

For our vertical profile this year we used our base design from last year. This year we wanted to improve our previous model. The first thing we did at the start of ROV this year was test out different non-water absorbent filaments that we could use when printing our cone. This would help with making sure that there were no air bubbles to enhance the precision of reaching neutral buoyancy. Another change we implemented includes fixing our problem of constantly having to unscrew and open our enclosure to change code. This year we decided to re-program our float code to the Arduino Cloud Agent. Arduino Cloud is an all-in-one IoT solution that allows for us to upload code via Wi-Fi, a solution to our problem of constantly reopening our enclosure. Arduino Cloud also allows us to share our projects with each other, allowing for multiple people to work on code at once. With this new change came difficulties. We had to rewrite and adjust our code on the new platform and rewire our entire robot. Initially, we struggled with combining the code for our transmitter, receiver, stepper motor, and pressure sensor, and this took a lot of trial and error before we finally achieved a working code.

Purpose/Intention

This float functions autonomously to collect data for multiple vertical profiles. A vertical profile is a collection of data showing a correlation between a certain quantity compared to height/altitude, and in this case, that quantity is temperature. The float is meant to independently ascend and descend 2.5 meters two separate times, the whole while collecting temperature readings onto an SD card, every 5 seconds, which will be collected after its mission. It is also programmed to hover at 2.5 meters for 45 seconds, or 10 readings, before ascending.

Buoyancy Engine

Buoyancy is defined in physics as the upward force that opposes the weight of a partially or fully submerged object, and it is represented by the equation $F = pVg$. The volume of the object (V), the exterior water pressure (p), and the gravitational force (9.8m/s^2 represented by g) are all equal to the buoyant force, which, when equal to the mass, means that the object is neutrally buoyant. In order to make an object move vertically in the water without the use of motors, one of two things must be able to be altered: The overall mass or the amount of water displaced by the float. This design utilizes the latter. In order to increase the volume and displace more water, the syringe, controlled by a linear actuator, pushes and pulls air in and out of the exterior pouch. When designing this engine, the most challenging part was figuring out the exact point where the float would be neutrally buoyant. When the pouch is inflated more than this point, it causes the float to ascend because the buoyant force is greater than the weight, and when the air is pulled all the way into the syringe, the float displaces the minimum amount of fluid, causing it to descend.

Coding the Linear Actuator

When deciding how we would approach the buoyancy engine, we knew that using a linear actuator would require more than just the Arduino. Since there are more variables than just on and off that the pins on an Arduino control, we needed to use either relays or a motor driver. The main difference between them was that the relays control the actuator by using simple high and low values. Meaning that the relays allow you to make it either open or close; however, there is no way to control the speed at which it works. The motor driver, while being slightly more complicated to develop, allows you to manipulate both. We chose to utilize a motor

driver because, even though speed isn't the most important factor, it allowed us to have that extra variable to work with when troubleshooting the buoyancy.

Materials and Hardware

There are a variety of parts that come from a variety of sources on this float. For starters, the sensor and converter for the sensor are both from the company Bluerobotics, while the syringe, pouch, linear actuator, motor driver, and converters are all sourced from Amazon. Even though it may not be the most fancy to look at, the pouch was made out of a water bottle, as water bottles are designed to keep water in and out, and it would be the most reliable material when working underwater.

Build Process

The process for designing this float started back in December. In the beginning, it took a lot of research and rough ideas before we settled on a design. Since there are so many ways you can create a buoyancy engine, we played around with a couple using very rough builds before narrowing it down to two different designs. While both utilized syringes, they applied them in different ways. The first design used the syringe to pull water into a compartment to increase the mass of the float. The second design, and the one we ultimately ended up going with, used an exterior "swim bladder" that could be used to displace more water. When putting the first design into motion, we began with the mechanical portion of the build. We prioritized the electrical components, knowing that they would need the most troubleshooting in the end. One of the most challenging aspects of the build was figuring out how to get all of the components to run efficiently off one Arduino. In addition, we utilized a wireless Arduino, so we had to make sure that, when at the surface, it would not lose connection with the computer.

How does the Data Collection and Retrieval work?

Since this float is fully autonomous and is not connected to the surface via a tether or any wires, it communicates with my computer via a wifi connection. However, since it travels underneath the surface of the water, and since wifi cannot travel through water, it will lose that connection as soon as it is deployed. The way we circumvented this issue was by having an SD card wired to the Arduino, which collects data readings every 5 seconds as it travels. The data feeds into the SD card during its profile, and once it reaches the surface, it is able to transmit all the data it has collected to the computer's dashboard, where we can then graph it to better show the relationship between temperature, depth, and water pressure. Inside the tube, the Arduino is placed at the top to ensure that it does not lose connection as we are deploying it. The rest of the electrical components are stored at the bottom of the tube so that the weights of the batteries, motor driver, and converters do not affect the weight distribution.