FINAL PROJECT REPORT

Hydrodynamics of Glider Flight

Claire Atkinson

University of Washington, Seattle, WA 98195

Abstract

ABSTRACT

This project investigates the flight mechanics of a class of Autonomous Underwater Vehicles (AUVs) known as gliders. The quantitative ocean data collected by these devices is useful to inform analysis of Earth's oceans. With specific engineering data on real world glider flight as well as public domain ocean environmental models, a custom Python application has been created. This simulation produces data that describes the vehicles' locations and energy states over time. The software is structured such that important parameters are specified in an easily-modified configuration file. The parameters altered include geographic area, the number of gliders, the maximum flight depth, the vehicle's available buoyancy range, and the glide angle. Then, with the data that the simulation produces, variations in energy consumption, uniformity of coverage, and the time required for each glider to reach their destination are analyzed. The oceans, which cover about 70% of the planet's surface, have a huge impact on the climate and health of the Earth as a whole. The result of this analysis is useful to real-world AUV operations by helping determine how to program them to fly more efficiently and maximize their utility as scientific instruments.

Keywords: glider, oceanography, Python simulation, hydrodynamics, ocean physics

1 | Introduction

1.1 Glider Function

Gliders belong to a class of Autonomous Underwater Vehicles (AUVs) that perform vertical profiling of important marine quantities like temperature and salinity. These are then used by oceanographers and others to help them gain a deeper understanding of the ocean environment.

These vehicles utilize the property of buoyancy to move without a propeller. Horizontal velocity is generated by wings, which convert some of the vertical buoyant force to lateral. They achieve volume change by moving oil to fill an external bladder while mass remains constant to adjust their density. When diving, the glider decreases its volume by opening a valve to allow oil back into an internal reservoir. This does not require energy since hydrostatic pressure does all of the work. Even though no actual pumping is done, this is sometimes termed "pumping in". To increase volume and glide upward, a "pump out" maneuver is performed where oil is expelled into an external bladder. This action requires energy to push against hydrostatic pressure.

When flying, the glider follows a parabolic path, where the "apogee depth" is the maximum depth achieved on a given dive. Sometimes a vehicle may perform multiple dives before surfacing in a maneuver called a "yo-yo". The minimum depth reached on these dives is known as the "perigee depth".

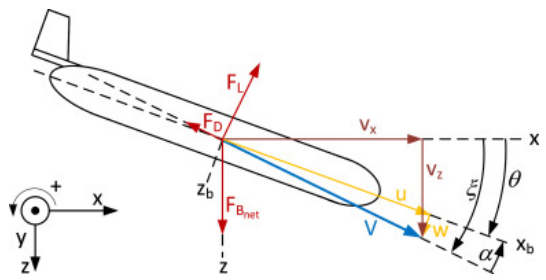

Figure 1: Graphic depicting the relevant quantities for glider flight [2]

The glider uses a movable internal mass to change its pitch angle (denoted θ). However, due to fluid dynamic effects, this is not the exact angle at which the glider flies. That angle is called the glideslope (denoted ξ). The difference between the pitch angle and glideslope is called the angle of attack (denoted α).

1.2 Project Purpose

The glider can only communicate while surfaced, so knowledge of its path during flight is hard to ascertain. This is made increasingly difficult since these vehicles are subject to currents that alter their direction. As such, it surfaces in a location that is difficult to predict since published current models often have large margins of error. Also, the locations of measurements over the dive are only known in the depth dimension.

This project aimed to evaluate and simulate the flight mechanics of these vehicles using Python to better understand the impact of currents and other factors. This can then be applied to real-world glider missions. For instance, better knowledge of such impacts can aid in a more efficient flight path to a given destination.

The guiding questions of this project were twofold:

1. How do factors like current, pitch angle, and apogee depth affect the motion of a glider?
2. Can a simulation be coded that effectively reproduces real-world glider flight?

2 | Methodology

A simulation was created using Python to recreate glider flight. We used a configuration file to make the code scalable for any constants (i.e. coefficient of lift, time ranges, locations, or numbers of gliders). At the beginning of the project, the code ran one glider at a time until the last one finished. By the end the simulation this was altered to run multiple gliders in parallel. Great lengths were taken to make the simulation an accurate reproduction including modeling the buoyancy engine and other components.

To ground the simulation in the real world, our model draws on published ocean model data. To find ocean depth at any given latitude and longitude, we used data from the GEBCO grid, which is a "global terrain model for ocean and land" [4]. The HYbrid Coordinate Ocean Model (HYCOM) was used to estimate currents, temperature, and salinity at a given latitude, longitude, depth, and time [1].

3 | Maximizing Flight Coverage

The first sub-project involved animating the flight paths of several gliders to analyze their ocean coverage. To start, the glider simulation was run with five gliders (still run in series at this point), which

produced a file containing each vehicle's latitude, longitude, and depth over time. This was used to quantify instances of a "circular area coverage gap" (CACG) over time. This was defined as any circle that could fit within the bounding box while not containing the current location of any glider.

I then created a 2D animation of the largest CACG as it evolved over time (Fig 2). Since all gliders flew in a mostly uniform front, the largest CACG was in the top left corner for the first half of flight and in the bottom right in the second half. Probing this in the future, it may be beneficial to stagger glider start times and avoid starting adjacent gliders in order. This would likely increase the coverage achieved by a single glider fleet.

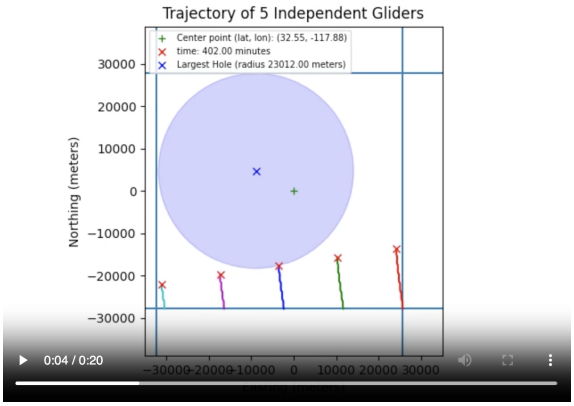

Figure 2: A screenshot of the animation depicting the largest "circular area coverage gap" over time

After creating the 2D animation, efforts were diverted to making a 3D animation of the same flight. This allowed for observation and analysis of idiosyncrasies of flight between each vehicle. For instance, in the middle of the animation, one glider stopped at a much shallower depth than the others. It appears that the vehicle hit the bottom, which was discovered by plotting the sea floor over the bounding region. This helped probe the accuracy of the simulation's function for future sub-projects.

4 | Revamping the Simulation

Over the next four iterations of the simulation, we changed many aspects of the code to improve it. In this effort, many functionalities were added and other previous ones were altered. All of this made the simulation an even closer representation of real-world glider flight.

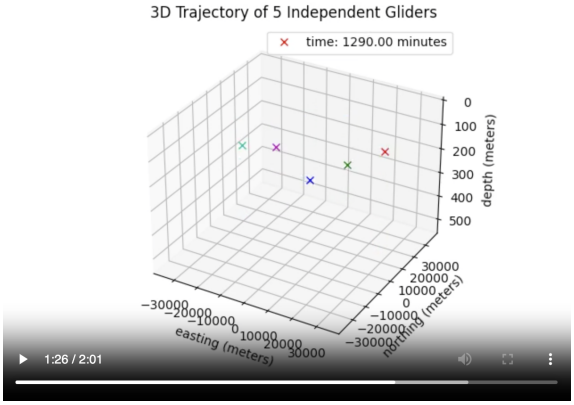

Figure 3: A screenshot of the 3D glider animation. The gliders are represented by the 5 multicolored x's

This was the step at which the configuration file was added to make previously hard-coded values easily mutable. This was divided into subsections delineating general information, location/time details, and glider-specific attributes. This allowed for variation between individual gliders that was not possible in previous iterations of the sim.

Additionally, energy calculations were built in to track the consumption over time while gliding and stop the sim when out of power. This was divided into two modes of energy usage, the hotel load and propulsion. The hotel load is a constant rate of energy consumption that is used continuously over time to run the vehicle's internal appliances. Propulsion, on the other hand, refers to the energy used pumping oil in and out to control the buoyancy state of the glider. Propulsion generally consumes more battery power but over a much smaller amount of time compared to the hotel load.

To calculate the energy usage, we looked to real-world engineering data on the Carina glider (See Table 1). The pressure at any given point is found using the depth which is converted to pressure by a conversion factor. In the buoyancy engine class that contains the functions for moving oil into an out of the internal reservoir, power for a given volume change is given by interpolating for motor power given pressure. The power required increases with depth as pumping out becomes harder with increased hydrostatic pressure to push against. This is supported by the following equation which states that

$$Power = \frac{dV}{dt}(Pressure) \tag{1}$$

where dV/dt represents the flow rate of oil into and

out of the reservoir. The calculated power is then multiplied by the time step of the simulation to produce energy used.

At this stage of the simulation, we instituted a distinct shorter time step for the pumping regime to obtain a higher resolution of the maneuver. This required the final position and energy data to be interpolated onto a common time grid at the end of the simulation so it could be entered into a common file.

The institution of an energy calculation was very helpful in tracking glider state over time since it is a very important real-world consideration during flight. At this point in the simulation, one flaw still remained, this being that energy was consumed in the simulation during both "pump-in" and "pump-out" maneuvers. We realized this in a later iteration of the sim and removed energy calculation while "pumping-in" since movement of oil into the internal reservoir is driven entirely by the pressure of the ocean.

The Carina data also included information on flow rate for given pressures. At first, we also interpolated this value from pressure while pumping. This became more cumbersome than it was worth since, unlike the motor power data, the flow rate seemed to vary around an average value with little to no pattern with depth. So, flow rate was instead made a variable in the configuration file that was set initially to 4 cc/s to match the data.

Then, we looked further into the speed calculation. In the first iteration, max speed was hard-coded, and the velocity was determined as a fraction of the maximum based on the buoyancy state. A maximum buoyancy state meant the glider traveled at the maximum speed. To make this more accurate, we went through a series of buoyancy calculations that were then implemented in the code.

The buoyant force acting on the glider at any given time is equal to the weight of the glider minus the weight of the displaced water. This is true when the downward direction is defined as positive as it is in the simulation. Since the mass of a volume V of seawater at density ρ is ρV, this gives the equation $F_{net} = g(m_{gl} - \rho V)$. Defining $\rho_r V_r$ as the density of seawater and displaced volume of a glider at which it achieves a neutral state, $0 = g(m_{gl} - \rho_r V_r)$, which means that the mass of the glider can be defined as $m_{gl} = \rho_r V_r$. This left us with the following equation for buoyant force

$$F_{net} = g(\rho_r V_r - \rho V) \tag{2}$$

with ρ and V being current seawater density and volume respectively. This makes intuitive sense as well. Increasing the volume from its neutral state ($V > V_r$) while keeping ρ constant will make $F_{net} < 0$, meaning the glider will go toward the surface (decrease in depth). Decreasing the density from its neutral state ($\rho < \rho_r$) while maintaining V, on the other hand, will make $F_{net} > 0$, and the glider will naturally go to greater depths. Other changes will similarly impact glider motion.

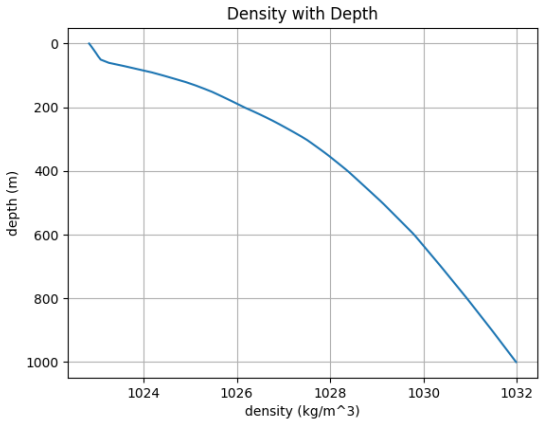

Figure 4: A plot of density over depth produced using HYCOM data at a set latitude and longitude

Although glider volume would be easy to procure, the current simulation did not have a method to find density at a certain depth. Typically, this follows the curve in Figure 4 where it increases sharply over the first 100 meters due to salinity and then slowly grows after that due to further decreases in temperature. To replicate this effect, we were able to use the dens(S, T, P) function from the seawater package of python that takes arguments of salinity, temperature, and pressure. Salinity and temperature were obtained from the HYCOM files while pressure was found using the conversion factor from depth.

Using this calculated buoyancy, we could more accurately calculate the glider speed. To do this, we initially used

$$F_{lift} = \frac{1}{2}\rho v^2(k_l \alpha) = Bcos(\xi) \tag{3}$$

$$F_{drag} = \frac{1}{2}\rho v^2(k_{dp} + k_{di}\alpha^2) = Bsin(\xi) \tag{4}$$

as our equations for dictating glider motion where k_l is the coefficient of lift, k_{dp} is the coefficient of profile drag, and k_{di} is the coefficient of induced drag. The coefficient terms for lift and drag are vehicle-dependant and in our sim took values $k_l = 0.057115$, $k_{dp} = 0.000488$, and $k_{di} = 0.028614$.

These values, when plugged into both equations together, give a quadratic equation that can be used to solve for α at a given ξ:

$$k_{di}\alpha^2 - k_l tan(\xi)\alpha + k_{dp} = 0. \quad (5)$$

Once this relation has been established between the glideslope and angle of attack, the values can be plugged back into the equation for lift force isolating v as follows

$$v = \sqrt{\frac{2Bcos(\xi)}{\alpha k_l \rho}} \quad (6)$$

to find the total through-water velocity of the glider. From this, the horizontal and vertical components of velocity are given by $vcos(\xi)$ and $vsin(\xi)$ respectively. Since we know the angle of attack and glideslope at which these speeds occur, we also know the pitch angle ($\theta = \xi - \alpha$). This produces a shape known as a glide polar that is iconic to glider flight (Fig 8). In our sim, we defined an interpolator that gives the components of horizontal and vertical velocity for the current pitch angle. These are then multiplied by a coefficient determined by the current buoyancy and density to give the actual velocities.

The positive or negative direction of the vertical velocity is determined by the sign of the calculated buoyant force. To direct the glider's motion, the horizontal velocity is further broken down into easting and northing directions based on the nautical bearing between the glider's position and its final destination. The effect of currents is also added in at this step.

In the real world, the pitch angle is set by the glider engineer in the commands written into code. In the simulation, though, there is no one controlling the glider's flight in real-time. To find pitch at a given time step, then, we assume that it scales linearly with the buoyant force. This means that the pitch will be zero at a neutral state, and when buoyant force is maximally positive the glider will have maximum upward pitch (specified in the configuration file). This assumption allows for a smooth apogee maneuver, a process that occurs when the vehicle reaches maximum depth and begins "pumping out" to fly to the surface. Perhaps in later iterations of the simulation it would be beneficial to allow pitch commands to be entered prior to the running of the simulation to better mirror real-world glide environments.

Another change involved moving the functionality from series to parallel. This means that, while the gliders were initially run one at a time, we altered the simulation to run each step or "update" of each glider at the same time. This allowed us to have all gliders keep running until the last one stopped and to better compare their flights side-by-side.

Lastly, we implemented code that produces a kml file after the simulation runs. This allows a user to visualize the surfacing positions of each glider over the flight on Google Earth Pro (Fig 5). Each surface position pin also has a pop-up that contains information on exact latitude and longitude, remaining energy, and simulation time.

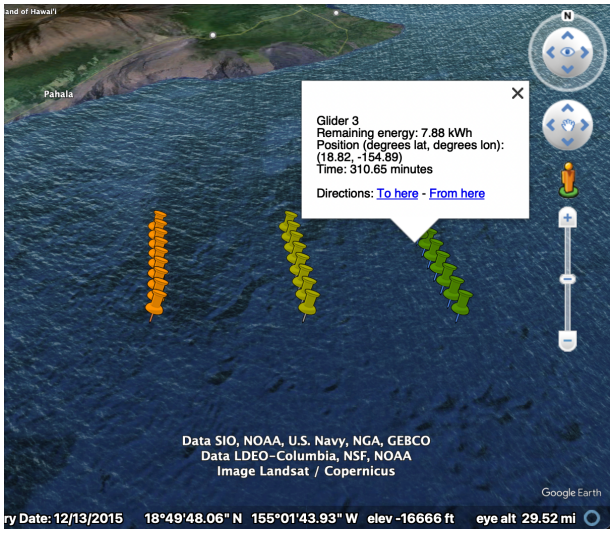

Figure 5: Contents of a kml file produced during a simulated flight off the coast of Hawaii

5 | CFD Analysis

Using data from computational fluid dynamic (CFD) software (Table 2), the simulation's velocity calculation was able to be further improved. Graphs of the lift and drag force in terms of alpha are shown in Figures 18 and 19. Instead of arbitrary coefficients for lift, profile drag, and induced drag, we were able to find equations of the coefficients in terms of alpha. This led us to shift our equations of glider motion to

$$F_{lift} = \frac{1}{2}\rho v^2 C_l(\alpha) = Bcos(\xi) \quad (7)$$

$$F_{drag} = \frac{1}{2}\rho v^2 C_d(\alpha) = Bsin(\xi), \quad (8)$$

folding all of the effects of the angle of attack into the coefficients C_l and C_d.

The fits resulting from the CFD data are shown in Figure 6. As suggested by Graver et. al (2003) [5], C_d demonstrated a quadratic trend in α and C_l v.s. α was linear.

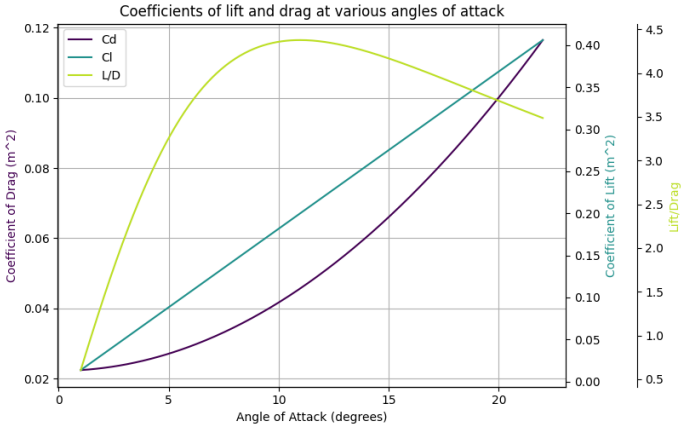

Figure 6: A graph of the coefficients of lift and drag plus the lift over drag ration versus the angle of attack

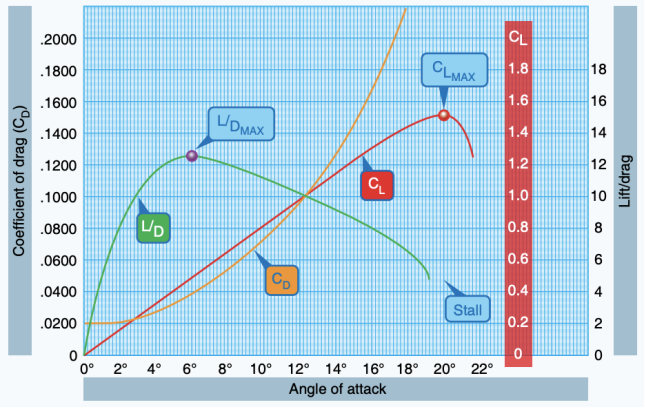

Figure 7: A graph of the coefficients of lift and drag as well as the lift over drag ratio for air-based flight [3]

The quantity graphed in lime green in Figure 6 - lift over drag ratio - is another important quantity in aerodynamics and subsequently hydrodynamics. According to the *Pilot's Handbook of Aeronautical Knowledge* (2023), "aircraft with higher L/D ratios are more efficient than those with lower L/D ratios" [3]. Figure 7, sourced from this same handbook, shows a graph of these quantities for air-based flight that closely mirrors our own. We are only missing the stall region, which is likely due to the maximum angle of attack in our data being 20 degrees.

Since our graph matched the real-world analogue in the region that our data covered, we were satisfied that the CFD data was a good representation of glider flight. Additionally, our graph of glideslope and pitch angle (Fig 20) followed a similar trend as that in Graver et al. (2003) [5] with the only marked difference being an arbitrary sign convention. This further increased confidence in the model.

This led to further analysis. We took our fitted equations of C_l and C_d alongside equations 7 and 8 to produce a graph of the horizontal and vertical components of velocity for a given pitch angle. The method was similar as that detailed in the previous section. In terms of the total through-water velocity v and the glideslope ξ, the horizontal and vertical components are given by $vcos(\xi)$ and $vsin(\xi)$ respectively. This produces a glide polar, which is shown in Figure 8.

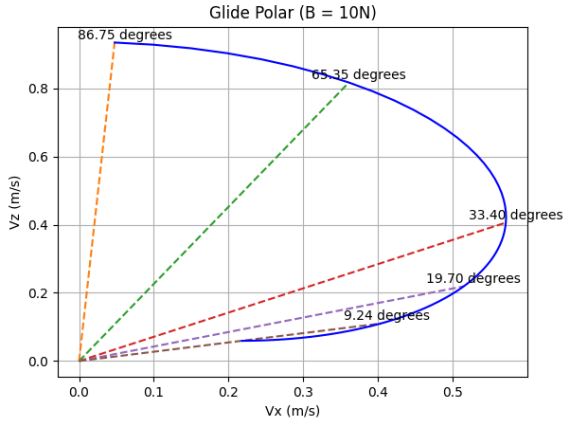

Figure 8: A graph of vertical and horizontal velocities at given pitch angles known as a "glide polar". This polar was produced at a buoyancy of 10 N and a density of 1025 $\frac{kg}{m^3}$ using the curve fits from the CFD data

The dashed lines represent the pitch angle at which a certain horizontal and vertical velocity occur. From the graph, one can ascertain that the maximum horizontal velocity occurs at a pitch angle of about 33 degrees, whereas vertical velocity is greatest when the glider is pitched essentially straight down.

We were able to incorporate the information from this glide polar into the greater simulation. When the glider objects are initiated, the curve fit coefficients for C_l and C_d are read in from the configuration file. These are then entered as arguments into a function that returns the components of the horizontal and vertical velocities that do not depend on density and buoyancy as well as the pitch angle at which each occurs. This is possible since the glider lift equation with velocity isolated can be broken down into

$$v = \sqrt{\frac{2Bcos(\xi)}{\rho C_l}} = \sqrt{\frac{2B}{\rho}}\sqrt{\frac{cos(\xi)}{C_l}}. \tag{9}$$

Two interpolator objects are defined to give the

$\sqrt{\frac{cos(\xi)}{C_l}}$ term for the current pitch angle multiplied by $cos(\xi)$ and $sin(\xi)$ for the horizontal and vertical components respectively. When the velocity is being calculated during runtime, this is then multiplied by $\sqrt{\frac{2B}{\rho}}$, with the buoyancy and density being calculated in real time. This further increased the accuracy of our velocity calculation.

6 | Hover Mechanism

Achieving a stable hover state underwater is another objective of glider flight. The goal of this state is to enable the glider to sit at a certain depth indefinitely with minimal energy required to maintain it. We modified our simulation to replicate this functionality and find the best method to achieve this state.

The first rule implemented was quite rudimentary. Given an intended hover depth, we pumped oil in when above and pumped oil out when below that depth. This was, as always, bounded by the maximum and minimum buoyancy states specified in the configuration file. This produced a very inefficient mechanism due to oscillatory behavior around the hover depth (Fig 9). Since it is highly unlikely the glider will reach equilibrium at the specified depth, its pitch and buoyancy continually change, albeit at smaller increments, over time.

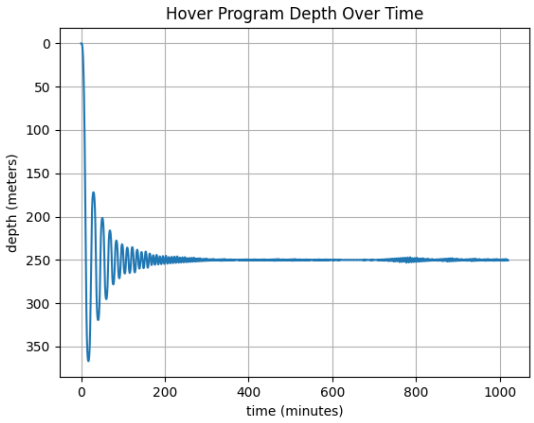

Figure 9: Oscillatory depth over time achieved by initial hover strategies

Additionally, this introduces a risk for hovers below 50 meters because increased pressure creates the potential to lock the oil valve in the open position if letting oil in below this depth. In the worst case scenario, this could rupture the internal reservoir and cause the entire glider to explode.

To mitigate this risk and improve the stability of the hover state, we tested many more methods. For instance, a "dead band" was added as a parameter in the configuration file. If the glider is within this range above or below the hover depth, the buoyancy state is not altered. This allowed the glider to reach a vertical equilibrium state at which there is no oscillation. To achieve this state more rapidly and avoid overshoot, a slowing depth was instated at which the glider begins pumping out to increase its buoyancy a certain distance above the hover depth.

In our next attempt, we worked from the hypothesis that each depth has a corresponding neutral density which the glider could achieve if it knew the correct buoyancy state. In our code, we manipulated the variables and functions so that the glider would take a density reading - an allowed functionality - once it reached the hover depth. By rearranging equation (2), we can solve for the volume at which a neutral hover state will be achieved at that depth. The buoyancy engine is then set to that state permanently and the glider is enabled to drift to and sit at that depth. This appears to be sustainable indefinitely, as the glider maintained that depth until being pushed out of the bounds of the simulation set by the HYCOM and GEBCO files downloaded.

One attempt of this method at a hover depth of 300 meters is depicted in Figure 10. This new type of graph with four subplots of important quantities was used as one means to judge the effectiveness of each method. It appears that setting the buoyancy state and letting glider come to vertical equilibrium makes for a very efficient energy usage. Once it has reached the hover state, the only energy consumption comes from the background hotel load. This is also a safer technique since it does not require oil to be pumped in below the surface at all.

Following this initial experimentation, we were able to tackle the more pressing issue of maintaining such a hover state within a limited bounding region. For our purposes, we defined this area as a square with side lengths of 2000 meters.

The stable buoyancy state method works well to achieve a hover state, but the glider is quickly pushed out of bounds by the current. This can be visualized in Figure 21, which shows how the glider gets down to the depth and then is entirely at the whims of the current.

As such, we slightly modified the equilibrium state method so that the glider would be able to stay in the box for a longer period of time. It begins in the same manner by gliding quickly down to the target depth and taking a density reading. It then glides at that

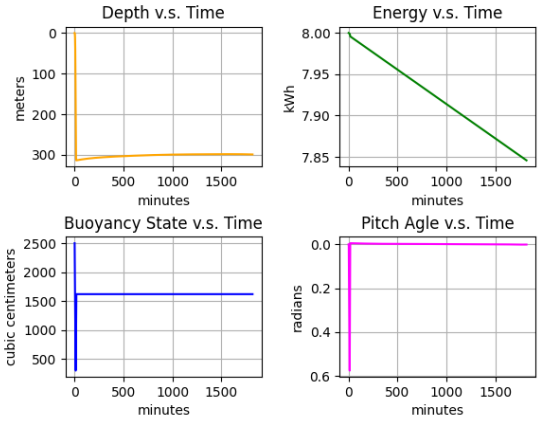

Figure 10: Visualization of four important glider quantities over time during a hover simulation when setting the buoyancy state and letting the vehicle reach vertical equilibrium

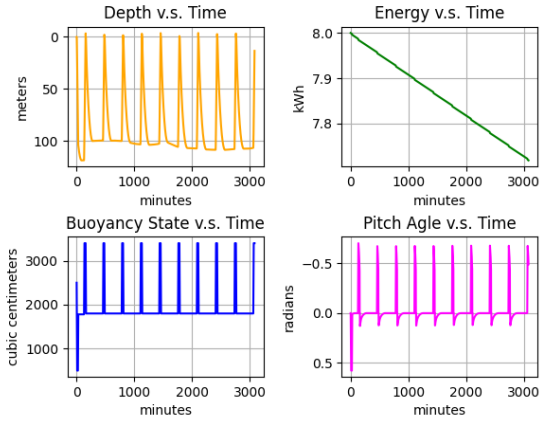

Figure 11: Visualization of four important glider quantities over time during the best hover strategy within bound to-date

depth for a set time labeled as "hoverLimit" in the configuration file. The vehicle then ascends in the opposite direction. The difference between its dive point and surface point should give an approximation of the current direction and magnitude at the hover depth. To make this possible, the "hoverLimit" should be decently greater than the amount of time it takes to descend and ascend. This minimizes the influence of the current experienced on the way down in the difference calculation.

Once the test dive is complete, the glider now has an estimate of the current and the ideal buoyancy state at that depth. During future dives it is flown in the direction opposing the current and sets its buoyancy state in advance so that it effortlessly glides to the goal depth. Since there is no propulsion aside from currents when hovering, the glider must fly upward and then back down periodically to oppose the currents. Another quantity termed "hoverIterationTime" was defined as the amount of time the vehicle will hover during each dive before re-orienting.

This tactic resulted in the longest hover attempt yet, lasting a little over **2 days**. This is much better than the first attempt we made with our very first approach that went out of bounds after 40 minutes. Figure 11 depicts the state of the depth, energy, buoyancy state, and pitch angle over time during this attempt. Additionally, figure 12 shows the top-down view of glider position overtime with the red x denoting the starting position. As one can see, what actually pushed the vehicle out of the box in this case was the glider propulsion and not the current. As such, it appears that an ideal strategy would be to find conditions in which the glider propulsion and currents cancel each other out.

As such, there is still much to be done to craft an optimal a hover strategy. For instance, one could play with different values of "hoverLimit" and "hoverIterationTime" to find a combination that achieves a balanced hover. Additionally, one could estimate the current after every dive to maintain a more accurate magnitude and direction over time. Our brief impression of hover strategies barely scratched the surface of this deep and involved topic.

7 | Real World Data

To further inform the simulation's function, we performed analysis on data from actual glider dives. This comes in the form of a so-called "crash log" that contains the data shown in Figure 13. Each file of this kind includes information on depth, pitch, roll, VBD, and heading that is all superimposed onto one busy plot. This demonstrates the myriad variables and their complicated variations that come into play when flying one of these vehicles

From this, we attempted to look into many similar quantities as the CFD analysis. However, this presented various complications. First, the only way to track glider horizontal displacement is data on the diagonal distance between the glider and the ship. This is based on a system similar to sonar in which a pulse of sound is timed as it travels to the glider and back to the ship. This is highly variable since the ship itself is pushed around by the currents (Fig 22).

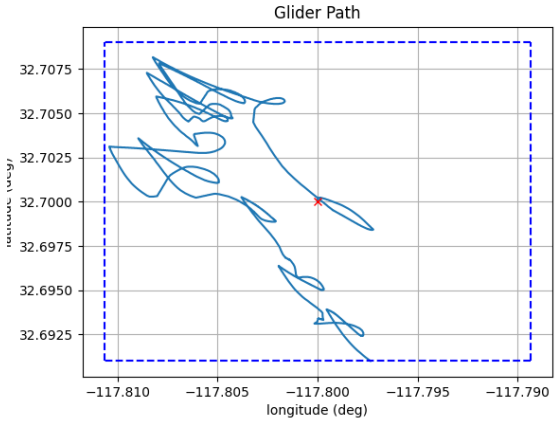

Figure 12: Display of glider position over time with the starting point in red and the bounding box in dashed blue

Additionally, it is difficult to ascertain the direction of the glider, especially since the heading measurements oscillate so much.

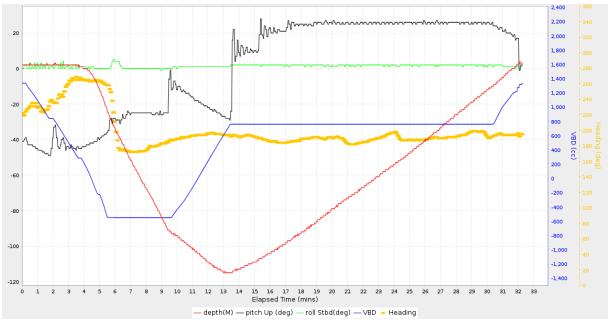

Figure 13: Busy plot of crash log data from a glider flown on Feb 26th in the Puget Sound

This deficiency was actually useful when determining the depth averaged current (DAC) experienced by a glider during flight. The heading is set by a glider as it flies based on where it is currently pointed and where its destination is related to that. It then changes its heading to point toward the destination. As such, we took information the glider's waypoint (destination) and changes in heading to estimate the glider's position over time without currents. This was then used to calculate the average current experienced over each dive based on the difference between the estimated and actual final positions.

The result of this analysis is depicted in Figure 14, the data for which was gathered during a flight from March 25th to 28th, 2024 off the coast of Southern California. The figure has green arrows representing the estimated DAC superimposed on the path of the glider. The glider is underwater for the blue portions and idling on the surface at the red segments of the graph. From our calculations, it appears there was a southeast current of approximately 0.1 m/s magnitude in this region at this time.

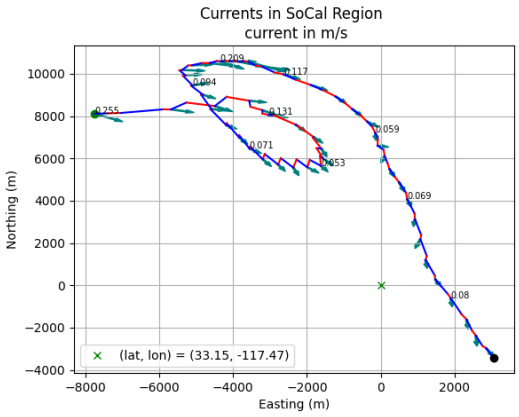

Figure 14: Current estimates represented by the green arrows for a glider flight that took place from March 25th to 28th, 2024 in the SoCal region with glider path superimposed (blue = underwater)

The current in the Socal region during this same time period was also estimated using the HYCOM model. However, this dataset proposes the exact opposite result with the currents travelling in the northwest direction (Fig 23). This illustrates just how difficult any sort of current speculation can be and how little is known about inner ocean mechanisms.

We then analyzed the current estimates from the crash log data further. Looking at the DAC experienced by the glider for different dive depths, we observed an inverse trend (Fig 15). The average depth as opposed to the maximum was chosen as a metric because it takes into account the fact that gliders are experiencing the effect of currents throughout the dive, not just at the apogee. Since we only have data on current averaged over depth, this consideration is necessary and more accurate overall. The trend also appeared slightly stronger in this case. This result aligns with the known phenomenon that currents are strongest at the surface since larger DACs were generally observed at lower average depths.

Other than this information on the currents, we came to few conclusive results due to the many difficulties with the data available as mentioned. More analysis on all of these effects is instrumental to gain a deeper understanding of gliders and how they fly. This

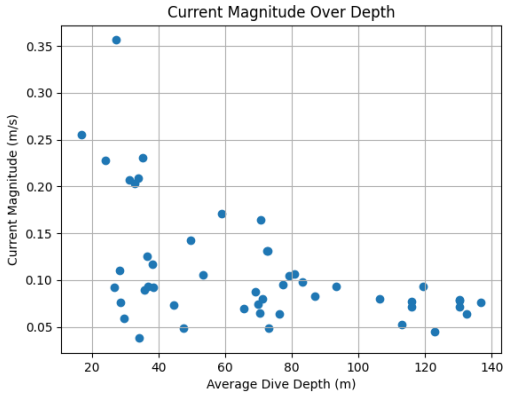

Figure 15: Estimated DAC magnitude versus average depth of dive from March 25th-28th SoCal data

is one way in which the simulation can be useful because it mimics many of these complex variables and mechanisms seen in the real world.

8 | Simulating Glider Transit

In preparation for an upcoming glider flight, we were asked to run our simulation to predict the optimal apogee depth. Shallower apogee depths allow the vehicle to surface and report information back more often. That, however, leaves them to the whims of the stronger surface currents. With deeper dives, the stronger currents are avoided at the expense of increased times between surfacings. Our goal was to find the apogee depth that best balances these concerns.

During this mission, the glider will fly from Oceanside, California to San Clemente Island in the waters off of the Southern California coast. Since this flight was initially planned for late June, the HYCOM files from June 2023 were used to obtain current, salinity, and temperature estimates. Although the time of the trip was pushed back, the use of this assortment of files should not vastly impact the outcome.

Keeping all other values constant, we ran the simulation at apogee depths of 100 meters through 900 meters in increments of 100 meters. A top-down view of the resulting nine flights is shown in Figure 16. It appears that deeper dives followed more direct paths to the destination, which makes sense due to the greater magnitude of surface currents. Figure 24 shows this figure overlaid over the actual global position on Google Earth for reference.

The manipulation of apogee depth also impacted

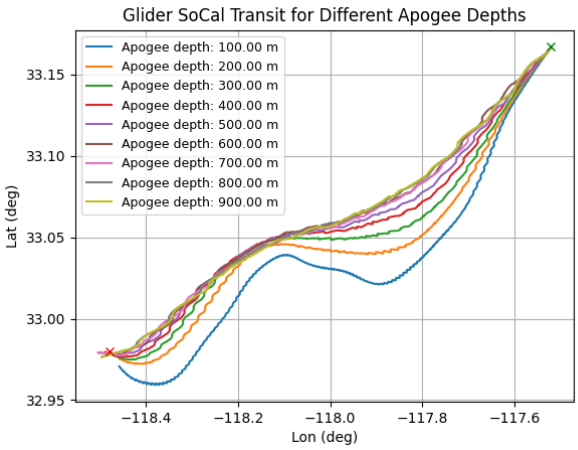

Figure 16: Top-down view of the transit from Oceanside, CA (green x) to San Clemente Island (red x) for nine different apogee depths

many other important variables as shown in Figure 17. Average dive time increased linearly with depth since the glider had to travel a greater vertical distance with each increasing apogee depth. However, this does not seem to have affected efficiency because total energy and total transit time both decreased with depth. Figure 25 further breaks down the energy usage into propulsion and hotel load, showing both decreased similarly in an exponential-esque pattern.

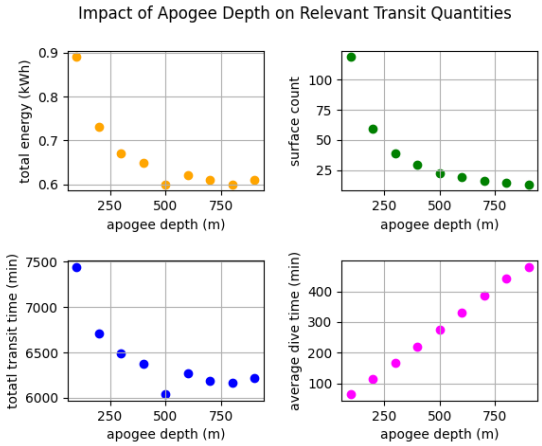

Figure 17: Graphs showing the impact of apogee depth on four important transit quantities

As expected, the surface count also decreased with depth. One the one hand, it allowed deeper dives to take more direct paths. On the other hand, deeper dives often didn't surface close enough to the endpoint,

meaning they overshot and had to glide back somewhat inefficiently.

We are still unsure the cause of the outlier at 500 meters for both total energy and transit time. One possible explanation is that gliders flying to lower depths took much longer to fall within an acceptable radius to the endpoint when flying below 500 meters. This is a question that requires further analysis.

With all these variables in consideration, we suggest that the mission this summer be flown at a maximum depth of **500 meters**. A vehicle flown with this apogee is able to avoid strong currents while not losing efficiency with unnecessary vertical displacement. All analysis showed that benefits either even out or are optimal point at this point.

9 | Conclusion

This project explored real-world data and used mathematical analysis to develop a simulation that replicates true glider flight. Using a configuration file, it has been made scalable to any location and time as well as all combinations of parameters. It has become a helpful tool to analyze and optimize these parameters for a given glide or hover manuever.

Still, the simulation can always be made more accurate. Fluid dynamics is such a complex subject, and there are so many variables like roll and yaw that have not yet been incorporated into our code. Additionally, the a mechanism for determining the pitch angle separate from the buoyancy would make it even closer to actual conditions.

That being said, this iteration is a very good approximation to true conditions as it stands. We hope that this application can be applied to many upcoming flights and hovers as it was for the Oceanside to San Clemente transit. It is our goal that this technology will aid gliders in their important work of ocean profiling. Through this, oceanographers and others will be able to increase understanding of the ocean environment and our world as a whole.

Acknowledgements

I would like to thank Peter Brodsky for being an amazing mentor throughout this project as well as the Office of Naval Research (ONR) for providing funding for this work. Also, a huge thank you to Lucille Shield for allowing me to build off of her simulation for this project.

References

[1] Center for Ocean-Atmospheric Prediction Studies. (2024). HYbrid Coordinate Ocean Model. HYCOM.

[2] Eichhorn, M., Aragon, D., Shardt, Y. A. W., & Roarty, H. (2020). Modeling for the performance of navigation, control and data post-processing of underwater gliders. Applied Ocean Research, 101. https://doi.org/10.1016/j.apor.2020.102191

[3] Federal Aviation Administration. (2023). Pilot's Handbook of Aeronautical Knowledge. U.S. Dept. of Transportation.

[4] GEBCO Compilation Group. (2023). GEBCO 2023 Grid. https://doi.org/10.5285/f98b053b-0cbc-6c23-e053-6c86abc0af7b

[5] Graver, J., Bachmayer, R., Leonard, N.E., Fratantoni, D.M., & Hole, W. (2003). Underwater glider model parameter identification.

10 | Appendix

Tables

Pressure (psi)	Volumetric Flow Rate (cc/s)	Motor Power (Watts)
0	4.45	14.3
101	4.49	18.7
203	4.37	22.4
303	4.29	25.9
402	4.41	29.8
503	4.22	33.1
604	4.06	37.0
709	4.14	40.3
806	4.1	43.7
910	3.98	48.0
1008	3.94	51.6
1098	4.06	54.7
1211	3.91	58.8
1316	3.79	62.4
1409	3.94	66.2
1515	4.52	68.4
1614	3.67	73.4

Table 1: Carina flow rate and motor power data

Angle of Attack (degrees)	Vehicle Lift (Newtons)	Vehicle Drag (Newtons)
0	-0.007	7.544
2	7.968	7.768
5	19.718	8.853
10	74.870	13.107
20	115.882	32.976

Table 2: Data produced by computational fluid dynamic (CFD) software giving the relationship of the angle of attack with lift and drag force. The above numbers assume a through-water speed of 0.8 m/s

Figures

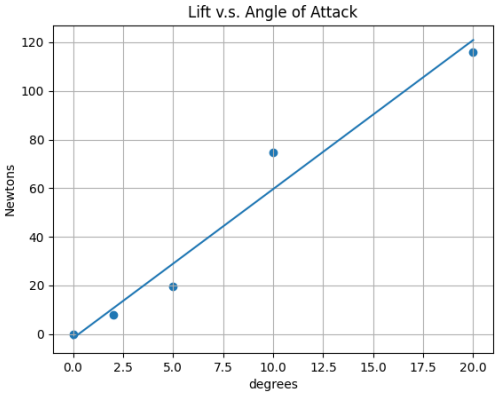

Figure 18: A linear curve was fit to the CFD data of lift force at a given angle of attack: $Lift(\alpha) = -1.679 + 351.246\alpha$

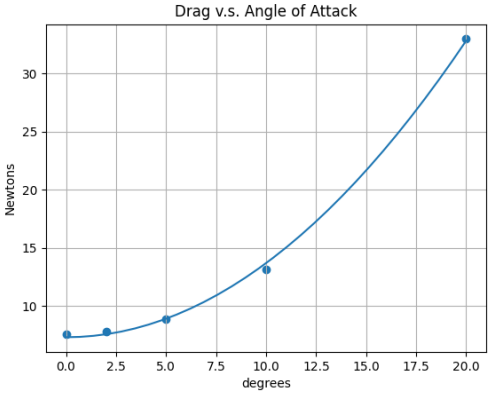

Figure 19: A quadratic curve was fit to the CFD data of drag force at a given angle of attack: $Drag(\alpha) = 7.295 + 209.589\alpha^2$

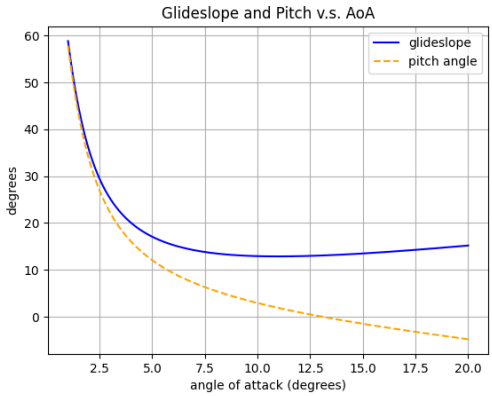

Figure 20: A graph of glideslope and pitch angle at various angles of attack based on the CFD data

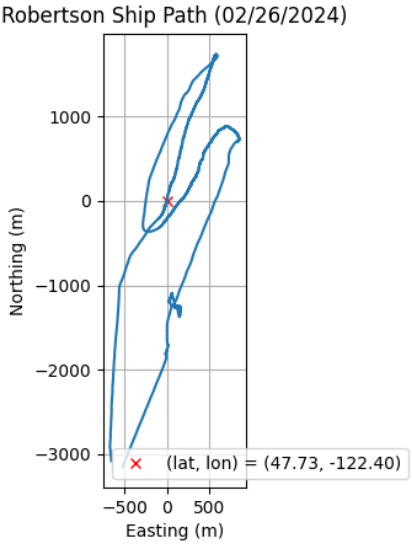

Figure 22: A graph of the Robertson ship path in the Puget Sound while being pushed around by currents during a February 2024 glider mission

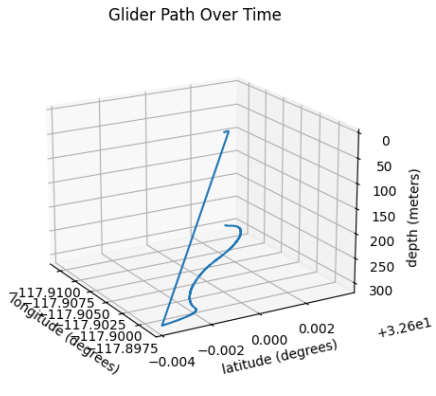

Figure 21: A 3D visualization of the vertical equilibrium hover strategy

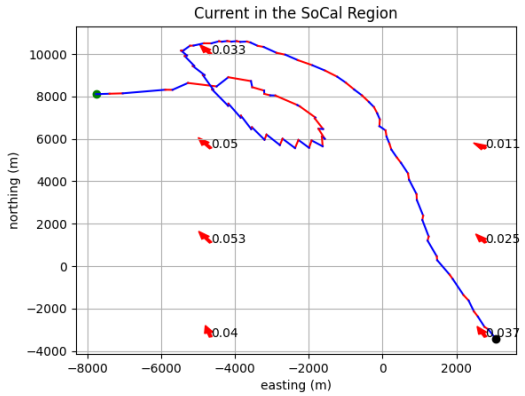

Figure 23: Currents suggested by the HYCOM model in the SoCal region from March 25th-28th, 2024 with real glider path superimposed

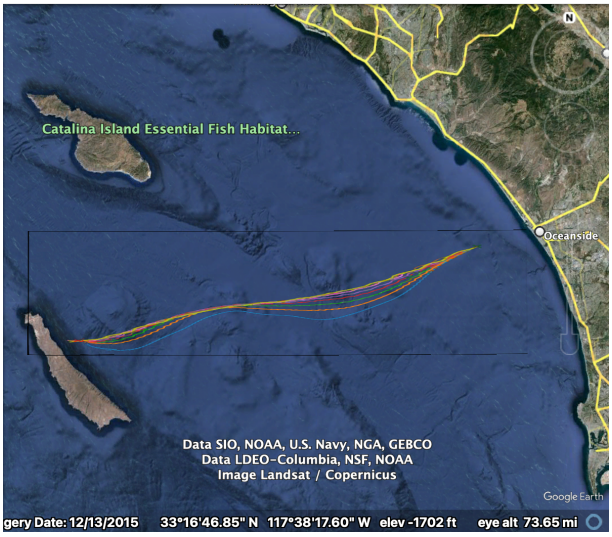

Figure 24: The simulated transit from Oceanside, CA to San Clemente Island at various apogee depths overlaid onto Google Earth

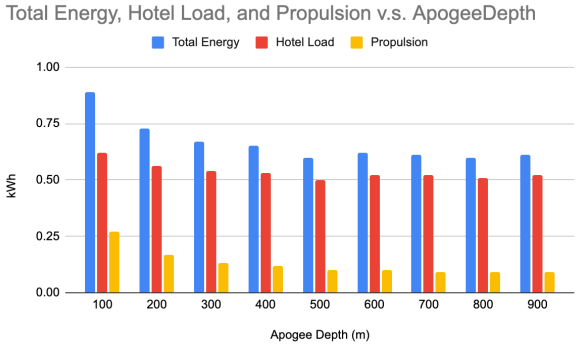

Figure 25: A breakdown of energy usage per apogee depth into propulsion and hotel load