Computer Engineering Program

College of Engineering and Computer Science

Sprint 2021

Underwater Remotely Operated Vehicle

EGCP 471 - 5/27/21

Prepared by: Russell Guillermo, Winston Do, Israel Garcia, and Alvaro McNaughton

Table of Contents

Abstract

The goal for this senior design project was to design a cost effective ROV that can be easily used by an operator with minimal training. The objective is to enter our designed ROV into Marine Advanced Technology Education (MATE) competition, which is pending in the Spring of 2021. The ROV will have to complete a certain task for entry. The tasks are to be able to maneuver under water and grab objects such as trash, coral, or claims , image processing of those objects, and be able to identify the health of a mussel bed or coral reef using image recognition. To complete these tasks our ROV will have a robotic arm and dual cameras. Maneuverability will be proved by four thrusters all controlled through specialized control software that links it with an off the shelf Microsoft XBOX gamepad powered by a Raspberry Pi. The thruster and the robotic arm is mounted on a PVC exoskeleton that serves as the anchor point between the external devices and the internal enclosure which houses the electronics. So far, the internal enclosure proved to be too small to house all the necessary electronics and thus a larger one was procured. Furthermore, An incompatibility between the motor control board and the thrusters caused the team to switch to a new Electronic Controller System for each of the four individual motors. This necessitated a complete rewrite of the thruster control software. All the revisions will be completed in the Winter of 2021 for an underwater test.

Background

A remotely operated underwater vehicle (ROUV or ROV) is a vehicle built for marine purposes that can forgo a human operator within. Many environments, especially underwater ones are hazardous to human operators and having the vehicle be remotely operated protects human life.

Underwater environments present several challenges for humans working. The environment lacks oxygen, light and possesses significant atmospheric pressure, which represents a significant danger to human divers. Divers performing jobs in underwater environments require them to bring their own oxygen tanks, which can occupy a significant amount of mass and volume, limiting a diver's ability to carry equipment or material to the job site. For extremely deep water environments, divers also have to contend with decompression sickness, commonly called the "bends." Ascending too rapidly will create nitrogen bubbles in the blood, which can block off circulation. To prevent decompression sickness, divers have to either acende slowly or be placed in decompression chambers. Both methods require hours to complete. People working in submarines or submersibles do not need to contend with decompression sickness. However, manned undersea craft need to bring their own life support systems such as oxygen and heating. This will limit the payload of a manned underwater vehicle. An ROV does not need to contend with either of these environmental challenges.

The ROV can be stripped of life support and only carry mission critical devices. This can also reduce the weight and size of the ROV compared with a manned vehicle, reducing material and maintenance costs for the user. Reduced mass also means the ROV does not need as powerful propulsion to maneuver underwater and a reduced size profile allows a ROV to enter extremely size constrained areas that a larger craft would be unable to. One of the key distinguishing features of an ROV is the umbilical cable that transfers power and data from a surface ship or station to the ROV itself. Since electromagnetic radiation, such as radio waves, have a poor travel distance within an underwater medium, an umbilical tether is necessary for sending data back and forth between the ROV and the surface. There are two broad categories for the external design of an ROV. These are an enclosed and open design. An enclosed design houses all available components such as propulsion and sensors in an enclosure that reduces hydrodynamic resistance. The open design has an open frame to which all the components are attached to such as the underwater thrusters and robotic arm. The enclosed design typically is built for higher speed applications such as being towed behind a surface ship. Open designs are used for applications that have stationary job sights.

RJE International is a Southern California based company that designs, develops and manufactures underwater equipment for naval, offshore oil refining and marine scientific communities. Part of the specialized equipment that they provide are ROVs. RJE International provided this project with engineering and financial support. Engineering advice ranged from power and data link issues as well as integration and testbench advisory.

Design Specifications

Throughout the design of the remotely operated vehicle (ROV), the MATE ROV Competition hand book was continuously consulted in order to meet requirements/specifications for our device. Therefore, such constraints were classified under the sections of Physical/Mechanical, electrical, communications, control, and MATE on site operations, and based on these constraints, certain parts were integrated to our ROV. Therefore, a close tab was kept on each component's weight which contributed to the design changes made throughout the development of the ROV as knowledge and guidance from RJE International was inputted.

The ROV is intended to be neutrally buoyant. This will ensure that the ROVs depth will be entirely controlled by the thrusters. Under the physical/mechanical requirements, our main focus was to achieve an electronic housing that is capable of operating to the depth of 7 meters, and must not exceed a diameter of 92 cm along with a maximum weight of 35kg in air for the overall ROV, including the tether. Therefore, a close record of overall weight was maintained. The software control components of the ROV are both powered by a Raspberry PI 4 Model B single board computer and can be broken into two major modules: the control software and the camera control software. The external structure of the ROV can be broken down into three modules, the exoskeleton framework, the internal enclosure and the umbilical tether.

The exoskeleton framework encloses the internal enclosure. It gives the ROV an attachment point for the thruster and robotic arm, as well as providing the enclosure with protection against impact. The use of PVC pipes was selected for the use in the exoskeleton. PVC pipes have the advantage of being both strong and cheap. Additionally, using PVC pipes for the exoskeleton allows us to change the dimensions of the frame when necessary as the pipes can be cut down with off the shelf tools. Currently the PVC exoskeleton is 26.67 cm wide, 58.42 cm long and 27.31 cm tall. The weight of the PVC framework is 2.582 kg. The four thrusters were oriented on the PVC framework, with two located on the ROV's dorsal spine and two on the right and left wings of the exoskeleton. The dorsal thrusters would control ascent and descent maneuvers while the side thrusters would control forward, backward and turning maneuvers.

The internal enclosure is the core of the ROV. It houses all the necessary control electronics and protects them from the water. As this component is critical to a functioning ROV and its failure can lead to a complete destruction of the ROV, the decision was made to incorporate a watertight enclosure from a professional marine company, BlueRobotics. This was an expensive yet proven method of keeping our electronics dry. The enclosure is a cylindrical tube made of acrylic with a 6cm diameter and a height of 11.75cm which has an overall weight of 2.582kg.

For all communications and power needs to the ROV, a tether with a length of 18 feet will be used. To issue power, a 14 gauge will be used to deliver a nominal voltage of 48 VDC with an in-line fuse of 30A with all power conversion being performed on board the ROV. Additionally, for the live feed and still images, an internet cable will be used to provide this form of communications which will cut down on the weight that was seen by the HDMI cable. Furthermore, the ethernet cable will be connected to a laptop which will be connected to the

Raspberry PI 4 via VNC, virtual network connection, and SSH, secure shell. This will also allow for GPIO signals to be sent from the host computer to the raspberry pi which will reduce the number of penetrators needed. Therefore, for our ROV controller, the ethernet cable will be used to control the ROV which will result in a total of 2 cables from our 'base' to the ROV.

The control software is the main method for a user to maneuver the ROV. The objective was to design a control system that allowed an operator with no knowledge of thruster orientation to use and be able to control the ROV and subsequently the Raspberry Pi (RPi) through an umbilical tether over an ethernet connection. The controller gamepad is an off the shelf Xbox 360 controller Microsoft manufactures for their video game console. It was chosen due to its familiarity and analog input. Initially, the software utilized a motor controller HAT expansion board and subsequent API. The Xbox controller needed to be directly connected to the RPi over USB for the RPi to work. However, this presented several problems. First, it bloated the tether as another connection to the surface was required. Secondly, USB connections degrade significantly over a longer tether length. Finally, it required a complicated startup procedure. The semester goal was to completely refactor the control software to utilize the new electronic speed controllers (ESCs) instead of the motor controller HAT expansion board and the Raspberry PI's native GPIO.

The current iteration of the control software consists of two modules, Pygames and GPIOZero. One of the core differences from the previous iteration was to interpret the Xbox controller on the host computer and then forward that data to the RPi through the ethernet connection and control the GPIO. The GPIO would then drive the appropriate ESC and thrusters to control the craft. Pygames is a Python package made to produce video games easily through the Python Language. It was useful as it allowed a Windows 10 machine, which was used as the host computer, to interpret signals from a video game controller. It also provided a nice diagnostic UI as well as other features such as controlling the update speed of the control loop. GPIOZero was used to connect to the RPi over a networked connection, allowing a remote computer to control the RPi's GPIO over the network.

At its core, the software consists of various helper functions, an arming function and a main control loop. The arming function would calibrate the ESCs as per the manufacturer's specification. The main control loop would get the state of the controller, clean up the data for the RPi to interpret, update the diagnostic UI and forward that over the network. This control loop would update over 60 times a second. The arming function was also improved from the last iteration. Previously, it would arm each individual ESC serially, a process which took a decent amount of time. The new arming function takes advantage of concurrency through multithreading to improve the startup speed by a factor of four. Now all four ESCs are calibrated simultaneously. Additionally, other control modules were added such as one that interprets a setting configuration INI file. This allows settings such as thruster GPIO pins and network configs to be manipulated through an external text file without the user having to go through the Python code directly.

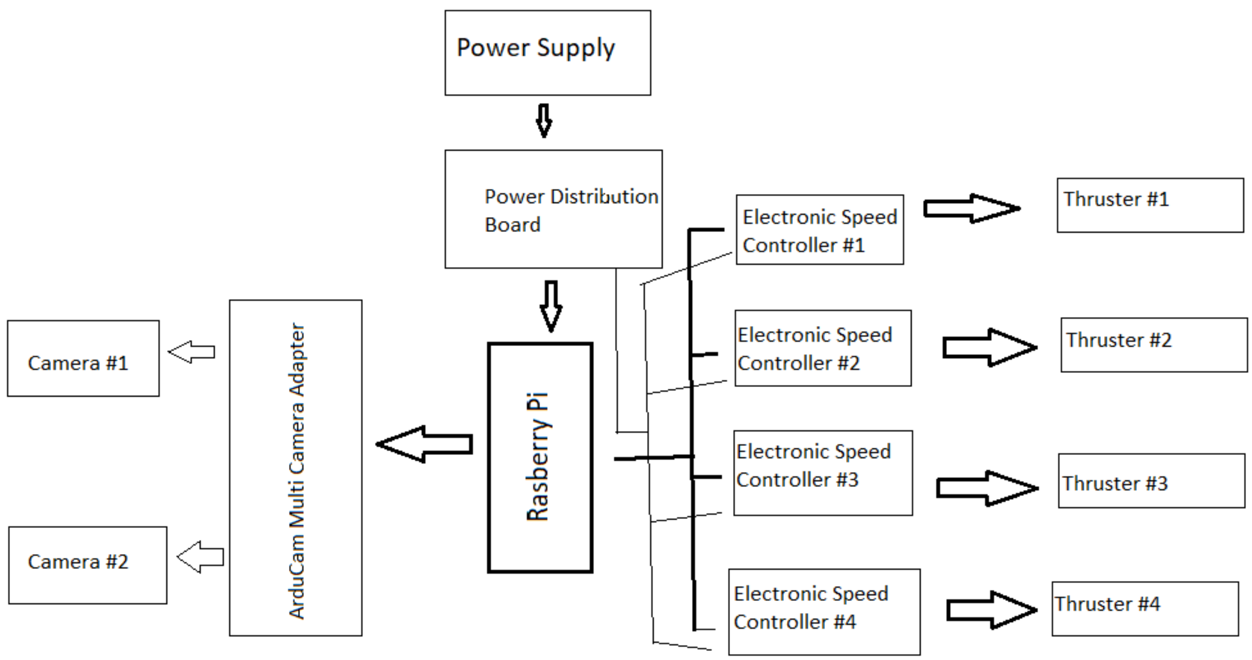
Additionally, the camera switching capability will be handled by a secure shell command from the host computer to the raspberry pi. Furthermore, in order to maintain a waterfree solution, we enclosed two camera modules inside the main container connected to the multi-camera adapter to the raspberry pi (Arducam Multi Camera Adapter). These camera modules consist of Sony's IMX219 8-megapixel sensor which will allow for the use of high-definition live feed and still photography. Additionally, our camera orientation will enable us to have a front facing view coupled with a bottom view of the ROV environment. However, due to software incompatibility, our dual camera setup had to be temporarily unable which resulted in only one camera being used in the device.

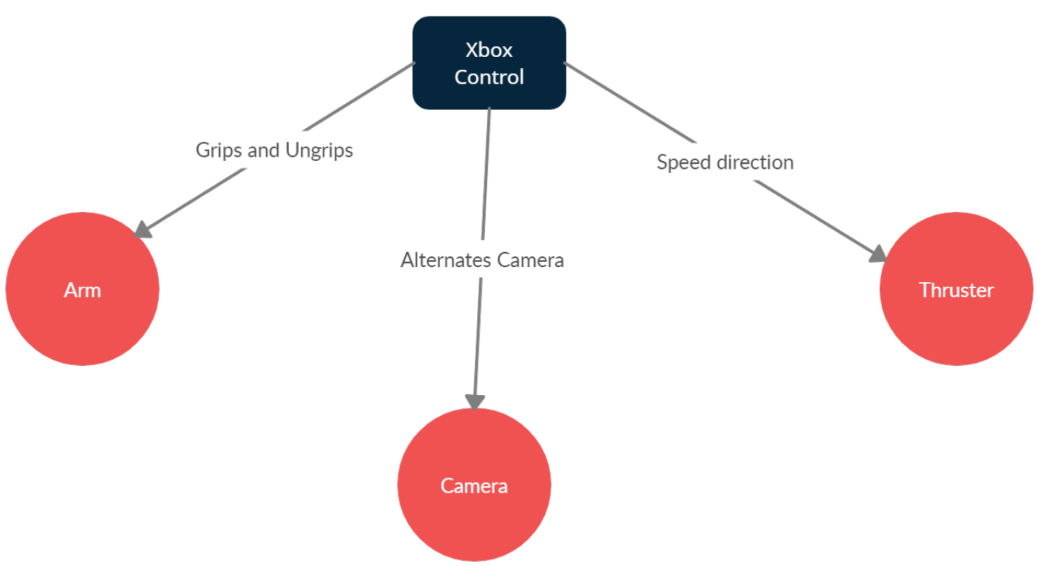
CpE SO(2) – The ROV requires electrical, mechanical and computer engineering abilities to design a solution to meet public health and safety for the environment. As the underwater remotely operated vehicle is needed to explore the oceans geography and takes on tasks to help preserve the sea life. With its easy maneuverability and robotic arm it can reach and go places underwater that might be harmful for humans. With this technology the ROV can help keep track of the health of the ocean, seabed, and coral reefs.

CpE SO(5) – The underwater remotely operated vehicle is a project that takes more than one person to do. With the work of our team we were able to divide the task up and work on each task individually. To meet the overall objective of getting the ROV assembled and put underwater we all needed to collaborate on what we have worked on and put together the whole ROV. When one member of the team ran into a problem we collectively thought of a solution so we could reach the end goal.

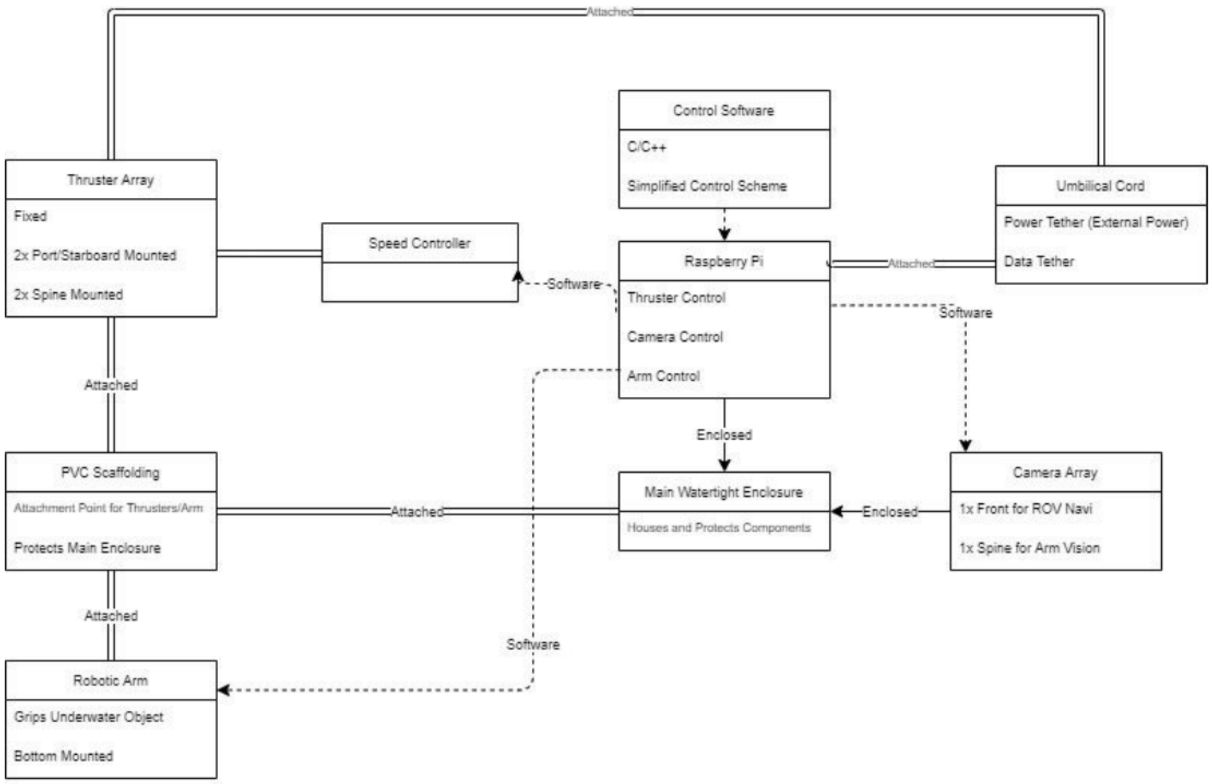
CpE SO(7) – Since this is the first time our group has made an ROV project it required lots of background research on other types of ROVs. We gathered every bit of information needed to make a working ROV and applied what we gathered to the ROV. This project also required lots of trial and error and we learned a lot from what worked best and what didn't learn from our mistakes. There were times when we were stuck on a problem but we used our resources such as the RJE international engineers and talked with a few colleagues to gather more information on how to solve that problem.

Block Diagram and Functionality

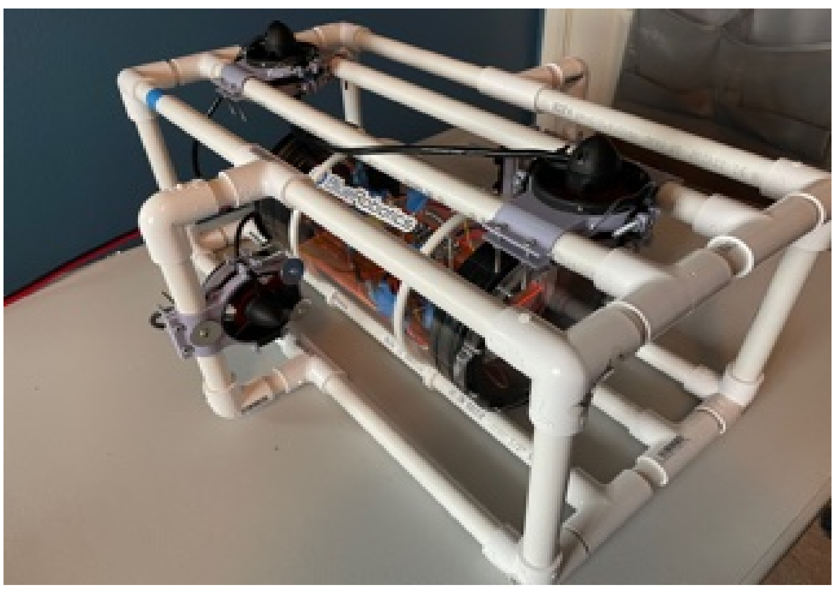




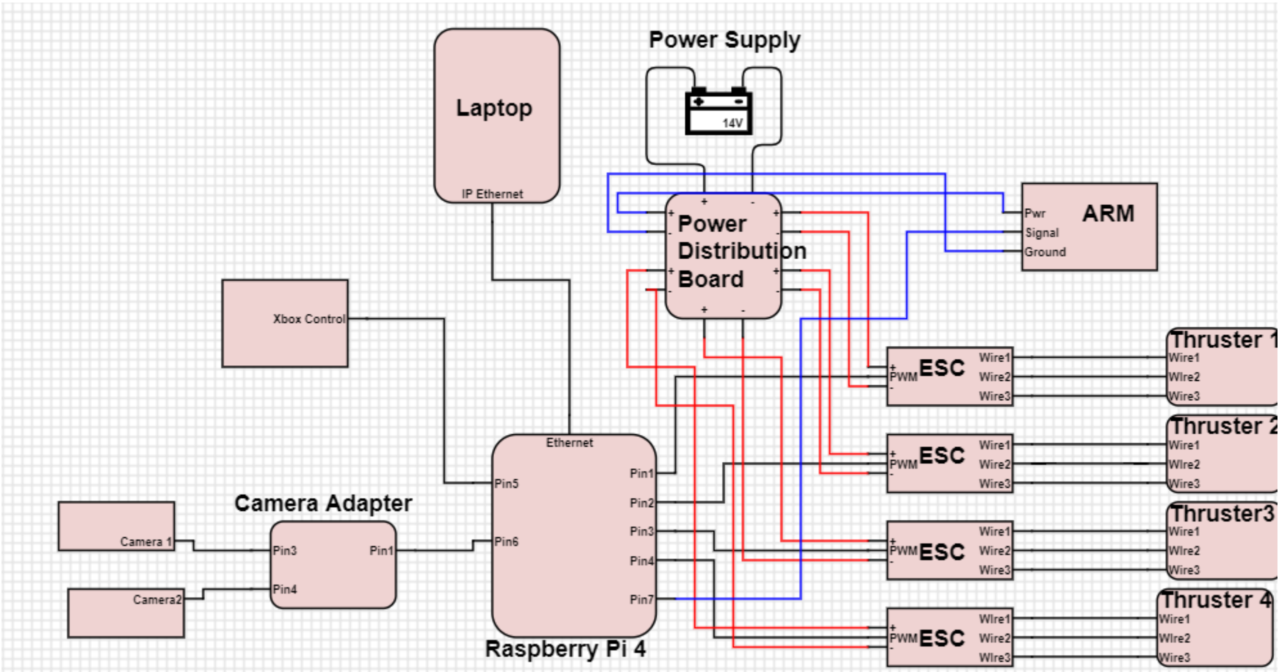
Flowchart



Attached
Control Software
C/C++
Simplified Control Scheme
Thruster Array
Fixed
2x Port/Starboard Mounted
2x Spine Mounted
Speed Controller
Software
Raspberry Pi
Thruster Control
Camera Control
Arm Control
Attached
Umbilical Cord
Power Tether (External Power)
Data Tether
Software
Attached
Enclosed
PVC Scaffolding
Attachment Point for Thrusters/Arm
Protects Main Enclosure
Attached
Main Watertight Enclosure
Houses and Protects Components
Enclosed
Camera Array
1x Front for ROV Navi
1x Spine for Arm Vision
Attached
Software
Robotic Arm
Grips Underwater Object
Bottom Mounted



Schematic Diagram



In the schematic the main component is the Raspberry Pi. Through the raspberry Pi we can control the robotic arm, camera, and the four thrusters through an Xbox controller. The Xbox controller will be programmed to change the speed and direction of the thrusters just by using the two joysticks. Pressing the trigger button on the controller will allow full throttle to the thrusters in any direction depending on the user's input of the joystick. The Xbox controller will also control the camera to use with one of the buttons. One button on the controller will be programmed to flip through each camera angle. Lastly the Xbox controller will control the robotic arm. It will be programmed to use a button on the Xbox controller to close the grippers and to press again to release the gripper. As of right now we do not have the actual arm, so we do not know specifically how the schematic will work. A power distribution board is also needed to distribute the power between the electronic speed controller and the robotic arm. The ESC will have a signal wire connecting to the raspberry Pi to control the PWM and duty cycle to the thrusters. For the camera feedback we are using an IP ethernet cable which will transfer data from the raspberry pi cameras to a laptop. This will allow us to get live camera feedback when our ROV is under water. Lastly our power supply will be connected to an AC outlet which will then be converted to DC power to deliver power to our ROV. The power supply will have a fuse box attached to it regulating the current going in at 40 amps. The power supply will then be connected to our power distribution board to distribute the power needed for each component.

Parts List

Part	#	Price	Shipping	Taxes	Total
Tethers (Polyethylene Rope)	1	15.95	0.00	1.28	17.23
Cat 6 Ethernet 25'	1	24.99	2.17	2.00	29.16
Thruster	4	45.99	0.00	14.72	60.71
Motor	1	32.99	0.00	2.64	35.63
PVC Pipes	1	15.98	0.00	1.28	17.26
Watertight Enclosure	1	144.00	12.92	11.52	168.44
Power converter	1	160.00	0.00	14.4	174.4
Video game controller	1	14.99	0.00	1.20	16.19
Raspberry Pi Camera Mod v2	2	19.89	0.00	1.59	43.00
Raspberry Pi 4 Model B - 4 GB	1	55.00	0.00	4.40	59.40
Usb ext/16.4t	1	18.99	3.99	1.52	24.50
LED light	1	4.46	0.00	0.36	4.82
Glue/Sealant	1	6.99	0.00	0.56	7.55
Power connector	1	40.00	0.00	3.20	43.20
Polymorph	1	19.99	0.00	1.60	21.59
Silicone grease	1	20,85	0.00	1.67	22.52
14 awg wire	1	42.61	0.00	3.41	46.02
Fuses	1	12.99	0.00	1.04	14.03
Fuse box	1	23.99	0.00	1.92	25.91
Arducam multi camera adapter	1	49.99	0.00	4.00	53.99
hdmi to usb	1	14.99	0.00	1.20	16.19
15 ft hdmi to hdmi	1	15.99	0.00	1.28	17.27
camera case	2	5.9	0.00	1.12	7.02
Micro-hdmi to Hdmi adapter	1	6.95	0.00	0.56	7.51
Zero2Go omini - Multi-channel	1	19.95	11.43	1.84	33.22
2x20 socket riser header for raspb	1	1.95	0.00	0.156	2.106
ESC shark series	4	20.99	0.00	1.94	91.72
Inflatable pool	1	59.99	0.00	4.80	64.79

Budget

470 1st Semester			471 2nd Semester	
Total Budget	$1,965.54		Previous Budget	$676.45
Winston Do	$550.40		Winston Do	$66.79
Russell Guillermo	$308.00		Russell Guillermo	$253.53
Israel Gracia Figueroa	$248.76		Israel Gracia Figueroa	$63.64
Alvaro McNaughton	$181.93		Alvaro McNaughton	$50.00
Total Bought	$1,289.09		Total Bought	$433.96
Remaining	$676.45		Remaining	$242.48

Performance Analysis

Every group member completed the task they were assigned. This resulted in having all four thrusters work with the motor controller board, live camera feedback through the ethernet, thruster control software with gamepad integration to control all the components of the ROV and a built frame with 3D prints to hold our main enclosure and thrusters. Each component was tested and verified separately.

Utilization of a dual camera set up on the Raspberry Pi was not implemented due to software incompatibility. As a result, a single camera configuration was used with its orientation being front facing. However, with careful analysis, the issue was pinpointed to be from the use of the motor controllering software along with the camera software. Specifically, from the use of wiringpi (camera end) and pigpio (motor control end) as both relate to how the raspberry pi GPIO are used.

As the new thruster control system no longer utilizes the planned motor controller board, the software control system for the thrusters needs to be adjusted to use the Raspberry Pi's (RPi) native GPIO and ESC. The new iteration of the control software of controlling the RPi through a network connection. It also has new features that make it easier for an operator to control the ROV and configure it such as a diagnostic UI and an external INI file. The software is also less performance demanding and has significant start up speeds. These were achieved by improving the code as well as moving the processing to a more powerful machine instead of the RPi itself.

The 3D prints were also tested and fit perfectly onto the frame and were able to hold the thrusters and enclosure in place. The 3D printed connector also proved to be adjustable which allows the thrusters to be adjusted on the exoskeleton if necessary. After analyzing the amount of components we need to fit inside the enclosure we decided to get a bigger enclosure to house all the electronic components. This will require replacing one of the 3D printed parts that holds the internal enclosure to the exoskeleton frame.

Results

An exoskeleton framework was created with PVC pipes. This was integrated with an internal enclosure that houses all the main components such as the four thruster electronic electronic speed control (ESC) and Raspberry Pi. The Raspberry Pi is the main device that will control the entire ROV and process visual data from the cameras. The thrusters that help move the ROV in different directions, ESC which are needed for the thrusters to work with the raspberry pi, camera that will be the eyes of the ROV to see where its going underwater and to be able to detect objects when needed, and the enclosure which will waterproof all the previous components mentioned before. To secure those items, we created 3D printed pieces that will fit and hold them perfectly without interfering with their purpose underwater. We tested the main enclosure underwater before we put any electronic components inside for any water leaks. During the water test the ROV kept flipping over and tipping because of the placement of the main enclosure. After fixing the issue the ROV is able to maneuver perfectly fine in the water. The ROV does have some problems diving under the water as the ROV as a whole is too buoyant and floats back up. The ROV camera does work with the software but at times was only able to run one camera due to overheating the pi.

Problems Encountered

The problems we have encountered throughout making the ROV are static discharge for the camera , thrusters not receiving any power or current, too many wires coming from the ROV through the tether, and the enclosure being small and having a slight leakage. We fixed these solutions throughout the semester. The first problem we encountered was having the camera break because of static discharge, this would be a problem because when our ROV is underwater the camera can be discharged randomly and causing us to lose vision underwater. This was fixed by creating a chassis for our camera which held the camera in place and protected it from any possible discharge. Additionally, camera software incompatibility was encountered when assembling all components of the ROV which resulted in the raspberry pi to reboot when launching the camera. However, due to the issue being found in late stages of the project, a solution was to have the ROV use a single camera setup.

The core problems with the control software was that the main module that interprets Xbox controller games, would output junk values occasionally. This resulted in undefined behavior when the ROV initially starts up but would be resolved once any input was pressed. The right and left triggers were initially mapped to the dorsal thrusters which would descend and ascend the craft respectively. However, the Pygames module did not interpret the right trigger properly, resulting in only the thrusters running at half power despite the right trigger being depressed fully. It is unknown what caused this issue and a workaround was devised by simply mapping a binary button (right bumper) to have the dorsal thrusters descend the craft. During water testing, it was found that the ROV could pitch uncontrollably. To solve this issue pitch controls would have to be implemented using the dorsal thrusters. This implementation could be handled through software.

Another problem we encountered during the water testing the ROV was too buoyant and at times would tip over. This was due to the placement of the main enclosure, which is the main source of buoyancy, at the bottom of the ROV frame and it tipped the ROV over most of the time. We fixed this issue by changing the placement of the main enclosure at the top of the frame and drilled holes into the PVC pipes to allow water density into the frame. This somewhat fixed our issue as the ROV was equally balanced. After further testing the ROV was still too buoyant so we added 5 pounds to the bottom of the ROV and it was able to move a little better.

After a couple of water testing the ethernet cable on the ROV did not work anymore, this might have happened because of faulty wires. So we fixed this issue by replacing the ethernet cable and making a better connection to the pi so the ethernet cable isn't moving around as much while the ROV is moving. When tested underwater, the side thrusters which were designed to move the ROV right or left were spinning on the pvc pipe which made us drill a screw in the middle of the 3D mount to stop movement and help maneuver the ROV .

Individual Task

Member	Task
- Russel Guillermo	Exoskeleton - Waterproofing - Lightning - Main electronic enclosure mounted - Buoyancy - Testing Electricity and wiring - Wire placement - ESC and thruster connection General Research - Power divergence from the tether

- Winston Do	Thruster Control Software
	- Gamepad and Thruster Control Software
	- Motor Control Integration
	- Diagnostic UI
	- Remote GPIO over ethernet from Raspberry Pi to host computer
	- Software module integration
	    - INI configuration
	    - Camera integration
	    - Wrapper module to integrate control and camera modules
	- General documentation

- Israel Garcia Figueroa	Camera - Parallel processing for the camera which was updated to single processing due to raspberry pi hardware limitations - Camera software was changed to use socket communication for live feedback - Secure shell command to initiate socket communication Power - Researched and implemented a method to power on raspberry Pi through GPIO pin - Buck and step up convertor
- Alvaro McNaughton	Frame - Enclosure mounting - 3D Print - STL design - Water tested 3D parts - Thrusters mounting

Conclusion

Overall, great progress has been made to the ROV despite Covid-19 restrictions which include both hardware and software. For the hardware, there were some fixes towards the design of the ROV and the placement of the main enclosure. The ROV frame needed holes to be drilled into to cancel out some buoyancy. The main enclosure which originally was at the bottom of the frame has been moved to the top of the frame to have most of the buoyancy at top. There were minor fixes towards the 3D prints as we needed to resize it for the main enclosure and also drilled a screw straight through the thruster mount to keep the thrusters from turning when firing.

The software was completely refactored from the previous semester. Now it performs well and controls the ROV over an ethernet cable through a host computer. The RPi no longer interprets controller data and much of that complexity and calculations are now handled by the host computer, which is significantly more powerful. This also allows the RPi to handle other calculations such as the camera.

After changes towards the hardware aspect and fixes in the software the ROV was able to maneuver in the water controlled by the xbox controller. The ROV was also able to deliver live camera feedback at the same time.

Future Work

In the future we plan on adding a robotic arm to the ROV which is required to enter competition and for practical use of the RoV to grab and move items in the water.

References

Pulse Width Modulation (PWM) on the Raspberry Pi with Python programming. (2016, July 5). MBTechWorks. https://www.mbtechworks.com/projects/raspberry-pi-pwm.html

2021 Explorer Class. (2020, September 14). MATE ROV Competition. http://files.materovcompetition.org/2021/2021_EXPLORER_Manual_14Sept2020.pdf

"'Remotely Operated Vehicle Committee of the Marine Technology Society,'" *rov.org*, 10-Oct-2017. [Online]. Available: http://rov.org/rov_categories.cfm. [Accessed: Sep-2020].

Source Code

RPi Software
https://github.com/winstondo/RoV_ThrusterControlSystem

Acknowledgements

1. RJE International
2. AUV Senior Design Team
3. LifeSaver Board Senior Design Team