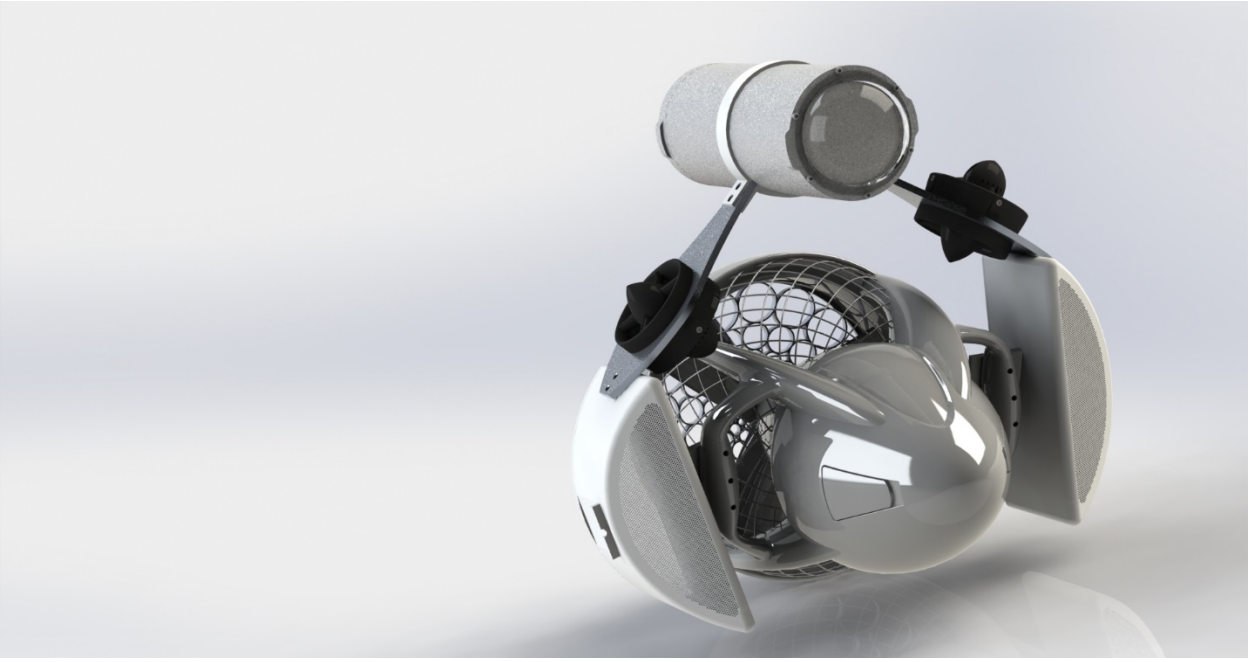
Free Willy

A pollution-collecting platform



April 29, 2020
Interdepartmental Senior Design Project 2019-2020
Fall Final Report for Senior Design and M&T Integration Lab

I. Cover Page

Page Count: 12+2 integration lab additional pages = 14

Team:
Roshan Benefo, Mechanical Engineering
George Pandya, Mechanical Engineering
Wesley Sheker, Electrical Engineering

Advisors:
Dr. Ani Hsieh (Primary advisor)
Dr. Chinedum Osuji (Co-Advisor)
Dr. Douglas Jerolmack (Particle detection advice)

II. Executive Summary [Outcome 3.2]

Most reservoirs use large fixed purifying equipment to filter out common pollutants from the water. However, reservoirs are susceptible to contamination by pollutants that cannot be screened out by the fixed filtration equipment. These include pollutants from chemical plants, biomatter, and microplastic pollution. In cases like these, municipalities, such as Cornwall, England in 1989, which suffered the worst mass poisoning event in British history when a chemical plant poisoned the reservoir, have no way to clean the reservoir because there does not exist a rapidly deployable method of cleaning specific types of acute pollution.

To resolve this, the team has designed a mobile pollution collection platform capable of integrating with custom filtration canisters to remove a variety of potential pollutants. Our platform allows for inexpensive vehicle realizations, dynamic mission deployment, and high-quality mission data analysis. The separate, attachable filtration canister system allows for replacing filters to match the acute pollution observed in a given freshwater reservoir. Over the course of the year, the team has successfully designed and verified a vehicle based on a seascooter system with a custom electrical floorplan, a communications system design involving a buoy and an Amazon Web Services custom software stack for mission data and execution control, a sensor fusion algorithm and PID controller for path execution, and an implementation of a trajectory planning and data analysis dashboard. The team has also fabricated a 2-stage microplastic filtration canister realization of our pollution control platform, including the sensor suite used to control and diagnose failures within the canister. While the team was unable to manufacture our final vehicle realization because of lab access constraints, the team was able to verify functionality of the vehicle based on a CAD simulation. The team was also able to successfully demonstrate control of our vehicle's main propellor by retrofitting our vehicle's seascooter body prior to the lab access restrictions.

Ultimately, our design could be deployed to clean-up acute contamination of freshwater reservoirs. By deploying a small fleet of Plastic Free Willy vehicles specially equipped with a filter designed for the given pollutant, dangerous acute contamination of microplastics, mercury, lead, oil, or other pollutants can be controlled. Furthermore, our design is more cost-efficient than installing permanent fixed filtration systems capable of dealing with any pollutant, including rare contaminants.

III. Overview of Project [Outcome 4.2]

Small-scale distributed pollutants are endemic to the world's water supplies. History is littered with times such pollutants have caused significant human or environmental damage: more recent examples include the recent public health crisis in Flint, Michigan, when lead from aging pipe infrastructure contaminated the city's water supply. Microplastics are another well-known example of such pollutants; currently, these small particles are found in every major body of water on the planet, threatening the marine ecosystem, and posing serious health risks for a large amount of the food chain, including humans.

To tackle these pollutants, a number of changes are needed. First, regulation needs to be passed to ban their production: for example, governments need to pass laws preventing companies from producing goods that use microplastics, and regulations that limit how companies can dispose of materials that may degrade into microplastics.

However, this is not sufficient to clean up the pollutants that are *already* in the world's water supplies. Current solutions to clean up existing pollutants involve the use of fixed filtration stations at effluent sites and areas where water is taken to be made into drinking water. While these solutions are helpful, they are unable to make much of a dent on the pollutants existing in water *not* near critical inflow or outflow infrastructure. These pollutants still impact the environment in damaging ways, killing wildlife and poisoning plants and humans. Rather, what's also needed is an easily deployable and scalable mobile system to help collect pollutants.

Our Free Willy platform seeks to serve this niche. The Free Willy system is a system of autonomous vehicles designed to collect pollutants in reservoirs. Due to the diverse nature of pollution (there are many kinds of pollutants in the world's water supplies), the team designed Free Willy to include filtration canisters that carry filters that can be swapped out to filter many different kinds of small-scale pollutants. As a proof-of-concept, the team elected to demonstrate Free Willy's ability to collect microplastics[1].

Our Free Willy platform consists of three main components: the vehicle, the AWS pipeline designed to be used in conjunction with a communication buoy, and the user interface and mission control system. A diagram of the system is included in Appendix Figure 3.1. The vehicle consists of a primary propeller retrofitted from a seascooter, three control propellers, two pollution collection canisters, and the main electrical floorplan consisting of a Raspberry Pi capable of both real-time data collection and vehicle control (including support for autonomous control). A block diagram of our system is in Appendix Figure 3.2.

IV. Technical Description

This year, the team completed a CAD design of the entire vehicle; implementation of the retrofitting of the main propeller; computer design of the electrical floorplan; realization of the user interface, AWS stack, and mission control software; and a microplastic realization of the filtration canister complete with particle detection system.

Specs, Requirements, Economic, and Social Considerations [Outcomes 2.1, 2.3]

The overarching requirements were to build a cheap, rapidly deployable, and modular vehicle that can be quickly outfitted to collect any type of pollution. As such, manufacturing techniques and material costs became quite important. Additionally, the vehicle needed to be able to house a sensor suite that can provide inertial navigation, control of all major systems, and communication.

The main economic concern driving the design is the fact that municipalities need to deploy these systems en masse, often on small budgets (generally, municipalities do not have much funding for reservoir cleanup). Thus, affordability is key--the team targeted a per-vehicle cost of less than $500. One of the ways to reduce the cost of the system is to make it user friendly, allowing the system operator to have

[1] The team was ultimately unable to perform this testing, due to the pandemic that swept the world in the spring semester.

minimal training and experience. As such, the team focused heavily on ensuring that the system is as easy to use as possible.

Iterations/Alternative Designs [Outcome 2.2]

The team initially started with a seaglider design for the vehicle. Seagliders are variable buoyancy vehicles; they control their velocity and depth with a variable buoyancy engine, allowing them to operate for months at a time, traveling thousands of kilometers under a single charge. The team was initially attracted to this design because its initial focus was on microplastic collection in an ocean– designs like this would allow fleets of vehicles to effectively cover large swaths of ocean. In the fall term, the team had performed extensive thermodynamic and fluid analysis on such a vehicle and had sized and designed a seaglider. In the spring term, however, the team pivoted to cleaning reservoirs because it was determined that reservoir cleanup would be far more tractable (and testable) for small-scale cleanup vehicle fleets like the ones the team was designing; this shift negated the need for a long-range vehicle like a seaglider. The current design of Free Willy was chosen over the seaglider to minimize the cost and complexity of the vehicle.

Technical Description of Vehicle Design [Outcomes 1.1, 6.1 (for software)]

To fulfill the team's goal of building a system that is cheap, rapidly deployable, and able to be outfitted to collect any type of pollution, the vehicle was built onto an existing sea scooter. Though this meant that the team couldn't make major modifications to the structure of the sea scooter for fear of hindering its functions, it did afford a very cost effective and easily sourceable base on which to mount the team's custom-designed parts onto. These custom designed parts include the airbox, control motors, and filtration system.

There are several advantages to using this design. Using a cheap, readily available sea scooter enables rapid manufacturing and deployment in even the poorest regions. Additionally, all of the pollution control components are externally mounted, which allows for extreme modularity when it comes to fitting these vehicles for specific pollution cleanups.

The airbox is centrally mounted, providing ballasting and roll stability for optimal hydrodynamics. This airbox contains all of the control systems, along with the sensor and communications suite that enables the user data tracking and mission planning dashboard. Two waterproof brushless motors mounted anhedrally provide 6lbs of thrust and are capable of yaw control, while minimizing drag in the primary axis of motion; one waterproof brushless motor mounted on the bottom of the craft helps it control its depth. The motors are mounted onto easily machined aluminum that provide structural support.

The design of the filter is the result of several iterations of prototyping and testing, both through physical experimentation with pressures, flow rates, and particle contaminations, as well as with extensive computational fluid dynamics simulations. These iterations allowed the team to maximize the volume of small contaminants that can be filtered, while ensuring that the device isn't susceptible to fouling by large objects.

In the end, the team designed a two stage filter with the angled intake that deflects away large objects. A key design feature is that these two filters can be swapped for the right filter for any cleanup. Finally, the filter was designed to be able to be mounted onto a sea scooter using only six screws, making for extreme modularity.

Seascooter Retrofitting

The seascooter had a built-in safety system for control of the propeller where the rider had to hold down two buttons at all times to maintain the propeller. This was accomplished with a simple lever connected to a magnet, such that when the buttons were pressed the magnet was moved into position outside of what appeared to be an inductor box on the main circuit. In order to preserve the watertight seal of the main propeller circuitry, the team instead designed a simple system using a servo motor to control the movement of the lever. As a result, the propeller could be turned on remotely and the team were able to gain full control over the seascooter without compromising the waterproof sealing.

Design of Electrical Floorplan

To minimize area overhead given the limited airbox space, the entire electrical floorplan was implemented such that it could be controlled by a single Raspberry Pi. The floorplan consists of pins for 2, ESC brushless motors to control the control surface propellers, pins for control over the servo used to control the main propeller system, pins for a 9DOF IMU and GPS system, and pins to control the particle detection system for the filtration canister.

Since the Raspberry Pi possesses significant data processing capabilities but only limited GPIO pin access, the team needed to hijack the full capabilities of the Raspberry Pi. To accomplish this, the team utilized the PIGPIO library.

The PIGPIO library works by running a daemon in the background while providing access to a Python library to control the GPIO pins. It is capable of communicating via the standard I2C medium speed link protocol at 100kbps or an SPI interface at speeds of between 32kbps to 8Mbps. Either protocol is sufficiently low latency to control our relatively low-speed vehicle.

However, the 9DOF IMU system (Adafruit 9-DOF absolute orientation IMU BNO055) I2C protocol faces a clock stretching bug when interfacing with the Raspberry Pi (the bug causes the Pi to ignore slave clock stretching and sample the data line at the un-stretched positive clock edge). As a result, the team needed to make another modification to use the UART mode of the BNO055 for IMU communication.

After testing the ESC control system, the servo system, and the filtration canister control separately, the electrical floorplan was designed as a PCB in Altium; however, the team were unable to manufacture the PCB because the team were unable do a final test of the IMU system and the final motors because of the coronavirus pandemic. The final PCB CAD is shown in Appendix figures 4.1-4.2.

AWS Stack, User Interface, and Mission Planning Software Effort

To realize the Free Willy system, the team implemented a user interface to allow for in-mission tracking of Free Willy Vehicles. To provide the user interface with realtime data, the team built an AWS pipeline to ingest data from Free Willy, store it, and display it on an attractive dashboard. In this the team had a number of goals:

1. Acquire data from Free Willy every 10 minutes. Since Free Willy is a submersible that dives to variable depths, when Free Willy is submerged, it cannot wirelessly transmit data. To get real time data, the team could have designed some sort of follower boat that would connect via wire to Free Willy, and wirelessly transmit Free Willy sensor data– however, the team decided that real-time data would not add much value to our end users, and necessitate an unnecessary increase in the system price. As such, the team elected to have the vehicle surface every 10 minutes, and transmit.
2. Display data in a visually attractive, easily readable format.
3. Store data for further processing, post-mission.

The AWS pipeline takes in data from the raspberry pi via AWS IOT Core. The Pi acts as an MQTT broker; in that, it publishes data to a topic that IOT Core is subscribed to. Because most of this effort was done after the team lost access to the hardware, due to the closures caused by the coronavirus pandemic, the team simulated a raspberry pi MQTT broker via Node Red. The full AWS pipeline is described in the appendix Figures 4.3. The hypothetical buoy design to transmit data from the Free Willy to the AWS stack is described in Appendix Figure 4.4.
The team also built a robust mission planning software package for our system. The software is designed to be easily usable: all the user needs to do is input a PNG and PGw (easily accessible via OpenStreetMaps) of the reservoir they would like Free Willy to clean. OpenStreetMaps images prove particularly useful for our case; they are generally quite accurate (geographic features are typically accurate to 1.57m)[2], and come in clean colors– water features have distinctly different colors from land features, and are consistently colored (all water in OpenStreetMaps looks the same). This allows us to, via Python's CV2 computer vision library, easily create an occupancy grid of the reservoir that the team can plan on.

Since the team knew the color OpenStreetMaps uses to display water, the team first created a mask of that color to remove all other elements in the image other than the reservoir. This removes most of the other elements in the image; however, sometimes, the team found that other small bodies still remain outside of the reservoir. To remove them, the team executed an erosion and dilation on the image, flattened the image (transforming it into an NxM 2D array), and drew contours around each of the remaining elements. Then, the team found the largest contour and removed all of the image that is not within the largest contour. The team tested this method over many images, and believes it is robust– by doing the first erosion and dilation step, the team is able to remove small artifacts remaining in the image, and by removing everything but that which is in the largest contour, the team can remove any large artifacts (the team assumes that the target reservoir is the largest body of water in the image).

[2] https://www.tandfonline.com/doi/abs/10.3846/20296991.2016.1160493

The team then was left with an occupancy grid that the team can plan on. To develop a path plan on this grid, the team leveraged Atsuki Sakai's grid based sweep coverage path planner algorithm, modifying it to import custom occupancy grids, and adding a depth component. The algorithm takes an arbitrary occupancy grid and creates a "zig-zag" path on it, with predefined path radii (the distance between two "zig-zags"). The team outputted a matrix of grid positions and depths from the algorithm (the robot is designed to go back and forth in the zig-zag at varying depths until it reaches a depth either of 10 meters, or 2 meters above the minimum recorded depth of the reservoir).

This zig-zag is not localized, however-- the coordinates are in grid space. To translate the grid space coordinates to world coordinates, the team leveraged the PGw file. PGw files contain data stored in the Esri file format– Esri data contains information to populate a 6 parameter affine transformation matrix that can be used to translate grid data into world data.

The team then was left with two latitude and longitude vectors, and a depth vector. The team interpolated on these vectors to form a fully realized path, and, by adding a time vector, the team was able to generate a trajectory for the robot to follow. To generate the time vector, the team estimated the turning radius of each part of the desired path: the tighter the turn, the more time the team gave the robot to complete the turn. A sample output of the trajectory planner is included in Appendix figure 4.5.

Further work on this trajectory planner could seek to develop a number of capabilities of our robot. Firstly, the team could integrate Open Street Maps directly into the application, allowing the user to simply enter the name of a reservoir, and for our software to automatically grab the PNG and PGw file of the reservoir. Secondly, the software could allow the user, post-generation, to adjust the desired trajectory of the robot via a drag and drop method, where the user could drag nodes in the generated trajectory to adjust it. Third, the software could, knowing the dynamic limitations of the robot, attempt to generate a smooth trajectory from the generated zig-zag. This would allow the robot to have faster mission times, and not need to slow down before entering a tight turn.

Sensor Fusion and PID Controller Software Effort

The perception and control stages of the robot are accomplished via a sensor fusion algorithm that, via a Kalman Filter, combines data from the robot's GPS and IMU, and PID controller.

Since the team did not have access to its hardware while designing our sensor fusion algorithm, the team created Python scripts to simulate GPS and IMU data. To do this, the team created ground truth location and orientation data, and added noise to it, informed by variance information found in our sensors' data-sheets. The data is fused in the Kalman filter and combined with known control inputs from the vehicle and a physics-based plant model of the vehicle's dynamics. To build our filter, the team combined filtered IMU quaternion orientation data with IMU acceleration measurements to generate world-frame acceleration measurements, and combined that with GPS data, the vehicle's control inputs, and our plant, to estimate the true location and orientation of the vehicle. The plant model is a simple 3-D constant acceleration model combined with drag estimation (where drag is estimated as a second degree function of velocity), and a constant buoyancy and gravity term. In testing, the team was able to demonstrate that

the filter significantly reduces the amount of noise in the system. That being said, the filter's job was made significantly easier by the fact that the team was generating sensor noise– Kalman Filters assume that sensor variance comes from a Gaussian distribution; as such, given that our noise was also pulled from a Gaussian, the team made the filter's job a lot easier. Further development would test the filter in the field to evaluate its performance given non-Gaussian data. A sample of our Kalman filter testing is included in Appendix figure 4.6.

The filter feeds data into a PID controller that the team developed. To design the PID, the team elected to make two simplifying assumptions about the vehicle physics: that the team could decouple its planar dynamics (motion along a constant depth) from its depth dynamics. This is an assumption that many similar vehicles in practice use: most small-scale AUVs typically decouple their depth control from planar control, and attempt to keep the vehicle's nose more or less level with the surface at all times. This meant that the team could treat its vehicle's control as two separate problems: one 2-dimensional MIMO PID, and one 1-D SISO PID. The team also elected to discretize the vehicle's trajectory, and feed the data into the PID at a rate of 1 Hz. In choosing this value, the team realize that the team needed to balance two wishes– if the team fed the desired trajectory data into the robot too quickly, the team would oversaturate the integrators on the PID – the team expected the robot to constantly lag behind the target, causing the integrators to constantly increase. This would result in the robot executing more and more aggressive control outputs.

The team first designed the MIMO control law. The vehicle has three thrusters capable of controlling its planar motion– as such, the team could decouple its forward/backward motion and turning motion. The team built two PID's: the first along the vector distance between the robot and its target. the team treated this distance as the "error" which would generate equivalent control inputs for the robot's main thruster: as such, the farther the robot was from the target, the more it would increase its thrust (up to a limit given by the thrust limitations of the robot's main thruster).

The two side-thrusters of the robot would be in charge with controlling the robot's lateral distance from the target. Based on the lateral distance and forward from the target, the team calculated desired angles--where the nose should point if the robot wanted to point directly at the target. Initially, the team calculated this angle via world-frame coordinates (since our Kalman Filter would be giving us world frame coordinates, and our desired trajectory was in world-frame coordinates). However, this caused problems when the target would pass through the world-frame positive x-axis (the atan2 function the team used to calculate the desired angle would jump, causing excessive control inputs). To avoid this problem, the team localized the desired angle of the robot to the robot's current nose angle, translating the world frame "error" (the difference between the world target coordinates and the world vehicle coordinates) into a frame of reference movable with the vehicle. As long as the vehicle is able to keep its nose somewhat close to the target (within a 180 degree range), the team believed that the vehicle should not run into similar problems while tracking targets, until it gets close to the target. When the vehicle gets close to the target, it risks the target passing the positive x axis of its coordinate frame: to avoid this problem, the team implemented a "good-enough" strategy, where the vehicle would determine that it has reached the target when it gets within 1 meter of the target.

The PID controller for depth control is relatively simple; the team developed a simple PID around an "error" based on the vehicle's current depth data and the desired depth. To test our controller, the team developed simple plant functions in Matlab, using constant acceleration/torque models involving drag estimations, and estimated vehicle buoyancy/mass data. the team calculated drag in the same way the team did when developing the Kalman Filter plant model-- the team treated it as a square function of velocity. The team built a version of our controller in Simulink and tuned the controller gains via root locus plots and bode/margin plots. To test its effectiveness against a desired trajectory, the team had the vehicle follow a relatively tight 5-meter radius circle in Simulink, before implementing it in Python. The outputs of this test is included in Appendix figure 4.7.

Filter Canister Design and Detection Sensor Suite

The first semester filtration canister design and detection sensor suite effort is included in Appendix Figure 4.8-4.17. See Appendix Figures 4.18-25 for renderings of the Free Willy vehicle platform with the filtration canister design.

Conclusion

The development of Free Willy proved to be a challenge that tested the team's mechanical, electrical, and software engineering skills. Overall, the team was able to develop a system that the team believes is robust. The team built a robust vehicle capable of cleaning pollutants with the team's filtration canisters. The vehicle is fully controllable by the team's Raspberry Pi and PCB, and pollutants are analyzed with the team's particle detection sensor. A number of software packages support the system to fully realize it and make it easily usable: an AWS software stack, a communications pipeline, a data dashboard, a trajectory planner, a PID controller, and a sensor fusion algorithm.

V. Self-Learning [Outcomes 7.1, 1.1]

The team selected the Raspberry Pi platform, which all team members learned from scratch. Certain team members also learned Python for the first time for programming on the Pi. This learning was supplemented with online resources. No team members had existing experience in particle detection or in particle filtration. Moreover, this area represented the primary discipline for the first segment of the project. To educate ourselves on this topic, the team performed background research and proceeded to reach out to subject matter experts. To that end, the team had discussions with Dr. Chinedum Osuji and Dr. Douglas Jerolmack about filtration design and particle detection respectively. The team followed up these discussions by prototyping a filtration system and performing a calculation, as well as deciding on a methodology for particle detection. Throughout this process, the team discussed with our primary advisor, Dr. Ani Hsieh, regarding weight and power among other considerations for our vehicle. These discussions structured our constraints for the vehicle, which further defined the constraints for each component of the filtration canister. Specifically, Dr. Hsieh suggested the team move away from the original glider design and consider using a propeller-based vehicle, as exemplified by our final design. The team also benefited from meaningful feedback from peers. This feedback informed the decision to switch to a reservoir clean-up focus.

Our team heavily used AWS to develop our dashboard. No one on the team had any AWS experience prior to this-- rather, to develop our dashboard, the team spent weeks reading through the AWS online documentation, and watching tutorial YouTube videos. Furthermore, no one on the team had any prior experience with sensor fusion– to develop our Kalman Filter, the team talked with classmates that had taken controls classes, and read materials on sensor fusion from Purdue University and Georgia Tech. Lastly, no one on the team had any experience with communications– as such, to develop our communications software, the team talked with classmates that had done BLE communications projects, read through Albert Huang's "Introduction to Bluetooth Programming", and read through documentation online about cellular transmission.
Leveraging our interdisciplinary background, the team was able to successfully learn both the platform and subject matter material necessary to successfully design the Free Willy platform and the microplastic realization on that platform. This was facilitated by our background in certain areas, such as microcontrollers (ESE350), fluid mechanics (MEAM302), and UAVs (MEAM 543).

VI. Ethical and Professional Responsibilities [Outcomes 4.1, 4.3]

Especially since the design proposed intends to mitigate acute environmental and social problems, it is imperative that aspects of the design do not worsen environmental or human health problems. Specifically, the design should not:

1) Create an additional source of significant environmental pollution
2) Contaminate the freshwater reservoirs in which they are pursued
3) Lead to any adverse human health problems
4) Create an undue financial burden on municipalities

The team intends to design our pollution control platform within the bounds of this social context. One ethical issue the team may face is being able to prove that enough pollutants have been removed so that the pollution threat is neutralized. The team plans to address this issue by providing access to pollution collection data to all prospective clients in reservoir clean-ups while acknowledging limits to our solution when they arise.

VII. Meetings

The team established a standing meeting time of Tuesdays from 6:30-9PM. In the fall, these meetings began on Tuesday, September 17th and continued as a standing meeting until Tuesday, November 19th. In the Spring meetings began on Tuesday, January 21st and continued as a standing meeting until Spring break.
The team had a meeting with Dr. Jerolmack to discuss particle detection considerations on November 7.
The team had a meeting with Dr. Osuji on October 16th and with Yizhou Zhang, one of Dr. Osuji's postdoctorate students, on September 25th.
The team had meetings with our primary advisor, Dr. Ani Hsieh, on November 22nd, September 9th, February 14th. Furthermore, the team sent detailed email updates on the project on March 19th.
Additional team meetings occurred ad hoc to address specific aspects of the project such as the canister design, vehicle CAD, detection system, etc.
In addition, the team has communicated over either text or call at least twice a week throughout the year.

VIII. Proposed Schedule and Milestones [Outcome 5.3]

The team made the decision to segment the project into two primary deliverables:
1) A fully realized microplastic filtration canister with a control system and sensor suite
2) A pollution control platform vehicle with a microplastic filtration realization of the platform

The team addressed the first deliverable during the fall semester and the second deliverable during the spring semester.
The team further divided the filtration canister realization to the following specifications:
- Proof of concept particle detection – MILESTONE MET on October 17th.
- Basic sizing estimates of the filtration canister – MILESTONE MET on October 17th.
- Characterization of micro filter – MILESTONE MET on November 5th.
- Completion of water flow sensor design and driver programming – MILESTONE MET on November 5th.
- Binary detection of presence or absence of added microplastic particles – MILESTONE MET on November 14th.
- Complete CAD realization of canister design – MILESTONE MET on November 14th.
- Complete canister design sensor suite (detection and flow sensors) – MILESTONE MET on November 22nd.
- Complete CAD realization of vehicle – MILESTONE MET on November 22nd.
- Complete manufacturing of filtration canister – MILESTONE MET on November 22nd.
- Conclude testing and verification of filtration canister – MILESTONE MET on December 10th.
- Completed electrical floorplan of vehicle; order PCB – MILESTONE PARTIALLY MET on February 28th. The PCB was never ordered because of shipping delays on the seascooter followed by the coronavirus pandemic.
- Completed manufacturing of vehicle, begin integration – MILESTONE POSTPONED due to coronavirus pandemic.
- Testing and verification of fully integrated vehicle and canister – MILESTONE POSTPONED due to coronavirus pandemic.
- Completed AWS software stack for mission tracking and data analysis – MILESTONE COMPLETED on March 18th.
- Completed trajectory planner algorithm – MILESTONE COMPLETED on March 20th.

-Completed PID controller – MILESTONE COMPLETED on March 21st.

-Completed Kalman Filter – MILESTONE COMPLETED on March 24th.

IX. Discussion of Teamwork [Outcome 5.2]

The overall system architecture design was performed as a team exercise, as was the planning of each stage of design. As an interdisciplinary project, there was a natural division of tasks relating to the project. To coordinate tasks, the team met frequently during the term to provide updates, and get help on deliverables. The primary areas in which each team member contributed and plan to lead are listed below, although collaboration across segments of the project was a constant throughout the semester:
Wesley, an ESE major, was in charge of the vehicle's electronics, and helped with the software:

Fall: Particle detection system design; Particle detection system signal processing algorithm; Water flow sensor driver programming; Raspberry Pi platform set-up; Vehicle stabilization sensors and control;
Spring: Engine power and control; Electrical floorplan design; PCB Design; Motor Actuation
Roshan, a MEAM major with a keen interest in software, helped with the mechanical design, and was in charge of designing most of the vehicle's software:
Fall: Canister sizing estimates; Vehicle sizing estimates; Filter characterization; Vehicle CAD realization;
Spring: CFD Testing; Vehicle propeller and engine design; AWS Stack Design and implementation; Trajectory Planning algorithm realization; PID controller realization
George, a MEAM major, was in charge of the vehicle mechanical design:
Fall: Filtration canister design; Filtration canister CFD simulation; Filter characterization; Filtration canister manufacturing; Vehicle Waterproofing;
Spring: Vehicle actuator design; Vehicle CAD finalization; Vehicle engine verification; Vehicle manufacturing

X. Budget and Justification [Outcome 2.5]

Purpose	Unit Price	Quantity	Total Price	Purpose	Unit Price	Quantity	Total Price
Particle Detection Materials	$191.44	1	$191.44	Motor for water thrusters.	$26.99	1	$26.99
Digital Water Flow meter	$7.99	1	$7.99	to run between airbox and wet environment	$4.00	7	$28.00
Lens for focusing laser diode (250mm focal length)	$11.50	1	$11.50	watertight electronics box and wet environment that	$4.00	7	$28.00
Mortar and Pestle and Glitter for Microplastic Detection	$21.31	1	$21.31	End cap of watertight electronics enclosure.	$24.00	1	$24.00
Microfilter Fixture	$0.43	8	$3.44	Airbox and O-rings for waterproofing electronics	$99.00	1	$99.00
Waterproofing Flow Rate Sensor	$18.00	1	$18.00	waterproof electronics enclosure	$39.00	1	$39.00
Waterproofing Flow Rate Sensor	$5.62	1	$5.62	Servo for powering	$42.99	1	$42.99
Clampdown Fixture for Flow Rate Sensor	$2.30	4	$9.20	Inserts for screwing together 3d printed parts	$9.57	1	$9.57
O-rings	$12.28	1	$12.28	estimation and location of underwater robot	$73.00	1	$73.00
Flathead Screws for Holding Together Structural Elements	$8.80	1	$8.80	Thrusters for underwater bot	$119.00	2	$238.00
allows camera and laser line of sight access to filter	$7.63	4	$30.52	scooter as the base of our vehicle.	$240.00	1	$240.00
			$320.10				**$848.55**

Figure 10.1: Detailed Budget

This is the final realized budget for the project. The cost for the spring budget was approximately $250 more than originally anticipated because of a choice to retrofit an existing core vehicle instead of manufacturing a vehicle from scratch using materials available in the lab.

XI. Standards and Compliance [Outcome 2.4]

While standards in the United States for the development of autonomous UUVs have not been well-established, the team followed the ASTM standards for the development of UUVs. Specifically, the team relied on:

Standard	Description	Relevance

ASTM F2595-07	Promulgates standards to guide development of autonomous UUVs, including definitions as to what qualifies as a UUV and various communication standards	Specifies format of presented data (such as water column data measurements, water current measurements, and others). Relevant to future presentation of any data collected by pollution control vehicle
ASTM F2541-09	Promulgates standards for autonomy and control requirements for UUVs	Directly specifies recommended commands for autonomous vehicle control and unit standards

Other indirectly relevant standards cited by the now-withdrawn ASTM standards include IEEE/EIA 12207 Industry Implementation of International Standards, ISO/TC 2211 Geographic Information/ Geomatics, IEEE 1003.23 Guide for Developing User Open System Environment (OSE) Profiles, and others. Note that the ASTM UUV standards have been withdrawn without replacement in 2016 and that the IEEE has not promulgated UUV regulations. In fact, one company that operates unmanned sea vehicles (USVs) cites only COLREGS, MARPOL, SOLAS, and STCW (all general international seafaring rules) directly.[3]

XII. Work done since last semester

The fall main goal was understanding how to design a filter to effectively screen out microparticles. Building on physical experiments and CFD simulation from the fall, the team was able to design an improved filtration system in the spring with an angled inlet that deflects particles large enough to cause fouling, such as fish and seaweed, was also tested. In addition, the team made the filter more modular by realizing that a broader need existed for an adaptable filtration platform that can be quickly deployed in order to clean localized spills of specific pollutants.

To fulfill these new design challenges, the team designed a new filtration canister with self-contained filter meshes that can be quickly inserted and removed from the filter. The filter mesh assemblies included two panels sandwiching a filter that slides into a slot in the filter canister. The geometry of this system was created so that a new filter mesh can be cut in as little as 30 seconds using a laser engraver. Finally, the team significantly changed its approach from a neutral-buoyancy engine-propelled vehicle to a much more dynamically controllable submersible based on an off-the-shelf sea scooter for propulsion. The team then built fixtures for the newly designed filtration systems, control systems, and electronics. In addition, the team devised a system for waterproofing the electronics. The team built a number of software packages surrounding the vehicle system– the AWS pipeline, user interface dashboard, trajectory planner, PID controller, and communications package.

XIII. Discussion and Conclusion [Outcome 7.2]

[3] https://sea-machines.com/wp-content/uploads/2018/09/SeaMachines-CodeOfConduct-_Aug-2017.pdf

This senior design project was an effort to solve a need in reservoir pollution cleanup in an innovative way. The final solution of creating a modular filtration, navigation, and sensor package and mounting it onto an off the shelf sea scooter was reached after several iterations of filter, vehicle, and sensor design.

In embarking on this project, the team learned a great deal about engineering design, including setting constraints, managing budgets, iterating, testing, and validating. Finally, the team learned to manage a complex project with several changing goals and due dates. Though the final design was not able to be assembled due to the COVID-19 Lockdown, the team is proud of its work and confident in the robustness of the design of Plastic Free Willy.

XIV. Business Analysis and Review of Value Proposition (M&T Integration Lab)

Free Willy is designed to be sold as a system directly to customers interested in performing reservoir cleanup. The customers will operate the vehicle, with the company providing technical support. In this, the customer will be in charge of generating trajectories for the vehicle, fitting custom filters into our easily detachable filter holder, deploying the vehicle, collecting the vehicle, and disposing of the pollutants the vehicle collects.

In order to articulate a business plan for this product, the first item of business is to understand the underlying market. In China, the water pollution treatment equipment manufacturing industry is an 83B industry with a projected CAGR of 8.7% over the next five years. Out of this industry, water pollution control equipment accounts for 39% of this industry. With reservoir treatment at 5% of this industry, there is an overall market of $1.61B in China for pollution control in reservoirs. In this industry, most competitors are domestic, although most manufacturers of pollution control systems are purpose built and static. However, our product is differentiated by its ability to be designed for any pollution type, which can give us a competitive advantage. (Zhang 2019)

Finally, the Chinese market is 36.5% of the overall global air and fluid filtration market, which gives a projected overall market of $4.6B for reservoir pollution control. (Freedonia Focus Reports 2017). Though most of the reservoir pollution control market is dominated by large-fixed equipment for common pollutants, a significant need exists for non-stationary dynamically deployable pollution control to control one-off pollution events. A strong offering for this market would be differentiated by being affordable, easily deployable, and able to be quickly refitted to filter a specific pollutant.

As far as pricing the plan is to adopt a cost plus price strategy of $350. Estimates are that a single unit would cost $325 to produce, including the off-the-shelf sea scooter. Because by design this product needs to be bought in bulk, the product is expected to be profitable on the basis of volume purchased. That being said, the largest part of the production cost is the cost of the sea scooter. If a partnership can be forged with a sea-scooter brand such as Yamaha, costs could be potentially lowered further, which would further aid gross margins. Fixed production cost is also expected to be minimal, as the only major components that are being made in house are the filtration canisters. The tooling required for this is based on relatively simple plastic injection molding that is not expected to be a major cost center for the company.

Because reservoirs are typically operated by municipalities, the customer base is well-known and accessible. As such, a direct-to-consumer sales and distribution strategy would best serve these municipalities, the target market. If a partnership can be forged with a major sea scooter manufacturer, the company could leverage the existing distribution and servicing network of a company such as Yamaha to further reduce sales and servicing cost.

Finally, there currently are no competitors in this space, although the sea scooter manufacturers are likely to be the first to become competitors due to their ability to adapt their devices the same way the company is.

This can be potentially solved using patents, and likely patentable elements include the modular filtration system, system for mounting the filters and control systems onto the sea scooters, and the mission planning dashboard, including the trajectory planner and data dashboard. These potential five patents are certainly novel, as no patents exist describing the precise implementation the team created. However, whether the patents are nonobvious remains to be seen. A thorough patent search and evaluation of patentability pursuant to the 5 KSR Factors for Nonobviousness is necessary to make a final determination for patentability. The success of this venture against competitors hinges on having defensible intellectual property.

XV. Appendices

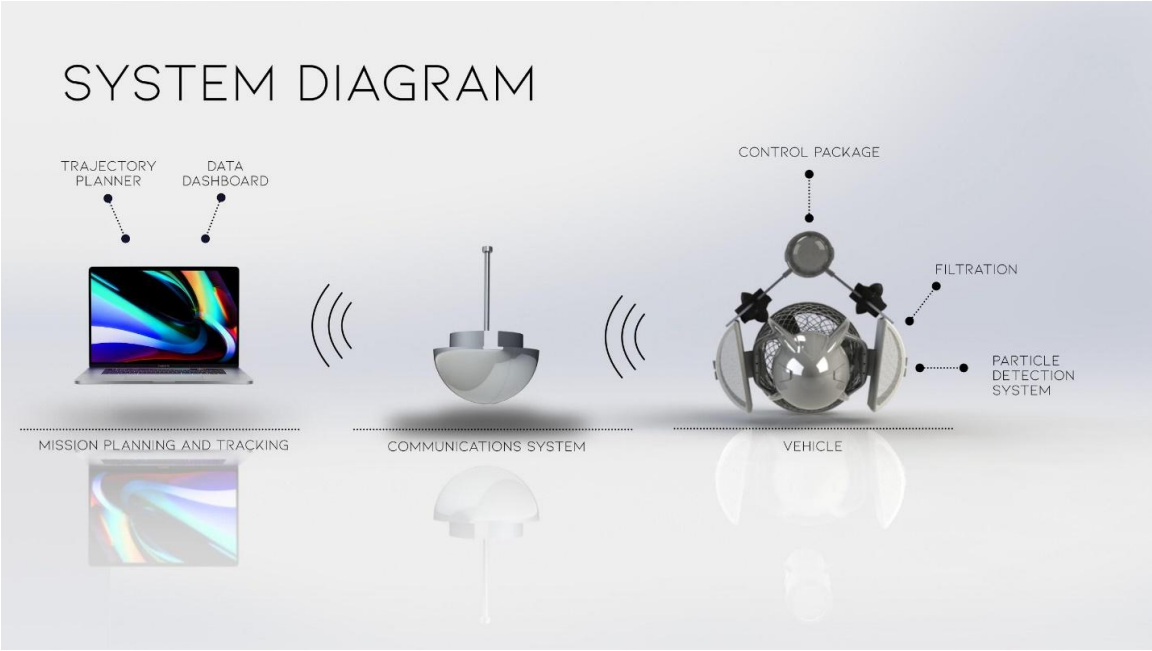


Figure 3.1: System Diagram

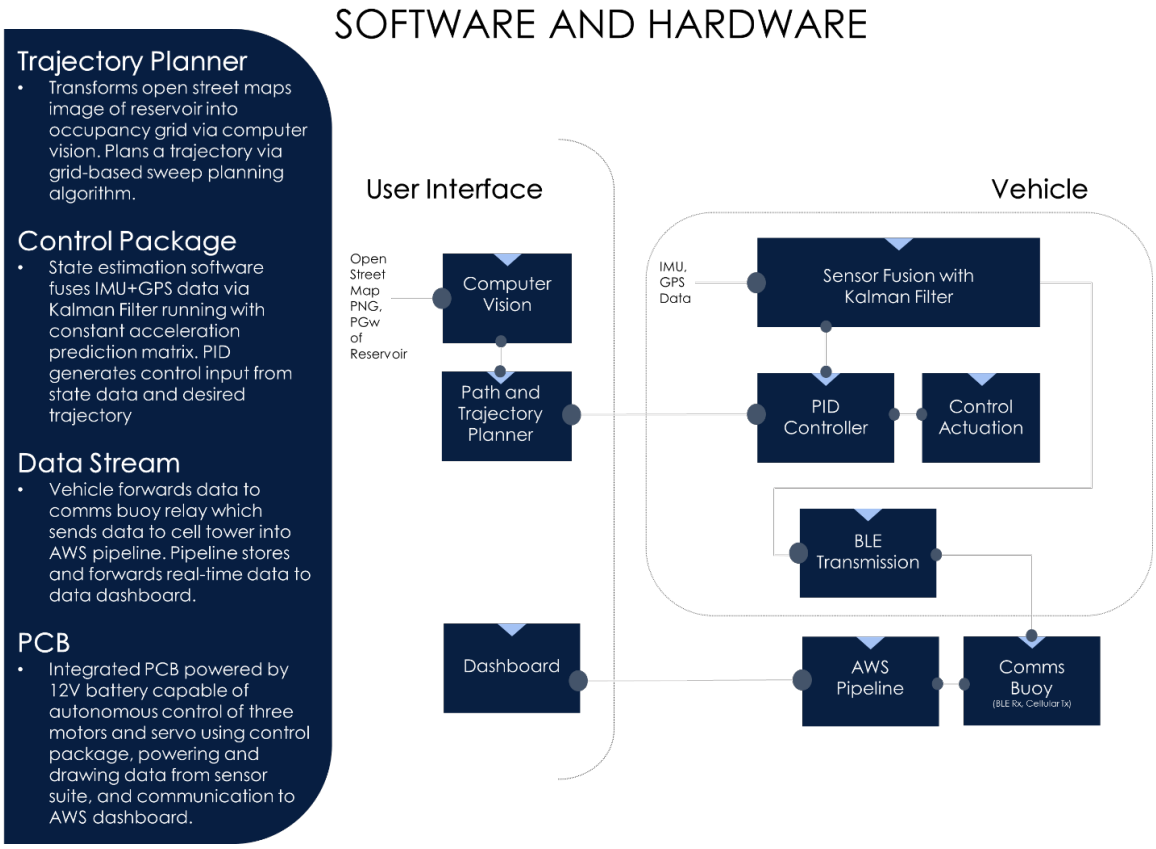


Figure 3.2: Free Willy Platform Block Diagram

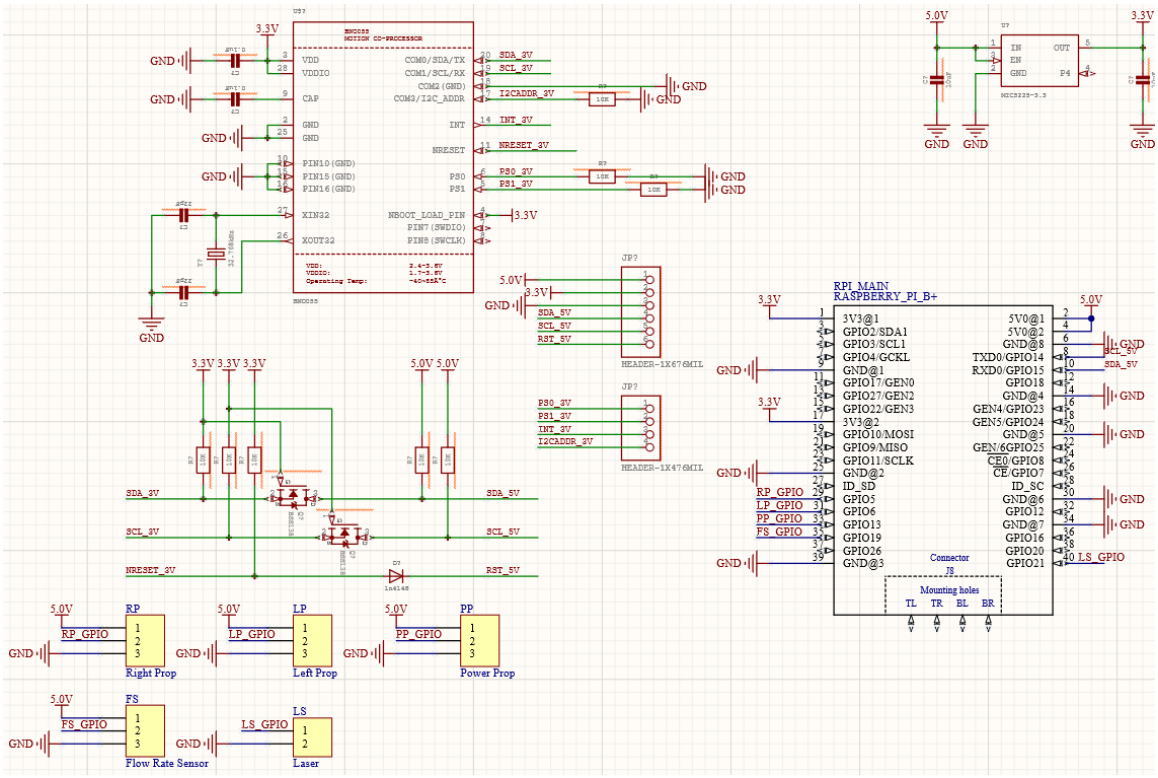


Figure 4.1: Electrical Floorplan Schematic

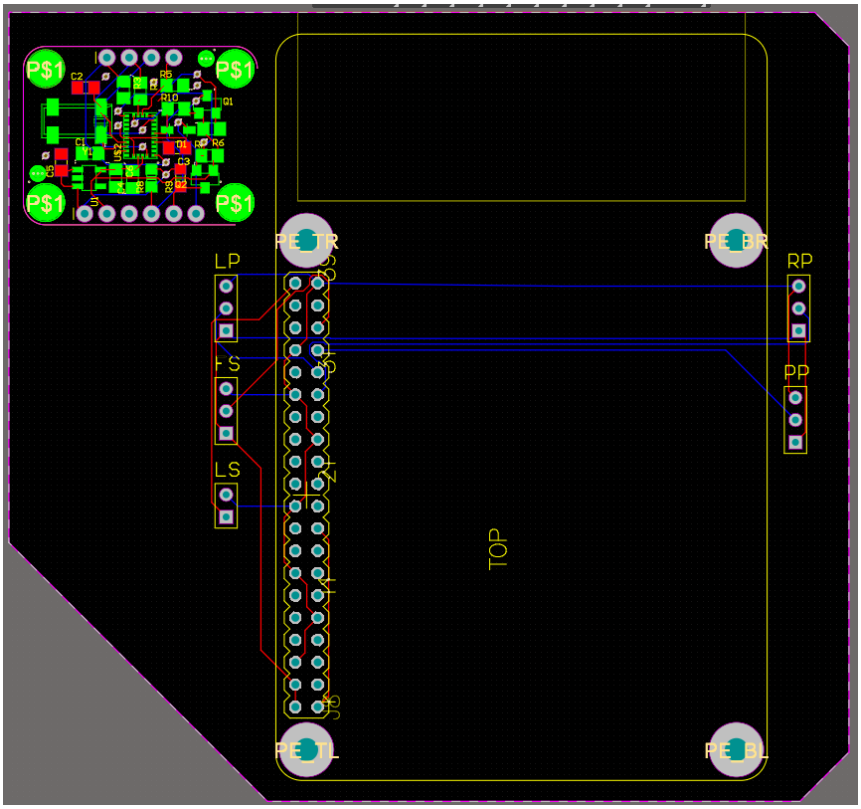


Figure 4.2: PCB

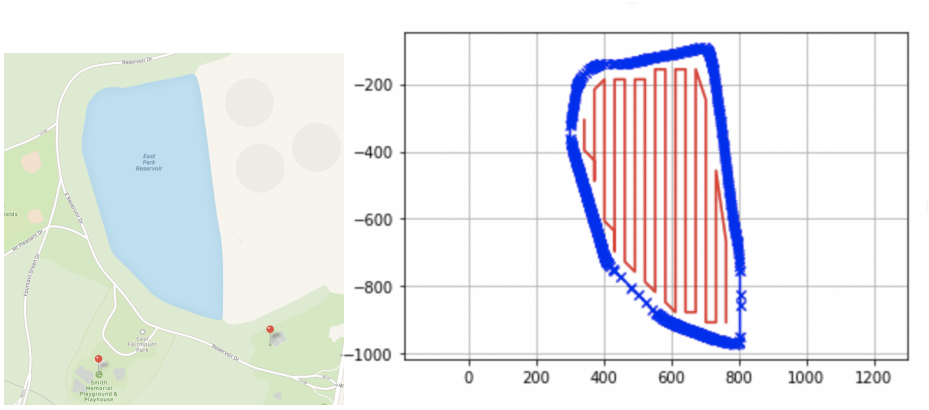
An "Act" function in IOT core forwards the subscribed data to a Lambda function, which sends the data to two places. The first place it sends the data to is a DynamoDB table. The DynamoDB is an easily scalable storage container that can be updated and polled in real time. As such, the team used Dynamo to send data directly to our dashboard; however, the team clear the data in Dynamo after every mission, as the cost of storing large amounts of data in Dynamo is high. A Lambda function grabs data from Dynamo and forwards it to AWS API Gateway, where the data is served to our frontend dashboard. Currently, the dashboard reads in three data types: location data (latitude and longitude), orientation data (Euler angles-- roll, pitch, and yaw), and particle detection data. To display the location data, the team leveraged Mapbox, an online map provider with an easily accessible API. To allow orientation to be easily digestible, it is displayed with vector graphics– for example, if Free Willy rolls from 0 degrees to 40 degrees, a vector image of the vehicle on the dashboard will roll from 0 to 40 degrees. When the system is deployed on the field, the dashboard can be easily adapted to different uses: if a user wishes to access data from a different sensor, for example, the dashboard can be modified within minutes to display the new sensor's data.

The lambda function taking data from IOT Core also sends the data to Kinesis Firehose, which sends the data to an Amazon S3 bucket, where the data is stored (at low cost) indefinitely, allowing organizations to perform post-mission data analysis. A block diagram of our full AWS pipeline is included in the figure above. Information about transmitting data from the Free Willy to the AWS stack using a hypothetical buoy is described in the Appendix below (figure 4.4).

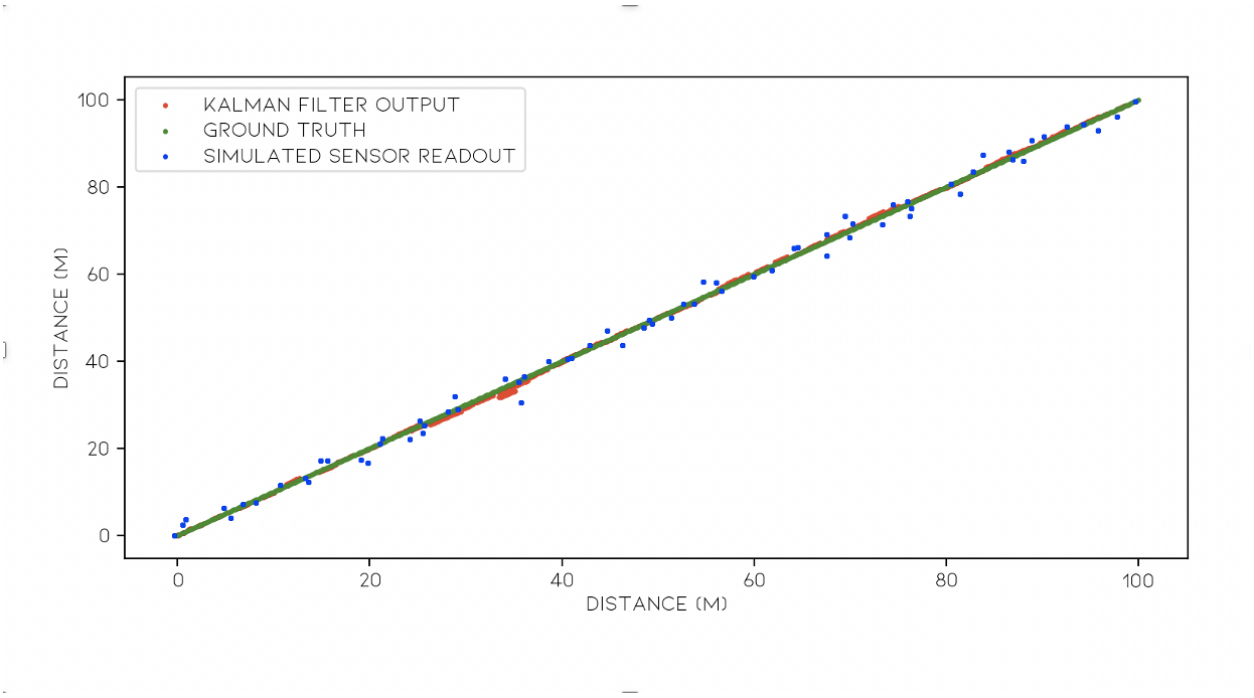
Figure 4.3: AWS Stack and Description

To actually get data from Free Willy to the AWS stack, the team elected to form a centralized communications structure centered on a floating communications buoy. This buoy would communicate to AWS via cellular transmission; it would contain a raspberry pi with a Hologram NOVA chip with a UBLOX SARA-U201 RX/TX module. The module would transmit over Band 5 (850 MHz)-- Band 5 is the band typically used by cellular providers in most areas the team surveyed, at 18 dBm. Vehicles are designed to communicate with the buoy via BLE. To enable this, the team developed cellular and BLE transmission software, although the team were unable either. the team elected to centralize the communication for two reasons. Firstly, the team simply did not have space on Free Willy for the added hardware: by analyzing the expected power the communications system will take up during its idle and active stages, the team were able to size a battery for the vehicle. the team determined that, although our battery is small, due to the severe space constraints the team had in the vehicle, the team would be unable to place the battery (or increase the size of our current battery) and communication hardware. Furthermore, communication from Free Willy via cellular transmission requires a large antenna; this antenna would be subject to water-logging and add too much volume and weight to the vehicle.

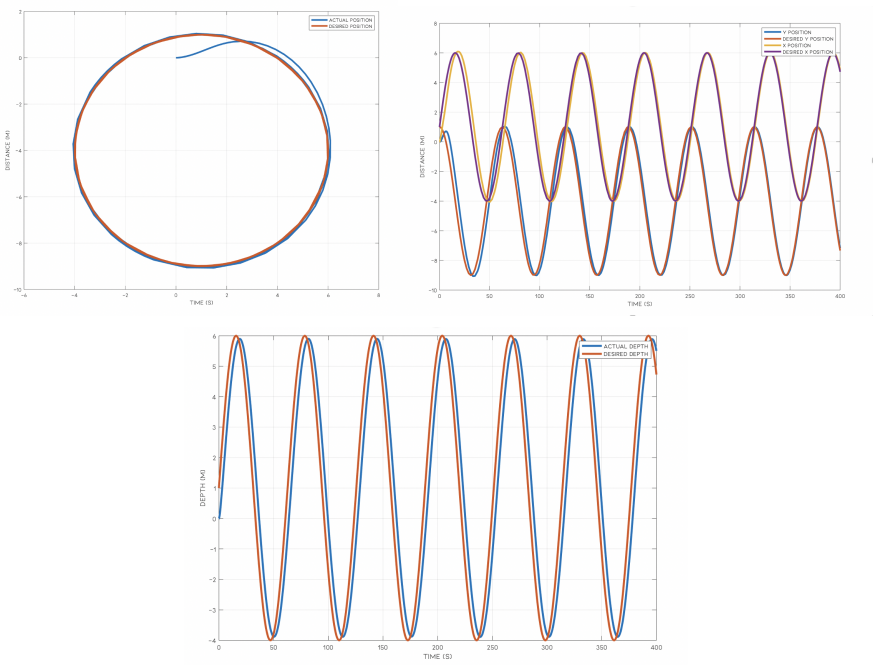
Figure 4.4: Hypothetical Buoy Set-up



Appendix Figure 4.5: Trajectory Planner Output. On the left is a maps image of the targeted reservoir; on the right is a 2D image of the desired trajectory.



Appendix Figure 4.6: Kalman Filter testing. The team simulated data from an IMU and GPS, and put it through our Kalman filter to see how much the filter would be able to eliminate sensor noise.



Appendix Figure 4.7: PID Controller testing. The team had the simulated vehicle travel repeatedly in a 5 meter radius circle to test the MIMO planar controller; the image on the upper left shows the path the vehicle took, whereas the image on the upper right shows the vehicle's progress through time. The bottom image shows our depth test; we had the simulated undergo rapid surfacing and diving maneuvers, to test our SISO depth controller.

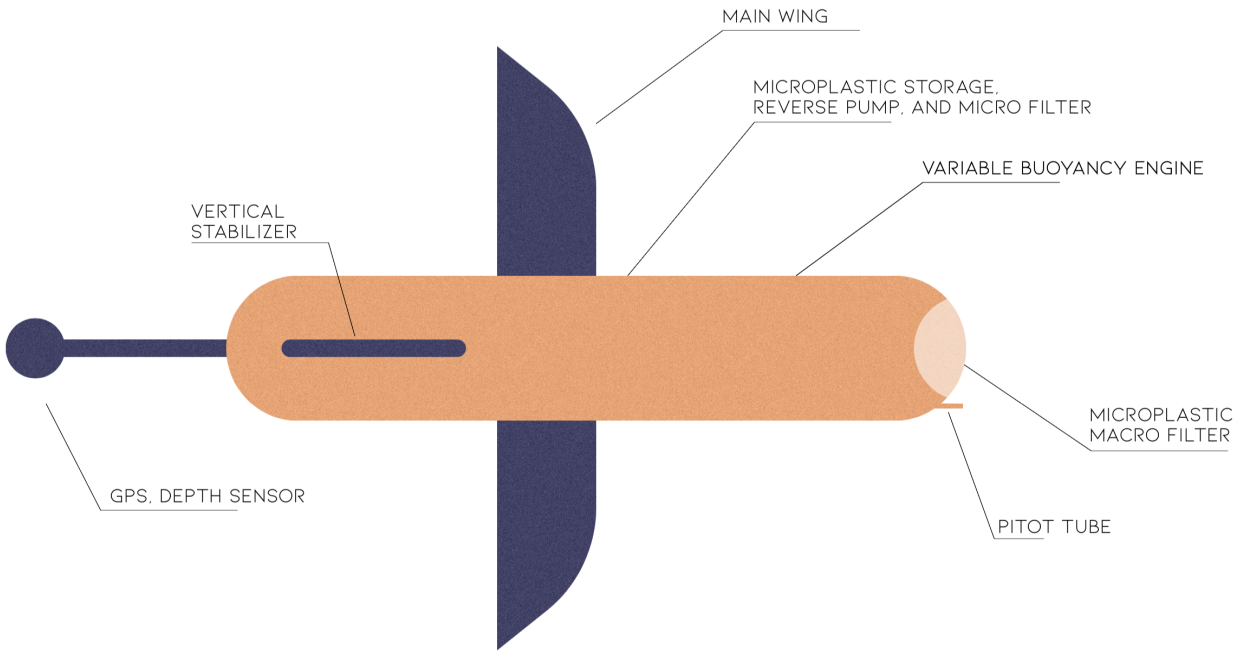


Figure 4.8: Submersible Block Schematic

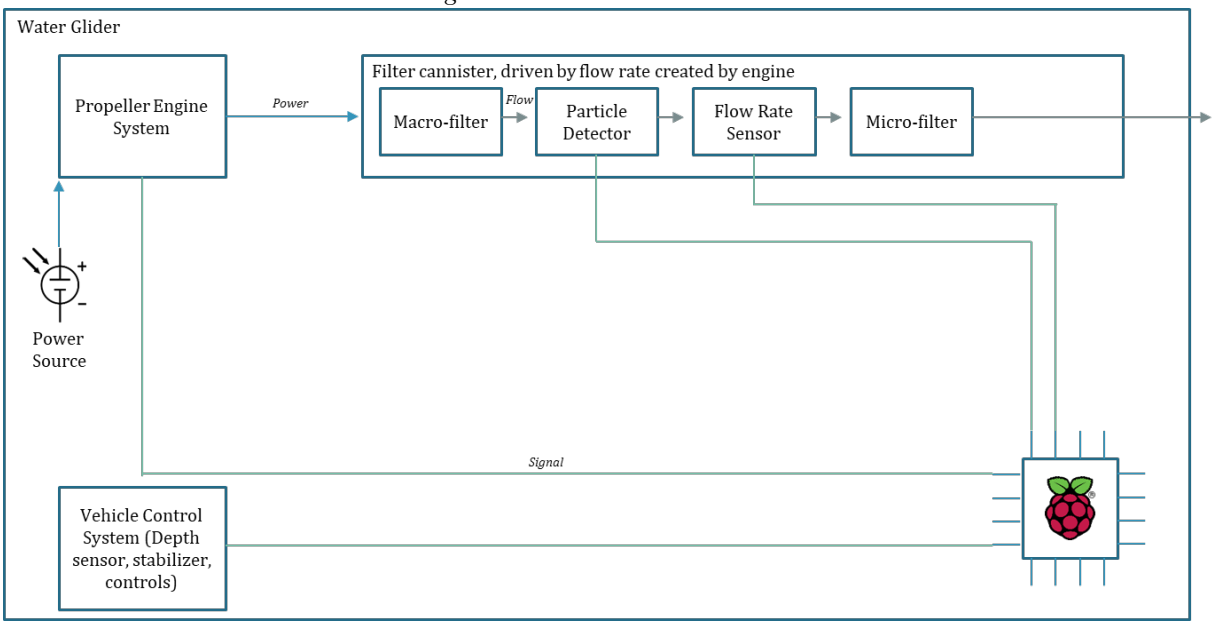
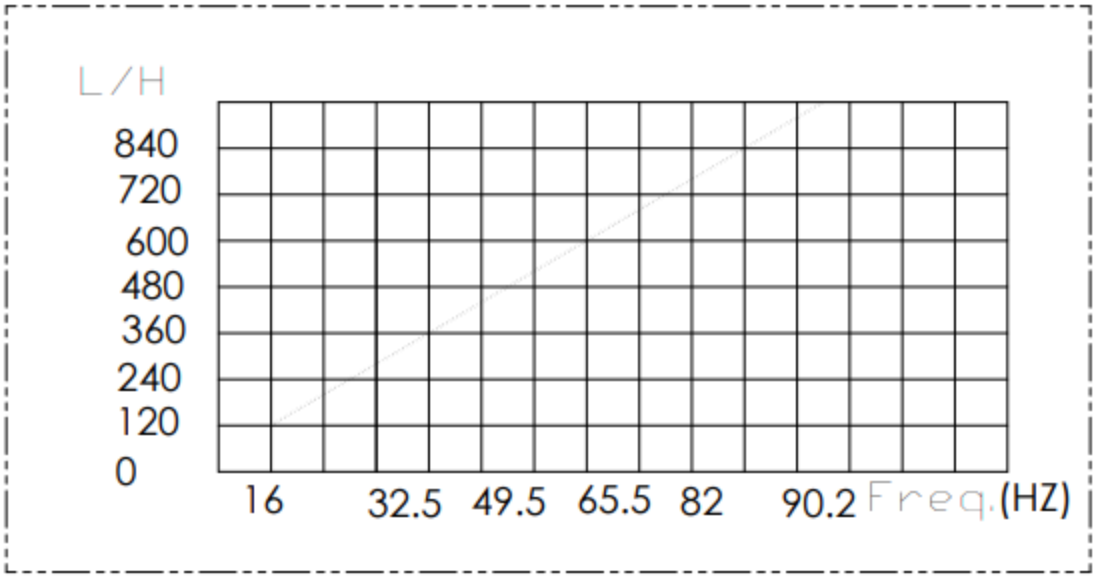


Figure 4.9: Water Glider System Block Diagram

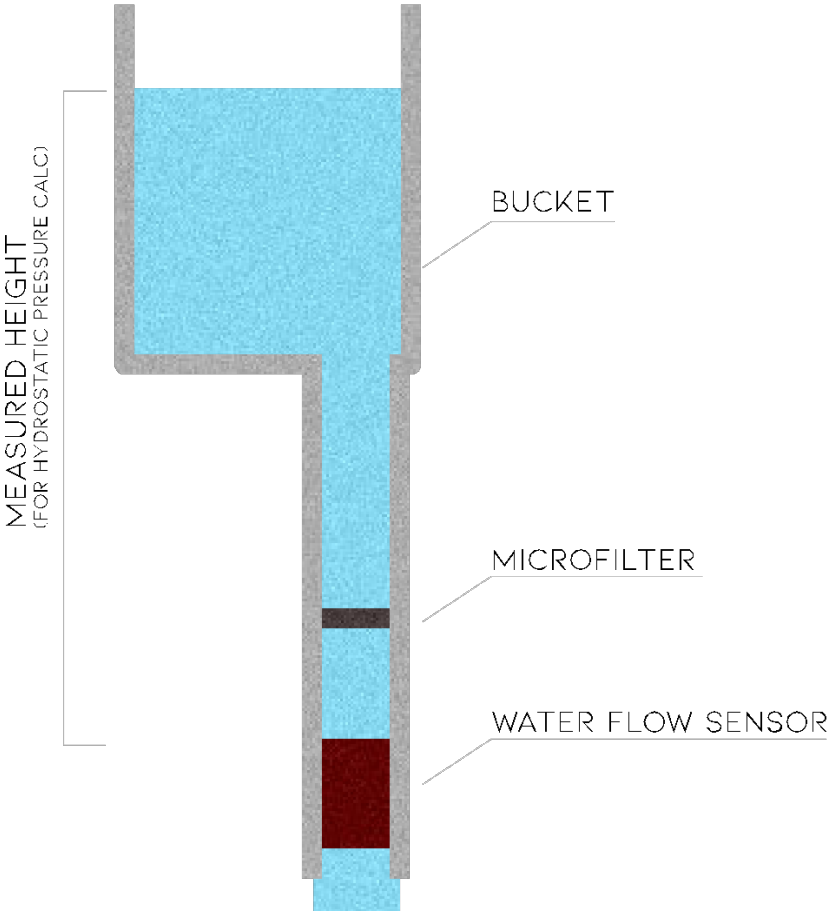
Flow (L/H)	Frezq.(HZ)	Erro range
120L/H	16	±10
240L/H	32.5	
360L/H	49.3	
480L/H	65.5	
600L/H	82	
720L/H	90.2	



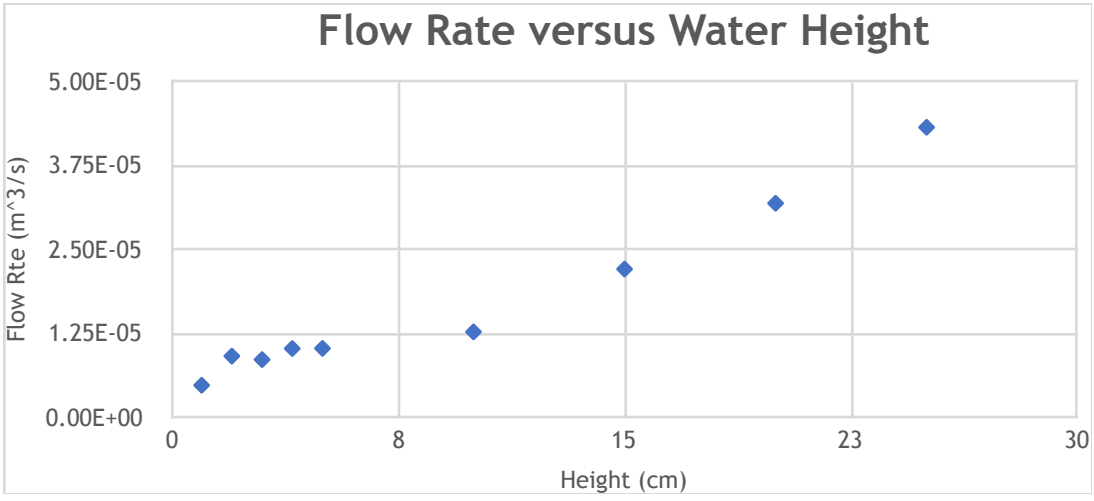
Appendix Figure 4.10: Water Flow Sensor Response (in L/hour)

```python
#Setup library and input pin
from time import sleep
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
flow_pin = 12
GPIO.setup(flow_pin, GPIO.IN)

#Detect rising edge
def waterflow(input_pin):
    edge = GPIO.wait_for_edge(input_pin, GPIO.RISING, timeout = 2000)

    if edge is not None:
        period_start = time.time()
    else: return 0

    edge = GPIO.wait_for_edge(input_pin, GPIO.RISING, timeout = 2000);

    if edge is not None:
        period_end = time.time()
    else: return 0

    period = period_end - period_start

    freq = 1 / period
    flow_rate = 7.2592*freq + 3.6434

    return flow_rate

print(waterflow(flow_pin))
```

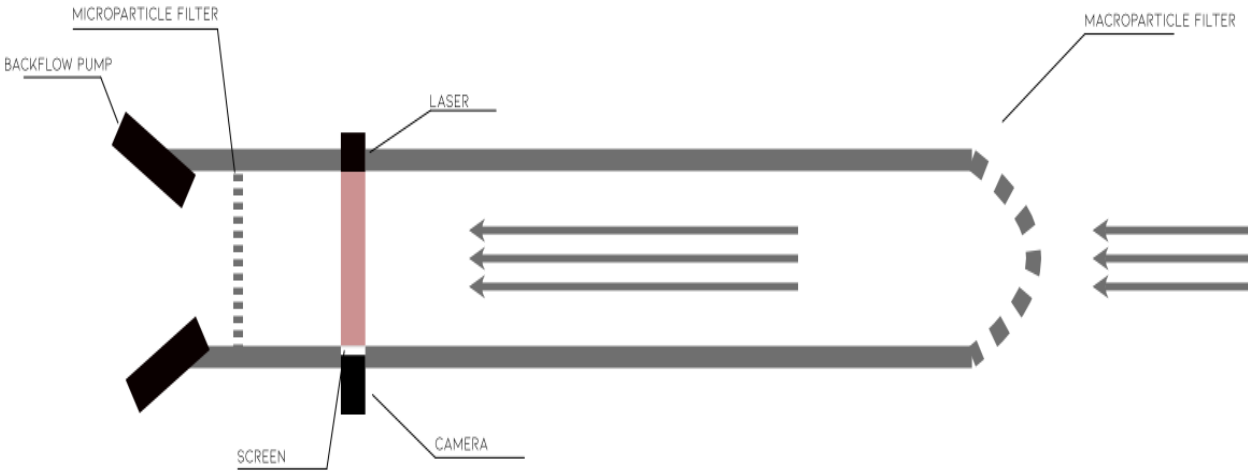
Appendix Figure 4.11: Water Flow Sensor Driver



Appendix Figure 4.12: Microfilter Characterization with Water Flow Sensor



Appendix Figure 4.13: Microplastic Filter Flow rate vs. Height



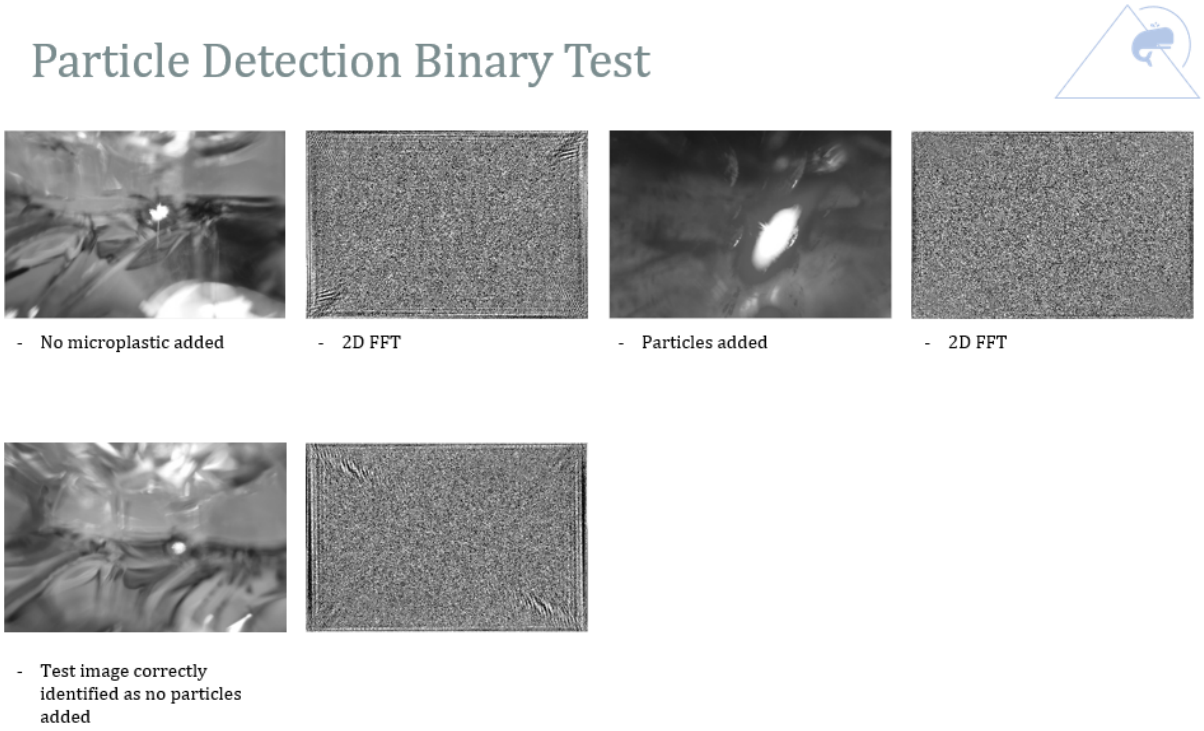
Appendix Figure 4.14: Particle Detection System Schematic

```python
#Setup camera library and global camera counter
from picamera import PiCamera
from time import sleep
camera_count = 0

#Setup GPIO library
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
laser_pin = 11
GPIO.setup(laser_pin, GPIO.OUT)
time.sleep(1)
GPIO.output(laser_pin, True)

#Setup PCA libraries
import numpy as np
from numpy import linalg as LA

#Take camera resources
camera = PiCamera()

#setup library
from PIL import Image

#Define pca function
def pca(arr):
    arr_fft = np.fft.fft2(arr)
    arr_cov = np.cov(arr_fft)
    arr_eigvs, arr_eigs = LA.eig(arr_cov)
    return np.absolute(arr_eigs)

#Define photo function
def take_photo(my_str):
    global camera_count
    camera_count += 1
    camera.start_preview()
    sleep(1)
    camera.capture(my_str)
    camera.stop_preview()
    return my_str
```

```
41
42  #Define nearest neighbor comparison
43  def nearest_neighbor(sample, train):
44      comparison = np.zeros((train.shape[2]))
45      for i in range(0, train.shape[2]):
46          temp = (sample - train[:,:,i])
47          comparison[i] = LA.norm(temp)
48          val_temp = comparison[i]
49      return np.argmin(comparison), np.amin(comparison)
50
51  #Define training set runs
52  def train_me(num_train):
53      train = np.zeros((480, 480, num_train))
54      for i in range(num_train):
55          my_str = '/home/pi/Desktop/camera/temp' + str(camera_count) + '.p
56
57          raw_input("Press Enter to do train sample")
58          take_photo(my_str)
59
60          #Process picture
61          photo = Image.open(my_str).convert('L')
62          temp_arr = np.asarray(photo)
63          pca_temp = pca(temp_arr)
64          train[:,:,i] = pca_temp
65      #Save training set
66      np.savez('/home/pi/Desktop/camera/train.npz', train)
67      return train
68
69  def test_me(train):
70      raw_input("Press enter to start test")
71      my_str = '/home/pi/Desktop/camera/test.png'
72      take_photo(my_str)
73      photo = Image.open(my_str).convert('L')
74      test_arr = np.asarray(photo)
75      pca_test = pca(test_arr)
76      test_out = nearest_neighbor(pca_test, train)
77      return test_out
78
79  #Conduct sample detection training and testing
80  train_set = train_me(5)
```

```
print(test_me(train_set))
print(test_me(train_set))
print(test_me(train_set))
print(test_me(train_set))

#Cleanup GPIO
GPIO.output(laser_pin, False)
GPIO.cleanup()
```

Appendix Figure 4.15: Particle Detection Program



Appendix Figure 4.16: Grayscale Images from Transmission Test

The team began with sourcing filter meshes that are known to work and building a canister around them. After speaking with Professor Hsieh and Professor Osuji, the team became aware that its filter needed to be resistant to fouling by macroparticles. As such, the team designed a two-stage filter, with the first stage containing an angled mesh with an aperture size of 1mm angled relative to the flow of the water. This angling ensured that the bulk of macroparticles are deflected away from the canister, and the 1mm mesh size ensured that only microparticles (<1mm in diameter) make it to the microfilter. Though the original prototype was a machined box, the final version used a 3D printed design with more curved lines for better hydrodynamics, as informed by CFD simulations.

The team also designed the filter based on physical experiments to understand the relationship between pressure and flow rate through the filters. After verifying the effectiveness of a purchased waterflow sensor with a control trial, the team utilized the analog sensor and the Python driver to characterize the microplastic filter. By using a hydrostatic setup with a filter under a bucket of water, the team was able to

calculate the water flow rates at different pressures on the filter ($P = \rho g h$), which allows for an approximation of the filter's permeability constant k using the following relationship (Darcy's Law):

$$Q = \frac{k A \Delta P}{\mu L}$$

Where Q is the volumetric flow rate in m^3/s, A is the cross-sectional area of the filter in m^2, and L is the filter thickness in m, and μ is the dynamic viscosity of water in this experiment. Using Darcy's law, the team found that the hydrodynamic resistance of the filter varies from 1.38e8 $Pa*s/m^3$ at 25cm (incremental water height above pipe water column height) to 7.61e8 at 1cm.

In addition to developing a waterflow sensor to characterize the filter and detect filter mechanical failures in the canister, the team also developed a particle detection system based on the Raspberry Pi and optical circuit set-up introduced by James A Grant-Jacob *et al* (2019).[4] See the detection system schematic on the filtration canister above.

The particle detection sensor consists of a 5mW 650nm red laser diode (Digikey Product ID 1057) and a Raspberry Pi camera module, as well as the optional addition of a focusing lens and a semitransparent screen. The sensor set-up assumes that the presence of microplastic particles or other micro-scale debris creates an observable transmission pattern difference when the laser diode is shown through a fluid sample and the output camera records the image. If this assumption holds, a signal processing technique should be able to map a series of control sample concentrations of microplastics to the image captured by the camera. In this project, the team deviated from the convolutional neural network approach performed by Grant-Jacob et al. for two reasons: 1) the CNN approach is considerably more computationally intensive and would thus require additional hardware and power budgets on the vehicle, and 2) the CNN approach would be less adaptable to changing fields, as the training data requires WiFi connection and a powerful off-Raspberry Pi computer to re-train the CNN.

Instead, the team developed a signal processing program based on the same kind of techniques used in facial recognition.

First, the team performed a 2D DFT on a grayscale 2D matrix representation of the image as a way to ensure image shift invariance would not impact the nearest neighbor comparison:

$$X_k = \sum_{n=0}^{N-1} e^{-i2\pi k(\frac{n}{N})} x_n$$

Where X_k is the 2D FFT in vector notation, k is the discrete frequency, n is the index vector of multidimensional array x_n and N is the vector notation of the length of the matrix along each index. The team then proceeded to calculate the covariance of the 2D FFT signal and calculated the PCA transform using the unitary matrix defined as the matrix of eigenvectors of the covariance matrix:

$$X_i^{PCA} = T^H(X_k - \underline{X_k})$$

$$T^H = [v_0\, v_1 \ldots\, v_n]$$

$$v_i \; is \; a \; solution \; to: A_i \Sigma = \lambda v_i$$

$$\Sigma = cov(X_k) \; where \; X_k \; is \; a \; vectorized \; 2D \; FFT \; of \; the \; original \; image$$

[4] James A Grant-Jacob et al 2019 *Environ. Res. Commun.* 1 035001

After performing this calculation on a set of training data, the same PCA transformation can be performed on testing data. The resulting PCA signal is compared against the training PCA vectors, and the lowest energy error signal (computed as the squared norm of the error signal) implies that the training image was sampled for that particular concentration. The full realization of this algorithm including GPIO control over the laser is shown in the appendix figures above, including an example of the kind of images captured by the optical set-up shown.

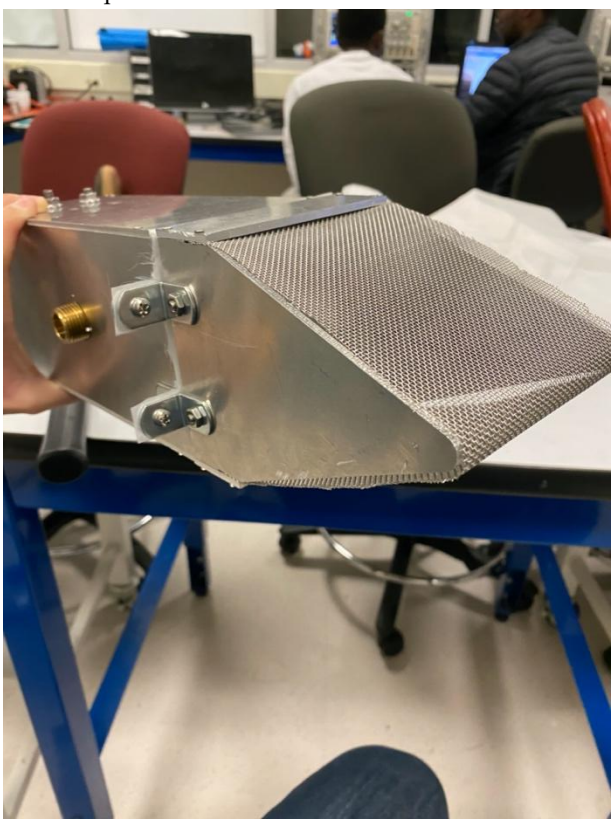
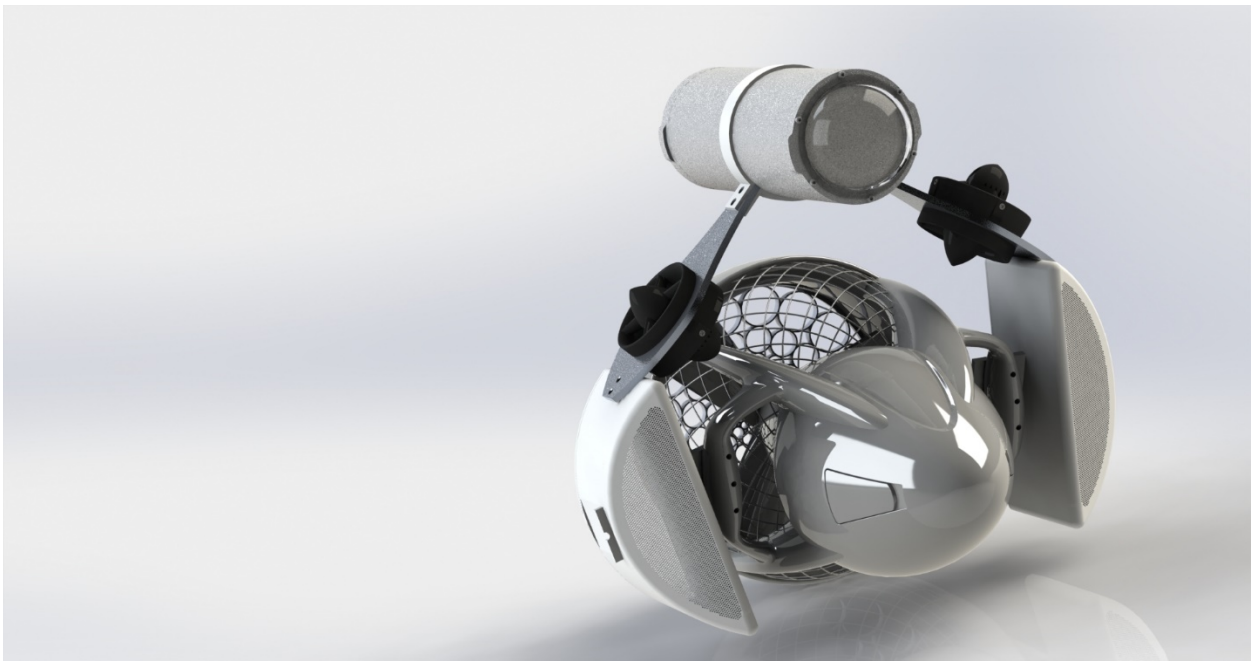
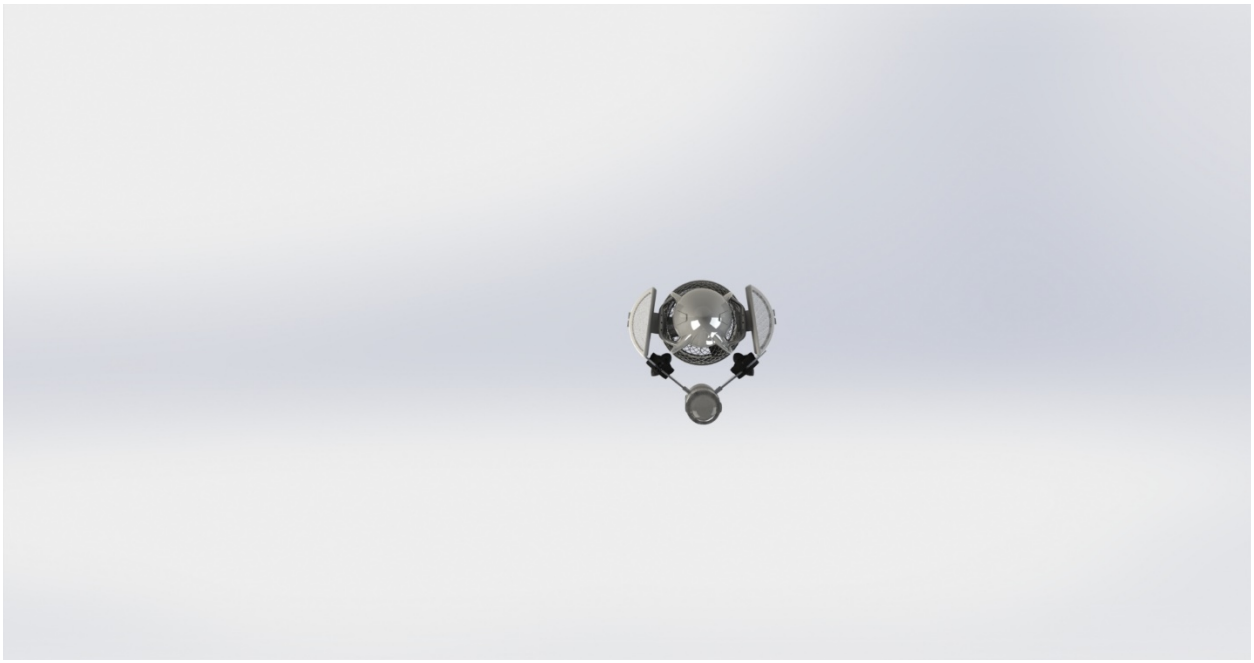
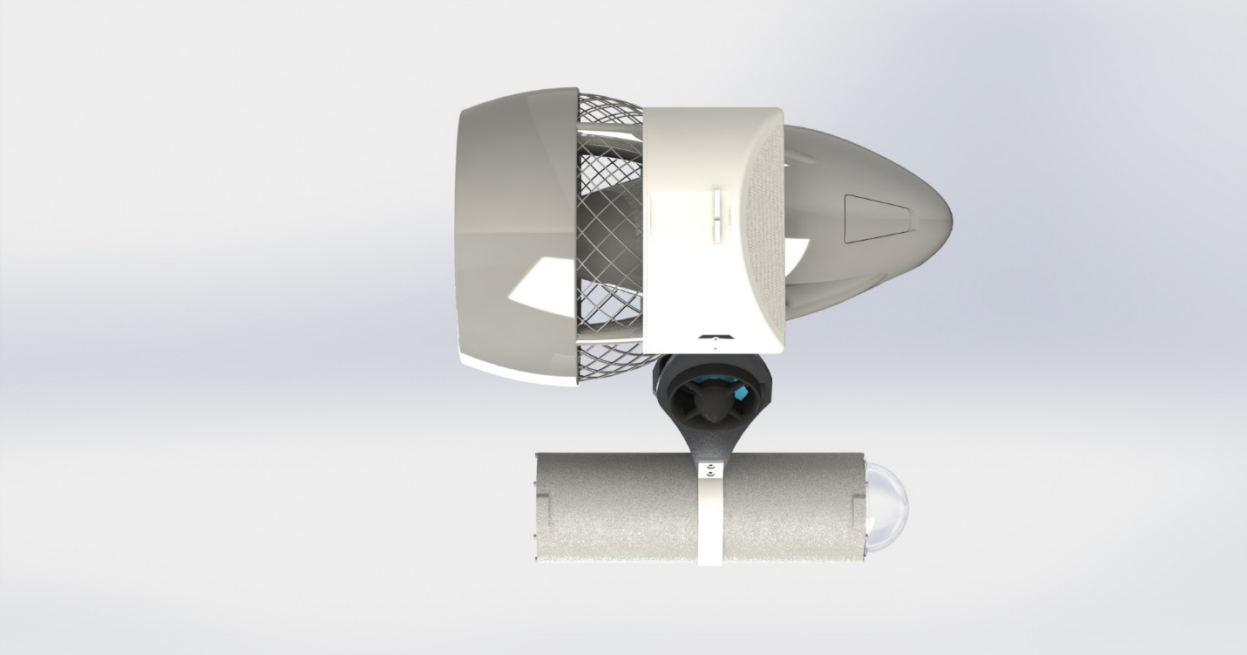
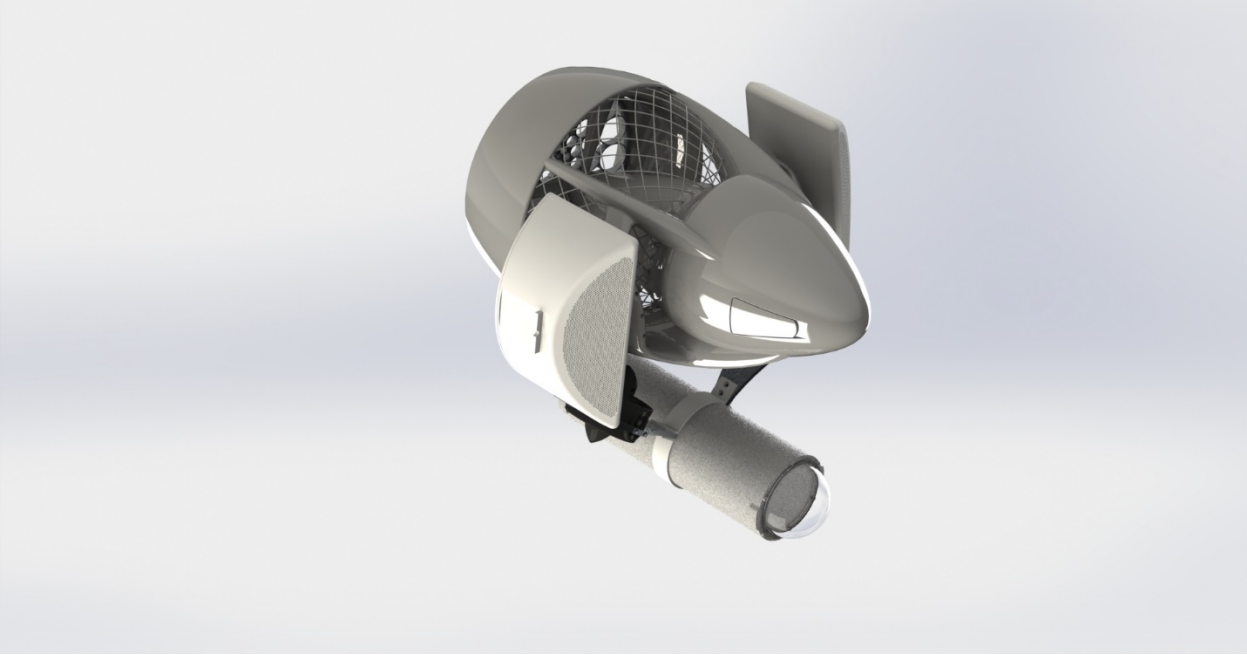


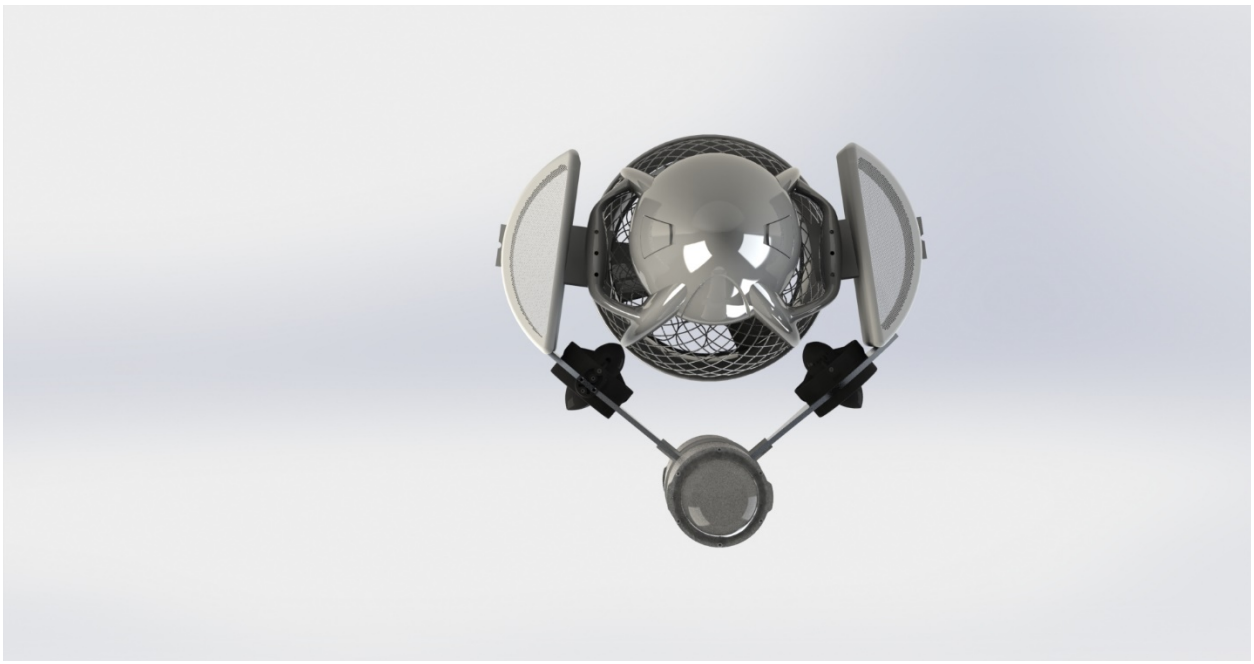
Figure 4.17: First Semester Filtration Canister and Detection System Design Effort

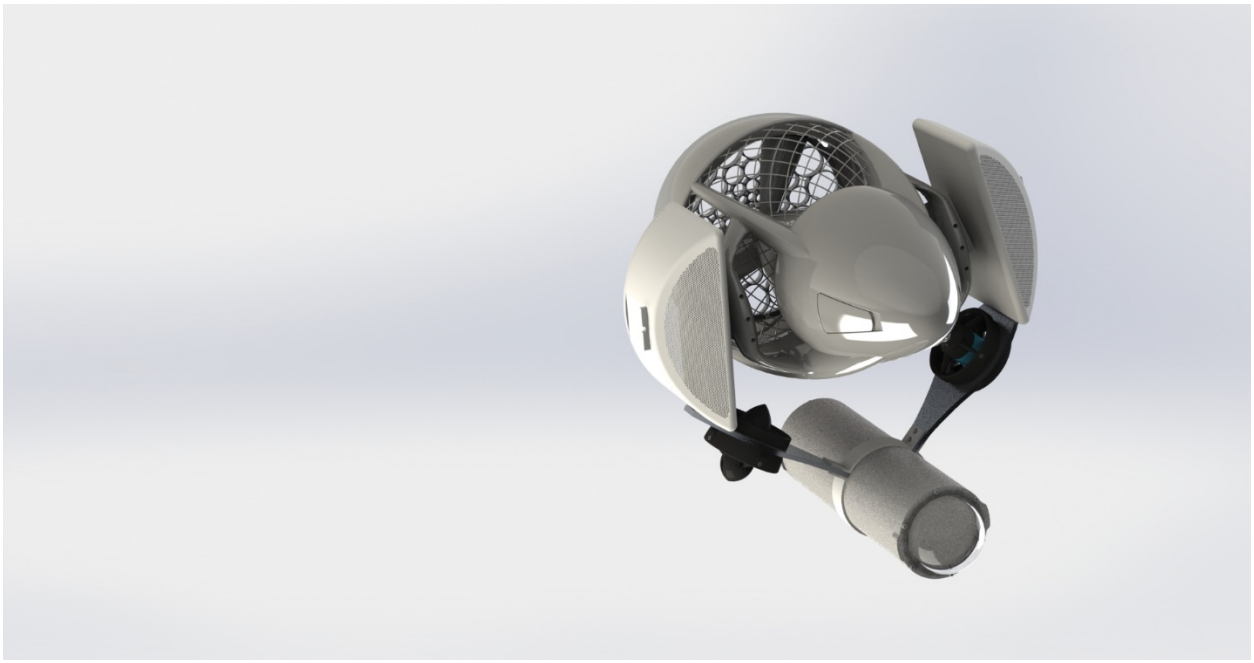


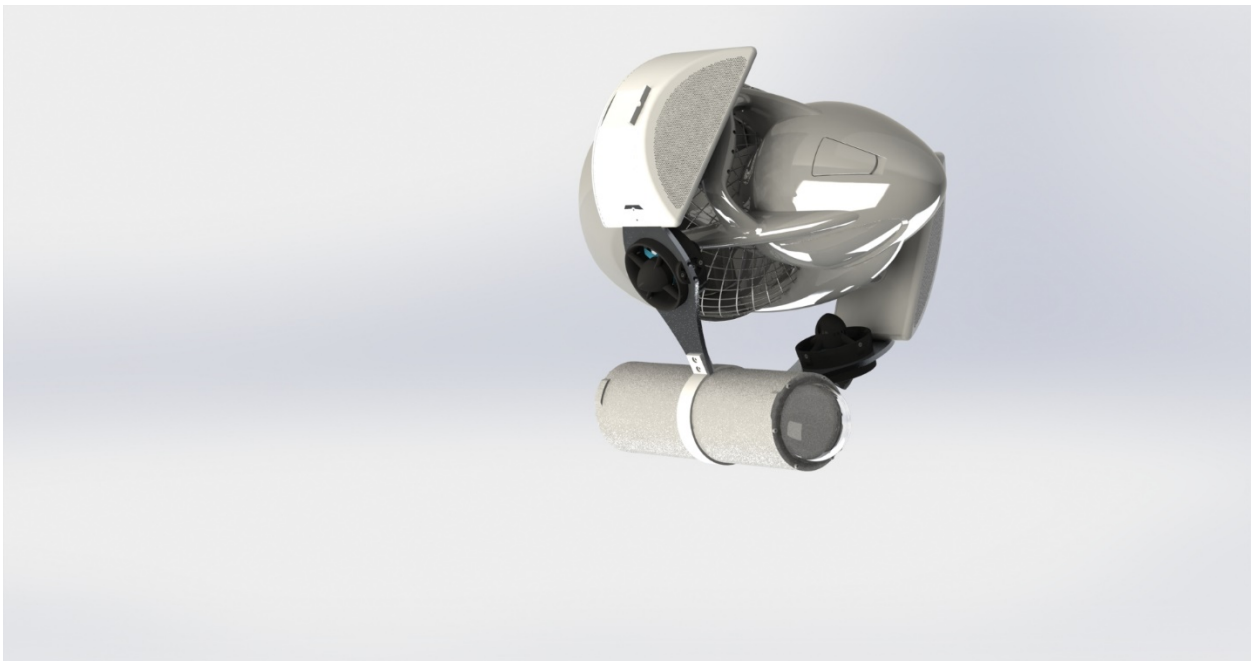












Appendix Figures 4.18-4.25: Renderings of Final Free Willy Vehicle