RYERSON UNIVERSITY

Faculty of Engineering, Architecture and Science

Department of Electrical & Computer Engineering
Program: Biomedical Engineering

Course Number	**BME 639**
Course Title	Control Systems and Bio-robotics
Semester/Year	Winter 2020
Instructor	Dr. Saba Sedghizadeh

Lab/Tutorial Report NO.	3

Report Title	Introduction to PI, PD and PID Controllers

Submission Date	April 10, 2020
Due Date	April 11, 2020

Name	Student ID	Signature*
Kushal Shah	xxxx43903	K.S.
[redacted]	[redacted]	[redacted]

(Note: remove the first 4 digits from your student ID)

**By signing above you attest that you have contributed to this submission and confirm that all work you have contributed to this submission is your own work. Any suspicion of copying or plagiarism in this work <u>will result in an investigation of Academic Misconduct and may result in a "0" on the work,</u> an "F" in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at: http://www.ryerson.ca/senate/policies/pol60.pdf.*

<u>Part 1:</u>

***Note: The closed loop transfer function in section A and B of Part One have been denoted by "transf" variable in the MATLAB Code**

A1:

As we know in a PI controller, the pole is always located at the origin therefore changing the kp and pi will not affect the pole location. However, zero depends on the Ti value thus zero located at -1/Ti value. Therefore increasing the Ti will make the zero closer to the origin, and decreasing the Ti will shift the zero to the higher frequencies and move away from the origin.

A2:

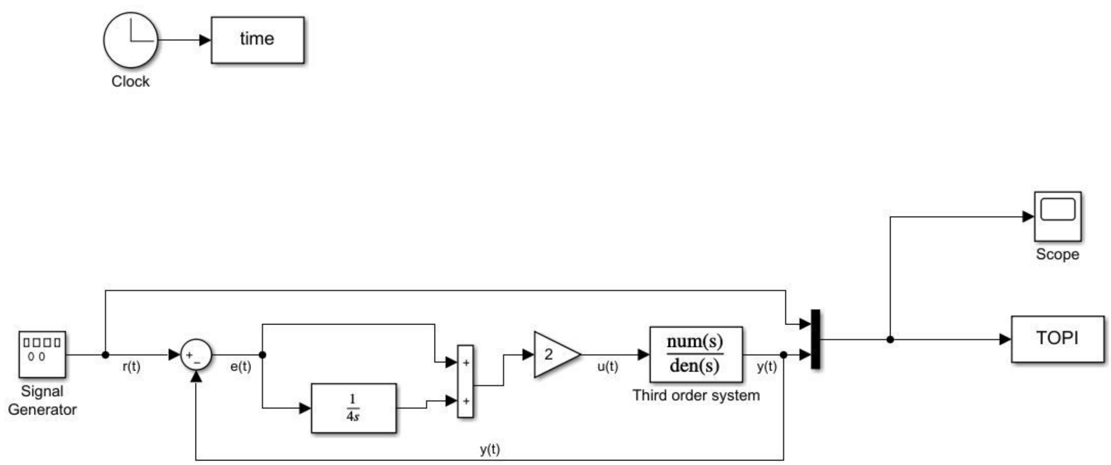

Figure 1. Simulink print of the closed loop system

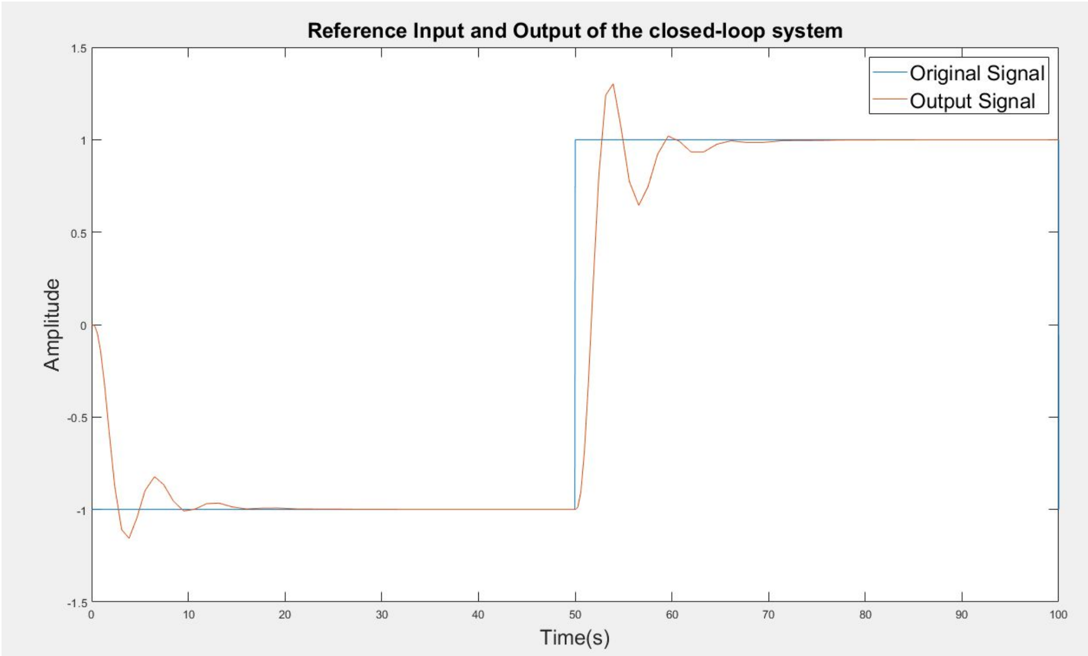

Figure 2. Reference input and output plot of the closed loop system. The blue line represents the original signal while the orange line represents the output signal.

A3:

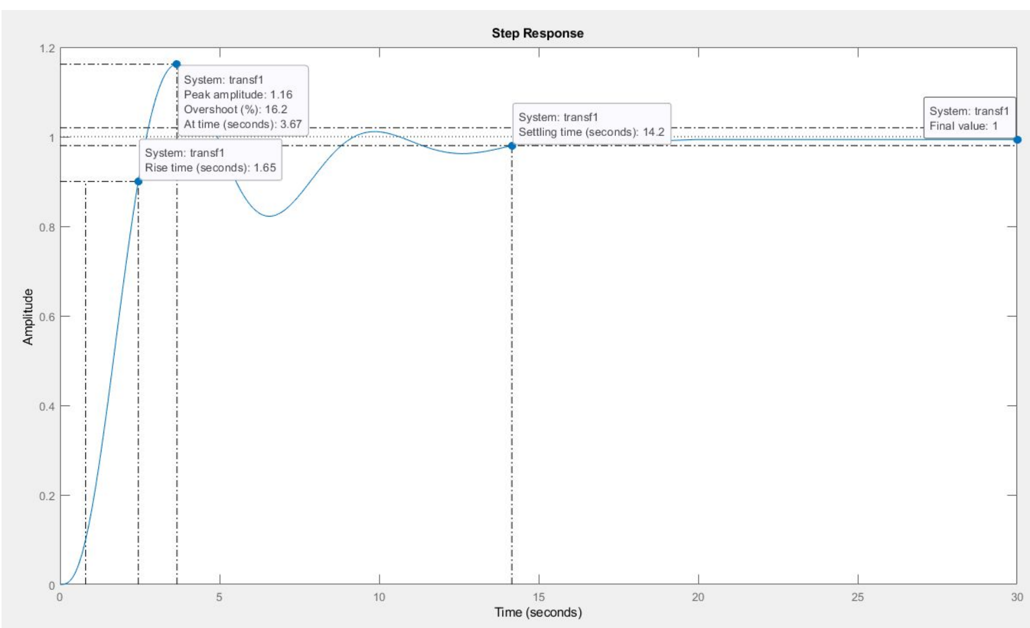

Figure 3. Plot of the closed loop transfer function PI controller with Ti = 4, Kp = 2. The x axis here has been extended to 30 seconds to get a clear view of steady state error of 1.

Command Window

```
>> kp = 2; Ti = 4;
>> num = kp*[Ti 1];
>> den = [Ti 0];
>> PI1 = tf(num,den)

PI1 =

  8 s + 2
  -------
    4 s

Continuous-time transfer function.

>> num = [1];
>> den = [1 3 3 1];
>> sys1 = tf(num, den)

sys1 =

            1
  ---------------------
  s^3 + 3 s^2 + 3 s + 1

Continuous-time transfer function.

>> transf1 = feedback(PI1*sys1,1)

transf1 =

               8 s + 2
  ----------------------------------
  4 s^4 + 12 s^3 + 12 s^2 + 12 s + 2

Continuous-time transfer function.

>> figure; stepplot(transf1)
```

Figure 4. Closed loop transfer function of the system and Code for plot of the PI controller with Ti = 4, Kp = 2

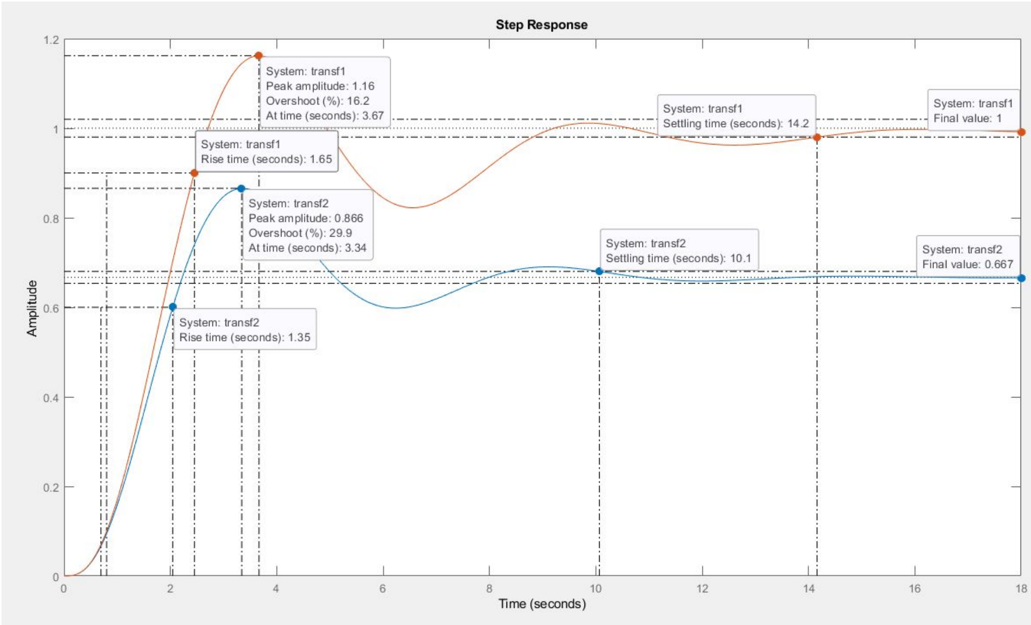

Figure 5. Plot of the closed loop system with the PI controller and with the P controller. The orange plot represents the PI controller and the blue plot represents the P controller.

```
>> kp = 2;
>> transf2 = feedback(kp*sys1,1)

transf2 =

          2
  ---------------------
  s^3 + 3 s^2 + 3 s + 3

Continuous-time transfer function.

>> figure; stepplot(transf2,transf1)
```

Figure 6. Extended code for the step plot of closed loop transfer function of P controller and PI controller.

Table 1.

	Rise time (tr)	%Max. O.S.	Settling time (ts)	Steady state error (ess)
PI Controller Ti = 4, Kp = 2	1.65	16.2	14.2	0
P Controller Kp = 2	1.35	29.9	10.1	0.333

From the above table, it is evident that the rise time is almost the same and adding the PI controller doesn't change the speed of the system. However it enhances the steady state portion of the system by eliminating the steady state to 0. Adding a PI controller also increases the settling time of the system, and decreases the % Max. overshoot.

A4:

Command Window

```
>> kp = 2; Ti = 2;
num = kp*[Ti 1];
den = [Ti 0];
PI2 = tf(num,den)

PI2 =

  4 s + 2
  -------
    2 s

Continuous-time transfer function.

>> num = [1];
den = [ 1 3 3 1];
sys1 = tf(num,den);
transf2 = feedback(PI2*sys1,1)

transf2 =

               4 s + 2
  -------------------------------
  2 s^4 + 6 s^3 + 6 s^2 + 6 s + 2

Continuous-time transfer function.
```

```
transf3 =

               16 s + 2
  ----------------------------------
  8 s^4 + 24 s^3 + 24 s^2 + 24 s + 2

Continuous-time transfer function.

>> kp = 2; Ti = 4;
num = kp*[Ti 1];
den = [Ti 0];
PI1 = tf(num,den)

PI1 =

  8 s + 2
  -------
    4 s

Continuous-time transfer function.

>> transf1 = feedback(PI1*sys1,1)

transf1 =

               8 s + 2
  ----------------------------------
  4 s^4 + 12 s^3 + 12 s^2 + 12 s + 2

Continuous-time transfer function.

>> figure; stepplot(transf1, transf2, transf3)
>>
```

Figure 7. Closed loop transfer function for PI1,PI2,PI3 and code for the step plot of the three transfer functions

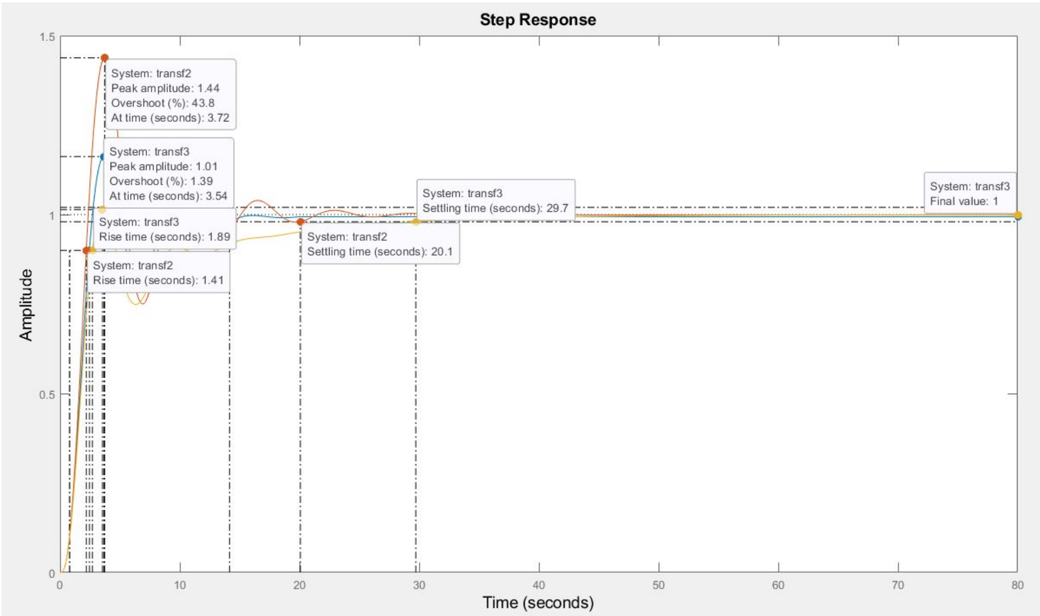

Figure 8. Step plot of the input and output of the closed loop transfer function for Ti = 2, Ti = 8, and Ti = 4

Table 2.

	Rise time (tr)	%Max. O.S.	Settling time (ts)	Steady state error (ess)
PI Controller Ti = 2, Kp = 2	1.41	43.8	20.1	0
P Controller Ti = 8, Kp = 2	1.89	1.39	29.7	0

We can see from the above table and comparing with A3 that decreasing the Ti increases the overshoot and vice versa. Thus, increasing the Ti makes the system more stable. The settling time increases as the Ti increases. Ti = 8 has the highest settling time. Since the oscillations reach a maximum peak of 0, although it's stable it takes a while to reach the steady state error and this increases its settling time. The rise time is the same for all controllers with different Ti, so increasing or decreasing the Ti does not have an effect on the rise time. Increasing or decreasing the Ti have no effect on the steady state error and they are all 0.

A5:

This code is a continuation of the above figure. The bode plot is calculated with the following line of code:

figure; bode(PI1,PI2,PI3)

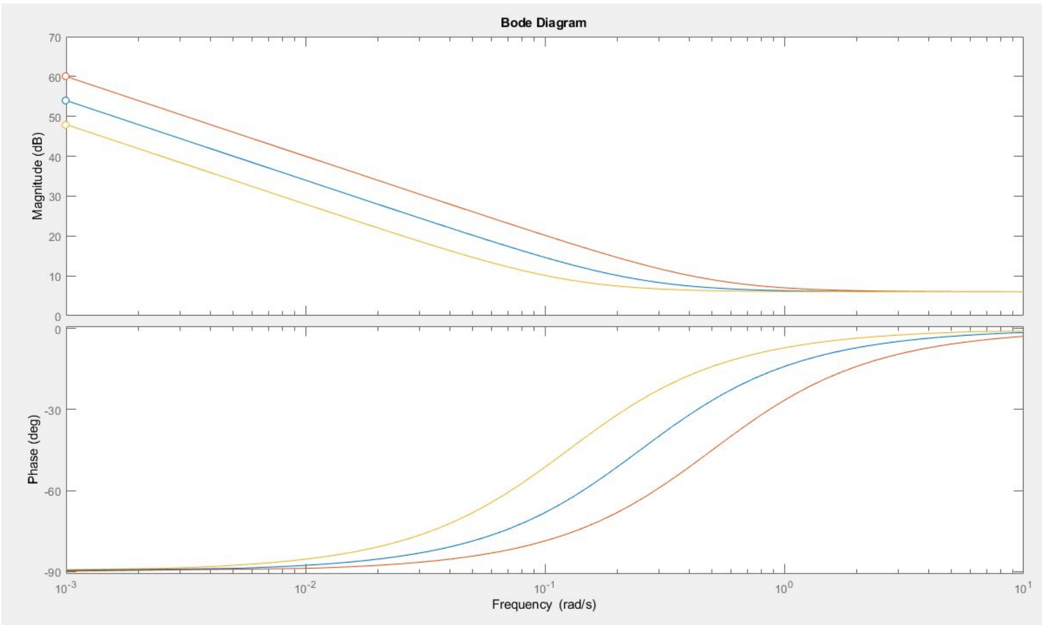

Figure 9. Bode plot of all PI controllers with Ti = 2,4,8 and Kp = 2.

The blue plot is the original signal with Ti = 4, orange plot is Ti = 4, and the yellow plot is Ti = 8. We know that PI controller is type 1 transfer function since at low the frequency the graph starts with the slope of -20*B and the phase starts with -90 degrees. The magnitude plot all meet at the corner frequency of 0. Changing the Ti value affects the corner frequencies. By decreasing the Ti, the corner frequencies shift to the higher frequency and phase plot approach 0 degrees at the higher frequencies.

B1:

DC Gain: $\boxed{K_p}$

$$\frac{T_d \cdot s}{\beta} + 1 = 0$$

$$\frac{T_d s}{\beta} = -1$$

Pole: $\boxed{s = \frac{-\beta}{T_d \cdot s}}$

$$K_p\left[T_d(1+\beta)s + \beta\right] = 0$$

$$T_d(1+\beta)s + \beta = 0$$

Zeroes: $\boxed{s = \frac{-\beta}{T_d(1+\beta)}}$

High freq gain: $\frac{K_p T_d(1+\beta)}{T_d} = \boxed{K_p(1+\beta)}$

Figure 10. Calculation of the pole, zero, and gain of the PD controller.

The pole is always located at the origin so the increasing or decreasing the Td has no effect on the poles. The zeroes however can be affected by the Td. Thus, increasing the Td makes the zero closer to the origin and decreasing the Td moves the zero away from the origin.

B2:

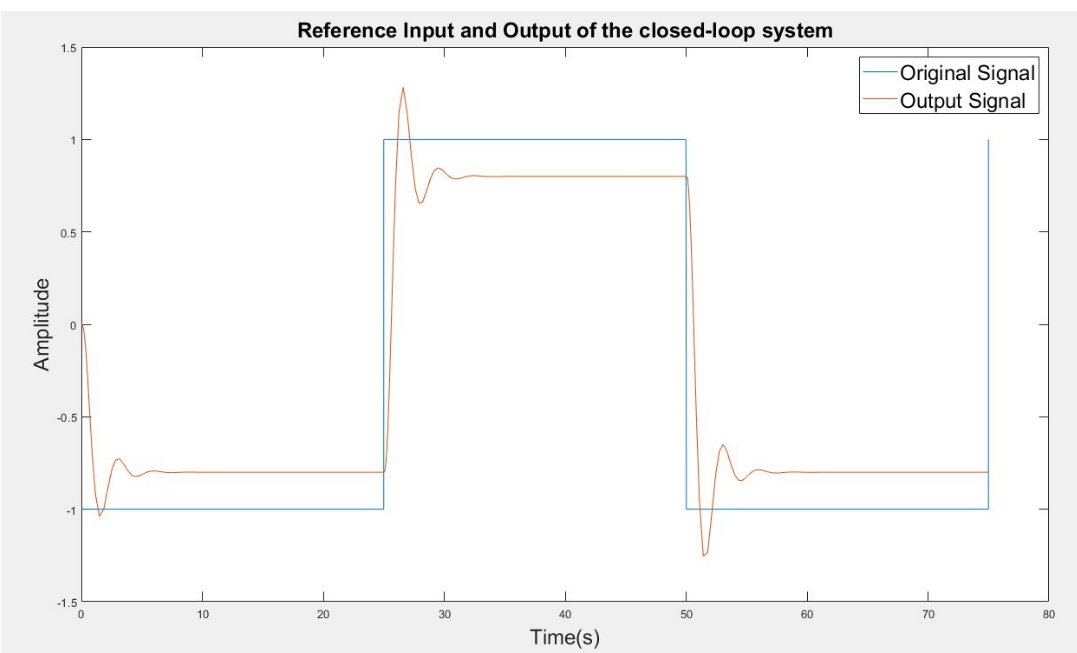

Figure 11. Reference input and output plot of the closed loop system. The blue line represents the original signal while the orange line represents the output signal.

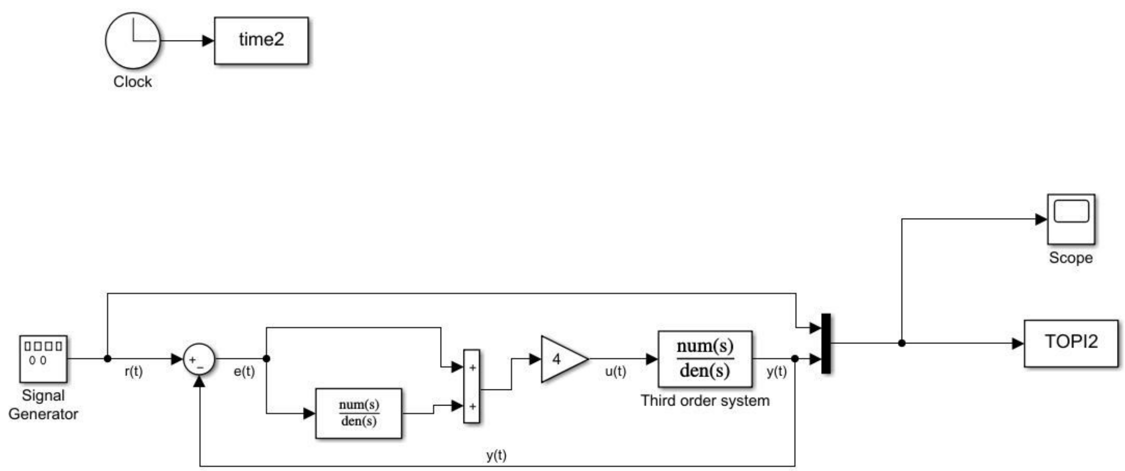

Figure 12. Simulink model of the closed loop system

B3:

```
>> num = kp*[Td+Td/10 1];
>> den = [Td/10 1];
>> PD1 = tf(num, den)

PD1 =

  4.4 s + 4
  ---------
  0.1 s + 1

Continuous-time transfer function.

>> transf1 = feedback(PD1*sys1, 1)

transf1 =

                 4.4 s + 4
  ---------------------------------------
  0.1 s^4 + 1.3 s^3 + 3.3 s^2 + 7.5 s + 5

Continuous-time transfer function.

>> figure; stepplot(transf1)
>>  kp = 4;
>> transf2 = feedback(kp*sys1,1)

transf2 =

            4
  ---------------------
  s^3 + 3 s^2 + 3 s + 5

Continuous-time transfer function.

>> figure; stepplot(transf2, transf1)
>>
```

Figure 13. Closed loop transfer function (transf1, transf2) and the code for the step plot of the PD controller with Td = 2, kp = 4 and P controller with kp = 4.

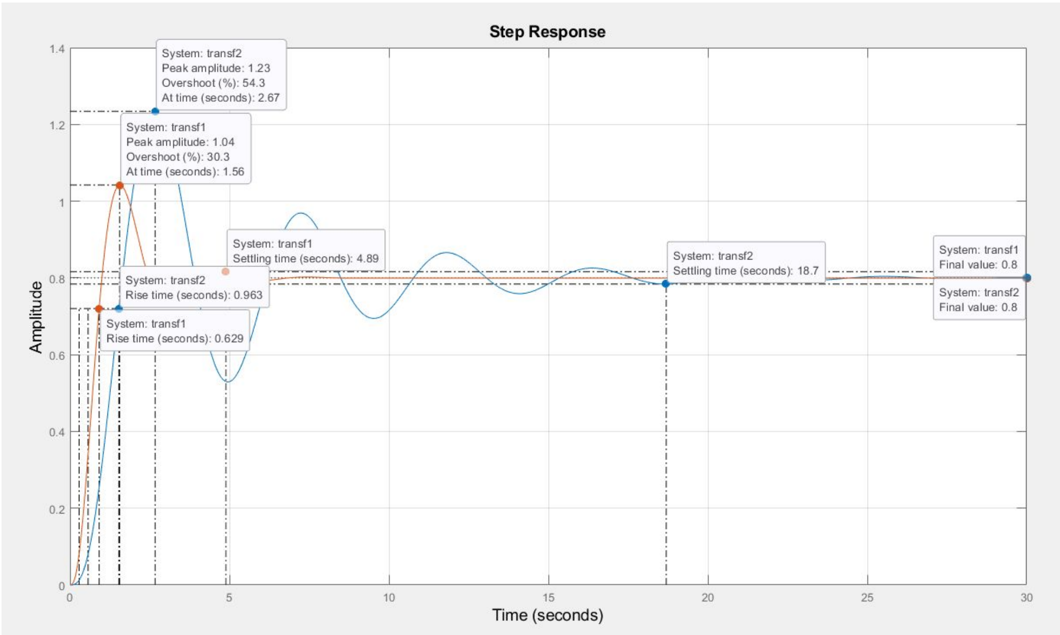

Figure 14. Plot of the PD (orange) and P (blue) controller

Table 3.

	Rise time (tr)	%Max. O.S.	Settling time (ts)	Steady state error (ess)
PD Controller Td = 2, Kp = 4	0.629	30.3	4.89	0.2
P Controller Kp = 4	0.963	54.3	18.7	0.2

The P controller with Kp = 4 i.e. the blue plot. Adding the PD controller enhances the transient response as it clearly decreases the overshoot and the number of oscillations and thus makes the system more stable. In terms of the settling time, the P controller is a lot longer than the PD controller. The PD controller becomes faster, less overshoot, and stable however it is unaffected in terms of steady state error. Thus, the PD controller enhances the transient response and not the steady state part.

B4:

Command Window

```
>> kp = 4; Td = 1;
num = kp*[Td+Td/10 1];
den = [Td/10 1];
PD1 = tf(num,den)

PD1 =

  4.4 s + 4
  ---------
  0.1 s + 1

Continuous-time transfer function.

>> transf1 = feedback(PD1*sys1,1)

transf1 =

                 4.4 s + 4
  ---------------------------------------
  0.1 s^4 + 1.3 s^3 + 3.3 s^2 + 7.5 s + 5

Continuous-time transfer function.

>> kp = 4; Td = 2;
num = kp*[Td+Td/10 1];
den = [Td/10 1];
PD2 = tf(num,den)

PD2 =

  8.8 s + 4
  ---------
  0.2 s + 1
```

```
Continuous-time transfer function.

>> transf2 = feedback(PD2*sys1,1)

transf2 =

                  8.8 s + 4
  --------------------------------------
  0.2 s^4 + 1.6 s^3 + 3.6 s^2 + 12 s + 5

Continuous-time transfer function.

>> kp = 4; Td = 0.1;
num = kp*[Td+Td/10 1];
den = [Td/10 1];
PD3 = tf(num,den)

PD3 =

  0.44 s + 4
  ----------
  0.01 s + 1

Continuous-time transfer function.

>> transf3 = feedback(PD3*sys1,1)

transf3 =

                    0.44 s + 4
  -------------------------------------------
  0.01 s^4 + 1.03 s^3 + 3.03 s^2 + 3.45 s + 5

Continuous-time transfer function.

>> figure; stepplot(transf1, transf2, transf3)
>>
```

Figure 15. Closed loop transfer function (transf1, transf2, transf3) for the three controllers with the code for the step plots.

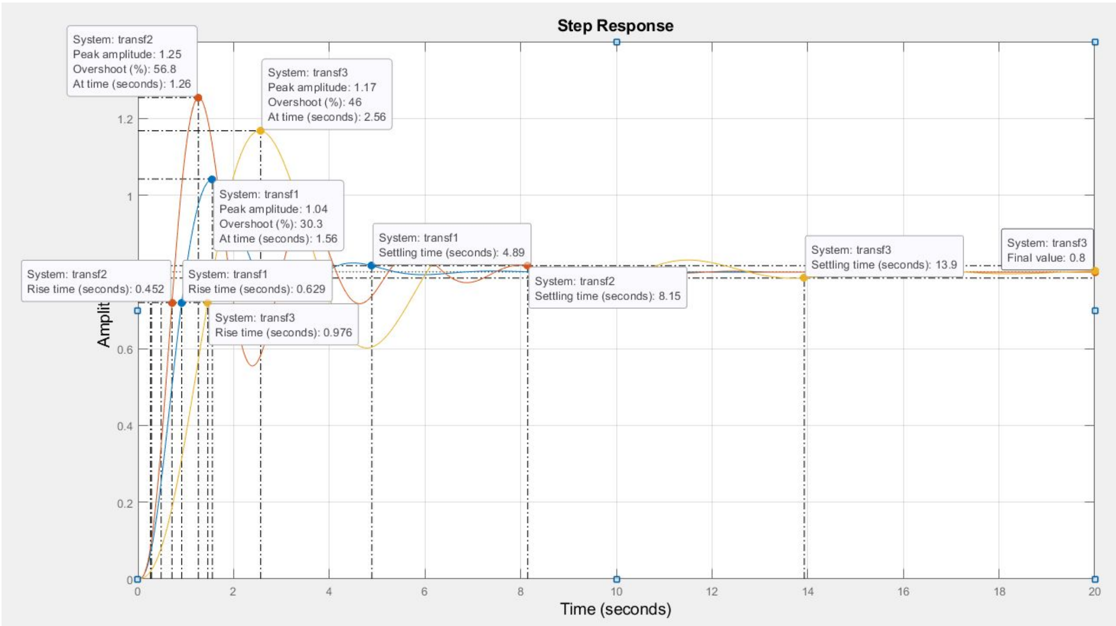

Figure 16. Step plots the three PD controllers with kp = 4 and Td = 1 (blue),2 (orange), 0.1 (yellow).

Blue plot is the original PD controller with Td = 1, orange is Td = 2, yellow is Td = 0.1. It is evident that increasing the Td increases the % max overshoot. In both cases for Td = 0.1 and 2 the systems become less stable. The original PD controller with Td=1 and Kp = 4 is the ideal signal i.e. it is the most stable. In addition the phase margin can be obtained for the original PD controller and not any other. In terms of the settling time, the blue plot takes the least time and thus gives better results. For the steady state error, it doesn't change even if the Td is increased or decreased. After observing the rise time we see that increasing the Td increases the rise time and thus the speed of the system and decreasing the Td decreases the speed of the system.

Table 4.

	Rise time (tr)	%Max. O.S.	Settling time (ts)	Steady state error (ess)
PD Controller Td = 2, Kp = 4	0.452	56.8	8.15	0.2
P Controller Td = 0.1,Kp = 4	0.976	46	13.9	0.2

B5:

```
>> figure; stepplot(transf1, transf2, transf3)
>> figure; bode(PD1,PD2,PD3)
fx >>
```

Figure 17. Code for the bode plot of the three PD controller functions.

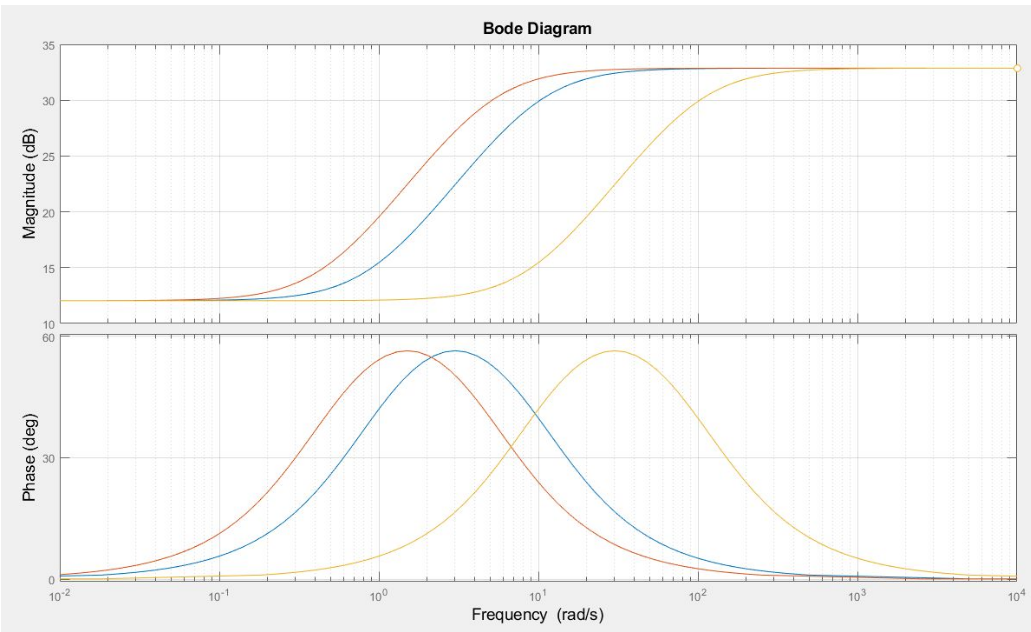

Figure 18. Bode plot of the three PD controllers with kp = 4 and Td = 1 (blue),2 (orange), 0.1 (yellow).

It is evident from the above magnitude plot that increasing or decreasing the Td does not affect the high frequency gain and low frequency gain. Based on the behaviour, this resembles a lead compensator because there is a pole added in the equation denominator in the transfer function of the PD controller: Td.s / B + 1 which converts the PD controller into a lead controller. In addition, the positive phase and the low gain at low frequency and high gain at high frequency relates to the behaviour of a lead compensator. Thus Td has no effect on the high frequency and low frequency gain. It only shifts the pole zero location and also has no effect on the maximum value in the phase plot.

C1)

$$\frac{U(s)}{E(s)} = K_p\left(1 + \frac{1}{T_i s} + \frac{T_d s}{\frac{T_d}{\beta}s + 1}\right)$$

DC Gain:

$$\lim_{s \to 0} K_p\left(1 + \frac{1}{T_i s} + \frac{T_d s}{\frac{T_d}{\beta}s + 1}\right)$$

$$= K_p\left(1 + \frac{1}{T_i(0)} + \frac{T_d(0)}{\frac{T_d}{\beta}(0) + 1}\right)$$

$$= K_p\left(1 + \infty + 0\right)$$

$$= \infty$$

High Frequency Gain:

$$\lim_{s \to \infty} K_p\left(1 + \frac{1}{T_i s} + \frac{T_d s}{\frac{T_d}{\beta}s + 1}\right)$$

$$= \lim_{s \to \infty} K_p\left(1 + \frac{1}{T_i s} + \frac{T_d s}{\frac{T_d}{\beta}s + 1} \cdot \frac{\beta}{\beta}\right)$$

$$= \lim_{s \to \infty} K_p\left(1 + \frac{1}{T_i s} + \frac{T_d s \beta}{T_d s + \beta}\right)$$

$$= K_p(1 + \beta)$$

When comparing these two gains to the PI and PD controllers, we can see exactly what the PID controller takes from both. The DC gain is similar to the PI controllers, which corresponds to the lower frequencies. The higher frequency gain is similar to the PD controller.

C2)

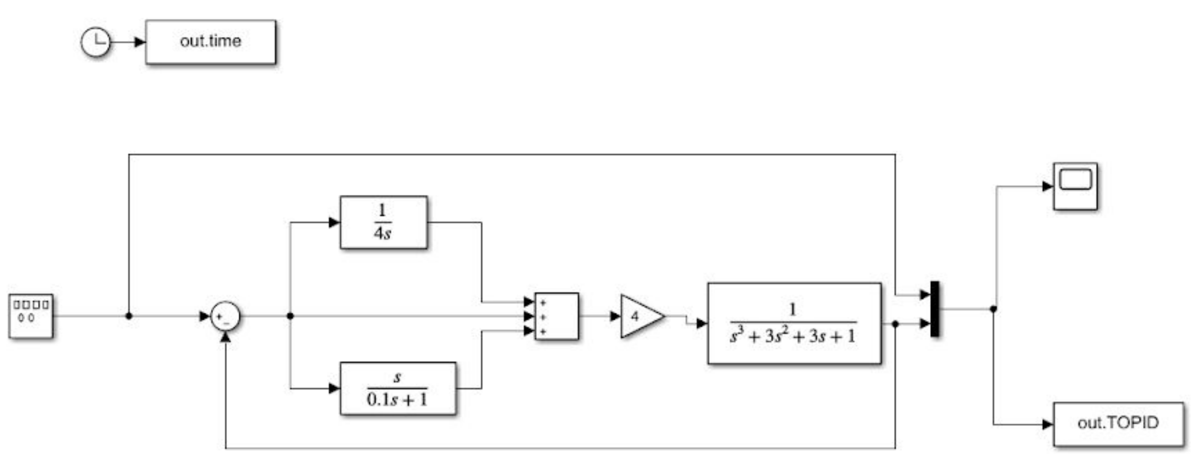

Figure 19. Simulink diagram of the proportional-integral-derivative controller.

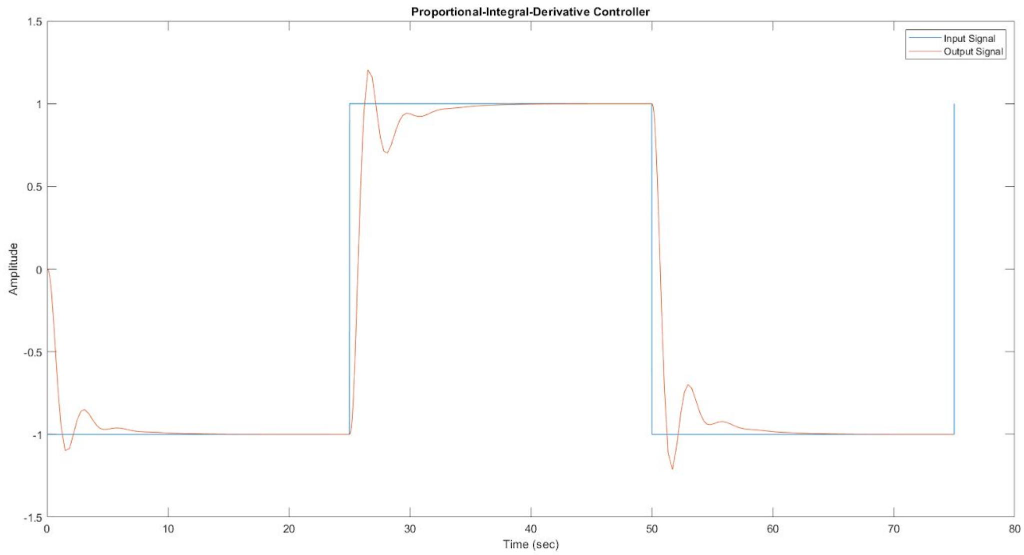

Figure 20. Plot of input signal (blue) and output signal (red) of the proportional-integral-derivative controller.

When looking at figure 20., we can see that the PID controller is working to the theoretical assumptions. The overshoot and rise time are both low, and the steady-state error is essentially zero. Therefore, like previously mentioned this PID controller is taking characteristics from both the PI and PD controllers. The output signal is behaving quite similarly to the input square-wave signal, which is ideally what we want.

C3)

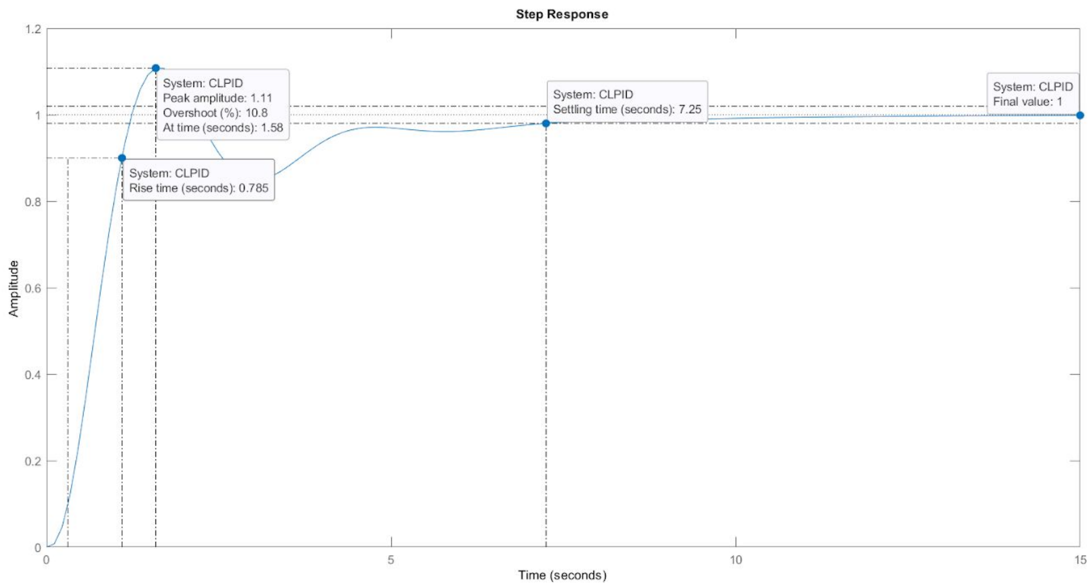

Figure 21. Step response of the proportional-integral-derivative controller.

Table 5. Characteristics of the proportional-integral-derivative controller.

	Rise Time (tr)	%Max Overshoot	Settling Time (ts)	Steady-State Error (ess)
PID Controller (Td = 1, Ti = 4, Kp = 4)	0.785 sec	10.8%	7.25 sec	0

To compare the PID controller to the PI and PD controllers we will split it up into the four characteristics labelled in each table:
Rise Time: The rise time of the PID controller is similar to the rise time of the PD controller, with the PI controller being almost double that of the PID controller.
%Max Overshoot: The PID controller has a similar max overshoot to the PI controller in A3. The PD controller max overshoot is five times as high as the PD where the PI controller is max two times greater.
Settling Time: The settling time of the PID is similar to the PD controller, compared to the PI controller. But there is still a decent difference between the two times.
Steady-state error: The steady-state error of the PID controller is similar to the PI controller, but also still quite close to the error of the PD controller.

C4)

D1)

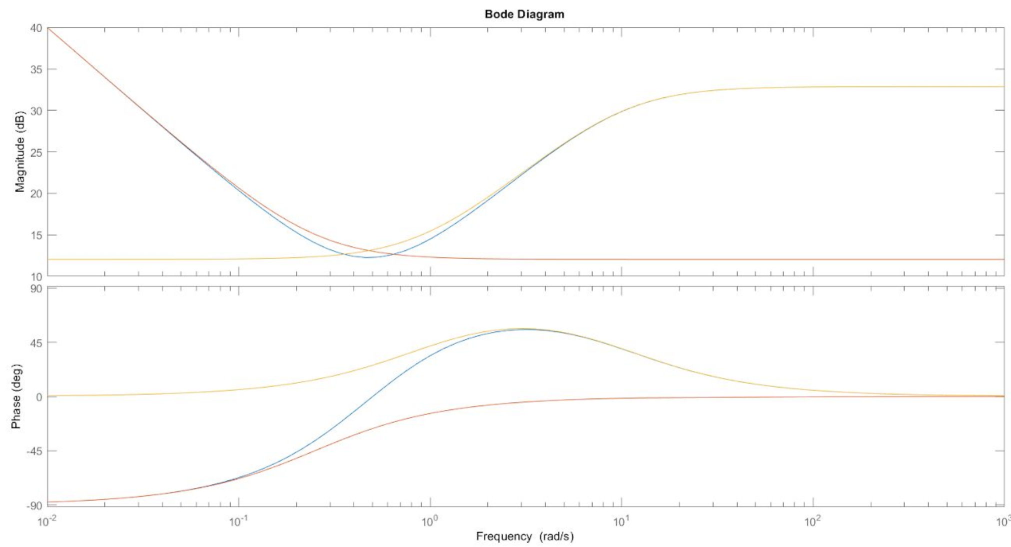

Figure 22. Bode plot of the PID (blue), PI (red), and PD (yellow) controllers.

Using this plot we can easily compare the three controllers. We will split up the comparison into two sections, high frequencies and low frequencies. Firstly, in the low frequencies we can see that the PID controller acts similar to the PI controller. They have similar gains and they have a very similar starting frequency and frequencies in general (when in the low frequencies). Secondly, in the high frequencies the PID controller follows the PD controller. The PID controller follows both the gain and phase of the PD controller in the high frequencies. Lastly, all three controllers have the same ending gain and phase. This also confirms that at the transient part the PID controller follows the PD controller, and at the steady state part the PID controller follows the PI controller.

D1)

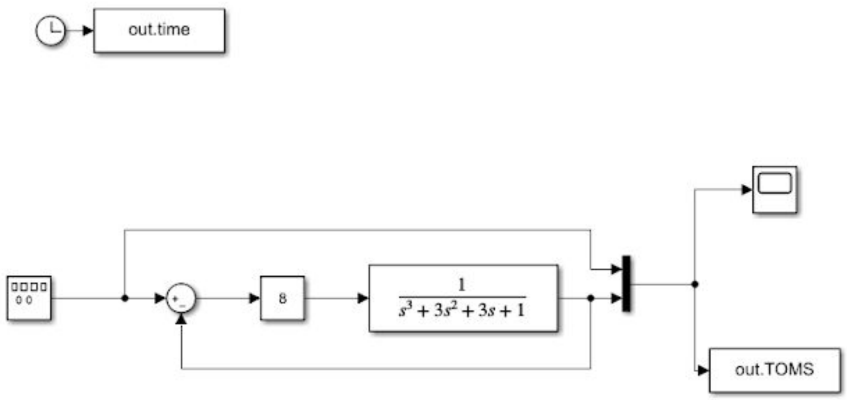

Figure 23. Simulink diagram of the ZIegler-Nichols approach of controller tuning.

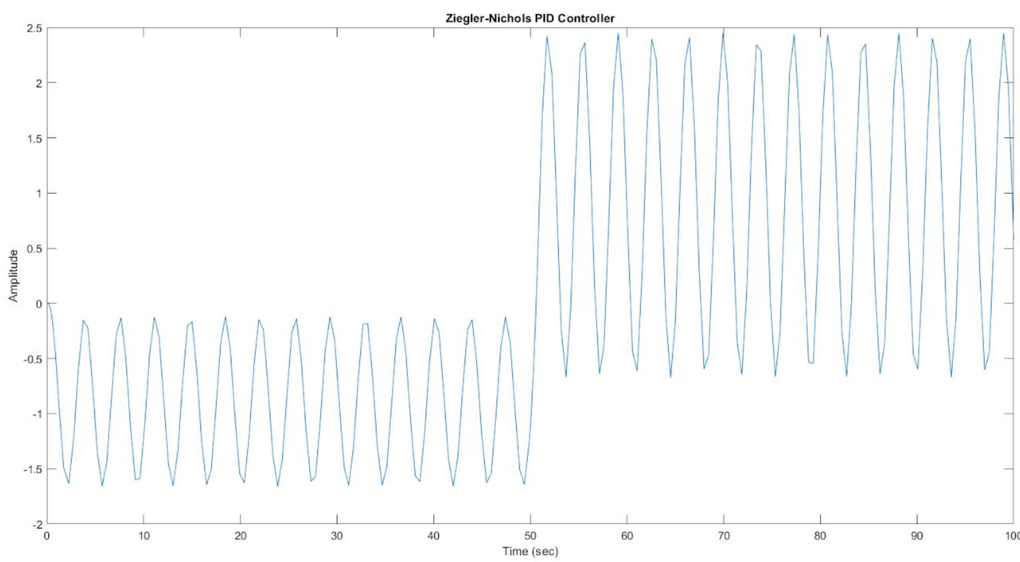

Figure 24. Plot of the output of the above simulink model (Figure X) with the slider gain at 8. Which corresponds to the point of marginal stability.

D2) Using the plot we can determine that:

Ku = 8;
Tu = 3.6
Kp = 4.8
Ti = 1.8
Td = 0.225

D3)

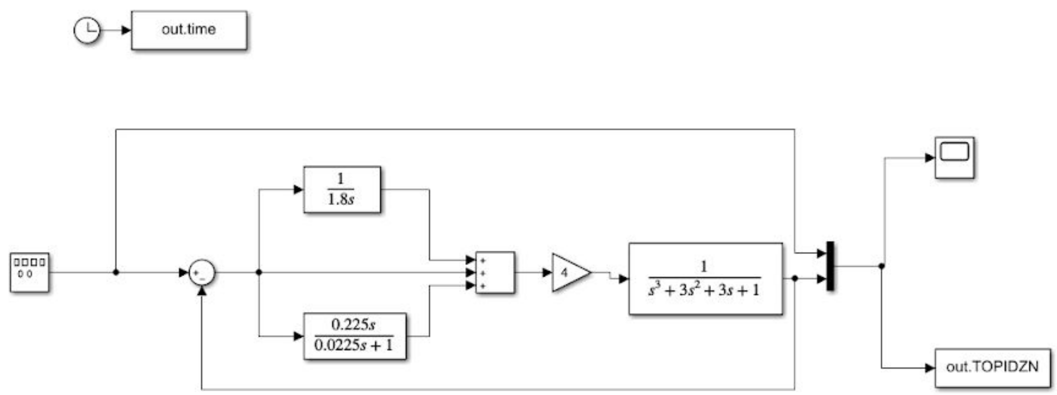

Figure 25. Simulink model of the PID controller with the newly calculated parameters in part D2.

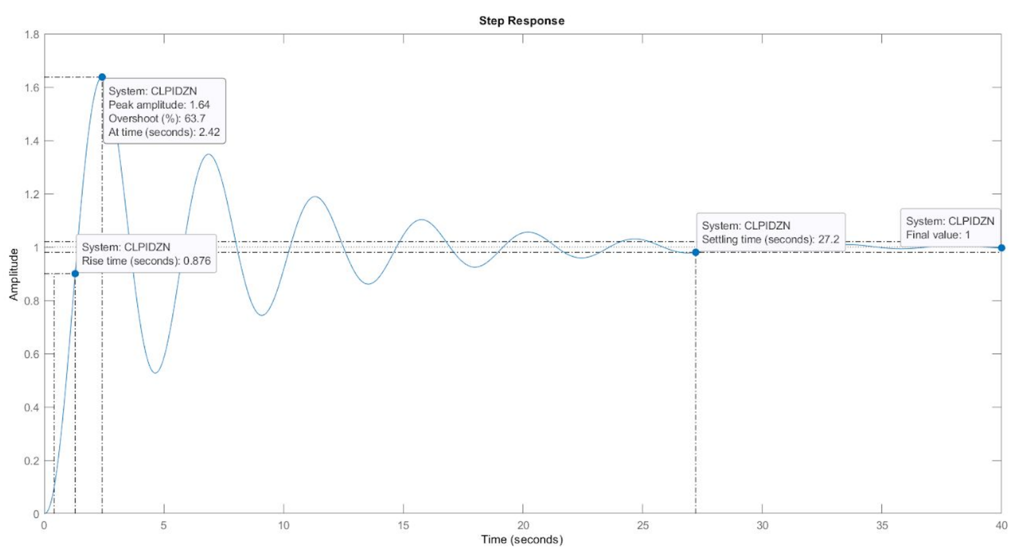

Figure 26. Step response of the PID controller in Figure X, with characteristics highlighted.

Table 6.

	Rise Time (tr)	%Max Overshoot	Settling Time (ts)	Steady-State Error (ess)
PID Controller (Td = 1, Ti = 4, Kp = 4)	0.876 sec	63.7%	27.2 sec	1

To compare the two PID controllers we will split it up into the characteristics outlined in the tables:

Rise Time: Both PID controllers have very similar rising time, being only about 0.1 seconds off.
%Max Overshoot: The controller made in part D has a very large overshoot compared to the one in part C. It is about a 50% difference between them.
Settling Time: The new controller (part D) has a very long settling time compared to the one in part c. This new controller is just over 3 times slower to settle than the controller in part C.
Steady-state error: The steady state errors are the same, both having errors of 0.

Appendix:

PART 1:

```
%A2

tB1 = out.time.time;
iB1 = out.TOPI.data(:,1);
oB1 = out.TOPI.data(:,2);

figure(1)
plot(tB1,iB1)
hold on
plot(tB1,oB1)
xlabel('Time(s)','FontSize', 18)
ylabel('Amplitude','FontSize', 18)
title('Reference Input and Output of the closed-loop system','FontSize', 18)
legend('Original Signal','Output Signal','FontSize', 18);


%B2

tB1 = out.time2.time;
iB1 = out.TOPI2.data(:,1);
oB1 = out.TOPI2.data(:,2);

figure(1)
plot(tB1,iB1)
hold on
plot(tB1,oB1)
xlabel('Time(s)','FontSize', 18)
ylabel('Amplitude','FontSize', 18)
title('Reference Input and Output of the closed-loop system','FontSize', 18)
legend('Original Signal','Output Signal','FontSize', 18);

%% C2

tC2 = out.TOPID.time;
inC2 = out.TOPID.data(:,1);
outC2 = out.TOPID.data(:,2);

figure(1)
plot(tC2,inC2)
```

```
hold on
plot(tC2,outC2)
title('Proportional-Integral-Derivative Controller')
legend('Input Signal', 'Output Signal')
ylabel('Amplitude')
xlabel('Time (sec)')

%% C3

kp = 4;
Ti = 4;
Td = 1;

num = kp*[Ti*Td/10+Ti*Td Ti+Td/10 1];
den = [Ti*Td/10 Ti 0];
PID = tf(num,den);

num = [1];
den = [1 3 3 1];
sys = tf(num,den);

CLPID = feedback(PID*sys,1); %closed-loop transfer function

figure(2)
stepplot(CLPID)

%% C4

kp = 4;
Ti = 4;
Td = 1;

num = kp*[Ti 1];
den = [Ti 0];

PI = tf(num,den);

num = kp*[Td+Td/10 1];
den = [Td/10 1];

PD = tf(num,den);

figure(3)
```

```
bode(PID,PI,PD)

%% D1

tD1 = out.TOMS.time;
inD1 = out.TOMS.data(:,1);
% Gain = 8
outg8 = out.TOMS.data(:,2);


figure(3)
plot(tD1,outg8)
title('Ziegler-Nichols PID Controller')
ylabel('Amplitude')
xlabel('Time (sec)')

%% D3

tD3 = out.TOPIDZN.time;
inD3 = out.TOPIDZN.data(:,1);
outD3 = out.TOPIDZN.data(:,2);

figure(5)
plot(tD3,inD3)
hold on
plot(tD3,outD3)
title('Ziegler-Nichols Tuning Method')
legend('Input Signal', 'Output Signal')
ylabel('Amplitude')
xlabel('Time (sec)')

%%

kp = 4.8;
Ti = 1.8;
Td = 0.225;

num = kp*[Ti*Td/10+Ti*Td Ti+Td/10 1];
den = [Ti*Td/10 Ti 0];
PID = tf(num,den);

num = [1];
den = [1 3 3 1];
```

```
sys = tf(num,den);

CLPIDZN = feedback(PID*sys,1); %closed-loop transfer function

figure(6)
stepplot(CLPIDZN)
```

Personal Summaries:

Kushal:
500843903

In the first part, we investigated the PI, PD, and PID controllers and how they affect the stability and properties of a given system. For the PI controller we found that it doesn't change the speed of the system. However it enhances the steady state portion of the system by eliminating the steady state to 0. Adding a PI controller also increases the settling time of the system, and decreases the % Max. overshoot. From the bode diagram of all PI controllers we also found that changing the Ti value affects the corner frequencies. By decreasing the Ti, the corner frequencies shift to the higher frequency and phase plot approaches 0 degrees at the higher frequencies. For the PD controller we found that it enhances the transient response by decreasing the overshoot and the number of oscillations and thus makes the system more stable. Overall, the PD controller becomes faster, has less overshoot, and is stable however it is unaffected in terms of steady state error. We also found that increasing the Td increases the % max overshoot. For the steady state error, it doesn't change even if the Td is increased or decreased. After observing the rise time we see that increasing the Td increases the rise time and thus the speed of the system, and decreasing the Td decreases the speed of the system. Based on the behaviour, it resembles a lead compensator because there is a pole added in the equation denominator in the transfer function of the PD controller: Td.s / B + 1 which converts the PD controller into a lead controller.Thus Td has no effect on the high frequency and low frequency gain. It only shifts the pole zero location and also has no effect on the maximum value in the phase plot. In parts C and D, we combined the PI and PD controllers to make PID controllers which preserve the characteristics of the previous controllers like PI’s DC gain and PD’s high frequency gain. This was achieved through the Ziegler-Nichols approach where a gain slider was used to facilitate the oscillations and other analysis. A PID controller reduces the effects of a system’s transient response while improving its steady state error. Overall, this lab is a great practice to model different controllers which has crucial applications in the electrical and biomedical engineering industry.

Peter Zastawny

500813230

In this lab we explored PI, PD, and PID controllers and looked at their characteristics to help understand what improvements they can make to a system. In part A, we first looked at PI controllers. We know that PI controllers are used to reduce or remove the steady state error of the system. By adding another pole to the system, a pole that is at least 10 times away from the dominant poles, it can help the steady state error. We also saw how changing Ti shifts the corner frequencies to a higher frequency. In part B, we looked at the PD controller. The function of the PD controller is to enhance the transient response of the system and also increase the stability. This controller can also reduce the overshoot and the settling time of the system, by adding another zero to the system. Which,this zero, can also be considered a derivative when in the ‘s’ domain. We also noticed how changing Td increases the overshoot, rise time and decreases the speed of the system. Lastly, we looked at the PID controller. This controller is the culmination of the previous two controllers. It provides all the benefits of both the PI and PD controllers. It can improve the steady state error and the transient part of the system, while also increasing the stability. In part D, we looked at a practical approach at creating a PID controller. This approach is called the Ziegler-Nichols approach. It uses a gain slider to find where the system is in oscillations, at this point we can use characteristics such as the amplitude to calculate the necessary components of a PID controller. This lab is the culmination of most of the work done in this course, to help improve and control electrical and mechanical systems.

BME639: Control Systems Lab 2 Grading Sheet

Part 1A: DC Servo Motor Modeling Using First-Principles **/12**

Part 1B: DC Servo Motor Modeling Using Experiments **/8**

Part 1C: DC Servo Motor Speed Control **/7**

Part 1D: DC Servo Motor Position Control **/7**

1. **Part 2A: Time Response of a Lead Compensator** **/9**

2. **Part 2B: Lead Compensator and Second-Order Systems** **/11**

3. **Part 2C: Time Response of a Lag Compensator** **/9**

4. **Part 2D: Lag Compensator and Second-Order Systems** **/12**

/10

1. **General Formatting: Clarity, writing style, grammar, spelling, layout of the report 2.**

Total Mark for the Collaborative Part of the Report /85

Partner 1 (name): Partner 2 (name):

Summary Mark /15 Summary Mark /15

Interview Mark Pass / Fail Interview Mark Pass / Fail

TOTAL: /100 TOTAL: /100

Before submitting your report, your TA asks questions about your report. If there is no consistency between your oral answer and your report, you will lose 50% of your total mark. For this part, Pass or fail will be circled next to your name accordingly.