LAB 5: PERFORMANCE ANALYSIS WITH PID CONTROLLER

Course Number	**MENG 3510**
Course Title	**Control Systems**
Semester/Year	**Winter / 2025**

Lab/Tutorial Report No.	**Lab 5**
Report Title	**Performance Analysis with PID Controller**
Section No.	0NA
Group No.	N/A
Submission Date	February 5th, 2025
Due Date	

Student Name	**Signature***	**Total Mark**
Michael McCorkell		**/ 50**

* By signing the above, you attest that you have contributed to this submission and confirm that all the work you have contributed to this submission is your work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a ZERO on the work or possibly more severe penalties.

https://academic-regulations.humber.ca/2021-2022/17.0-ACADEMIC-MISCONDUCT

LAB 5 Grading Sheet

Student First Name: Michael McCorkell	
Part 1: Proportional Control	**/15**
Part 2: Proportional-Derivative (PD) Control	**/15**
Part 3: Proportional-Integral-Derivative (PID) Control	**/15**
General Formatting: **Clarity, Writing style, Grammar, Spelling, Layout of the report**	**/5**
Total Mark	**/50**

LAB 5: PERFORMANCE ANALYSIS WITH PID CONTROLLER

OBJECTIVES

- To study the performance and stability of a high-order system under PID control
- To learn how to realize PID controller in MATLAB/Simulink
- To study how the tuning parameters of PID controller affect the control system stability and performance

DISCUSSIONS OF FUNDAMENTALS

PROPORTIONAL-INTEGRAL-DERIVATIVE (PID) CONTROL

A PID controller is a feedback control mechanism that is widely used in industrial control systems and a variety of other applications requiring continuously modulated control. A PID controller continuously calculate an error value $e(t)$ as the difference between reference input $r(t)$ and actual output $y(t)$:

$$\boldsymbol{e(t) = r(t) - y(t)}$$

and generates a control signal based on proportional, integral, and derivative terms (denoted by P, I, and D. respectively). The complete PID control signal can be expressed mathematically as a sum of the three terms:

$$\boldsymbol{u(t) = K_p\left[e(t) + \frac{1}{T_i}\int e(\tau)d\tau + T_d\frac{de(t)}{dt}\right]}$$

The PID controller can also be described by the transfer function:

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

and the corresponding block diagram can be implemented as below:

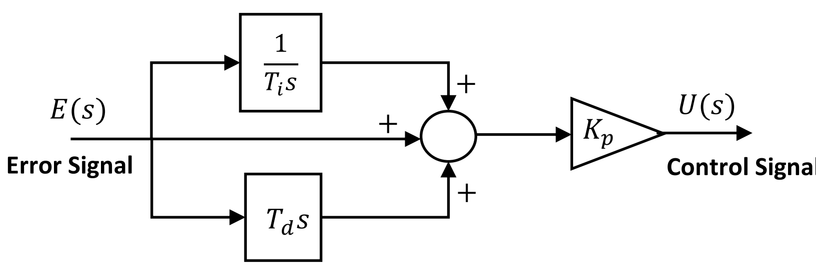


Figure 1: Block diagram of PID controller

The PID controller has couple of special cases:

<u>P Controller:</u>

$$\boldsymbol{u(t) = K_p e(t) \quad \rightarrow \quad U(s) = K_p E(s) \quad \rightarrow \quad G_c(s) = K_p}$$

<u>PI Controller:</u>

$$\boldsymbol{u(t) = K_p\left[e(t) + \frac{1}{T_i}\int e(\tau)d\tau\right] \quad \rightarrow \quad U(s) = K_p\left(1 + \frac{1}{T_i s}\right)E(s) \quad \rightarrow \quad G_c(s) = K_p\left(1 + \frac{1}{T_i s}\right)}$$

<u>PD Controller:</u>

$$\boldsymbol{u(t) = K_p\left[e(t) + T_d\frac{de(t)}{dt}\right] \quad \rightarrow \quad U(s) = K_p(1 + T_d s)E(s) \quad \rightarrow \quad G_c(s) = K_p(1 + T_d s)}$$

PROPORTIONAL CONTROL ACTION

The **proportional control action (P-term)** drives the system based on the ***current value*** of error signal $e(t)$. The error signal is multiplied by the ***proportional gain*** $\boldsymbol{K_p}$, which can be found either experimentally or calculated based on system requirements such as rise time, overshoot, …. Using P control alone will result in a ***steady-state error*** (unless the system has an integrator, type 1 or higher). This is because as $y(t)$ approaches $r(t)$, the error $e(t)$ becomes smaller and smaller, and the proportional term will eventually become too small to maintain $e(t)$ to be zero.

INTEGRAL CONTROL ACTION

The **integral control action (I-term)** accounts for ***past values*** of error signal $e(t)$ and integrates them over time to produce the I-term with the ***integral time constant of*** $\boldsymbol{T_i}$. The integral term can ***eliminate the steady-state error*** e_{ss}. This is because even when the proportional term becomes very small, the integral term can be still large to keep pushing $e(t)$ all the way to zero. However, using proportional and integral control only (PI control) may decrease the relative stability of the control system and results in large oscillations in time response $y(t)$.

DERIVATIVE CONTROL ACTION

The derivative control action (D-term) utilizes an estimate of the ***future trend*** of error signal $e(t)$. It accounts for current ***rate of change*** of $e(t)$ to produce the D-term with the ***derivative time constant of*** $\boldsymbol{T_d}$. The derivative term can be used to ***deal with the overshoot and oscillation***, because it effectively acts as added damping in underdamped systems. In addition, to ***improve the stability*** of systems with the derivative term, a ***filter*** is often added to eliminate spikes in the derivative component of the feedback signal due to signal **noise** and **quantization** of the stepwise reference change.
Tuning a controller in PID mode requires careful adjustment of the *proportional gain* (K_p), the *integral time constant* (T_i), and the *derivative time constant* (T_d) to properly address the control requirements of the process.

PART 1: Proportional Control

Consider the following third-order transfer function model of a system.

$$\boldsymbol{G(s) = \frac{105}{s^3 + 15s^2 + 71s + 105}}$$

The **transfer function** of a **proportional controller** is:

$$\boldsymbol{u(t) = K_p e(t)} \qquad \rightarrow \qquad \boldsymbol{G_c(s) = K_p}$$

1. Build the following closed-loop system with **proportional control** and unity feedback in **Simulink**. Generate a unit-step signal with **Step Time** at $\boldsymbol{t = 0}$ as the reference input. Assign the proportional controller gain as $\boldsymbol{K_p = 2}$.

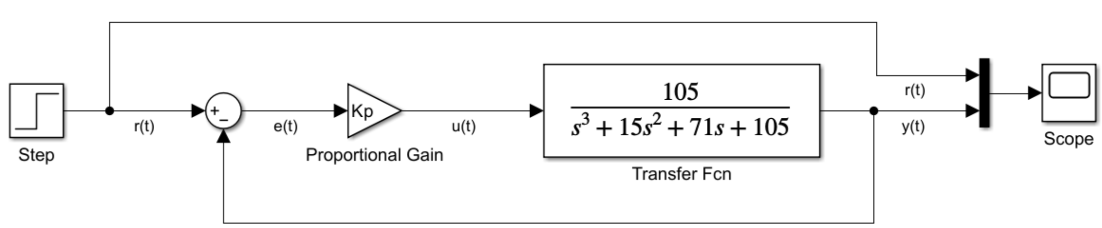


Figure 2: Closed system with P control

2. Click on the **Model Settings** icon in the **MODELING** tab to open the **Configuration Parameters** window. Click on the **Solver details**. Set the **Relative tolerance** value to **1e-12** to increase the resolution of the simulation. Then click **OK**.
3. **Save** the Simulink model diagram as **Lab5.slx.** Set the simulation to run for **10 sec**. **Run** the simulation by clicking on the **Run** button. Check the reference input and the system output in **Scope**.
4. Increase the proportional controller gain to $\boldsymbol{K_p} = \mathbf{4}, \mathbf{6}, \mathbf{8}, \mathit{and}\ \mathbf{10}$. **Run** the simulation and check the output signal.

   Do oscillations increase by increasing the proportional gain?

   YES **NO**

   Does the stability of the closed-loop system decrease by increasing the proportional gain?

   YES **NO**

   Determine for which values of the following proportional gains the closed-loop system is **stable, marginally stable,** or **unstable**?

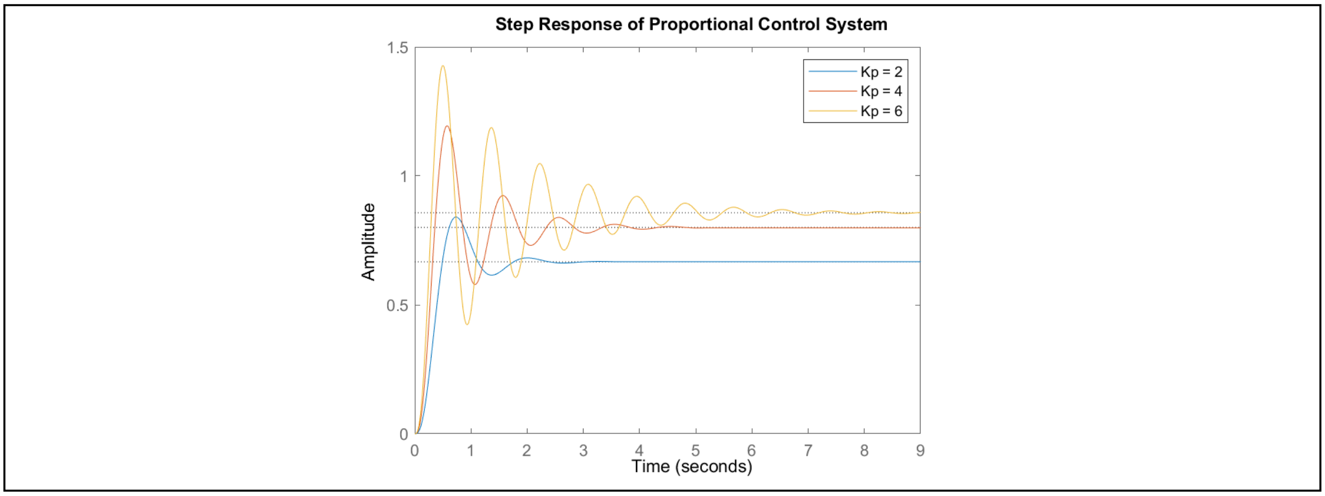
   $\boldsymbol{K_p} = \mathbf{4.12},$ $\boldsymbol{K_p} = \mathbf{8.15},$ $\boldsymbol{K_p} = \mathbf{9.14},$ $\boldsymbol{K_p} = \mathbf{12.16},$

5. Create following script code in **MATLAB** to plot the input-output signals of the closed-loop system for the proportional controller gain of $\boldsymbol{K_p} = \mathbf{2}, \mathbf{4},$ and $\mathbf{6}$.

```
figure;
for Kp = [2 4 6]
    num = 105*Kp;
    den = [1 15 71 105];
    sys_OL = tf(num,den);
    sys_CL = feedback(sys_OL,1);
    step(sys_CL), hold on
end
title('Step Response of Proportional Control System')
legend('Kp = 2','Kp = 4','Kp = 6')
```

Provide the graph below.

Step Response with P Control with $\boldsymbol{K_p} = \mathbf{2}, \mathbf{4}, \mathbf{and}\ \mathbf{6}$



6. Determine the required time-domain performance characteristics of the closed-loop system using the step plot and insert them in **Table 1**.

Table 1: Control System with P Control

	Rise Time	Percentage of Overshoot	Settling Time	Steady-state Error
$K_p = 2$	0.301	26.1%	2.11	0.143%
$K_p = 4$	0.212	49.3%	3.18	0.2%
$K_p = 6$	0.173	66.6%	5.76	0.333%

7. Explain the effect of increasing the **proportional controller gain** K_p on the performance characteristics of the closed-loop system.

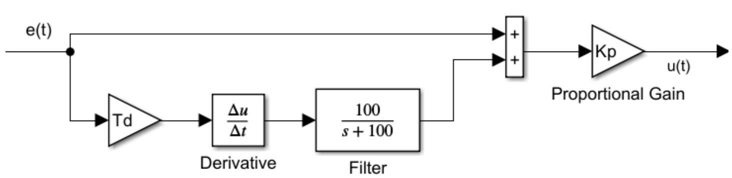
Rise time deacreases, overshoot increases, settiling time and SSE is increasing.

PART 2: Proportional-Derivative (PD) Control

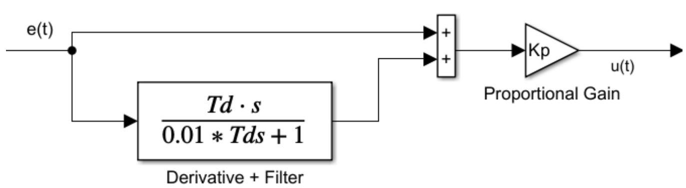
The **transfer function** of a **PD controller** is:

$$u(t) = K_p\left[e(t) + T_d\frac{de(t)}{dt}\right] \quad \rightarrow \quad U(s) = K_p(1 + T_d s)E(s) \quad \rightarrow \quad G_c(s) = K_p(1 + T_d s)$$

The **PD controller** can be realized in **Simulink** using a **Gain block** and a **Derivative block**. However, in practical applications, a ***filter*** is often added to eliminate spikes in the derivative component of the feedback signal due to signal **noise** and **quantization** of the stepwise reference change.



Therefore, the overall derivative term and the filter can be simplified as a single block as shown below by a **Transfer Fcn** block:



8. Modify your **Simulink** model to build the following closed-loop system with **PD control**. Assign the PD controller gains as $K_p = 4$ and $T_d = 0.2$.

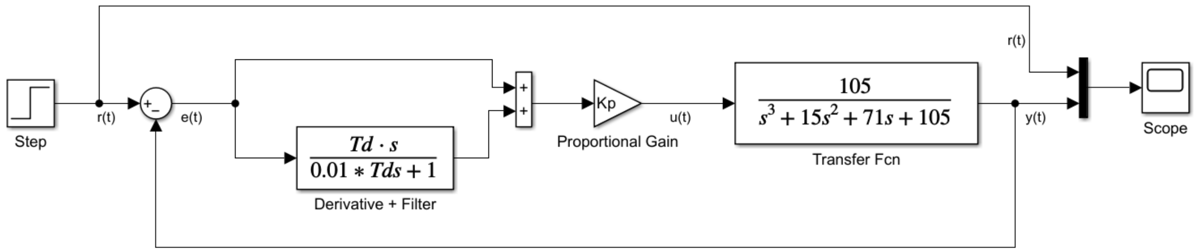


Figure 3: Closed system with PD control

9. Set the simulation to run for **3 sec**. **Run** the simulation. Check the reference input and the system output in **Scope**. Does the transient response (rise-time, overshoot) and the steady-state error of the output signal improve in comparison with the proportional controller only with $K_p = 4$ in **Step 5**?

Transient Response: **YES** **NO**

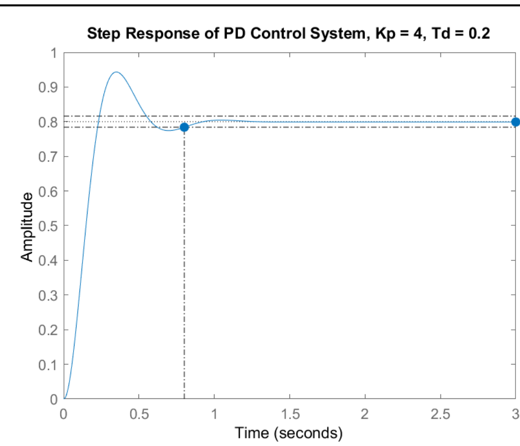
Steady-State Error: **YES** **NO**

10. Create the following script code in **MATLAB** to plot the input-output signal of the closed-loop system for the **PD controller** with $K_p = 4$, $T_d = 0.2$.

```
G = tf([105],[1 15 71 105]);
Kp = 4;
Td = 0.2;
Gc_PD = tf(Kp*[1.01*Td 1],[0.01*Td 1]);
sys_OL = G*Gc_PD;
sys_CL = feedback(sys_OL,1);
figure;
step(sys_CL), xlim([0 3])
title('Step Response of PD Control System, Kp = 4, Td = 0.2')
```

Provide the graph below.

Step Response with PD Control ($K_p = 4, T_d = 0.2$)



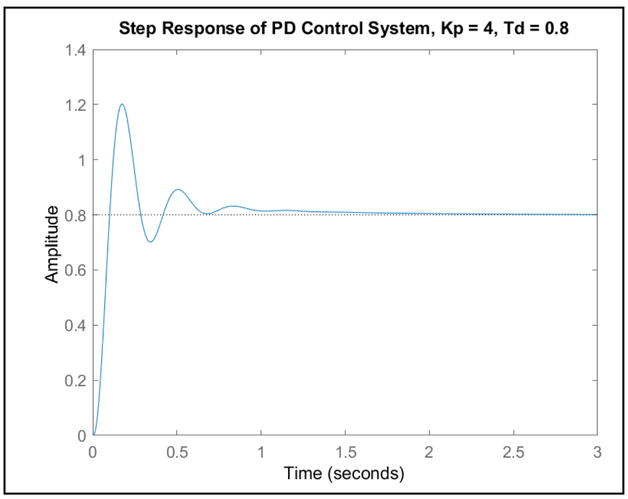
11. Determine the required time-domain performance characteristics of the closed-loop system using the step plot and insert them in the first row of **Table 2**.

Table 2: Control System with PD Control

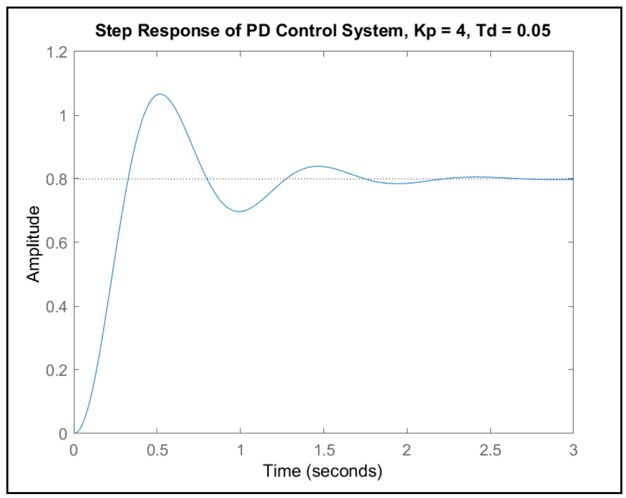
	Rise Time	Percentage of Overshoot	Settling Time	Steady-state Error
$T_d = 0.2, K_p = 4$	0.156	17.9%	0.8	0.2%
$T_d = 0.8, K_p = 4$	0.0634	50.2%	0.969	0.2%
$T_d = 0.05, K_p = 4$	0.213	33.3%	1.66	0.2%

12. Vary the derivative time constant to $T_d = 0.8$ and $T_d = 0.05$ in your **MATLAB** script file. Run the code and provide the graphs below.

PD Control ($K_p = 4, T_d = 0.8$)



PD Control ($K_p = 4, T_d = 0.05$)



13. Determine the required time-domain performance characteristics of the closed-loop system using the step plot and insert them in the second and third rows of **Table 2**.

14. Explain how does the change of the **derivative time constant** T_d affects the performance of the closed-loop system? Which PD controller do you like the best? Why?

It doesnt change the steady state error so id like to choose this one because of its low error rate

PART 3: Proportional-Integral-Derivative (PID) Control

The **transfer function** of a **PID controller** is:

$$u(t) = K_p\left[e(t) + \frac{1}{T_i}\int e(\tau)d\tau + T_d\frac{de(t)}{dt}\right] \quad \rightarrow \quad G_c(s) = K_p\left(1 + \frac{1}{T_i s} + T_d s\right)$$

15. Modify your **Simulink** diagram to build the following closed-loop system with **PID control**. Assign the PID controller gains as $K_p = 4,\ T_d = 0.2$ and $T_i = 1$.

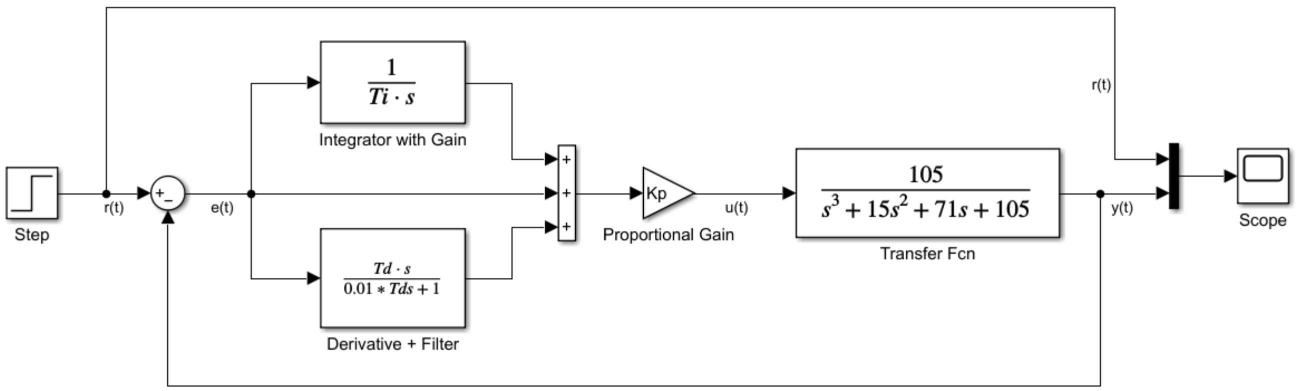


Figure 4: Closed system with PID control

16. Set the simulation to run for **6 sec**. **Run** the simulation. Check the reference input and the system output in **Scope**. Does the steady-state error of the output signal improve in comparison with the PD controller with $K_p = 4,\ T_d = 0.2$ in **Step 10**?

YES **NO**

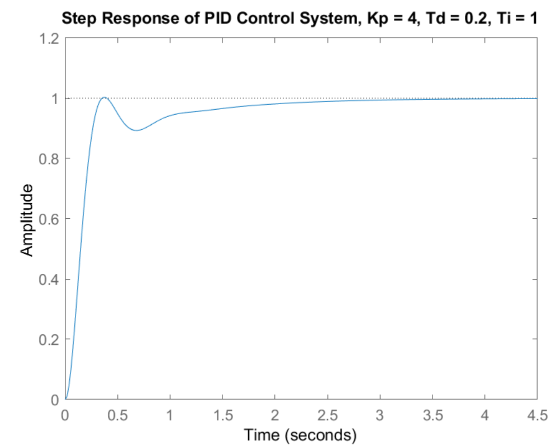
17. Create the following script code in **MATLAB** to plot the input-output signal of the closed-loop system for the **PID controller** with $K_p = 4,\ T_d = 0.2$ and $T_i = 1$.

```
G = tf([105],[1 15 71 105]);
Kp = 4;
Td = 0.2;
Ti = 1;
Gc_PID = tf(Kp*[1.01*Ti*Td 0.01*Td+Ti 1],[0.01*Ti*Td Ti 0]);
sys_OL = G*Gc_PID;
sys_CL = feedback(sys_OL,1);

figure;
step(sys_CL)
title('Step Response of PID Control System, Kp = 4, Td = 0.2, Ti = 1')
```

Provide the graph below.

Step Response with PID Control ($K_p = 4, T_d = 0.2, T_i = 1$)



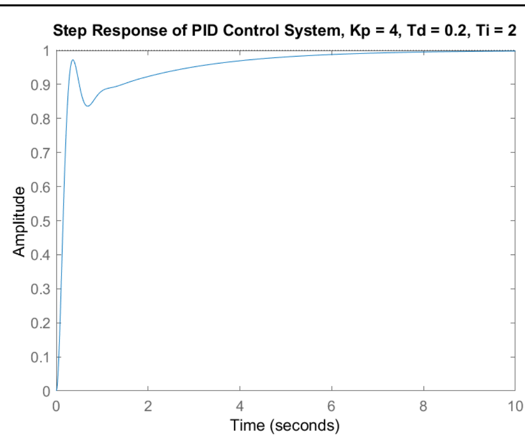
18. Determine the required time-domain performance characteristics of the closed-loop system using the step plot and insert them in the first row of **Table 3**.

Table 3: Control System with PID Control

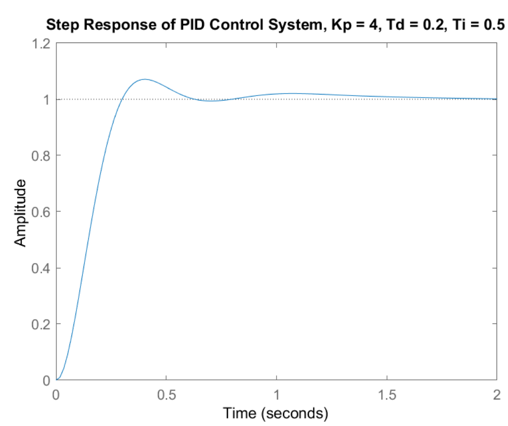
	Rise Time	Percentage of Overshoot	Settling Time	Steady-state Error
$T_i = 1, K_p = 4, T_d = 0.2$	0.208	0.325%	1.95	0%
$T_i = 2, K_p = 4, T_d = 0.2$	0.216	0	4.93	0%
$T_i = 0.5, K_p = 4, T_d = 0.2$	0.197	7.12%	1.12	0%

19. Vary the integral time constant to $T_i = 2$ and $T_i = 0.5$ in your **MATLAB** script file. Run the code and provide the graphs below.

PID Control ($K_p = 4, T_d = 0.2, T_i = 2$)



PID Control ($K_p = 4, T_d = 0.2, T_i = 0.5$)



20. Determine the required time-domain performance characteristics of the closed-loop system using the step plot and insert them in the second and third rows of **Table 2**.

21. Explain how does the change of the **integral time constant** T_i affects the performance of the closed-loop system? Which PID controller do you like the best? Why?

Increase Ti to have less overshoot, this has a steady state has a 0% which is the best.