Project Report

IE-415 Control of Autonomous

Name:- Dave Preet A.

Student - id: - 202201072

Project Name:- Control of Wheeled Inverted Pendulum

Under the Guidance of:- Prof. Sujay Kadam

Abstract:

The project focuses on the modelling, control, and simulation of a Wheeled Inverted Pendulum system using MATLAB and Simulink. The system 's nonlinear dynamics are stabilized using state-space control techniques, specifically Linear Quadratic Regulator (LQR) and state feedback control. Additionally, a state observer is designed to estimate unmeasured states, enhancing system robustness. The simulation results validate the effectiveness of the proposed controllers in maintaining the pendulum's upright position and minimizing the cart's displacement. This project demonstrates the application of core concepts from system dynamics, control theory, and state estimation.

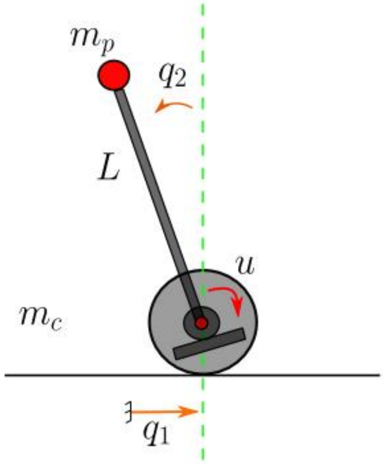
Introduction:

Among the various systems studied in control theory, the Wheeled Inverted Pendulum, also known as the cart-pendulum system, stands out as a classic benchmark problem. This system is inherently nonlinear and unstable, which makes it an ideal testbed for exploring various control strategies. Its dynamics resemble those of many real-world applications, such as self-balancing robots, autonomous vehicles, and mobile robotic platforms.

In this project, we aim to model, simulate, and control a Wheeled Inverted Pendulum system using MATLAB and Simulink. The focus will be on developing a state-space model of the system, designing controllers using techniques like State Feedback Control and Linear Quadratic Regulator (LQR), and implementing a state observer for estimating unmeasured states. These methods are essential for achieving robust performance and stability, especially in systems where direct measurement of all states is not feasible.

System Modelling:

1. Dynamic Equation of the system:



The Wheeled Inverted Pendulum system consists of a pendulum attached to a wheeled cart. The cart can move along a horizontal track, while the pendulum is free to swing in the vertical plane. The primary objective is to apply a control force to the cart so that the pendulum remains upright while maintaining the desired position of the cart.

Parameters of the System:

- m_c : Mass of the cart (kg)
- m_p : Mass of the pendulum (kg)
- L : Length of the pendulum (distance from the pivot to the center of mass) (m)
- G : Acceleration due to gravity (m/s²)
- u : Control force applied to the cart (N)
- q_1 : Horizontal displacement of the cart (m)
- q_2 : Angle of the pendulum from the vertical (rad)
- d_1, d_2: Damping coefficients for the cart and pendulum, respectively (N·s/m, N·s·rad)

To derive the equations of motion, we use Newton-Euler mechanics or the Lagrangian approach. Here, we'll outline the process using the Lagrangian method, which involves calculating the kinetic and potential energy of the system

Kinetic Energy (T):

$$T = \frac{1}{2}m_c\dot{q}_1^2 + \frac{1}{2}m_p\left(\dot{q}_1^2 + L^2\dot{q}_2^2 + 2Lq_1\dot{q}_2\cos(q_2)\right)$$

Potential Energy (V):

$$V = m_p gL\cos(q_2)$$

Lagrangian ($\mathcal{L}$):

$$\mathcal{L} = T - V$$

By applying the Euler-Lagrange equation:

$$\frac{d}{dt}\left(\frac{\partial\mathcal{L}}{\partial\dot{q}_i}\right) - \frac{\partial\mathcal{L}}{\partial q_i} = Q_i$$

where Q_i represents the generalized forces, we can derive the nonlinear equations of motion for the system.

2. State-Space Representation:

To control the Wheeled Inverted Pendulum, it is essential to linearize the nonlinear equations of motion around the upright equilibrium point (i.e., $q_2 = 0, \dot{q}_2 = 0$). The linearized state-space representation allows us to apply linear control techniques like State Feedback and LQR.

Define the state vector as:

$$x = [\mathrm{q}_1, \mathrm{q}_2, \dot{q}_1, \dot{q}_2]^T$$

The input to the system is the force applied to the cart (u), and the output can be either the cart position (q_1) or the pendulum angle (q_2)

The linearized state-space equations take the form:

$$\dot{x} = Ax + Bu$$

$$y = C\,x + D\,u$$

Where:

- A is the system matrix.
- B is the input matrix.
- C is the output matrix.
- D is the feedforward matrix (often zero for many systems).

3. Derivation of the Matrices:

Using the parameters of the system, the matrices for the state-space representation are derived as follows:

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \dfrac{gm_c}{m_p} & -\dfrac{d_1}{m_c} & \dfrac{d_2}{Lm_c} \\ 0 & -\dfrac{g(m_c+m_p)}{Lm_c} & -\dfrac{d_1}{Lm_c} & -\dfrac{d_2(m_c+m_p)}{L^2 m_c m_p} \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{m_c} \\ \frac{1}{Lm_c} \end{bmatrix}$$

For C matrix, If we want to choose q_2 as the factor then,

$$C = [0 \quad 1 \quad 0 \quad 0\,]$$

For q_1, $C = [1 \quad 0 \quad 0 \quad 0]$

$$D = [0]$$

Dynamic Model of the System:

The equations of motion for the described system can be derived using different methods such as Lagrange-Formalism or Newton-Euler. The result is in any case a nonlinear system in the following form

$$M(q)\ddot{q} + h(q,\dot{q}) = g_q u$$

With

$$M(q) = \begin{pmatrix} m_e + m_p & -Lm_p \cos q_2 \\ -Lm_p \cos q_2 & L^2 m_p \end{pmatrix}$$

$$h(q,\dot{q}) = \begin{pmatrix} Lm_p \dot{q}_2^2 \sin q_2 \\ -L\, g\, m_p \sin q_2 \end{pmatrix} + \begin{pmatrix} d_1\, \dot{q}_1 \\ d_2\, \dot{q}_2 \end{pmatrix}, \qquad g_q = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

where d_1 and d_2 are the so-called damping factors representing friction in the cart displacement and the joint respectively.

A state-space representation can be derived using the equations above defining $x = \binom{q}{\dot{q}}$.

$$\dot{x} = F(x) + G(x)u = \begin{pmatrix} \dot{q_1} \\ \dot{q_2} \\ -M^{-1}(q)\, h(q,\dot{q}) \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ M^{-1}(q)\, g_q \end{pmatrix} u$$

Matlab Simulation of Results:

```
clc
clear
clear all
% Define system parameters
mc = 1.5;  % mass of the cart (kg)
mp = 0.5;  % mass of the pendulum (kg)
L = 1;     % length of the pendulum (m)
g = 9.82;  % gravity (m/s^2)
d1 = 0.01; % damping coefficient for the cart
d2 = 0.01; % damping coefficient for the pendulum
```

Define the matrices

```
A = [0,0,1,0;
    0,0,0,1;
    0, (g*mp)/mc,-d1/mc,-d2/(L*mc);
    0, (g*(mc+mp))/(L*mc), -d1/(L*mc), -(d2*mc + d2*mp)/(L^2*mp*mc)];

B=[0 ; 0 ; 1/mc ; 1/(L*mc)];
% C = [0;1;0;0]; % q_2 as the output
C = [1;0;0;0]; % q_1 as the output
D =0;
```

Built the system

```
sys = ss(A,B,C',D)
```

```
sys =


  A =

           x1       x2       x3       x4

   x1       0        0        1        0
```

```
  x2          0          0          0          1
  x3          0      3.273  -0.006667  -0.006667
  x4          0      13.09  -0.006667   -0.02667


 B =

         u1
  x1      0
  x2      0
  x3 0.6667
  x4 0.6667


 C =

      x1  x2  x3  x4
  y1   1   0   0   0


 D =

      u1
  y1   0
```

Here We can see that our poles are at the Positive side so, system is likely to be the unstable.

```
eig(A) % or the pole of the system
```

```
ans = 4×1

         0
   -3.6327
    3.6043
   -0.0050
```

```
Sc= ctrb(sys)
```

```
Sc = 4×4

         0    0.6667   -0.0089    2.1824
         0    0.6667   -0.0222    8.7295
    0.6667   -0.0089    2.1824   -0.1455
    0.6667   -0.0222    8.7295   -0.5383
```

```
rank(Sc)
```

ans = 4

For the q1 system is controllable ,here we can see that it is full rank matrix.

Same for the q2 system is also controllable.

```
So = obsv(sys)
```

So = *4×4*

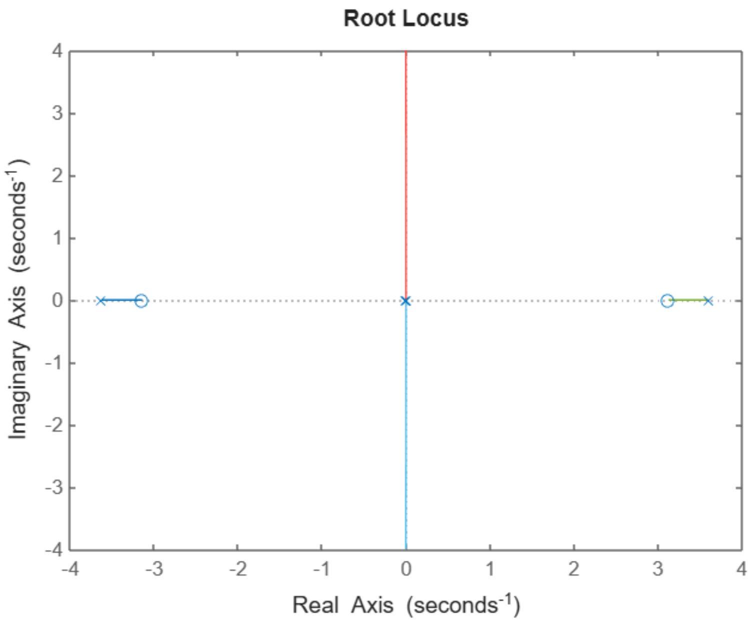
```
    1.0000         0         0         0
         0         0    1.0000         0
         0    3.2733   -0.0067   -0.0067
         0   -0.1091    0.0001    3.2736
```

For the q1 system is seem to be observable , but for the q2 system is not observable.

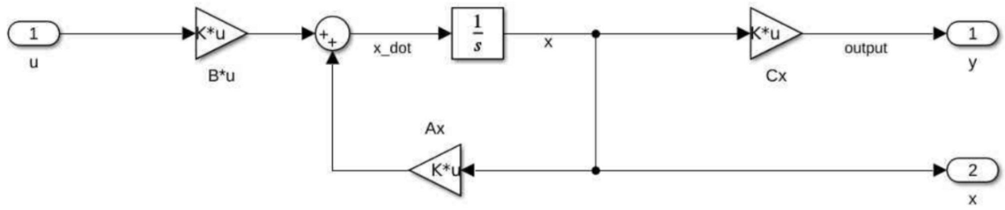
```
rank(So)
```

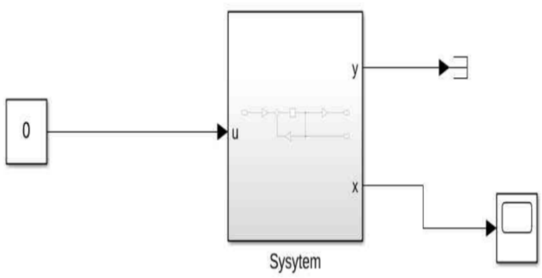
ans = 4

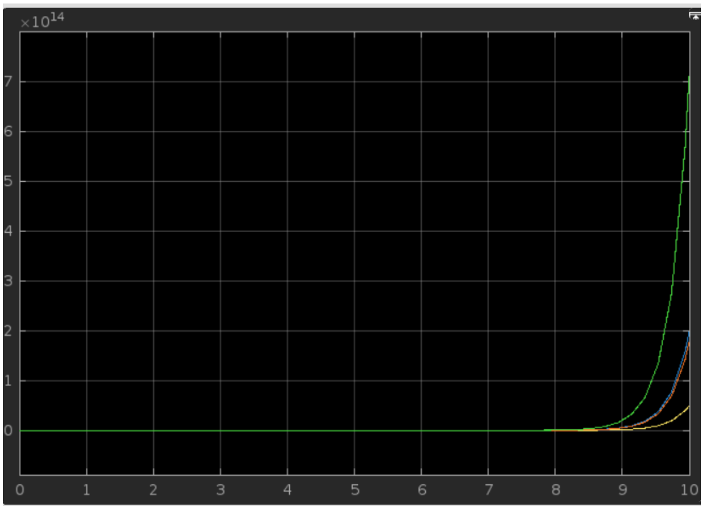
```
rlocus(sys);
```



For this we can given that .slx file for the







Here we can see that output of the scope is the unbounded, means system is unstable.

So, we have to do the feedback signals

And we will use the ackerman formula for it.

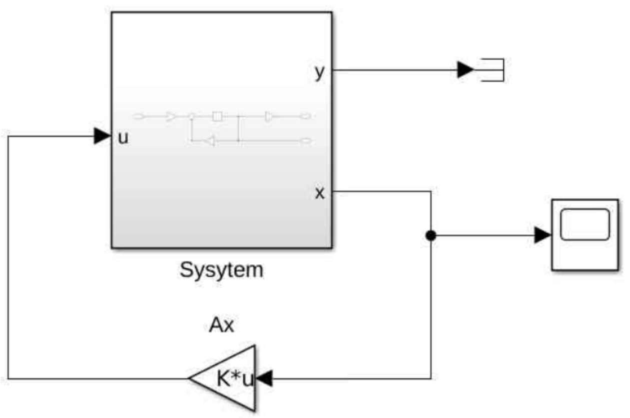
Initial Condition

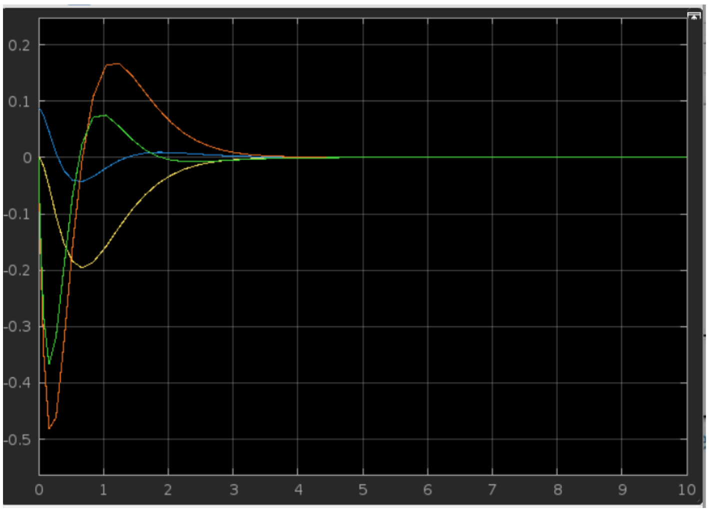
```
x0=[0; 5*pi/180 ; 0 ; 0];
```

Build Controllor

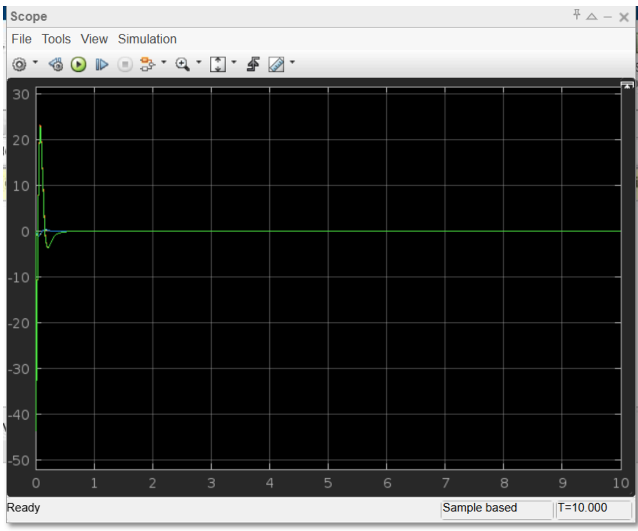
```
des_pole=[-3;-3;-3;-3];
K = acker(A,B,des_pole)
%
```

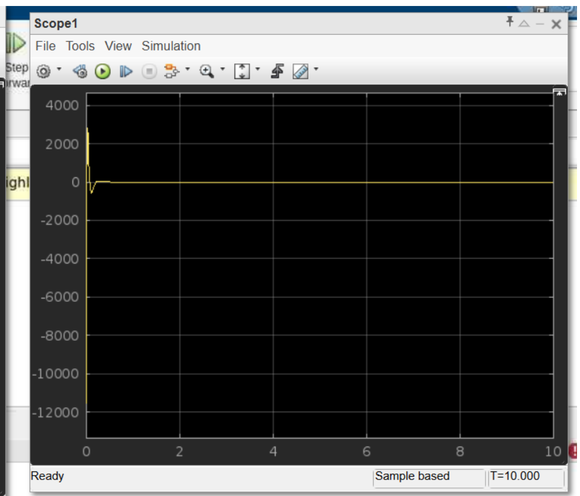
```
K = 1×4

  -12.3727   113.3432   -16.5321    34.4821
```





Note that whenever we increase the value of the des_poles , graph getting faster to the steady state .**But there is trade-off between input and output.** Because If we increase the stability our input feedback goes up.





Here we can see that input is goes upto 4000. When we do just 10 times des_poles.

Now for the more suitability, we will also use LQR for it.

```
Q= eye(4);
R=0.1;
K_lqr=lqr(A,B,Q,R)
```

K_lqr = *1×4*

-3.1623   72.7343   -6.6580   21.7182

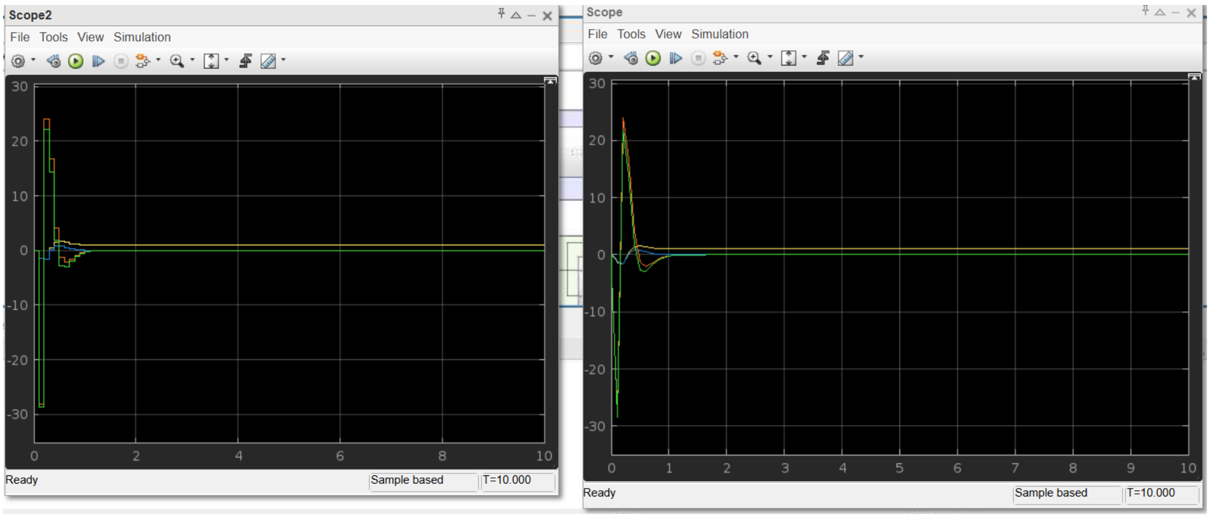
In real life, basically due to digital signal we have to discretise the signals so we must have to use the discrete time signal for it.

For discretising we will use Zero-order Hold.

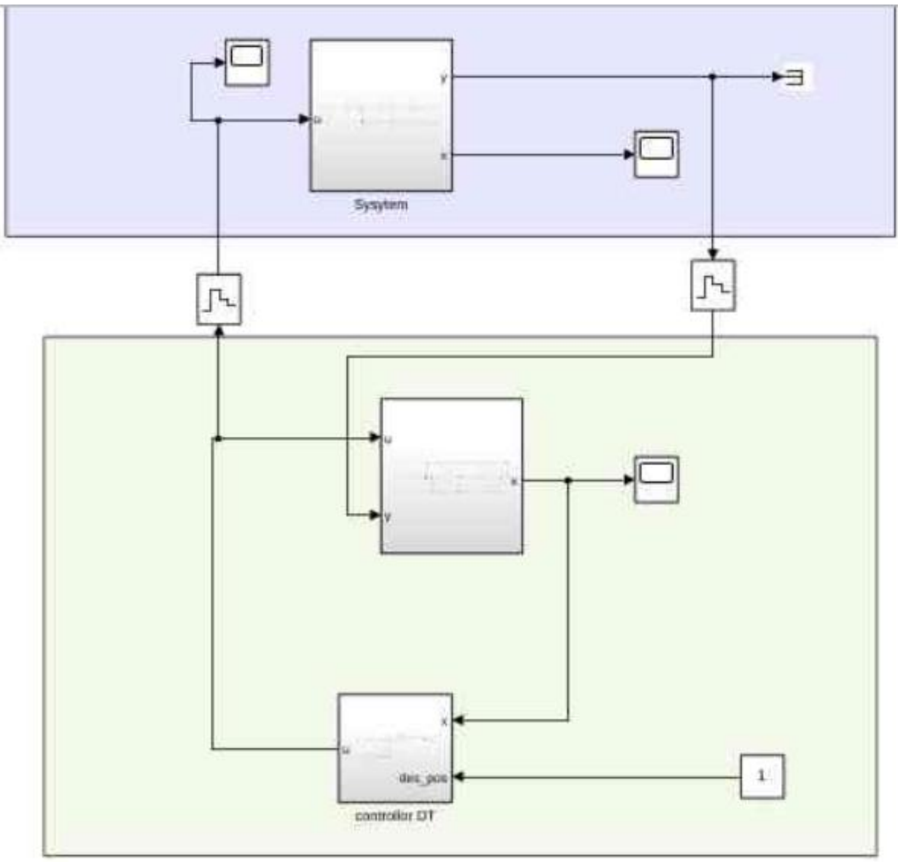
One final thing we introduce that is observer, because it is good that we have somehow reconstruct the state and which give us the optimal output.

But we need states ,but we only have the output of the system using the sensors so, state observer is simply reproduction of the system, we simply use the system with discrete time system.

So observer simply reconstruct the state of the inputs and then gives it to the controller



Here we can see that first picture is output of the state observer which is reconstruct the state and second is real states.



This is the final version of my system and now we will do the system and second gray system has observer and continuous_DT system.

Now we replace the system with the dynamic system which is uploaded in the github