Benchtop Depth Simulator Design for the Testing of Coastal Profiling Floats

Pierre Baudin[1,2] **and Gene Massion**[1]

[1]Monterey Bay Aquarium Research Institute, Internship Program, Moss Landing, 95039, USA
[2]University of California, Santa Cruz, 95064, USA

ABSTRACT

A Coastal Profiling Float (or CPF) is an autonomous system that, when deployed, ascends and descends the water column measuring and logging scientific data at specific depths. It is desirable to be able to robustly test the behavior of these systems in a controlled lab environment without the need of a 500 meter seawater tank. A variable pressurized air source provides an effective way to simulate the pressure associated with changes in depth and the rate of pressure change associated with specific velocities.

Introduction

The world's oceans currently contain thousands of profiling floats gathering data to help us understand the changing temperature of the globe. Having a distributed network of autonomous sensing systems allows scientists to zoom out and look at effects on a global system level. MBARI Scientists and Engineers with the Chemical Sensors Lab have been designing profiling floats for use specifically in coastal waters. Using a distributed network of coastal floats, scientists can collect data regarding the health of ecosystems to better inform public policy regarding commercial fishing and other impactful human activities.
During profiling missions, it is possible that CPFs will encounter situations not expected by the engineer who programmed the system. Getting caught on kelp or hitting a surprisingly steep sea floor would be a couple of examples. It is important to be able to imagine and test these situations in the lab. Previously, lab testing was done using a hand pump on the CPF's pressure transducer. This is an imprecise method that lacks consistent repeatability when trying to simulate specific conditions. By using a computer controlled air pressure system, we can test system behavior in a more robust and repeatable manner.

Methods

In order to control the levels seen by the CPF's pressure transducer, a current controlled proportional valve connected to a pressurized air source was used. The current used to command the valve position was generated by a National Instruments NI 9265 Digital to Analog converter. The first step in creating this simulator was to be able to command specific pressure values from software. Determining the relationship between commanded current and pressure output is required. By plotting current and pressure at seven test points it was determined that the response is very linear.

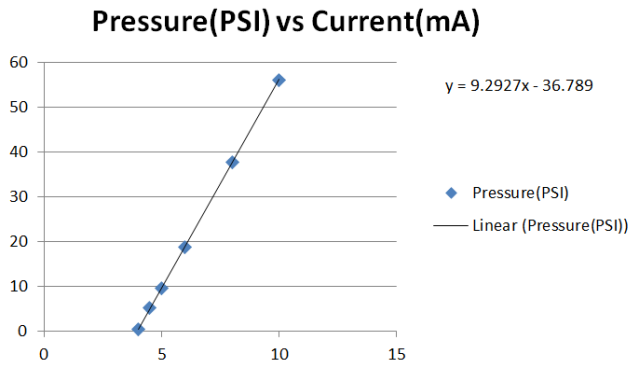

Figure 1. Pressure Response

One decibar of pressure is approximately equivalent to one meter of seawater. A simple unit conversion of PSI to decibar gives a known relationship between commanded current and the resulting simulated depth. Using this relationship, a simple pressure command application was created in the C# programming language.

During a mission the CPF generates two logs, one records the scientific data and the other records the internal engineering data. In the lab, the engineering data log can be accessed as it's written over a bluetooth link. Additionally, engineering data logs from previous missions are recorded and saved at MBARI. This means that the design and testing of a simulator can be done using pre-existing data and does not require the presence of a float in the lab. In order to replicate the data link received by a live CPF, a simple program was created that reads lines from a datalog and transmits them over bluetooth at the same interval done by the CPF. This has the added benefit of being able to select small

specific sections of the log that would be helpful in testing the simulator. Testing from existing logs and testing with the live float is exactly the same when viewed from the perspective of the simulator software.

A previously existing piece of software built by MBARI engineers titled ”CPF Console Monitor” was used as the skeleton for the simulator software. the Console Monitor parses the data received over the bluetooth link and displays this information in a window. Pressure command functionality was added in and the simulator was built from this base. Two different approaches were used to implement the desired simulation behavior: Open Loop Direct Command, in which the user designates what the pressure change should be, and Automated Command, in which the the system generates a pressure value based directly on the output of the CPFs control system.

Open Loop Control

This level of functionality is a necessary design stepping stone towards fully automated behavior. Being able to command a specified pressure is a basic need of the system. This is also true of a constantly changing pressure ramp. A constant pressure ramp is a continuously updating structure, the design of which also comes into play when building more complex automated behaviors. In the process of building this functionality, a basic automated simulator was created. This simulator read the state of the system and triggered the constant pressure change equating to the desired velocity of the system. This automated functionality was not left in the final product. This is because there is no benefit to this simpler automated operation when compared to the more advanced automated control implemented later. The Open Loop command functionality left in the final product boils down to two simple options: the user can either enter a specific commanded depth or trigger a pressure ramp equivalent to some constant velocity. The control interfaces for these commands can be seen in figure 2

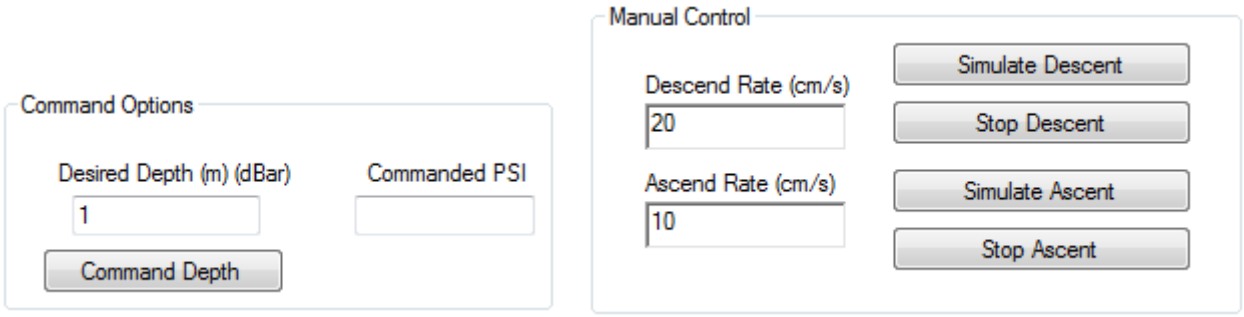

Figure 2. Manual Control Parameters

Automated Control Using Dynamic Model

A CPF uses a buoyancy engine to move up and down in the water column. This is implemented with a hydraulic pump that moves fluid between bellows inside the body of the float and a pair of external bellows. This changes the amount of seawater displaced by the system which in turn changes the buoyant force acting on the float. The amount of seawater displaced by the system can be calculated if we know the turning rate of the hydraulic pump and the amount of fluid displaced per turn. The turning rate of the pump is a factor controlled by a PID system in the float’s software and its value is transmitted over the Bluetooth link. This means that we can create a dynamic model that tells us the forces on the system through the course of a profiling mission. From force and mass can determine acceleration with the simple relationship given by Newton’s second law. Knowing acceleration, we can integrate to get velocity. Knowing velocity we can integrate once more to get position from which we can determine the pressure that must be fed to the system.

A Dynamic Model of the system was created by Gene Massion and intern Laughlin Barker during the internship program in 2014. It was used to design the control system that runs on the CPF. Contained within the model are the Dynamics of interest to this simulator and those have been isolated and presented in figure 3. This simplified selection of the model shows only the dominant change in buoyant force and drag. Using this model the proper output pressure from the simulator system can be quickly calculated.

This dynamic model driven behavior is only functional during the parts of the CPF mission in which the control system is active. Otherwise the data relating to turn rate of the pump is not transmitted. This means that in order to simulate a full profile there are a few things that need to be commanded directly. Sea floor depth is one factor, this represents the pressure value at which the system stops increasing the command. This triggers the CPF’s entry into the ”Anchor” state. The CPF

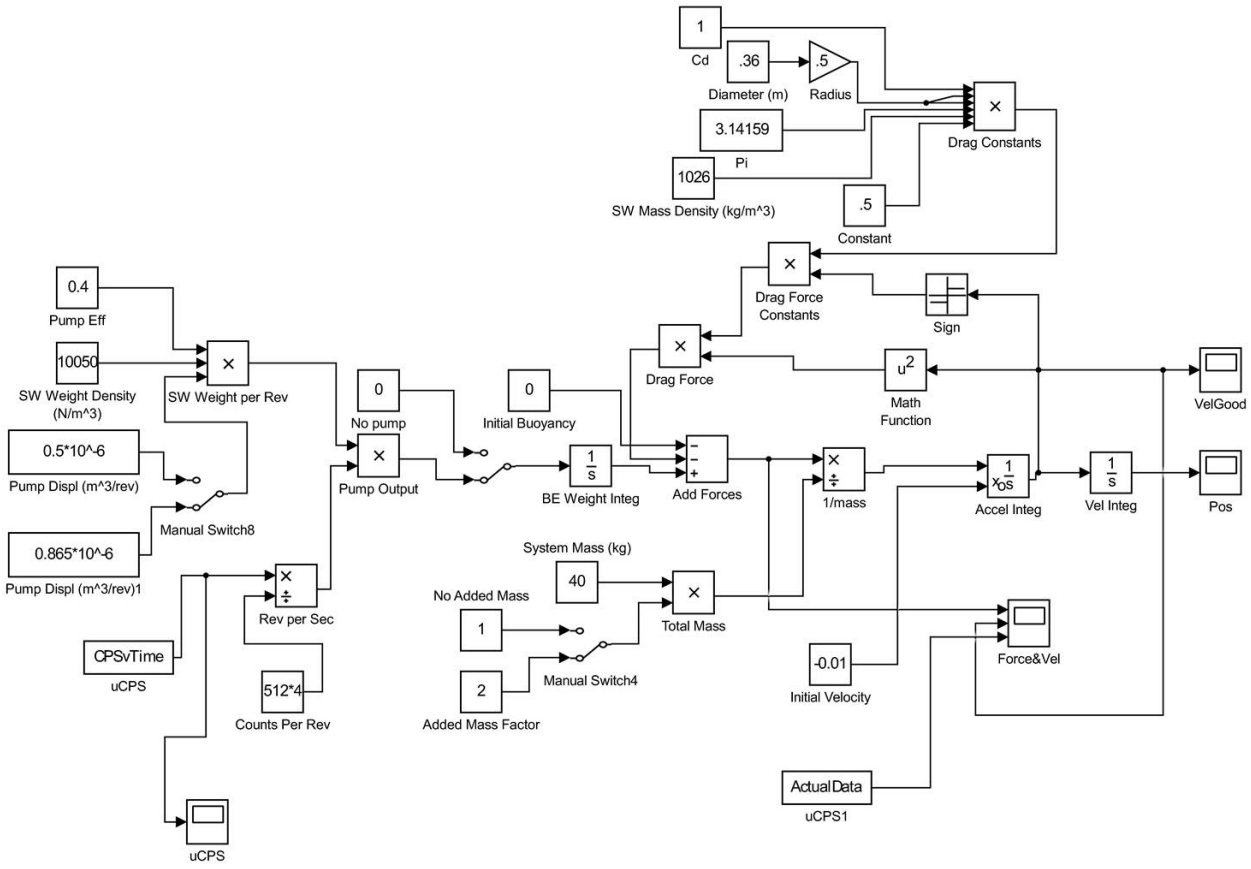

Figure 3. Dynamic Model

will then enter the ”DeAnchor” state which changes the buoyancy until the system begins floating fast enough to trigger the ”Ascent” state which starts up the control system once more.

Live Data Visualization

It is helpful, for the purposes of this simulator, to be able to see how elements of the system change from moment to moment. Using an open source package called LiveCharts Geared, I was able to implement efficient and responsive data visualization options. This includes dynamic visibility of the values of interest and the ability to zoom in on target areas. The visualization window can be seen in figure 4.

Results

Dynamic Model Verification

The resulting software application is able to trigger multiple consecutive profile simulations on the real CPF hardware. We also are interested in seeing if the outputs generated by the simulator can replicate previously recorded mission profiles. In order to test this, previous mission logs can be fed into the ”CPF Spoofer” application which transmits them line by line over Bluetooth on the same interval in which they were recorded. An initial run of this test yielded the output shown in figure 5 The blue trace represents the actual velocity of the system during the descent phase of its mission, the red trace is the expected velocity based on the model. This initial test shows a large discrepancy between the two traces. The simulator output does not stabilize around any reference point and at its peak only reaches about halfway to the desired set point. This lead me to believe the model was flawed. There few values in the model that are not either hardware specifications, or

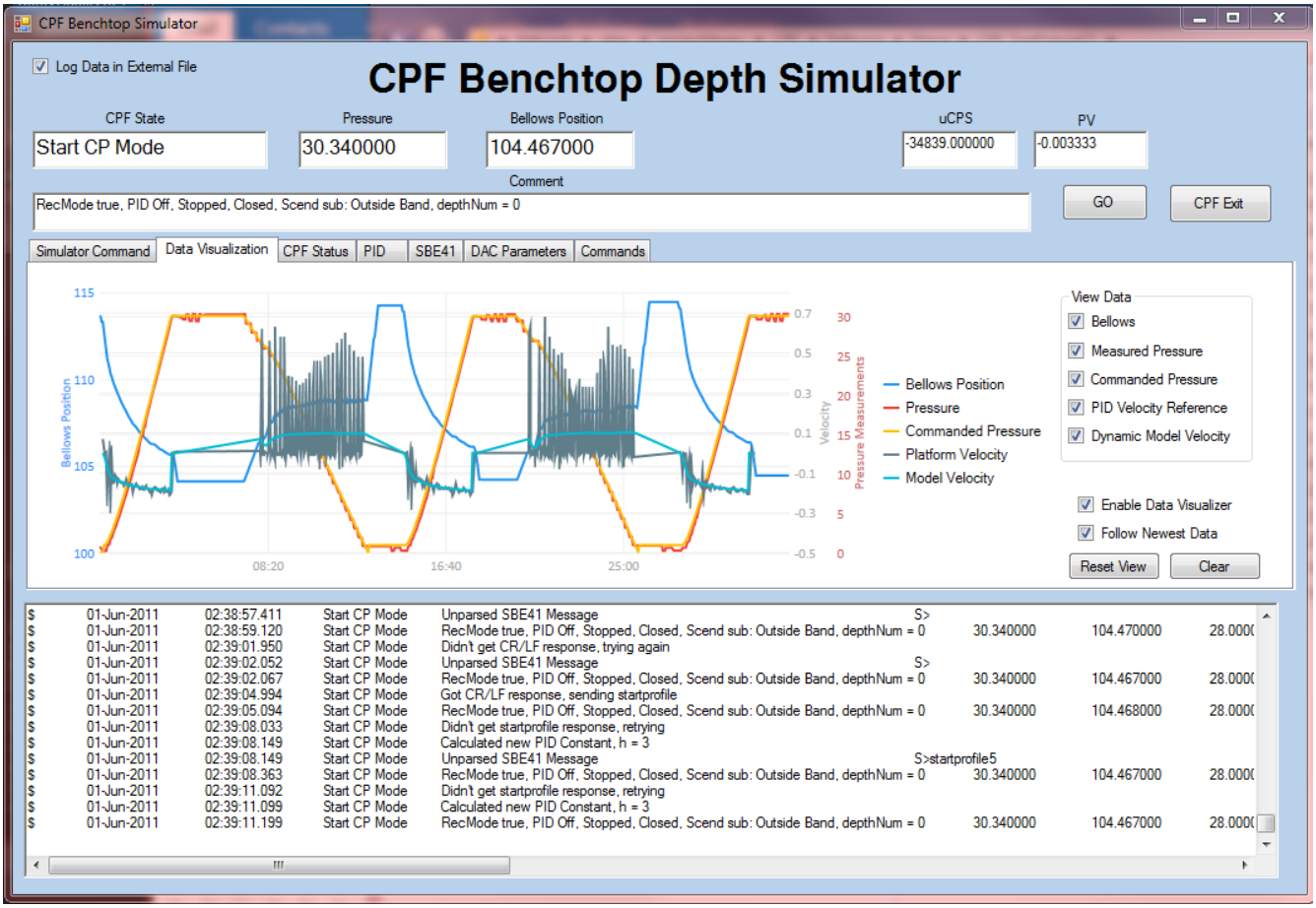

Figure 4. Data Visualization Functionality

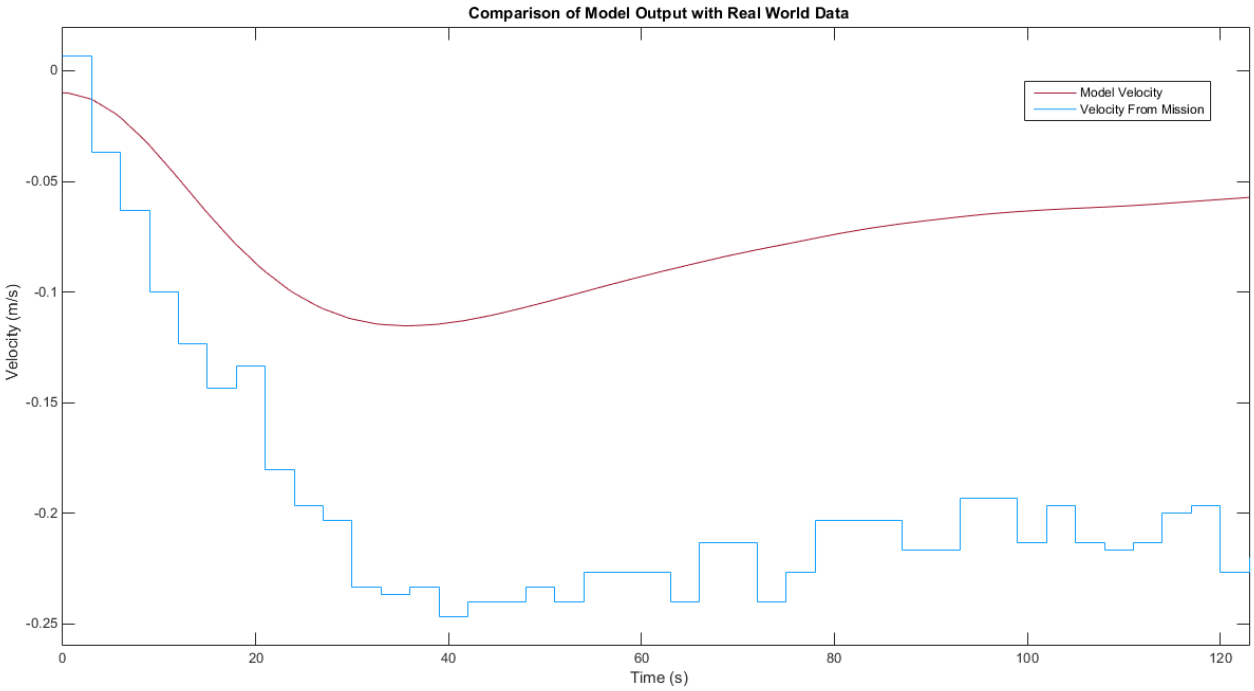

Figure 5. Dynamic Model Verification Test

experimentally determined. The largest value that is a result of estimation is pump efficiency. That being the case, I wanted to know if varying this parameter would bring the model output closer to the real values. the result can be seen in figure 6

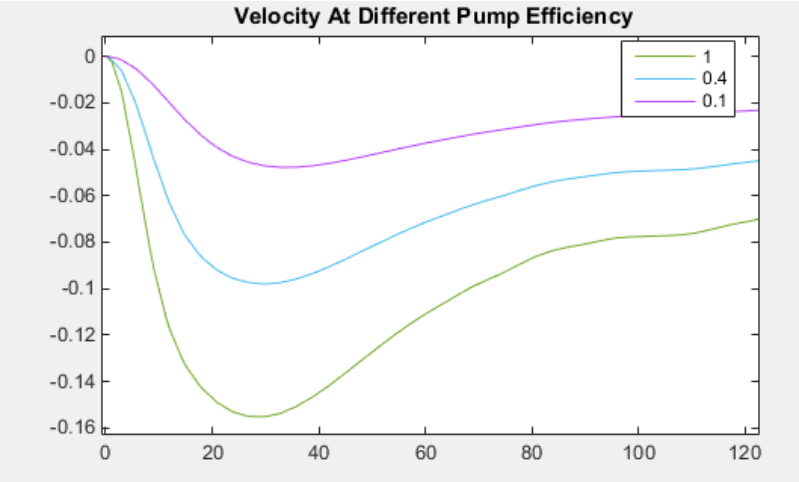

Figure 6. Pump Efficiency Parameter Variation

Varying this parameter does not make the output closer to the known behavior of the CPF. After seeing this, I reexamined my assumptions about the simulation. In the experimental set up used to generate this model output it's assumed that the float starts from neutral buoyancy. This is not the actual case, the float does not enter it's "Descend" state until it is already sinking. By experimenting with a few different initial condition values, I found that an initial downward force of two Newtons gives the a response that stabilizes around the correct set point this can be seen in figure 7. However, the model velocity stabilizes faster than the actual output. The previous experiment showed that this rise time is primarily influenced by the pump efficiency factor.It appears that initial force of 2N with a lower pump efficiency results in the behavior observed in the real system

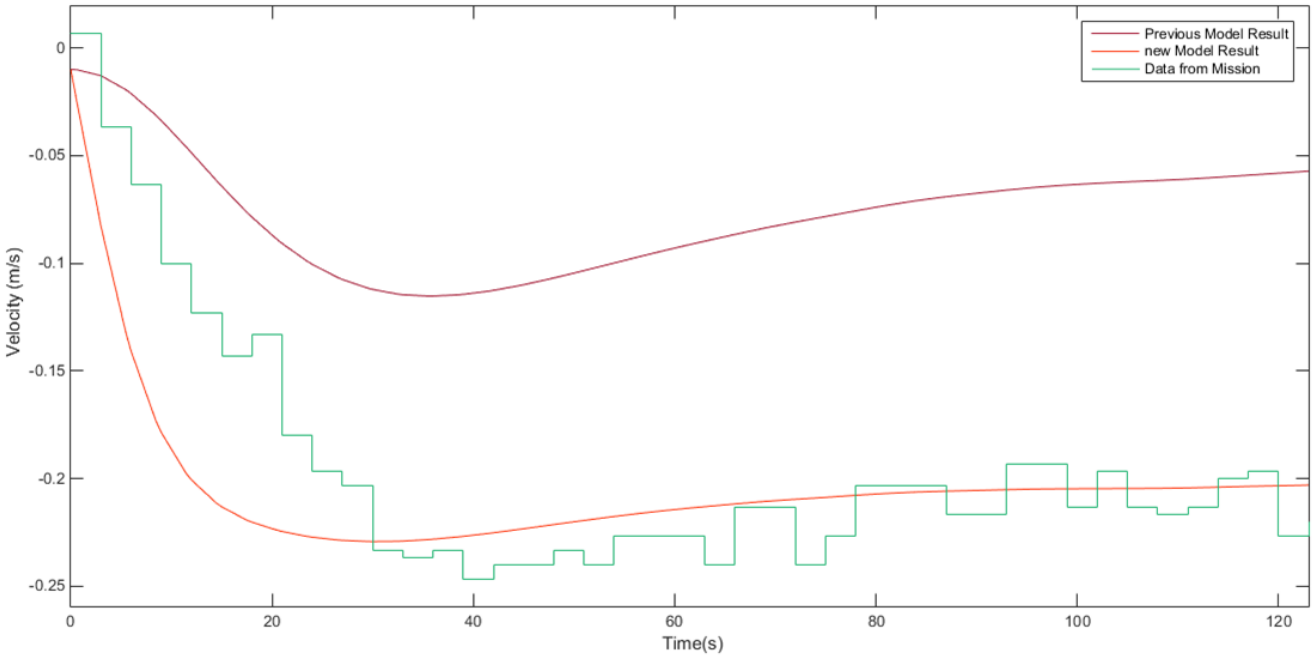

Figure 7. Dynamic Model Verification Test

Hardware Analysis

Clearly visible in the live data view is that the measured platform velocity trace is comprised of a large thin spikes. This can be seen in figure 8 This is only present during the 10CM/s Ascend phase, looking at the pressure, we can see that it appears to be comprised of many flat steps rather than the smooth line we can see during the Descent at 20CM/s. This appears to be an issue with the pressure regulator resolution. With a higher resolution regulator we would see a smoother output that would better represent the changes seen during a typical profiling mission.

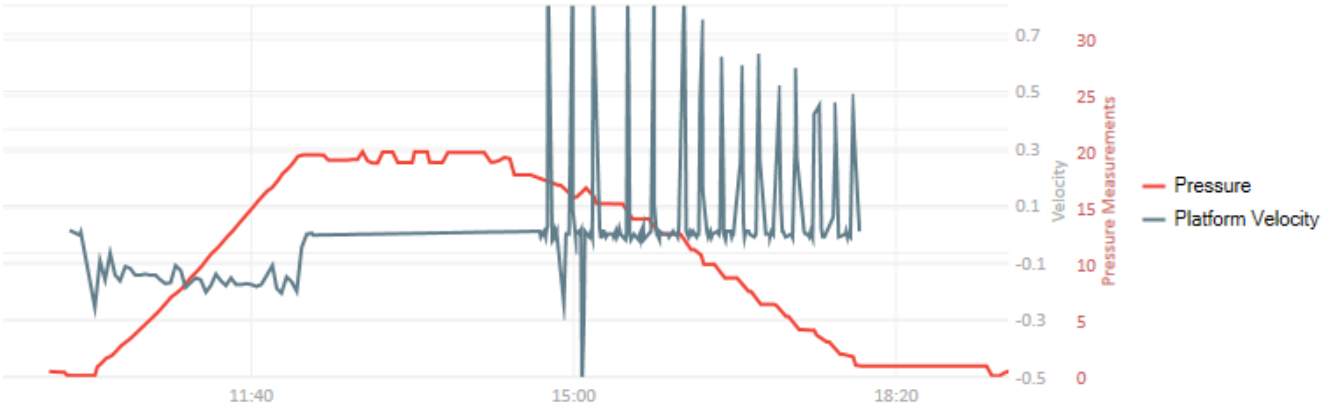

Figure 8. Stepped response on Ascent

Conclusion

The result of this project was a functional testing platform that allows for the examination of system behaviors during profile missions. Through a mixture of the automated functionality and the open loop command options, irregular pressure profiles can be commanded to simulate unexpected forces.

Acknowledgements

Thank you the Packard Foundation and MBARI for creating the Internship Program. Thanks to George Matsumoto and Linda Kuhnz for running the program. And thank you to my project mentor Gene Massion for the many helpful insights and continued guidance with the work done this summer.