	Purchased
	804	6 thrusters and speed controllers	Purchased
	35.04	Xbox controller	Purchased
	90.05	Robotic arm	Purchased
	112.15	Ultrasonic sensor	Purchased
	347.21	USB extender, a-b cables, arduino mega	Purchased
	55.97	Wiring supplies	Purchased
	4.97	Dupont wire jumper kit	Purchased
	35.47	Bullet connectors and heat shrink	Purchased
	42.56	Soldering station and tape	Purchased
	227.45	4 new batteries and a charger for testing	Purchased
	928.2	Underwater endcap connectors	Purchased
	23.84	Glues and shrink wrap	Purchased
Total	**2755.74**		
Tether Purchases	69.02	80' tether sheeth	Purchased
	29.79	100' power cable	Purchased
Total	**98.81**		
Travel Purchases	1136.96	Hotel in St. John’s for 8 team members	Purchased
	4852.2	Flights BOS-YYT for 8 team members	Purchased
	379.84	Rental car for stay in St. John’s	Purchased
Total	**6369**		
Total team expenses	**11165.84**		

System Integration Diagram

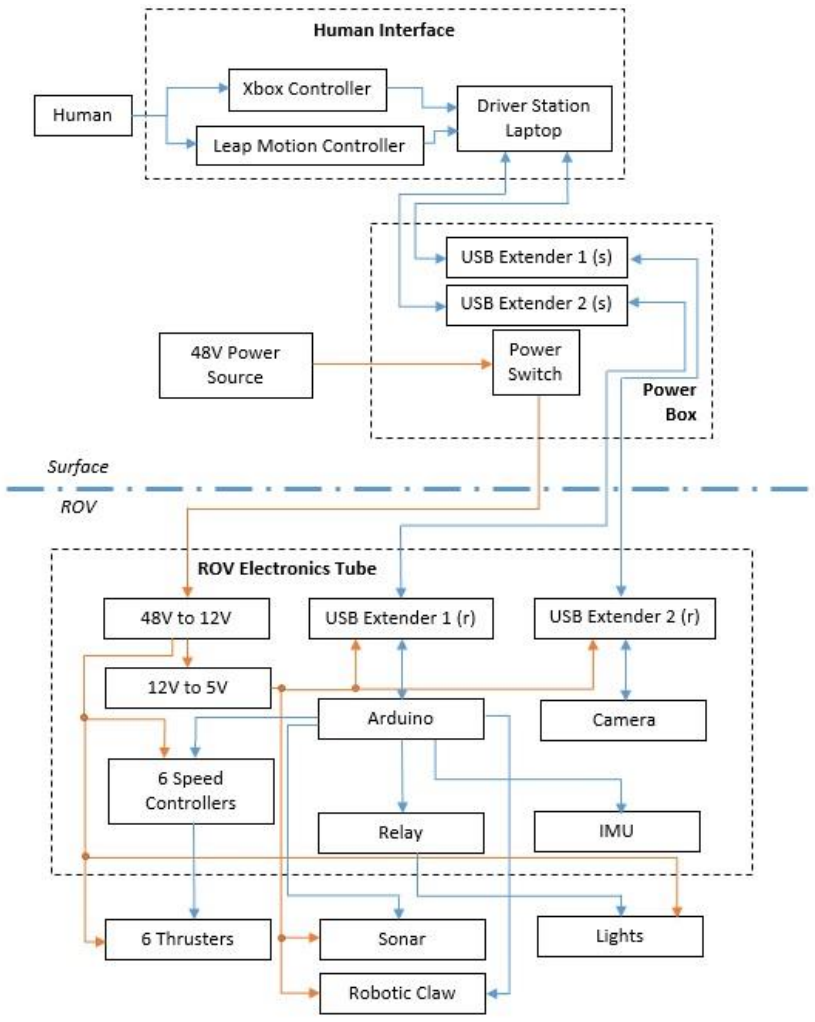


Figure 5: The system integration diagram (SID) including the human interface (surface), power box (surface), the ROV electronics tube (ROV), and the external components on the ROV (ROV).

Design Rationale

Initial Design

The overall design was inspired by other ROVs that are manufactured worldwide. Most ROVs have 6 degree of freedoms to control movement in all orientations. This year's design allows for 5 degrees of freedom and mobility, excluding roll. Outside of the thruster design to ensure mobility, it is desired to have unrestrained fluid flow through the frame of the chassis. This was a major concern as fluid flow enables the ROV to push water, which according to Newton's 3rd Law, will push the ROV back allowing restricting the ROV from maneuvering through the water.

Another design attribute was to create a compact size with few failure points. Limiting failure points is one of the most important and emphasized design aspects. One failure point is waterproofing. This year’s design addressed this by building the chassis around a single enclosed tube that would house all of the electronics in a single dry environment. This limited the points of failure to two, the electronics tube end cap and the dome on the front of the electronics tube. As for the size of the chassis, it was ideal for the design to be a size fit for traveling, as it will be traveling to the MATE ROV Competition. In order to transport the ROV to the competition, it is best to keep the overall chassis as compact as possible.

Cost Schedule and Risk Analysis

Each product demonstration conducted during the competition can receive a maximum of 100 points, for a total of 300 points across all demonstrations. Mission tasks can be very elaborate and may demand the incorporation of a component not normally found on the ROV, except to only complete that specific task. Such is the case of measuring the grounding of anodes during the third demonstration. However, something simple like a hook can be utilized to complete many of the tasks. Thus the hook is more valuable than the underwater multi-meter attachment in terms of accruing points, while being much less costly to implement. Given time constraints during the academic year, the budget restraints from a finite set of donors, and the travelling costs to participate in the competition; not all mission tasks at the MATE competition can be feasibly completed by the UNH ROV team this year. As such, each demonstration task has been broken down into their respective points and weighed on the expense it would cost to pursue the points.

Through analysis of the competition, the UNH ROV team was able to give priority to certain ROV subsystems to effectively complete mission tasks. This was done by adding up all the points for each mission task that would require use of specific category of subsystem.

Table 2: Point value analysis

Name of Subsystem	Number of Points	% Total From Product Demo
Manipulator	110	36.7
Camera	80	26.7
Sonar	60	20.0
Voltmeter	40	13.3
Flow Sensor	10	3.3

The UNH ROV team decided to pursue the top three most heavily weighted subsystems to compete with in the MATE competition. However in further analysis of the requirements of each specific mission task, it was determined that a single manipulator would not be sufficient enough to obtain all of the points relevant to the manipulator subsystem. In the mission tasks, there is a wide range of weights and geometries that the manipulator subsystem must be able to handle. It would be too costly to develop a single robust robotic manipulator to be able to lift the heavier objects as well as have the precision for the objects with more complex geometry. It would also involve too much scheduling time that would take away time from the other significant portions of the ROV. And lastly, having a more complex manipulator increases the number of both hardware and software based failure modes.

Two separate manipulators were developed in addition to the static hook: the robotic manipulator and the static manipulator. With respect to the mission tasks, heavier objects which require a robust design make up 45.5% of the manipulator-related mission tasks while lighter objects with more complex geometry make up the remaining 54.5%. Because the quantification of these manipulator based mission tasks are nearly equal in point value, it was deemed necessary to split the tasks between two more specific manipulator subsystems. The cost of using these two systems may at first seem greater than that of one manipulator, however leveraging the design of last year's robotic manipulator alleviated most of the cost of prototyping and designing a new manipulator. With respect to schedule it was unnecessary to allocate man hours to design a new prototype. In contrast, from the robotic manipulator, the static manipulators are relatively rigid apart from fasteners or springs. They do not require a power source to operate, and this reduces a potential mode of failure for the appendage.

The camera quality was of the next highest priority given that it was the next highest in the point value analysis. An off-the-shelf Microsoft HD Lifecam was chosen as the camera of choice, given its low cost and 1080p quality. Furthermore there is little risk in using this camera because the drivers automatically worked with the Microsoft based operating system for the driver station. Also because of the camera's point priority, the camera feed was given its own isolated USB extender to minimize loss of resolution.

The last subsystem that was implemented on the ROV is the sonar. The sonar of choice was the MaxBotix 7076 sonar system. This sonar system was very easy to further waterproof, considering it was already water resistant with a rating of IP67. Because of the all-in-one nature of this sensor, it is very easy to use and therefore reduces the amount of time it takes to implement the sensor.

The remaining subsystems in the point value analysis were deemed not worth the overall cost, schedule, and risk associated with implementing them on the ROV design. This is because of their low point value and the time that it would take to develop and prototype these subsystems from scratch. In conclusion, the point value analysis paired with a cost, schedule, and risk analysis enabled the UNH ROV team to make efficient and worthwhile design decisions in developing this year's ROV.

Chassis and Frame Design

To begin modeling the ROV, three dimensional computer aided design (3D CAD) software was used. At UNH, students are provided with an all-in one CAD software package called SolidWorks. SolidWorks allows for CAD parts and assemblies to be designed and modeled. Additional SolidWorks also allows for drawings, animations, and simulations to be completed all within the same file regime. UNH ROV utilized this software extensively to complete the modeling, designing, and simulating for this project, due to its intuitive nature and well as its ease of access for UNH students.

Basic constraints derived from the MATE ROV Competition design requirements and Graduate Research requirements were used to develop the initial design for the ROV. First and foremost, a neutrally buoyant frame is the best configuration for maneuverability to achieve the maximum propulsive force from the thrusters. However, due to the small possibility of multiple failure modes within the ROV, such as a water leak or loss of communication from the driver station, the frame is designed to be slightly positively buoyant when fully equipped. If for some reason the ROV stopped functioning while under deep sea conditions, the positive buoyancy would bring the ROV up to the surface for easy recovery. The magnitude of the positive buoyancy is very slight, and close to neutral in order to retain the maneuverability that is demanded.

Another general constraint is in order to maximize the resulting velocity from a given propulsive force, the ROV must displace the least amount of water possible along its movement. Thus there should be an absolute minimum of surface area that obstructs water flow through the ROV, yet retain structural integrity to withstand the propulsive forces without any deformation. A smaller frame that still has minimal solid obstructions is ideal, yet still places the thrusters far enough apart to ensure maneuverability in the respective degree of freedom. The ROV's dimensions are 30.48 x 46.99 x 50.8 cm.

Symmetry of the ROV is also necessitated due to cost effectiveness and ease of machining the components, as well as keeping the ROV's movement intuitive to the symmetric human interface. Furthermore, symmetry guarantees that the water displacement during movement is equally separated across the ROV and there is no maneuverability loss between opposite directions on the same degree of freedom.

In order to minimize the surface area attributed to buoy attachments that keep the frame slightly positively buoyant, the frame must be constructed out of the least dense materials without being overly costly. Therefore Aluminum 6061 was chosen for the endcaps of the electronics tube, the thruster mounts, as well as for the U-channels that make up the cross-member supports. The side panels are made out of polycarbonate and cutouts are machined in the side panels to minimize the obstructing surface area during translational movement. Finally, the single electronics tube is made out of acrylic. All frame components were machined at UNH except for the endcaps which were machined by Eastern Precision Machining. This frame composition is strong enough to withstand the propulsive forces while being less dense than equivalent-cost alternatives, allowing a minimal amount of buoyancy additions to be required, which improves the overall agility of the ROV.

The final design for the frame was heavily influenced on being modular. Modularity of the chassis is incredibly important to the success of this project due to the variety of different sensors and manipulators required to complete all MATE mission tasks as well as housing the photodetector array for graduate research. For competition, the ROV must be able to support one robotic claw, one static manipulator, and a hook attachment. Both manipulators needed to be easily accessible during competition, since they will be swapped out between product demonstrations. Both appendages attached to the ROV at all times would inhibit the movement of the ROV as well as create more modes of failure.

Flow Analysis for the Frame Design

From the detailed MATE manual it was determined that the ROV would need to maneuver through water with a max velocity of 1mph (17.6 in/s). Taking into account these parameters and the costs of the different thrusters a decision needs to be made for the final design for the ROV.

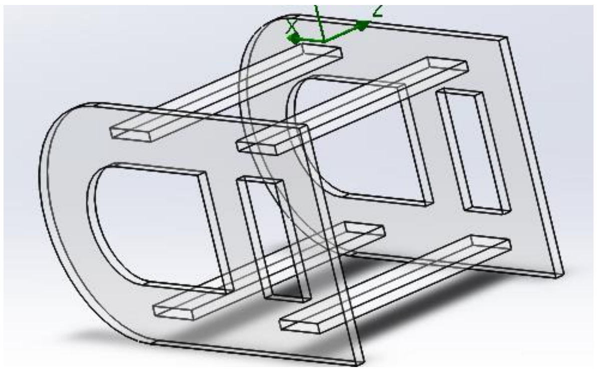


Figure 6: SolidWorks representation of the simplified

Using SolidWorks, the ROV design represented in was simplified and remodeled, as seen in figure 5. Due to capabilities of calculations, the model was simplified to just the frame of the ROV. The model was simplified in order to compare the SolidWorks Flow Simulation calculated drag force with hand calculations. It would be nearly impossible to determine the drag coefficient while incorporating components such as bolts and screws. The tubular mid-section was also excluded as the focus of this study was to evaluate the flow through the frame of the chassis. Another study was conducted to evaluate the fluid disruption caused by the tubular mid-section.

Using SolidWorks, flow simulations were conducted to determine the drag force acting on the model in a sideways motion (Flow in the direction normal to the side plates). The unit system defined in the simulation is IPS (in-lb-s). The simulation is an external analysis with closed cavities and without any physical properties. Liquid water was chosen as the fluid because the ROV will be driven in aquatic conditions. The roughness of the model will be considered zero, as most plastics are very smooth and therefor negligible. It was determined for modeling purposes that the operation depth would be at 10ft with a pressure of 19.1396 lbf/in^2. The ROV will also be operating in a pool with a temperature of 65°f. The velocity is acting in the Z direction into the face of the plate. Different velocities were input for a series of trials to determine the force acting on the model, seen below in figures 8 and 9.

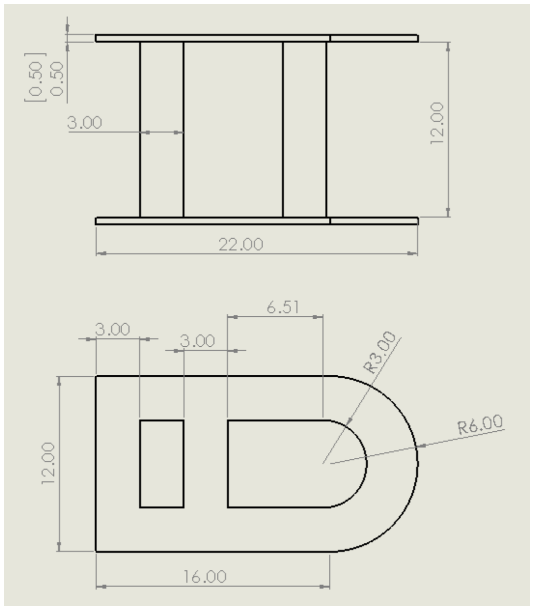


Figure 7: Schematic of model [4]

A plot showing the simulation flow trajectories is shown in figure 7. The max tested velocity interacts with the frame at 22.0 in/sec. Once the flow contacts the side plate the simulated flow of water is mainly dispersed around the outer perimeter of the frame. It can be seen that water will pass through the interior of the frame. Note there is a decrease in velocity and flow gets trapped between the two plates.

The velocity that interacts with the first side plate was needed to determine the maximum force acting on the frame. This simulation plot was conducted for each velocity to analyze the optimum velocity and force to operate efficiently.

A cut plot was also used to visually analyze the velocities acting on the frame. The velocity acting on the midsection of the frame is presented in figure 9, with a top-view orientation.

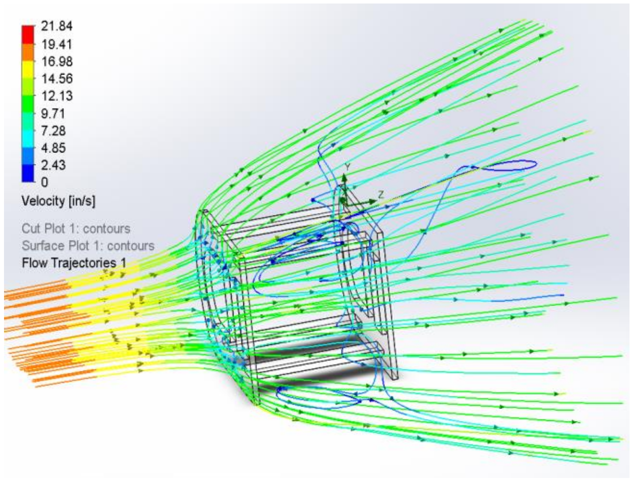


Figure 8: Flow simulation trajectories for a velocity of 22.0 in/s. [4]

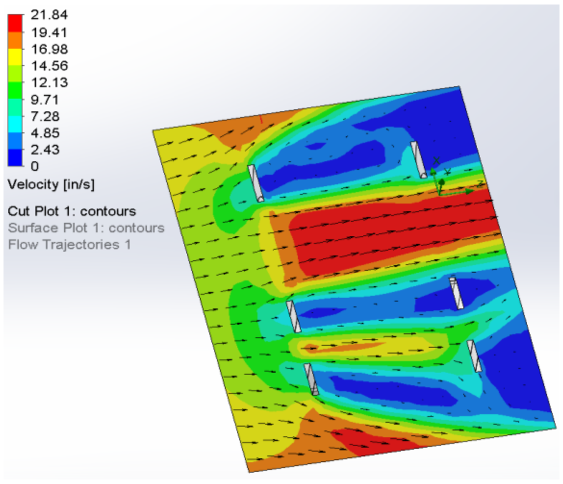


Figure 9: A top-view midsection velocity cut plot for a velocity of 22.0 in/s [4]

A global goal was set to determine the force in acting on the face of side plate. The Goal was calculated for each velocity and recorded in Table 3.

Table 3: Goal plot result [4]

Velocity (in/s)	Force [lbf]
4.4	0.355091389
8.8	1.438678674
13.2	3.292741521
17.6	6.995641254
22.0	9.225133453

In order to validate the calculations from SolidWorks simulation hand calculations were performed. The drag force acting on an object can be defined by equation (1):

$$F_d = \frac{1}{2}\rho v^2 C_d A \qquad \text{Equation 1}$$

Where,

F_d is the total drag force acting on the object
ρ is the density of fluid
v is the velocity of object relative to the fluid
C_d is the drag coefficient of the object
A is the projected area of the object that sees flow

For the ROV's purposes, the final drag force was the desired value. The density used was the density of water at 65 degrees due to MATE specifications. The velocity used was varied from 0.25-1.25 MPH (4.4-22.0 in/sec) to get a graph of the drag forces at various velocities, but most importantly reaching and exceeding the max required velocity as specified by the MATE competition. The drag coefficient was modeled as a rectangular box. This is because the configuration of the chassis can be simplified to a square box with various holes cut out which inhibit flow, giving the design a large coefficient of drag.

The area was calculated using one of the side plates since this is the projected area seen by the flow. The drag forces were calculated and tabulated in Table 4.

Table 4: Hand calculated values of drag force.

Velocity (in/s)	Force [lbf]
4.4	0.331365
8.8	1.325315
13.2	2.98223
17.6	5.301765
22.0	8.283941

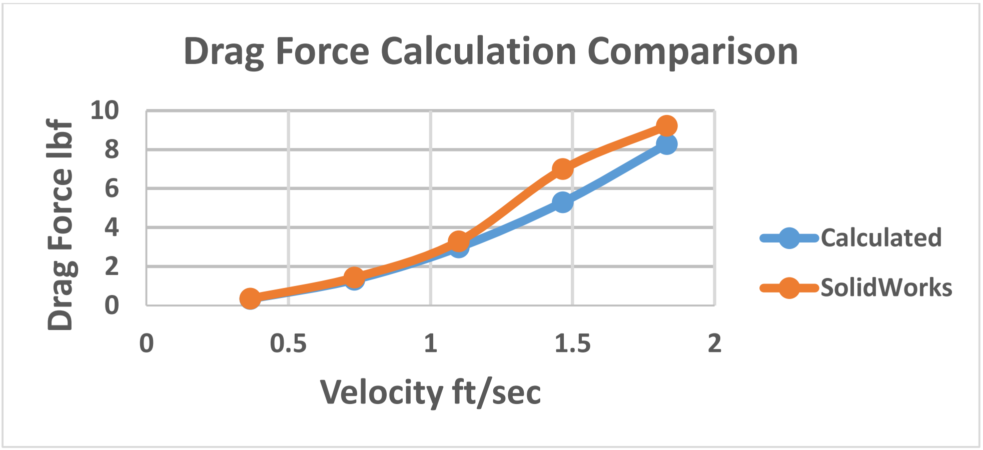


Figure 10: Graph Comparing the Calculated and Simulated Drag Force

Table 5: Table of Percent Errors between Hand Calculations and Simulations

Velocity (in/sec)	Hand Calc F_d (lbf)	SolidWorks F_d (lbf)	Percent Error
4.4	0.331364862	0.355091389	6.681808581
8.8	1.325314858	1.438678674	7.879717547
13.2	2.982229536	3.292741521	9.430196167
17.6	5.301765497	6.995641254	24.21330219
22	8.283940811	9.225133453	10.20248268

The MATE competition specifies that the ROV must reach a maximum speed of 17.6 in/sec (1 MPH). Hand calculations shows that the force on the side plates of the ROV, resulting from the 17.6 in/sec velocity would be 5.302 lbf. This value is expected to be a rough estimate due to the non-exact approximation of the drag coefficient. The results of the drag force from the SolidWorks Simulation resulted in a required thrust of 6.99 lbf. It must be noted that this value may not match perfectly with real life drag forces. This number is sure to be close though, as all of the assumptions included are fitting for the purposes of the operations of the ROV. Ultimately since two thrusters are oriented for each translational direction of motion, the Blue Robotics T100 thrusters are perfectly capable of handling these drag forces with a factor of safety nearing 2.

Propulsion Design

A propulsion system allows the ROV to utilize underwater thrusters in a specific configuration to grant operational range of motion. The chassis team defined that the ROV requires 5 degrees of freedom to successfully complete all mission tasks during competition. In order to achieve this range of motion, a six thruster configuration was designed. Thrusters were placed on the center axes for their respective jobs. Ultimately the 8 thruster configuration provided X, Y, and Z translational motion, as well as pitch and yaw rotation.

Figure 11 – Blue Robotics T100

Figure 12 – Seabotix BTD-150

Seabotix BTD-150 and Blue Robotics T100 thrusters were evaluated in order to determine the best means for propulsion. The Seabotix thrusters operate using brushed DC motors, require 19 VDC, have a 4.25 maximum continuous amperage rating, and can operate at a maximum depth of 150 meter. The Blue Robotics T100 thrusters operate using brushless DC motors, require 12 VDC, have an 11.5 maximum continuous amperage rating, and have no distinct depth rating due to the open motor design. The cost of the Seabotix brand thruster is $780 and the cost of the Blue Robotics brand thruster is $144.

Parameters defining thruster performance include data gathered from steady-state and transient responses; rise time, peak time, settling time, and thrust force per given percentage of voltage input. The experiment utilized the NI-DAQ digital SoftScope and DAQ interface to measure the voltage output of an Interface SM-25 S-Type Load cell to determine the force applied to a rigid vertical member by the thruster in a 20 gallon fish tank.

The Steady States of both the Seabotix BTD-150 and the Blue Robotics T100 were measured by recording the input voltage given to each motor and comparing that to the output reading on the strain gauge. To accurately compare both motors together, a normalization of the input voltage was used. This was done by changing the caparison from voltage input to the motor, to percent voltage input to the motor. For the BTD-150 the 19.1 maximum voltage was divided evenly into percent of inputs. Therefore 10% voltage input for the BTD-150 actually produced a 1.91 V voltage input, while the 10% input for the T100 corresponded to a 1.2 V voltage input. This normalization allows UNH ROV to compare both motors together without giving one an advantage over the other.

A portable 20 gallon fish tank was used for the experimentation. The small testing platform caused some limitations in terms of the testable range of the thrusters. To avoid introducing disturbances in the readings of the force transducer, the input voltage for the experiment was limited to a range between -40% and 40%. A performance relationship can be produced for each motor by directly comparing the ratio of the output thrust and the percent voltage input. Although the range of input voltages was limited due to the experimental setup, it is likely the thrusters will not be used to their full potential on the ROV, in order to allow for precise movements.

Both forward and backwards thrust for the Seabotix BTD-150 and Blue Robotics T100 thrusters were produced and are graphically represented in Figure 13 and Figure 14 respectively.

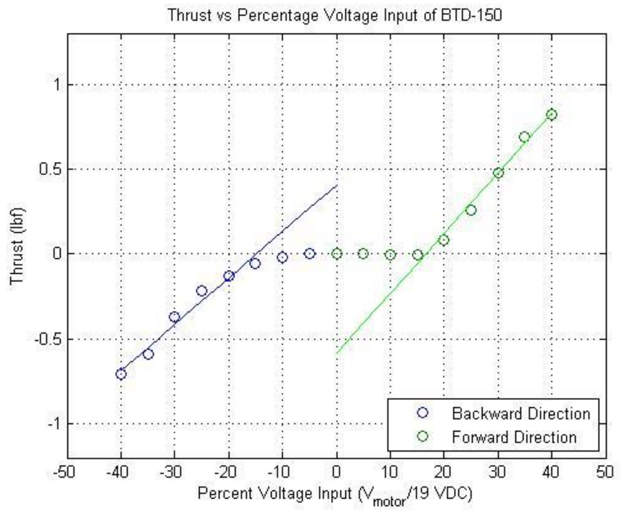


Figure 13: Output Thrust vs Percent Voltage Input of Seabotix BTD-150

Figure 13 shows the thrust output of the Seabotix BTD-150 thruster. The blue region of the graph represents the backward thrust of the motor, and the green region of the graph represents the forward thrust of the motor. It is important to note that both the -5% and the 5% input voltage readings returned a thrust output of 0 lbf due to a "dead zone" programmed into the Arduino code. The programmed 'dead zone' is necessary in order to reduce accidental excitation of the thrusters on the ROV (controlled using an Xbox controller joystick which is never at a true neutral position).

The linear representation of the collected output thrust and percent voltage input data is determined in order to compare the thrusters using the performance ratio. The forward and backward thrust performance lines have slopes of 0.0352 lbf/%V_{in} and 0.0274 lbf/%V_{in} respectively. The recorded forward and backward thrusts were fairly similar in magnitude, although the performance slope suggests that the thrust in the forward direction reacts better to increased or decreased input voltages. It was found that there is a 22.16% difference between the performance of the forward and backward thrust capabilities of the Seabotix BTD-150 thruster.

Figure 14 represents the plotted data of the thrust output for the Blue Robotics T100 thruster. The same methodology was applied to the T100 as the BTD-150. The forward and backward thrust performance lines have slopes of 0.0429 lbf/%V_{in} and 0.0366 lbf/%V_{in} respectively. Similarly with the BTD-150 thruster, the T100 thruster also exhibits a better forward thrust performance. The difference in performance between forward and backward thrust capabilities was determined to be 14.69%. It was concluded from the results of the experiment that the Blue Robotics T100 thruster's output thrust magnitude was larger than the Seabotix BTD-150 thruster per percent voltage input. The differences between the thrust output per given percent voltage input for each thruster were calculated to be 33.57% for the backward thrust and 21.88% for the forward thrust.

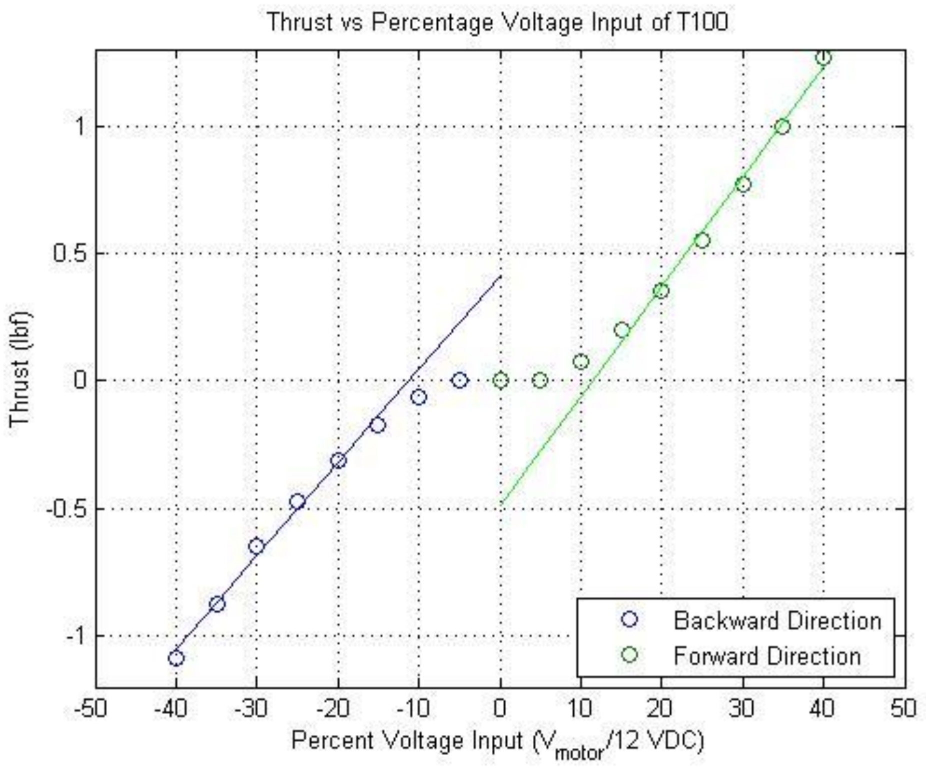


Figure ~~-~~14~~:~~ Output Thrust vs Percent Voltage Input of Blue Robotics T100

Transient responses were performed in order to poll data from the dynamic systems. Both of the motors were modeled as second order systems therefore, each motor will have characteristics of second order systems. In order to collect data from the thrusters to compare their responses, the thrusters were connected to the experimental setup used for the steady-state input recordings. In order to record the transient response of both thrusters, the output thrust was recorded with respect to time. The transient response for the T100 thruster was recorded using a step input from 20% thrust to 40% thrust. The BTD-150 thruster was recorded using a step input from 20% to 40% thrust as well.

This was achieved with the BTD-150 thruster by starting with the voltage at 20% of its potential power and then spiking it by turning the dial on the voltage supply provided in the lab. Once 20% voltage was stable in the tank, the dial was spun to a voltage nearest the 40% mark for the BTD-150 thruster. This step input into the system resulted in the following plot seen in figure 15 below.

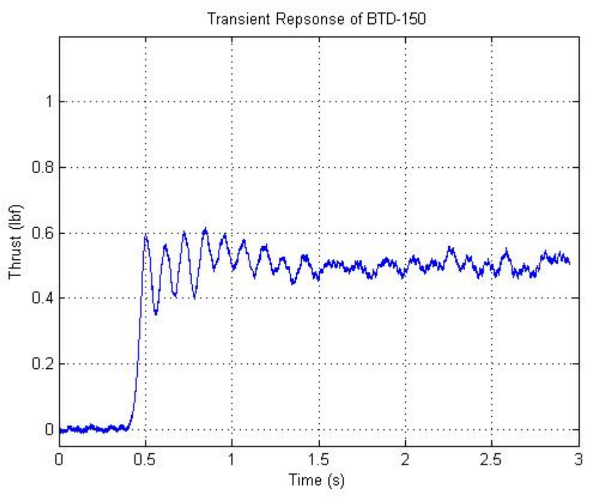


Figure 15: Transient response of the BTD-150 thruster

The step input of the T100 thruster was achieved by prompting the Arduino to control the thruster from 20% thrust to 40% thrust. Represented in figure 16, the T100 thruster generated a much larger overshoot than the BTD-150, suggesting the damping ratio of the BTD-150 thruster is slightly larger than the T100 thruster. The overshoot, although generally unwanted, aids in providing a quick response to reaching peak thrust.

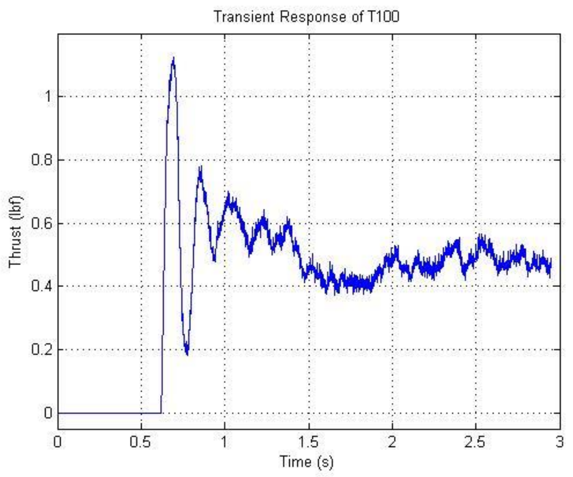


Figure 16: Transient response of the T100 thruster

These transient responses are very important statistics that relate to the build of the UNH ROV. This is because the rates of response are critical to determining which motor will perform better when trying to maneuver precisely. These fine movements will be necessary in many mission tasks at the MATE international ROV competition. Many factors play into the decision to use either the Blue Robotics T100 thruster or the Seabotix BTD-150 thruster including cost, thrust capabilities, performance with respect to voltage input, and percent overshoot given a step input. Because the thrusters will be changing 'speeds' constantly, it is important they do not overshoot their goal. The transient response data can be used to create a better control system for the ROV as well as allow the driver to react to known flaws in the propulsion system.

The rise time of the thrusters provides information about how each thruster initially reacts to a step input. ROV thrusters will be subject to quickly changing speeds and directions, therefore an understanding of how quickly they reach the maximum thrust at the given final voltage input will allow the driver to better react to the propulsion system, similarly to the overshoot. The settling time refers to the time at which the response settles to a value ranging within 2% of the steady-state value. This information is important when changing from one constant speed to another. By knowing the settling time of the thruster, it is possible to understand better how much thrust is being executed; therefore, a better estimation of the velocity of the ROV can be calculated.

It was found that the rise time of the BTD-150 thruster was .0944 seconds, while the rise time of the T100 thruster was .0209 seconds. A quicker rise time of the thruster will result in a quicker response of the robot, though it may result in overshoot. As seen in Figure 10, the T100 thruster has considerably more overshoot than the BTD-150 thruster. The T100 had a max percent overshoot of 90.23%. The BTD-150 also had a varying amount of overshoot. The percent overshoot seen with the BTD-150 was 26.45%. For UNH ROV it was necessary to decide whether or not the system would desire overshoot. It is thought that overshoot in this scenario would hinder the capabilities of stable driving. Therefore, in this case it is best to go with the BTD-150 which has less overshoot for more precise driving.

The Seabotix BTD-150 and Blue Robotics T100 thrusters both performed well throughout the experimentation. The performance of the thrusters were compared using a common ratio between the two; the relationship between percent input voltage and total output thrust. Based on this performance relationship, the T100 thruster performed better both forward and backward than the BTD-150 thruster by 17.95% and 33.57% respectively. It was found that the output thrust of the T100 thruster was higher than the BTD-150 thruster for identical percent input voltages, based on the recorded experimental data. This relationship is important for determining the capabilities of the thrusters in comparison to the provided datasheets from the respective companies. Because the output thrust per percent input voltage was found to be greater with the T100 thruster, it was determined that the thruster provided necessary output to be viable as a replacement for the BTD-150 thruster on the ROV propulsion system.

Control Design

Drive Control

The ROV is controlled from the surface using an XBOX 360 controller as well as the Leap Motion Controller to make driving intuitive. The digital values from the controller are read in to the driver station and are subsequently sent to the Arduino using serial communication. An Arduino script then utilizes the controller values to control different thrusters based on the desired movement. The left analog stick moves the ROV translationally, while the right analog stick controls pitch and yaw motions. The right and left triggers cause the ROV to ascend and descend, respectively.

Pressing the right bumper on the XBOX 360 controller initiates Leap Motion Control, which is used to control the external manipulator. The Leap Motion Controller maps a user’s hand in 3D space using infrared light. Activating Leap Motion Control caps the thrusters to 5% thrust for fine-tuned movement of the ROV. The ROV control system utilizes the palm velocity function of the Leap Motion SDK to control each degree of freedom that the XBOX 360 controller can. Of course, the XBOX 360 controller is the master controller in the HID hierarchy and overrides any Leap Motion controller input as a safety mechanism.

GUI

Since the view of the ROV will be obstructed, the Driver Station is equipped with a Graphical User Interface (GUI) to assist the driver while maneuvering. The GUI is programmed in Java and acts as a Heads-up Display (HUD), containing different elements that alert the driver to the status of the ROV. A high definition camera feed is displayed in the background, using video data from the webcam in the dome of the ROV which is transferred through the tether. Overlaid on top of the camera feed is a table that displays values from the sensors and measurements in real-time. A cube is also displayed over the video stream to show the driver the orientation of the ROV and direction that it is currently traveling in. Graphics are also displayed to show the status of the dive light, as well as the manipulator. The GUI is also used to measure objects using different methods.

The video stream is the primary method of precise navigation for the driver and acts as a platform for the multiple image processing capabilities of the GUI. The GUI's image processing capabilities were developed using the Open Source Computer Vision Library (OpenCV), an open-source image processing library that has support for Java. OpenCV provides a function to obtain video data from webcam peripherals. After connecting to the desired camera, individual camera frames are obtained as matrices. The matrices are converted to a Java BufferedImage type so that the image is easy to manipulate. BufferedImages are represented using 3 bytes, using 1 byte for red, green, and blue colors. When each frame is converted to a BufferedImage, the GUI is updated so that the current frame is shown. The frames are sequentially shown at a rate of 30 fps to create the video. Depending on the size of the screen and the supported resolutions of the connected webcam, the resolution of the camera stream can be changed to provide video using 1280x720 pixels (720p) or 1920x1080 pixels (1080p).

Lag occurred in the GUI when video streaming was originally implemented due to the large amount of computations being done simultaneously. In order to prevent lag and allow for concurrency, multithreading was added. Instead of using the Java Thread class, the SwingWorker class was used because it handles cases that are specific to Swing, one of the Java libraries used to create user interfaces. The Swing library utilizes three types of threads in concurrent programs: initial threads, the event dispatcher thread, and worker threads. Swing events occur on the event dispatcher thread, so the video capture and other processes that require their own threads are executed using worker threads. When worker threads finish, code is required to merge them with the event dispatcher thread, as well as thread cleanup. SwingWorker intrinsically executes cleanup to improve the clarity and simplicity of code. Each task that is responsible for a large amount of computations is run in its own worker thread in order to ensure that the GUI remains responsive.

The user interface is designed so that the video stream will be displayed in varying resolutions in the back of the frame used for the GUI. Various components and interactions occur on a transparent panel (called a glass pane) that is placed over the video. The first component that's placed on the glass pane is a semi-transparent table, which displays different measurements and sensor values for the robot.

The depth of the robot is the first value that's displayed in the table, which is obtained from the pressure transducer sensor. Since the values from the sensors are obtained through the C++ program that also runs on the Driver Station, Transmission Control Protocol (TCP) was implemented in the Java program to communicate with the C++ program. The Java program running the GUI acts as a TCP server, while the C++ program acts as a client. The C++ program sends the relevant sensor values as packets in a specific order, which the Java program receives and displays. Since the programs are executed on the same computer, the IP address remains constant at 127.0.0.1 (localhost), while the port can be changed.

User Datagram Protocol (UDP) was also considered as a means of communication between the programs, since data is continually sent while the ROV is on. While UDP continually sends information and is intrinsically simpler than TCP, UDP provides no guarantee that information will arrive correctly and sequentially. Since the GUI is the only way to accurately determine the status of the ROV, it's important to receive accurate information. TCP was ultimately used instead of UDP because of the drawbacks that UDP possessed.

In addition to the depth of the ROV, real-time object measurements are displayed in the table, including the distance to an object, the height of the object and the width of the object. The distance to the object is obtained by using a sonar which is connected to the ROV. The sonar sends out a pulse and outputs a voltage that corresponds to the distance. The sonar information is sent to the C++ program from the Arduino, so it's also sent to the GUI through TCP communication. One of the mission tasks for the MATE competition will be to measure underwater objects, which is why the height and the width of objects are placed in the table. Using the distance provided by the sonar, the GUI can measure objects in multiple ways. One method is for the driver to draw a bounding box around an object by dragging the mouse. Since the field of view of the camera is known, the pixel distance of the height and width of the bounding box is used in conjunction with the distance obtained from the sonar to calculate the dimensions of the object.

Components other than the table are located on the GUI to alert the driver to the status of the ROV. A cube was created using the Java binding for the OpenGL API (JOGL), a library for 2D and 3D graphics. The cube gives the current orientation of the ROV and the direction that it's going. Since the view of the ROV will be obstructed, the driver will have to rely on the camera to maneuver the ROV, relying on reference points and objects. If there are no reference points, the driver can utilize the orientation cube, the depth measurement and the dimensions of the pool to determine where to go.

Graphics also show the driver the status of the Leap Motion Controller and the dive lights. Using JOGL, a graphic was created that resembles the dynamic robotic manipulator that the driver can controller using the Leap Controller. The graphic changes depending on whether the manipulator is open or closed, which tells the driver what needs to be done to grab an object. The C++ program controls the Leap Controller, so it continually sends a Boolean value to the GUI through TCP, which says whether the manipulator is open or closed, as well as an integer value reporting on the current rotation of the manipulator.

An image is also displayed on the GUI that resembles a light. The color of the graphic changes based on the current status of the dive light: yellow signifies that the light is on, while white signifies that the light is off. The dive lights are controlled by the XBOX 360 controller through the C++ program, so the status of the light will also be sent in a packet to the GUI through the TCP connection.

Ultimately, the graphical user interface is the primary connection that the driver has with the ROV. The GUI provides real-time status updates about the robot, including video; object measurements; sensor values; ROV orientation; and the statuses of the manipulator and the dive lights. Using Transmission Control Protocol allows the GUI to communicate with the C++ program that runs on the Driver Station to obtain pertinent information for calculations and measurements for the GUI's components. Concurrency was added to ensure that the GUI remains responsive. Computationally-intensive tasks such as video streaming and server communication are assigned their own threads to ensure that they don't collide with other tasks. The GUI is one of the several human interface devices that is used to help the drivers operate the ROV in an intuitive and easy way.

Robotic Manipulator Control

As part of UNH ROV's goal in bringing seamless integration of humans and robotic technologies, the Leap Motion controller was chosen as the controller to control to the robotic manipulator. It creates a three-dimensional representation of a human hand and maps specified values to the servos on the robotic manipulator. This creates a very fluid and intuitive interaction with between the human pilot and the ROV, thus allowing for a more natural method of remote object manipulation.

The Leap Motion controller is a small, commercially available product use for development of virtual reality technology. It uses 3 infrared (850 nm) LEDs and 2 cameras to shine an array of infrared dots and uses the cameras in the sensor to detect the three-dimensional position of each joint of the skeleton of a human hand. The sensing range of the device is limited to only 2 feet above the device, but boasts an impressive 200 Hz data refresh rate through USB 3.0 and a one-hundredth of a millimeter accuracy. The Leap SDK offers a wide range of tools for developers to use in many different programming languages, which is what the UNH ROV team took advantage of.

This year's robotic manipulator is the MKII Robotic Claw from DAGU. It features two degrees of movement that represent a wrist motion and a pinching motion. Each metal gear servo rotates 180°, which allows for the same freedom as seen with a human wrist and pinch. To prevent damage to the pinching servo, a spring–loaded clutch is included to allow the claw to close completely without the servo burning out.

The Leap Motion Sensor then is mapped using the corresponding joints in a human hand. Using the Leap SDK, the values for palm rotation about the center of the palm, while using the thumb as reference, are mapped directly to the servo values for the wrist movement of the robotic manipulator. Next, the Cartesian coordinates for the tip of the index finger and thumb are retrieved from the Leap SDK and the three-dimensional distance is calculated and mapped directly to the pinching servo of the robotic manipulator.

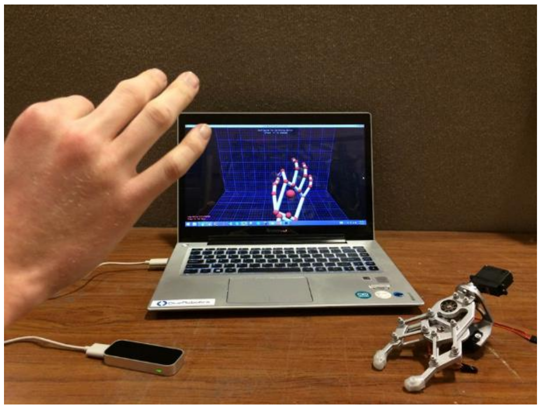


Figure 17 - Leap Motion controller

IMU Feedback for Mission Tasks

The Inertial Measurement Unit (IMU) and pressure transducer are integral parts of the ROV control system. These sensors are able to give critical information to the Pilot concerning the orientation of the ROV as well as its depth underwater. Feedback control is also important because it alleviates work that the Pilot must do to keep the ROV in a stationary position.

The MPU-6050 IMU measures 6 degrees of freedom utilizing both the MEMS accelerometer and MEMS gyro contained in a single chip on the GY-521 breakout board. It contains a 1024 byte FIFO buffer as well as a Digital Motion Processor (DMP) which acts as a small computer on board the breakout board to do fast calculations of the read outs from the MPU-6050 chip. The DMP is accessed by the ROV to read filtered acceleration and angular displacement values and use it in pitch feedback control.

The ROV requires pitch feedback control because during each mission task, the ROV may be picking up different items with varying weights, which the weights for ballast system cannot account for as the ROV is underwater and ballast to a specific load. In this case, the Pilot can activate an auto-leveling function on the Xbox 360 controller where the ROV uses Sliding Mode Control (SMC) to pitch the ROV so that the gyroscope vale for pitch reads 0°. This auto-leveling function remains active as long as the Pilot does not use additional pitch control, which is controlled with the up and down motions on the right analog stick. This overrides the auto-leveling function until the right analog stick is returned back to its neutral position, where the auto-leveling function takes over. This mode can be toggled on and off with a push of a face button.

The pressure transducer is used for depth feedback control of the ROV. This is necessary because the ROV is designed to be positively buoyant as an emergency recovery system in case communication to the ROV fails. Due to this, the Pilot would need to continuously be descending with the ROV by holding down the right trigger on the Xbox 360 controller. To alleviate the Pilot of this task, the pressure transducer sends a voltage corresponding to the pressure at which it is exposed to. Using hydrostatics, the depth in water can be found and can be utilized by the control system. Again, Sliding Mode Control (SMC) is used to maintain constant depth while the Pilot does not have to ascend or descend triggers active. Once they become active, this mode is overridden by the Pilot’s input, just like the pitch feedback control.

These control feedback methods are extremely helpful to the Pilot so that he can focus on orienting the ROV to complete mission task without worrying about these degrees of freedom. This limits the complexity of operating the ROV in critical, mission task situations.

Sonar Feedback

The Sonar system offers another place to implement intelligent controls as well as image processing. It is necessary to measure the size and therefore distance of objects underwater for certain mission tasks to receive 20% of the total points for mission tasks. The Sonar system works in conjunction with the camera and the HUD to give real-time size and distance data to the Pilot.

The Sonar system used on the ROV is the MaxBotix 7076 Sonar system. It is an all-in-one system acting as both the transmitter and receiver of the 42 kHz acoustic signal. It is designed for outdoor distance measuring in mal-weather, with an Ingress Protection Rating of IP67, but can easily adapted for underwater conditions by epoxying the leads and using a silicon based sealant around the transducer.

The ROV control system reads the voltage from the Sonar, which correspond to the distance an object is away. This calibration constant was found by placing objects in the water at given distances and measuring the slope of the line created by those data points. With the calibration constant in place, the Sonar sensor has a 10 cm resolution over a range of 25 meters in water.

To measure the size of objects underwater, the data from the Sonar is used in image processing performed by the Driver Station of the surface. The GUI receives human input by identifying two pixels at the ends of a line by clicking, dragging and drawing a line between two points of an object underwater on the video feed. The GUI takes these two pixel locations, the field of view of the camera, and the distance to the object and uses trigonometry to find the length of the line and the corresponding scaled size of the object underwater.

The primary limitation of this method is the distance that the Sonar returns to the size measuring algorithm. The Sonar must be perpendicular to the object it is trying to measure underwater to achieve the least amount of error in its measurement. Also, as the Pilot is drawing a line of an object to measure, the Pilot must orient the ROV to be perpendicular to measure the full length of the height or width of the object, instead of an auxiliary view of it. This limitation is only solvable through Pilot control. It is up to the Pilot to place the ROV in orientations to best measure underwater objects.

Safety

Safety was a high priority when working on and testing the UNH ROV. Two people worked on the ROV at any given time in order to reduce the risks that a lone worker may face in a potentially dangerous ocean engineering laboratory. Machine shop safety was enforced by a professional machinist. Eye protection, life jackets, and proper lifting techniques were used at all appropriate times. Finally, testing was performed on a component level, a dry level and then ultimately a wet level in order to reduce any potential risk factors.

Challenges

Waterproofing of the electronics tube was a challenge that we overcame through a number of elements of the overall design of the ROV. To start, we used a single electronics tube to minimize locations where water could creep in and destroy our electronics. To seal the end caps on the electronics tube we integrated a double O-ring design on our end caps. On the camera end of the tube we designed an extra wide flange and used a soft neoprene gasket between it and the camera dome to ensure a watertight seal. Before sealing the electronics tube we applied marine grease on the flange and O-ring area for extra security.

Skill Gained/Lesson Learned

Throughout the course of the build the team members left with many positive work experiences. The entire act of building an aquatic vehicle from the chalk board to something that is fully operational on the bottom of the pool consistently pushed each and every one of us to accomplish more than we thought possible. We constantly pushed ourselves beyond our previous best growing as engineers throughout the build. This year our team did not consist of any computer science majors which at first seemed like it was going to hinder the team's success. Due to the lack of computer science majors, members of the controls team had to step up to the plate and fill in where help was needed. This added a lot of learning to the controls team's effort throughout the build and design of the computer system.

On top of persevering through the lack of dedicated coders the team also became well-adjusted to acting as test engineers. With some of the build on the chassis the team would incur little speed bumps along the way in which there would be need for slight redesigns. Luckily this year waterproofing went through completely unhitched so there were zero electrical catastrophes. We developed and operated in a very organized structure as to design, dry test and then finally water test all of our subsystems: chassis, airtight chambers, robotic manipulators and thrusters. With this structure on our side each team member gained a grasp for better and worse ways to go about testing a product throughout its developmental stages

Nearly all of the machining for the UNH Aquacats ROV was accomplished in house in 2 of our machine shop facilities on campus. A few of the students on the chassis team had acquired prior approval through class credits to use the equipment on their own. This included every type of machine today's machinist would use with the exception of a CNC in which the team members would work under supervision of approved personnel for safety purposes. The hours in the shop were valuable experience gained by team members becoming more accustomed to the proper ways to machine parts. This is essential experience as knowing how parts are made are very important when designing them in the field of engineering as it is mandatory that the part can actually be made through various processes.

Future Improvements

UNH ROV made many improvements to this year's design over previous iterations, however there are still some key areas in which the design could be further improved. The first is the inclusion of a virtual reality vision system used for camera control and a more immersive human-robot interaction. This enables the pilot to become even more aware of his or her surroundings in the underwater environment. Second is the cost of some critical components such as the USB extenders and waterproof connectors. The constraints required for the design are far less critical than the specified limits of these two products. By choosing to purchase items that do not have such a large factor of safety, cost can be saved with the addition of just a small amount of risk. Lastly, the team would also like to mentor a SeaPearch group in order to spread values of STEM Education in the greater Durham, area.

Reflection

"Having a chance to apply my skills and knowledge of engineering in an ocean environment challenged both the way I think as well as the way my team thinks. It requires a greater emphasis on the risk analysis of specific parts, cables, and sensors in a not-so-typical environment: in water. Having been on other robotics teams in high school, competing in the MATE ROV Competition allowed me to leverage what I know about leading a robotics team, but also learn something new by designing for a much harsher environment."

-Nathaniel Cordova
Operations Manager, Class of 2015

"This project was a grand ol' time."
-Shaun (the sleeper) Hespelein
Chassis Captain, Class of 2015

"Moving from classroom and primarily theoretical work to application of my gained engineering skills was a great transition from schooling to real-life applications. Being able to see the progression of the design phase through the assembly and testing of the individual components and the ROV as a whole was a broad experience that was worth the time taken. It was an interesting experience taking our knowledge of mechanical engineering and applying it to ocean engineering."
-Jarrett Linowes
Team Captain, Class of 2015

"This project was a fantastic way to test and prove my skills as a flourishing mechanical engineer. There were many tests along the way in which I worked long hours with my teammates in order to adhere to the problems and resolve them with high performance solutions."
-Sayward Allen
Treasurer and Chassis Team member

References

Fox, Robert W., Alan T. McDonald, Philip J. Pritchard, and John C. Leylegian. *Fluid Mechanics*. Hoboken, NJ: John Wiley, 2012. Print.

Ogata, Katsuhiko. *System Dynamics*. Englewood Cliffs, NJ: Prentice-Hall, 1978. Print.

Parker O-ring Handbook. Lexington, KY: Parker Seal Group, O-Ring Division, 2001. Print.

Acknowledgements

Advisors: Professor Thein and Professor Swift

Graduate Advisor: Firat Eren

UNH College of Engineering and Physical Sciences, Dean's Office

Tara Hicks Johnson, SeaPearch

Dr. Martin Renken, Keyport Navsea contact

Scott Campbell

Kevin Maurer, Portsmouth Navy Shipyard, Judge

Team Sponsors