Indian Institute of Technology, Gandhinagar

Ball and Beam Experiment

ES-245 Control Systems

Group 12

Authors:

Abhinav Singh Yadav

Tamanna Meena

Ankeshwar Rutesha

Pasala Greeshma

Roll Numbers:

22110011, 22110268, 22110024, 22110182

October 22, 2024

Contents

1 Introduction **3**
1.1 Problem Statement 3
1.2 Assumptions 3
1.3 Objective 3
1.4 Setup Diagram 3

2 Task 1: Define the System **3**
2.1 Write the System Dynamics Equations 3
2.2 Transfer Function 6
2.3 State Space Representation 6

3 Task 2 : Analysis of the system **7**
3.1 Perform the following analysis using MATLAB:
1. Plot poles/zeros
2. Plot open loop step response (servo gear angle in 1 radian step) . 7
3.2 Identify the type of the system and mention your observation from the poles/zeros plot and the open loop response of the system. 10

4 Task 3 : PID **10**
4.1 Design a controller with unity feedback and plot the performance of the following controllers for varying gains:
1. Proportional controller
2. Proportional-derivative controller
3. Proportional-integral-derivative controller 10
4.2 Mention your observations on all three controller types and how they affect transient response, steady-state response, and set point error. . 24
4.3 Design a controller to follow the following criteria-
settling time of less than 3 seconds and overshoot of less than 5 %. . 29

5 Task 4: Simulation of ball and beam in MATLAB **32**
5.1 Build the ball beam model in Simulink and generate the system's open loop response. 32
5.2 Linearize the model and design a compensator to meet the following design criteria - overshoot of less than 5 percent and settling time of less than 5 seconds. Generate the system's closed-loop response. . . . 34
5.3 Develop a Simscape model for the ball beam system. 40

6 Task 5: Physical System **42**
6.1 Develop a CAD model of the system and build the physical system. . 42
6.2 Design a PID controller to stabilize the ball on the beam at the given distance from the end. Comment on your observations and challenges in implementing the PID control. 45

7 Problems Faced **46**

8 Acknowledgement and References **46**

9 References **47**

1 Introduction

1.1 Problem Statement

As shown in the figure below, a ball is placed on a beam, where it rolls under the influence of gravity. The ball has one degree of freedom along the length of the beam. The beams angle is adjusted using a lever arm connected to the beam at one end and to a servo motor gear at the other. As the servo gear rotates by an angle about its axis, the lever arm angle changes from the horizontal.

1.2 Assumptions

The ball remains in contact with the beam at all times. The ball performs pure rolling motion (rolling without slipping).

1.3 Objective

Design a controller to manipulate the balls position on the beam, with the input being the servo gear angle and the output being the balls position.

1.4 Setup Diagram

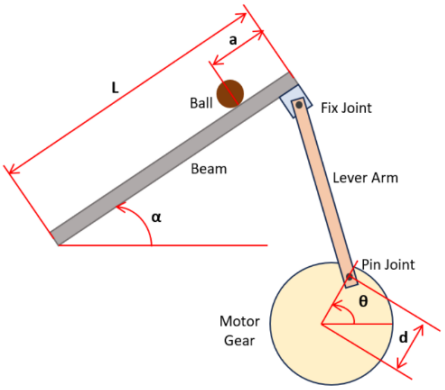


Figure 1: Ball and beam schematic

2 Task 1: Define the System

2.1 Write the System Dynamics Equations

Design Criteria

- Settling time < 3 seconds
- Overshoot $< 5\%$

System Dynamics Equations:

The system involves a ball on a beam, which can pivot due to motor action. The motor rotates a lever arm by an angle θ, thereby changing the beam's angle α. The ball rolls along the beam, maintaining contact without slipping.

Key Variables:

- L: Length of the beam = 0.3m
- a: Position of the ball on the beam
- α: Angle of the beam from the horizontal
- θ: Motor gear angle from its reference position
- d: Distance from the motor gear axis to the lever arm pivot = 0.05 m
- m: Mass of the ball = 0.0023267 kg
- R: Radius of the ball = 0.02 m
- g: Gravitational acceleration = 9.81 m/s^2
- J: Ball's moment of inertia (hollow sphere) = $6.2045 \times 10^{-7}\,\text{kg}\,\text{m}^2$

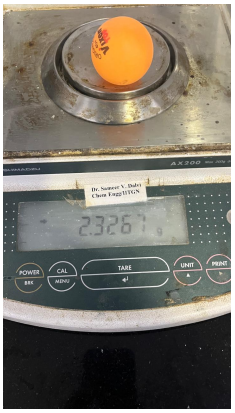


Figure 2: Weighing the ball in the chemistry lab

System Dynamics Equations:

- **Torque and Rotation:**

$$\tau = J\ddot{\theta}$$

The torque applied by the motor results in angular acceleration of the lever arm.

- **Newtons Second Law for Rotational Motion (Beam):**

$$\alpha = \frac{d}{L} \cdot \theta$$

- **Derivation using Lagrangian:**

  The kinetic energy K for the system is given by:

$$K = \frac{1}{2} m_b \dot{r}^2 + \frac{1}{2} J \left(\frac{\dot{r}}{R_b} \right)^2 + \frac{1}{2} \left(J_s + m_b r^2 \right) \dot{\alpha}^2 + \frac{1}{2} J_\ell \dot{\alpha}^2 \tag{1}$$

  where:

  - m_b is the mass of the ball.
  - J is the beam's moment of inertia.
  - R_b is the radius of the ball.
  - J_s is the moment of inertia related to the beam's support.
  - J_ℓ is the moment of inertia of the beam about its length.
  - r is the linear displacement of the ball along the beam.
  - $\dot{r}$ is the time derivative of r, representing the velocity of the ball.
  - $\dot{\alpha}$ is the angular velocity of the beam.

  The potential energy P of the system is given by:

$$P = \frac{1}{2} m_b g r \sin \alpha + m_B g r \sin \alpha \tag{2}$$

  where:

  - m_B is the mass of the beam.
  - g is the acceleration due to gravity.
  - α is the angular displacement of the beam.

  Lagrangian and Equations of Motion

  The Lagrangian function L is the difference between the kinetic and potential energies:

$$L = K - P \tag{3}$$

  The Euler-Lagrange equation is used to derive the equations of motion:

$$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{q}} \right) - \frac{\partial L}{\partial q} = Q \tag{4}$$

where q represents the generalized coordinates (such as r and α), and Q represents generalized forces.

Applying the Euler-Lagrange equation for the ball and beam system yields the following dynamic equations:

$$0 = \left(\frac{J}{R_b^2} + m_b\right)\ddot{r} + m_b g \sin\alpha - m_b r\dot{\alpha}^2 \tag{5}$$

$$\left(\frac{J}{R_b^2} + m_b\right)\ddot{r} = -m_b g\alpha \tag{6}$$

These equations represent the motion of the ball and the torque balance on the beam.

Torque Equation and Beam Angle

The torque τ produced by the motor applied to the end of the beam can be expressed as:

$$\tau = J_\ell \ddot{\alpha} \tag{7}$$

Solving for α, we have:

$$\alpha = \frac{d}{L}\theta \tag{8}$$

Finally, the equation of motion for the beam becomes:

$$\left(\frac{J}{R_b^2} + m_b\right)\ddot{r} = -m_b g\frac{d}{L}\theta \tag{9}$$

2.2 Transfer Function

Taking the Laplace transform:

$$\left(\frac{J}{R^2} + m\right)A(s)s^2 = -mg\frac{d}{L}\Theta(s)$$

$$P(s) = \frac{A(s)}{\Theta(s)} = -\frac{mgd}{L\left(\frac{J}{R^2} + m\right)}\frac{1}{s^2}$$

2.3 State Space Representation

Define state variables:

- $x_1 = r(t)$
- $x_2 = \dot{r}(t)$

State-space representation:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -\frac{mgd}{L\left(\frac{J}{R^2}+m\right)} \end{bmatrix} \theta$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

3 Task 2 : Analysis of the system

Assume suitable values of the parameters in the transfer function, including the mass of the ball (m), radius of the ball (R), lever arm offset (d), length of the beam (L), balls moment of inertia (J), balls position on beam (a), beam angle, and servo gear angle.

3.1 Perform the following analysis using MATLAB:

1. Plot poles/zeros

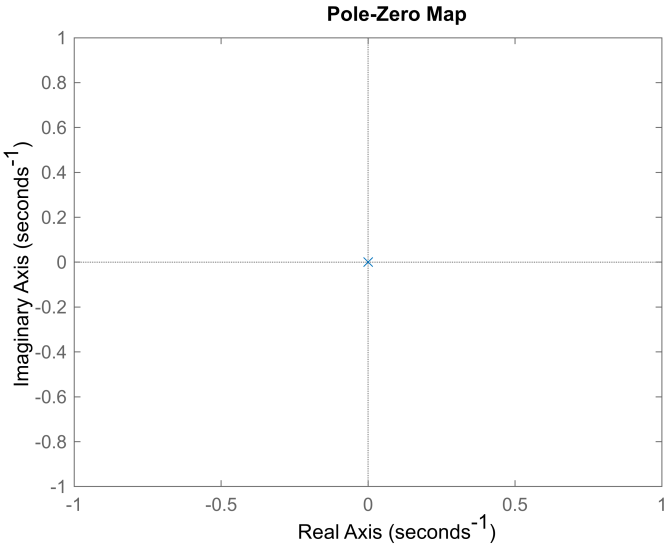
2. Plot open loop step response (servo gear angle in 1 radian step)

Solution:
For our further plots, we will use the practical values we will make practically to help us solve further tasks

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
pzmap(P_ball)
```

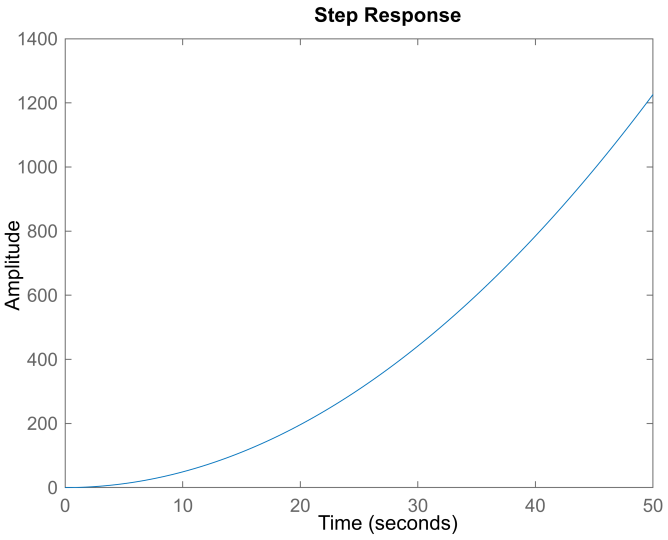


1

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
%pzmap(P_ball)
step(P_ball)
```



3.2 Identify the type of the system and mention your observation from the poles/zeros plot and the open loop response of the system.

Solution: The Ball and Beam system is a type II system with two poles at the origin, as seen in the pole/zero map above.
The open loop system will be unstable since the poles are not strictly in the left half plane, as seen in the step response above.
From the plot, it is clear that the system is unstable in the open loop, causing the ball to roll right off the end of the beam.
Therefore, some method of controlling the ball's position in this system is required.

4 Task 3 : PID

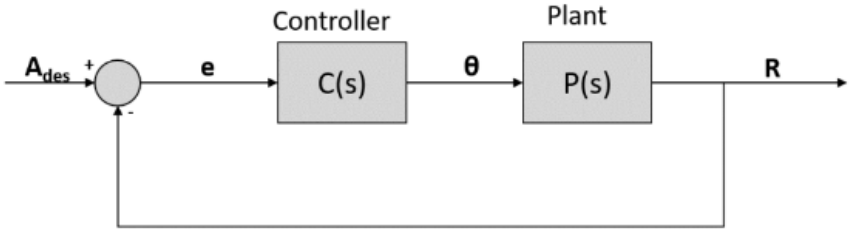
4.1 Design a controller with unity feedback and plot the performance of the following controllers for varying gains:

1. **Proportional controller**
2. **Proportional-derivative controller**
3. **Proportional-integral-derivative controller**

Solution:
The open-loop transfer function of the plant for the ball and beam experiment is given below:

$$P(s) = \frac{A(s)}{\Theta(s)} = -\frac{mgd}{L\left(\frac{J}{R^2} + m\right)}\frac{1}{s^2} \qquad \left[\frac{m}{\text{rad}}\right] \tag{10}$$

The block diagram for this example with a controller and unity feedback of the ball's position is shown below. First, we will study the response of the system shown above



when a proportional controller is used. Then, derivative and/or integral control will be added if necessary.

Recall, that the transfer function for a PID controller is:

$$C(s) = K_p + \frac{K_i}{s} + K_d s = \frac{K_d s^2 + K_p s + K_i}{s}$$

From here we can write equations as:

$$\frac{R(s)}{A_{\text{des}}(s)} = \frac{\left(\frac{K_d s^2+K_p s+K_i}{s}\right)\left(\frac{-mgd}{L\left(\frac{J}{r^2}+m\right)}\cdot\frac{1}{s^2}\right)}{1+\left(\frac{K_d s^2+K_p s+K_i}{s}\right)\left(\frac{-mgd}{L\left(\frac{J}{r^2}+m\right)}\cdot\frac{1}{s^2}\right)}$$

1. Proportional Controller

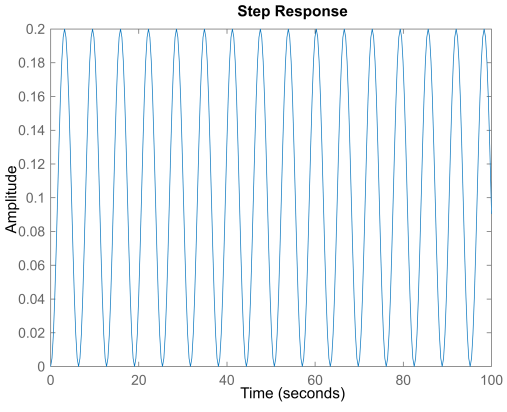
The closed-loop transfer function for proportional control with a proportional gain (K_p) equal to 1, 5 and 0.5 can be modelled by the following lines of MATLAB code. Also, we can model the system's response to a step input of 0.10 m. We can see that by increasing Kp, the time period reduces. :

A. K_p=1

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
Kp1 = 1;
C1 = pid(Kp1); %controller1 input
sys_cl=feedback(C1*P_ball,1);  %setting unity feedback
step(0.10*sys_cl)  %step response
axis([0 100 0 0.2])
```



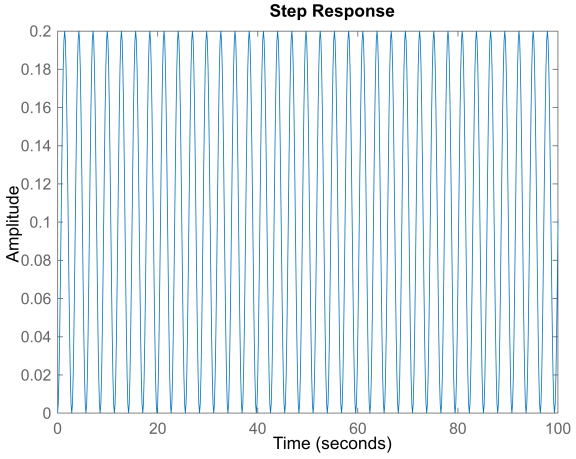
1

B. K_p=5

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
Kp2 = 5;
C2 = pid(Kp2); %controller2 input
sys_cl=feedback(C2*P_ball,1);  %setting unity feedback
step(0.10*sys_cl)  %step response
axis([0 100 0 0.2])
```



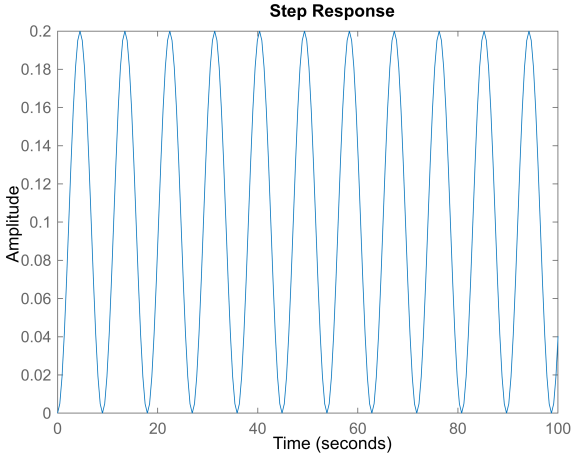
1

C. K_p=0.5

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
Kp3 = 0.5;
C3 = pid(Kp3); %controller3 input
sys_cl=feedback(C3*P_ball,1);  %setting unity feedback
step(0.10*sys_cl)  %step response
axis([0 100 0 0.2])
```



1

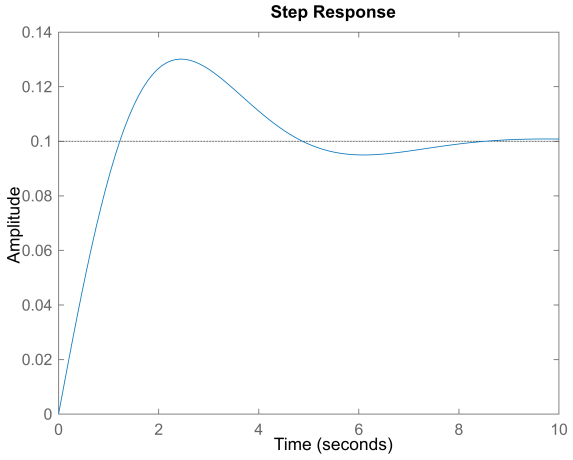
2. Proportional-derivative Controller Now, we will add a derivative term to the controller. We have taken three cases, from which the solution can be obtained from the second case.

1. **(1)** K_p=K_d=**1**

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
Kp1 = 1; %setting Kp
Kd1 = 1;  %setting kd
C1 = pid(Kp1,0,Kd1);  %controller
sys_cl=feedback(C1*P_ball,1);  %unity feedback
t=0:0.01:5;  %time
step(0.10*sys_cl)  % step response for 0.10m
```



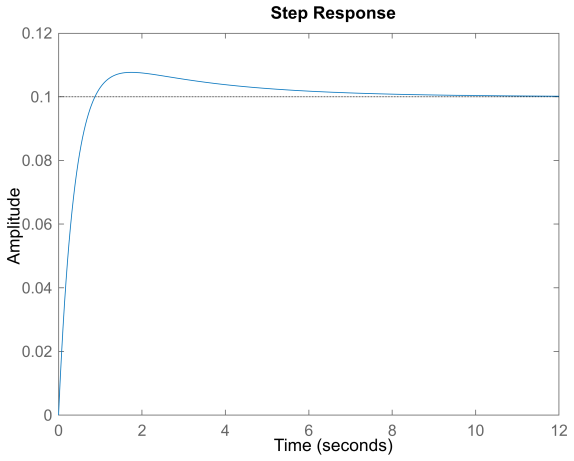
1

2. **(2)** K_p**=1,** K_d**=3** (K_p<K_d)

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
Kp2 = 1; %setting Kp
Kd2 = 3;  %setting kd
C2 = pid(Kp2,0,Kd2);  %controller2
sys_cl=feedback(C2*P_ball,1);  %unity feedback
t=0:0.01:5;  %time
step(0.10*sys_cl)  % step response for 0.10m
```



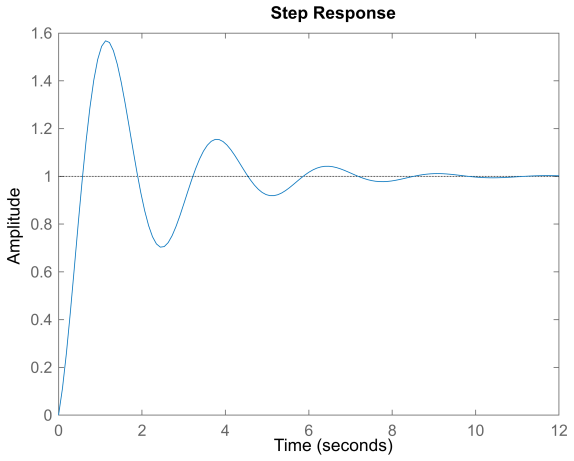
1

3. **(3)** K_p=**6**,K_d=**1** (K_p>K_d)

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
Kp4 = 6; %setting Kp
Kd4 = 1;  %setting kd
C4 = pid(Kp4,0,Kd4);  %controller4
sys_cl=feedback(C4*P_ball,1);  %unity feedback
t=0:0.01:5;
step(sys_cl);    % step response
```



1

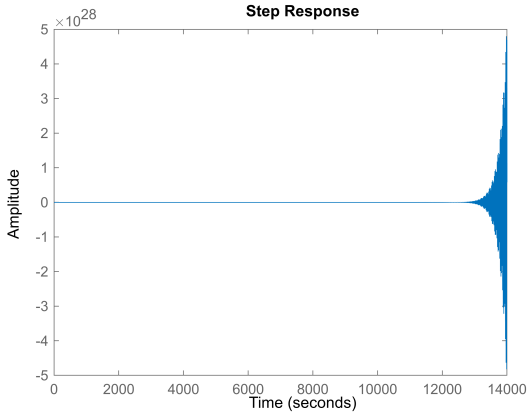
3. Proportional-integral derivative Controller Now, we will add an integral term to the controller. We have taken seven cases, from which the solution can be obtained from multiple cases that we have made. However, we can see that some responses are overshooting.

1. **(1)** K_p=K_d=K_i=**1**

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
Kp1 = 1; %setting Kp
Ki1 = 1; % setting Ki
Kd1 = 1;  %setting kd
C1 = pid(Kp1,Ki1,Kd1);  %controller1
sys_cl=feedback(C1*P_ball,1);  %unity feedback
t=0:0.01:5;
step(sys_cl);    % step response
```



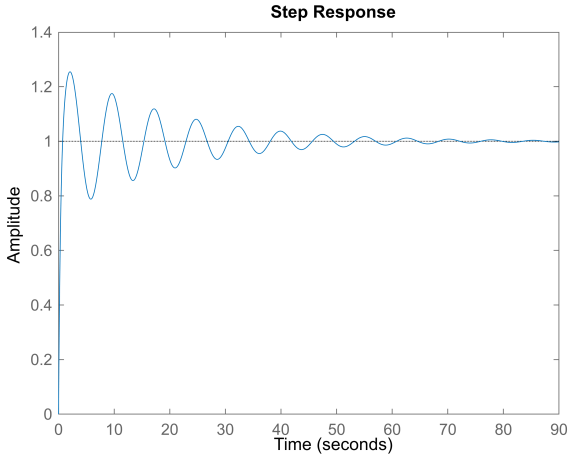
1

2. **(2)** K_p**=1,** K_i**=2,** K_d**=3** (K_p<K_i<K_d)

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
Kp2 = 1; %setting Kp
Ki2 = 2; % setting Ki
Kd2 = 3;  %setting kd
C2 = pid(Kp2,Ki2,Kd2);  %controller2
sys_cl=feedback(C2*P_ball,1);  %unity feedback
t=0:0.01:5;
step(sys_cl);    % step response
```



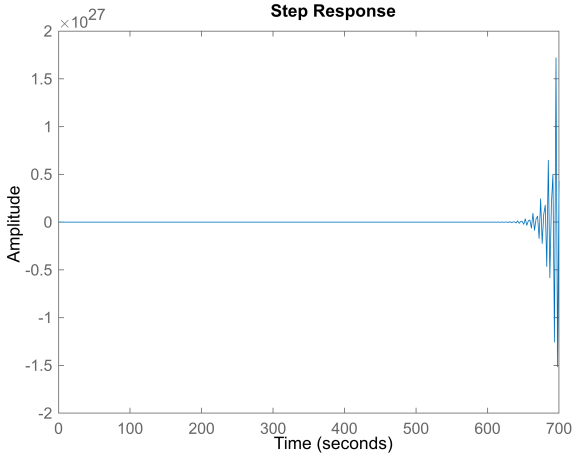
1

3. **(3)** K_p**=1,** K_i**=3,** K_d**=2** (K_p<K_d<K_i)

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
Kp3 = 1; %setting Kp
Ki3 = 3; % setting Ki
Kd3 = 2;  %setting kd
C3 = pid(Kp3,Ki3,Kd3);  %controller3
sys_cl=feedback(C3*P_ball,1);  %unity feedback
t=0:0.01:5;
step(sys_cl);    % step response
```



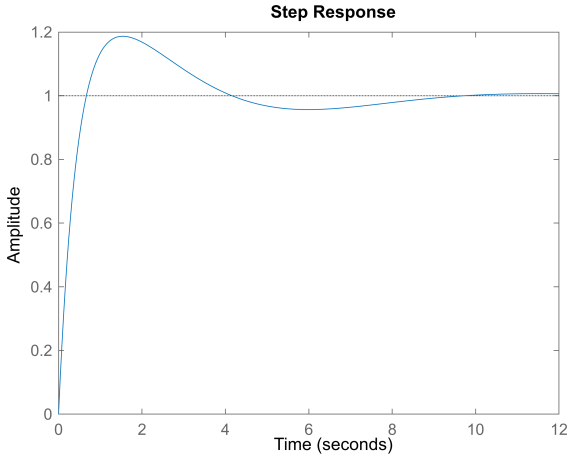
1

4. **(4)** K_p**=2,** K_i**=1,** K_d**=3** (K_i<K_p<K_d)

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
Kp4 = 2; %setting Kp
Ki4 = 1; % setting Ki
Kd4 = 3;  %setting kd
C4 = pid(Kp4,Ki4,Kd4);  %controller4
sys_cl=feedback(C4*P_ball,1);  %unity feedback
t=0:0.01:5;
step(sys_cl);    % step response
```



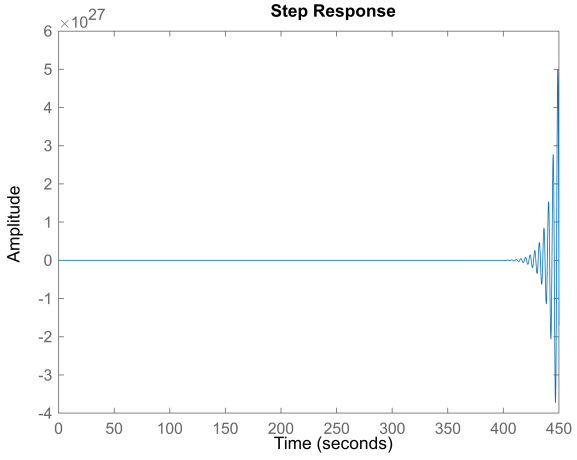
1

5. **(5)** K_p**=2,** K_i**=3,** K_d**=1** (K_d<K_p<K_i)

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
Kp5 = 2; %setting Kp
Ki5 = 3; % setting Ki
Kd5 = 1;  %setting kd
C5 = pid(Kp5,Ki5,Kd5);  %controller5
sys_cl=feedback(C5*P_ball,1);  %unity feedback
t=0:0.01:5;
step(sys_cl);    % step response
```



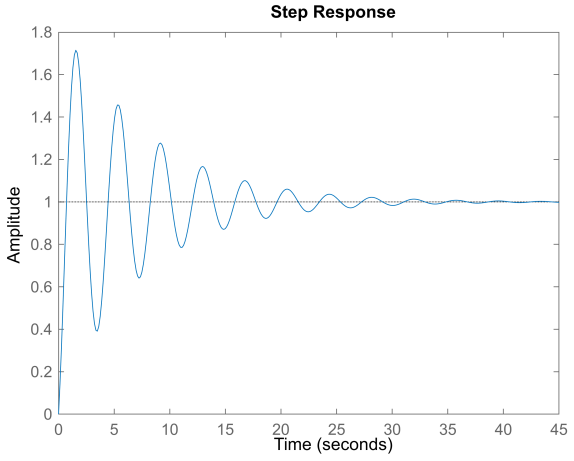
1

6. **(6)** K_p**=3,** K_i**=2,** K_d**=1** (K_p>K_i>K_d)

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
Kp6 = 3; %setting Kp
Ki6 = 2; % setting Ki
Kd6 = 1;  %setting kd
C6 = pid(Kp6,Ki6,Kd6);  %controller6
sys_cl=feedback(C6*P_ball,1);  %unity feedback
t=0:0.01:5;
step(sys_cl);    % step response
```



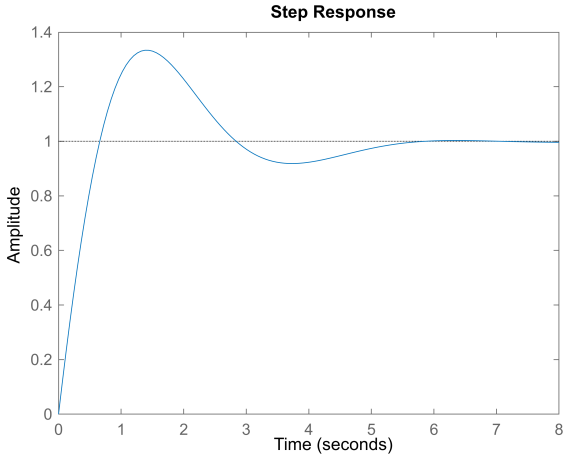
1

7. **(7)** K_p**=3,** K_i**=1,** K_d**=2(**K_p>K_d<K_i**)**

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
Kp7 = 3; %setting Kp
Ki7 = 1; % setting Ki
Kd7 = 2;  %setting kd
C7 = pid(Kp7,Ki7,Kd7);  %controller7
sys_cl=feedback(C7*P_ball,1);  %unity feedback
t=0:0.01:5;
step(sys_cl);    % step response
```



1

4.2 Mention your observations on all three controller types and how they affect transient response, steady-state response, and set point error.

Observations:

1. **Proportional Controller:**

   (a) **Transient Response:** The proportional gain (K_p) primarily affects the transient response. As K_p increases, the system becomes more sensitive, leading to faster oscillations (higher frequency) in the system's response.

   - In the first plot with $K_p = 1$, the oscillations are relatively slow.
   - In the second plot with $K_p = 5$, the frequency of oscillations increases significantly, indicating a faster response.
   - In the third plot with $K_p = 0.5$, the oscillations are slower compared to the other two. This shows that reducing the gain results in a slower transient response.
   - **Rise time**
     - Increasing the proportional gain decreases the rise time.
     - Rise time is the time it takes for the output to reach a certain percentage of the setpoint.
   - **Settling time**
     - Increasing the proportional gain improves settling time, especially at high gains. We cannot see its impact here as only a proportional controller is used. We can see this by adding derivative or integral parts. In this case, the system oscillates continuously and settles only at a particular time interval.
     - Settling time is the time it takes for the output to stay within a certain setpoint tolerance.
   - **Overshoot**
     - Increasing the proportional gain increases the overshoot. But here, we see the same amplitude in all three cases, i.e. 0.2.
     - Overshoot is the amount the output exceeds the setpoint before settling.

   (b) **Steady-State Response:** A proportional controller alone does not eliminate the steady-state error. There is still an offset between the desired

set point and the actual response.

- In all three cases, oscillations occur around a certain amplitude, but the system does not converge to a stable value.
- This suggests that the steady-state error is not completely eliminated while increasing K_p affects the oscillations.

(c) **Set point error:** The system experiences a set point error, meaning the response does not exactly reach the desired value even at a steady state. This is a characteristic limitation of using only a proportional controller.

- The set point error slightly decreases as K_p increases. For example, in the second plot ($K_p = 5$), the oscillations are closer to the target compared to the third plot ($K_p = 0.5$).
- However, without the use of an integral action, the error persists even with high gains.

In summary, increasing the proportional gain (K_p) speeds up the system's response and slightly decreases the set point error. Still, it also introduces faster oscillations and higher frequencies in the transient response. A proportional controller alone cannot eliminate steady-state error.

2. **Proportional Derivative Controller:**

(a) Case 1: $K_p = 1$, $K_d = 1$
Transient Response:
- **Overshoot**: There is a signifact overshoot, with the peak amplitude reaching approximately 0.135.
- **Rise Time**: The system takes longer to reach its peak compared to other cases.
- **Settling Time**: The system settles around 8 seconds.
- **Oscillations**: Noticeable oscillations occur before the system stabilizes. It goes up and comes down and then again goes up at the end of 10 seconds.
Steady-State Response and Setpoint error: The system eventually settles at the desired set point with minimal steady-state error.The system eventually reaches the set point with negligible steady-state error.
Conclusion: The system exhibits an underdamped response with significant oscillations due to low values of K_p and K_d.

(b) Case 2: $K_p = 1$, $K_d = 3$
Transient Response:
- **Overshoot**: Minimal overshoot.
- **Rise Time**: The rise time is shorter than in Case 1, indicating a quicker response.

- **Settling Time**: The system settles around 5-6 seconds, faster than Case 1.
- **Oscillations**: The system is overdamped, with no significant oscillations.

Steady-State Response and Setpoint error: The system reaches the steady-state smoothly with no overshoot. The system smoothly reaches the set point without any noticeable error.

Conclusion: The higher derivative gain K_d reduces oscillations and provides a smoother, overdamped response, leading to a shorter settling time and minimal overshoot.

(c) Case 3: $K_p = 6$, $K_d = 1$

Transient Response:
- **Overshoot**: There is a significant overshoot, with a peak amplitude reaching approximately 1.6.
- **Rise Time**: The rise time is fast due to the high K_p, but the overshoot is large.
- **Settling Time**: The system takes around 10 seconds to settle, with damped oscillations.
- **Oscillations**: There are 3-4 oscillations before the system stabilizes, with the amplitude gradually decreasing.

Steady-State Response and Set Point error: The system eventually stabilizes but with considerable transient oscillations. The system eventually stabilizes near the set point with no significant set point error after oscillations die down. The high proportional gain K_p induces significant oscillations and overshoot, but the system reaches the set point after some oscillations. The set point error is negligible but transient response is poor due to large overshoot and oscillations.

Conclusion: The high proportional gain K_p induces oscillations, while the lower derivative gain K_d is insufficient to dampen them quickly. This case shows a typical underdamped response with large overshoot.

3. **Proportional Integral Derivative Controller:**

(a) **Case 1:** $K_p = K_i = K_d = 1$

- **Transient Response:**
  - **Rising Time:** The system initially rises quickly after 13000 seconds but becomes unstable almost immediately, about in 1000 seconds.
  - **Overshoot:** Extremely high overshoot, oscillations growing exponentially.
  - **Settling Time:** The system does not settle due to the increasing

oscillations.

- **Steady-State Response:** There is no defined steady-state value as the system becomes unstable.
- **Set-Point Error:** The system does not reach the set point due to the unbounded oscillations and divergence.

(b) **Case 2:** $K_p = 1, K_i = 2, K_d = 3$ $(K_p < K_i < K_d)$

- **Transient Response:**
  - **Rising Time:** The rising time is moderate, around 2-4 seconds, with a smooth initial response.
  - **Overshoot:** There is some overshoot, but it is relatively small and controlled.
  - **Settling Time:** The system settles after approximately 40-50 seconds, with a few oscillations before reaching stability.
- **Steady-State Response:** The system reaches a steady state with minimal oscillations and small steady-state error.
- **Set-Point Error:** The set-point error is very small, with the system achieving the desired value after oscillations decay.

(c) **Case 3:** $K_p = 1, K_i = 3, K_d = 2$ $(K_p < K_d < K_i)$

- **Transient Response:**
  - **Rising Time:** The rising time is faster, around 50 seconds, but the integral action dominates, causing instability later on.
  - **Overshoot:** There is a significant overshoot, leading to oscillations.
  - **Settling Time:** The system does not settle properly, as the oscillations persist due to the high integral gain.
- **Steady-State Response:** The system remains unstable and fails to achieve a steady-state value.
- **Set-Point Error:** The set-point error cannot be determined due to the ongoing oscillations and instability.

(d) **Case 4:** $K_p = 2$**,** $K_i = 1$**,** $K_d = 3$ $(K_i < K_p < K_d)$

i. **Transient Response:**

- **Rise Time:** The rise time is about 1 second, which is relatively quick.

- **Settling Time:** The system settles at around 10 seconds.
- **Overshoot:** The overshoot is approximately 20%, indicating moderate oscillation before settling.

ii. **Steady-State Response:**

- The system reaches a steady-state value close to the desired value with minimal error.

iii. **Set Point Error:**

- There is a small steady-state error with improved overall performance compared to previous configurations.

(e) **Case 5:** $K_p = 2$, $K_i = 3$, $K_d = 1$ $(K_d < K_p < K_i)$

i. **Transient Response:**

- **Rise Time:** The rise time is faster, around 50 seconds, but the integral action dominates, causing instability later.
- **Settling Time:** The system does not settle properly, as the oscillations persist due to the high integral gain.
- **Overshoot:** The system exhibits very high overshoot (exceeding 10^{27}), indicating significant instability.

ii. **Steady-State Response:**

- The system never reaches a stable steady state within a reasonable time, showing signs of instability.

iii. **Set Point Error:**

- The system is highly unstable and fails to control effectively.

(f) **Case 6:** $K_p = 3$, $K_i = 2$, $K_d = 1$ $(K_p > K_i > K_d)$

i. **Transient Response:**

- **Rise Time:** The rise time is faster, at around 2 seconds.
- **Settling Time:** The system settles after approximately 40 seconds.
- **Overshoot:** The overshoot is moderate (around 70%) but stable.

ii. **Steady-State Response:**

- The system successfully reaches steady state with minimal oscillation.

iii. **Set Point Error:**

- The steady-state error is small, and the system performs well compared to the previous case.

(g) **Case 7:** $K_p = 3$, $K_i = 1$, $K_d = 2$ $(K_p > K_i > K_d)$

i. **Transient Response:**

- **Rise Time:** The rise time is faster, at around 1-2 seconds.
- **Settling Time:** The system settles after approximately 7 seconds.
- **Overshoot:** The overshoot is less as compared to other cases (around 30-40%) but stable.

ii. **Steady-State Response:**

- The system successfully reaches a steady state with minimal oscillation.

iii. **Set Point Error:**

- The steady-state error is minimal, and the system performs well compared to the previous case.

4.3 Design a controller to follow the following criteria- settling time of less than 3 seconds and overshoot of less than 5 %.

The open-loop transfer function of the plant for the ball and beam experiment is given below:

$$P(s) = \frac{A(s)}{\Theta(s)} = -\frac{mgd}{L\left(\frac{J}{R^2} + m\right)}\frac{1}{s^2} \qquad \left[\frac{m}{\text{rad}}\right] \tag{11}$$

The design criteria for this problem are:

- Settling time less than 3 seconds
- Overshoot less than 5

If we refer to any of the PID control problems for continuous systems, the PID transfer function was expressed as

$$C(s) = K_p + \frac{K_i}{s} + K_d s = \frac{K_d s^2 + K_p s + K_i}{s}$$

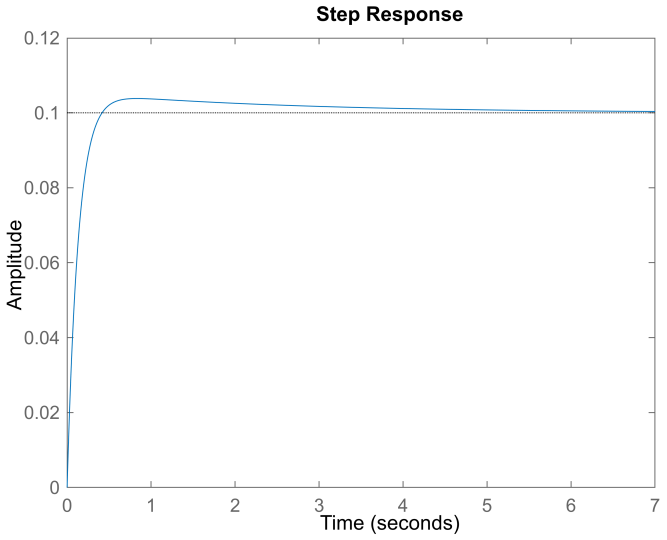
Solution: $K_p = 3$, $K_d = 8$
Transient Response:
- **Overshoot**: There is no overshoot, and the response is quick.
- **Rise Time**: The rise time is very fast, with the system reaching close to the set

point quickly.
- **Settling Time**: Settles around 2-3 seconds.
- **Oscillations**: The system remains overdamped, with no oscillations.
Steady-State Response and Set point error: The system reaches the set point without overshoot or oscillation. The system reaches the set point exactly with no steady-state error.
Conclusion: This configuration meets the design criteria, providing a fast, overdamped response with the shortest settling time and approximately no overshoot. Below is the plot for the response we have generated for this case:

```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02;   % in m (radius of the ball)
g = -9.81;
L = 0.3;   % in m (length of the beam)
d = 0.05;  % in m (lever arm offset)
J = 6.2045e-7;  % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
```

```
Kp3 = 3; %setting Kp
Kd3 = 8;  %setting kd
C3 = pid(Kp3,0,Kd3);  %controller3
sys_cl=feedback(C3*P_ball,1);  %unity feedback
t=0:0.01:6;  %time
step(0.10*sys_cl)  % step response for 0.10m
```



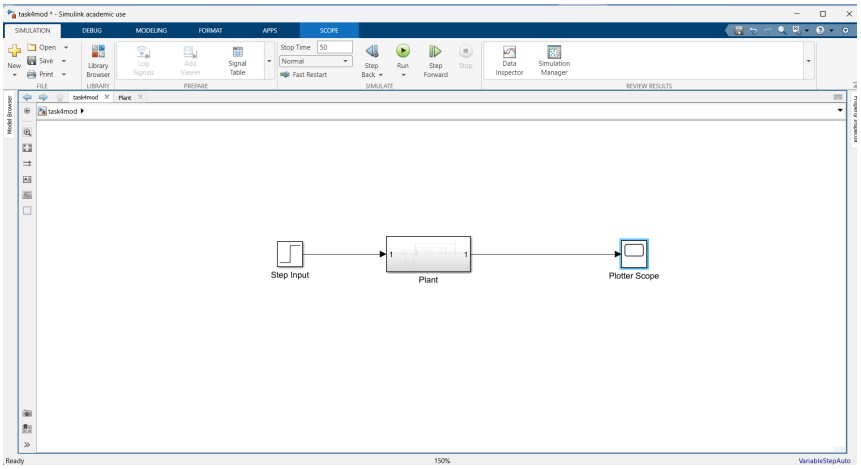
1

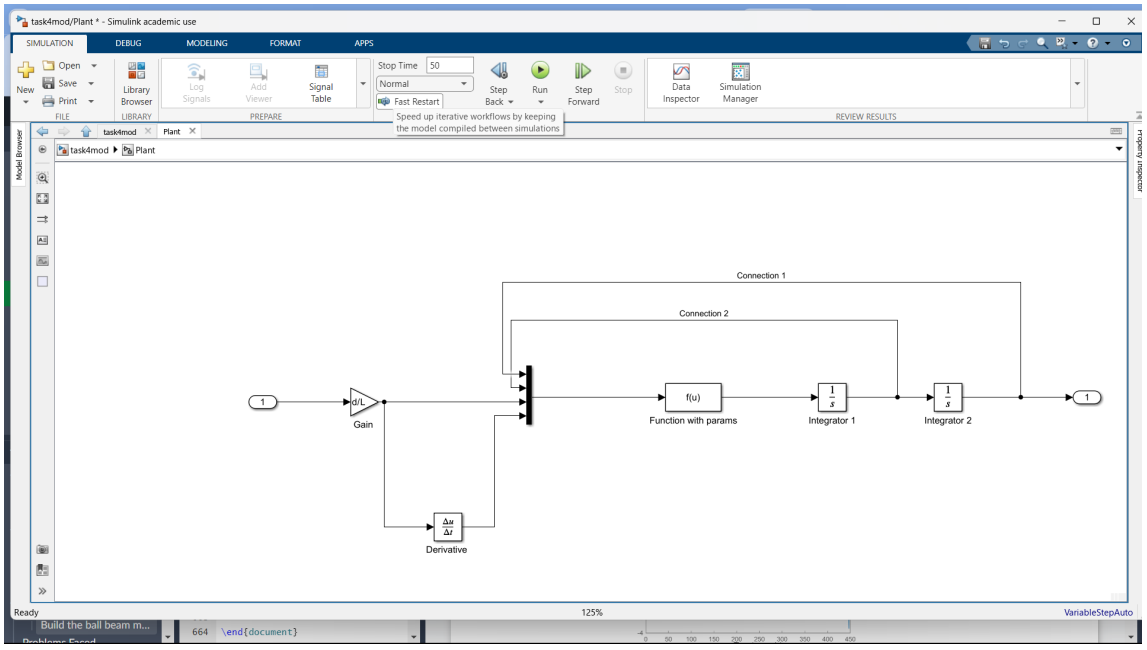
Figure 3: Solution Response

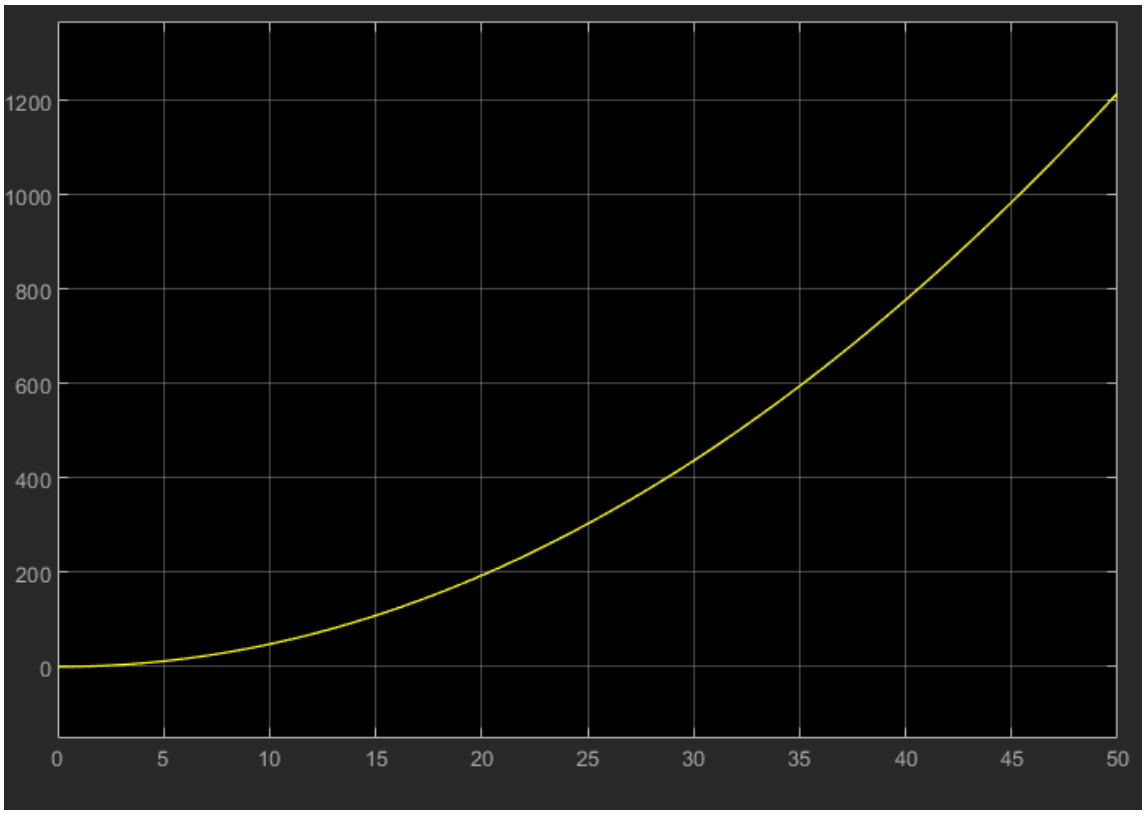
5 Task 4: Simulation of ball and beam in MATLAB

5.1 Build the ball beam model in Simulink and generate the system's open loop response.

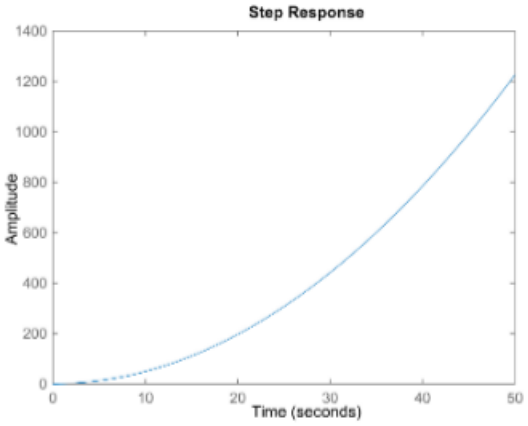
We made the plant with the same transfer function that we have. We will build the model in the third part of this task, where we have to build the whole simscape model.
The following images show the system, plant and the system's response for 50 seconds.







From this, we can compare with the graph that we plotted in task 3:



5.2 Linearize the model and design a compensator to meet the following design criteria - overshoot of less than 5 percent and settling time of less than 5 seconds. Generate the system's closed-loop response.

First, we will look for the step response and the root locus of our open loop system for our design criteria.

The design criteria can also be visualized on the root locus using the `sgrid` command, which creates a grid showing lines of constant damping ratio and natural frequency. The damping ratio (ζ) and natural frequency (ω_n) are determined by equations that link them to the specified maximum per cent overshoot (M_p) and settling time (T_s) requirements:

$$M_p = e^{-\zeta\pi/\sqrt{1-\zeta^2}} \tag{12}$$

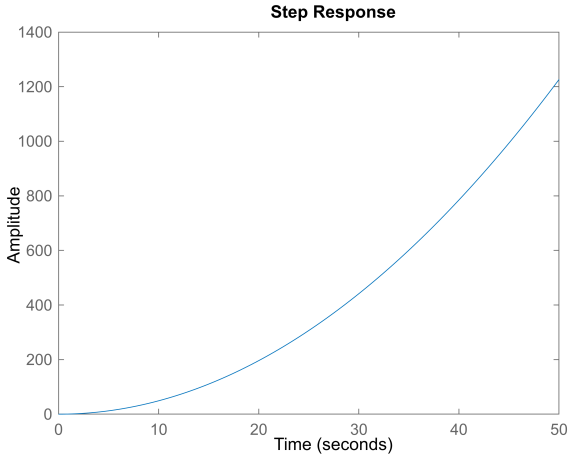
$$T_s = \frac{4}{\zeta\omega_n} \tag{13}$$

It's important to note that the equation for T_s assumes that the system has settled when the response stays within 2% of its final value. Using these equations, the damping ratio and natural frequency were calculated as 0.69 (approximately 0.7) and 1.142, respectively. Below is the corresponding plot generated in MATLAB.

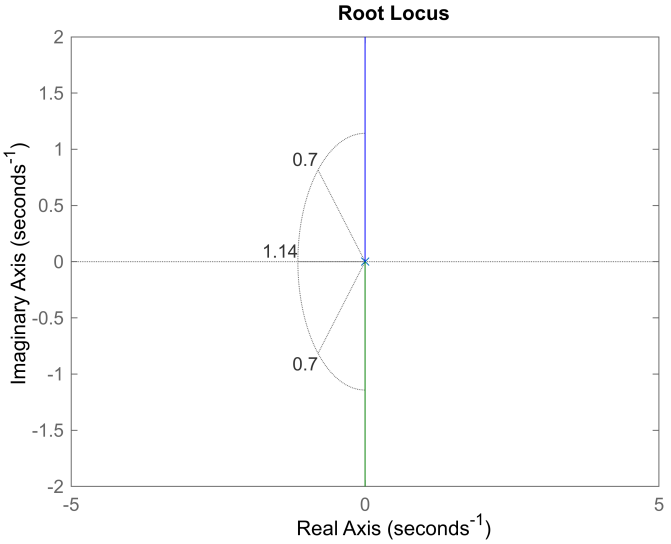
```
clear
close all
clc
m = 0.0023267; % in kg (mass of the ball)
R = 0.02; % in m (radius of the ball)
g = -9.81;
L = 0.3; % in m (length of the beam)
d = 0.05; % in m (lever arm offset)
J = 6.2045e-7; % in Kg/m^2 ( Moment of inertia for hollow sphere (2/3*M*R^2))
s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
```

```
P_ball =

  0.981
  -----
   s^2

Continuous-time transfer function.
Model Properties
```

```
step(P_ball)
axis([0 50 0 1400])
```




```
rlocus(P_ball)
sgrid(0.7, 1.1428571429)
axis([-5 5 -2 2])
```




```
zo = 0.01;
po = 5;
C=tf([1 zo],[1 po]);

rlocus(C*P_ball)
sgrid(0.70, 1.1428571429)
axis([-5 5 -2 2])
[k,poles]=rlocfind(C*P_ball)
```

```
Select a point in the graphics window
```

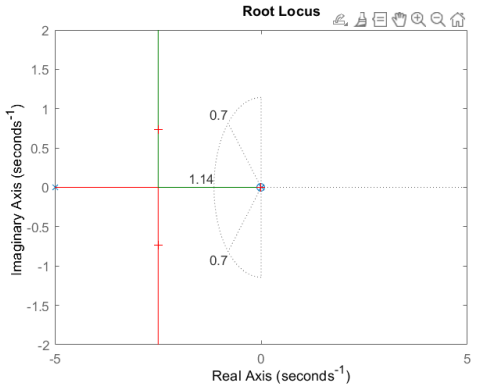
2

The region between the two dotted diagonal lines indicates where the per cent overshoot is less than 5 per cent. The area outside the curved line marks where the settling time is under 5 seconds. It's important to note that no part of the plot meets these design criteria. To address this, we will introduce a lead compensator to the system, aiming to shift the root locus into the left-hand plane for improved stability.

Lead Controller: A first-order lead compensator tends to shift the root locus into the left-hand plane.

$$C(s) = K_c \frac{(s + z_0)}{(s + p_0)}$$

where the magnitude of z_0 is smaller than that of p_0. Now, let's incorporate the compensator into the plant and examine the root locus. We will place the zero near the origin to cancel out one of the poles while positioning the compensators pole to the left of the origin to pull the root locus further into the left-hand plane. The code for this is partly in the image above and partly below: Now, the branches of

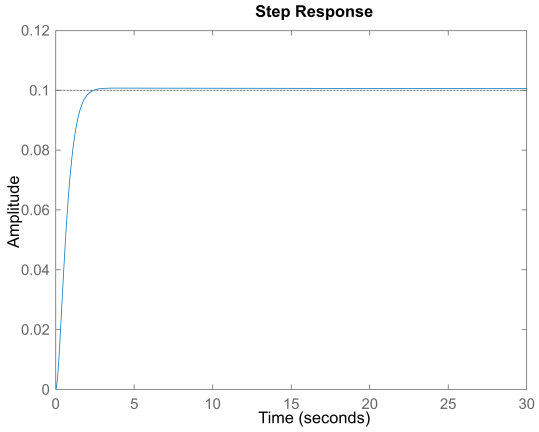



```
selected_point = -2.4905 + 0.7345i
k = 6.9465
poles = 3×1 complex
  -2.4950 + 0.7345i
  -2.4950 - 0.7345i
  -0.0101 + 0.0000i
```

```
sys_cl=feedback(k*C*P_ball,1);
t=0:0.01:30;
figure
step(0.10*sys_cl,t)
```

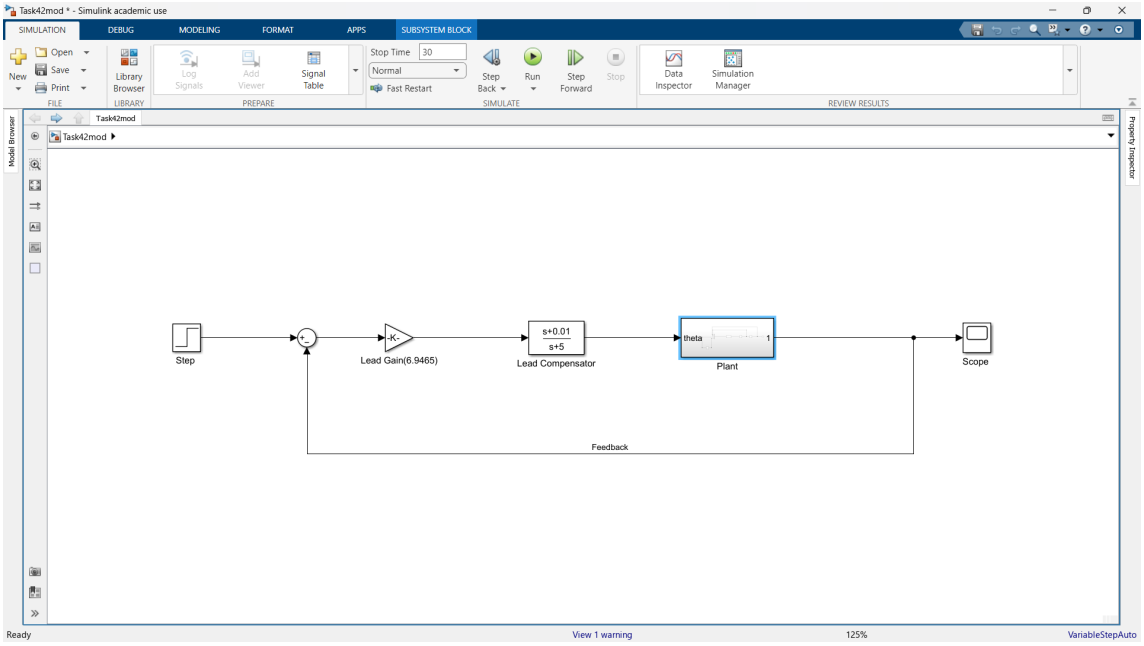
3

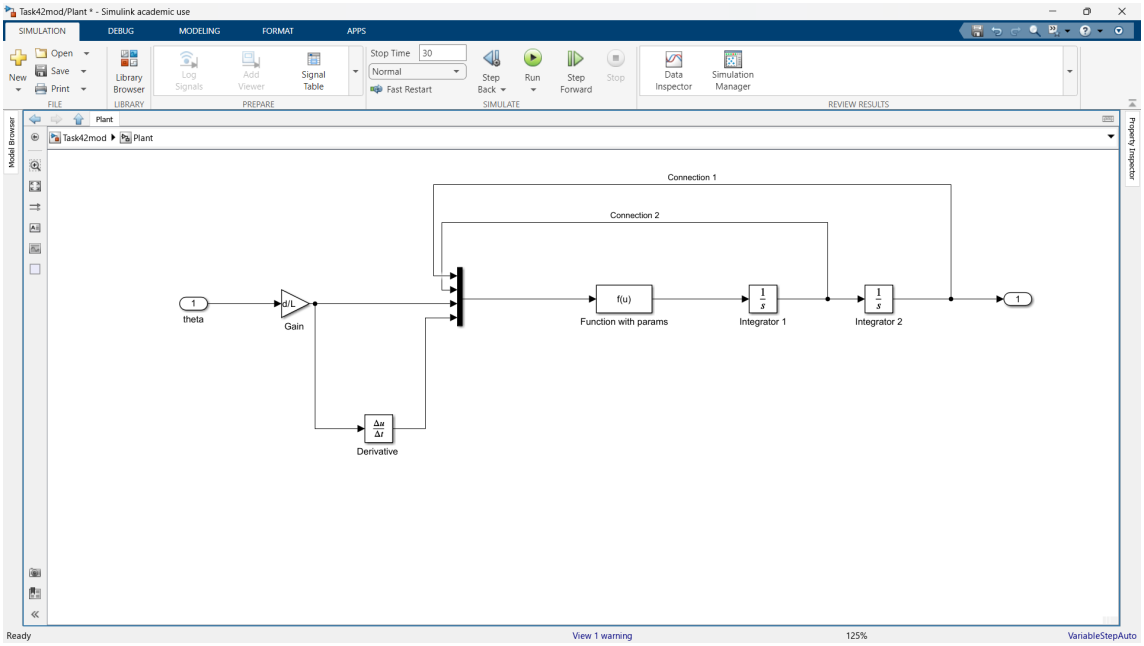
the root locus are within our design criteria.
Selecting the Gain: With the root locus now shifted into the left-hand plane, we can choose a gain that meets our design requirements. The `rlocfind` command will assist in determining this gain. Remember that the values shown in the MATLAB command window might not be identical, but they should be of a similar order of magnitude. We can then plot the systems response using the selected gain.



4

Now we can make a model in Simulink process the values we have got. The following are the images of the model we have made:
The corresponding output with what we have is the following:
Hence, we have successfully made a lead and lag controller to make our system reach the desired output.


Task42mod * - Simulink academic use
SIMULATION
DEBUG
MODELING
FORMAT
APPS
SUBSYSTEM BLOCK
Stop Time
30
Normal
Fast Restart
Step Back
Run
Step Forward
Stop
Data Inspector
Simulation Manager
Task42mod
Step
Lead Gain(6.9465)
s+0.01
s+5
Lead Compensator
theta
Plant
Scope
Feedback
Ready
View 1 warning
125%
VariableStepAuto


Task42mod/Plant * - Simulink academic use
Plant
Connection 1
Connection 2
theta
d/L
Gain
f(u)
Function with params
Integrator 1
Integrator 2
Derivative
Ready
View 1 warning
125%
VariableStepAuto

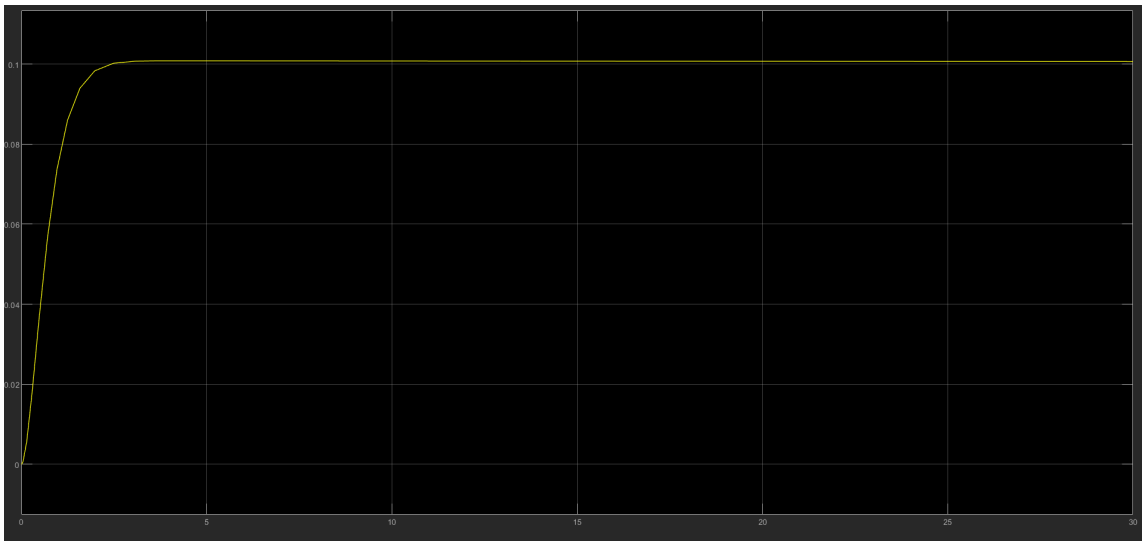
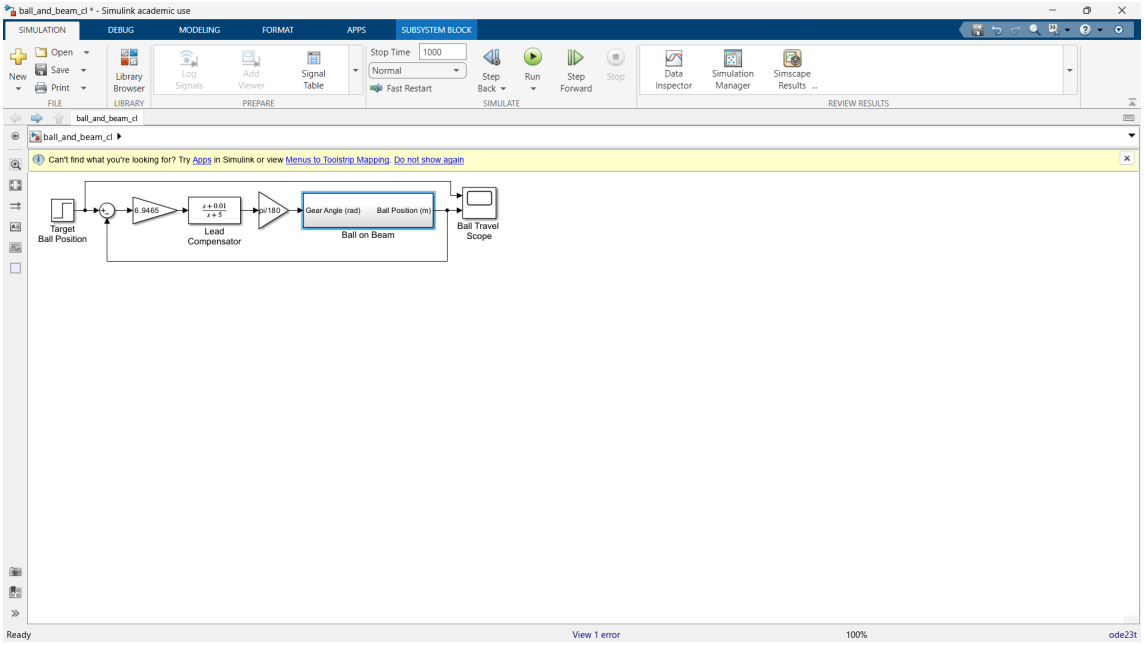
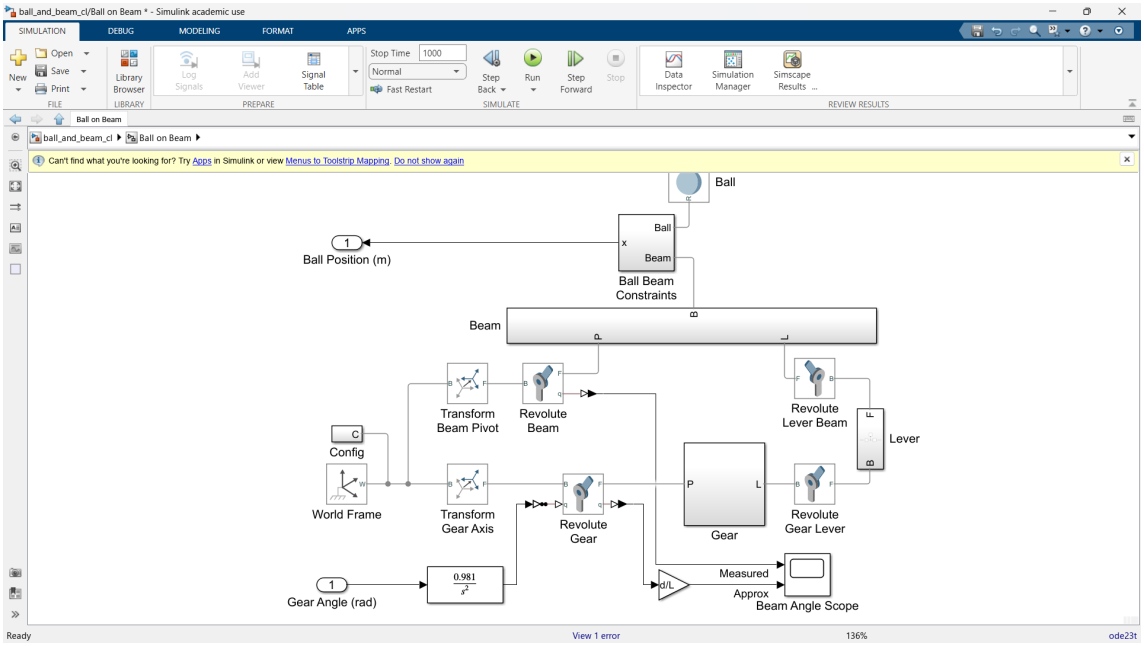


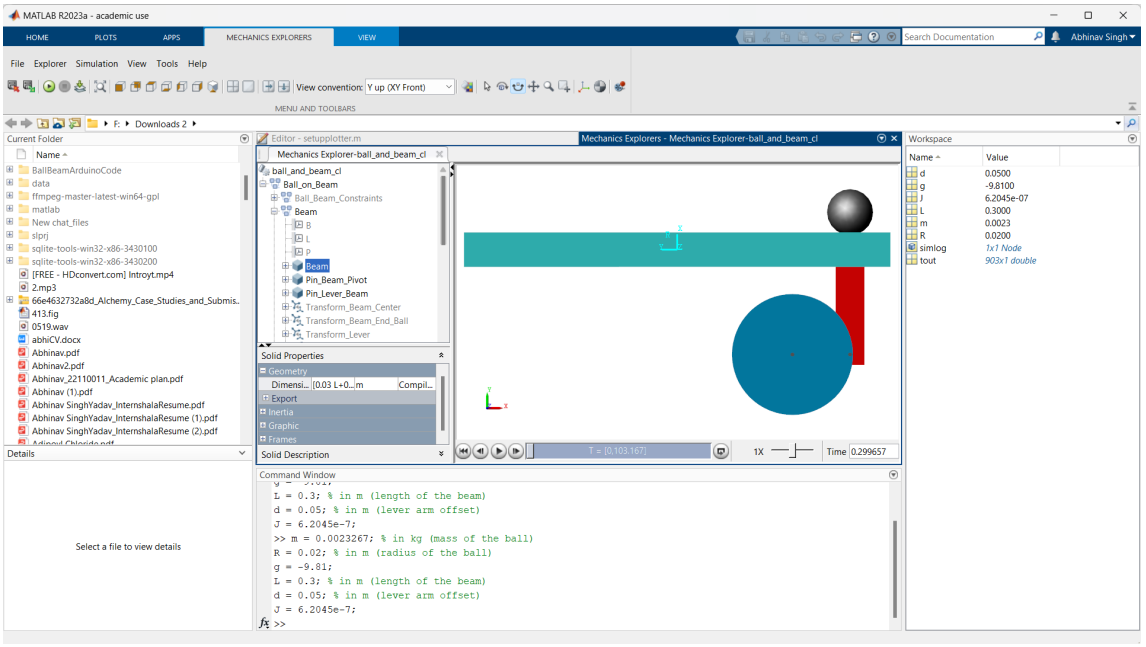
Figure 4: Output (Result)

5.3 Develop a Simscape model for the ball beam system.

Following are the images of the simscape model we have developed.
As a result, we didn't get the proper animation for stability as it should be coming out theoretically. The ball should be in the middle of the beam after some seconds, but the system is overshooting in the animation.


ball_and_beam_cl * - Simulink academic use
SIMULATION
DEBUG
MODELING
FORMAT
APPS
SUBSYSTEM BLOCK
Stop Time
1000
Normal
Fast Restart
Step Back
Run
Step Forward
Stop
Data Inspector
Simulation Manager
Simscape Results ...
REVIEW RESULTS
ball_and_beam_cl
Can't find what you're looking for? Try Apps in Simulink or view Menus to Toolstrip Mapping. Do not show again
Target Ball Position
6.9466
Lead Compensator
Gear Angle (rad)
Ball Position (m)
Ball on Beam
Ball Travel Scope
Ready
View 1 error
100%
ode23t


ball_and_beam_cl/Ball on Beam * - Simulink academic use
Ball on Beam
Ball
Ball Position (m)
Ball Beam Constraints
Beam
Transform Beam Pivot
Revolute Beam
Revolute Lever Beam
Lever
Config
World Frame
Transform Gear Axis
Revolute Gear
Gear
Revolute Gear Lever
Gear Angle (rad)
Measured
Approx
Beam Angle Scope
Ready
View 1 error
136%
ode23t



6 Task 5: Physical System

6.1 Develop a CAD model of the system and build the physical system.

Note: Since fretboard was used later and can be counted as optional, such as a CAD drawing, we have not included it.

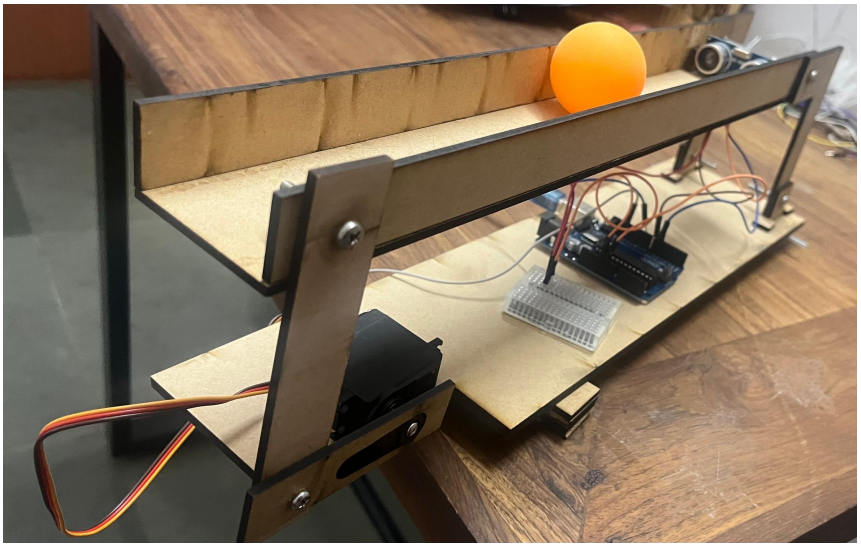


Figure 5: Front view of the physical model

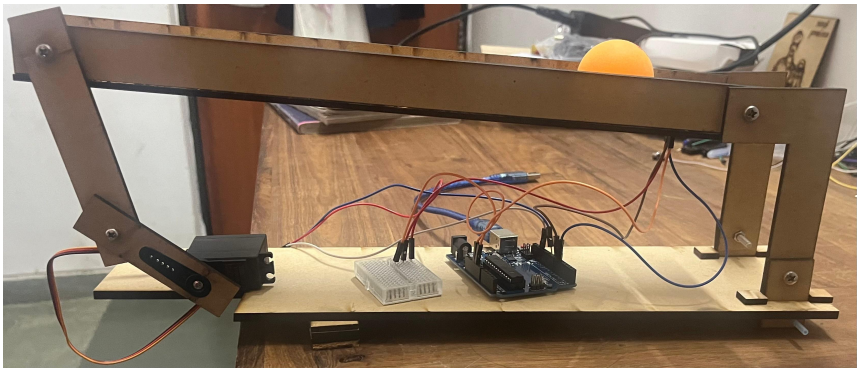


Figure 6: Motor at different angle

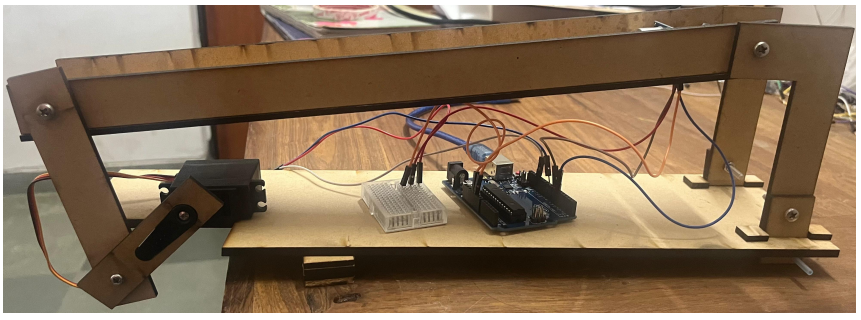


Figure 7: Motor with different angle

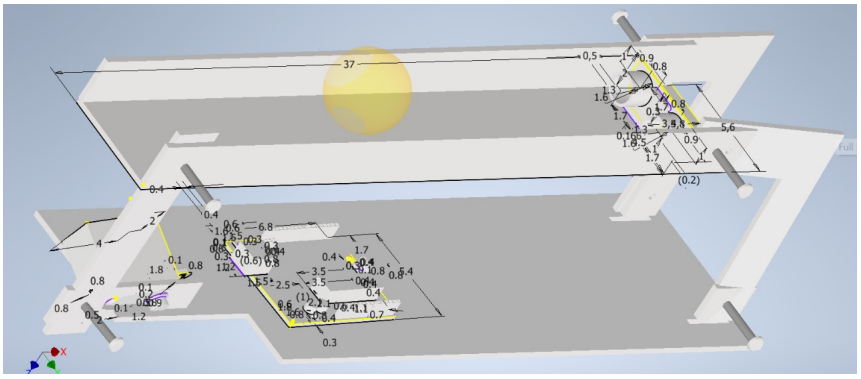


Figure 8: Cad model

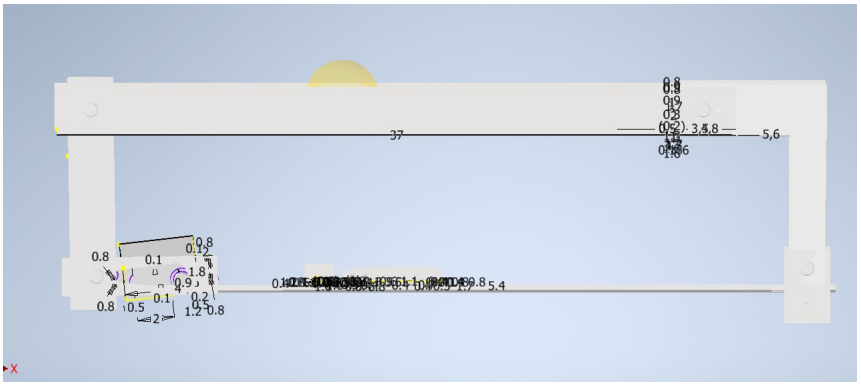


Figure 9: Side view of the cad model

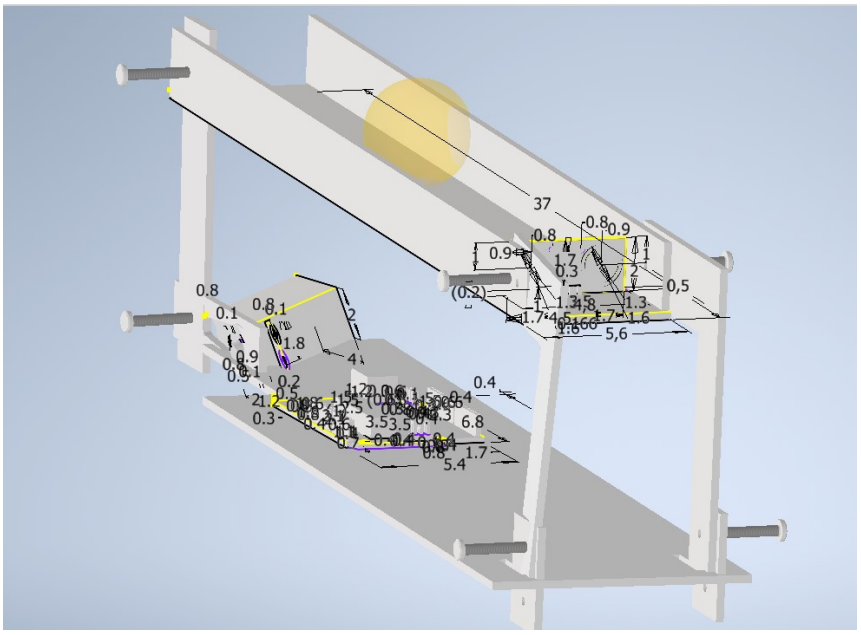


Figure 10: back view of the cad model

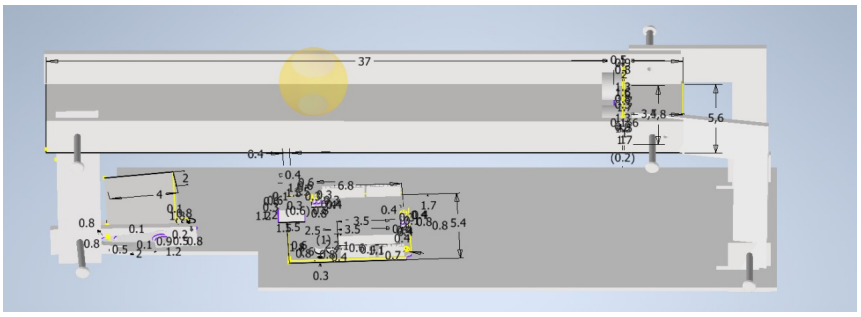


Figure 11: Top view of the cad model

6.2 Design a PID controller to stabilize the ball on the beam at the given distance from the end. Comment on your observations and challenges in implementing the PID control.

We have completed the tuning of the PID controller to achieve optimal performance in stabilizing the ball on the beam at the specified distance from the end. The final tuning values used were: $K_p = 3$, $K_i = 1$, and $K_d = 2$, which successfully minimized the error and improved the system's response.

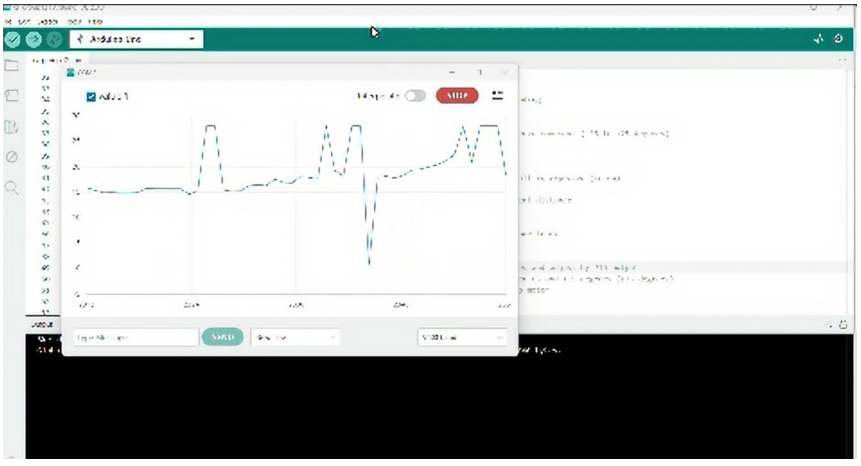


Figure 12: Plot for PID Controller

- **Response Time:** Achieving a balance between a fast response and overshoot has been difficult. A high proportional gain can lead to quicker stabilization but may also cause the system to overshoot the desired position.
- **Stability Issues:** As we increase the integral gain to eliminate steady-state error, the system has shown signs of instability, resulting in oscillations around the setpoint. This necessitates careful tuning to avoid excessive oscillation.

- **Motor Behavior:** The motor continues to rotate even after the beam reaches the midpoint, leading to overshooting and instability in the system, which requires further control adjustments.
- **Noise Sensitivity:** The derivative term can make the system sensitive to noise in the measurement of the ball's position, leading to erratic control actions. We are considering implementing filtering techniques to mitigate this issue.
- **System Dynamics:** Understanding the dynamics of the physical system has required iterative testing and adjustment. The actual behavior of the ball on the beam can differ from the theoretical model, necessitating real-time adjustments to the PID parameters.

7 Problems Faced

1. **Motor Issues:** The motor occasionally rotated 360 degrees unpredictably, disrupting the stabilization of the ball.
2. **Simulink Complexity:** As first-time users of Simscape, we faced challenges in creating and understanding the block diagrams effectively.
3. **Model Accuracy:** The initial Simulink models did not accurately represent the physical dynamics of the ball and beam system, necessitating extensive validation and adjustments.
4. **Arduino Code Challenges:** Writing the Arduino code for controlling the system proved difficult, involving significant debugging and refinement to achieve desired functionality.
5. **Sensor Calibration:** Proper calibration of the sensors was challenging, leading to inaccuracies in feedback and control.

8 Acknowledgement and References

We sincerely thank Professor Vineet Vashista for giving us this incredible opportunity and for his constant encouragement and motivation throughout the project. His belief in our potential pushed us to take on this challenge and grow technically and personally.

We are also deeply grateful to Mr.Rajdeep and Mr.Jenishkumar for their invaluable assistance. Their guidance and patience in clarifying our doubts significantly impacted our understanding and progress.

This project has taught us many things and inspired us to surpass our limits. We are truly thankful for all the support we have received.

9 References

These are the references used:

1 https://mechatronicstutorials.blogspot.com/2014/07/balancing-of-ball-on-beam-u
html

2 https://www.researchgate.net/publication/312965764_Modelling_and_
control_of_ball_and_beam_system_using_PID_controller

3 https://ctms.engin.umich.edu/CTMS/index.php?example=BallBeam&section=
SimulinkSimscape