# BIOE 420: The Final Project

*Group 81 - The Narcotics Ninjas*

Naraen Palanikumar, Aryan Shah, Alec Wahl

May 2023

## 0.1 Background

Anesthesia can be defined as the use of various medications that prevent pain during surgery; it can be general or localized to a specific part of the body [1]. Surgical technologies tried to create controls and systems that are cloud-based to ensure the monitoring of doses and patient care is ensured. The aim of this project is to create a controller that provides adequate depth of anesthesia. This is important because if it is too light then patients could feel the procedure taking place, and if it is too deep then there may be postoperative complications like vomiting, pain, infection, hypertension, and in some cases even death [2]. This model will be developed using EEG signals of alertness and use that as feedback. According to the UICU alertness scale, the system will be optimized close to 50. Above 60 would indicate too much pain for the subject, while values below 40 would indicate postoperative complications. The sensitivity of patients to the anesthesia drug can also vary and this must be taken into account for a suitable controller.

## 0.2 System Response Analysis

In order to identify the system contained within the virtual patient model, we had to clear two main criteria. The first is to select a very long, random, and varying input to properly test the system response to varied inputs across a realistic scale. The second is to open the loop so that we can take a look at the relationship between its outputs and inputs more directly.

We first examined the response of the system to a step input to get a general idea of the system's behavior. We noticed that the system reached a steady state after around 2000 seconds, which gives us a p-value of 2000 with a time-step of 1 second. This means that our testing input should be at least 3 times this length at 6000 seconds. We went with a 7500-second input in order to have ample outputs and inputs to estimate the transfer function.

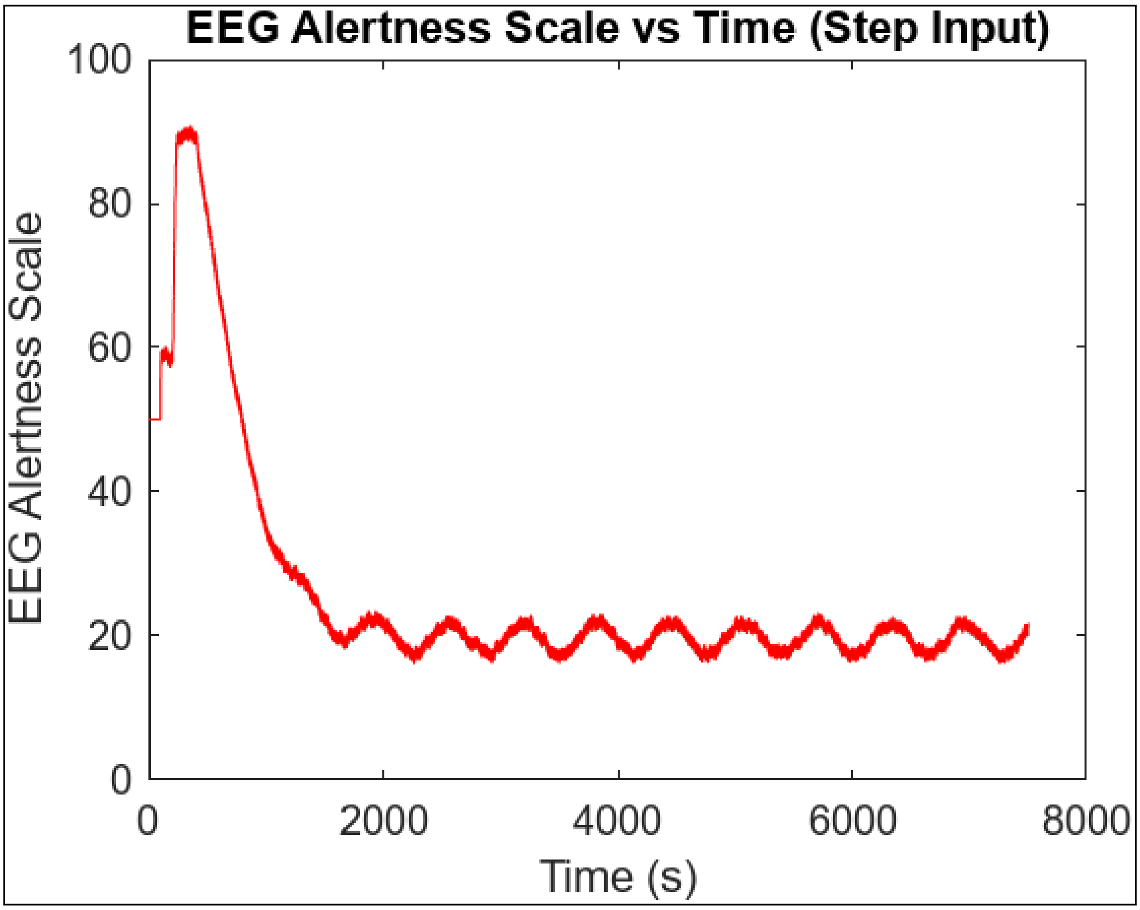


Figure 1: Response of our plant system to a step response.

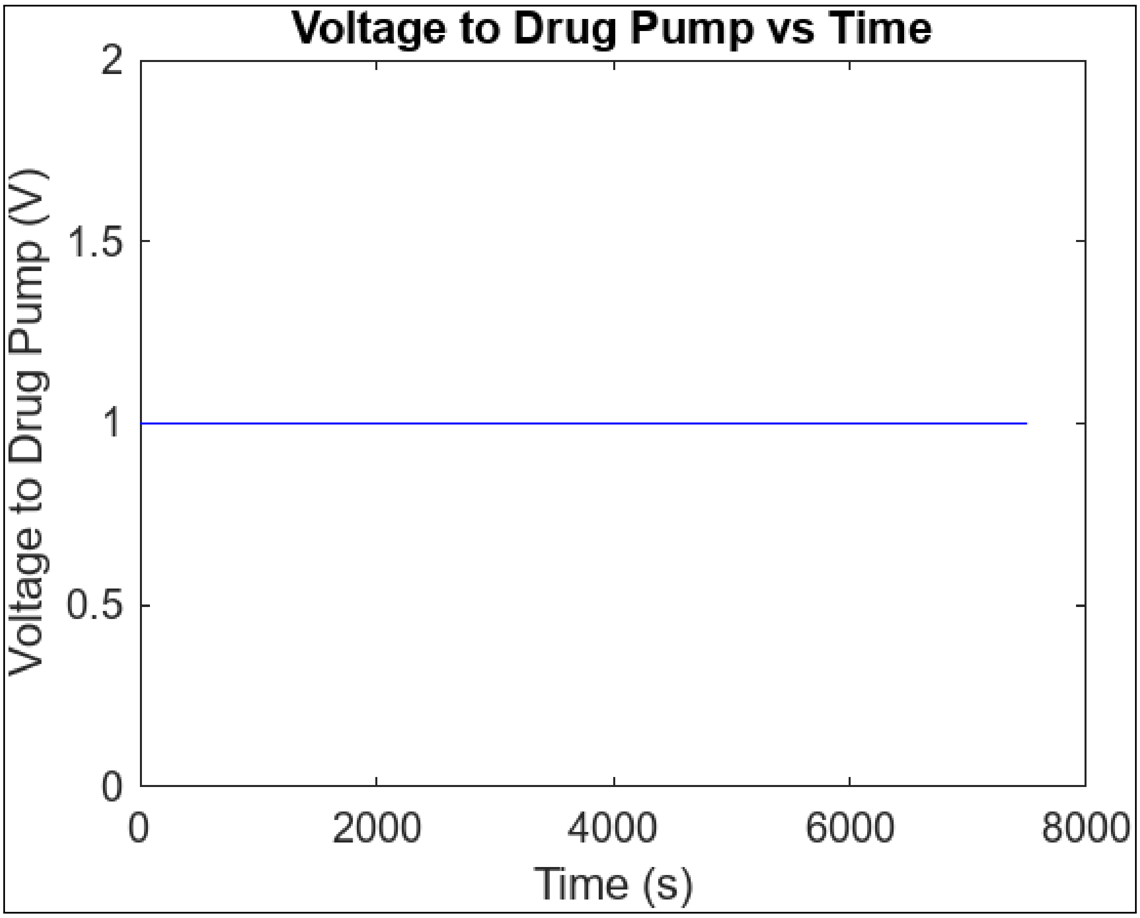


Figure 2: The step response used to begin understanding our plant system.

We decided to use an array of 5 pulse generators as the input to our patient model. Since there were 3 main parameters that could be modified within each of the pulse generators (amplitude, period, and pulse width) we decided to randomly generate numbers within a reasonable range for each of these parameters in all our pulse generators.

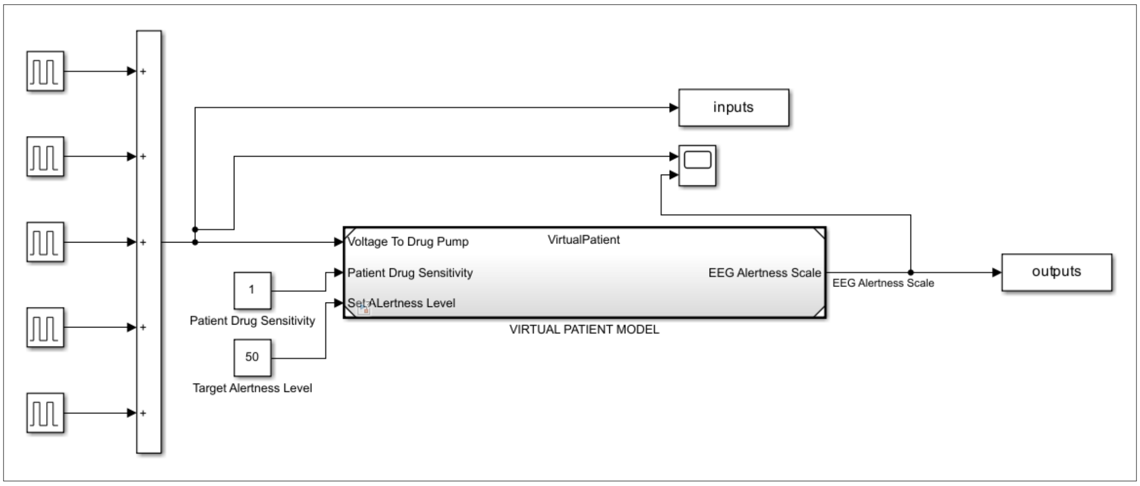


Figure 3: Image of our open loop testing setup with the pulse generators summed on the left-hand side of the image.

Since the maximum input voltage to the drug pump was 5V, we needed our amplitude values, to sum up to 5 in order to simulate a realistic input voltage which was 5V at most. We came up with 5 numbers (0.2, 1.2, 0.8, 1.1, 1.7) that summed up to 5 which we set as our pulse generator amplitudes. Next, we used MATLAB's "randi" function to generate 5 numbers between 10 and 2000 for the periods in seconds of our pulse generators (command: randi([10, 2000], 1, 5), output: 154, 789, 1205, 1923, 27). We chose this range for the periods as we felt as if it would represent a representative range of high and low frequencies across our system testing period. Finally, we used the same function to generate 5 numbers between 3 and 20 for the pulse width as a percentage of the period (command: randi([3, 20], 1, 5), output: 5, 19, 10, 17, 7). We chose this range for our pulse widths as we felt that it would increase the variation of our input without the widths being so large that the pulses meld together. We opened the loop by removing the controller and directly attaching our summed array of inputs to the drug pump voltage port of the plant function. A scope was used to visually inspect the input and output, and two variables labeled "inputs" and "outputs" were used to send the input and output to our workspace for system identification of the loop transfer function. Upon completion of running the system, we had a plot of our input and output which suggested a high level of randomness in both. The variation in our output response is far higher than what was seen in the step response and represents a large range of possible EEG values, which was ideal for identifying our system.

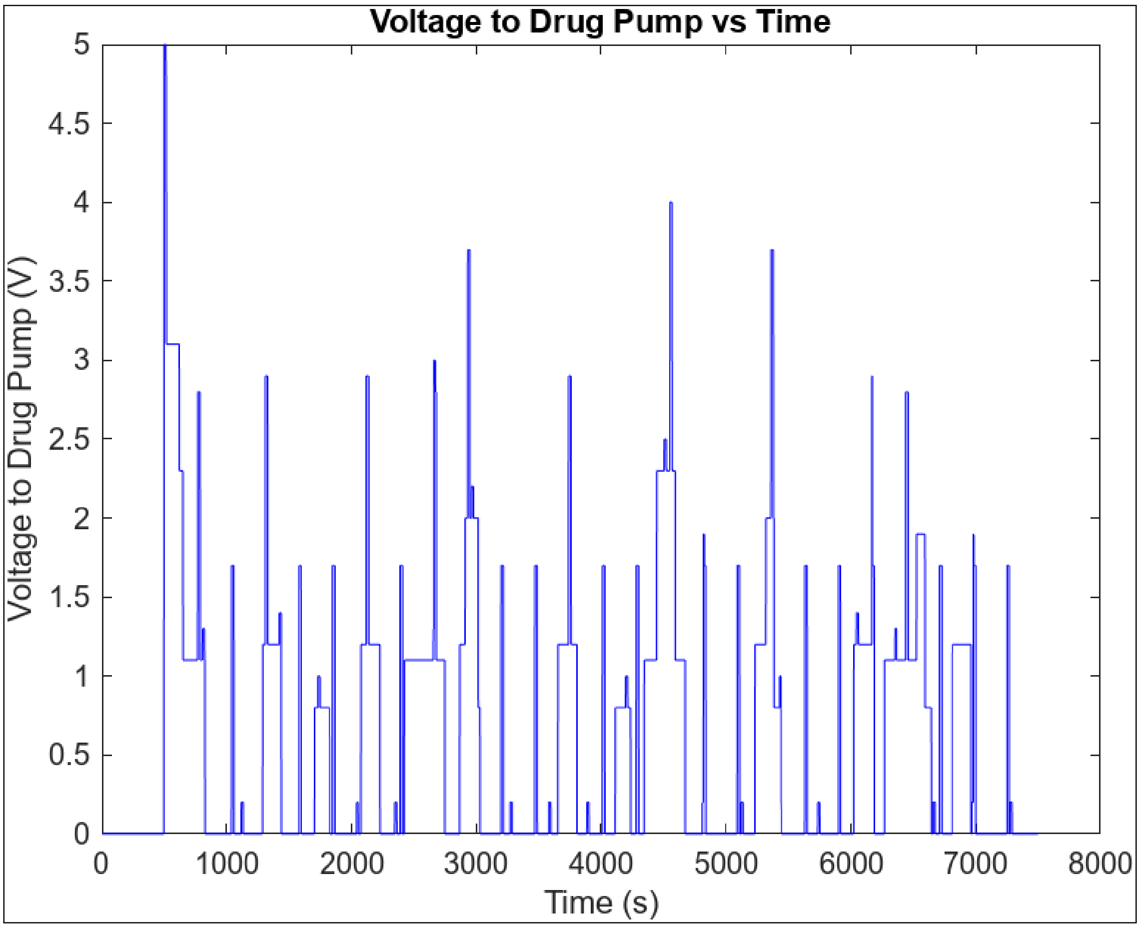


Figure 4: Input to our plant that was used for further system identification.

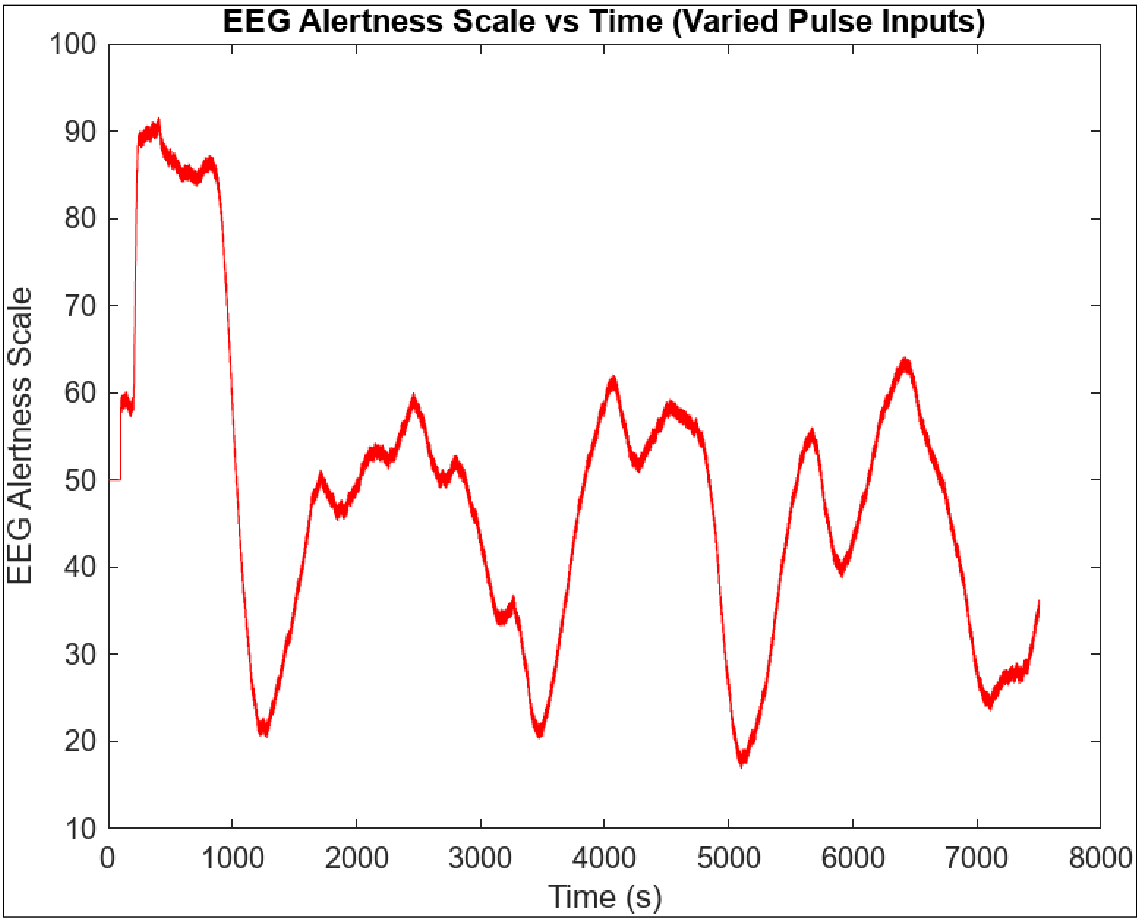


Figure 5: Output response from our plant that was used for further system identification.

We then ran a parametric identification routine to find the transfer function of the plant system, shown below.

```
hs_plant =

                                 -0.001133
  exp(-357*s) * ------------------------------
                s^2 + 0.008204 s + 1.725e-05

Continuous-time transfer function.
```

Figure 6: Open loop transfer function characterizing our plant system.

## 0.3 Controller Design

The technique applied by our team to design the controller was applying the built-in MATLAB function pidTuner to the closed loop function calculated from our parametric-fit code above that was then applied to our system. PidTuner allows the user to manually change rise, settling, and overshoot times while adjusting to different Kp, Ki, and Kd values. Pictured below are the values we found with pidTuner that minimized our settling and overshoot the most. PidTuner also ensured us that our closed-loop system was stable and outputted the closed-loop system's gain and phase margins.

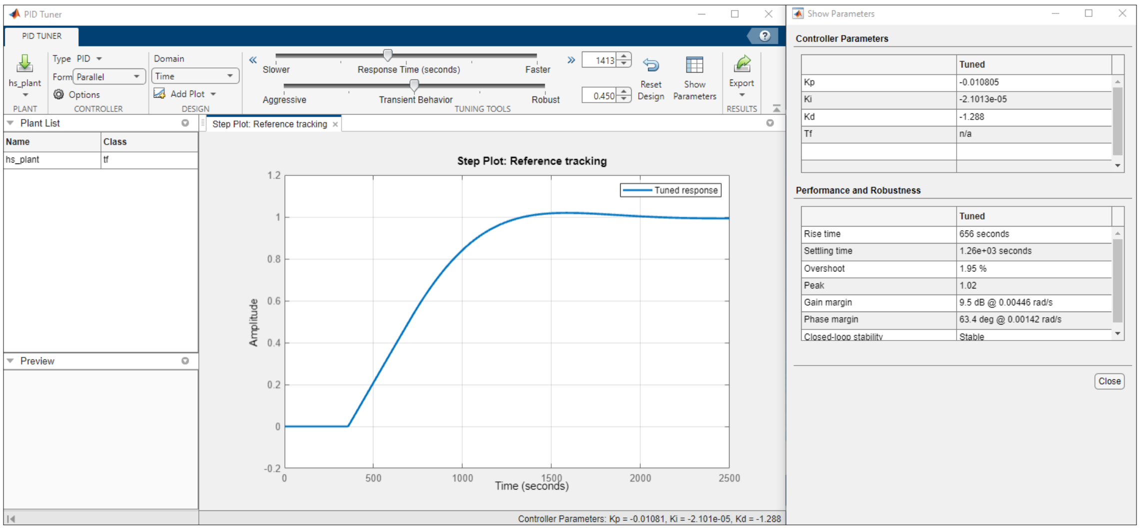


Figure 7: Parameters entered into our PID tuner and the relevant outputs.

The output from this PID tuner was then turned into a transfer function using MATLAB's "pid" command, resulting in the following transfer function for our controller.

```
hs_controller =

  -128.8 s^2 - 1.081 s - 0.002101
  -------------------------------
            s^2 + 100 s

Continuous-time transfer function.
```

Figure 8: Closed loop transfer function characterizing our controller system.

We analyzed the stability of this transfer function through a pole plot, which showed that both poles were in the left-hand plane of the graph, indicating stability for this closed-loop transfer function.

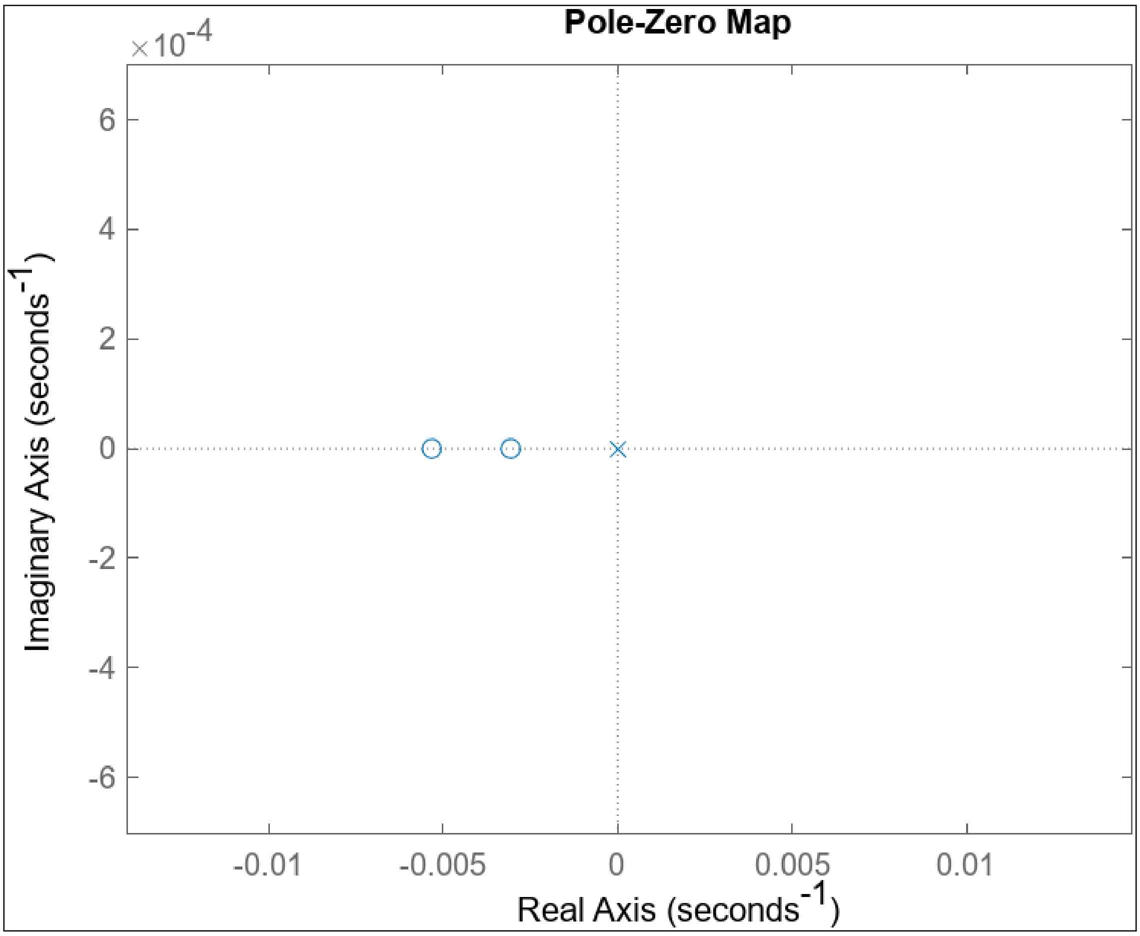


Figure 9: Pole zero plot showing the stability of our controller transfer function.

## 0.4 Design Trade-offs

The biggest trade-off of our current system is that we had to have a higher settling time to ensure our controller overshoot did not go over 10%. Also, due to our current controller being a PID controller, our system suffers from similar issues that systems with PID controllers have like struggles with robust changes to system dynamics or unforeseen large disturbances. Also, our PID controller uses the derivative term which due to its aggression is often not used in real applications so this controller might not be the safest for implementation in real-world situations. PID controllers are inherently non-linear systems and hence can have limited effectiveness for non-linear systems [3].

Moreover, the input to our master model was varied step responses, six in our case. We chose this method over using a Randn due to Randn having the following problems:

- Lack of frequency control: the frequency input is not directly controllable and may result in an inaccurate parameter estimation or analysis of system behavior.
- Non-ideal noise properties: randn inputs have non-zero means or non-uniform distributions which result in inaccurate estimates of noise properties and effects the accuracy of the system tested.

Using the varied step responses was better to use as it allowed us to excite different dynamic modes of the system at both high and low frequencies, improve parameter estimations, and overall lead to a more robust analysis of the system. Due to us using a limited amount of varied step input responses, there is a possibility that we could have got a better or different fit in our measured and predicted values which resulted in a different transfer function which in turn can affect our rise time and decrease our settling time. From the analysis below it is clear to see our settling time is greater than 1000 seconds which is not ideal as it does not get to the 50-point mark as soon as possible.

Theta values were pretty large, our 4 theta values which represented the offset to the system was 100. Even though this angle was large our predicted and measured values were almost perfectly aligned and hence it made sense to use it. However, this came with considerable trade-offs. The offset angle refers to the deviation of the output from the desired value, which meant our system deviated significantly from the desired value. This problem could be due to the fact that the system is not well calibrated or there are errors in the control algorithm [4].

Our time delay that was programmed into the system has both positive and negative effects. In the proposal, our time was delayed by 500 seconds and this functioned to filter out some of the noise as it acts as a low pass filter and attenuates the high frequencies. However, there are also some negative effects of a time delay: performance of the system is reduced as it can cause slow response times, be very sensitive to time changes, and in some cases be difficult to analyze a system due to the additional time component [5]. Time delay can also lead to more oscillations in a system and cause instability which is problematic when trying to control for overshoot values, settling time, and other factors

## 0.5 Controller Evaluation

In order to ensure the stability of our overall system, we treated our overall system as a loop transfer function. To do this, we multiplied the controller transfer function by the plant transfer function. The new overall system transfer function is pictured below.

```
loop_tf =

                      0.1459 s^2 + 0.001224 s + 2.38e-06
  exp(-357*s) * ----------------------------------------
                   s^4 + 100 s^3 + 0.8205 s^2 + 0.001725 s

Continuous-time transfer function.
```

Figure 10: Loop transfer function of characterizing our entire system of both the plant and designed controller.

In order to test the stability of our loop transfer, we created bode plots which are pictured below, as seen both the phase and gain margins are positive which means the function is stable.

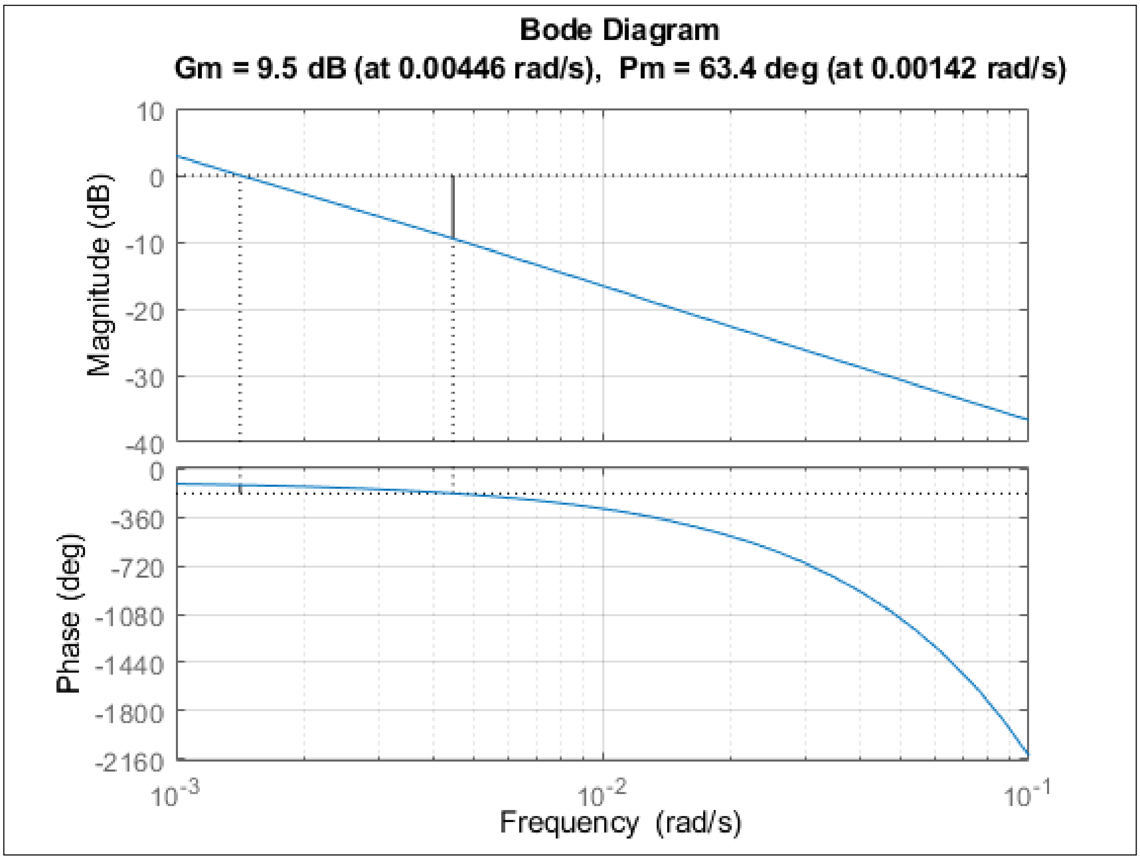


Figure 11: Bode plot demonstrating the stability and characteristics of our entire system loop transfer function.

In order to solve for time delay, we used the time delay function we learned in class which consists of

$$PhaseMargin/(GainCutoffFrequency * (180/\pi))$$

. Using the data taken from the bode plots we found our max time delay for our system to be **247.9835 seconds** or 4.1 minutes. So this controller would probably work well for cloud computing.

We also created frequency response plots of the overall system in order to characterize it as seen below.

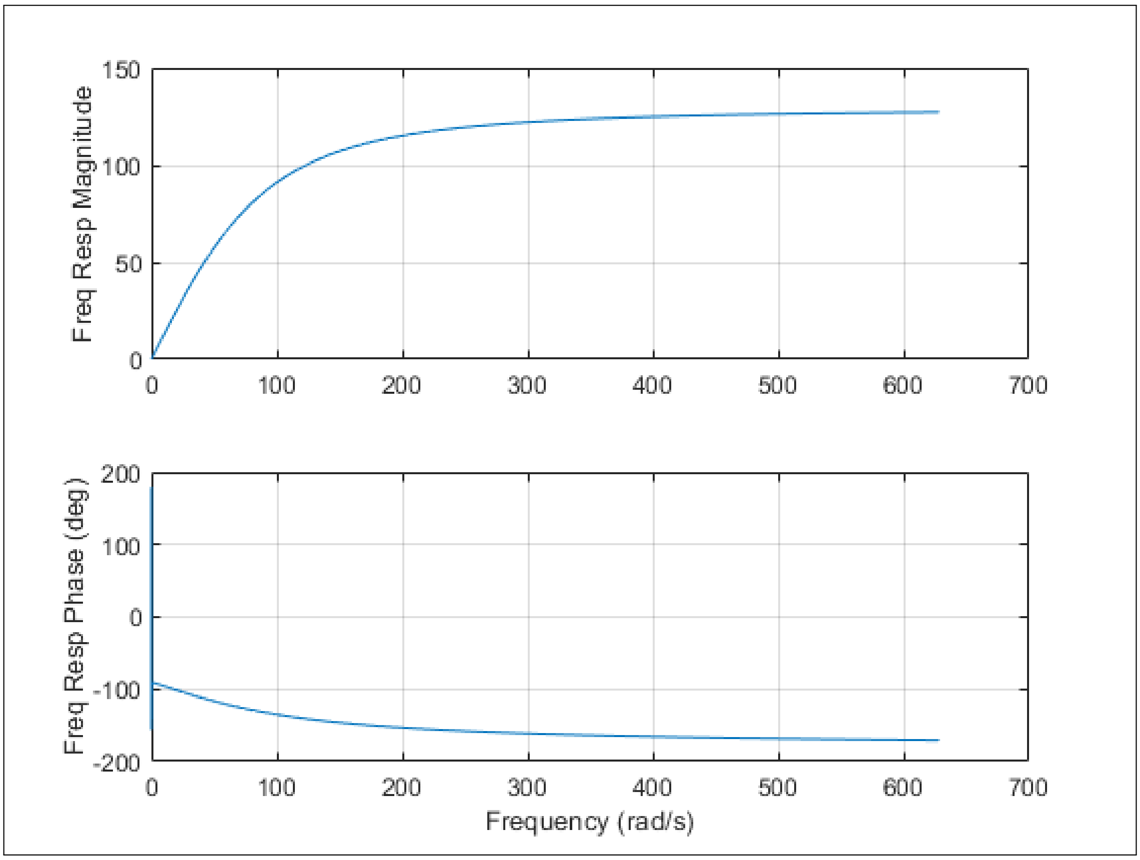


Figure 12: Frequency response phase and magnitude of our system.

In order to characterize our controller, we ran the controller through a range of patient sensitivities ranging from 0.9 to 1.1. We then assessed the performance of our controller by checking the 10% settling time, 90% settling time, percent overshoot, and the steady state error of these different patient situations as seen in the table below.

| Sensitivity | Settling Time 10% (sec) | Settling Time 90% (sec) | Percent Overshoot | Steady State Error |
|---|---|---|---|---|
| 1 | 1011.7 | 1612.6 | 14.17 | 0.0669 |
| 1.1 | 950.08 | 1238.2 | 19.12 | 0.0600 |
| 0.9 | 1134.2 | 1672.7 | 8.720 | 0.0731 |

We found that the controller responses for the extreme ranges of patient sensitivities were within the acceptable ranges of 40 to 60 on the EEG alertness scale.

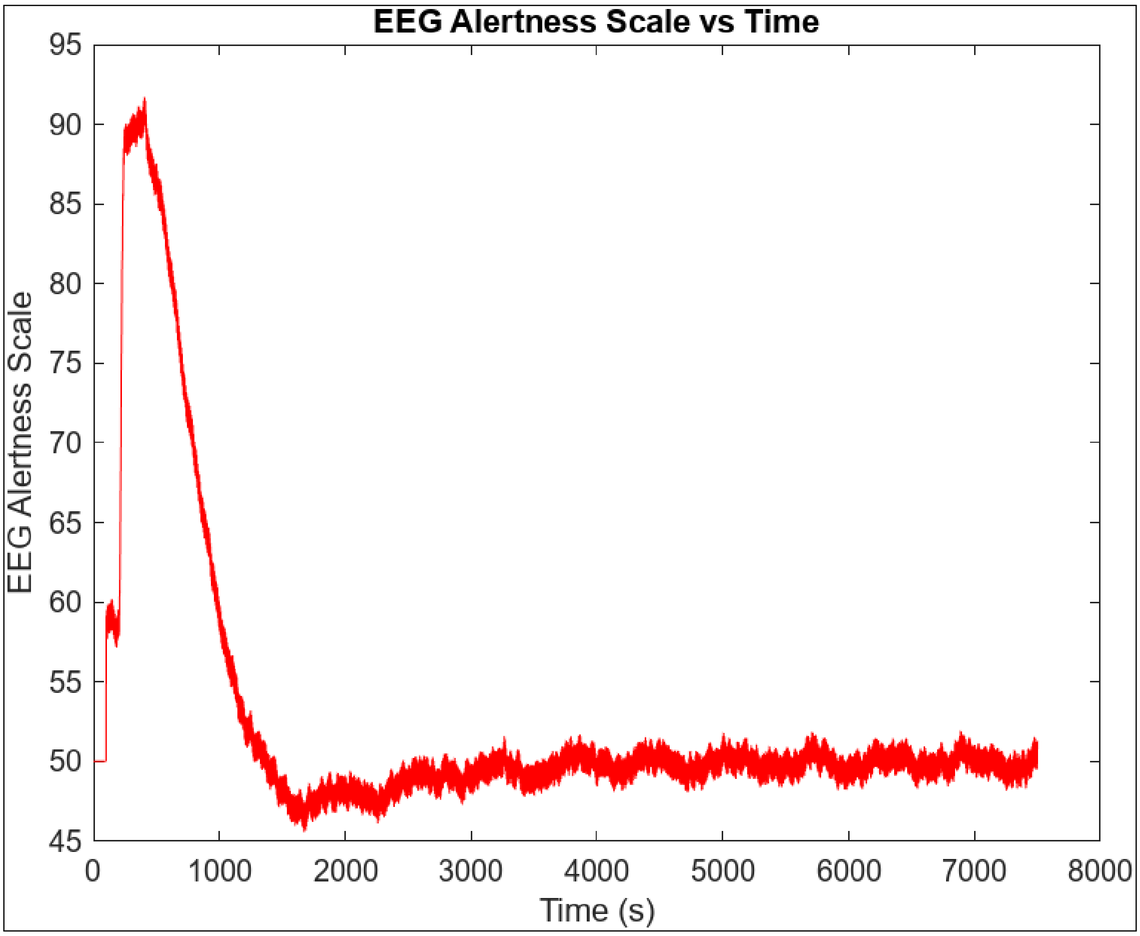


Figure 13: Response of our entire loop transfer function given a patient sensitivity of 0.9.

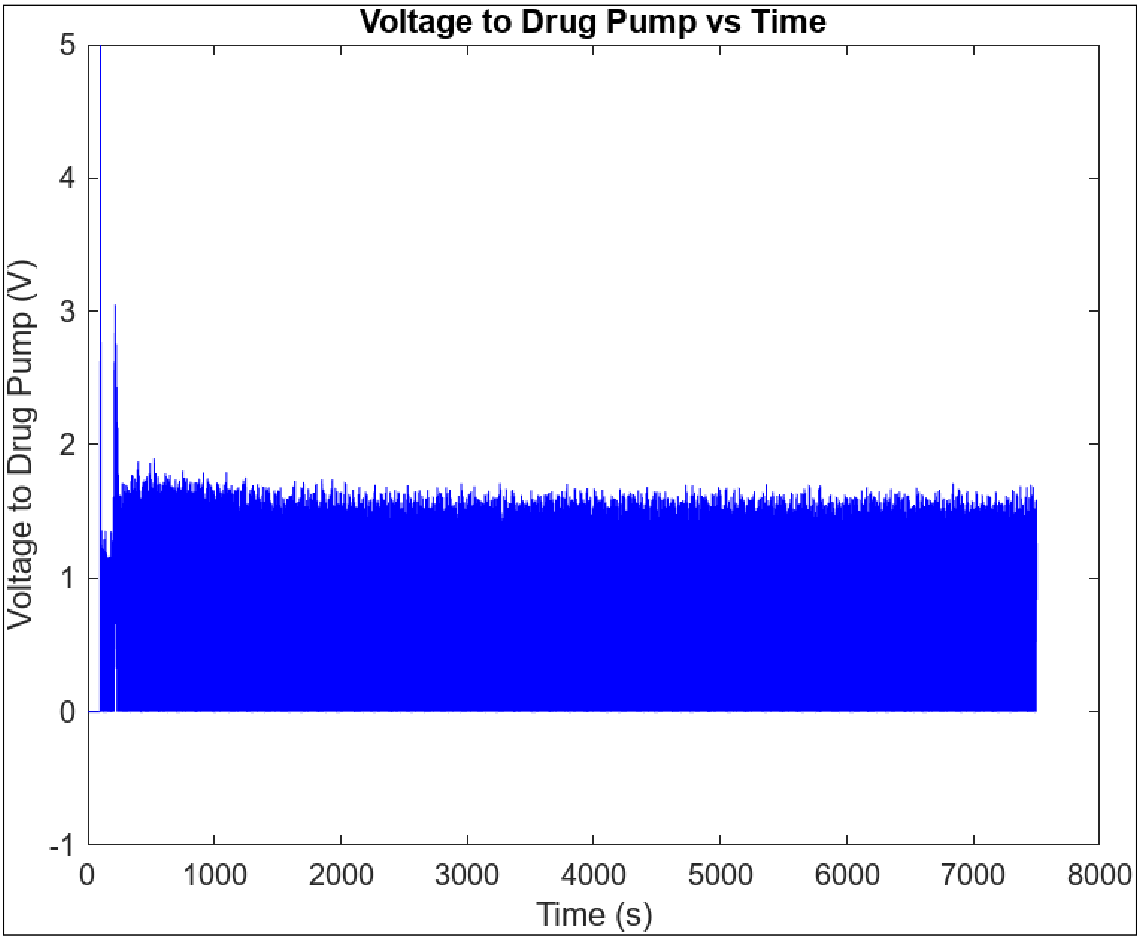


Figure 14: Input to our entire loop transfer function given a patient sensitivity of 0.9.

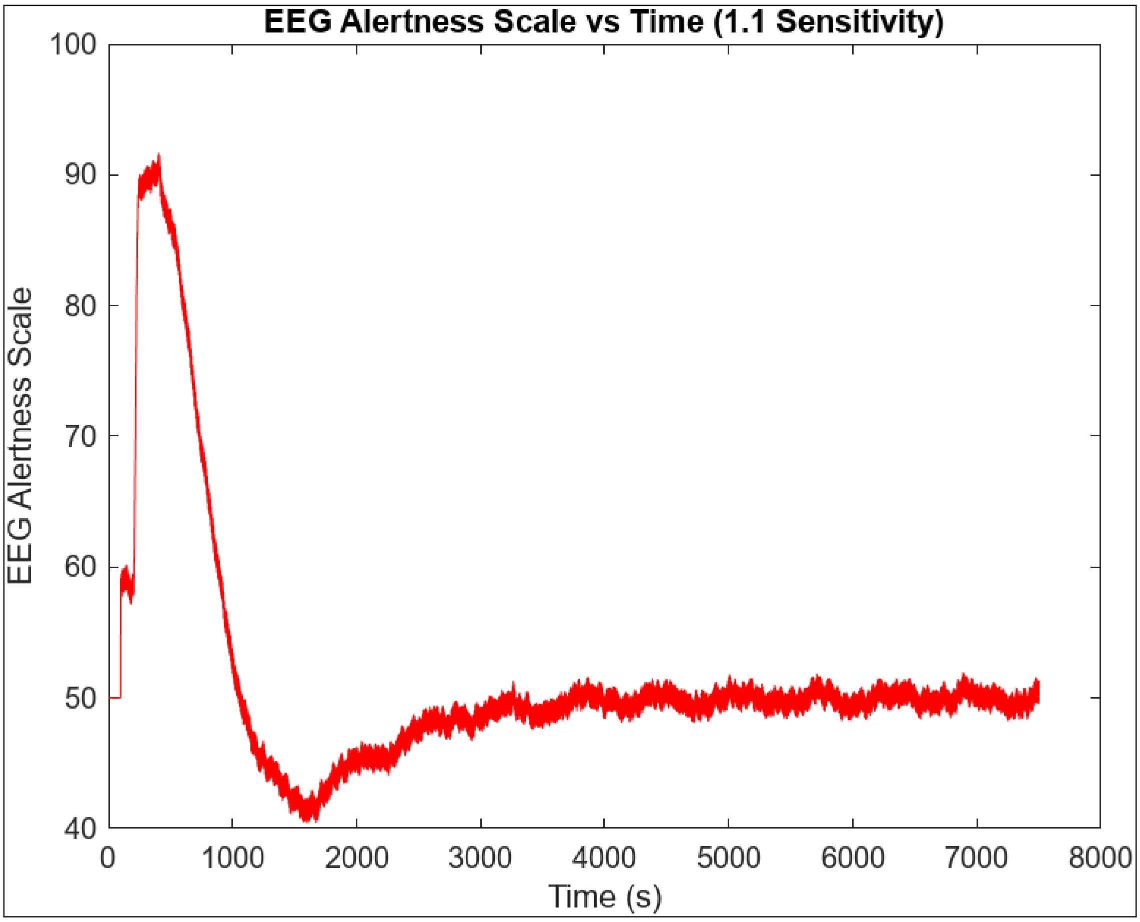


Figure 15: Response of our entire loop transfer function given a patient sensitivity of 1.1.

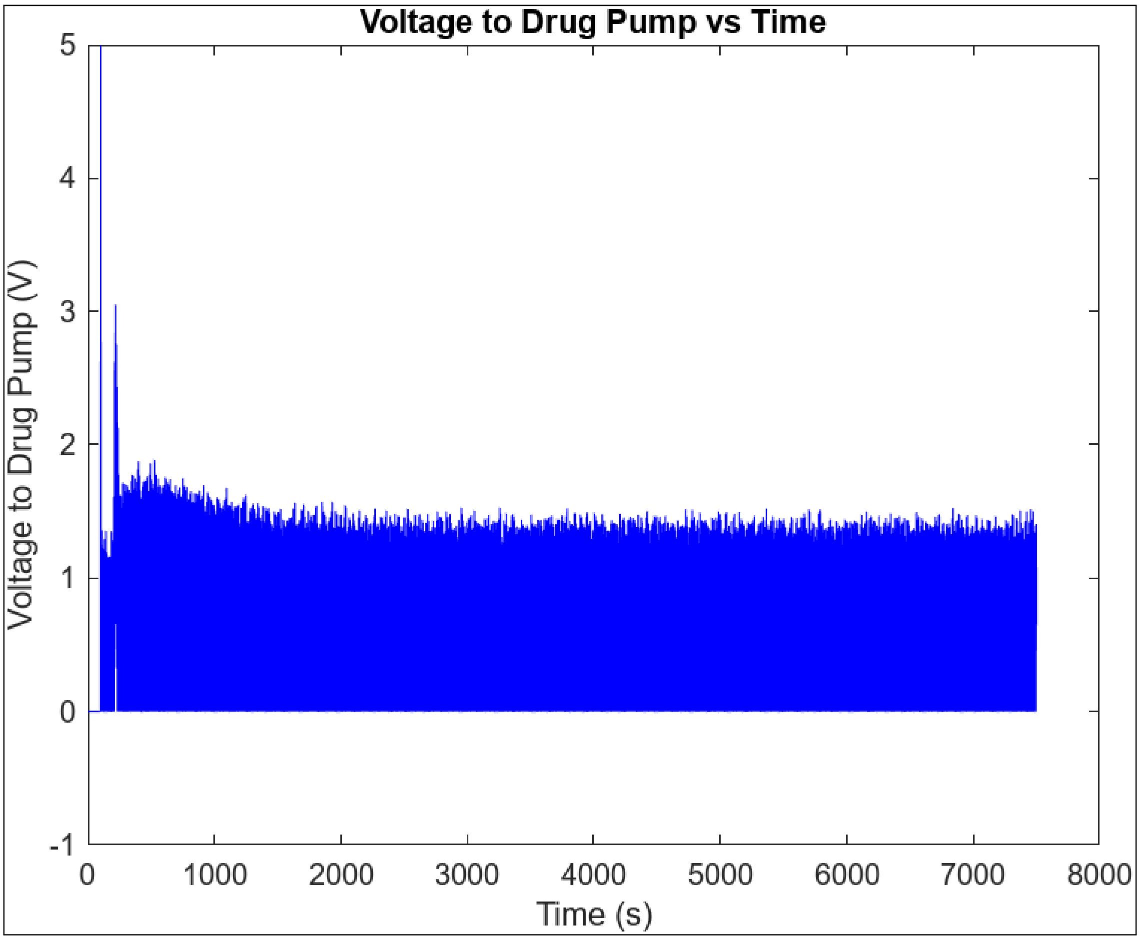


Figure 16: Input to our entire loop transfer function given a patient sensitivity of 1.1.

## 0.6 Controller Performance

Our controller performed very well connected to the plant, as it reached a constant 50 on the EEG alertness scale which was ideal to sedate the patient without too many side effects.

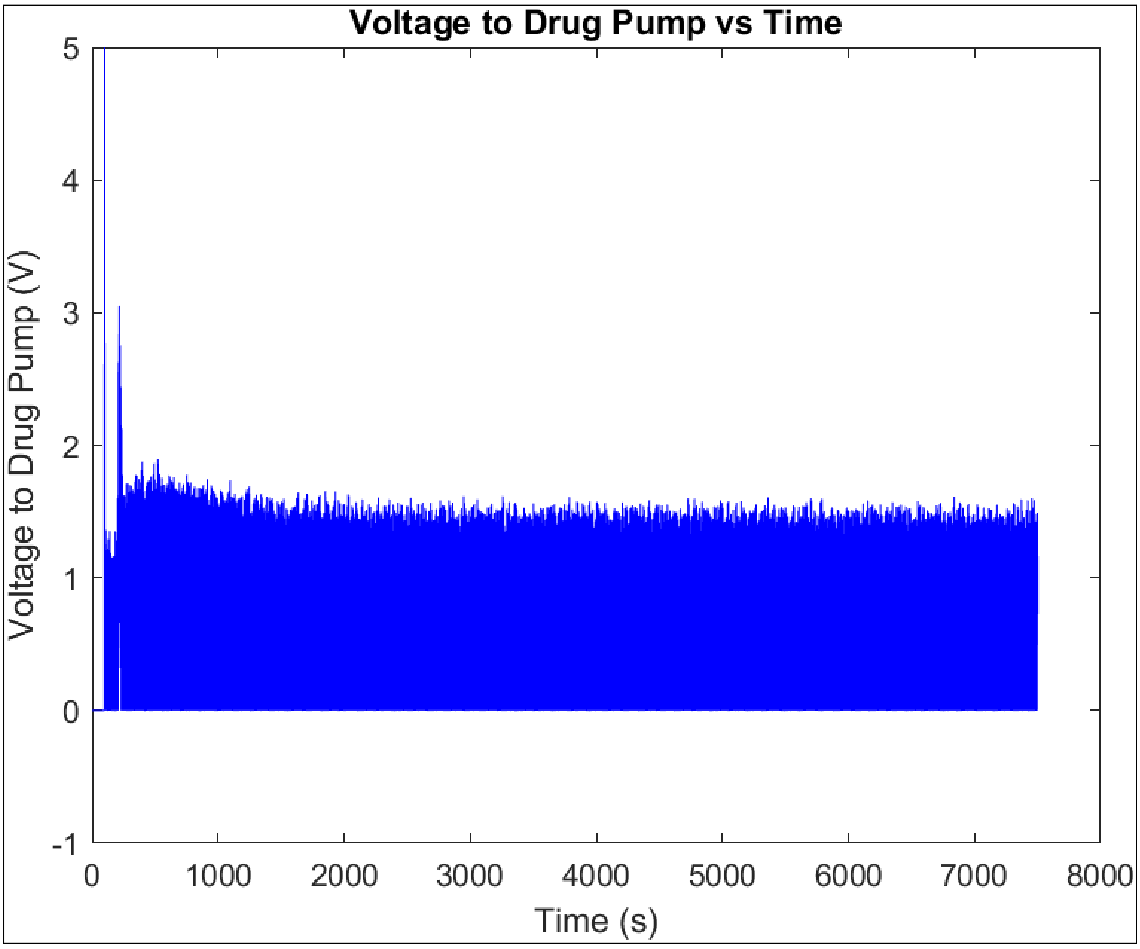


Figure 17: Input to our entire loop transfer function given a patient sensitivity of 1.

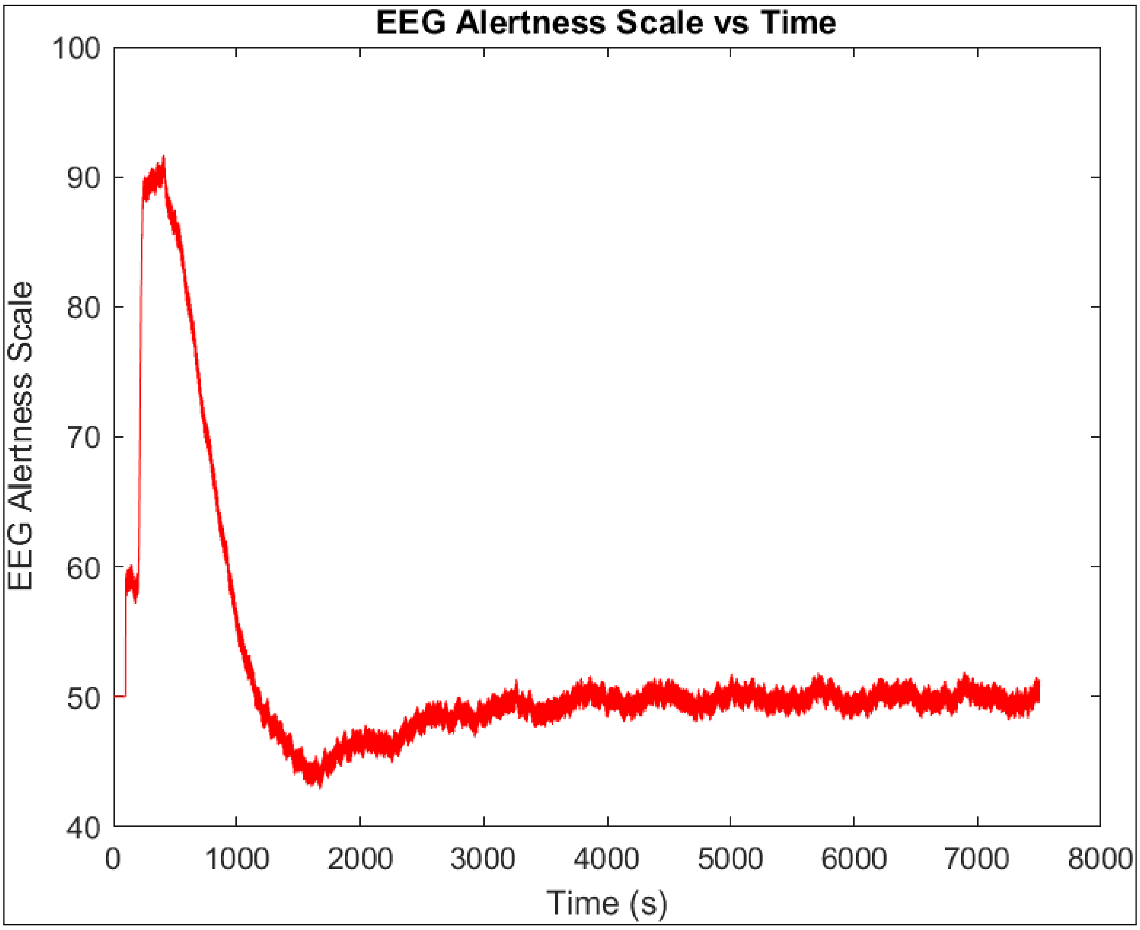


Figure 18: Response of our entire loop transfer function given a patient sensitivity of 1.

## 0.7 Closing Remarks

In conclusion, we feel our device is potentially useable for medical applications by UICU. As seen above, our controller is still able of stabilizing in different patient sensitivities but sensitivities over 1.1 might potentially cause the output to dip outside of the safety zone of 40-60 on the EEG alertness scale, it gets close however it is still within range (with an overshoot of 19.2%). The overall system lies very close to the true value of 50 on the alertness scale and hence should not cause the patient too much pain or too many postoperative complications. Our graph settles after approximately 2000 seconds. Moreover, this device is more than useable as the amount of lag that is tolerable to the system is around 4.1 minutes before the system becomes unstable. Testing our controller using the pole zeros proved that the controller was stable and testing the overall loop transfer function using bode plots showed stability in the negative feedback loop from the patient's EEG to the controller. Due to the device's stability and high lag tolerance, it would be advisable. Another potential issue is still due to our device relying on the derivative term of the PID controller which means the controller is somewhat aggressive and might struggle under certain disruptive or largely deviant patient input data. In addition, reaching 50 would be ideal as soon as possible and not after 2000 seconds. Overall more improvements can be made to the variance of inputs to see if results differ and testing needs to be done clinically before more can be said about the device.

## 0.8 References


[1] Lone, P., Wani, N., Ain, Q., Heer, A., Devi, R., & Mahajan, S. (2021). Common postoperative complications after general anesthesia in oral and maxillofacial surgery. National Journal of Maxillofacial Surgery,12(2), 206. https://doi.org/10.4103/njms.njms_66_20

[2] Shin, H. W., Kim, H. J., Jang, Y. K., You, H. S., Huh, H., Choi, Y. J., Choi, S.U., & Hong, J. S. (2020). Monitoring of anesthetic depth and EEG bandpower using phase lag entropy during propofol anesthesia. BMC Anesthesiology, 20(1). https://doi.org/10.1186/s12871-020-00964-5

[3] Chang, W.-D., & Yan, J.-J. (2005). Adaptive robust PID controller design based on a sliding mode for uncertain chaotic systems. Chaos, Solitons & Fractals, 26(1), 167{175. https://doi.org/10.1016/j.chaos.2004.12.013

[4] M. M. Farooq and S. A. Qureshi, (2010). Effects of Time Delay on Feedback Control Systems. International Journal of Computer Applications, 1-5

[5] Khalil, A. F., & Wang, J. (2015). Stability and Time Delay Tolerance Analysis Approach for Networked Control Systems. Mathematical Problems in Engineering, 2015, 1{9. https://doi.org/10.1155/2015/812070