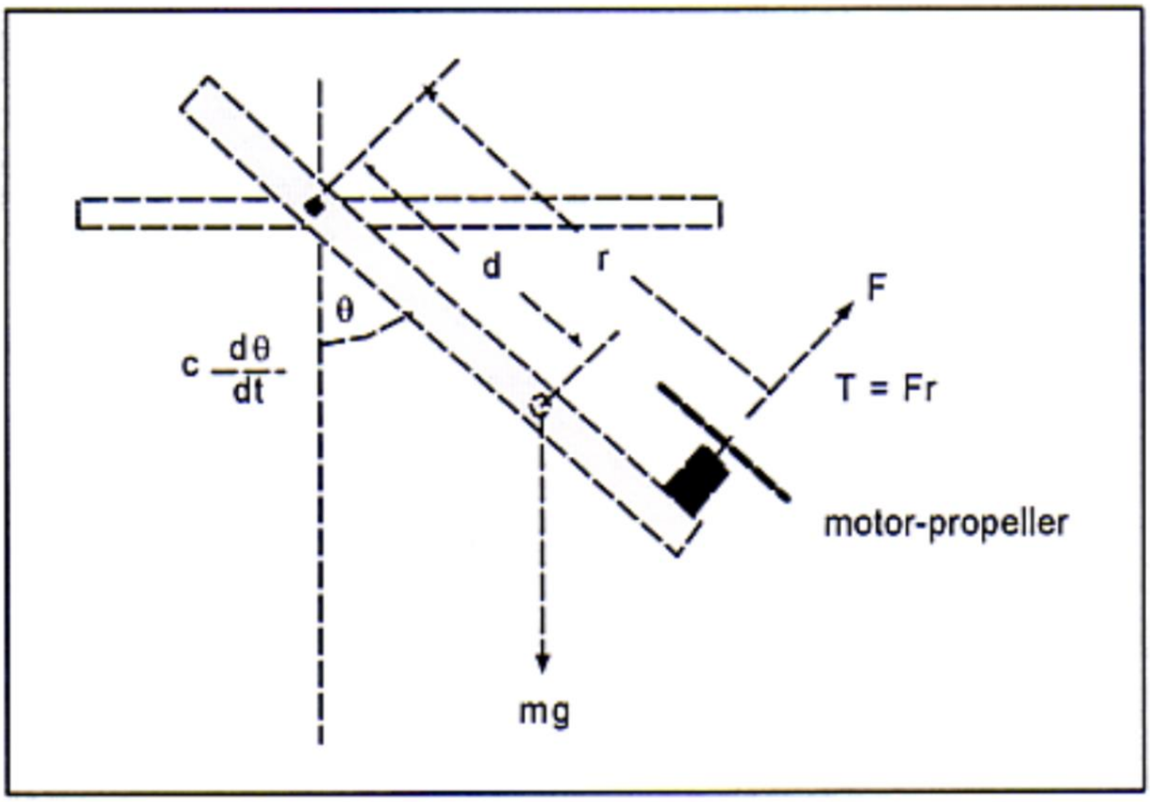
Andre Webber
300286354

EEEN315 Project Report 2:

Closed Loop Analysis of a Motorised Pendulum

Andre Webber-LaHatte



Abstract

This report analyses the closed loop response of a propeller driven pendulum based on simulation and mathematical models. Various tools and techniques are used to design a controller that will achieve a desired transient response. The controller designed through simulated models is applied to a physical model which is adjusted to achieve the desired response.

1. Introduction

This is the second report for EEEN315. This report expands on the first report, in which an open loop transfer function was derived for a propeller driven pendulum. Discussed will be various tools that can be applied to an open loop transfer function to understand how it will respond in a closed loop system. These tools will be used as the basis for compensator design. The compensators designed will be applied to a simulated model and adjusted accordingly to shift the response to have desired characteristics. The results of this exercise will be used to explore the impact of different compensators to a system in terms of stability. A controller will also be applied to a physical system based on observation made during simulation, achieving a settling time of less than 2 seconds, and minimal overshoot to a setpoint input of $20°$.

2. Background

2.1 Root Locus

A root locus is a graphical tool used to qualitatively describe a control system [1]. For some transfer function, its root locus shows the path that open loop poles will follow in a closed loop configuration, as proportional gain (K_p) is increased. By viewing the point at which a root locus intersects the imaginary axis (from the left half plane), the proportional gain that will drive a system to instability can be determined. This tool is particularly useful when analysing a system of order higher than 2.

System design through root locus involves applying compensators that add additional poles and zeros. Figure 1 shows an example of a root locus plot [1]. In this example, the desired transient characteristics are achieved by moving an open loop pole to the location of point B which is not on the root locus path. By adding certain compensators, the root pocus path can be shifted to intersect point B, thus there will be some level of proportional gain that moves the pole to the desired location.

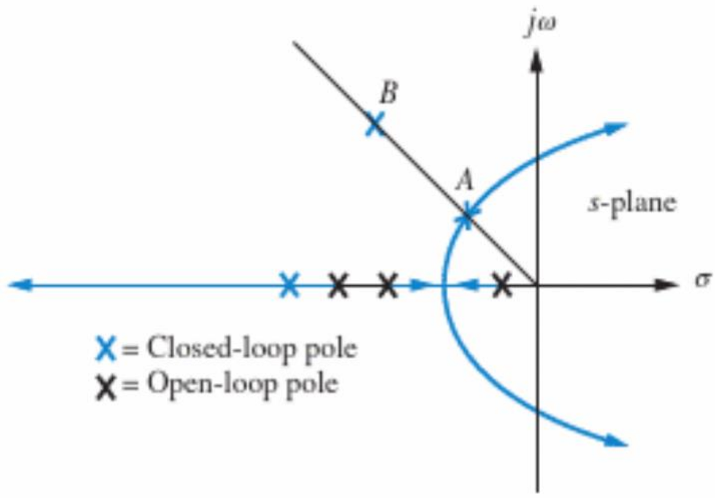


Figure 1: Example of a root locus plot [1].

2.2 Bode

Bode diagrams provide a graphical representation of a transfer function's frequency response [1]. The gain and phase at each frequency are plotted on 2 separate axes as a funtion of $\log \omega$. They are useful for modelling a transfer function from physical data. Finding the stability of non-linear systems. Using a logarithmic plot is useful as if the gain and phase frequency responses are known for each term in a transfer function, they can be plotted individually and graphically summed.

Compensator design can be done using bode plots by reading the gain and phase margin [1]. The gain margin is given by how far below 0dB the gain is at the frequency where phase crosses $-180°$ (phase crossover frequency). Similarly, the phase margin is how much the phase is above $-180°$ at the frequency where gain crosses 0dB. These values give information about the stability and damping ratio of a closed loop system. Different types of compensators can be added with gain and phase characteristics to manipulate the gain and phase margins such that they correspond with the desired transient response.

2.3 Ziegler Nicholl's

Ziegler Nicholl's is a method for tuning PID controllers [2]. The method starts with finding the proportional gain that will drive the system unstable. this can be read from the imaginary axis crossing on a root locus, or the gain margin from a bode plot. The gain (K_{MAX}) and frequency of oscillation (f_O) at this point are used to determine the values for the K_p, integral gain $(K_i), and$ derivative gain (K_d). Figure 2 shows a description of how to use this method in implementing a P, PI, and PID controller. K_{MAX} and f_O can also be read from a bode plot, by looking at the gain margin, and phase crossover frequency.

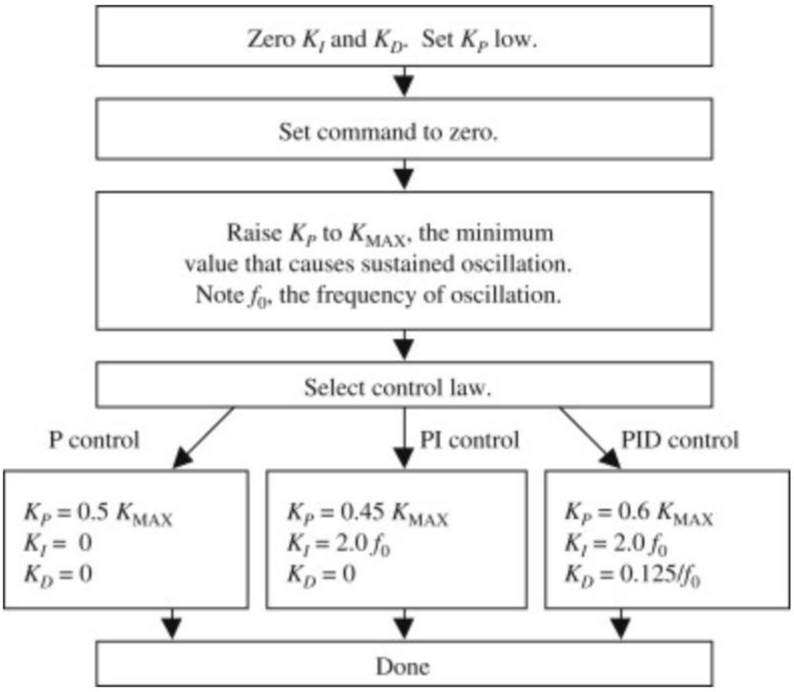


Figure 2: Ziegler Nicholl's PID tuning process [2].

3. Methodology

The main objective of this exercise is to analyse the closed loop response of a propeller driven pendulum. This analysis will provide the basis for tuning a PID controller to acheive desired step response characteristics. Equation 1 shows the open loop transfer function for this system. The tools described in section 2 will be applied to Eq. 1, to provide basis for compensator design. The effects of applying compensators will be observed, and adjustments made accordingly.

$$\frac{\Theta(s)}{V(s)} = \frac{1.9577}{0.0054s^4 + 0.0623s^3 + 0.4541s^2 + 2.5529s + 6.4052} \tag{1}$$

Due to health and safety considerations, analysis will first be done via simulation. These considerations allow for testing the impact of various compensators, without the concern of causing injury through unintentionally pushing the system into instability. Once the desired step response characteristics are achieved via simulation, the response of the physical system will be analysed. Once PID controllers have been tunes for each the simulated and physical version of the system, and result recorded, the Ziegler Nicholl's tuning method will be applied to provide a basis for comparison.

3.1 Calibration

Simulation of this system was done using Simulink. Setting up the simulation required some calibration. The simulation included a model of the pendulum based on data acquired in previous iterations of EEEN315 (ECEN315). An 8-bit value is used to indicate the duty cycle of the PWM being applied to the propeller motor. The model pendulum's potentiometer output to a simulated ADC which output a 10-bit value. Calibration involved setting up gain blocks so both the input and output could be specified as an angle. This was done using a simulated 10V power supply, as well as using the model pendulum's "angle" output, which is used to simulate looking at the pendulum. When closing the loop, the output angle is subtracted from the input angle, with this error being compensated and fed through the input gain block. Prior to closing the loop, the effectiveness of the simulation as an open loop controller was investigated using a 20° setpoint, and varying the supply voltage.

3.2 Proportional Compensator

There is a logical starting point when analysing a closed loop system. This starting point involves viewing how far K_p can be increased before pushing a system unstable. Increases to proportional gain will reduce the steady state error of a system, however it will never decrease it to zero. In the case of the propeller driven pendulum, the steady state will be at the point where the error fed back into the input will result in no change to the voltage driving the propeller. Figure 3 shows the root locus of the transfer function in Eq. 1, as discussed in section 2, the level of gain that push the system into instability is found by looking at the imaginary axis crossing point.

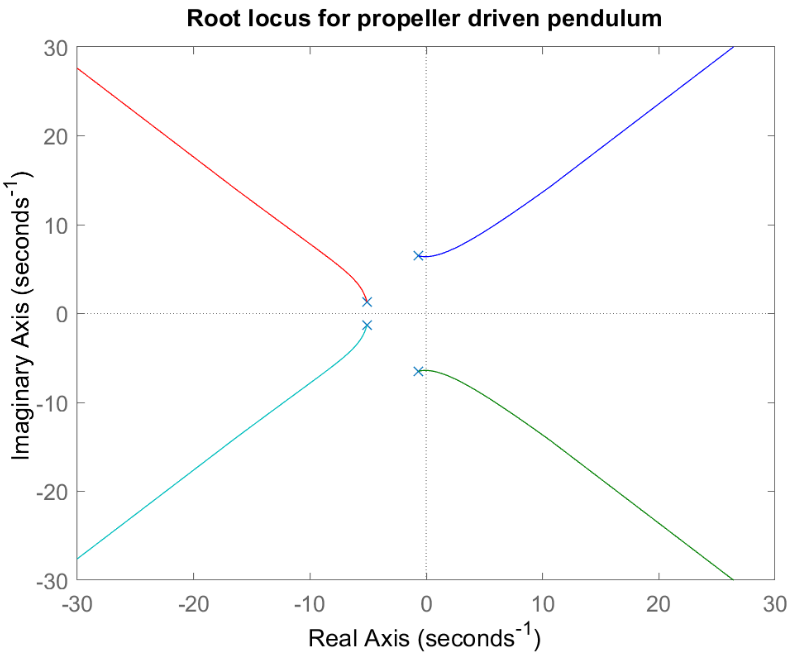


Figure 3: Root locus for propeller driven pendulum

Equation 2 shows how the closed loop transfer function can be found when proportional gain is applied in a closed loop configuration. The poles are a transfer function are located at points where the denominator is equal to zero. When the denominator of Eq. 2 is rearranged in terms of K_p, the maximum value for K_p is found by inputting the value of s at the root locus imaginary crossing point.

$$G_f(s) = \frac{K_p G(s)}{1 + K_p G(s)} \tag{2}$$

3.3 PI compensator

Once the appropriate level of proportional gain has been established, the steady state error to a step input can be reduced to zero by applying a compensator that adds a pole at $s = 0$. This is done by adding a branch in parallel with K_p such as in figure 4. The $\frac{1}{s}$ shown in this figure is an integrator in the Laplace domain. When such a branch is added, the equation for the compensator being applied to the plant is shown in Eq. 3.

$$C(s) = \frac{K_p s + K_i}{s} = \frac{K_p\left(s + \frac{K_i}{K_p}\right)}{s} \tag{3}$$

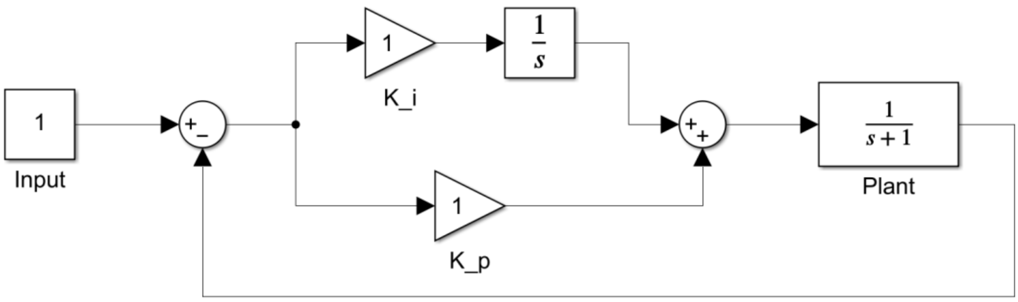


Figure 4: PI compensator block diagram

Figure 5 is the bode plot for the transfer function in Eq. 1. Note that there is no unity gain crossing. The addition of an integrator branch will add infinite gain at low frequencies, and thus introduce a unity gain crossing. Compensator design should be done such that phase margin introduced is no lower than $60°$, and impact the gain margin as little as possible. Figure 6 shows a bode plot for a PI compensator. The point where the gain changes from decreasing at $\frac{20dB}{decade}$, is given by the ratio between K_i and K_p as shown in Eq. 4.

$$\omega_b = \frac{K_i}{K_p} \tag{4}$$

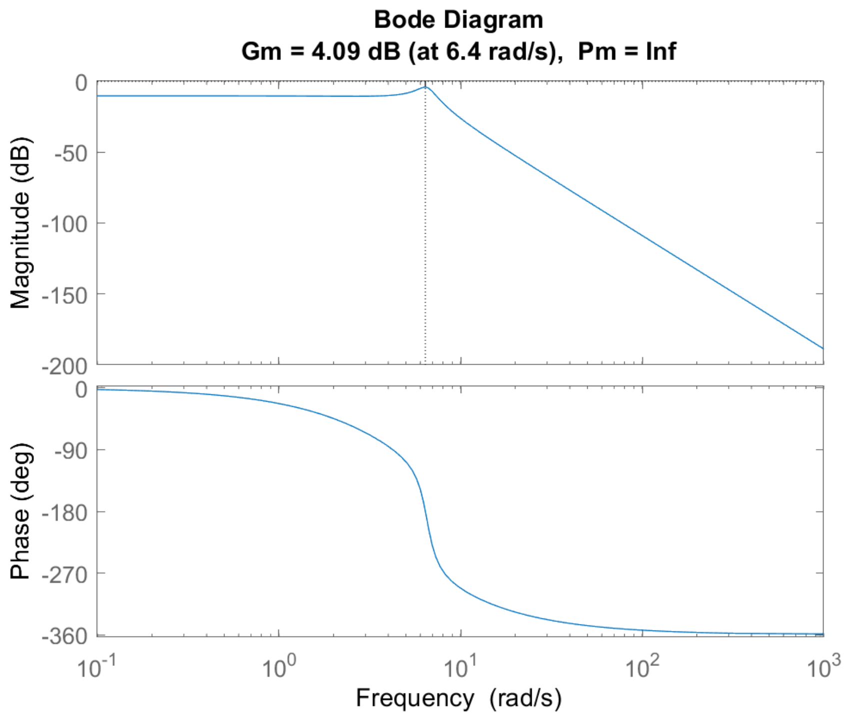


Figure 5: Bode plot for propeller driven pendulum

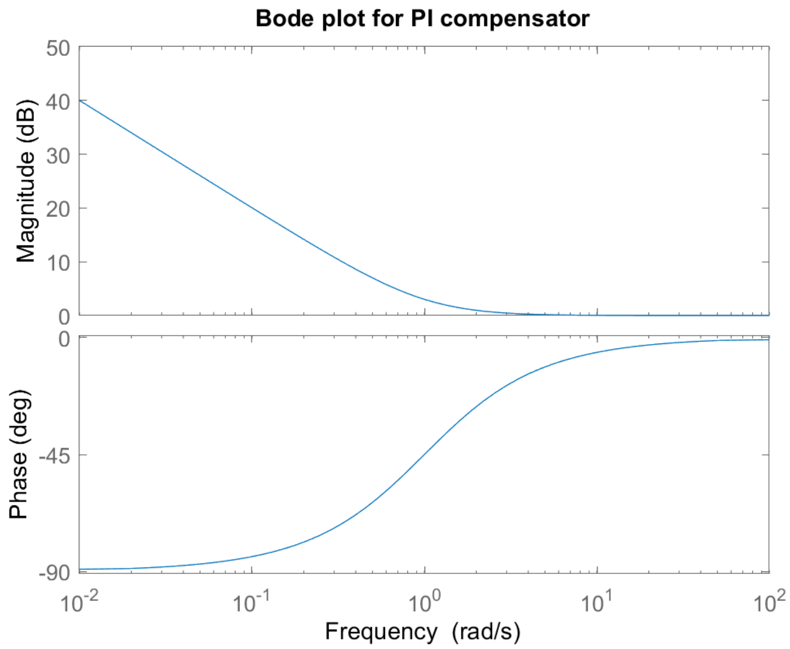


Figure 6: Bode plot for a PI compensator

By setting $K_i = 1$, ω_b can be set by varying the value of K_p. Bode plots will be used to determine the appropriate ω_b, which will be applied to the simulated model, and adjusted accordingly to achieved as close to the desired step response as is possible. This can be achieved by applying the appropriate proportional gain found by applying methods described in section 3.1 prior to each branch as in figure 7.

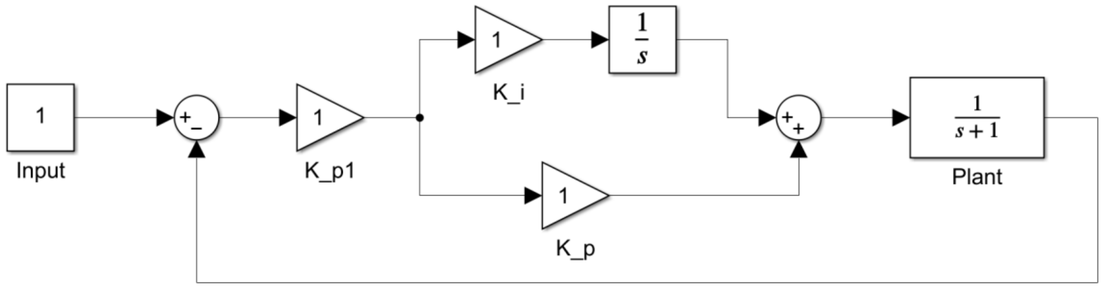


Figure 7: Gain applied to PI compensator

3.4 PD Compensator

A PD compensator is like a PI compensator, however, uses a derivative as opposed to an integral. Figure 8 shows a bode plot for a PD compensator. Observe that the phase moves from 0° to 90° around ω_b. Equation 5 shows the function for a PD compensator. In this case ω_b is determined by $\frac{K_p}{K_d}$. By designing a PD compensator that applied 0dB gain prior to ω_b, and setting ω_b higher than the unity gain crossing, these compensators can be used to increase the phase margin. As can be seen in figure 3, there is no unity gain crossing for the open loop transfer function shown in Eq. 1. For this reason, PD compensation will be implemented by adding a derivative branch to a PI compensator to counteract any potentially instability introduced, and to further supress oscillations to reduce overshoot.

$$C(s) = K_d s + K_p = K_d \left(s + \frac{K_p}{K_d} \right) \tag{5}$$

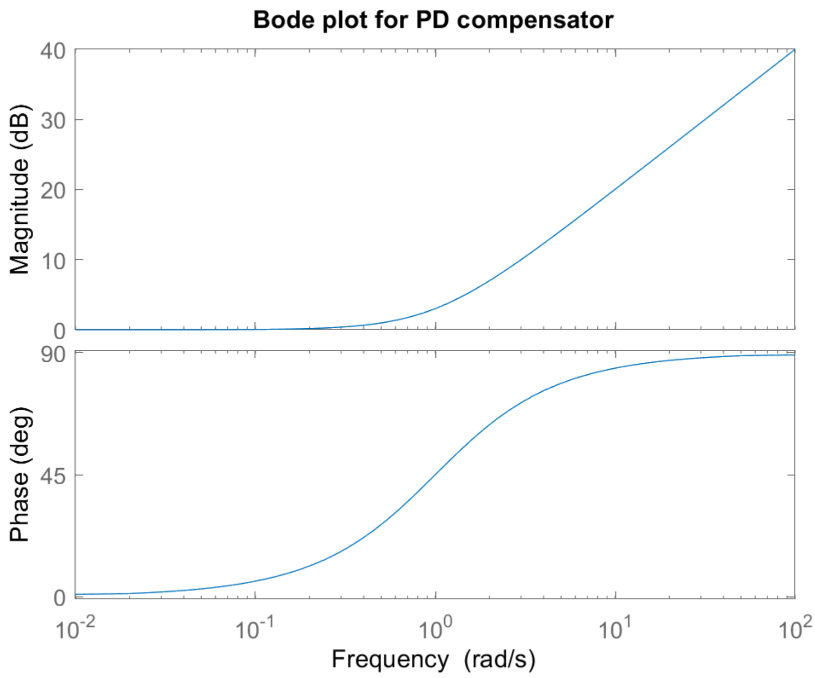


Figure 8: Bode plot for a PD compensator

4. Results

4.1 Open loop

After calibration, the simulation appeared to be an effective open loop controller. For a 10V supply, the output settled on the 20° setpoint with no error. However, if the supply voltage was changed, the output did not settle on 20°. Figures 9, 10, and 11 illustrate this. By closing the loop, variances in supply can be compensated with appropriate controllers that will drive the output to settle at 0 error.

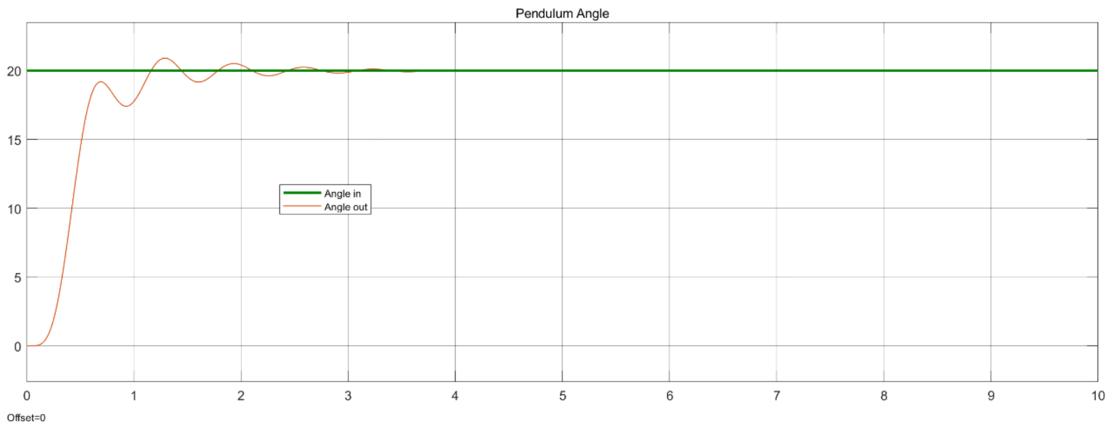


Figure 9: Open loop response, 10V supply

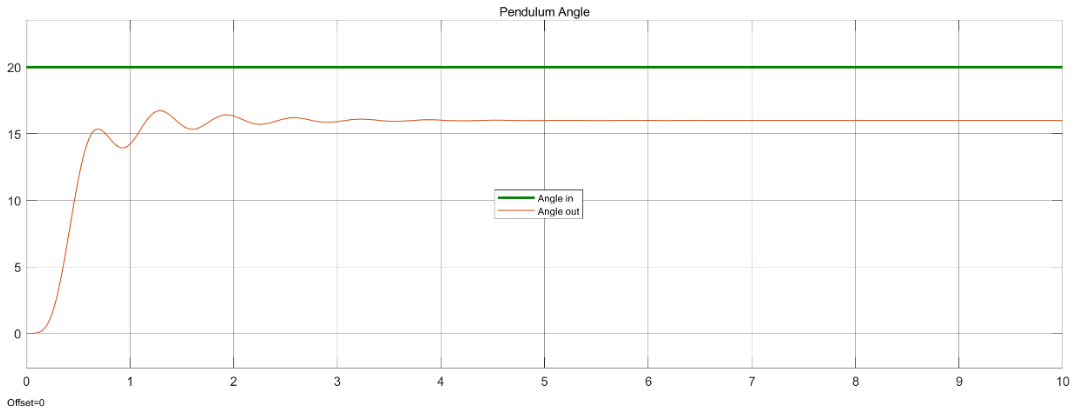


Figure 10: Open loop response, 8V supply

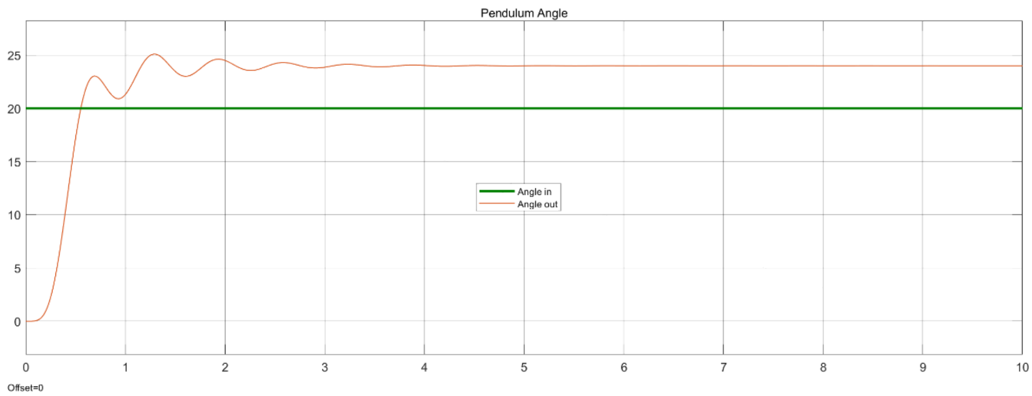


Figure 11: Open loop response, 12V supply

4.2 PI controller

Figure 12 shows the closed loop, unity gain response of this system, with a setpoint of $20°$. See in this figure, the steady state settles on $10°$ giving a 50% steady state error, settling in approximately 15 seconds. From the root locus shown in figure 3, the level of K_p that will drive the system unstable is approximately 1.6. Figure 13 shows the response for the same setpoint as in figure 12. This level of K_p drove the system unstable. From this point the value of K_p was reduced until the response of the system was no longer unstable at 1.25. Figure 14 shows the response for this level of gain, with a $20°$ setpoint. Observe that the system appears to be settling on approximately $12°$, however, is taking over 3 minutes to settle.

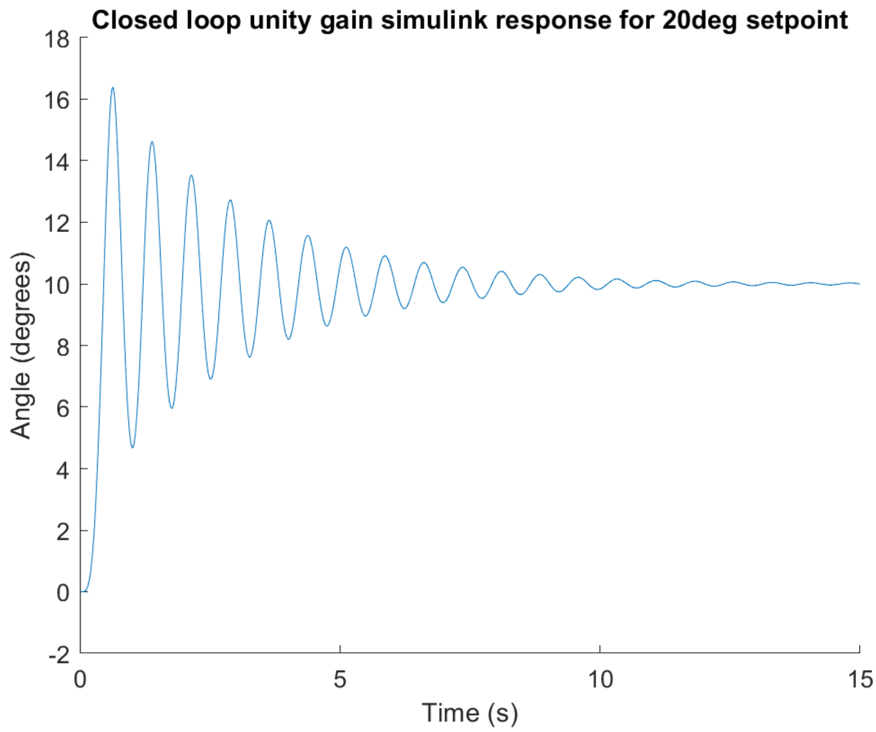


Figure 12: Proportional compensator response, K=1

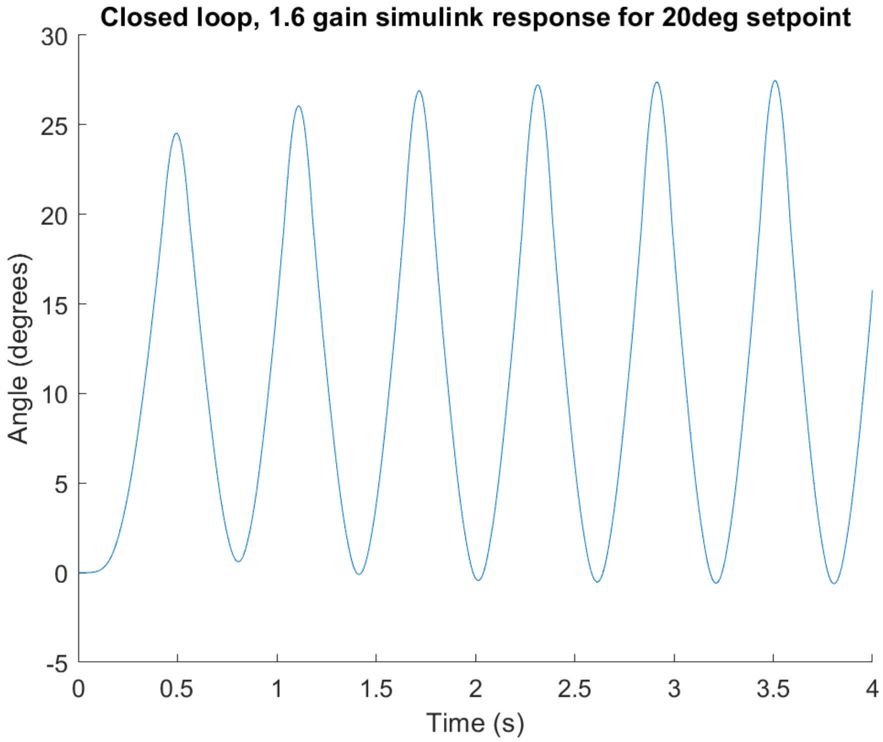


Figure 13: Proportional compensator response, K=1.6

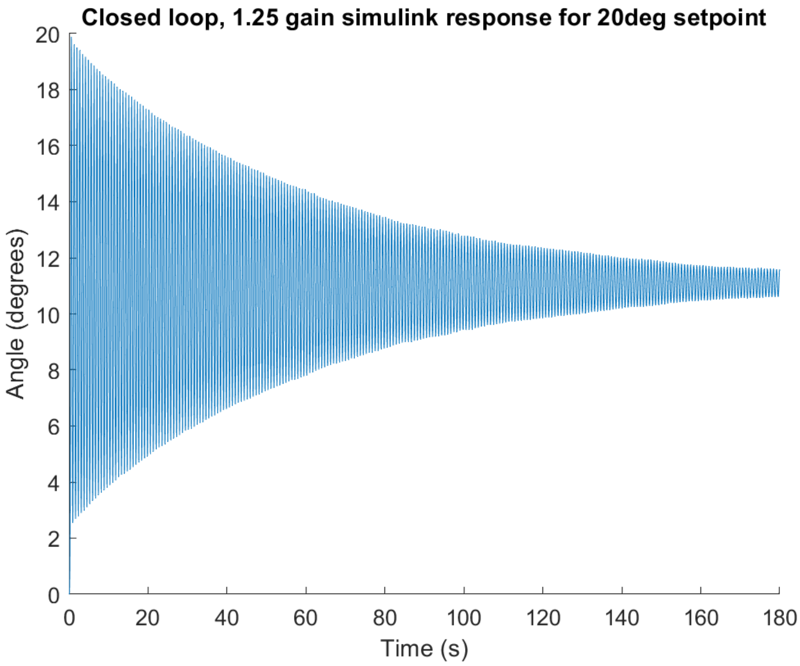


Figure 14: Proportional compensator response, K=1.25

The result of increasing K_p as far as possible without pushing the system to instability was a minimal decrease in steady state error, with a significant increase in settling time. This is not a good trade-off, and will result in adding complexity to PI and PD compensator design. For this reason, a value of $K_p = 1$ will be used when designing the PI compensator.

Refer to figure 5 observe a gain margin of 4.09dB at 6.4rad/s. To ensure no gain was added at this frequency, with minimal impact on phase, the initial PI compensator was designed with $\omega_b = 1rad/s$ by setting $K_p = 1$. Figure 15 shows the bode plot with this compensator applied. Not the gain margin is mostly unchanged, with a phase margin of $100°$ introduced. This suggests the step response will be stable, with minimal oscillation. Figure 16 shows the simulated response to this PI compensator. Observe minimal overshoot, however this response is quite underdamped, taking approximately 10 seconds to settle.

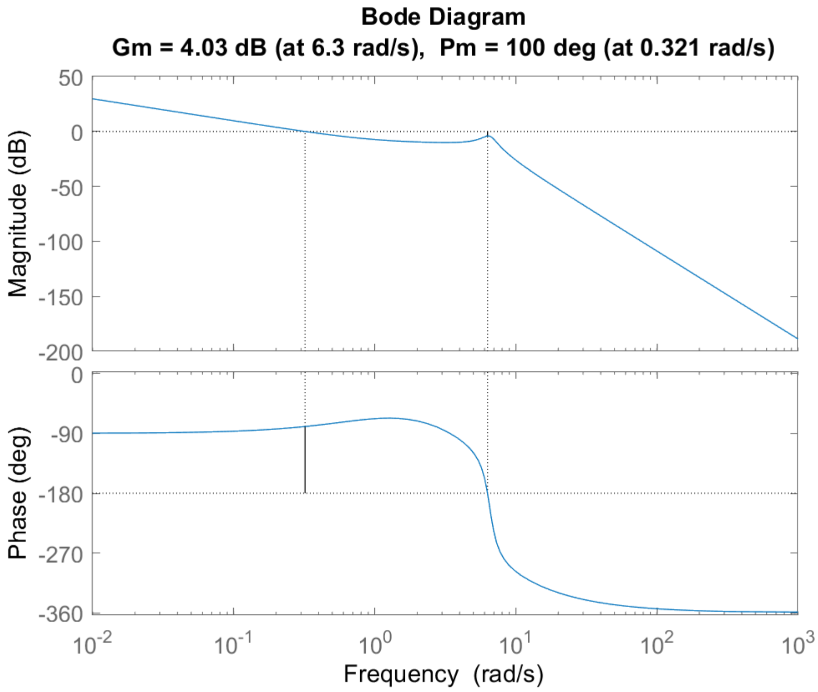


Figure 15: Bode plot with PI compensator applied to pendulum $\omega_b = 1rad/s$

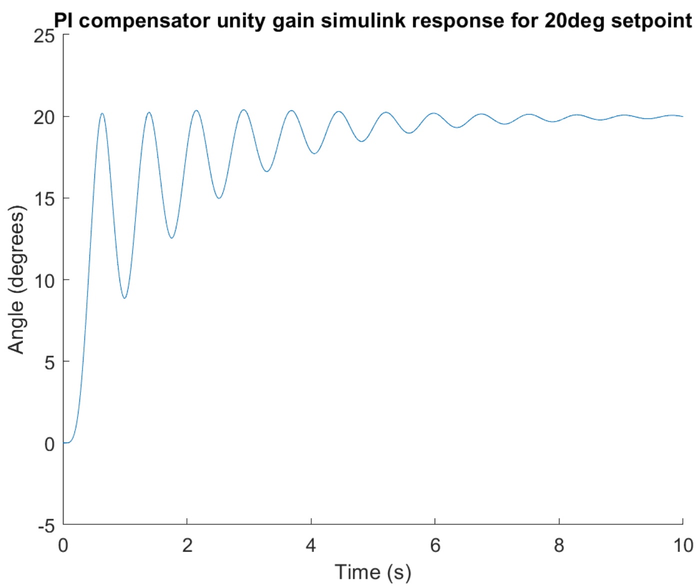


Figure 16: Response with PI compensator applied $\omega_b = 1rad/s$

Setting $K_p = 0.15625$ placed ω_b at 6.4rad/s which gave a very good step response with almost no overshoot, settling in approximately 3 seconds, seen in ***figure 17***. The values of K_i

and K_p were increased using the method described in section 3.2 to keep $\omega_b = 6.4rad/s$ until a settling time of approximately 2 seconds was achieved, this had minimal overshoot which was deemed acceptable. This ended with $K_i = 1.3$ and $K_p = 0.203125$, ***figure 18*** shows the response with the compensator applied. As the PI controller alone was able to achieve such a fast and stable response, no derivative branch was needed.

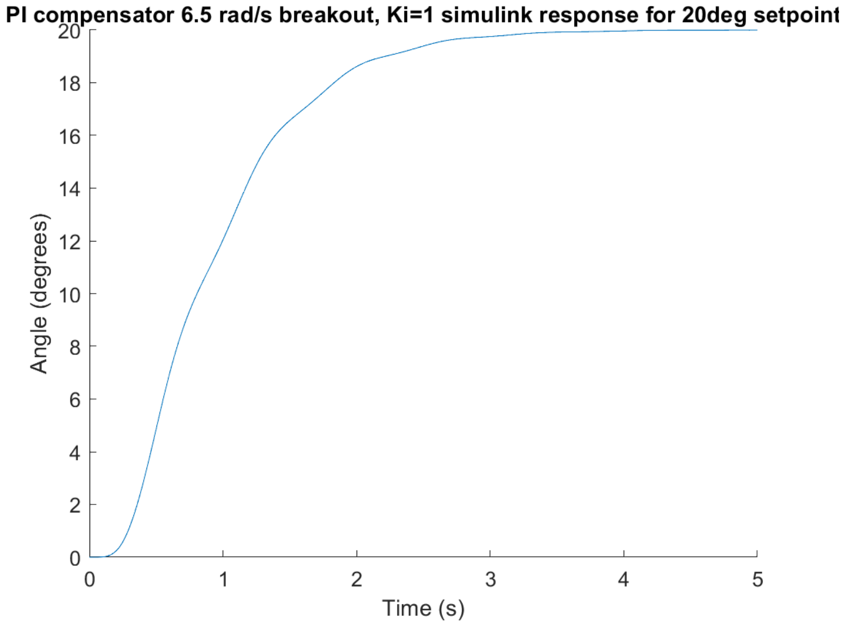


Figure 17: Response with PI compensator applied, $\omega_b = \frac{6.4rad}{s}\ K_i = 1$

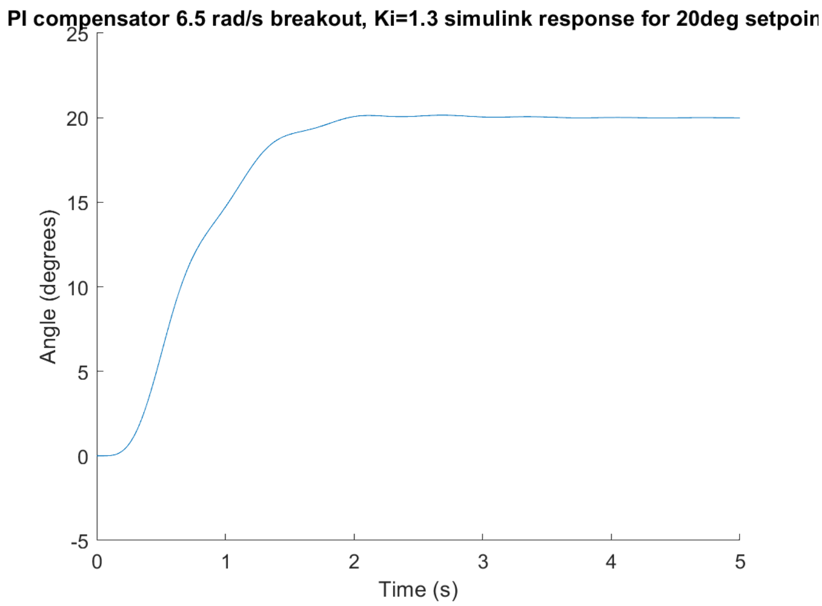


Figure 18: Response with PI compensator applied, $\omega_b = \frac{6.4rad}{s}\ K_i = 1.3$

4.3 Ziegler Nicholl's PID

Using the tuning method describe in section 2.3, both a PI and PID controller were designed. Table 1 shows the gain values for each controller, with ***figures 19 and 20*** showing the response with this tuning.

Table 1: Gain values using Ziegler Nicholl's method

	K_p	K_i	K_d
PI	0.72	3.334	0
PI	0.96	3.334	0.075

The responses from both the PI and PID controller using this method have much more overshoot and oscillation than when using the gain values discussed in section 4.2.

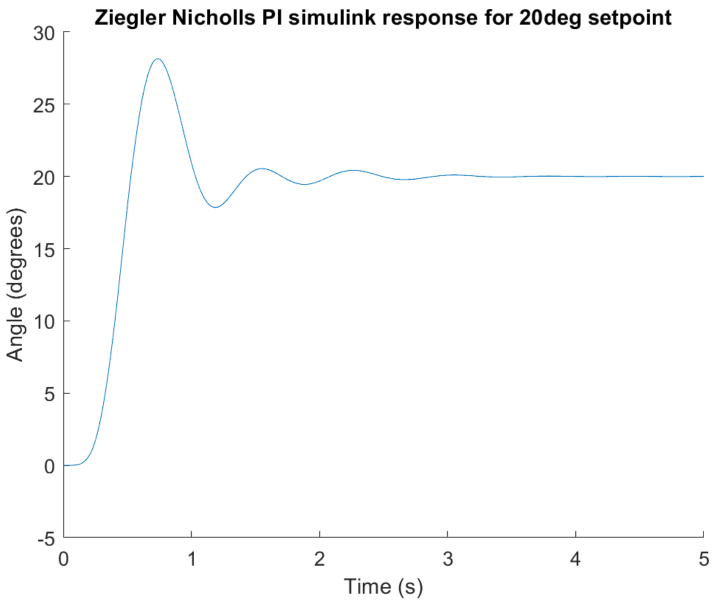


Figure 19: Ziegler Nicholl's PI

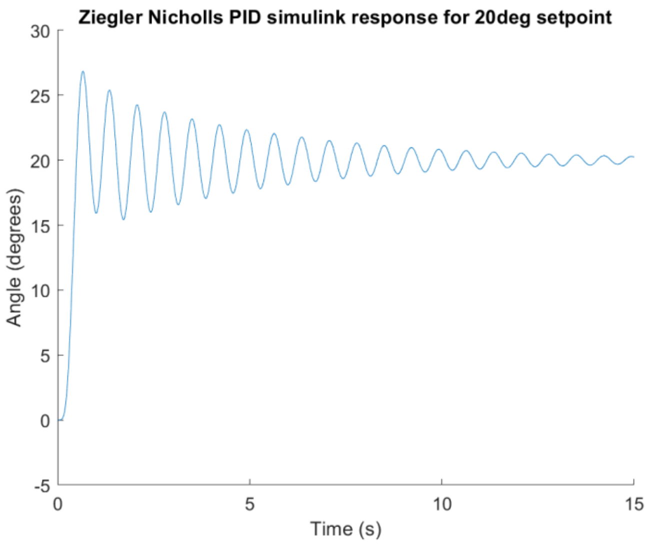


Figure 20: Ziegler Nicholl's PID

4.4 Physical Pendulum

Upon setting up the physical system, using the gain values calculated through the simulated model was a good starting point for tuning the PID controller. When applying a step input of a $20°$ setpoint the system had a rise time below 2 seconds. There was a lot more overshoot present in the physical system than the simulated version, taking approximately 5 seconds to settle. Applying a similar method to tuning the simulated pendulum, the PI compensator was adjusted to have a breakout point of 1rad/s with a proportional gain of 1. This had the impact of reducing thew overshoot, however had an approximately 3 second rise time. The integral gain was raised to raise ω_b until the rise time fell below 2s. $\omega_b = 1.3rad/s$ returned to a rise time less than 2 second, however, had some overshoot which was minimised by applying proportional gain before the PI compensator, such as in figure 7 of 0.6. The result of this was a rise time of approximately 2 seconds with minimal overshoot.

A disturbance was applied to the pendulum by pushing on it. The results observed for the above value of gain showed the system has retained stability as the pendulum would return to its setpoint no matter how hard it was pushed. The response to a disturbance resulted in a relatively large overshoot when returning to the setpoint which can be mitigated by further reducing proportional gain. This however, significantly increased the rise time, so was not a great trade-off.

5. Conclusion

Observing an analysing the closed loop response of a propeller driven pendulum has resulted in the application of a tuned PI controller. They key observation made from this analysis have been the impact of varying the supply voltage in an open loop configuration. While the calibrated open loop model was able to accurately settle on the setpoint, variances in the supply voltage resulted in the presence of error. Applying PID compensators though negative feedback deals with these constraints by pushing the output to settle at zero error. Focus must be made on stability so any variance in supply voltage impacts only the transient characteristics, without driving a system unstable.

The controller designed using tools described in section 2 is a PI controller, rather than a PID controller. This was able to achieve a settling time less than 2 seconds, with minimal overshoot. Applying a derivative branch in the physical system may have helped reduce overshoot when the system responds to a disturbance.

The process of modelling the behaviours of a propeller driven pendulum has given me insight into control systems engineering, and the importance of using closed loop controllers. The application of graphical tools when designing compensators has reinforced the theory taught in lectures, and allowed me to gain an understanding of the impact caused by different types of compensators.

6. References

[1] N. S. Nise, Control Systems Engineering, 8th ed., Hoboken, New Jersey: Wiley, 2019.

[2] G. Ellis, "Chapter 6 - Four Types of Controller," in Control System Design Guide: Using Your Computer to Understand and Diagnose Feedback Controllers, 4th ed., G. Ellis, Ed. Oxford, United Kingdom: Butterworth-Heinemann, 2012, pp. 111-113.