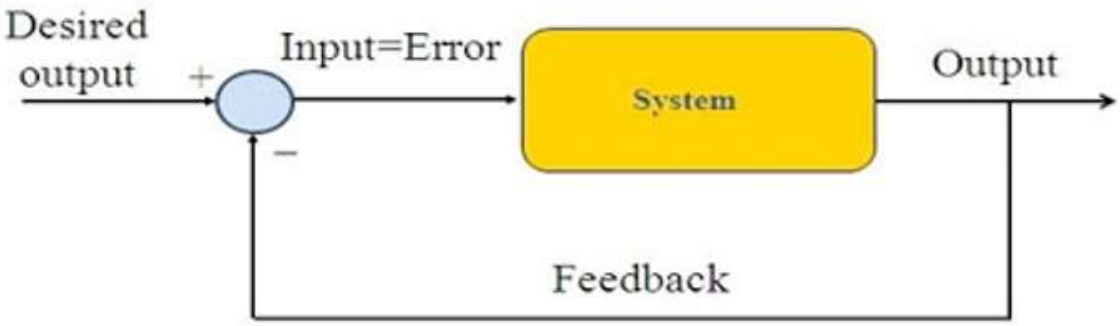
MEEM 4775 Project 2 Report

DESIGN OF A CONTROL SYSTEM



Created & Submitted by: Vrushaketu Mali

02-Nov-2020

Table of Contents

Chapter 1. Introduction

Most of the dynamic systems which we consider in the practice are having a compensator or controller. The role of controller is to make the system follow a specific command as well as to avoid any disturbance. When the system is coupled with the controller then the feedback is necessary in order for controller to determine what action it needs to take. For example, a controller may reduce the input if the actual output is more than required output and may increase the input if the actual output is less than the required output. Therefore, a system with the compensator or controller is called as closed loop system with feedback mechanism. There are several types of the control system such as PID, State Feedback, LQR. The role of the control engineer is to design a control system, specifically design the control system gains such that closed loop system will have desired response characteristics.

Chapter 2. Summary

This project aims at developing a control system for the given transfer function of a plant considering the sensor dynamics. Thus, the project involves designing the control system parameters for required time domain characteristics of the closed loop system, writing a MATLAB script and building a mathematical model for simulation of the system. There are two control systems designed for the given plant and then the performance of both the controllers is compared.

Accomplishments:

1. In this project, first of all we have analyzed the system's stability and open loop system response.
2. Then, we have designed the PID control system for required time domain characteristics of the closed loop system response.
3. As per the first stage design results, the PID controller is then tuned for meeting required specifications.
4. Parallelly, we have designed state feedback controller and then it is tuned for meeting the required specifications.
5. As mentioned in the project statement, there are incentives for the good controller design as per the system response. Therefore, the incentives for each controller design are calculated and the compared for determining the good controller between two.

Chapter 3. Project/Problem Statement

Consider a dynamic system whose transfer function is

$$\frac{Y}{U} = \frac{25}{s^2 + 4s + 25}$$

where Y is position and U is force. You must design a position control system where the sensor used to measure position can be modeled as a first order transfer function with a dc gain of one and a time constant of 0.1 second. The actuator is effort limited and can produce forces between 2.5 Newtons. The mandatory closed loop requirements are:

1. steady state error due to a unit step input is < 5%
2. %Mp < 5%
3. ts;1% < 3 seconds

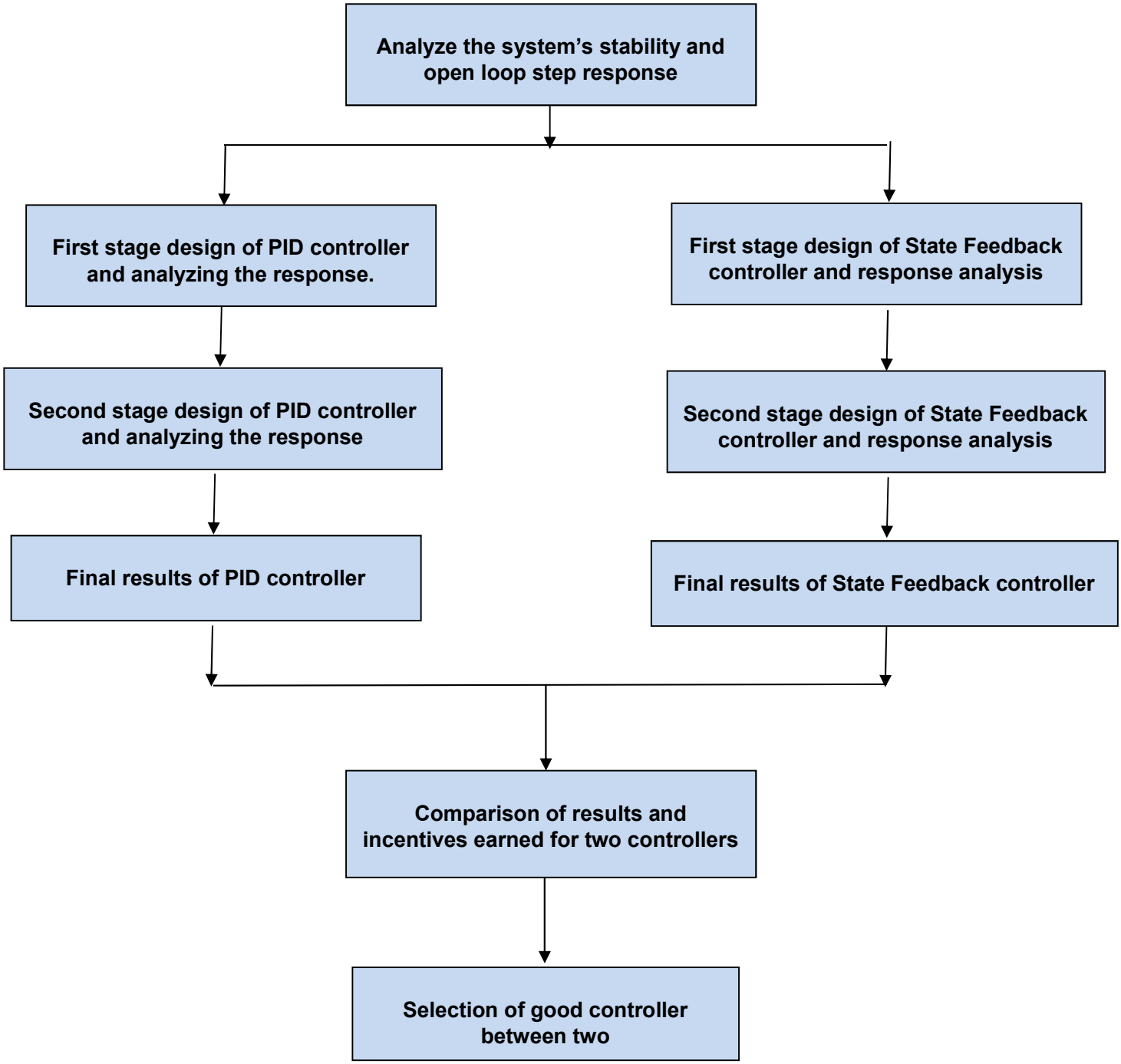
You have free rein to decide on command filters, control system topology, and the control law. Like most things in life, there are incentives for having a design that exceeds the requirements:

1. $10 for each percentage decrease from the requirement: %Mp < 5%
2. $20 for each second reduction in the requirement: ts;1% < 3 seconds
3. $20 for each percentage improvement in steady state error.
4. There are larger actuators that can be used, but there is a price penalty of $50 per Newton beyond the 2.5 Newtons with a maximum of 4 Newtons.
5. There are both better and worse sensors that can be used in terms of their time constant. The fastest sensor possible has a time constant of 0:04 seconds. The cost for faster sensors is $140 per second and the savings for using slower sensors is $35 per second.
6. There is a small incentive for having a control system that doesn't over tax the system's microprocessor controller. A time step of 0.01 seconds is free. The incentive/penalty for adjusting the update rate is $300 per second with a minimum update rate of 0.001 seconds.

Write a MATLAB function that calculates the $ incentive given the components used and the requirements achieved.

2. Design a controller to meet the minimum requirements using the standard sensor and actuator. In your report, document your design via a block diagram and plots illustrating its closed loop performance, including the actuator response.
3. Continue working on your control design to maximize your $ incentive. In your report, document this new design, as you did above, and also your incentive.

Chapter 4. Project Approach

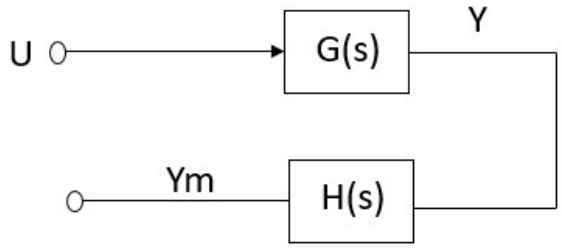


Chapter 5. Open Loop System Stability & Step Response

Before running into designing control system and analyzing the closed loop system response, we first analyze the open loop system stability and its step response.

5.1 Block diagram

In order to determine the stability of system, we form the characteristic equation of system considering the sensor dynamics. For that, we need to take the reference of block diagram which is shown below.



The plant transfer function is given in the project statement as

$$G(s) = \frac{Y}{U} = \frac{25}{s^2 + 4s + 25}$$

The first order transfer function for sensor with DC gain 1 and time constant of 0.1 seconds is given as

$$H(s) = \frac{Y_m}{Y} = \frac{1}{0.1\,s + 1}$$

Considering the block diagram above, we can write the transfer function from input to measured output as

$$G(s) * H(s) = \frac{Y_m}{U} = \frac{25}{(s^2 + 4s + 25) * (0.1\,s + 1)}$$

$$\frac{Y_m}{U} = \frac{25}{(s^2 + 4s + 25) * (0.1\,s + 1)}$$

$$\frac{Y_m}{U} = \frac{25}{(0.1\,s^3 + 1.4\,s^2 + 6.5\,s + 25)}$$

5.2 Stability Analysis

In order to determine the stability of the system, we need to find the roots of the characteristic equation of the system.
As per the derived transfer of the system, the characteristic root equation is

$$0.1\ s^3 + 1.4\ s^2 + \ 6.5\ s + 25 = 0$$

After running the below command in MATLAB

Open loop system stability

```
% Roots for the stability
p = [0.1 1.4 6.5 25];  % define the coefficients
r = roots(p); % roots of the equation
```

We get the roots of the characteristic equation as

```
r = 3×1 complex
   -10.0000 + 0.0000i
    -2.0000 + 4.5826i
    -2.0000 - 4.5826i
```

All the roots of the characteristic equation are negative that means all the poles of the open loop system are in LHP of S plane. Thus, the system is stable.

5.3 Step response

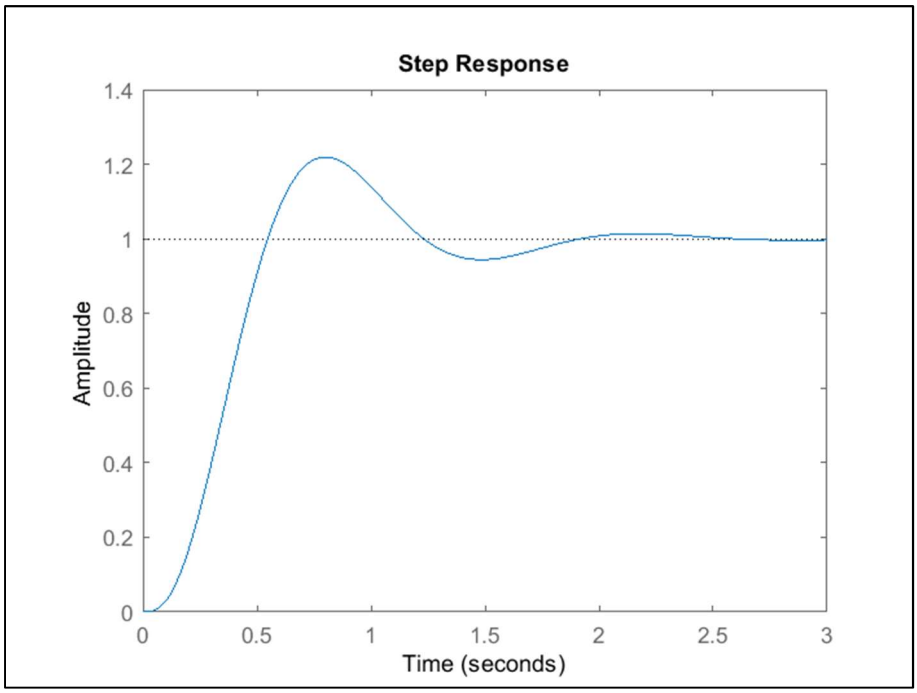
Now plotting the step response of open loop system for which we enter below code in MATLAB script.

Open loop system step response

```
%% Plant & sensor Parameters

% Plant
numG = 25;  % Numerator of plant transfer function
denG = [1 4 25];  % Denominator of plant transfer function

G = tf(numG,denG);  % transfer function of the plant

% Sensor
Tau = 0.1;  % Time constant of sensor
numH = 1;  % Numerator of sensor transfer function
```

```
denH = [Tau, 1];  % Denominator of sensor transfer function
H = tf(numH, denH);  % transfer function of the sensor

TF_OL = G*H;  % Create the transfer function of open loop

step(TF_OL)  % create the step response plot
```

We get below plot and information for the step response



Overshoot	21.92 %
Settling Time	1.77 s

The overshoot in the system is 21.92% which is way out of our requirement. Therefore, we need to design the controller for this system.

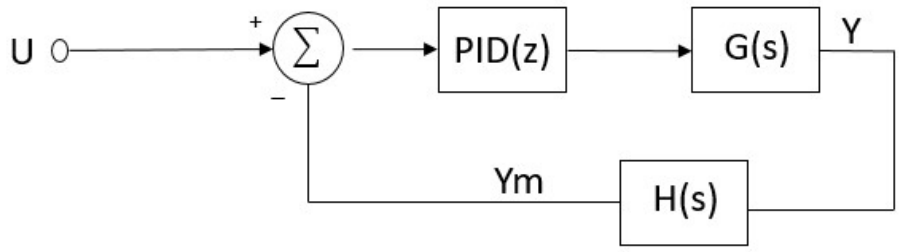
Chapter 6. Design of PID Controller

A proportional-integral-derivative controller (PID controller) is a control loop feedback mechanism. A PID controller calculates an error value as the difference between a measured process variable and a desired setpoint. The controller attempts to minimize the error by adjusting the process through use of a manipulated variable.

The PID controller algorithm involves three separate constant parameters, and is accordingly called three-term control: the proportional, the integral and derivative values, denoted P, I, and D. Simply put, these values can be interpreted in terms of time: P depends on the present error, I on the accumulation of past errors, and D is a prediction of future errors, based on current rate of change. The weighted sum of these three actions is used to control the input to the plant in order to achieve the desired closed loop results.

6.1 Block diagram:

The block diagram of a closed loop system using discrete PID controller is given as

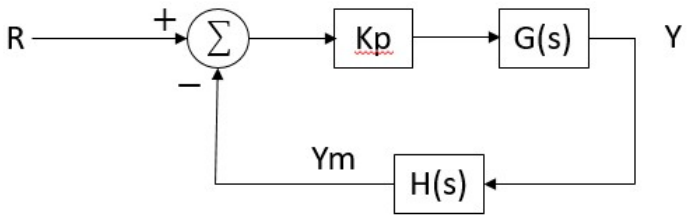


6.2 First Stage Design:

6.2.1 Proportional Only Controller:

We will first start with the proportional controller.

With proportional controller only, the block diagram of the system becomes



The transfer function for this system with block diagram reduction technique can be written as

$$\frac{Y_m}{U} = \frac{\frac{25\ K_p}{(s^2 + 4s + 25)}}{1 + \frac{1}{(0.1\ s + 1)} * \frac{25\ K_p}{(s^2 + 4s + 25)}}$$

$$\frac{Y_m}{U} = \frac{25\ K_p\ (0.1\ s + 1)}{(0.1\ s^3 + 1.4\ s^2 + \ 6.5\ s + 25 + \ 25\ K_p)}$$

<u>Calculating Ultimate gain K_u</u>

Using Zeigler Nichols Method for controller tuning, We first determine the ultimate controller gain.

The ultimate controller gain is defined as the proportional gain of the control system at which the system becomes marginally stable.

We can calculate the ultimate gain by doing Routh Array analysis for stability.

As derived from the transfer function above, the characteristic root equation for proportional controller only is $(0.1\ s^3 + 1.4\ s^2 + \ 6.5\ s + 25 + \ 25\ K_p)$

Write the Routh Array for this polynomial

$s^3:$	0.1	6.5
$s^2:$	1.4	$25 + 25K_p$
$s:$	a	0
$1:$	b	0

Calculating the values of a and b

$$a = \frac{(1.4 * 6.5) - 0.1 * (25 + \ 25\ K_p)}{1.4} = 4.71 - 1.78\ K_p$$

$$b = \frac{\left(4.71 - 1.78\ K_p\right) * (25 + \ 25\ K_p)}{\left(4.71 - 1.78\ K_p\right)} = (25 + \ 25\ K_p)$$

In order for system to be stable, the values of ‘a’ and ‘b’ must be greater than 0.

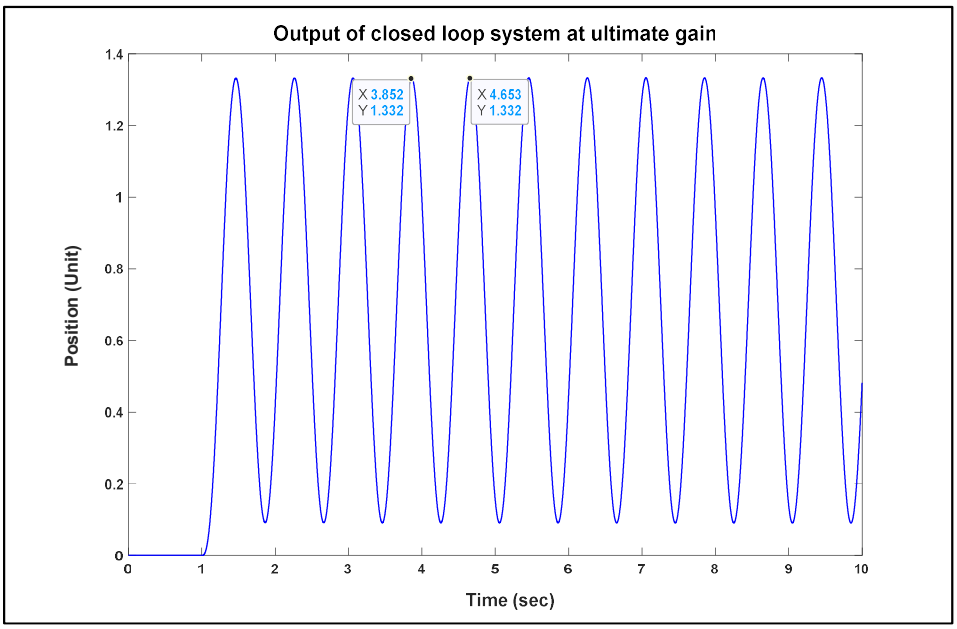
$4.71 - 1.78\,K_p > 0$

Thus, $K_p < 2.64$

Thus, at value of K_p at 2.64 the system will become marginally stable.

The ultimate controller gain K_u is 2.64.

Below graph shows that at proportional gain of 2.64, the system response becomes oscillatory with constant amplitude and frequency. Thus, system becomes marginally stable.



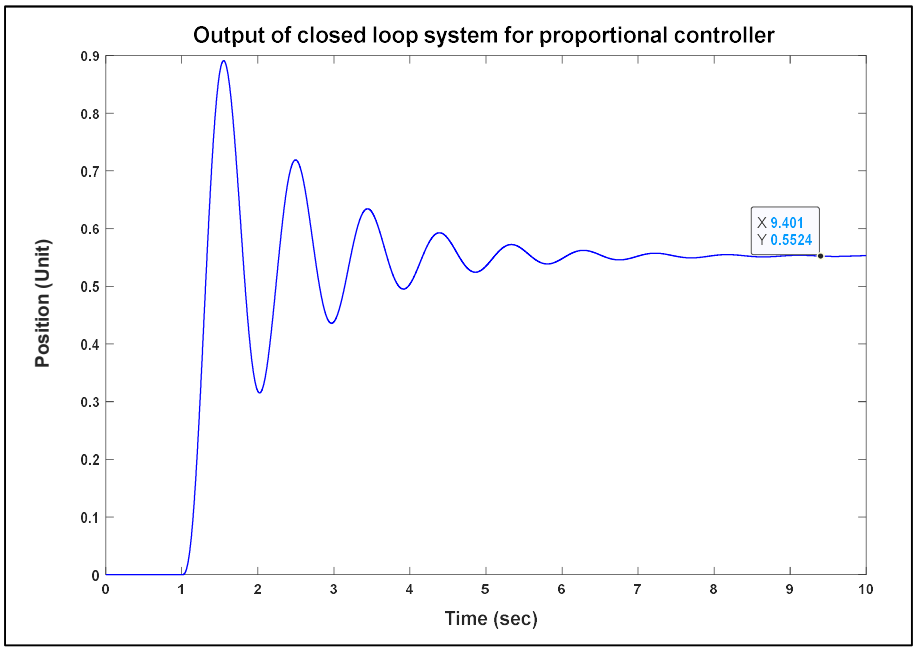
And the ultimate time period is 4.45 – 3.85 = 0.8 seconds.

As per Zeigler Nichols tuning method,

For proportional only controller, the gain K_p is defined as $K_p = 0.5 * K_u$

$$K_p = 0.5 * 2.64 = 1.32$$

When we use this proportional gain in the controller, then the system response can be plotted as



From the above figure, we can see that proportional controller only is not meeting the required specification of time domain response. Increasing the proportional gain furthermore will cause the steady state error to decrease but it will increase the oscillations and thus overshoot. Decreasing the proportional gain will cause the steady state error to increase again. Therefore, we cannot use proportional only controller. We can add the integral term and use PI controller however, using the integral gain also causes the overshoot to increase and also increases the settling time. We need to damp the oscillations as well. Therefore, we need to add integral gain for taking care of settling time and derivative gain in order to take care of the overshoot and settling time.

Proportional only controller is failed. We need to add integral as well as derivative gain.

6.2.2 Proportional Integral Derivative Controller (PID):

The block diagram of the closed loop system with PID controller is already shown. The transfer function of the closed loop system with PID controller can be given as

$$\frac{Y_m}{U} = \frac{\left(K_p + \frac{K_i}{s} + K_D\, s\right) * \frac{25}{(s^2 + 4s + 25)}}{1 + \frac{1}{(0.1\, s + 1)} * \left(K_p + \frac{K_i}{s} + K_D\, s\right) * \frac{25}{(s^2 + 4s + 25)}}$$

Which after resolving becomes

$$\frac{Y_m}{U} = \frac{2.5\ K_D\ s^3 + (2.5\ K_p + 25\ K_D)s^2 + (2.5\ K_I + 25\ K_P)s + 25\ K_I}{0.1\ s^4 + 1.4\ s^3 + (6.5 + 25\ K_D)\ s^2 + (25 + 25\ K_p)\ s + 25\ K_i}$$

As per Zeigler Nichols tuning method,

For PID controller, the gains are defined as

$K_p = 0.6 * K_u$

$K_I = 0.5 * P_u * K_p$

$K_D = 0.125 * P_u * K_p$

We have already calculated the ultimate gain K_u as 2.47 and the ultimate time period is 0.8 seconds.

From this we can calculate the starting values of the gains

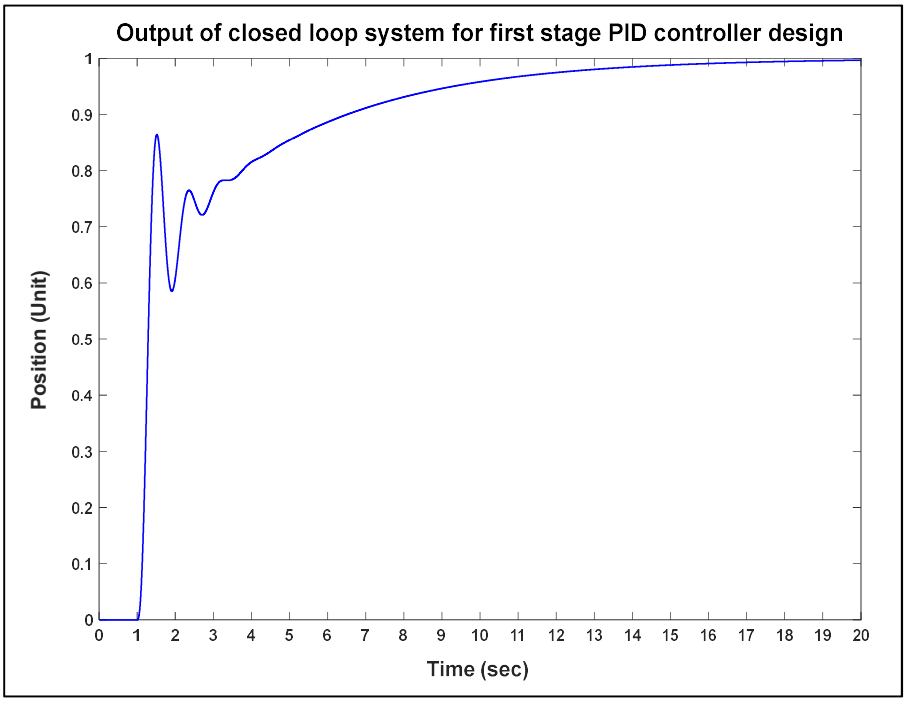
$K_p = 0.6 * 2.47 = 1.482$

$K_I = 0.5 * 0.8 * 1.482 = 0.5928$

$K_D = 0.125 * 0.8 * 1.482 = 0.1482$

6.2.3 Results of first stage of Proportional Integral Derivative Controller (PID):

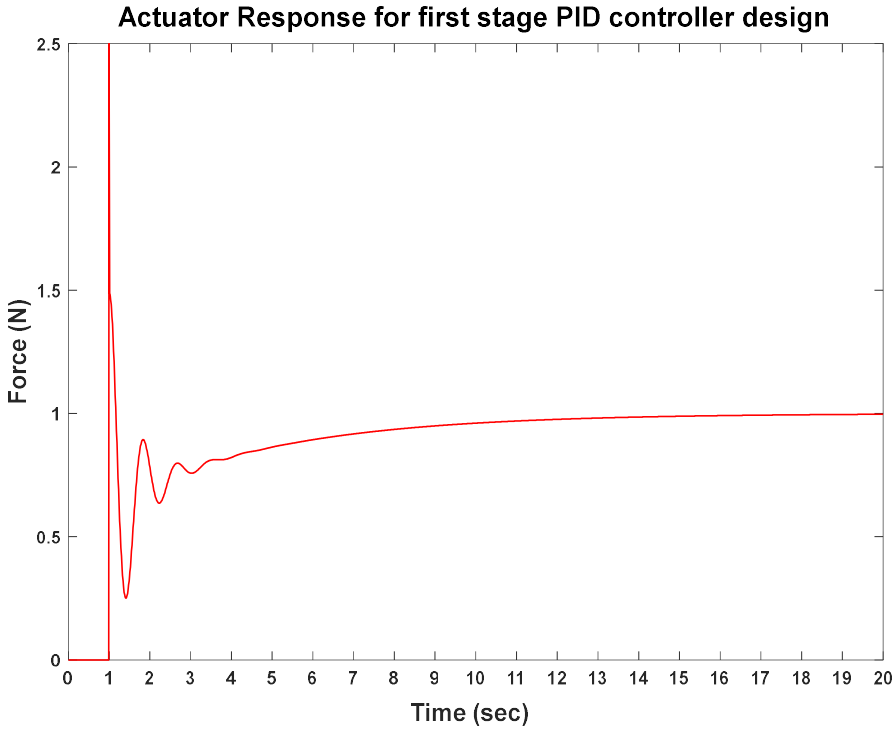
With these gain values, if we simulate the system then we get the below plot and characteristics of the closed loop system response.



The above figure shows the output of closed loop system at starting point of PID controller gains by Zeigler Nichols method applied to this system. Definitely, the system response is out of specification.

Overshoot	0 %
Settling Time	14.57 s
Steady State error	0.34 %
Max Actuator Force	**2.5N**
Actuator Saturated	**Yes**

This means that the overshoot and steady state error is within the specification but the settling time is way out of control.



The above figure shows the actuator response to the first stage design of the PID controller. The actuator is saturated at 2.5N.

First stage of PID controller is failed. We need to tune the parameters.

6.3 Second Stage Design (Improvement for finalizing the design):

In previous step, we saw that at starting point of the gains calculated by Zeigler Nichols method, the system is not responding as per the required specifications.

The settling time is very large. In order to decrease the settling time, we need to increase the derivative gain.

So the multiple iterations are performed with hit and miss approach in order to reach to final values of the gains. Out of those, last three iterations are documented below.

As asked in the project report, the performance of the controller is then calculated in terms of the incentives. A separate MATLAB function is written for calculating the incentives. In the iterations below, the information about the characteristics of the closed loop system and the incentives gained for each iteration are mentioned.

6.3.1 First Iteration of the second stage design

Control gains values:

K_p	3.458
K_I	1.9365
K_D	0.5187

System Characteristics:

Overshoot	0 %
Settling Time	**6.87 s**
Steady State error	0.002 %
Max Actuator Force	**2.5N**
Actuator Saturated	**Yes**

Incentives:

Overshoot Incentive ($)	50
Settling Time Incentive ($)	**-77**
Steady State error Incentive ($)	100
Actuator Incentive/penalty ($)	0
Sensor Time constant Incentive/penalty ($)	0
Controller update rate Incentive/penalty ($)	0
Total Incentive ($)	**73**

6.3.2 Second Iteration of the second stage design

Control gains values:

K_p	3.705
K_I	2.6676
K_D	0.7781

System Characteristics:

Overshoot	0 %
Settling Time	**4.15 s**
Steady State error	2.72e-05 %
Max Actuator Force	**2.5N**
Actuator Saturated	**Yes**

Incentives:

Overshoot Incentive ($)	50
Settling Time Incentive ($)	**-23**
Steady State error Incentive ($)	100
Actuator Incentive/penalty ($)	0
Sensor Time constant Incentive/penalty ($)	0
Controller update rate Incentive/penalty ($)	0
Total Incentive ($)	**127**

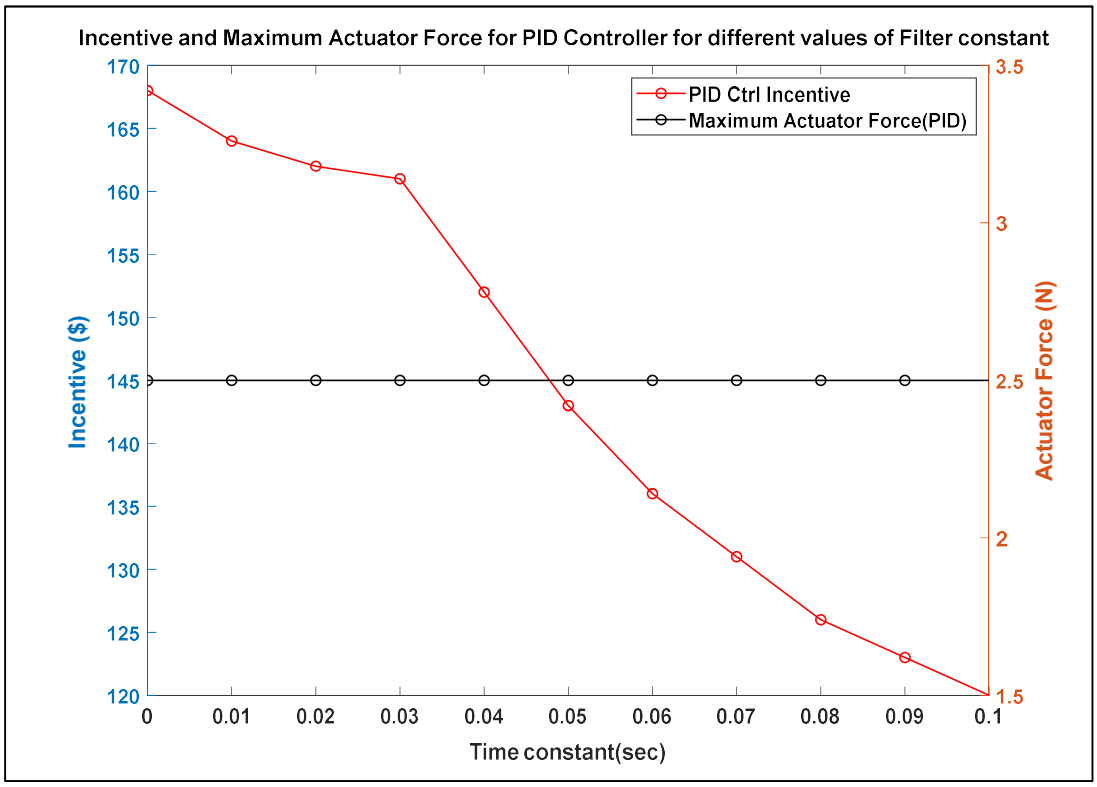
6.3.3 Third Iteration of the second stage design (finalized design)

From the previous two iterations, we noticed that even we improved on time domain characteristics, the actuator is still saturating. Therefore, we need to filter the input signal in order to see its effect on the actuator response.

The first order transfer function of input filter is $\frac{1}{Tau\ s+1}$

We will do the experiment of analyzing the effect of filter time constant from 0 to 0.1 seconds on the incentives earned and the maximum actuator force.

Below graph shows the variation analysis for incentives and actuator force for different values of filter time constant.



The above figure shows how the incentives of PID controller and the actuator force varies for different values of input filter time constant. We can see that the maximum actuator force does not change for any value of the time constant. Therefore, it does not make any sense to use the input filter for PID controller and we will just continue with working on the improvement of the closed loop system response.

For third iterations, below are the results.

Control gains values:

K_p	4.2731
K_I	3.2476
K_D	1.0042

System Characteristics:

Overshoot	0.1327 %
Settling Time	2.0716 s
Steady State error	4.42e-07 %
Max Actuator Force	**2.5N**
Actuator Saturated	**Yes (for short duration)**

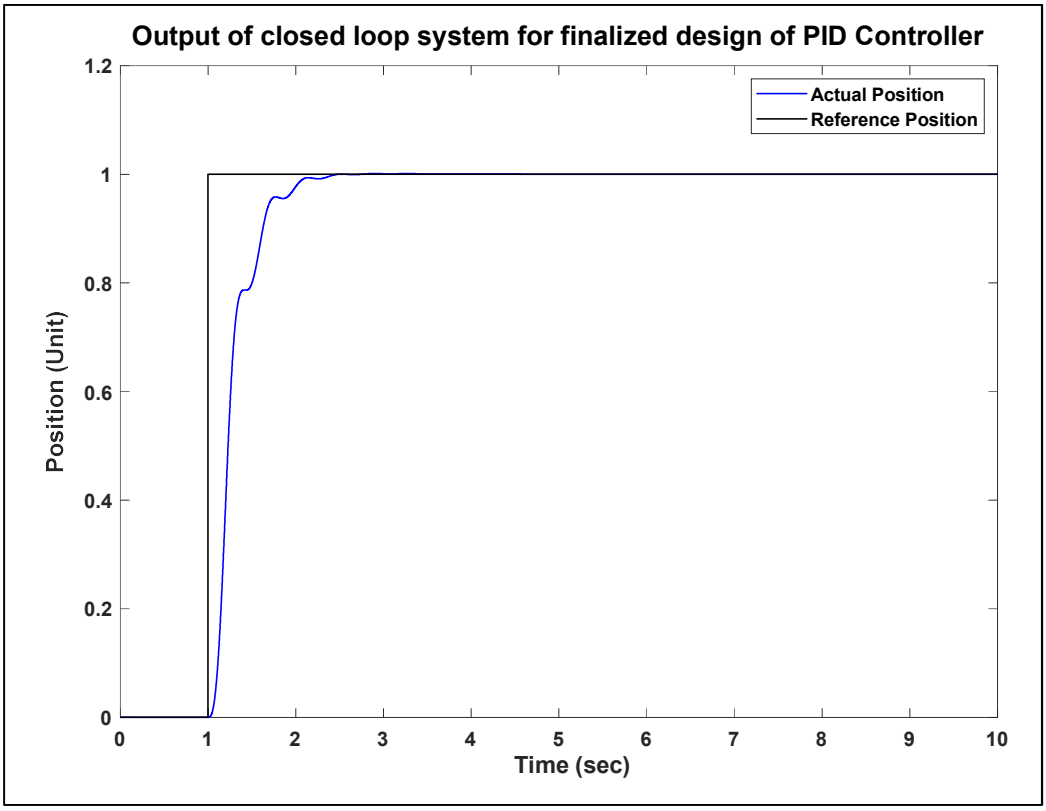
Incentives:

Overshoot Incentive ($)	49
Settling Time Incentive ($)	19
Steady State error Incentive ($)	100
Actuator Incentive/penalty ($)	0
Sensor Time constant Incentive/penalty ($)	0
Controller update rate Incentive/penalty ($)	0
Total Incentive ($)	**168**

Third iteration of second stage design of PID Controller is meeting the requirements.

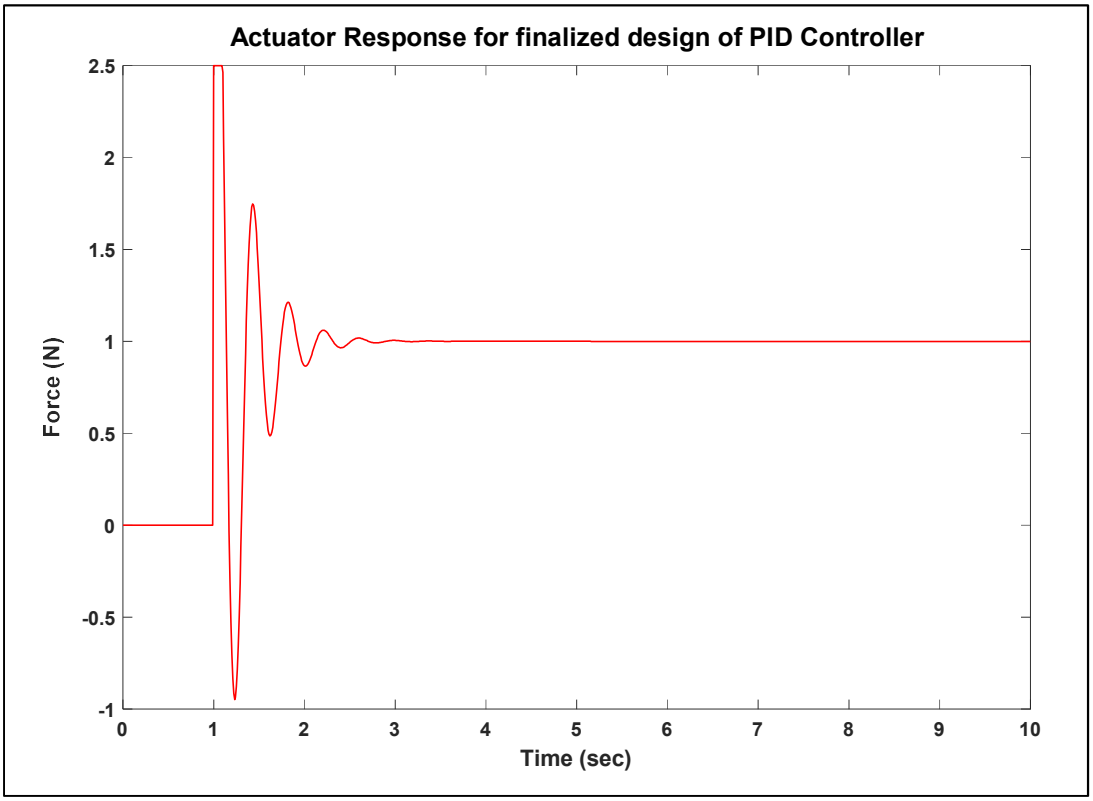
The plots for the final design are as below.

A) Output response with respect to the reference input and time:



The above figure shows the output response of the closed loop system with finalized design of PID controller (with gains mentioned in third iteration). The system is meeting the characteristic response criteria.

B) Actuator response with respect to time:



The above figure shows the actuator response of the closed loop system with finalized design of PID controller (with gains mentioned in third iteration). Initially the actuator reaches to its saturation value and then cycles back and forth and then settles down at 1 N. In order to have good response, the cycling of the actuator and approaching the saturated value is not desirable.

6.3.4 Comparison of three iterations of the second stage PID design

	% Overshoot	Settling time (s)	% Steady state error	Incentive ($)
First Iteration	0	6.87 s	0.002%	**73 $**
Second Iteration	0	4.15 s	2.72e-5%	**127 $**
Final Design	0.13 %	2.07 s	4.42e-7%	**168 $**

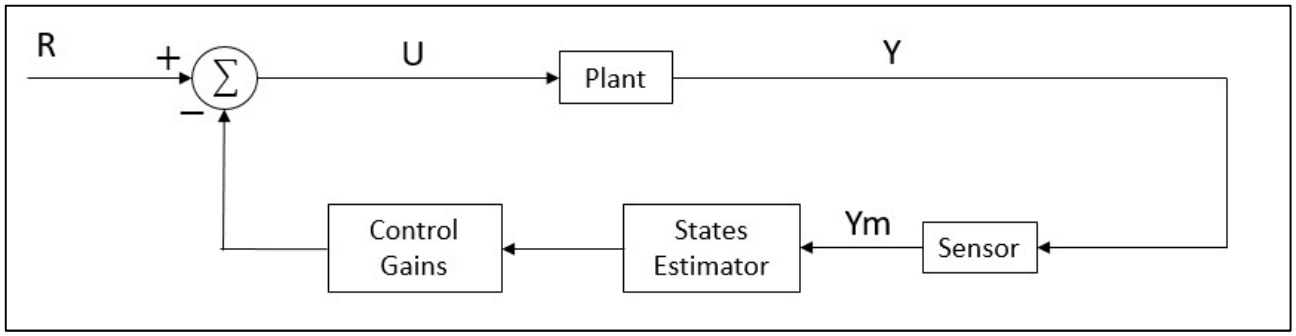
Chapter 7. Design of State Feedback Controller

The state feedback controller is a closed loop mechanism where the states of the plant or system under consideration are fed back to the input side in order to place the poles of the closed loop system at desired location so that we can achieved the desired response characteristics of the closed loop system.

The state feedback controller is governed by the controller gain matrix where number of the gain elements are equal to the number of states. If gain matrix K can be defined such that the eigen values of (A-B.K) matrix are at locations such that system is stable then the regulation of K is solved.

7.1 Block diagram:

The block diagram of a closed loop system using state feedback controller is given as



7.2 First Stage Design:

7.2.1 State space model of the given system:

We have been provided the transfer function of the open loop system which is re-written below.

$$\frac{Y}{U} = \frac{25}{s^2 + 4s + 25}$$

$$\frac{Ym}{Y} = \frac{1}{0.1\,s + 1}$$

Converting from transfer function to differential equation, we get

$$\ddot{y} + 4\,\dot{y} + 25y = 25\,u$$

$$0.1\,\dot{Ym} + Ym = 1$$

Highest derivative of y = 2, Highest derivative of Ym = 1

So, order of the system = 2 + 1 =3

Therefore, we need 3 state variables.

Let's define states input and output as below

$x_1 = y$
$x_2 = \dot{y}$
$x_3 = y_m$

$u = u$
$y = y_m$

So, the derivatives of the states would be

$\dot{x}_1 = \dot{y}$

$\dot{x}_2 = \ddot{y} = -4x_2 - 25\,x_1 + 25\,u$

$\dot{x}_3 = \dot{y_m} = 10 - 10\,y_m$

We can create the state space matrices by partially differentiating the derivatives equation with respect to each state

A = $\begin{bmatrix} 0 & 1 & 0 \\ -25 & -4 & 0 \\ 10 & 0 & -10 \end{bmatrix}$ B = $\begin{bmatrix} 0 \\ 25 \\ 0 \end{bmatrix}$ C = $\begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$ D = 0

We do not have access to the states from the transfer function. Therefore, we will use Kalman filter as a state estimator and will use those states with control gain as a feedback to the system in order to compute the error.

7.2.2 Check controllability:

In order to design a state feedback controller, a necessary and sufficient condition for the arbitrary pole placement is that the system under consideration must be controllable. Therefore, we will check the pair of A & B matrices derived above are controllable through MATLAB 'ctrb' command and calculated the rank of the controllability matrix.

The rank of the controllability matrix is 3 and thus system is completely controllable. So now we can proceed with the pole placement.

7.2.3 Initial Pole Placement:

The trickiest part is to determine the pole location of the closed loop system in order to estimate the gains of the controller.

As there are three states in the system considering sensor dynamics, the characteristic equation of the closed loop system with state feedback controller will be a polynomial of third order.

The third order polynomial can be represented as $(s+ \propto)(s^2 + 2\,\zeta\,\omega_n s + \omega_n^2)$

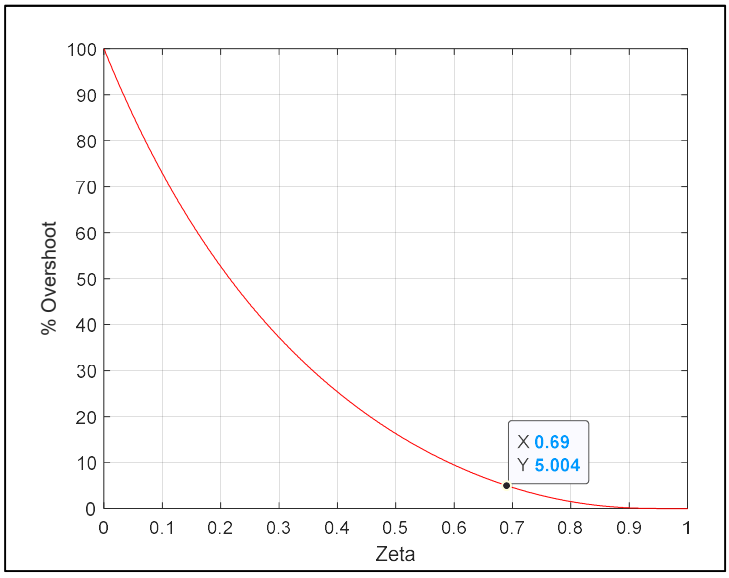
So the poles will be $-\propto,\ -\zeta\,\omega_n + \omega_n\sqrt{\zeta^2 - 1}$ & $-\zeta\,\omega_n - \omega_n\sqrt{\zeta^2 - 1}$

The closed loop system requirements are Overshoot less than 5% and the settling time less than 3 seconds.

We know the formula for overshoot as below.

$$M_p = e^{\frac{-\pi\,\xi}{\sqrt{1-\xi^2}}}$$

If we plot the graph of % overshoot against ζ then we get following



Thus, for less than 5% overshoot, the damping ratio must be greater than 0.6

We know the formula for settling time (1%) as below.

$$T_s = \frac{4.6}{\zeta\,\omega_n}$$

Let' select ζ as 0.8 and settling time as 3 seconds, then the minimum value of ω_n will be 1.92 rad/sec. Let' select the value of ω_n as 4.

Then select arbitrarily the value of $\propto$ as 8.

Then all three poles would be at $-8,\ -3.2+\ 2.4\,i\ \&\ -3.2-\ 2.4\,i$

Placing these poles using place command in MATLAB we can estimate the gains and then we can simulate the system with state feedback.

After running the MATLAB command and simulating the system, we get the control gains as

K_1	-0.072
K_2	0.016
K_3	-0.416

7.2.4 Results of first stage of State Feedback Controller:

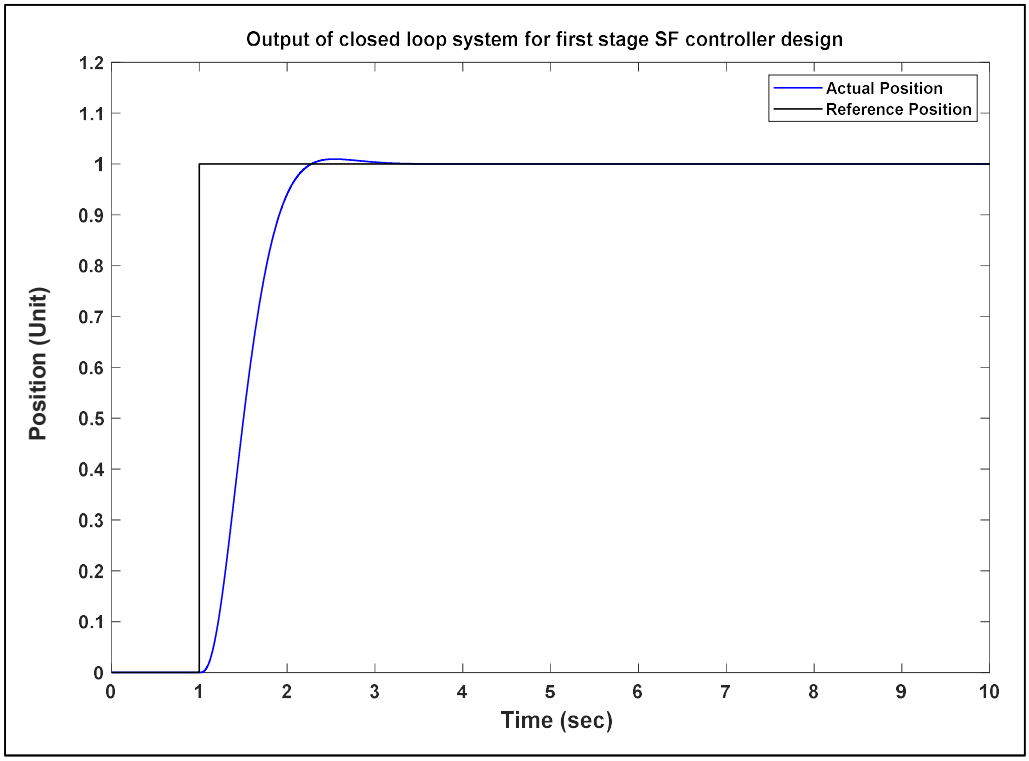
With these gain values, if we simulate the system then we get the below plot and characteristics of the closed loop system response. As asked in the project report, the performance of the controller is then calculated in terms of the incentives. A separate MATLAB function is written for calculating the incentives. In the iterations below, the information about the characteristics of the closed loop system and the incentives gained for each iteration are mentioned.

The closed loop system response characteristics as below

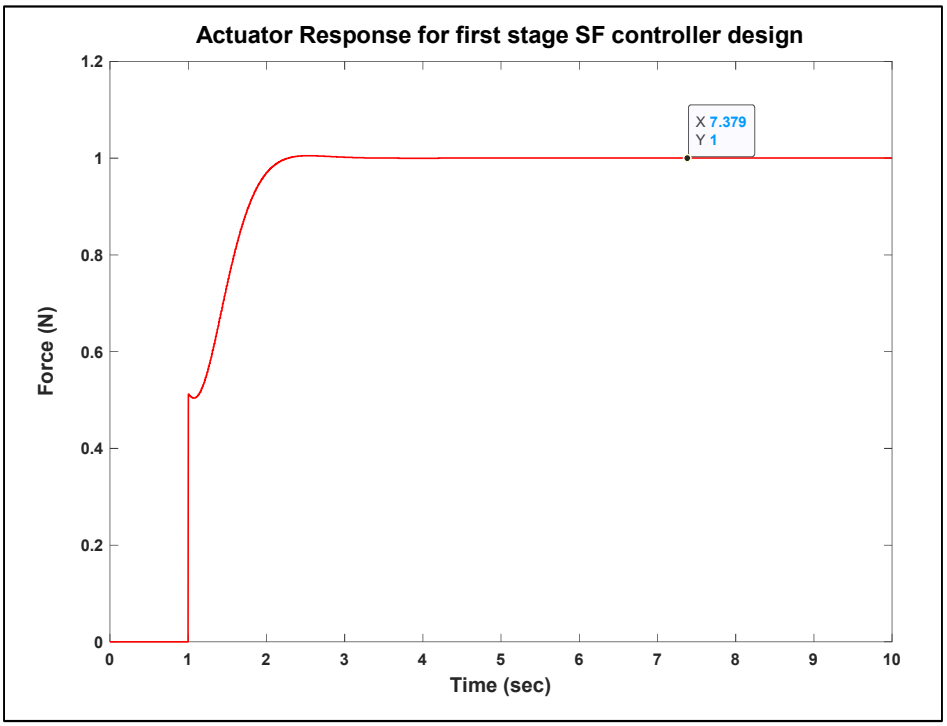
Overshoot	0.96 %
Settling Time	2.19 s
Steady State error	2.24e-11%
Max Actuator Force	1 N
Actuator Saturated	No

Overshoot Incentive ($)	40
Settling Time Incentive ($)	16
Steady State error Incentive ($)	100
Actuator Incentive/penalty ($)	0
Sensor Time constant Incentive/penalty ($)	0
Controller update rate Incentive/penalty ($)	0
Total Incentive ($)	**156**

Looking at the values, we can say that all the characteristics are within the specifications and thus the design is meeting the requirements. However, we can further tune the gains by varying the pole placement location and check for improvement.



The above figure shows the closed loop response of the system. Small overshoot is visible in the response which can be further reduced by improving the gains.



The above figure shows the actuator response to the first stage design of the SF controller. The actuator is not saturated and the maximum force value is 1N.

First stage of State Feedback controller is successful. We can still tune the parameters for improvement in the characteristics.

7.3 Second Stage Design (Improvement in the design):

In previous step, the closed loop system response is meeting all the specification. However, we can further tune the gains to maximize the incentives.

So the multiple iterations are performed with hit and miss approach for revising the pole placement in order to revise the gain values.

7.3.1 First Iteration of the second stage design

Since, we want to reduce the overshoot, we need to further increase the value of the damping ratio. Earlier we took it as 0.8. At this stage, we will not change the third pole which is -8. We will only try to change the pair of real and imaginary poles through damping ratio.

Let's increase the values of ζ & ω_n as 0.9 and 4.5 respectively.

With this, we get the poles value as $-8,\ -4.05+\ 1.96\ i$ & $-4.05-\ 1.96\ i$

After using place command, we get the control gains as

Control gains values:

K_1	-0.038
K_2	0.084
K_3	-0.314

System Characteristics:

Overshoot	0.0478 %
Settling Time	2.365 s
Steady State error	1.89e-12%

Incentives:

Overshoot Incentive ($)	50
Settling Time Incentive ($)	13
Steady State error Incentive ($)	100
Actuator Incentive/penalty ($)	0
Sensor Time constant Incentive/penalty ($)	0
Controller update rate Incentive/penalty ($)	0
Total Incentive ($)	**163**

7.3.2 Second Iteration of the second stage design

In the second iteration, let's further increase the value of damping ratio and analyze the system under overdamped case.

Let's increase the values of ζ & ω_n as 1.2 and 6.5 respectively.

With this, we get the poles value as $-8,\ -8+\ 4.5\ i$ & $-8-\ 4.5\ i$

After using place command, we get the control gains as

Control gains values:

K_1	1.89
K_2	0.4
K_3	-0.194

System Characteristics:

Overshoot	0 %
Settling Time	1.80 s
Steady State error	1.52e-12 %
Max Actuator Force	**2.5N**

Incentives:

Overshoot Incentive ($)	50
Settling Time Incentive ($)	24
Steady State error Incentive ($)	100
Actuator Incentive/penalty ($)	0
Sensor Time constant Incentive/penalty ($)	0
Controller update rate Incentive/penalty ($)	0
Total Incentive ($)	**174**

7.3.3 Third Iteration of the second stage design (finalized design)

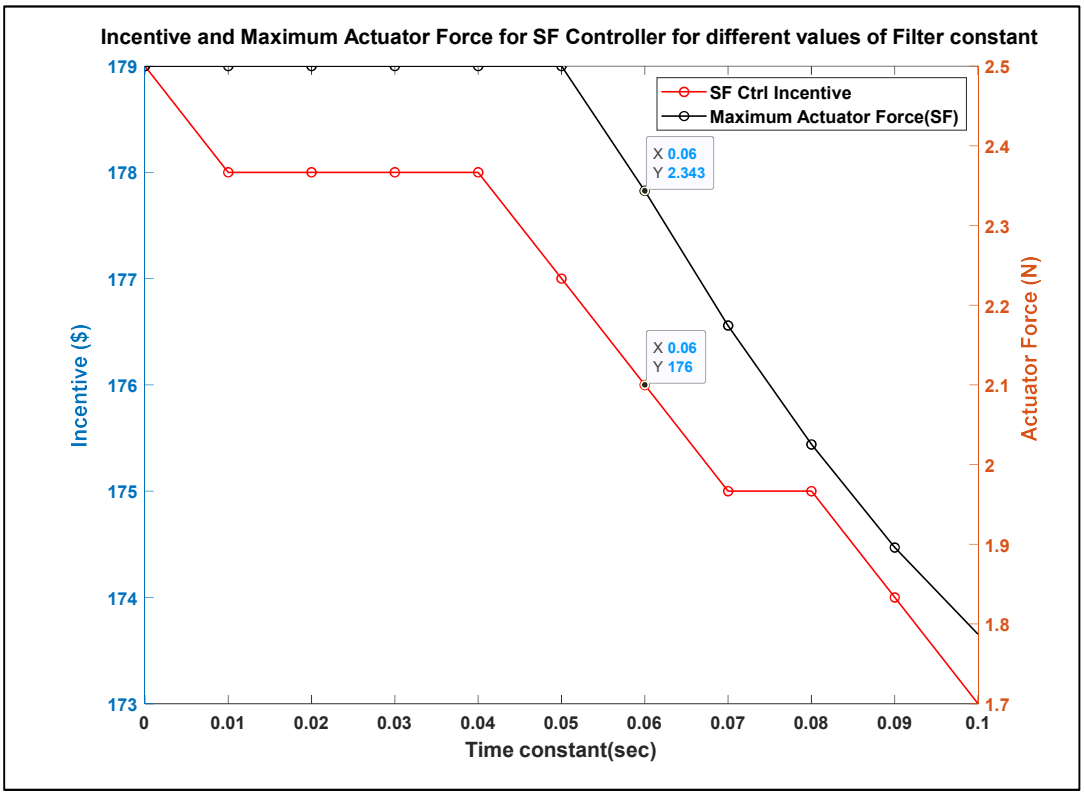
From the previous two iterations, we noticed that even we improved on time domain characteristics, the actuator is still saturating. Therefore, we need to filter the input signal in order to see its effect on the actuator response.

The first order transfer function of input filter is $\frac{1}{Tau\ s+1}$

We will do the experiment of analyzing the effect of filter time constant from 0 to 0.1 seconds on the incentives earned and the maximum actuator force.

Please note that the continuous transfer function is first converted to discrete before simulation.

Below graph shows the variation analysis for incentives and actuator force for different values of filter time constant.



The above figure shows how the incentives of SF controller and the actuator force varies for different values of input filter time constant. We can see that at time constant of 0.06 seconds, the incentive amount is decent which is 176 $ and the maximum actuator force is also not touching 2.5N. Thus, it makes sense to filter the input signal with time constant of 0.06 seconds.

The poles achieved from second iteration were -8, -8+4.5j , -8-4.5j.
In the last iteration, let's just increase the imaginary part to 10.

With this, we get the poles value as $-8,\ -8+\ 10\ i\ \ \&\ -8-\ 10\ i$

After using place command, we get the control gains as

Control gains values:

K_1	5.08
K_2	0.4
K_3	-0.832

System Characteristics:

Overshoot	0 %
Settling Time	1.69 s
Steady State error	6.21e-13%
Maximum Actuator Force (N)	2.342 N
Actuator Saturated	No

Incentives:

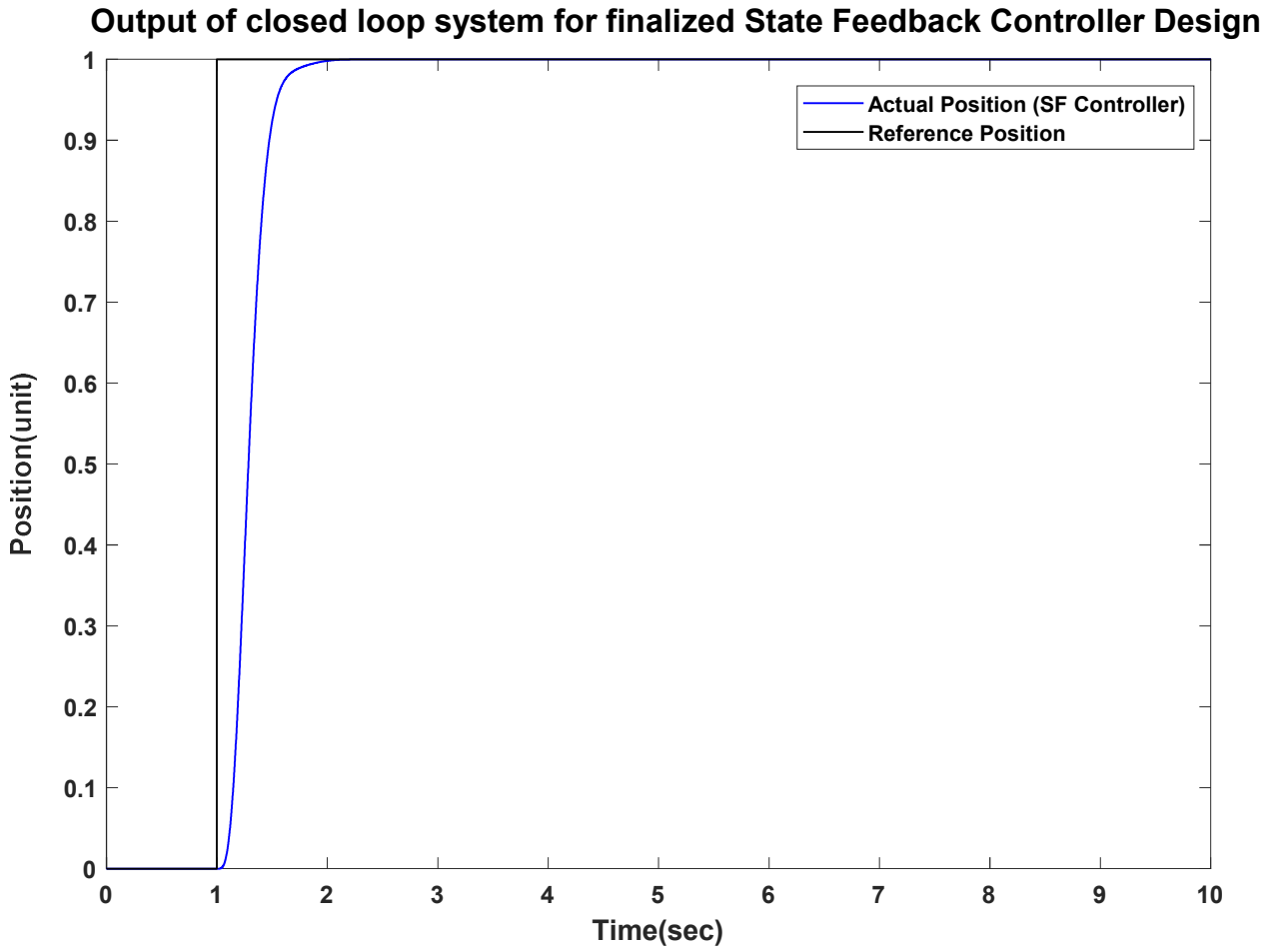
Overshoot Incentive ($)	50
Settling Time Incentive ($)	26
Steady State error Incentive ($)	100
Actuator Incentive/penalty ($)	0
Sensor Time constant Incentive/penalty ($)	0
Controller update rate Incentive/penalty ($)	0
Total Incentive ($)	**176**

After further playing with the poles, it was convinced that this is maximum value of incentive achieved. Thus, this design is finalized.

Third iteration of second stage design of SF Controller is meeting the requirements without actuator saturation.

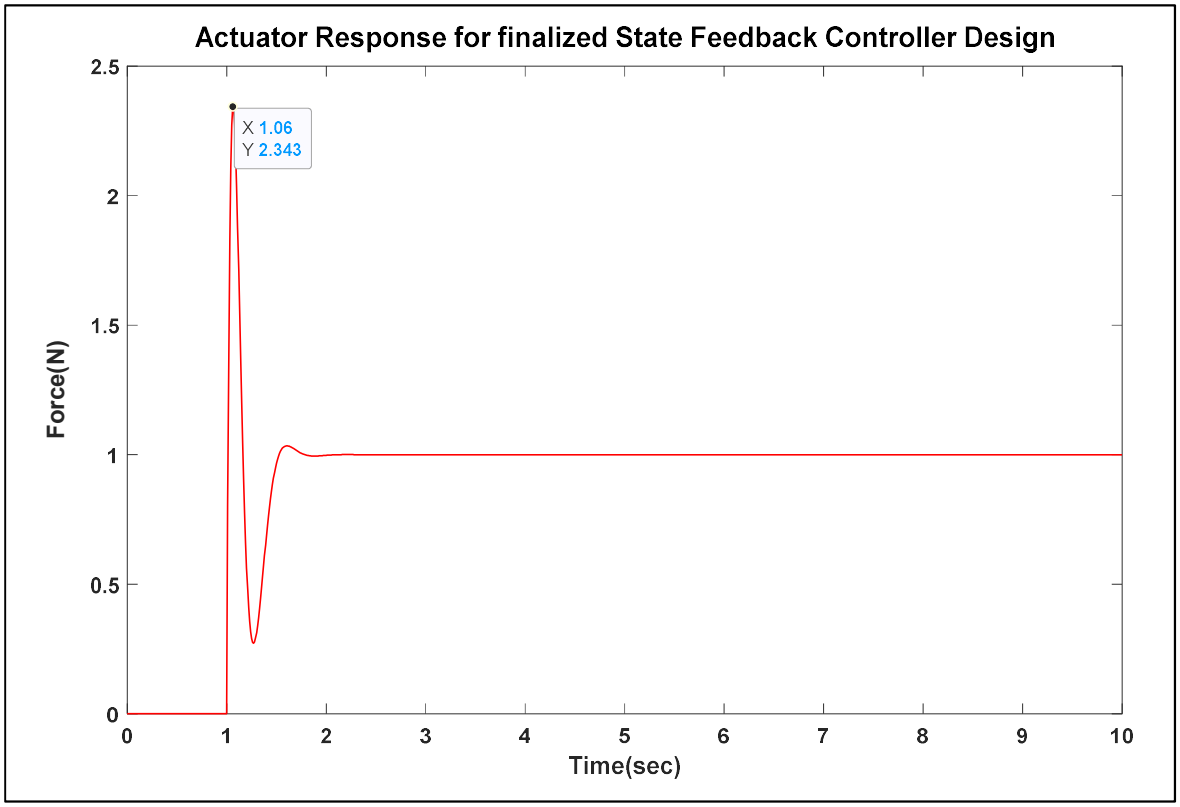
7.3.4 Response plots of the finalized design of state feedback controller

A) Output of closed loop system with respect to time:



The above figure shows the output response of the closed loop system with final design of State Feedback controller (with gains of controller mentioned in third iteration). System is meeting the performance criteria.

B) Actuator response with respect to time:



The above figure shows the actuator response of the closed loop system for final design of State Feedback controller (with gains mentioned in the third iteration). It can be seen that actuator has not reached to its saturation Force value and the cycling of the actuator is also not happening which is desirable from system response perspective.

7.3.5 Comparison of three iterations of the second stage design

	% Overshoot	Settling time (s)	% Steady state error	Incentive ($)
First Iteration	0.0478 %	2.365 s	1.89e-12%	**163 $**
Second Iteration	0 %	1.80 s	1.52e-12 %	**174 $**
Final Design	0 %	1.69 s	6.21e-13%	**176 $**

Chapter 8. Comparison of PID Controller and SF Controller (Selection of good controller)

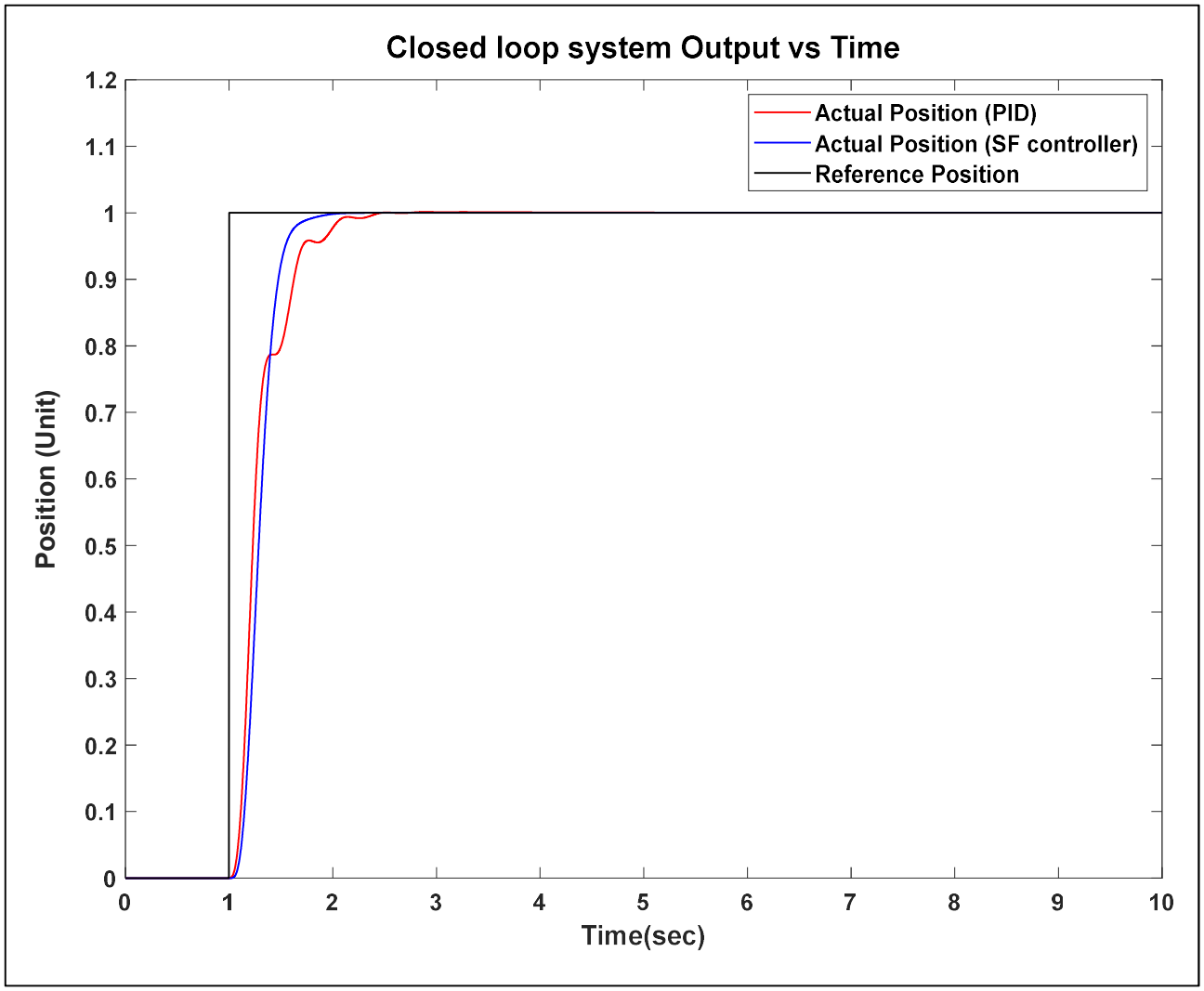
8.1 Comparison of the closed loop system characteristics and the incentives

Now as we have designed two separate controllers and also analyzed the characteristics of both, we can compare both of them in terms of the closed loop response specifications and the incentives earned.

Parameter	PID Controller	SF Controller
% Overshoot	0.13 %	0 %
Settling time (s)	2.07 s	1.69 s
% Steady state error	4.42e-7%	6.21e-13%
Sensor Time Constant (s)	0. 1 s	0. 1 s
Controller Update rate (s)	0.01 s	0.01 s
Actuator size (N)	2.5 N	2.34 N
Incentive ($)	**168 $**	**176 $**

SF Controller Wins!!

8.2 Output Response plots of both controllers



The above figures shows the overlapped closed loop response of both controllers with respect to the refence position. It is worth to notice that SF controller has better response as compared to PID controller.

8.3 MATLAB function for selection of good controller

Every time, we redesign the controller, we do not need to manually compare the performance of two controllers. Therefore, a MATLAB function is written which automatically picks the good controller based on its performances.

Please refer appendix D for this code.

Chapter 9. What will be in the command window after running the MATLAB script

After running the MATLAB script provided in the folder, you will see below information in the command window. It mentions the total incentives earned for each controller: PID as well as SF. Then based on the comparison of two values, it displays the conclusion of MATLAB function which is used to select the good controller.

After that, for individual controllers, you will see the individual incentives earned for different system response characteristics.

Command Window

```
--------------------------------------------------------------------------------
Your total incentive for PID controller is $168.000000.
--------------------------------------------------------------------------------
Your total incentive for SF controller is $176.000000.
--------------------------------------------------------------------------------
Based on maximum incentive criteria,your good controller of two designs is State Feedback Controller.
--------------------------------------------------------------------------------------------------------------
Below are the details of individual incentives.
                                                PID    SF
                                                ___    ___

    Overshoot Incentive ($)                      49     50
    Settling Time Incentive ($)                  19     26
    Steady State Error Incentive ($)            100    100
    Actuator Incentive/penalty ($)                0      0
    Sensor Time constant Incentive/penalty ($)    0      0
    Controller update rate Incentive/penalty ($)  0      0
    Total Incentive ($)                         168    176

--------------------------------------------------------------------------
```

Chapter 10. Conclusion

The project walked us through the two approaches for designing a controller for given plant and sensor system. The PID controller was proved to be less effective as compared to state feedback controller. However, one can still try to tune the parameters further and improve the design subjected to possibility and time constraints. The State Feedback controller was proven to be more effective as it earned more incentives. The response of closed loop system with state feedback controller proved to be less oscillatory and thus system has no overshoot.

This project gives an idea for taking an approach in designing a PID and SF controller. There are numerous ways to tune the parameters of PID controller and other ways to tune the gains of SF controller. All methods are not discussed in the project.

Referring this project report, one can solve the similar problem of control system design and can further evaluate other controller design options.

References

1. Canvas Notes-MEEM 4775 – Dr.Gordon Parker, Michiagn Technological University

2. *Feedback Control of Dynamic Systems* by Gene Franklin, J David-Powell, Abbas Emami-Naeini

3. *Linear Systems Control* by Elbert Hendrics, Ole Jannerup, Paul Sorensen.

APPENDIX A: MATLAB Code for setting up, configuring and simulating the Simulink model

Clean Up

```
clearvars;  % Clear the workspace
clc;  % Clear the command window
close('all');  % Close the open figures
Simulink.sdi.clear;  % Clear the simulink data inspector
```

Simulation Parameters

```
% Solver
simPrm.solTyp = 'Fixed-step';  % Solver type
simPrm.sol    = 'ode3';  % Solver type 2
simPrm.dt  = 0.001; % Integration step size
simPrm.tEnd = 10; % Simulation end time

% Input
Input.Ts = 1;  % Start time of input step
Input.U = 1;   % Amplitude of step input
Tf = 0;  % Input filter time constant
Umax = 2.5;    % Max allowed input
Umin = -2.5;   % Min allowed input
```

Open loop system stability

```
% Roots for the stability
p = [0.1 1.4 6.5 25];  % define the coefficients
r = roots(p); % roots of the equation
```

Open loop system step response

```
%% Plant & sensor Parameters
% Plant
numG = 25;  % Numerator of plant transfer function
denG = [1 4 25];  % Denominator of plant transfer function
G = tf(numG,denG);   % Plant transfer function

% Sensor
Tau = 0.1;  % Time constant of sensor
numH = 1;  % Numerator of sensor transfer function
denH = [Tau, 1];  % Denominator of sensor transfer function
H = tf(numH,denH);  % Sensor transfer function

% Input filter
Tf = 0.06;  % Input filtering time constant
```

```
numFSF = 1;  % Numerator of filter transfer function
denFSF = [Tf 1]; % Denominator of filter transfer function
ctf = tf(numFSF,denFSF);  % Create a continuous transfer function of filter
dtf = c2d(ctf,Control.Z);  % Convert to discrete transfer function of filter
[numd,dend] = tfdata(dtf,'v');  % Extract the numerator and denominator of
discrete transfer function

TF_OL = G*H;  % Create the transfer function of open loop

step(TF_OL);  % create the step response plot
S = stepinfo(TF_OL);  % create the step response information
```

PID Controller Design Parameters for different Iterations

- **First Stage Design**
- **First Iteration of second stage Design**

```
%% PID Controller Design Parameters

% First Iteration
% Control.Ku = 2.64;
% Control.Pu = 0.8;
% Control.Kp = 1.31*Control.Ku;
% Control.Ki = 0.7*Control.Kp*Control.Pu;
% Control.Kd = 1.5*Control.Kp*Control.Pu/8;
```

- **Second Iteration of second stage Design**

```
% Second Iteration
% Control.Ku = 2.64;
% Control.Pu = 0.8;
% Control.Kp = 1.4*Control.Ku;
% Control.Ki = 0.9*Control.Kp*Control.Pu;
% Control.Kd = 2.1*Control.Kp*Control.Pu/8;
```

- **Third Iteration of second stage Design/Finalized Design**

```
% Third Iteration or Final design
Control.Ku = 2.64;
Control.Pu = 0.8;
Control.Kp = 1.62*Control.Ku;   % Proportional gain ;
Control.Ki = 0.95*Control.Kp*Control.Pu;  % Integral Gain
Control.Kd = 2.35*Control.Kp*Control.Pu/8;   % Derivative gain

Control.Z = 0.01; % Controller Update rate
```

State Feedback Controller Design Parameters for different Iterations

- **Creating discretized state space model of the plant with sensor dynamics for using in Kalman Filter block**

```
P.A = [0 1 0;-25 -4 0;1/Tau 0 -1/Tau]; % A matrix
P.B = [0;25;0];  % B matrix
P.C = [0 0 1];  % C Matrix
P.D = [0];  % D Matrix


P.ss = ss(P.A,P.B,P.C,P.D); % Create state space model

P.ssd = c2d(P.ss,Control.Z);  % Discretized version
```

Check Controllability

```
Mc = ctrb(P.A, P.B);  % Create controllability matrix

rho_C = rank(Mc);   % Calculate the rank of controllability matrix
```

Plot damping ratio versus overshoot

```
zeta = 0:0.001:1;
Mp = 100*exp(-pi.*zeta./sqrt(1-zeta.^2));
plot(zeta,Mp,'r');
figure(1)
hold on
grid
xlabel('Zeta');
ylabel('% Overshoot');
```

- **First Stage Design for pole placement and control gains**

```
% desired closed loop poles

% First Stage
% cl.P=[-8; -3.2+2.4j; -3.2-2.4j];
% find the SF gains
% cl.K = place( P.A, P.B, cl.P );
```

- **First Iteration of second stage Design for pole placement and control gains**

```
% desired closed loop poles

% First Iteration
% cl.P=[-8; -4.05+1.96j; -4.05-1.96j];
% find the SF gains
% cl.K = place( P.A, P.B, cl.P );
```

- **Second Iteration of second stage Design for pole placement and control gains**

```
% desired closed loop poles

% First Iteration
% cl.P=[-8; -4.05+1.96j; -4.05-1.96j];
% find the SF gains
% cl.K = place( P.A, P.B, cl.P );
```

- **Third Iteration of second stage Design for pole placement and control gains/Finalized poles**

```
% desired closed loop poles

% Third Iteration
cl.P=[-8; -8+10j; -8-10j];

% find the SF gains
cl.K = place( P.A, P.B, cl.P );
```

- **Calculate the closed loop reference command gain**

```
% create the closed loop state space matrices
cl.A = P.A - P.B * cl.K;  % A matrix of closed loop system
cl.B = P.B;  % B matrix of closed loop system
cl.C = P.C; % C matrix of closed loop system
cl.D = P.D;  % D matrix of closed loop system

% closed loop state space object
cl.ss = ss(cl.A, cl.B, cl.C, cl.D);  % state space model of closed loop system
[cl.num, cl.den] = ss2tf(cl.A, cl.B, cl.C, cl.D);  % numerator and denominator of
closed loop transfer function

cl.refCmdGain = cl.den(end) / cl.num(1,end);  % reference command gain
```

Open, configure and run the Simulink model

```
open_system('p2Sim.slx');  % Open the simulink model
set_param('p2Sim','SolverType',simPrm.solTyp);  % set this solver type in
simulink parameters
set_param('p2Sim','Solver',simPrm.sol);     % set this integration method in
simulink solver
SimOut = sim('p2Sim','SignalLoggingName','sdata');  % Simulate the model and
store the data in sdata name
```

Extract the results of both controllers for plotting and calculation

```
%Handy constants
UPID = 1;
```

```
 YPID = 2;
 Ref = 3;
 USF = 4;
 YSF = 5;

 simPrm.t = 0:Control.Z:simPrm.tEnd;  %control system update rate goes from 0 to
10
 Results.Time = SimOut.tout;  % Grab the values of time
 Results.YPID = SimOut.sdata{YPID}.Values.Data(:,1);  % Extract the results of
actual position
 Results.YSF = SimOut.sdata{YSF}.Values.Data(:,1);  % Extract the values of
reference position
 Results.Ref = SimOut.sdata{Ref}.Values.Data(:,1);  % Extract the values of
controlled input by actuator
 Results.essPID = abs(Results.YPID(simPrm.tEnd/simPrm.dt,1)-
Results.Ref(simPrm.tEnd/simPrm.dt,1))*100/Results.Ref(simPrm.tEnd/simPrm.dt,1);
 Results.essSF = abs(Results.YSF(simPrm.tEnd/simPrm.dt,1)-
Results.Ref(simPrm.tEnd/simPrm.dt,1))*100/Results.Ref(simPrm.tEnd/simPrm.dt,1);
 Results.UPID = SimOut.sdata{UPID}.Values.Data(:,1);
 Results.USF = SimOut.sdata{USF}.Values.Data(:,1);
```

Calculate time domain characteristics (Step response information)

```
 StepPID=stepinfo(Results.YPID,Results.Time,'SettlingTimethreshold',0.01);  % Step
info of PID Controller
 StepSF=stepinfo(Results.YSF,Results.Time,'SettlingTimethreshold',0.01); % Step
info of SF Controller
```

Plot responses (makes use of a function)

```
 figure(1)
 myplot(Results.Time, Results.YPID,'Closed loop system Output vs
Time',1,'r','Time(sec)','Position (Unit)')
 hold on
 yticks(0:0.1:1.2);
 myplot(Results.Time, Results.YSF,'Closed loop system Output vs
Time',1,'b','Time(sec)','Position (Unit)')
 myplot(Results.Time, Results.Ref,'Closed loop system Output vs
Time',1,'k','Time(sec)','Position (Unit)')
 legend('Actual Position (PID)','Actual Position (SF controller)','Reference
Position');
```

Calculate Incentives earned for both designs (makes use of a function)

- **For PID Controller**

```
 Inc.PID =
incentives(StepPID.Overshoot,StepPID.SettlingTime,Results.essPID,Umax,Tau,Control.
Z); % Incentive of PID Controller
```

- **For SF Controller**

```
 Inc.SF =
incentives(StepSF.Overshoot,StepSF.SettlingTime,Results.essSF,Umax,Tau,Control.Z);
% Incentive of SF Controller
```

Estimate good controller between two (makes use of a function)

```
 goodctrl = max_incentive(Inc.PID,Inc.SF);  % Run the function of good controller
```

APPENDIX B: MATLAB function for plotting the figures

```
function myplot(x,y,ttl,LW,color,xlbl,ylbl)
plot(x,y,color,'LineWidth',LW);
xticks(0:1:10);  % Give the x axis ticks
title(ttl); % Give the title.
xlabel(xlbl);  % Give x label
ylabel(ylbl);  % Give y label
end
```

APPENDIX C: MATLAB function for calculating the incentives

```
function incentives = incentives(os,ts,err,u,tau,tstepz)
if os <= 5
    os_in = round((5-os)*10);
else
    os_in = round((5-os)*10);
end

if ts <= 3
    ts_in = round((3-ts)*20);
else
    ts_in = round((3-ts)*20);
end

if err <= 5
    err_in = round((5-err)*20);
else
    err_in = round((5-err)*20);
end

if u <= 2.5
    u_penalty = 0;
elseif 2.5 < u <=4
    u_penalty = round((u - 2.5)*50);
else
    u_penalty = round((u - 2.5)*50);
end

if tau == 0.1
    tau_in = 0;
elseif 0.04 < tau < 0.1
    tau_in = round((tau-0.1)*140);
elseif tau > 0.1
    tau_in = round((tau-0.1)*35);
else
    tau_in = 0;
end

if 0.001 <= tstepz < 0.01
    tstepz_incentive = -round((0.01-tstepz)*300);
elseif tstepz == 0.01
    tstepz_incentive = 0;
elseif tstepz < 0.001
    tstepz_incentive = 0;
else
    tstepz_incentive = -round((0.01-tstepz)*300);
end
```

```
 total = os_in + ts_in + err_in - u_penalty + tstepz_incentive;
 incentives = [total,os_in,ts_in,err_in,u_penalty,tau_in,tstepz_incentive];

 if total > 0
     fprintf('<strong>Your total incentive is $%f</strong>\n',total);
     fprintf('<strong>-----------------------------------------------------
</strong>\n');
 else
     fprintf('<strong>Your total penalty is $%f</strong>\n',-total);
     fprintf('<strong>-----------------------------------------------------
</strong>\n');
 end

 end
```

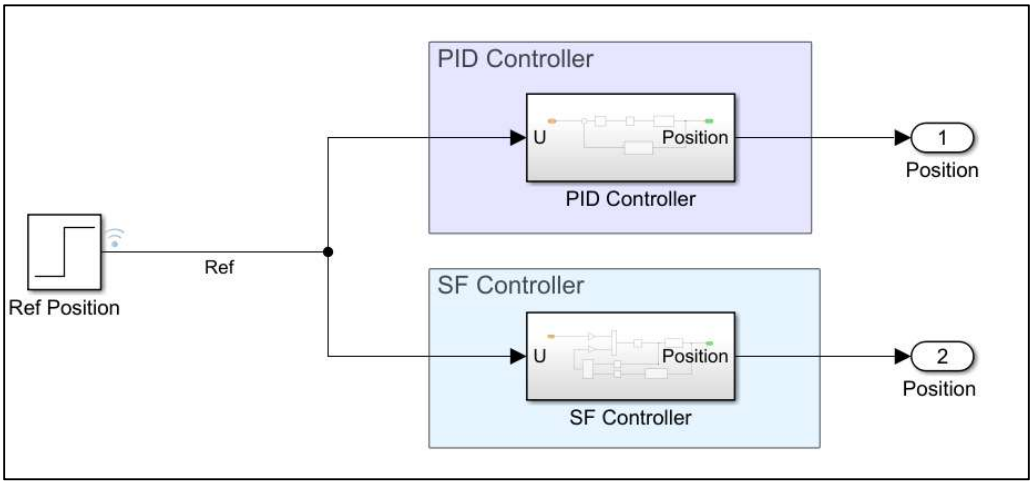
APPENDIX D: MATLAB function for selecting the good controller

```
function goodctrl = max_incentive(incPID,incSF)

max_incentive = max(incPID(1),incSF(1));

if max_incentive == incPID(1)
    goodctrl = 'PID';
    fprintf('<strong>----------------------------------------------------------------
------------------</strong>\n');
    fprintf('<strong>Your total incentive for PID controller is
$%f.</strong>\n',incPID(1));
    fprintf('<strong>----------------------------------------------------------------
------------------</strong>\n');
    fprintf('<strong>Your total incentive for SF controller is
$%f.</strong>\n',incSF(1));
    fprintf('<strong>----------------------------------------------------------------
------------------</strong>\n');
    fprintf(2,'<strong>Based on maximum incentive criteria,your good controller
of two designs is PID Controller.</strong>\n');
    fprintf('<strong>----------------------------------------------------------------
---------------</strong>\n');
    fprintf('<strong>Below are the details of individual
incentives.</strong>\n');
    T =
array2table([incPID(2),incSF(2);incPID(3),incSF(3);incPID(4),incSF(4);incPID(5),in
cSF(5);incPID(6),incSF(6);incPID(7),incSF(7);incPID(1),incSF(1)],...
        'VariableNames',{'PID','SF'},...
        'RowName',{'Overshoot Incentive ($)','Settling Time Incentive
($)','Steady State Error Incentive ($)','Actuator Incentive/penalty ($)',...
        'Sensor Time constant Incentive/penalty ($)','Controller update rate
Incentive/penalty ($)','Total Incentive ($)'});
    disp(T)

else
    max_incentive == incSF(1);
    goodctrl = 'State Feedback';
    fprintf('<strong>----------------------------------------------------------------
------------------</strong>\n');
    fprintf('<strong>Your total incentive for PID controller is
$%f.</strong>\n',incPID(1));
    fprintf('<strong>----------------------------------------------------------------
------------------</strong>\n');
    fprintf('<strong>Your total incentive for SF controller is
$%f.</strong>\n',incSF(1));
    fprintf('<strong>----------------------------------------------------------------
------------------</strong>\n');
    fprintf(2,'<strong>Based on maximum incentive criteria,your good controller
of two designs is State Feedback Controller.</strong>\n');
```

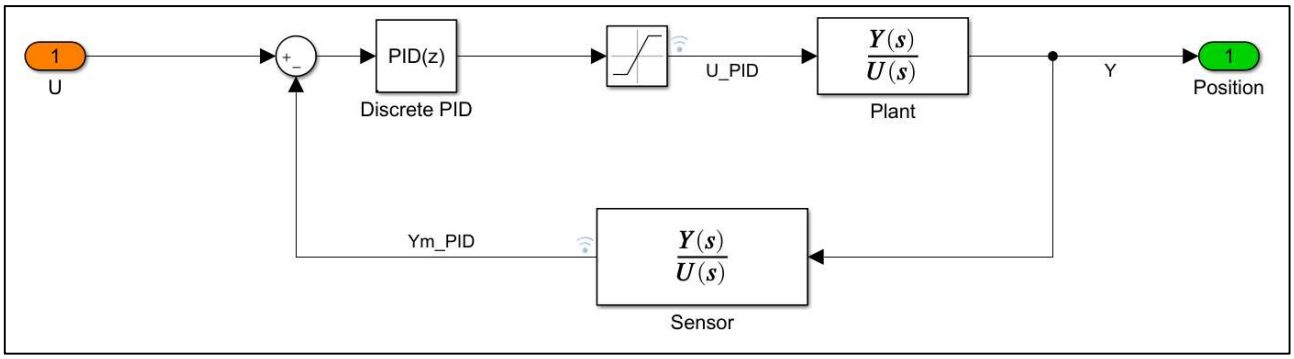
```
    fprintf('<strong>-----------------------------------------------------------
-----------------------------------------------</strong>\n');
    fprintf('<strong>Below are the details of individual
incentives.</strong>\n');
    T =
array2table([incPID(2),incSF(2);incPID(3),incSF(3);incPID(4),incSF(4);incPID(5),in
cSF(5);incPID(6),incSF(6);incPID(7),incSF(7);incPID(1),incSF(1)],...
        'VariableNames',{'PID','SF'},...
        'RowName',{'Overshoot Incentive ($)','Settling Time Incentive
($)','Steady State Error Incentive ($)','Actuator Incentive/penalty ($)',...
        'Sensor Time constant Incentive/penalty ($)','Controller update rate
Incentive/penalty ($)','Total Incentive ($)'});
    disp(T)
    fprintf('<strong>-----------------------------------------------------------
---------------</strong>\n');

 end

 end
```

APPENDIX E: Simulink Model

A) Top level model view



B) PID Controller Subsystem



C) State Feedback Controller Subsystem