A Report On

Control Systems

by **K. Sai Ruthik.**

Date of Submission:20/07/2024.

Training Program Name: Angstromers.

Github Link: Click Here

List Of Contents

Section 1
Introduction

1.1 Theory:

In the Introduction section we learnt system modeling emphasizes the importance of developing mathematical representations of dynamic systems, crucial for control design. Two primary representations are discussed: **State-space** and **Transfer function** models. Dynamic systems evolve over time according to fixed rules, often represented by first-order differential equations.

What are Dynamic Systems?

Dynamic systems are systems that change over time according to specific rules or equations. They are characterized by their state, which represents the configuration of the system at a given time. The behaviour of dynamic systems can be described using differential equations, which express how the state evolves based on inputs and initial conditions.

$$\dot{\mathbf{x}} = \frac{d\mathbf{x}}{dt} = \mathbf{f}\left(\mathbf{x}(t), \mathbf{u}(t), t\right)$$

Where:

x(t) : State Vector at time t.

u(t) : Vector of external inputs to the system at time t.

f : Function that describes how the state of the system evolves over time.

dx/dt :It signifies how the state of the system changes as time progresses.

What is State Space Representation?

The state-space representation is a mathematical model of a physical system as a set of input, output, and state variables related by first-order differential equations. The state-space model represents the system in terms of state variables, input variables, and output variables.

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}$$

$$\mathbf{y} = C\mathbf{x} + D\mathbf{u}$$

Where:

x˙: Time derivative of state vector
u: Input Vector
v: Output vector
A: (n*n) Input Matrix
B: (n*p) Input Matrix
C: (q*n) Output Matrix
D: (q*p) Output Matrix

What is Transfer Function?

The Laplace transform is a mathematical technique used to transform a time-domain function into a frequency-domain function. This transformation is particularly useful for analysing linear time-invariant (LTI) systems, as it converts differential equations into algebraic equations, simplifying the analysis and design of these systems.

The Laplace transform of a time domain function, $f(t)$, is defined below:

$$F(s) = \mathcal{L}\{f(t)\} = \int_0^\infty e^{-st} f(t)dt$$

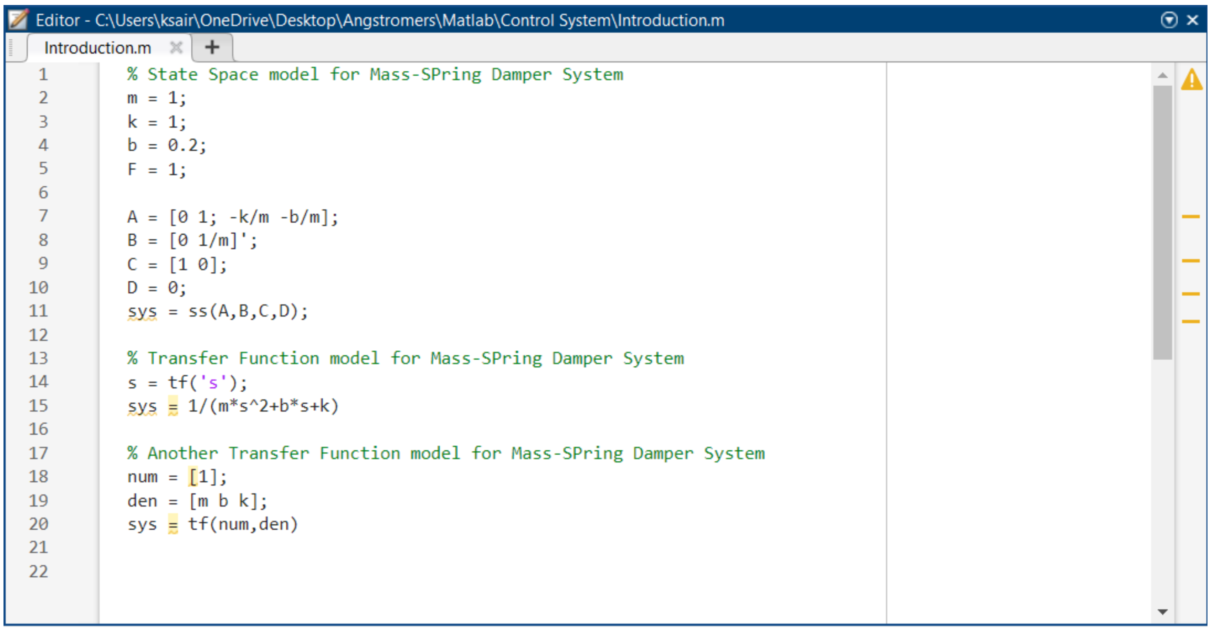
Where:

F(s):It is the Laplace Transform of f(t)
L{f(t)} :Denoted the Laplace Transform operator applied to f(t).

1.2 Mass Spring Damper System:

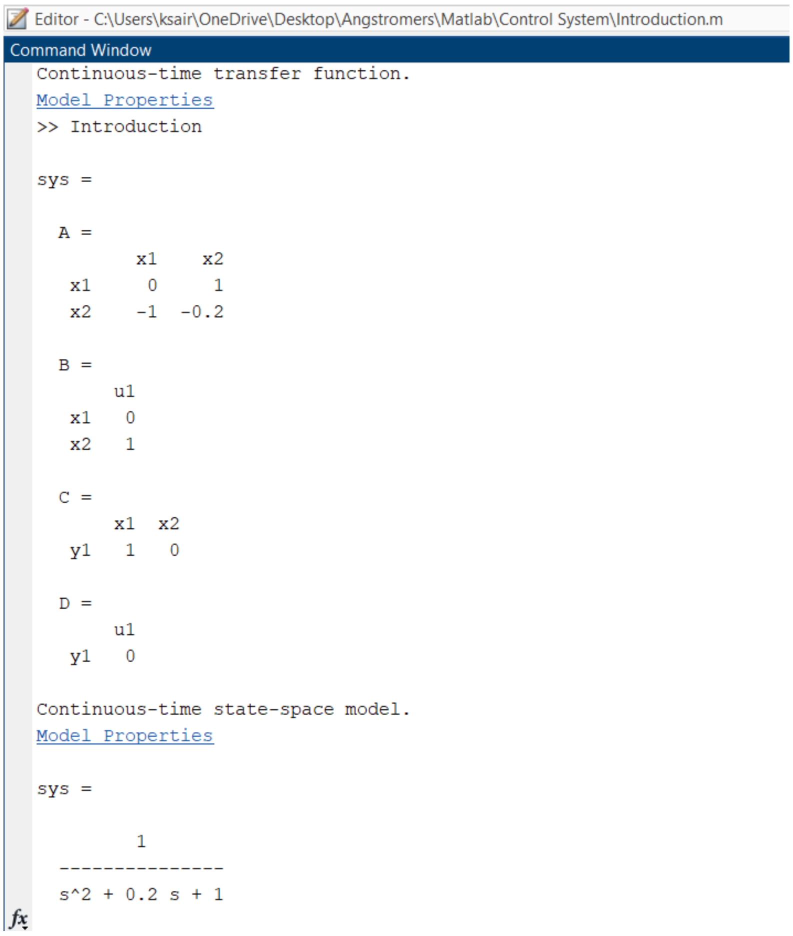
A mass-spring-damper system is a fundamental model used in mechanical engineering to describe the dynamics of a mass attached to a spring and a damper. This system is commonly used to analyze and understand vibration, control, and dynamic behaviour of mechanical systems.

1.3 Implementation:


Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\Introduction.m
Introduction.m

```matlab
% State Space model for Mass-SPring Damper System
m = 1;
k = 1;
b = 0.2;
F = 1;

A = [0 1; -k/m -b/m];
B = [0 1/m]';
C = [1 0];
D = 0;
sys = ss(A,B,C,D);

% Transfer Function model for Mass-SPring Damper System
s = tf('s');
sys = 1/(m*s^2+b*s+k)

% Another Transfer Function model for Mass-SPring Damper System
num = [1];
den = [m b k];
sys = tf(num,den)
```

Output:


Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\Introduction.m
Command Window

```
Continuous-time transfer function.
Model Properties
>> Introduction

sys =

  A =
          x1    x2
   x1      0     1
   x2     -1  -0.2

  B =
       u1
   x1   0
   x2   1

  C =
       x1  x2
   y1   1   0

  D =
       u1
   y1   0

Continuous-time state-space model.
Model Properties

sys =

         1
  ---------------
  s^2 + 0.2 s + 1
```

1.4 Use Case of Mass Spring Damp System:

Vehicle Suspension Systems:

In automotive engineering, the suspension system of a vehicle is modeled as a mass-spring-damper system to study and improve ride comfort and handling.

1)Objective: The primary goal of a vehicle suspension system is to isolate the vehicle body from road irregularities and to ensure good road handling and comfort for passengers.

2)Components:

- **Mass**: Represents the vehicle's body.
- **Spring**: Represents the suspension springs that provide support and return the vehicle to its normal height after compression.
- **Damper**: Represents the shock absorbers that dissipate kinetic energy and control the motion of the springs.

3)Outcome: By optimizing the suspension system, the vehicle can provide a smoother ride, improved safety, and better handling characteristics, enhancing the overall driving experience.

Section 2
Cruise Control

2.1 Introduction:

Cruise control is an automated system in vehicles designed to maintain a constant speed set by the driver, without the need for continuous pedal input. This system exemplifies a feedback control system, utilizing a speed sensor to monitor the vehicle's current speed. The core component, the controller, compares this speed with the desired set speed. When a discrepancy is detected, the controller sends signals to an actuator to adjust the throttle position, either increasing or decreasing the vehicle's speed as needed. This feedback loop ensures that the vehicle maintains the set speed even when road conditions change, such as going uphill or downhill. Modern cruise control systems often integrate with other vehicle systems for enhanced functionality, such as adaptive cruise control, which uses additional sensors to maintain a safe distance from other vehicles.

2.2 System Equation:

The System Equation for the Cruise Control system is:

Given the mass-damper system, the governing differential equation is:

$$m\dot{v} + bv = u$$

Where,

- m is the mass of the vehicle
- b is the damping coefficient
- v is the velocity of the vehicle
- u is the control input force

State Space Model:

The state-space model provides a mathematical framework to describe the dynamics of a system using a set of first-order differential equations. For the cruise control system, which is a first-order mass-damper system, the state-space representation captures the relationship between the input, the state, and the output of the system.

The equations of State Space Model for Cruise Control is:

$$\dot{\mathbf{x}} = [\dot{v}] = \left[\frac{-b}{m}\right][v] + \left[\frac{1}{m}\right][u]$$

$$y = [1][v]$$

Transfer Function Model:

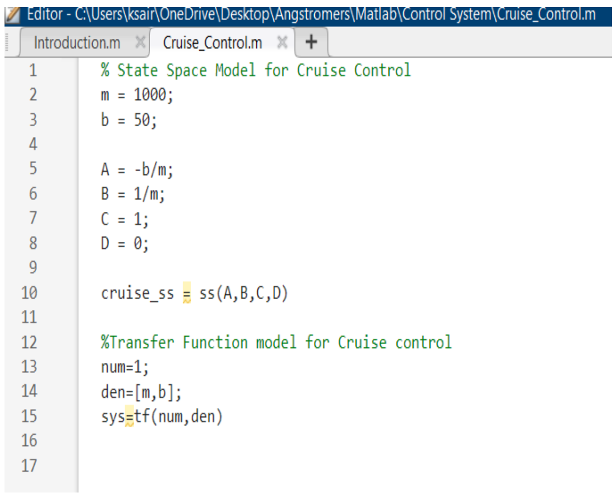
For the cruise control system, the transfer function P(s) is derived from the governing differential equation, capturing how the velocity V(s) of the vehicle responds to the input force U(s)

The transfer function for the cruise control system is given by:

$$P(s) = \frac{V(s)}{U(s)} = \frac{1}{ms+b} \qquad [\frac{m/s}{N}]$$

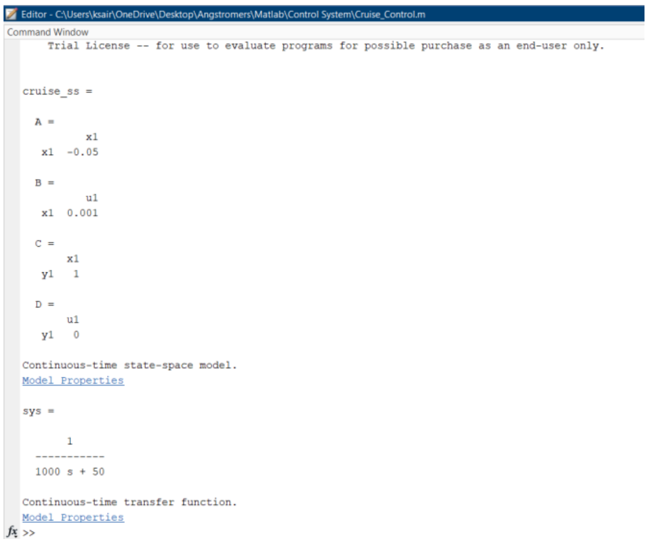
where s is the Laplace variable. This transfer function characterizes the relationship between the input force applied and the resulting vehicle speed in the frequency domain.

2.3 Implementation


Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\Cruise_Control.m
Introduction.m
Cruise_Control.m

```
% State Space Model for Cruise Control
m = 1000;
b = 50;

A = -b/m;
B = 1/m;
C = 1;
D = 0;

cruise_ss = ss(A,B,C,D)

%Transfer Function model for Cruise control
num=1;
den=[m,b];
sys=tf(num,den)
```

Output:


Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\Cruise_Control.m
Command Window

```
    Trial License -- for use to evaluate programs for possible purchase as an end-user only.


cruise_ss =

  A =
         x1
   x1  -0.05

  B =
         u1
   x1  0.001

  C =
       x1
   y1   1

  D =
       u1
   y1   0

Continuous-time state-space model.
Model Properties

sys =

       1
  -----------
  1000 s + 50

Continuous-time transfer function.
Model Properties
>>
```

2.4 Use Case of Cruise Control

Long Distance Driving Highway:

Implementation:

In long-distance highway driving, cruise control is employed to maintain a steady vehicle speed, improving driving comfort and efficiency. Once the driver sets the desired speed using the cruise control system, the system automatically adjusts the throttle to keep the vehicle at that speed. This reduces the need for constant foot adjustments on the accelerator, allowing the driver to focus more on the road and less on speed regulation.

Outcome:

The use of cruise control during highway trips results in a more relaxed driving experience and can lead to improved fuel efficiency. By maintaining a consistent speed, the vehicle avoids unnecessary acceleration and braking, which helps in reducing fuel consumption and wear on the braking system. Additionally, this consistent speed can contribute to safer driving by minimizing the chances of speeding and sudden speed changes.

SECTION 3
Motor Speed

3.1 Introduction

Motor speed, a crucial parameter in control systems, refers to the rate at which a motor's rotor turns. For a DC motor, the speed is directly related to the voltage applied to the armature and inversely related to the load torque. The motor's speed is influenced by several factors, including the armature current, which generates torque, and the back electromotive force (EMF), which opposes the applied voltage. In a typical DC motor setup, the speed can be controlled by adjusting the input voltage or by modifying the load conditions. Understanding and managing motor speed is essential for achieving precise motion control in various applications, from robotics and automotive systems to industrial machinery. Effective speed control ensures optimal performance, efficiency, and reliability of the system, making it a fundamental aspect of designing and implementing control strategies.

3.2 System Equation:

The system equation for the DC motor is:

Given the DC motor system, the governing differential equations are:

1)Torque Equation: $J\ddot{\theta} + b\dot{\theta} = Ki$

Where,

J is the moment of inertia of the rotor
b is the damping coefficient of the rotor
θ is the angular position of the rotor
$\theta^{\cdot}$ is the angular velocity of the rotor
$\theta^{\cdot\cdot}$ is the angular acceleration of the rotor
K is the motor torque constant
i is the armature current

2)Electrical Equation: $L\frac{di}{dt} + Ri = V - K\dot{\theta}$

Where,

L is the inductance of the armature
R is the resistance of the armature
i is the armature current
V is the input voltage
θ˙ is the angular velocity of the rotor
K is the back EMF constant (equal to the motor torque constant)

State Space Model:

The state-space model describes the dynamics of the DC motor using a set of first-order differential equations. For the DC motor system, the state variables are the rotational speed θ˙ and the armature current iii. The armature voltage V is treated as the input, and the rotational speed θ˙ is chosen as the output.
The equations of the State-Space Model for the DC motor are:

$$\frac{d}{dt}\begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} = \begin{bmatrix} -\frac{b}{J} & \frac{K}{J} \\ -\frac{K}{L} & -\frac{R}{L} \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{L} \end{bmatrix} V$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix}$$

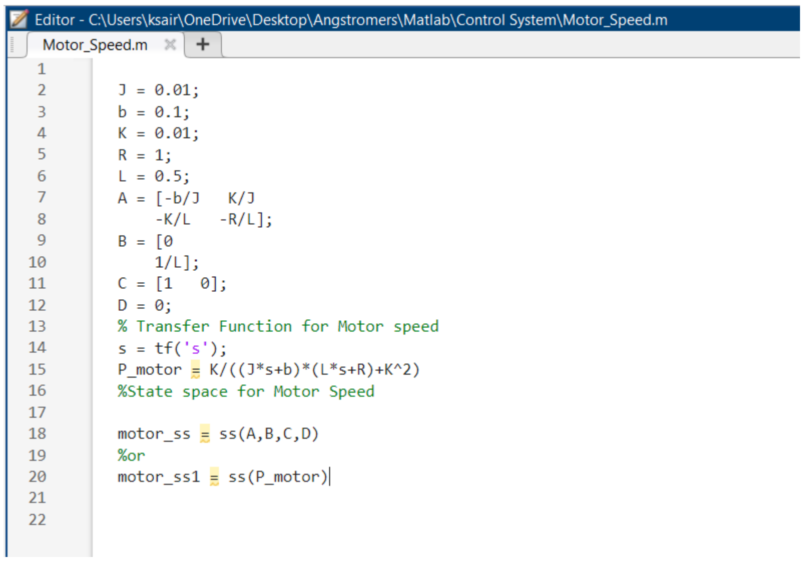
Transfer Function Model:

The transfer function provides a frequency-domain representation of the DC motor's response to an input voltage. It is derived from the Laplace-transformed governing equations and captures how the rotational speed θ˙ of the motor responds to the input voltage V.

The transfer function P(s) is given by:

$$P(s) = \frac{\dot{\Theta}(s)}{V(s)} = \frac{K}{(Js+b)(Ls+R)+K^2} \qquad [\frac{rad/sec}{V}]$$

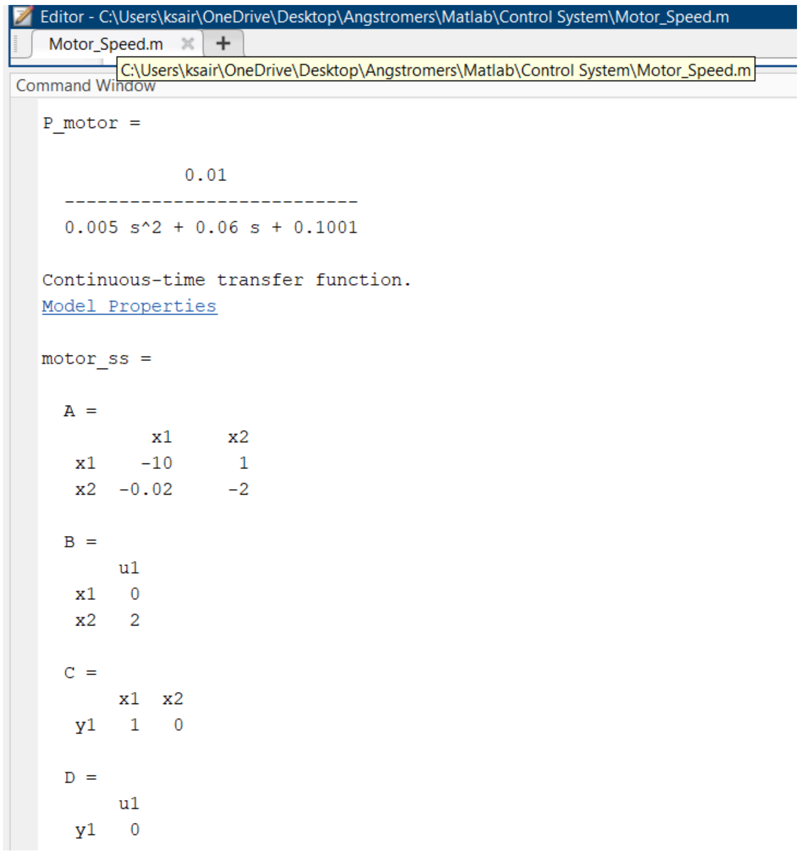
3.3 Implementation:


Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\Motor_Speed.m

Motor_Speed.m

```

J = 0.01;
b = 0.1;
K = 0.01;
R = 1;
L = 0.5;
A = [-b/J   K/J
    -K/L   -R/L];
B = [0
    1/L];
C = [1   0];
D = 0;
% Transfer Function for Motor speed
s = tf('s');
P_motor = K/((J*s+b)*(L*s+R)+K^2)
%State space for Motor Speed

motor_ss = ss(A,B,C,D)
%or
motor_ss1 = ss(P_motor)
```

Output:


Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\Motor_Speed.m

Motor_Speed.m

C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\Motor_Speed.m

Command Window

```
P_motor =

              0.01
  ---------------------------
  0.005 s^2 + 0.06 s + 0.1001

Continuous-time transfer function.
Model Properties

motor_ss =

  A =
           x1      x2
   x1     -10       1
   x2   -0.02      -2

  B =
       u1
   x1   0
   x2   2

  C =
       x1  x2
   y1   1   0

  D =
       u1
   y1   0
```

3.4 Use Case of Motor Speed

Automated Conveyor System

Implementation:

In an automated conveyor system used in a manufacturing plant, DC motors control the speed of conveyor belts to ensure precise and efficient material handling. For instance, the system utilizes DC motors with speed control to adjust the belt's speed based on real-time data from sensors monitoring the production line. The speed of each motor is regulated by a feedback control system, where the desired speed setpoint is compared to the actual speed of the conveyor belt. Adjustments are made through a PID controller that modulates the input voltage to the DC motor to achieve the desired speed.

Outcome:

Implementing motor speed control in the conveyor system leads to enhanced operational efficiency and consistency. The precise speed regulation ensures that materials are transported at the correct rate, minimizing delays and reducing the likelihood of bottlenecks. As a result, the production process becomes more streamlined, improving overall throughput and reducing downtime. Additionally, accurate speed control helps in maintaining product quality by ensuring that items are handled gently and processed in a timely manner, which ultimately leads to increased productivity and cost savings.

SECTION 4
Motor Position

4.1 Introduction

Motor position in control systems refers to the specific angular location of a motor's rotor, which directly influences the mechanical output of a system. Accurate position control is essential in applications where precise alignment or movement is required, such as in robotics, automated machinery, and aerospace systems. In a control system, the motor position is typically monitored using sensors like encoders or resolvers, which provide feedback on the rotor's angular displacement. This feedback is crucial for implementing control strategies that adjust the motor's input to achieve the desired position. Position control ensures that the motor reaches and maintains a specified angular position with minimal error, improving the system's performance and reliability.

4.2 System Equations

The system equation for the DC motor is:

Given the DC motor, the governing differential equations are:

$$J\ddot{\theta} + b\dot{\theta} = Ki$$

$$L\frac{di}{dt} + Ri = V - K\dot{\theta}$$

These equations describe the dynamics of the DC motor, including how the motor speed ($\dot{\theta}$) and the armature current (i) evolve in response to changes in input voltage (V) and mechanical load.

State Space Model:

The state-space model provides a mathematical framework to describe the dynamics of the DC motor system using a set of first-order differential equations. For the DC motor, the state variables are the motor position θ, motor speed θ˙, and armature current i. The armature voltage V is the input, and the motor position θ is the output.

The equations of the State-Space Model for the DC motor are:

$$\frac{d}{dt}\begin{bmatrix} \theta \\ \dot{\theta} \\ i \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{b}{J} & \frac{K}{J} \\ 0 & -\frac{K}{L} & -\frac{R}{L} \end{bmatrix}\begin{bmatrix} \theta \\ \dot{\theta} \\ i \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L} \end{bmatrix} V$$

$$y = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} \theta \\ \dot{\theta} \\ i \end{bmatrix}$$

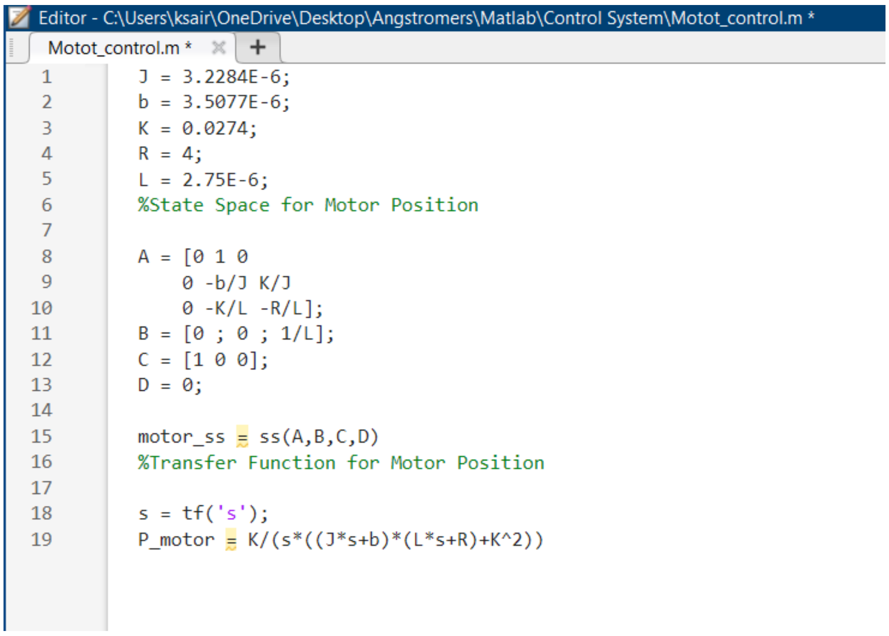
Transfer Function Model:

The transfer function represents the DC motor's response in the frequency domain, describing how the motor position θ(s) relates to the input voltage V(s). The transfer function is derived by applying the Laplace transform to the governing differential equations and then solving for the output-to-input ratio.

For the motor position θ(s), we divide the transfer function for speed by s to account for the integration of speed to obtain position:

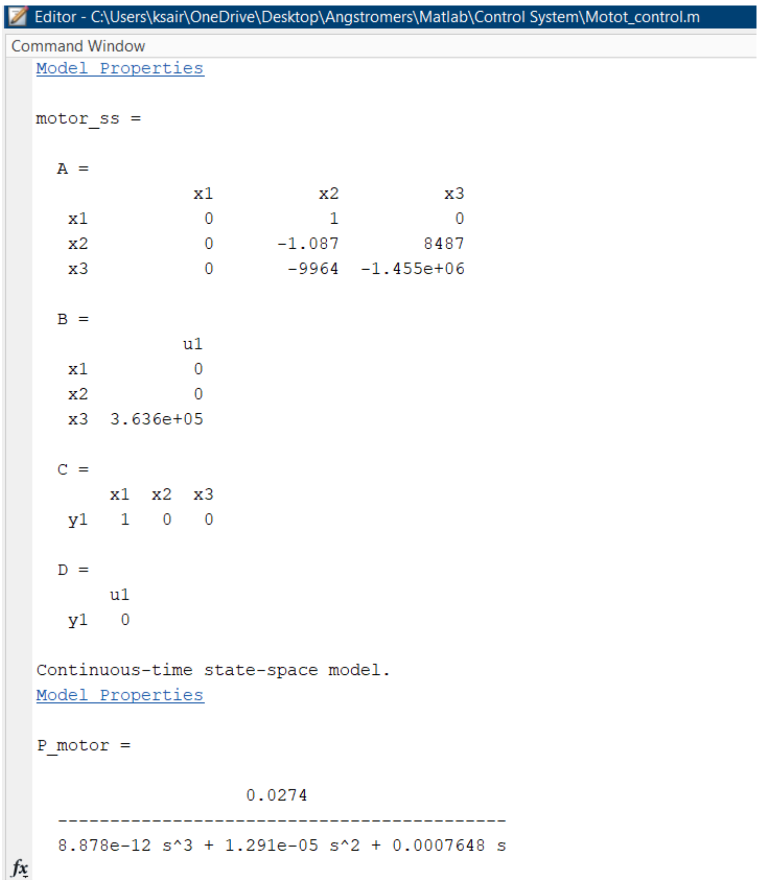
$$\frac{\Theta(s)}{V(s)} = \frac{K}{s((Js+b)(Ls+R)+K^2)} \qquad [\frac{rad}{V}]$$

4.3 Implementation


Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\Motot_control.m *
Motot_control.m *

```matlab
J = 3.2284E-6;
b = 3.5077E-6;
K = 0.0274;
R = 4;
L = 2.75E-6;
%State Space for Motor Position

A = [0 1 0
    0 -b/J K/J
    0 -K/L -R/L];
B = [0 ; 0 ; 1/L];
C = [1 0 0];
D = 0;

motor_ss = ss(A,B,C,D)
%Transfer Function for Motor Position

s = tf('s');
P_motor = K/(s*((J*s+b)*(L*s+R)+K^2))
```

Output:


Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\Motot_control.m
Command Window

```
Model Properties

motor_ss =

  A =
                x1          x2          x3
   x1            0           1           0
   x2            0      -1.087        8487
   x3            0       -9964  -1.455e+06

  B =
              u1
   x1          0
   x2          0
   x3  3.636e+05

  C =
       x1  x2  x3
   y1   1   0   0

  D =
       u1
   y1   0

Continuous-time state-space model.
Model Properties

P_motor =

                   0.0274
  -------------------------------------------
  8.878e-12 s^3 + 1.291e-05 s^2 + 0.0007648 s
```

4.4 Use Case of Motor Position

Automated Camera Pan-and-Tilt System

Motor position control is vital in automated camera systems used for surveillance and broadcasting. In such systems, precise positioning of the camera is necessary to capture desired angles and maintain focus on specific subjects. The camera's pan-and-tilt mechanism, which relies on DC motors, adjusts the camera's orientation horizontally and vertically. By accurately controlling the motor positions, the system ensures the camera can follow moving objects or cover a specified area. This capability is particularly useful in security systems where continuous monitoring of large areas is required, or in live broadcasting where camera angles need to be dynamically adjusted for optimal coverage.

Implementation and Outcome:

In practice, the pan-and-tilt system can be implemented using a feedback control loop where the camera’s position is constantly monitored using encoders. The encoder data, which provides the current position of the camera, is fed back into a PID controller that adjusts the motor's position accordingly. For instance, if the camera is set to track a moving object, the PID controller will continuously adjust the pan and tilt angles to keep the object in the frame. The outcome of this setup is a highly responsive and accurate camera system capable of maintaining precise positioning. This enhances surveillance capabilities by providing clear and stable images or video feeds, improves the effectiveness of live broadcasts by offering dynamic camera angles, and reduces the need for manual intervention in camera operations.

SECTION 5

Suspension

5.1 Introduction

In control systems, designing an automotive suspension system is crucial for enhancing vehicle performance and passenger comfort. The suspension system is responsible for absorbing shocks from the road and maintaining vehicle stability and handling. In many design scenarios, a simplified 1/4 vehicle model is used, representing one wheel of the vehicle along with its associated spring and damper components. This model abstracts the complex interactions of the full vehicle suspension system into a more manageable 1-D spring-damper system. For active suspension systems, an actuator is integrated to dynamically adjust the control force, optimizing the vehicle's response to road disturbances. By actively controlling the suspension forces, the system can minimize body motion and improve ride quality and handling.

5.2 System Equations

The system equation for the automotive suspension system is:

Given the 1/4 vehicle model, the governing differential equations are:

1. **Body Motion Equation**:

$$M_1\ddot{X}_1 = -b_1(\dot{X}_1 - \dot{X}_2) - K_1(X_1 - X_2) + U$$

Where,

M1 is the mass of the vehicle body
X1 is the vertical displacement of the vehicle body
$\dot{X}1$ is the velocity of the vehicle body
$\ddot{X}1$ is the acceleration of the vehicle body
b1 is the damping coefficient of the suspension system
K1 is the spring constant of the suspension
U is the control force applied by the actuator.

2)Wheel Motion Equation:

$$M_2\ddot{X}_2 = b_1(\dot{X}_1 - \dot{X}_2) + K_1(X_1 - X_2) + b_2(\dot{W} - \dot{X}_2) + K_2(W - X_2) - U$$

Where:
M2is the mass of the wheel assembly
X2 is the vertical displacement of the wheel
X˙2 is the velocity of the wheel
X¨2 is the acceleration of the wheel
b2 is the damping coefficient of the wheel
K2 is the spring constant of the wheel suspension
W is the road disturbance or input
W˙ is the velocity of the road disturbance

Transfer Function Model:

For the automotive suspension system, the transfer functions characterize how the vertical displacement difference X1(s)−X2(s) responds to different inputs. The system equations involve both the control force U(s) and road disturbance W(s).

1. Transfer Function G1(s):

When considering the effect of the control input U(s) only (with W(s)=0), the transfer function G1(s) is derived as follows:

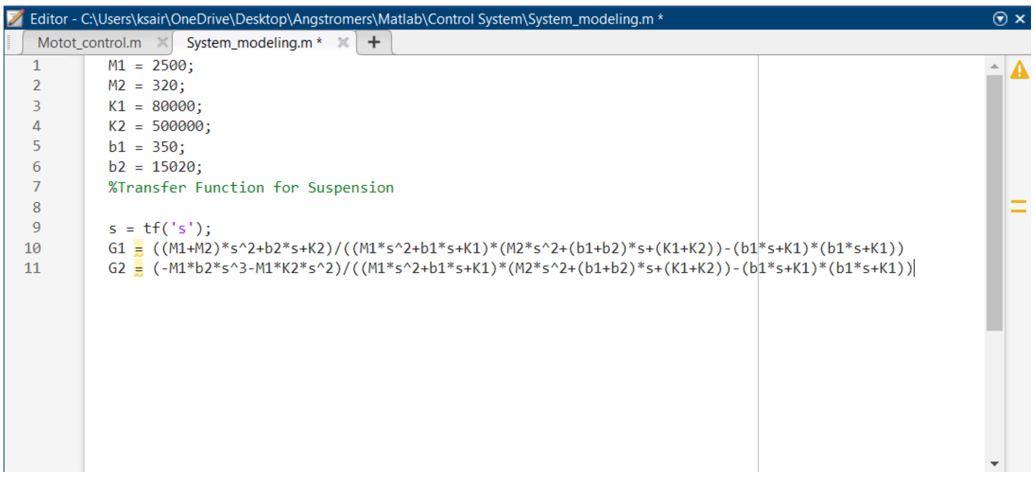
$$G_1(s) = \frac{X_1(s) - X_2(s)}{U(s)} = \frac{(M_1 + M_2)s^2 + b_2 s + K_2}{\Delta}$$

2. Transfer Function G2(s):

When considering the effect of the road disturbance W(s)W(s)W(s) only (with U(s)=0), the transfer function G2(s) is given by:

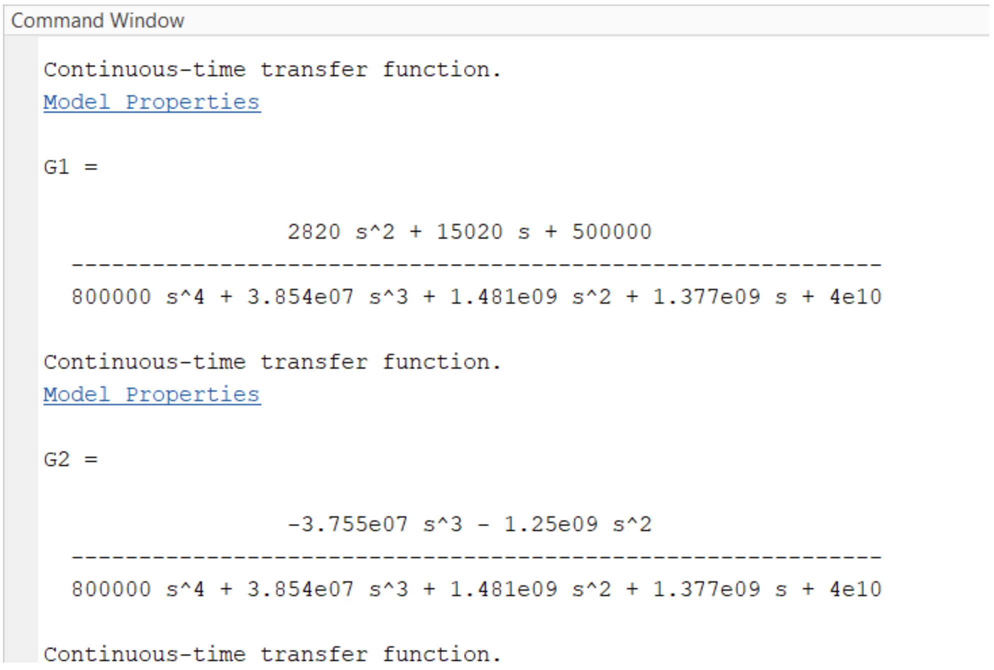
$$G_2(s) = \frac{X_1(s) - X_2(s)}{W(s)} = \frac{-M_1 b_2 s^3 - M_1 K_2 s^2}{\Delta}$$

5.3 Implementation:


Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\System_modeling.m *

```matlab
M1 = 2500;
M2 = 320;
K1 = 80000;
K2 = 500000;
b1 = 350;
b2 = 15020;
%Transfer Function for Suspension

s = tf('s');
G1 = ((M1+M2)*s^2+b2*s+K2)/((M1*s^2+b1*s+K1)*(M2*s^2+(b1+b2)*s+(K1+K2))-(b1*s+K1)*(b1*s+K1))
G2 = (-M1*b2*s^3-M1*K2*s^2)/((M1*s^2+b1*s+K1)*(M2*s^2+(b1+b2)*s+(K1+K2))-(b1*s+K1)*(b1*s+K1))
```

Output:



```
Continuous-time transfer function.
Model Properties

G1 =

                    2820 s^2 + 15020 s + 500000
  ------------------------------------------------------------
  800000 s^4 + 3.854e07 s^3 + 1.481e09 s^2 + 1.377e09 s + 4e10

Continuous-time transfer function.
Model Properties

G2 =

                    -3.755e07 s^3 - 1.25e09 s^2
  ------------------------------------------------------------
  800000 s^4 + 3.854e07 s^3 + 1.481e09 s^2 + 1.377e09 s + 4e10

Continuous-time transfer function.
```

5.4 Use Case of Suspension

Adaptive Suspension Control for Enhanced Ride Comfort

Implementation:

An adaptive suspension control system is implemented in a luxury vehicle to improve ride comfort and handling. The system uses a combination of sensors to monitor road conditions and vehicle dynamics in real-time. These sensors measure the road disturbances, vehicle body acceleration, and wheel displacement. Based on the data, an onboard controller adjusts the suspension characteristics by modulating the damping forces through adjustable shock absorbers. The control algorithm uses a model predictive control (MPC) approach to optimize the suspension response, reducing the impact of road bumps and vibrations on the vehicle's body. The algorithm continuously updates its control inputs based on predicted future disturbances, ensuring that the vehicle maintains optimal comfort and handling under varying driving conditions.

Outcome:

The adaptive suspension system significantly enhances ride comfort by minimizing the effects of road irregularities and vibrations transmitted to the vehicle occupants. The real-time adjustments made by the suspension system improve vehicle stability and handling, leading to a smoother driving experience. Passenger comfort is notably improved as the system effectively dampens large road disturbances and isolates the vehicle body from harsh road impacts. Additionally, the vehicle's handling and safety are enhanced as the adaptive system maintains better tire contact with the road, improving traction and control. Overall, the implementation of adaptive suspension control results in a more pleasant and controlled driving experience, aligning with the high standards expected in luxury vehicles.

SECTION 6
Inverted Pendulum

6.1 Introduction

The inverted pendulum system, mounted on a motorized cart, is a classic example frequently studied in control system courses and research due to its inherent instability and nonlinear dynamics. Without control, the pendulum would fall over, making the primary objective of the control system to balance it by applying a horizontal force to the cart. This system is analogous to the attitude control of a booster rocket during takeoff. The control input in this scenario is the horizontal force applied to the cart, while the outputs are the pendulum's angular position and the cart's horizontal position. By utilizing various control techniques, such as PID, root locus, and frequency response methods, the goal is to maintain the pendulum in an upright position despite disturbances. The system is particularly interesting because it can be used to explore both single-input, single-output (SISO) and single-input, multi-output (SIMO) control strategies, providing a rich platform for learning and experimentation in control theory.

6.2 System Equations

The System Equation for the Inverted Pendulum on a Cart system is derived from the equations of motion for both the cart and the pendulum. Given the linearized model around the vertically upward equilibrium position, the governing differential equations are:

$$(I + ml^2)\ddot{\phi} - mgl\phi = ml\ddot{x}$$

$$(M + m)\ddot{x} + b\dot{x} - ml\ddot{\phi} = u$$

Where,

M is the mass of the cart
m is the mass of the pendulum

b is the damping coefficient
I is the moment of inertia of the pendulum
l is the length to the pendulum's center of mass
g is the acceleration due to gravity
x is the horizontal position of the cart
ϕ is the angular deviation of the pendulum from the vertical
u is the control input force applied to the cart

State Space Model:

The state-space model provides a mathematical framework to describe the dynamics of a system using a set of first-order differential equations. For the inverted pendulum system, the state-space representation captures the relationship between the input force applied to the cart, the state variables (cart position, cart velocity, pendulum angle, and pendulum angular velocity), and the outputs (cart position and pendulum angle).

The linearized state-space representation for the inverted pendulum system is given by:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-(I+ml^2)b}{I(M+m)+Mml^2} & \frac{m^2gl^2}{I(M+m)+Mml^2} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-mlb}{I(M+m)+Mml^2} & \frac{mgl(M+m)}{I(M+m)+Mml^2} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{I+ml^2}{I(M+m)+Mml^2} \\ 0 \\ \frac{ml}{I(M+m)+Mml^2} \end{bmatrix} u$$

$$\mathbf{y} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u$$

Transfer Function Model:

The transfer function model for the inverted pendulum system captures the relationship between the input force applied to the cart and the resulting outputs: the pendulum's angular position and the cart's position. Below are the simplified transfer functions for both outputs.

Pendulum's Angular Position (Φ(s)) with respect to Input Force (U(s)):
Starting from the transfer function:

$$P_{pend}(s) = \frac{\Phi(s)}{U(s)} = \frac{\frac{ml}{q}s}{s^3 + \frac{b(I+ml^2)}{q}s^2 - \frac{(M+m)mgl}{q}s - \frac{bmgl}{q}} \qquad [\frac{rad}{N}]$$

We can also simplify the coefficients in the denominator similarly. Let:

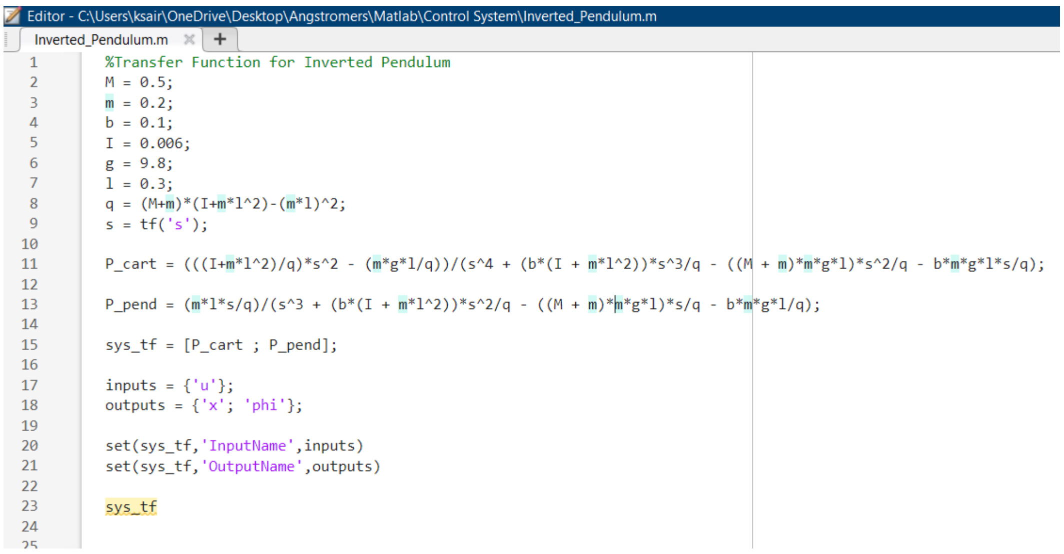
$$\begin{aligned} b_0 &= 1 \\ b_1 &= \frac{b(I+ml^2)}{q} \\ b_2 &= -\frac{(M+m)mgl}{q} \\ b_3 &= -\frac{bmgl}{q} \end{aligned}$$

Thus, the transfer function becomes:

$$P_{cart}(s) = \frac{X(s)}{U(s)} = \frac{\frac{(I+ml^2)s^2 - gml}{q}}{s^4 + \frac{b(I+ml^2)}{q}s^3 - \frac{(M+m)mgl}{q}s^2 - \frac{bmgl}{q}s} \qquad [\frac{m}{N}]$$

6.3 Implementation

Transfer Function:

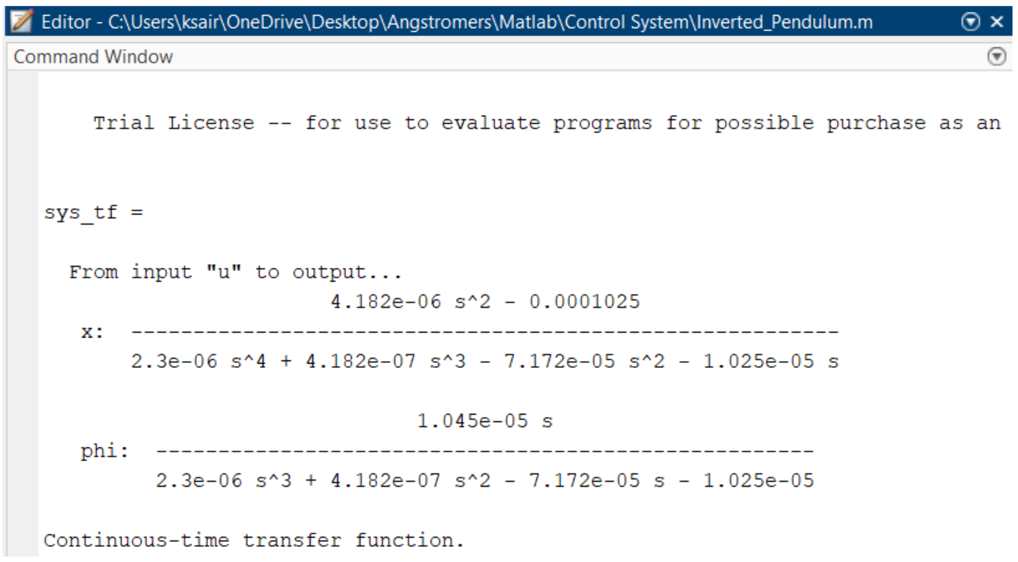


Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\Inverted_Pendulum.m

Inverted_Pendulum.m

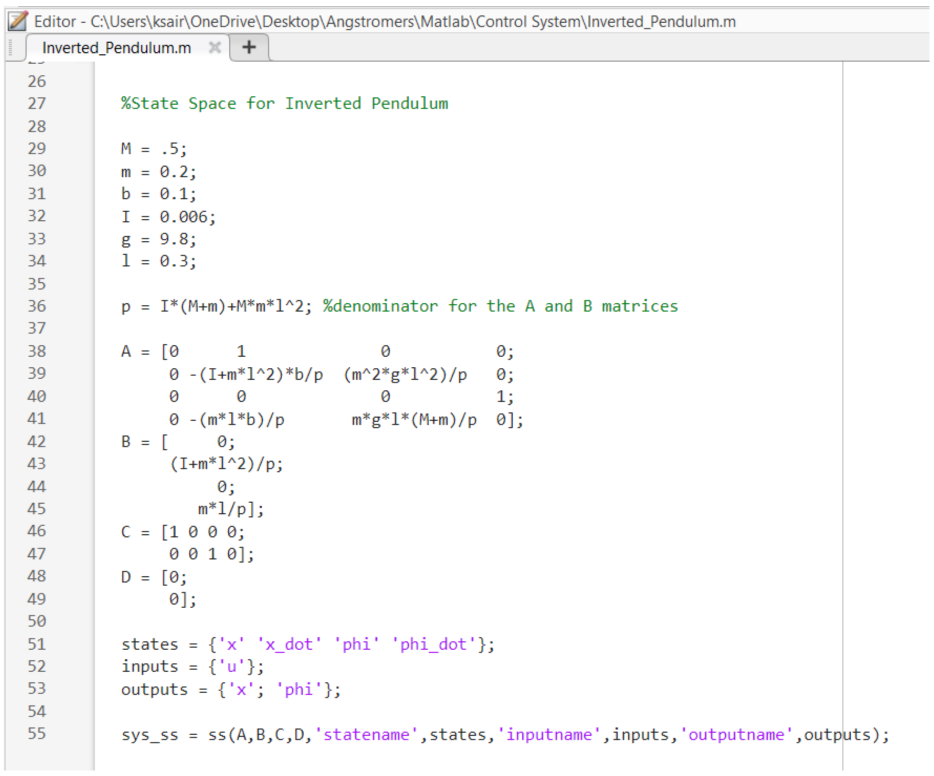
```
%Transfer Function for Inverted Pendulum
M = 0.5;
m = 0.2;
b = 0.1;
I = 0.006;
g = 9.8;
l = 0.3;
q = (M+m)*(I+m*l^2)-(m*l)^2;
s = tf('s');

P_cart = (((I+m*l^2)/q)*s^2 - (m*g*l/q))/(s^4 + (b*(I + m*l^2))*s^3/q - ((M + m)*m*g*l)*s^2/q - b*m*g*l*s/q);

P_pend = (m*l*s/q)/(s^3 + (b*(I + m*l^2))*s^2/q - ((M + m)*m*g*l)*s/q - b*m*g*l/q);

sys_tf = [P_cart ; P_pend];

inputs = {'u'};
outputs = {'x'; 'phi'};

set(sys_tf,'InputName',inputs)
set(sys_tf,'OutputName',outputs)

sys_tf
```

Output:


Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\Inverted_Pendulum.m
Command Window

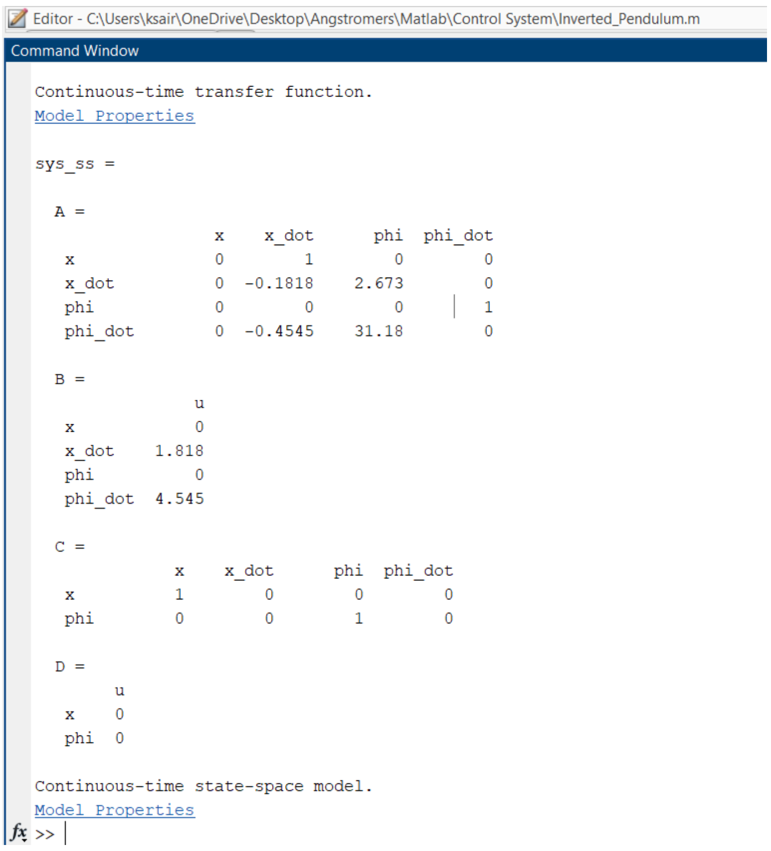
```
    Trial License -- for use to evaluate programs for possible purchase as an


sys_tf =

  From input "u" to output...
                     4.182e-06 s^2 - 0.0001025
   x:  ---------------------------------------------------------
       2.3e-06 s^4 + 4.182e-07 s^3 - 7.172e-05 s^2 - 1.025e-05 s

                            1.045e-05 s
   phi:  -----------------------------------------------------
         2.3e-06 s^3 + 4.182e-07 s^2 - 7.172e-05 s - 1.025e-05

Continuous-time transfer function.
```

State Space:


Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\Inverted_Pendulum.m
Inverted_Pendulum.m

```matlab

%State Space for Inverted Pendulum

M = .5;
m = 0.2;
b = 0.1;
I = 0.006;
g = 9.8;
l = 0.3;

p = I*(M+m)+M*m*l^2; %denominator for the A and B matrices

A = [0      1              0           0;
     0 -(I+m*l^2)*b/p  (m^2*g*l^2)/p   0;
     0      0              0           1;
     0 -(m*l*b)/p       m*g*l*(M+m)/p  0];
B = [     0;
     (I+m*l^2)/p;
          0;
        m*l/p];
C = [1 0 0 0;
     0 0 1 0];
D = [0;
     0];

states = {'x' 'x_dot' 'phi' 'phi_dot'};
inputs = {'u'};
outputs = {'x'; 'phi'};

sys_ss = ss(A,B,C,D,'statename',states,'inputname',inputs,'outputname',outputs);
```

Output:


Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\Inverted_Pendulum.m
Command Window

```
Continuous-time transfer function.
Model Properties

sys_ss =

  A =
                  x     x_dot       phi   phi_dot
   x              0         1         0         0
   x_dot          0   -0.1818     2.673         0
   phi            0         0         0         1
   phi_dot        0   -0.4545     31.18         0

  B =
                u
   x            0
   x_dot    1.818
   phi          0
   phi_dot  4.545

  C =
              x     x_dot       phi   phi_dot
   x          1         0         0         0
   phi        0         0         1         0

  D =
        u
   x    0
   phi  0

Continuous-time state-space model.
Model Properties
>> 
```

6.4 Use Case of Inverted Pendulum

Attitude Control of a Rocket During Takeoff

The inverted pendulum system is an ideal analogy for understanding the attitude control of a rocket during takeoff. In this scenario, the rocket's orientation must be maintained upright to ensure a stable ascent, akin to balancing an inverted pendulum on a moving cart. The control system must apply corrective forces to counteract any disturbances that might cause the rocket to tilt away from its desired vertical position.

To implement this, sensors are used to detect the rocket's tilt angle and rate of tilt. These measurements are fed into a control algorithm, typically a PID controller, which calculates the necessary thrust adjustments to maintain stability. The outcome of a successful control system is that the rocket remains upright and stable, minimizing deviations from its vertical trajectory. This ensures a safe and efficient ascent, preventing catastrophic failures due to tipping over. The control system's effectiveness can be measured by the quick correction of any disturbances and maintaining the rocket's angle within a small range around the vertical axis.

Implementation:

In a simulated environment, the control system for the rocket's attitude can be designed and tested using state-space models or transfer functions derived from the dynamics of an inverted pendulum. A PID controller can be tuned to provide the necessary corrections based on the feedback from the rocket's tilt sensors. The implementation involves setting up the equations of motion for the rocket, similar to the linearized equations for the inverted pendulum, and then designing the control algorithm to stabilize these equations.

The expected outcome of this implementation is that the rocket's attitude remains stable during takeoff. The control system should ensure that any deviations from the vertical position are corrected within a specified time frame, usually a few seconds, and that the rocket does not tilt beyond a critical angle that could lead to instability. The performance of the control system can be validated through simulations and, eventually, real-world testing, where the rocket is subjected to various disturbances to ensure the robustness of the control algorithm. Successful implementation leads to a stable ascent, reducing the risk of failure and improving the reliability of the rocket's launch.

SECTION 7
Air Craft Pitch

7.1 Introduction

The inverted pendulum is a classic problem in control systems and engineering due to its intrinsic instability and nonlinear dynamics. Essentially, it involves a pendulum with its center of mass above its pivot point, such that it must be actively balanced to prevent it from falling. This system serves as an archetype for understanding and designing controllers for unstable systems. It is widely studied in textbooks and research, often implemented using a motorized cart that moves horizontally to balance the pendulum. The control objective is to apply a force to the cart to keep the pendulum upright. The inverted pendulum problem is not only academically intriguing but also has real-world applications, such as in the design of self-balancing robots, Segways, and the attitude control of rockets. Due to its complexity and relevance, it is an essential problem for learning state-space representation, PID control, and other control strategies.

7.2 System Equations

The system equation for the aircraft pitch control system can be derived from the longitudinal equations of motion for the aircraft, assuming steady-cruise at constant altitude and velocity. Under these conditions, the longitudinal dynamics of the aircraft can be described by the following differential equations:

$$\dot{\alpha} = \mu\Omega\sigma[-(C_L + C_D)\alpha + \frac{1}{(\mu - C_L)}q - (C_W \sin\gamma)\theta + C_L]$$

$$\dot{\theta} = \Omega q$$

Where,

α is the angle of attack,

θ is the pitch angle,
q is the pitch rate,
δ is the elevator deflection angle (control input),
Cl, Cd, and Cw are aerodynamic coefficients,
μ,Ω, and σ are constants related to the aircraft's dynamics.

State Space Model:

The state-space model provides a mathematical framework to describe the dynamics of a system using a set of first-order differential equations. For the aircraft pitch control system, the state-space representation captures the relationship between the input (elevator deflection angle δ), the state (angle of attack α\alphaα, pitch rate q, and pitch angle θ), and the output (pitch angle θ) of the system.

The equations of the state-space model for aircraft pitch control are:

$$\begin{bmatrix} \dot{\alpha} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} -0.313 & 56.7 & 0 \\ -0.0139 & -0.426 & 0 \\ 0 & 56.7 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} 0.232 \\ 0.0203 \\ 0 \end{bmatrix} [\delta]$$

Since our output is the pitch angle, the output equation is:

$$y = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ q \\ \theta \end{bmatrix}$$

Transfer Function Model:

For the aircraft pitch control system, the transfer function P(s) is derived from the governing differential equations, capturing how the pitch angle Θ(s) of the aircraft responds to the input elevator deflection angle Δ(s). The Laplace transform of the linearized system equations, assuming zero initial conditions, yields the following transfer function:

$$sA(s) = -0.313A(s) + 56.7Q(s) + 0.232\Delta(s)$$

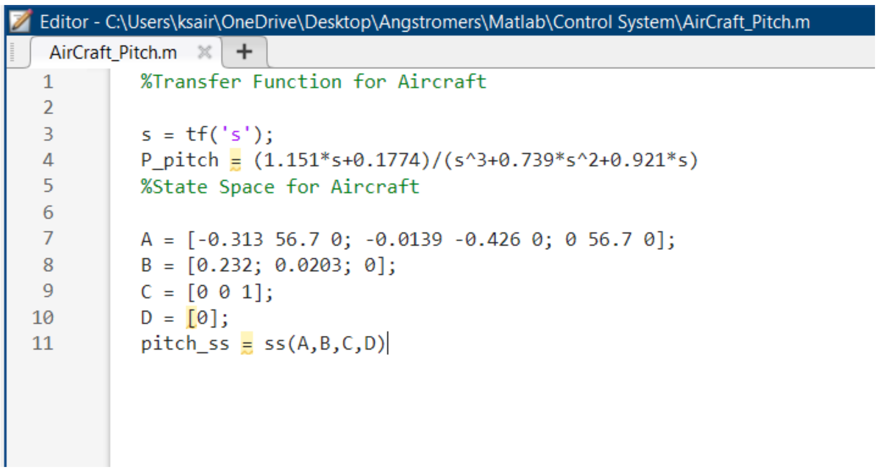
$$sQ(s) = -0.0139A(s) - 0.426Q(s) + 0.0203\Delta(s)$$

$$s\Theta(s) = 56.7Q(s)$$

After a few steps of algebra, the transfer function relating the pitch angle Θ(s) to the elevator deflection angle Δ(s) is:

$$P(s) = \frac{\Theta(s)}{\Delta(s)} = \frac{1.151s + 0.1774}{s^3 + 0.739s^2 + 0.921s}$$

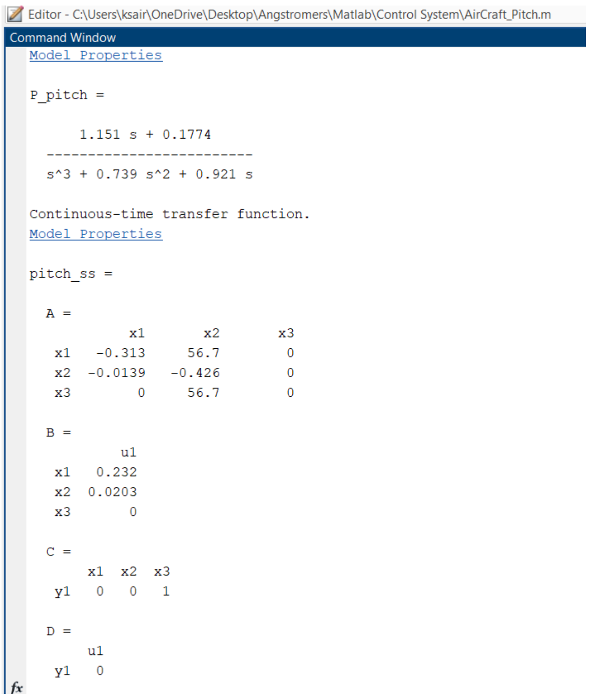
7.3 Implementation



Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\AirCraft_Pitch.m
AirCraft_Pitch.m

```
%Transfer Function for Aircraft

s = tf('s');
P_pitch = (1.151*s+0.1774)/(s^3+0.739*s^2+0.921*s)
%State Space for Aircraft

A = [-0.313 56.7 0; -0.0139 -0.426 0; 0 56.7 0];
B = [0.232; 0.0203; 0];
C = [0 0 1];
D = [0];
pitch_ss = ss(A,B,C,D)
```

Output:



Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\AirCraft_Pitch.m
Command Window

```
Model Properties

P_pitch =

      1.151 s + 0.1774
  -------------------------
  s^3 + 0.739 s^2 + 0.921 s

Continuous-time transfer function.
Model Properties

pitch_ss =

  A = 
            x1       x2       x3
   x1   -0.313     56.7        0
   x2  -0.0139   -0.426        0
   x3        0     56.7        0

  B = 
           u1
   x1   0.232
   x2  0.0203
   x3       0

  C = 
       x1  x2  x3
   y1   0   0   1

  D = 
       u1
   y1   0
```

7.4 Use Case of AirCraft Pitch

Landing Approach Optimization

The use case of aircraft pitch control is during the landing approach phase of a flight. Ensuring a smooth and safe landing requires precise control of the aircraft's pitch to manage the descent angle and rate. During the final approach, the autopilot system or the pilot manually adjusts the pitch to control the glide path angle. This is crucial for aligning with the runway correctly and maintaining a steady descent. The control system continuously monitors the aircraft's pitch angle, altitude, speed, and other parameters to make real-time adjustments, ensuring the aircraft descends at the optimal angle and speed for a safe landing.

In this scenario, a Model Predictive Control (MPC) approach can be employed for the pitch control system. MPC uses a model of the aircraft's dynamics to predict future states and optimize control actions over a finite time horizon. During landing, the MPC algorithm processes real-time data from onboard sensors and external sources like Instrument Landing System (ILS) signals. It then calculates the optimal pitch adjustments needed to follow the desired glide path. The outcome is a more precise and stable descent, minimizing the risk of hard landings or runway overshoot. By optimizing the landing approach through advanced pitch control, the system enhances landing safety and efficiency, contributing to smoother and more reliable operations in various weather conditions and at different airports.

Section 8
Ball & Beam

8.1 Introduction

The ball and beam system is a classic example used to demonstrate and test control theory concepts in engineering. In this system, a ball is allowed to roll along a beam whose angle can be adjusted by a lever connected to a servo motor. The primary objective is to control the position of the ball along the beam by varying the angle of the beam. This setup represents a dynamic system with one degree of freedom, where the ball's position is influenced by gravitational forces and the beam's inclination.

The challenge lies in designing a control system that accurately manipulates the ball's position by adjusting the beam's angle through the servo. The system requires careful consideration of the ball's rolling dynamics, which are governed by its mass, radius, and moment of inertia, as well as the beam's length and the gravitational acceleration. By leveraging control strategies such as PID control or more advanced techniques, engineers can develop controllers that maintain precise control over the ball's position, making the ball and beam system an insightful platform for studying control system design and dynamics.

8.2 System Equation

The system equation for an inverted pendulum on a cart is:

Given the mass and dynamics of the cart and pendulum, the governing differential equations are:

For the cart:

$$(M+m)\ddot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta = u$$

Where,

M is the mass of the cart
m is the mass of the pendulum
l is the length of the pendulum
x is the position of the cart
θ is the angle of the pendulum from the vertical
u is the control input force applied to the cart

State Space Model:
The state-space model provides a mathematical framework to describe the dynamics of a system using a set of first-order differential equations. For the ball and beam system, the state-space representation captures the relationship between the input, the state, and the output of the system.

State-Space Equations

For the ball and beam system, the state variables are chosen as the ball's position r, ball's velocity r˙, beam angle α, and beam angular velocity α˙. The input to the system is the torque applied to the beam, denoted by u. The state-space representation is given by:

$$\begin{bmatrix} \dot{r} \\ \ddot{r} \\ \dot{\alpha} \\ \ddot{\alpha} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{-mg}{\left(\frac{J}{R^2}+m\right)} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} r \\ \dot{r} \\ \alpha \\ \dot{\alpha} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} r \\ \dot{r} \\ \alpha \\ \dot{\alpha} \end{bmatrix}$$

Transfer Function Model:
The transfer function P(s) is derived from the governing differential equation, capturing how the ball position R(s) responds to the input torque U(s).

Starting from the linearized system equation:

$$\left(\frac{J}{R^2}+m\right)\ddot{r} = -mg\alpha$$

and using the relationship α=(d/L)* θ , the Laplace transform of the equation is:

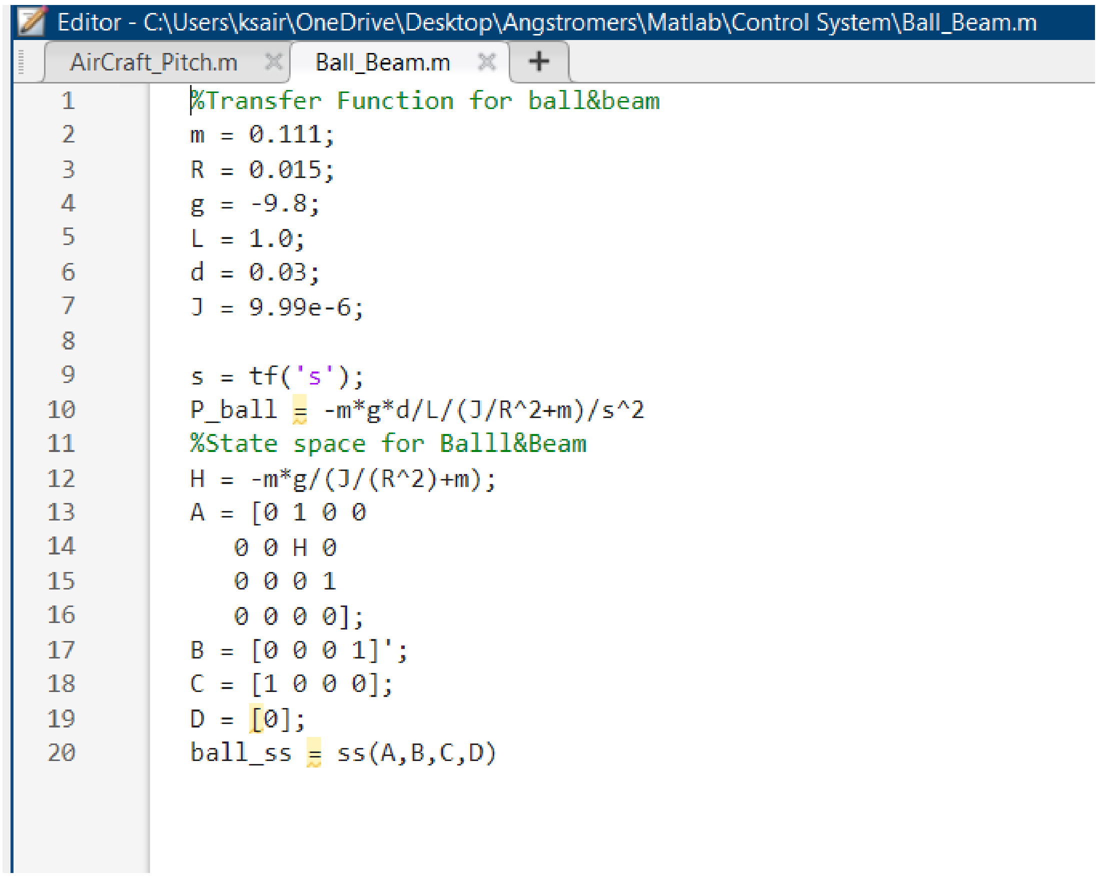
$$\left(\frac{J}{R^2}+m\right)R(s)s^2 = -mg\frac{d}{L}\Theta(s)$$

Rearranging to find the transfer function from the gear angle Θ(s) to the ball position R(s):

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

This transfer function characterizes the relationship between the input torque applied to the beam and the resulting ball position in the frequency domain.

8.3 Implementation

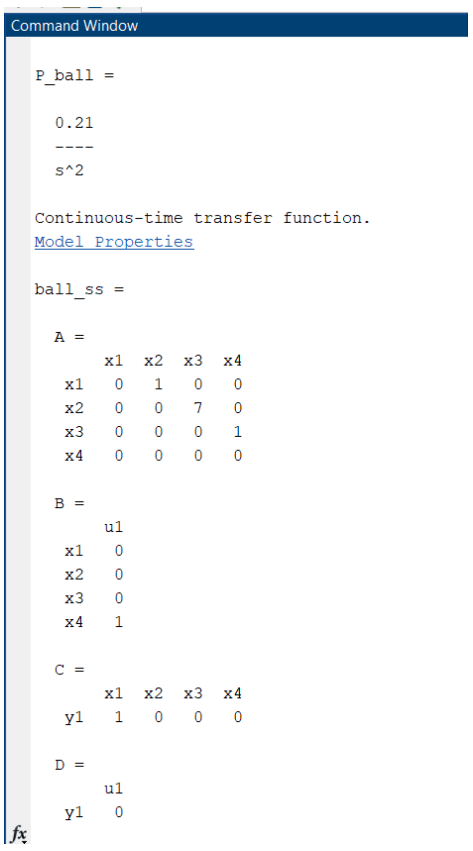


Editor - C:\Users\ksair\OneDrive\Desktop\Angstromers\Matlab\Control System\Ball_Beam.m

AirCraft_Pitch.m | Ball_Beam.m

```
%Transfer Function for ball&beam
m = 0.111;
R = 0.015;
g = -9.8;
L = 1.0;
d = 0.03;
J = 9.99e-6;

s = tf('s');
P_ball = -m*g*d/L/(J/R^2+m)/s^2
%State space for Balll&Beam
H = -m*g/(J/(R^2)+m);
A = [0 1 0 0
   0 0 H 0
   0 0 0 1
   0 0 0 0];
B = [0 0 0 1]';
C = [1 0 0 0];
D = [0];
ball_ss = ss(A,B,C,D)
```

Output:



```
P_ball =

  0.21
  ----
  s^2

Continuous-time transfer function.
Model Properties

ball_ss =

  A =
       x1  x2  x3  x4
   x1   0   1   0   0
   x2   0   0   7   0
   x3   0   0   0   1
   x4   0   0   0   0

  B =
       u1
   x1   0
   x2   0
   x3   0
   x4   1

  C =
       x1  x2  x3  x4
   y1   1   0   0   0

  D =
       u1
   y1   0
```

8.4 Use case of Ball & Beam

Advanced Robotic Manipulation

The Ball & Beam system is also employed in advanced robotic manipulation research, where precise control of movement and balance is crucial. In this context, the system serves as a simplified model for developing and testing control algorithms used in robotic arms and other articulated structures. By controlling the position of the ball on the beam, researchers can refine techniques for trajectory planning, dynamic balancing, and adaptive control, which are essential for robots operating in unstructured environments or performing delicate tasks.

Implementation and Outcome

To implement the Ball & Beam system in this context, a robotic arm is equipped with sensors to measure the ball's position and an actuator to adjust the beam's angle. A state-space model of the system is developed, and an advanced control algorithm, such as Model Predictive Control (MPC), is designed to predict and optimize the arm's movements to maintain the ball's balance. The outcome of this implementation is a robust control system capable of handling disturbances and uncertainties, demonstrating enhanced stability and precision. These advancements can be translated to real-world applications, such as robotic surgery or industrial automation, where precision and adaptability are paramount. The Ball & Beam system thus provides a valuable testbed for refining control strategies before deploying them in more complex scenarios.

Section 9
Conclusion

In the realm of control systems, dynamic systems such as cruise control, motor speed and position control, suspension systems, inverted pendulums, Ball & Beam setups, and aircraft pitch control are fundamental examples that demonstrate the principles of state flow and transfer functions. These systems provide a practical framework for understanding how various control strategies can be applied to achieve stability, precision, and desired performance in real-world applications. By analyzing the state-space representation and transfer functions of these systems, we can derive insights into their behavior and design appropriate controllers.

Through the study of these dynamic systems, we have learned several key concepts:

1. **State-Space Representation:** This approach provides a comprehensive way to model and analyze systems with multiple inputs and outputs. It allows for a detailed understanding of system dynamics, making it easier to design controllers that can handle complex interactions and achieve desired performance.
2. **Transfer Functions:** These mathematical representations describe the input-output relationship of a system in the frequency domain. They are particularly useful for analyzing the stability and response characteristics of linear time-invariant systems, helping in the design of filters and controllers.
3. **Control Strategies:** By applying various control techniques such as PID control, state feedback, and Model Predictive Control (MPC), we can achieve different performance objectives. For instance, maintaining a constant speed in cruise control, achieving precise motor speed and position control, ensuring comfort and stability in suspension

systems, balancing the inverted pendulum and Ball & Beam, and maintaining the desired pitch in aircraft.

4. **Practical Applications:** The study of these systems is not just theoretical; it has practical implications in engineering fields. For example, the principles learned from the Ball & Beam system can be applied to robotic manipulation, while concepts from motor control are essential in industrial automation.

Overall, the exploration of these dynamic systems has enhanced our understanding of control theory and its applications. It has demonstrated the importance of both state-space and transfer function approaches in designing effective controllers for a wide range of engineering problems, thereby bridging the gap between theory and practical implementation.