Introduction to Control and Robotic Systems

EE-386

Design-Project

Hunter Wilhelm Phillips

April 17, 2019

GitHub Repository at

https://github.com/robolux/EE_Controls

1. Introduction

The core of this design project lies in the simulation of an antenna angular position controller in Matlab Simulink. The goal of the controller is to provide an angular output that corresponds to the desired position of the system. It is noted that the controller must handle a simulated wind disturbance in the form of a step function. There will be several simulations undertaken including first a Proportional Derivative (PD) controller then a Proportional Integral Derivative (PID) controller. Each controller will be tested with and without an appropriately designed prefilter coupled to the simulation model. The extensive testing to be completed will provide a basis for comparison between each control system and its associated drawbacks and advantages. First the mathematical model will be formulated and the problem statement outlined in preparation for the controller design stage. With this established the control systems will be designed and implemented in Simulink to perform the necessary simulations and associated graphical data. The stability of the system will also be confirmed through the use of the Routh-Hurwitz Criterion, Nyquist Criterion, and Bode Plots. The results of the runs and stability study will be analyzed and formed into a conclusion concerning the scope of the design project.

2. Mathematical Modeling and Problem Formation

2.1 System Decomposition

To initially characterize the antenna controller system the block-diagram of the armature controlled DC motor system as seen in *Figure 1* needs to be reduced into a transfer function.

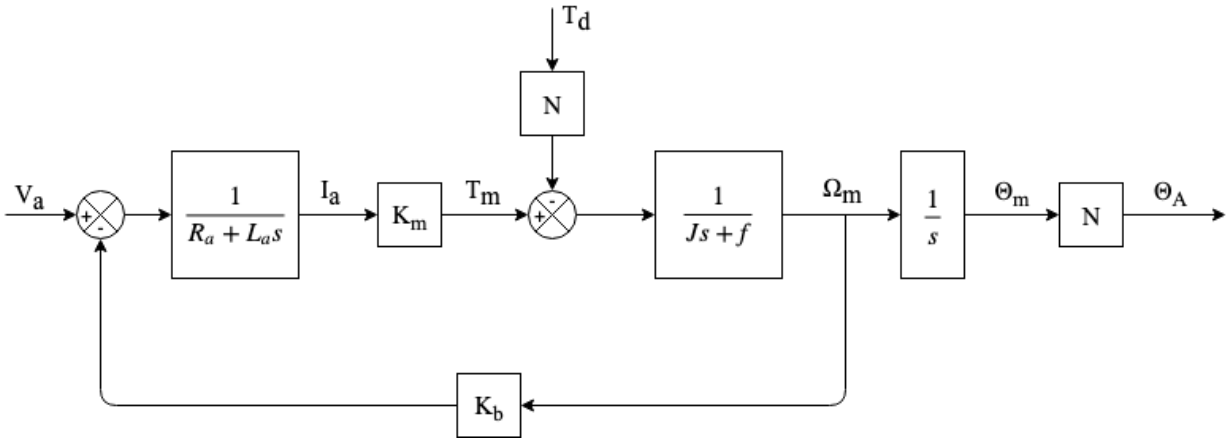

Figure 1: Block-diagram of an Armature Controlled DC Motor

To solve for the transfer function $\frac{\Theta_A}{V_A}$ in the block diagram; T_D is set equal to zero. Using simple block reduction techniques, the block diagram can be condensed into the transfer function system seen in *Figure 2*.

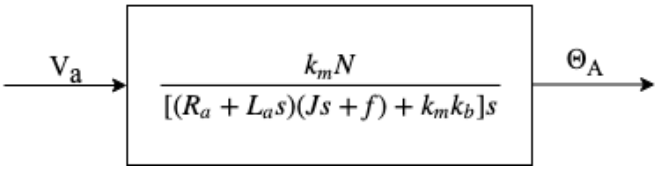

Figure 2: Transfer Function after reduction of Block-diagram

This finalizes our general block diagram of the system under examination, which prepares us for the next step, implementing the control for it.

2.2 PD Controller Model

After reducing the transfer function for the armature controlled DC motor it was time to derive the mathematical model for the PD controller. First the model without a prefilter was formed as shown in *Figure 3*.

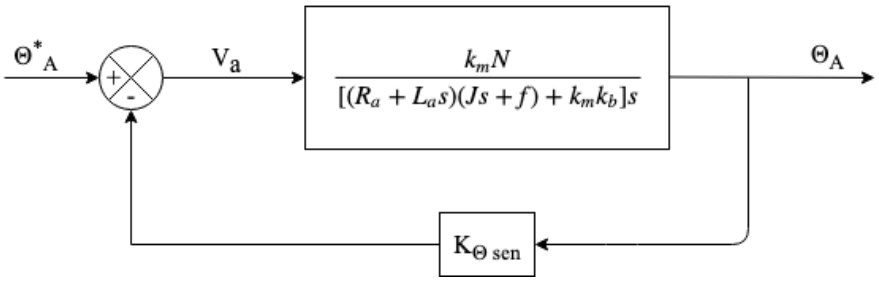

Figure 3: PD Controller Block-Diagram without Prefilter

The next step was to obtain the model of the PD controller with a prefilter as seen below in *Figure 4.*

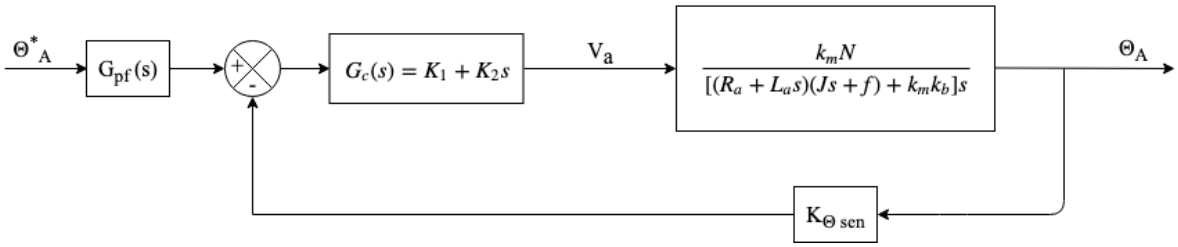

Figure 4: PD Controller Block-Diagram with Prefilter

2.3 PD Controller Design

Taking the derived mathematical model in Section 2.2 with the prefilter we are able to determine appropriate coefficients and steady state values. It is assumed that the PD controller case without a prefilter is a sufficiently simple offshoot of the model with the prefilter and detail in obtaining the same coefficients will be understood. After taking the transfer function seen in Figure 2, (1) can be obtained by plugging in the values given in the project prompt.

$$\frac{\theta_A}{\theta_A^*} = \frac{(k_1 + k_2 s)(0.06)}{0.05s^2 + (0.46 + 0.06k_2)s + 0.06k_1} \tag{1}$$

Designing an appropriate prefilter for this PD design results in (2) being brought forth:

$$G_{pf} = \frac{k_1}{k_1 + k_2 s} \tag{2}$$

Updating the transfer function by applying (2) with (1) through multiplicative properties in block-diagram reduction results in (3) being created:

$$\frac{\theta_A}{\theta_A^*} = \frac{1.2k_1}{s^2 + (9.2 + 1.2k_2)s + 1.2k_1} \tag{3}$$

To determine the appropriate ʒ and $\omega_n$values for our performance requirements, the set of equations in (4) must be used.

$$P.O. = 100 \cdot e^{-\frac{ʒ\pi}{\sqrt{1-ʒ^2}}} \qquad t_s = \frac{4}{ʒ\omega_n} \tag{4}$$

It was prescribed that the desired specifications are P.O. of 10% with a settling time (t_s) of less than or equal to 1.2 seconds. It was decided that a reasonable settling time was 0.8 seconds which adhered to the requirements. This resulted in a $ʒ = \mathbf{0.5912}$ and $\boldsymbol{\omega_n = 8.4574}$ that were then used to find the values of k_1 and k_2 by plugging equation the second order normal form with (3). This resulted in a $\boldsymbol{k_1 = 59.6063}$ and $\boldsymbol{k_2 = 0.67}$. After plugging the k values back into (3), the final transfer function for the system without the wind disturbance can be seen in (5):

$$\frac{\theta_A}{\theta_A^*} = \frac{71.52756}{s^2 + 10s + 71.52756} \tag{5}$$

2.4 PD Controller Error

Taking the original block diagram, setting $\theta_A^* = 0$, results in a new block diagram that is reduced into (6).

$$\frac{\theta_A}{T_d} = \frac{-N^2}{s(Js+f) + K_m N(k_1 + k_2 s + \frac{k_b s}{N})} \tag{6}$$

The next set of steps to determine the error equation can be seen in (7):

$$\begin{aligned} E_A(s) &= \theta_A^*(s) - \theta_A(s) \qquad \text{setting } \theta_A^* = 0 \\ E_A(s) &= -\theta_A(s) \end{aligned} \tag{7}$$

The next step is to find e_{ss} using the steps as seen below in (8):

$$\begin{aligned} T_d(t) = 180 \cdot 1(t-3.0) \rightarrow T_d(s) &= \frac{180}{s} e^{-3.0s} \\ e_{ss_{T_d}} = \lim_{s\to 0} s(\frac{180}{s} e^{-3.0s})(&\frac{N^2}{s(Js+f) + k_m N(k_1 + k_2 s + \frac{k_b s}{N})} \end{aligned} \tag{8}$$

Using the given values in the prompt the steady state error was found to be: $\boldsymbol{e_{ss_{T_d}} \approx 0.5}$

2.5 PID Controller Model

The next step was to model the PID controller using the originally derived plant mathematical model. Since the PD and PID controller share a base model without a prefilter, *Figure 3* can be reused to show the localized design. Now the PID controller with a prefilter was created as showcased in *Figure 5*.

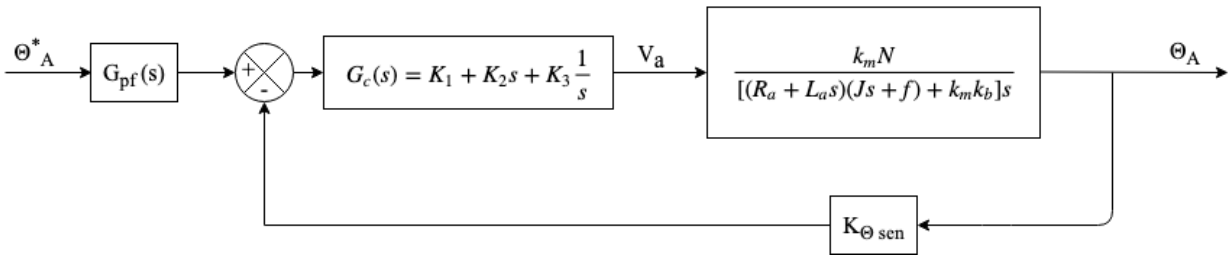

Figure 5: PID Controller Block-Diagram with Prefilter

2.6 PID Controller Design

Taking the block-diagram model reduced in Section 2.5, and performing block reduction techniques we can obtain that the transfer function is equal to (9).

$$\frac{\theta_A}{\theta_A^*} = \frac{0.06s\left(k_1 + k_2 s + \frac{k_3}{s}\right)}{0.05s^3 + (0.46 + 0.06k_2)s^2 + 0.06sk_1 + 0.06k_3} \tag{9}$$

It is observed that the case without a prefilter for the PID controller is simple enough to deduce without deriving the respective equation, thus knowing for this case with a prefilter the following transfer function form is assumed as shown in (10).

$$G_{pf} = \frac{k_3}{k_2 s^2 + k_1 s + k_3} \tag{10}$$

After multiplying (9) and (10) using block reduction techniques, (11) is the result:

$$\frac{\theta_A}{\theta_A^*} = \frac{1.2k_3}{s^3 + (9.2 + 1.2k_2)s^2 + 1.2k_1 s + 1.2k_3} \tag{11}$$

We can use the ITAE criterion in the form seen in (12) to help find our natural frequency:

$$s^3 + 1.75\omega_n s^2 + 2.15\omega_n^2 s + \omega_n^3. \tag{12}$$

Using $\boldsymbol{t_s = 0.8s}$, we find that $\boldsymbol{\omega_n = 12.5}$. From this, we get the k values of $\boldsymbol{k_1 = 279.9}$, $\boldsymbol{k_2 = 10.56}$, and $\boldsymbol{k_3 = 1627.6}$. Plugging these values into (11) condenses down to the final form of (13).

$$\frac{\theta_A}{\theta_A^*} = \frac{1953.12}{s^3 + 21.9s^2 + 335.9s + 1953.12} \tag{13}$$

2.7 PD Controller Error

Taking the initial block diagram, setting $\theta_A^* = 0$, allows for a new block diagram being reduced into (14).

$$\frac{\theta_A}{T_d} = \frac{-N^2 s}{(Js + f)s^2 + k_m\left(k_1 s + k_2 s^2 + k_3 + \frac{k_b s^2}{N}\right)N} \tag{14}$$

Using (7) to determine the error equation in the s-domain means that the error steady state value can be determined by the steps in (15).

$$T_d(t) = 180 \cdot 1(t - 3.0) \rightarrow T_d(s) = \frac{180}{s} e^{-3.0s}$$
$$e_{ss_{T_d}} = \lim_{s \to 0} s \left(\frac{180}{s} e^{-3.0s}\right)\left(\frac{N^2 s}{(Js + f) + k_m\left(k_1 s + k_2 s^2 + k^3 + \frac{k_b s^2}{N}\right)N}\right) \quad (15)$$

It can be seen that the s located on the upper side of the dominant transfer function results in zero divided by all other values, resulting in an $\boldsymbol{e_{ss_{T_d}} = 0}$**.** It can be noted that this is an expected outcome due to the higher accuracies with respect to the PID controllers' performance attributes.

3. Discussion of Controller Design

3.1 PD without Prefilter

The first and basic controller to be implemented in Simulink was the PD controller without a prefilter as showcased in *Figure 6.*

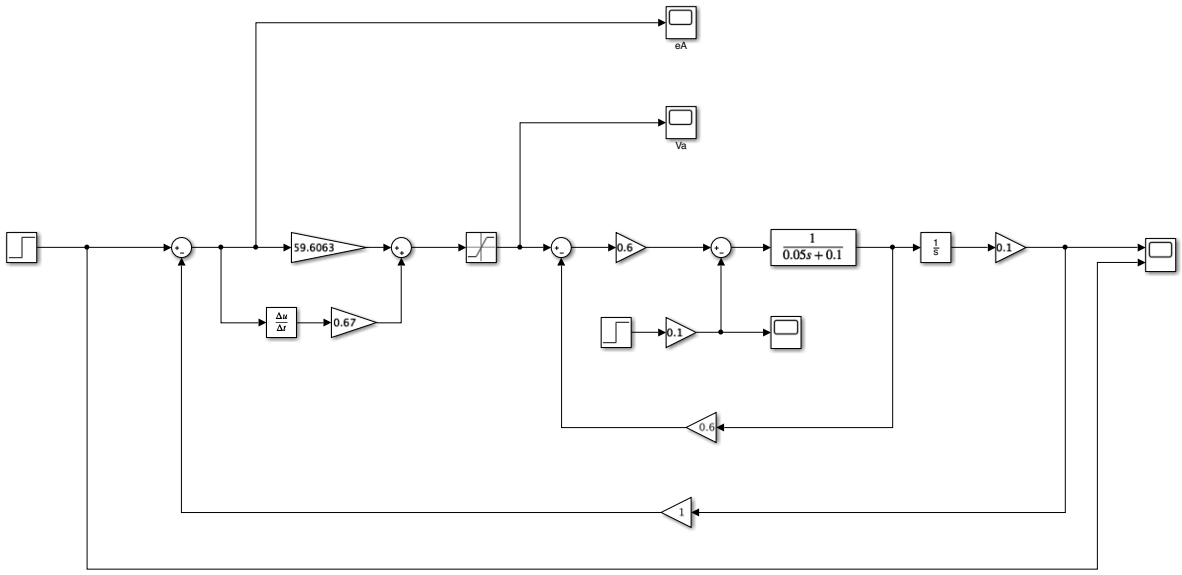

Figure 6: PD Controller without Prefilter in Simulink

When the sim was ran later on the dual input scope had the resulting curves turned on and off to provide two plots instead. This design was fairly trivial and next a prefilter was added to it.

3.2 PD with Prefilter

The second controller implemented in Simulink was the PD controller with a prefilter as seen in *Figure 7.*

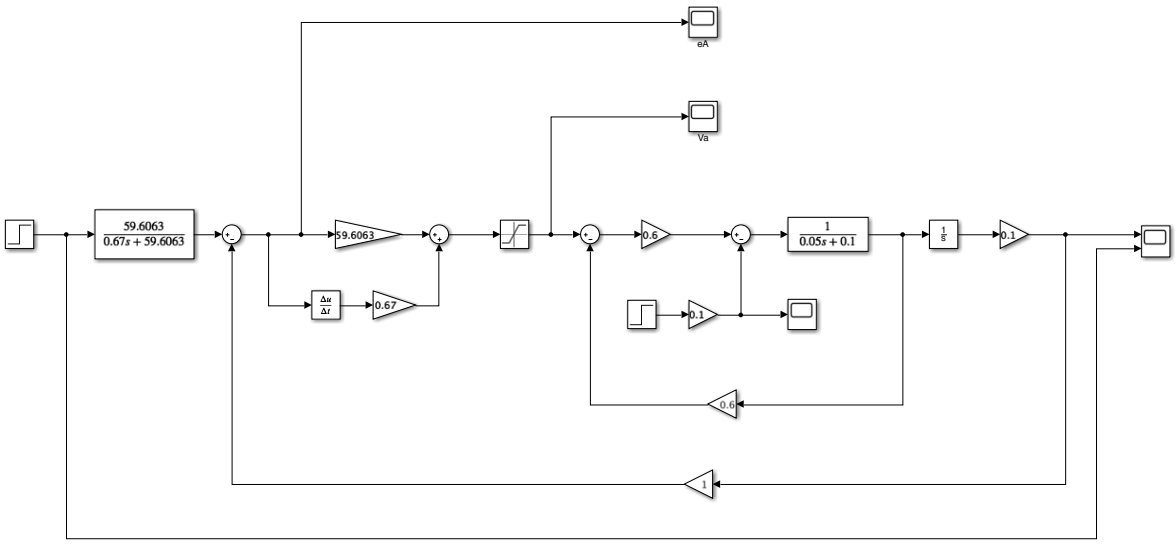

Figure 7: PD Controller with Prefilter in Simulink

The prefilter as seen is a fairly simple addition with a powerful result in the final simulations.

3.3 PID without Prefilter

The third controller that was created in Simulink was a PID controller without a prefilter as shown below in *Figure 8*.

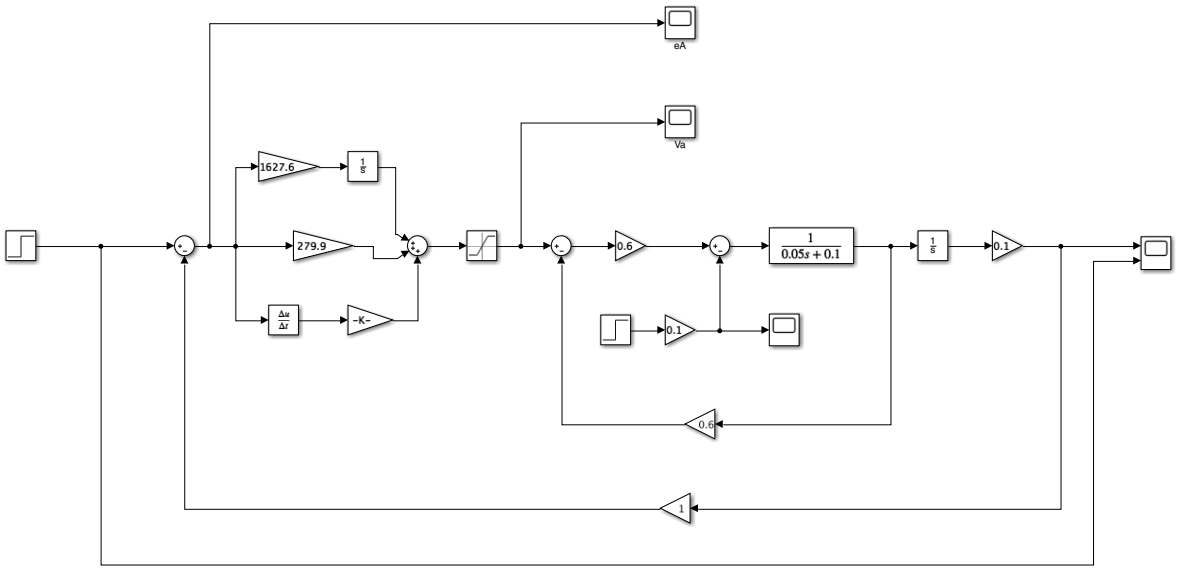

Figure 8: PID Controller without Prefilter in Simulink

The major difference of the integrator added with respect to the proportional and derivate aspects already seen before.

3.4 PID with Prefilter

The fourth and final controller developed was a PID controller with prefilter as observed in *Figure 9*.

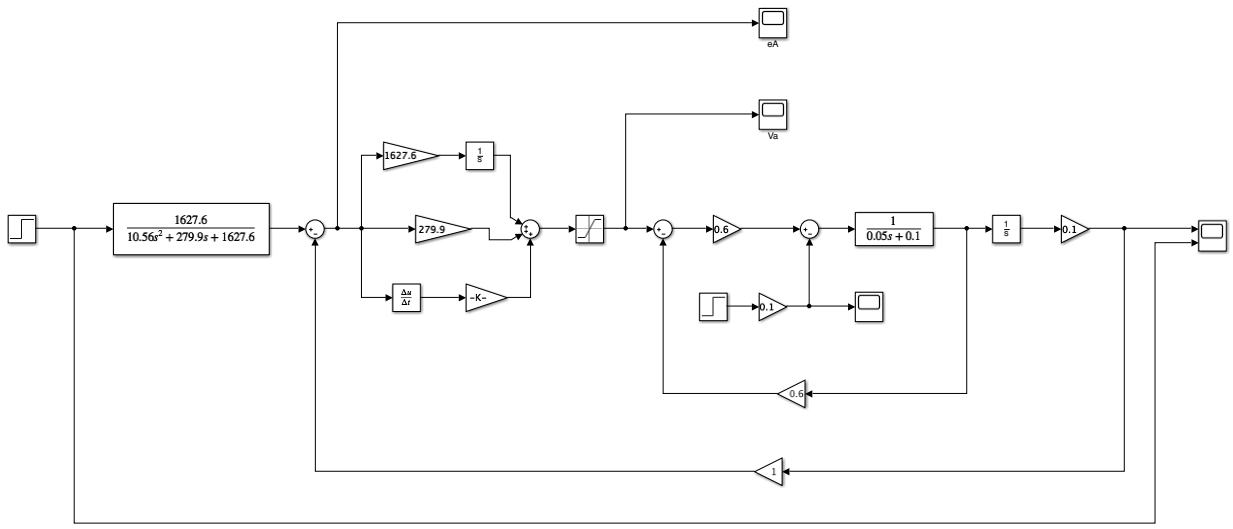

Figure 9: PID Controller with Prefilter in Simulink

The prefilter can observed to be one order higher than the previously seen PD controller, this is expected due to the integral term being added in the PID controller.

4. Stability Analysis

4.1 PD Controller

4.1.1 Routh-Hurwitz Criterion

The stability of the system is critical in ensuring that the outcome will come out as expected. The first test to ensure that stability is achieved is the Routh-Hurwitz Criterion being applied to the PD controller. The G(s) characteristic equation to quantify this analysis is selected to be $0.05s^2 + 0.5s + 3.58k_1$. The results of the criterion can be seen in *Table 1*.

Table 1: Routh-Hurwitz Criterion of Stability for PD Controller

s^2	0.05	3.58
s^1	0.5	0
s^0	3.58	0

It can be observed that there is no sign change occurring in the table, which means that the PD controller system is stable according to the Routh-Hurwitz Criterion.

4.1.2 Nyquist Criterion

The second test to verify the stability of the PD controller is the Nyquist Criterion. The code was written in Matlab as seen below.

PD Controller Nyquist Plot Matlab Script

```
f1 = figure;
k1_PD = 59.6063;
k2_PD = 0.67;
PI_num = [0.06*k2_PD 0.06*k1_PD];
PI_den= [0.05 (0.46+0.06*k2_PD) 0.06*k1_PD];
nyquist (tf(PI_num,PI_den))
title('Hunter Phillips Nyquist Plot for PD Controller')
xlim([-1.2 1.2])
grid
```

The resulting Nyquist plot can be observed below in *Figure 10*:

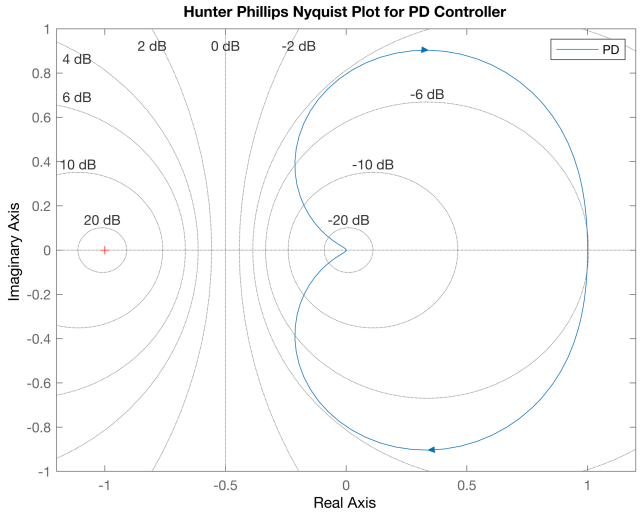

Figure 10: PD Controller Nyquist Plot

It can be seen in the resulting plot that -1 + 0j is not encircled by the plot, which means the PD controller system is stable according to the Nyquist criterion.

4.1.3 Bode Plot Technique

The third and final test to test the stability of the PD controller is the Bode plot technique. The code was composed in Matlab as seen below.

PD Controller Bode Plot Matlab Script

```
f2 = figure;
bode (tf(PI_num,PI_den)) % using previous vars defined in nyquist plot
title('Hunter Phillips Bode Plot for PD Controller')
grid
h1 = findobj(f2,'type','Axes');
legend(h1(1),'PD');
legend(h1(2),'PD');
```

The resulting Bode plot can be seen below in *Figure 11*:

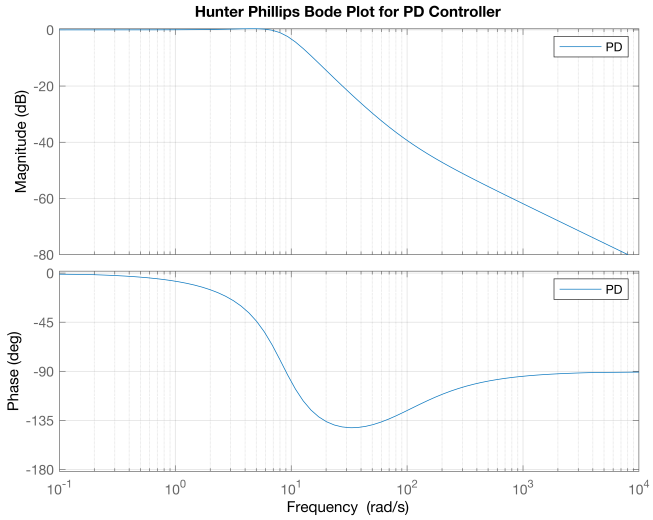

Figure 11: PD Controller Bode Plot

It can be observed that the gain is less than 1 dB across all phase in the frequency time steps. This shows that the system is stable according the Bode plot technique.

4.2 PID Controller

4.2.1 Routh-Hurwitz Criterion

The first analysis performed to make sure that stability is achieved by the Routh-Hurwitz Criterion being applied to the PID controller. The G(s) characteristic equation to use for this analysis is selected to be $0.05s^3 + 1.0936s^2 + 16.794s + 97.656$. The results of the criterion being applied can be seen in *Table 2*.

Table 2: Routh-Hurwitz Criterion of Stability for PID Controller

s^3	0.05	16.794
s^2	1.0936	97.656
s^1	12.33	0
s^0	97.656	0

As shown there is no sign change apparent in the results, which verifies that the PID controller is stable according to the Routh-Hurwitz criterion.

4.2.2 Nyquist Criterion

The second test to test the stability of the PID controller is the Nyquist Criterion. The code was programmed in Matlab as show below.

PID Controller Nyquist Plot Matlab Script

```
f3 = figure;
k1_PID = 279.9;
k2_PID = 10.56;
k3_PID = 1627.6;
PID_num = [0.06*k2_PID 0.06*k1_PID + 0.06*k3_PID];
PID_den = [0.05 (0.46+0.06*k2_PID) 0.06*k1_PID 0.06*k3_PID];
nyquist (tf(PID_num,PID_den))
title('Hunter Phillips Nyquist Plot for PID Controller')
xlim([-1.2 1.2])
grid
legend('PID')
```

The Nyquist plot produced can be seen in *Figure 12*:

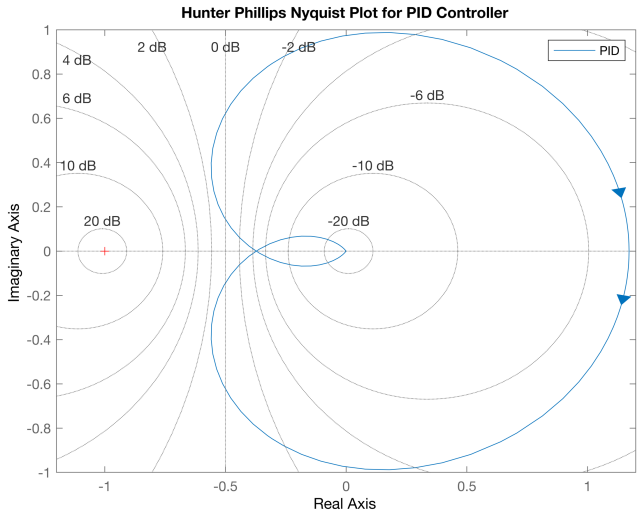

Figure 12: PID Controller Nyquist Plot

It can be observed in the plot that is pictured that -1 + 0j is not encircled by the plot, concluding that the PD controller system is stable according to the Nyquist criterion.

4.2.3 Bode Plot Technique

The final assessment to test the stability of the PID controller is the Bode plot technique. The code was created in Matlab as showcased below.

PID Controller Bode Plot Matlab Script

```
f4 = figure;
bode (tf(PID_num,PID_den)) % using previous vars
title('Hunter Phillips Bode Plot for PID Controller')
grid
h2 = findobj(f4,'type','Axes');
legend(h2(1),'PID');
legend(h2(2),'PID');
```

The final Bode plot can be seen below in *Figure 13*:

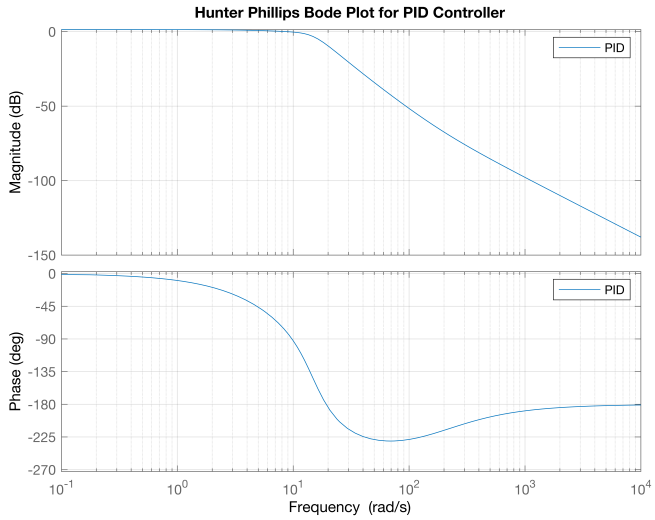

Figure 11: PID Controller Bode Plot

The gain is shown to be within the lower bounds of 0 dB which adheres to the guidelines in the Bode plot technique to satisfy that the system is stable. This final verification proves the stability of the PID controller under test.

5. Simulations

5.1 PD without Prefilter

The first simulation that was performed involved the PD controller without a prefilter designed in Section 3.1. The results from the simulation can be seen below in *Figures 12-16.*

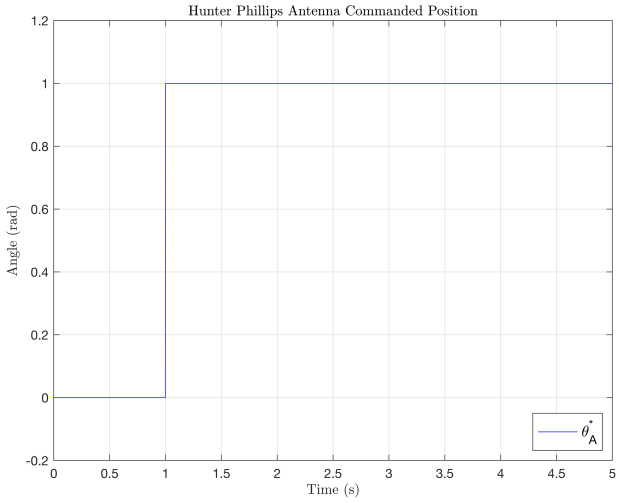

Figure 12: PD Plot of Θ_A^* vs Time

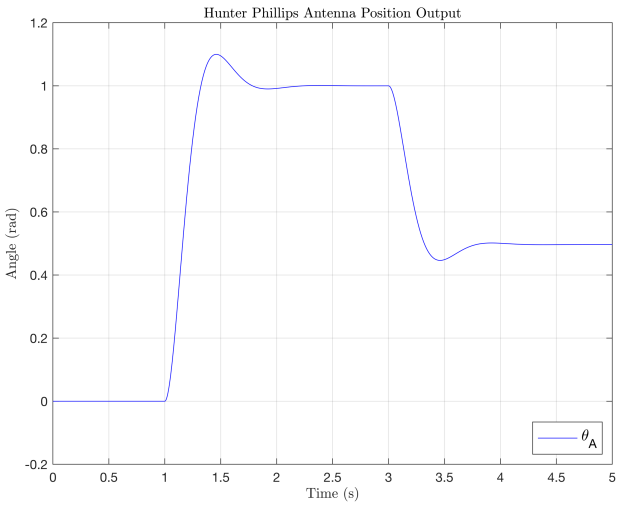

Figure 13: PD Plot of Θ_A vs Time

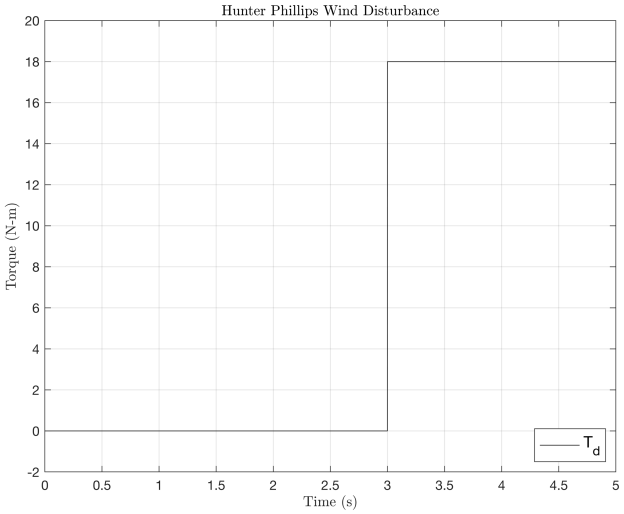

Figure 14: PD Plot of T_d vs Time

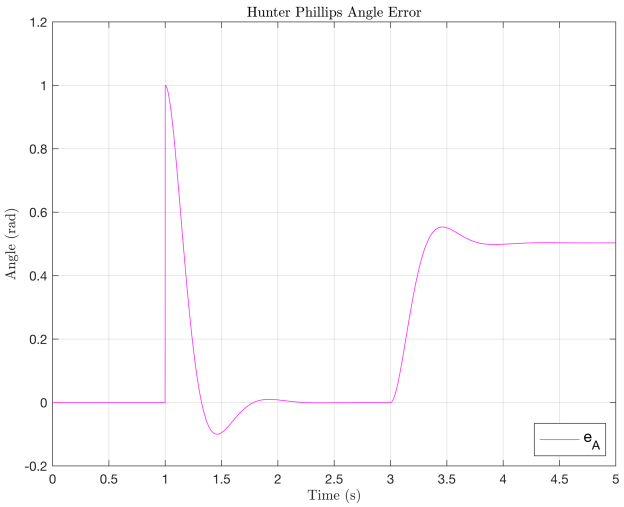

Figure 15: PD Plot of e_A vs Time

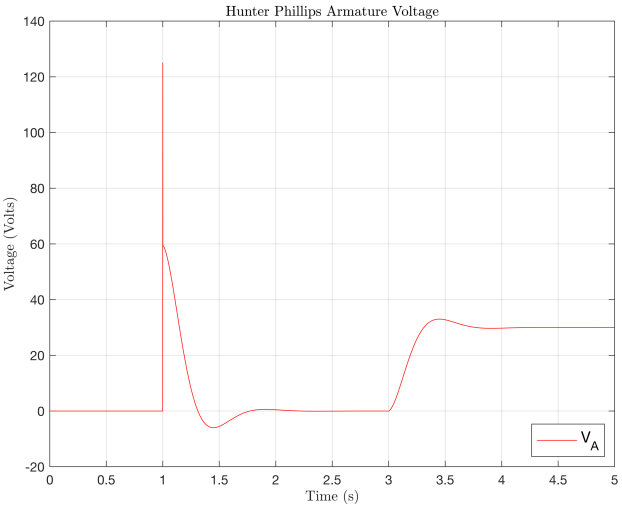

Figure 16: PD Plot of V_A vs Time

It can be observed that the steady state error value of 0.5 predicted in the error analysis proved to be correct with the PD controller unable to return to 0 radians. The main objective to compare these results with a prefilter applied to the model is now explored.

5.2 PD with Prefilter

The second simulation that was performed introduced the PD controller with a prefilter designed in Section 3.2. The plots obtained from the simulation can be seen below in *Figures 17-21.*

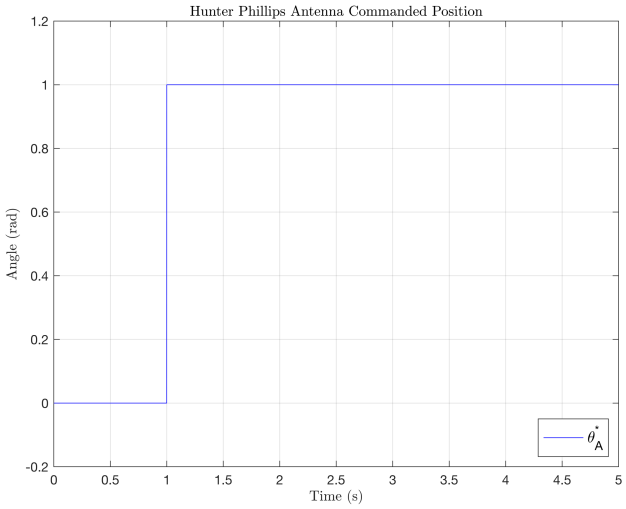

Figure 17: PD Plot of Θ_A^* vs Time

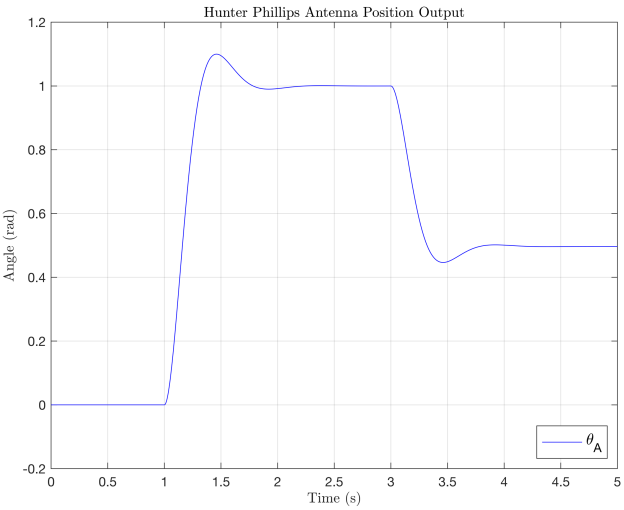

Figure 18: PD Plot of Θ_A vs Time

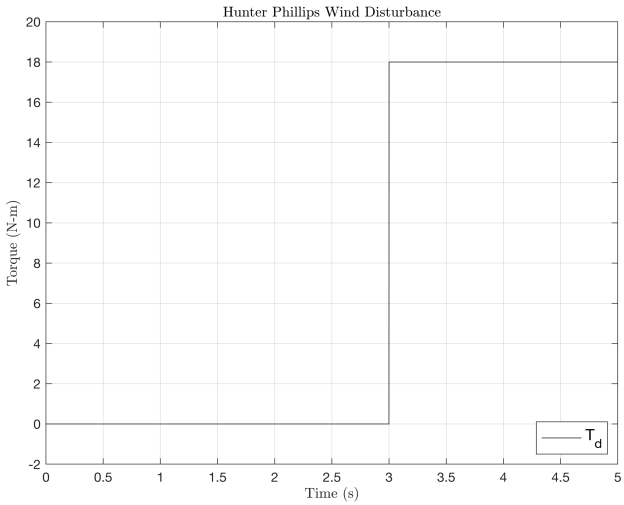

Figure 19: PD Plot of T_d vs Time

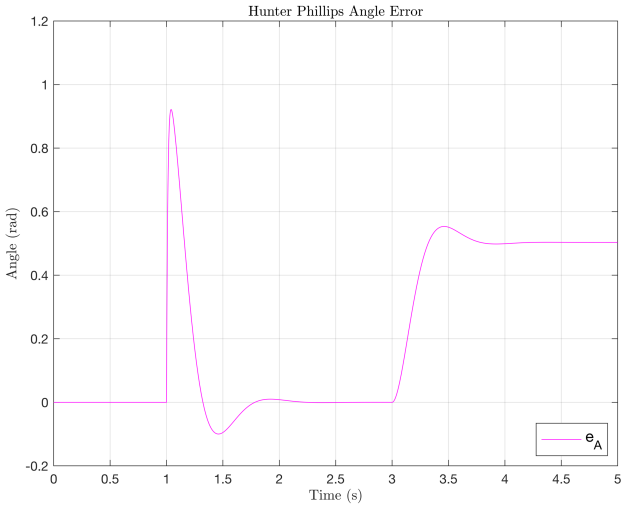

Figure 20: PD Plot of e_A vs Time

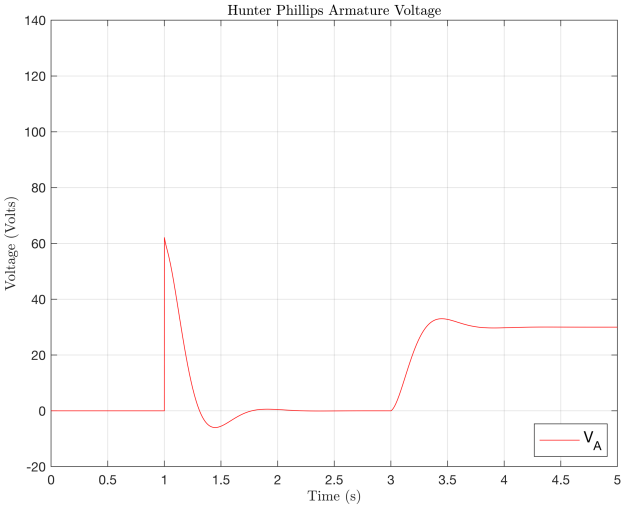

Figure 21: PD Plot of V_A vs Time

These results produce similar characteristic plots to the designs without a prefilter for the PD controller. However, it can be observed that there is less overshoot in the results with the prefilter which is the large advantage that a prefilter can give a system. This verifies that the prefilter is working correctly and validates the simulation test at hand.

5.3 PID without Prefilter

The third simulation that was ran used the PID controller without a prefilter designed in Section 3.3. The data was plotted with the simulation results as shown below in *Figures 22-26*.

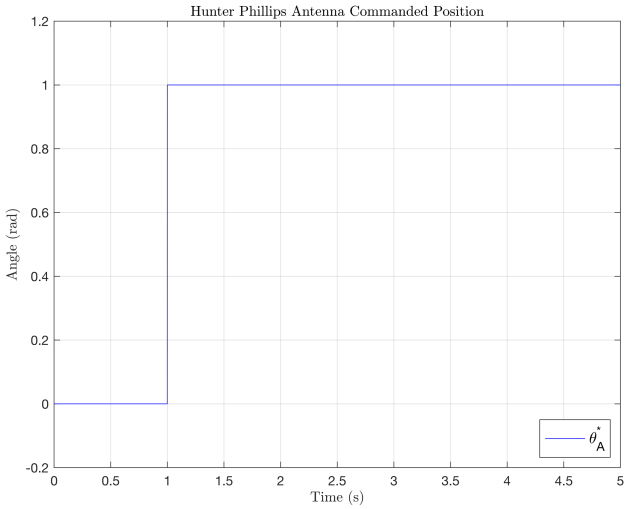

Figure 22: PID Plot of Θ_A^* vs Time

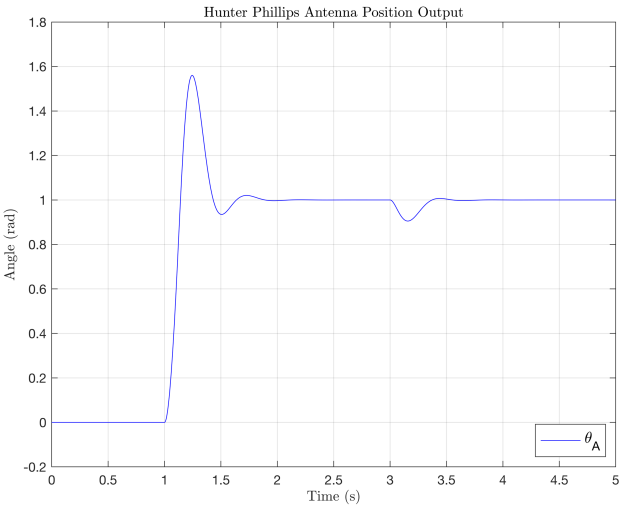

Figure 23: PID Plot of Θ_A vs Time

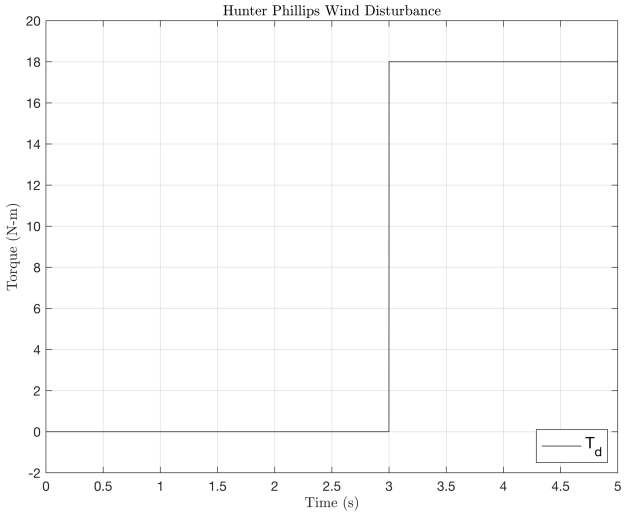

Figure 24: PID Plot of T_d vs Time

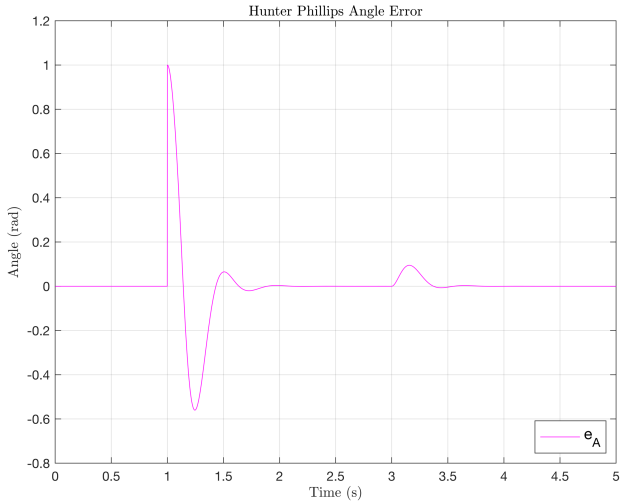

Figure 25: PID Plot of e_A vs Time

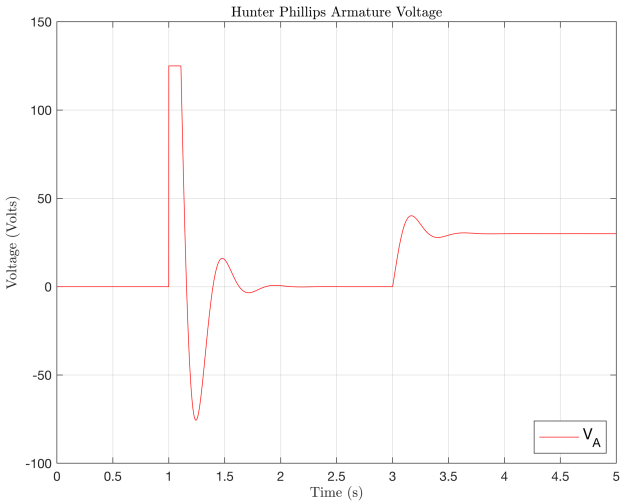

Figure 26: PID Plot of V_A vs Time

It can be noted that the PID controller is able to return the steady state error value to zero as expected from the error analysis. There is still significant overshoot which should be partially fixed by the prefilter implemented as the next component of the design project.

5.4 PID with Prefilter

The fourth and final simulation that was executed used the PID controller with a prefilter developed in Section 3.4. The data was plotted as seen in the simulation results as seen in *Figures 27-31*.

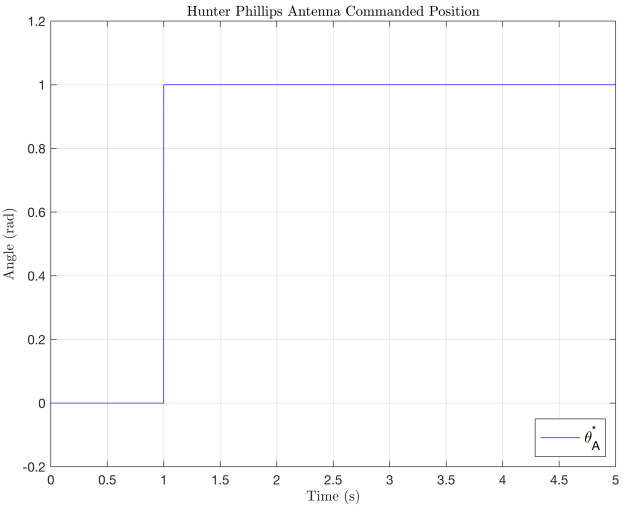

Figure 27: PID Plot of Θ_A^* vs Time

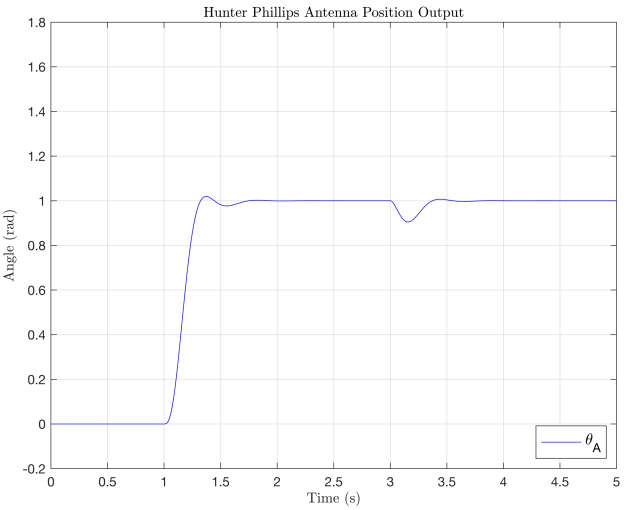

Figure 28: PID Plot of Θ_A vs Time

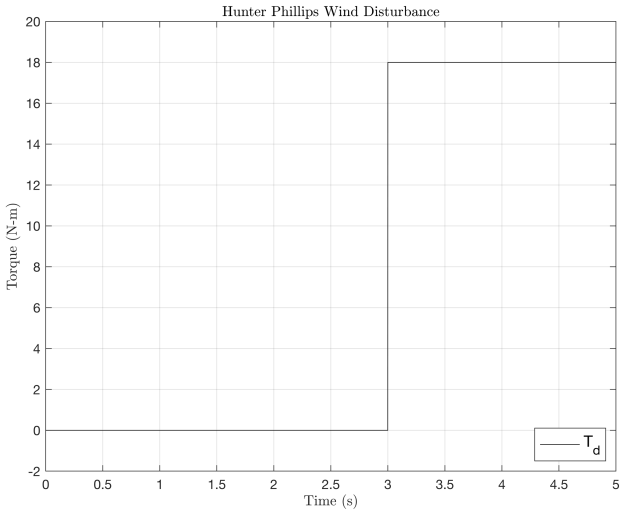

Figure 29: PID Plot of T_d vs Time

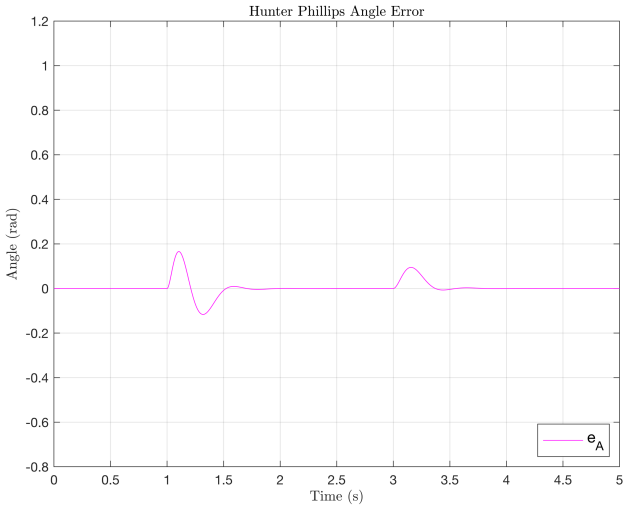

Figure 30: PID Plot of e_A vs Time

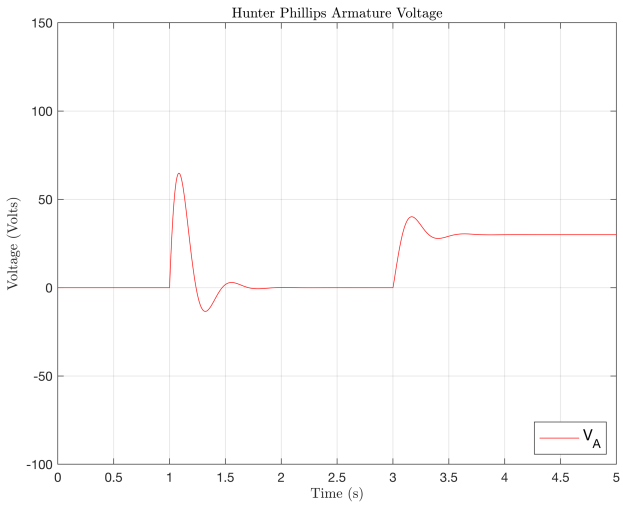

Figure 31: PID Plot of V_A vs Time

As expected the overshoot has been significantly reduced in the system response when the wind disturbance is applied. The prefilter has successfully done its job to a larger effect on the PID controller when compared to the PD controller, due to the PID controller's inherently large overshoot due to integrator windup.

5. Conclusion

Through this project, a successful simulation was developed with a wide gamut of

6. References

[1] Shtessel, Yuri. "Design Project." *Introduction to Control and Robotic Systems EE-386*, Spring 2019. The University of Alabama in Huntsville. Lectures.

[2] Dorf, R. C., & Bishop, R. H. (2016). Modern control systems (13th ed.). Boston: Pearson.