FINAL PROJECT

ENPM 667

Prathinav Karnala Venkata, Sarang Shibu
UID : 120380983 , 120254307
DID : pratkv , sarang

December 18, 2023

Contents

1 Introduction **3**

2 Equations of Motion **3**

3 Simulation **7**

3.1 Linearization and Controllability 7

3.2 LQR Controller 10

3.3 Observability 17

3.4 Luenberger Observer 20

3.5 Functions used for solving the Differential Equations for the Observer of Non-linear Systems 31

3.6 LQG controller 37

3.7 Function to implement LQG controller for Non linear system: 42

4 Assumptions **44**

5 Conclusion **46**

1 Introduction

The thorough analysis and management of a crane system supporting two suspended loads is the main focus of this project. The first step is to create the dynamic equations that control how the loads behave and how the crane moves. The system's intrinsic non-linearities are then resolved by linearizing it around a chosen equilibrium point, which improves the system's tractability for control design. After that, the main emphasis switches to using sophisticated control techniques, such as the Linear Quadratic Gaussian (LQG) controller, Luenberger observer, and Linear Quadratic Regulator (LQR) controller. To make sure the control strategies used are viable, the system's observability and controllability are closely scrutinized. A non-linear state-space representation is built, and the dynamic equations are expressed numerically. The entire procedure is carried out with MATLAB.The following figure displays the specified crane and suspended loads:

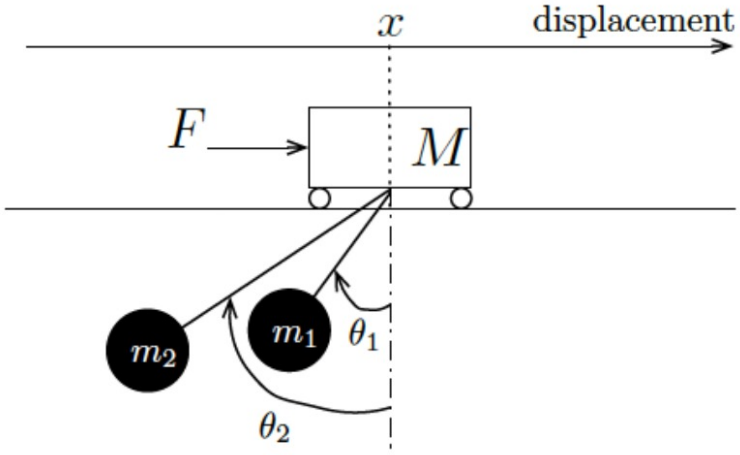

Figure 1: Crane-Load system.

2 Equations of Motion

Assume the following: the crane is oriented along the x-axis, two loads with masses of $m1$ and $m2$ and cable lengths of $l1$ and $l2$ are suspended underneath it, and the crane is subject to a force of F. The following defines the parameters that are utilized to derive dynamic equations:

M : mass of crane

m_1 : mass of load 1

m_2 : mass of load 2

l_1 : length of cable 1

l_2 : length of cable 2

θ_1 : angle made by the first load

θ_2 : angle made by the second load

F : external force

g : acceleration due to gravity

Position for mass m_1 w.r.t θ_1:

$$\mathbf{x}_{m1} = (x - l_1 \sin(\theta_1))\hat{i} + (-l_1 \cos(\theta_1))\hat{j} \tag{1}$$

Differentiating the position of m_1 w.r.t time we obtain the Velocity equation:

$$\mathbf{v}_{m1} = \dot{x} - l_1\dot{\theta}_1 \cos(\theta_1)\hat{i} + (l_1\dot{\theta}_1 \sin(\theta_1))\hat{j} \tag{2}$$

Position for mass m_2 w.r.t θ_2:

$$\mathbf{x}_{m2} = (x - l_2 \sin(\theta_2))\hat{i} + (-l_2 \cos(\theta_2))\hat{j} \tag{3}$$

Differentiating the position of m_2 w.r.t time we obtain the Velocity equation:

$$\mathbf{v}_{m2} = \dot{x} - l_2\dot{\theta}_2 \cos(\theta_2)\hat{i} + (l_2\dot{\theta}_2 \sin(\theta_2))\hat{j} \tag{4}$$

Kinetic energy of the system:

$$\begin{aligned} K = \frac{1}{2}M\dot{x}^2 + \frac{1}{2}m_1\left(\dot{x} - l_1\dot{\theta}_1\cos(\theta_1)\right)^2 + \frac{1}{2}m_1 l_1^2\dot{\theta}_1^2 \\ + \frac{1}{2}m_2\left(\dot{x} - l_2\dot{\theta}_2\cos(\theta_2)\right)^2 + \frac{1}{2}m_2 l_2^2\dot{\theta}_2^2 \end{aligned} \tag{5}$$

The potential energy of the system :

$$P = -m_1 g l_1 \cos(\theta_1) - m_2 g l_2 \cos(\theta_2) = -g[m_1 l_1 \cos(\theta_1) + m_2 l_2 \cos(\theta_2)] \tag{6}$$

Lagrange equation is the difference between Kinetic and Potential energies:

$$\begin{gathered} L = K - P \\ L = \frac{1}{2}M\dot{x}^2 + \frac{1}{2}(m_1 + m_2)\dot{x}^2 + \frac{1}{2}m_1 l_1^2\dot{\theta}_1^2 + \frac{1}{2}m_2 l_2^2\dot{\theta}_2^2 \\ -\dot{x}(m_1 l_1\dot{\theta}_1 \cos(\theta_1) + m_2 l_2\dot{\theta}_2 \cos(\theta_2)) + g[m_1 l_1 \cos(\theta_1) + m_2 l_2 \cos(\theta_2)] \end{gathered} \tag{7}$$

Further simplification:

$$L = \frac{1}{2}M\dot{x}^2+\frac{1}{2}(m_1+m_2)\dot{x}^2+\frac{1}{2}m_1l_1^2\dot{\theta}_1^2+\frac{1}{2}m_2l_2^2\dot{\theta}_2^2-\dot{x}(m_1l_1\dot{\theta}_1\cos(\theta_1)+m_2l_2\dot{\theta}_2\cos(\theta_2))+g[m_1l_1\cos(\theta_1)+m_2l_2\cos(\theta_2)] \tag{8}$$

Using Lagrangian mechanics we can write:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) - \frac{\partial L}{\partial x} = F \tag{9}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_1}\right) - \frac{\partial L}{\partial \theta_1} = 0 \tag{10}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_2}\right) - \frac{\partial L}{\partial \theta_2} = 0 \tag{11}$$

Examining (9):

$$\frac{\partial L}{\partial \dot{x}} = M\dot{x} + (m_1+m_2)\ddot{x} - m_1l_1\dot{\theta}_1\cos(\theta_1) - m_2l_2\dot{\theta}_2\cos(\theta_2) \tag{12}$$

$$\begin{aligned}\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) = M\ddot{x} + (m_1+m_2)\ddot{x} - \Big[m_1l_1\ddot{\theta}_1\cos(\theta_1) - m_1l_1\dot{\theta}_1^2\sin(\theta_1)\Big] \\ - \Big[m_2l_2\ddot{\theta}_2\cos(\theta_2) - m_2l_2\dot{\theta}_2^2\sin(\theta_2)\Big]\end{aligned} \tag{13}$$

Since there is no x term:

$$\frac{\partial L}{\partial x} = 0 \tag{14}$$

The first Lagrangian equation can be written as:

$$[M+m_1+m_2]\ddot{x}-m_1l_1\ddot{\theta}_1\cos(\theta_1)+m_1l_1\dot{\theta}_1^2\sin(\theta_1)-m_2l_2\ddot{\theta}_2\cos(\theta_2)+m_2l_2\dot{\theta}_2^2\sin(\theta_2) = F \tag{15}$$

Examining (10):

$$\frac{\partial L}{\partial \dot{\theta}_1} = m_1l_1^2\dot{\theta}_1 - m_1x\dot{\theta}_1\cos(\theta_1) \tag{16}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_1}\right) = m_1l_1^2\ddot{\theta}_1 - m_1l_1\ddot{x}\cos(\theta_1) - m_1xl_1\dot{\theta}_1^2\sin(\theta_1) \tag{17}$$

$$\frac{\partial L}{\partial \theta_1} = m_1l_1\dot{\theta}_1x\dot{x}\sin(\theta_1) - m_1l_1g\sin(\theta_1) \tag{18}$$

Substituting the above equations in (10) we obtain our second lagrangian equation:

$$m_1l_1^2\ddot{\theta}_1 - m_1l_1\ddot{x}\cos(\theta_1) - m_1xl_1\dot{\theta}_1^2\sin(\theta_1) - m_1xl_1\dot{\theta}_1^2\sin(\theta_1) + m_1l_1g\sin(\theta_1) = 0 \tag{19}$$

Further simplifying we get:

$$m_1l_1^2\ddot{\theta}_1 - m_1x\ddot{l}_1\cos(\theta_1) + m_1l_1g\sin(\theta_1) = 0 \tag{20}$$

Examining (11):

$$\frac{\partial L}{\partial \dot{\theta}_2} = m_2 l_2^2 \dot{\theta}_2 - m_2 x l_2 \dot{\theta}_2 \cos(\theta_2) \tag{21}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_2}\right) = m_2 l_2^2 \ddot{\theta}_2 - m_2 x l_2 \ddot{\theta}_2 \cos(\theta_2) - m_2 \dot{\theta}_2 x l_2 \dot{\theta}_2 \sin(\theta_2) \tag{22}$$

$$\frac{\partial L}{\partial \theta_2} = m_2 x l_2 \dot{\theta}_2 \dot{\theta}_2 \sin(\theta_2) - m_2 l_2 g \sin(\theta_2) \tag{23}$$

Substituting the above equations in (11) we obtain our third lagrangian equation:

$$m_2 l_2^2 \ddot{\theta}_2 - m_2 l_2 \ddot{x} \cos(\theta_2) + m_2 \dot{\theta}_2 x l_2 \dot{\theta}_2 \sin(\theta_2) - m_2 \dot{\theta}_2 x l_2 \dot{\theta}_2 \sin(\theta_2) + m_2 g l_2 \sin(\theta_2) \tag{24}$$

Further simplifying we get:

$$m_2 l_2^2 \ddot{\theta}_2 - m_2 l_2 \ddot{x} \cos(\theta_2) + m_2 g l_2 \sin(\theta_2) = 0 \tag{25}$$

The non-linear state space representation can be written using the following state variables x, $\dot{x}$, θ_1, $\dot{\theta}_1$, θ_2, $\dot{\theta}_2$.

$$X = \begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta}_1 \\ \theta_2 \\ \dot{\theta}_2 \end{bmatrix} \tag{26}$$

$$\dot{X} = \begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta}_1 \\ \ddot{\theta}_1 \\ \dot{\theta}_2 \\ \ddot{\theta}_2 \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \frac{-m_1 g \sin\theta_1 \cos\theta_2 - m_2 g \sin\theta_2 \cos\theta_2 - m_1 l_1 \dot{\theta}_1^2 \sin\theta_1 - m_2 l_2 \dot{\theta}_2^2 \sin\theta_2 + F}{M + m_1 + m_2 - m_1 \cos^2\theta_1 - m_2 \cos^2\theta_2} \\ \dot{\theta}_1 \\ \frac{-m_1 g \sin\theta_1 \cos\theta_2 - m_2 g \sin\theta_2 \cos\theta_2 - \frac{m_1 l_1 \dot{\theta}_1^2}{\sin\theta_1} - \frac{m_2 l_2 \dot{\theta}_2^2}{\sin\theta_2} + F}{(M + m_1 + m_2 - m_1 \cos^2\theta_1 - m_2 \cos^2\theta_2) l_1} - \frac{g \sin\theta_1}{l_1} \\ \dot{\theta}_2 \\ -\frac{m_1 g \sin\theta_1 \cos\theta_2 - m_2 g \sin\theta_2 \cos\theta_2 - m_1 l_1 \dot{\theta}_1^2 \sin\theta_1 - m_2 l_2 \dot{\theta}_2^2 \sin\theta_2 + F}{(M + m_1 + m_2 - m_1 \cos^2\theta_1 - m_2 \cos^2\theta_2) l_1} - \frac{g \sin\theta_2}{l_2} \end{bmatrix} \tag{27}$$

3 Simulation

These are the video links to execute the files:

- https://drive.google.com/file/d/1piEeFAcDY1x7stZfrM5wJp22VEXxCv7u/view?usp=drive_link
- https://drive.google.com/file/d/1bqpTn4DnDVs5c9zL9_2jD5czvwIqpGlt/view?usp=drive_link

3.1 Linearization and Controllability

In the problem statement we have to linearize about the equilibrium point $x = 0$, $\theta_1 = 0$, and $\theta_2 = 0$. This implies that:

$$x = 0 \tag{28}$$
$$\theta_1 = 0 \tag{29}$$
$$\theta_2 = 0 \tag{30}$$
$$\dot{x} = 0 \tag{31}$$
$$\dot{\theta_1} = 0 \tag{32}$$
$$\dot{\theta_2} = 0 \tag{33}$$

Here we use Jacobian Linearization and below is the code to compute it:

```
%Jacobian Linearization
clc;
close all;
%State variables and other system parameters
syms x x_dot theta1 theta1_dot theta2 theta2_dot F M m1 m2 g l1 l2

% state variables x_dot, theta1_dot, and theta2_dot
x_ddot = (F-(m1*sin(theta1))*(g*cos(theta1)+l1*theta1_dot*theta1_dot)-
    m2*sin(theta2)*(g*cos(theta2)+l2*theta2_dot*theta2_dot))/(M+m1*sin
    (theta1)*sin(theta1)+m2*sin(theta2)*sin(theta2));
theta1_ddot = (x_ddot*cos(theta1)-g*sin(theta1))/l1;
theta2_ddot = (x_ddot*cos(theta2)-g*sin(theta2))/l2;

J = [diff(x_dot,x) diff(x_dot,x_dot) diff(x_dot,theta1) diff(x_dot,
    theta1_dot) diff(x_dot,theta2) diff(x_dot,theta2_dot);
    diff(x_ddot,x) diff(x_ddot,x_dot) diff(x_ddot,theta1) diff(x_ddot,
        theta1_dot) diff(x_ddot,theta2) diff(x_ddot,theta2_dot);
    diff(theta1_dot,x) diff(theta1_dot,x_dot) diff(theta1_dot,theta1)
        diff(theta1_dot,theta1_dot) diff(theta1_dot,theta2) diff(
        theta1_dot,theta2_dot);
    diff(theta1_ddot,x) diff(theta1_ddot,x_dot) diff(theta1_ddot,
        theta1) diff(theta1_ddot,theta1_dot) diff(theta1_ddot,theta2)
        diff(theta1_ddot,theta2_dot);
    diff(theta2_dot,x) diff(theta2_dot,x_dot) diff(theta2_dot,theta1)
        diff(theta2_dot,theta1_dot) diff(theta2_dot,theta2) diff(
```

```
        theta2_dot,theta2_dot);
    diff(theta2_ddot,x) diff(theta2_ddot,x_dot) diff(theta2_ddot,
        theta1) diff(theta2_ddot,theta1_dot) diff(theta2_ddot,theta2)
        diff(theta2_ddot,theta2_dot);
    ];
J
```

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{2m_1\cos(\theta_1)\sin(\theta_1)\sigma_2}{\sigma_1^2} - \frac{\sigma_4}{\sigma_1} & -\frac{2l_1m_1\theta_1\sin(\theta_1)}{\sigma_1} & \frac{2m_2\cos(\theta_2)\sin(\theta_2)\sigma_2}{\sigma_1^2} - \frac{\sigma_3}{\sigma_1} & -\frac{2l_2m_2\theta_2\sin(\theta_2)}{\sigma_1} \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{g\cos(\theta_1) - \frac{\sin(\theta_1)\sigma_2}{\sigma_1} + \frac{\cos(\theta_1)\sigma_4}{\sigma_1} - \frac{2m_1\cos(\theta_1)^2\sin(\theta_1)\sigma_2}{\sigma_1^2}}{l_1} & -\frac{2m_1\theta_1\cos(\theta_1)\sin(\theta_1)}{\sigma_1} & -\frac{\frac{\cos(\theta_1)\sigma_3}{\sigma_1} - \frac{2m_2\cos(\theta_1)\cos(\theta_2)\sin(\theta_2)\sigma_2}{\sigma_1^2}}{l_1} & -\frac{2l_2m_2\theta_2\cos(\theta_1)\sin(\theta_2)}{l_1\sigma_1} \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{\frac{\cos(\theta_2)\sigma_4}{\sigma_1} - \frac{2m_1\cos(\theta_1)\cos(\theta_2)\sin(\theta_1)\sigma_2}{\sigma_1^2}}{l_2} & -\frac{2l_1m_1\theta_1\cos(\theta_2)\sin(\theta_1)}{l_2\sigma_1} & -\frac{g\cos(\theta_2) - \frac{\sin(\theta_2)\sigma_2}{\sigma_1} + \frac{\cos(\theta_2)\sigma_3}{\sigma_1} - \frac{2m_2\cos(\theta_2)^2\sin(\theta_2)\sigma_2}{\sigma_1^2}}{l_2} & -\frac{2m_2\theta_2\cos(\theta_2)\sin(\theta_2)}{\sigma_1} \end{pmatrix}$$

where

$\sigma_1 = m_1\sin(\theta_1)^2 + m_2\sin(\theta_2)^2 + M$

$\sigma_2 = m_1\sin(\theta_1)\sigma_6 - F + m_2\sin(\theta_2)\sigma_5$

$\sigma_3 = m_2\cos(\theta_2)\sigma_5 - gm_2\sin(\theta_2)^2$

$\sigma_4 = m_1\cos(\theta_1)\sigma_6 - gm_1\sin(\theta_1)^2$

$\sigma_5 = l_2\theta_2^2 + g\cos(\theta_2)$

$\sigma_6 = l_1\theta_1^2 + g\cos(\theta_1)$

```
% Substituting values into matrix A
A = subs(J, [x, x_dot, theta1, theta1_dot, theta2, theta2_dot], [0, 0,
    0, 0, 0, 0])
```

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{g\,m_1}{M} & 0 & -\frac{g\,m_2}{M} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{g+\frac{g\,m_1}{M}}{l_1} & 0 & -\frac{g\,m_2}{M\,l_1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{g\,m_1}{M\,l_2} & 0 & -\frac{g+\frac{g\,m_2}{M}}{l_2} & 0 \end{pmatrix}$$

```
B = [x_dot; x_ddot; theta1_dot; theta1_ddot; theta2_dot; theta2_ddot];
```

$$\begin{pmatrix} \dot{x} \\ -\frac{\sigma_1}{\sigma_2} \\ \dot{\theta_1} \\ -\frac{g\,\sin(\theta_1)+\frac{\cos(\theta_1)\,\sigma_1}{\sigma_2}}{l_1} \\ \dot{\theta_2} \\ -\frac{g\,\sin(\theta_2)+\frac{\cos(\theta_2)\,\sigma_1}{\sigma_2}}{l_2} \end{pmatrix}$$

where

$$\sigma_1 = m_1\,\sin(\theta_1)\,\left(l_1\,{\dot{\theta_1}}^2 + g\,\cos(\theta_1)\right) - F + m_2\,\sin(\theta_2)\,\left(l_2\,{\dot{\theta_2}}^2 + g\,\cos(\theta_2)\right)$$

$$\sigma_2 = m_1\,\sin(\theta_1)^2 + m_2\,\sin(\theta_2)^2 + M$$

```
B = subs(B, [x, x_dot, theta1, theta1_dot, theta2, theta2_dot, F], [0,
    0, 0, 0, 0, 0, 1]);
```

$$\begin{pmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{M\,l_1} \\ 0 \\ \frac{1}{M\,l_2} \end{pmatrix}$$

```
% Check for Controllability
C = [B A*B A*A*B A*A*A*B A*A*A*A*B A*A*A*A*A*B];
% Checking the rank of the Controllability matrix
rank(C)
```

$$\text{ans} = 6$$

```
% For the system to be controllable, its determinant should not be
    equal zero.
det(C)
```

$$-\frac{g^6\,{l_1}^2-2\,g^6\,l_1\,l_2+g^6\,{l_2}^2}{M^6\,{l_1}^6\,{l_2}^6}$$

```
% Simplifying the determinant of the Controllability matrix
simplify(det(C))
```

$$-\frac{g^6\,(l_1-l_2)^2}{M^6\,{l_1}^6\,{l_2}^6}$$

```
% As you can see above the determinant of the system is not zero,
    therefore
% the system is controllable.
```

Therefore, the Condition for controllability is given by (34):

$$l_1 \neq l_2 \tag{34}$$

3.2 LQR Controller

An essential part of control theory, the linear quadratic regulator (LQR) controller is made to find the best control policy for a linear system in order to maximize its performance. By taking control effort into account and minimizing the deviation of system states from desired values, this optimal policy minimizes a quadratic cost function. This is accomplished by the LQR controller, which allows for the design of a control law that dynamically adapts to preserve stability and provide the intended performance by calculating optimal feedback gains for the system states.

Here we are designing the LQR controller to minimize the Cost function $J(K, X_0)$

$$J(K, X_0) = \int_0^\infty (X^T(t)QX(t) + U_k^T RU_k)dt \tag{35}$$

The Q matrix and R matrix are designed such that it reduces the cost function using the Input signal U and state vector X.
The initial condition is taken as:

$$X_0 = [3, 0.3, 20, 1, 10, 2] \tag{36}$$

The Force F:

$$F = 1 \tag{37}$$

```
% LQR controller for Linear and Non-Linear System

clc;
close all;
syms x x_dot theta1_vals theta1_vals_dot theta2 theta2_dot F M m1 m2 g
     l1 l2

x_ddot = (F-(m1*sin(theta1_vals))*(g*cos(theta1_vals)+l1*
    theta1_vals_dot*theta1_vals_dot)-m2*sin(theta2)*(g*cos(theta2)+l2*
    theta2_dot*theta2_dot))/(M+m1*sin(theta1_vals)*sin(theta1_vals)+m2
    *sin(theta2)*sin(theta2));
theta1_vals_ddot = (x_ddot*cos(theta1_vals)-g*sin(theta1_vals))/l1;
theta2_ddot = (x_ddot*cos(theta2)-g*sin(theta2))/l2;
J = [diff(x_dot,x) diff(x_dot,x_dot) diff(x_dot,theta1_vals) diff(
    x_dot,theta1_vals_dot) diff(x_dot,theta2) diff(x_dot,theta2_dot);
     diff(x_ddot,x) diff(x_ddot,x_dot) diff(x_ddot,theta1_vals) diff(
         x_ddot,theta1_vals_dot) diff(x_ddot,theta2) diff(x_ddot,
         theta2_dot);
     diff(theta1_vals_dot,x) diff(theta1_vals_dot,x_dot) diff(
         theta1_vals_dot,theta1_vals) diff(theta1_vals_dot,
         theta1_vals_dot) diff(theta1_vals_dot,theta2) diff(
         theta1_vals_dot,theta2_dot);
     diff(theta1_vals_ddot,x) diff(theta1_vals_ddot,x_dot) diff(
         theta1_vals_ddot,theta1_vals) diff(theta1_vals_ddot,
         theta1_vals_dot) diff(theta1_vals_ddot,theta2) diff(
         theta1_vals_ddot,theta2_dot);
```

```
    diff(theta2_dot,x) diff(theta2_dot,x_dot) diff(theta2_dot,
        theta1_vals) diff(theta2_dot,theta1_vals_dot) diff(theta2_dot,
        theta2) diff(theta2_dot,theta2_dot);
    diff(theta2_ddot,x) diff(theta2_ddot,x_dot) diff(theta2_ddot,
        theta1_vals) diff(theta2_ddot,theta1_vals_dot) diff(
        theta2_ddot,theta2) diff(theta2_ddot,theta2_dot);
];
A = subs(J,[x,x_dot,theta1_vals,theta1_vals_dot,theta2,theta2_dot
    ],[0,0,0,0,0,0]);
B = [x_dot;x_ddot;theta1_vals_dot;theta1_vals_ddot;theta2_dot;
    theta2_ddot];
B = subs(B,[x,x_dot,theta1_vals,theta1_vals_dot,theta2,theta2_dot,F
    ],[0,0,0,0,0,0,1])
```

B value: $\begin{pmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{M l_1} \\ 0 \\ \frac{1}{M l_2} \end{pmatrix}$

```
% Substituting Masses M and lengths l1 and l2
B = subs(B,[M,l1,l2],[1000,20,10])
```

B value: $\begin{pmatrix} 0 \\ \frac{1}{1000} \\ 0 \\ \frac{1}{20000} \\ 0 \\ \frac{1}{10000} \end{pmatrix}$

```
B = double(B)
```

$$B = \begin{bmatrix} 1.0e-03 \\ 0 \\ 1.0000 \\ 0 \\ 0.0500 \\ 0 \\ 0.1000 \end{bmatrix} \tag{38}$$

```
A_vals = subs(A,[M,m1,m2,l1,l2,g],[1000,100,100,20,10,9.8])
```

A value:

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{49}{50} & 0 & -\frac{49}{50} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{539}{1000} & 0 & -\frac{49}{1000} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{49}{500} & 0 & -\frac{539}{500} & 0 \end{pmatrix}$$

```
A_vals = double(A_vals)
```

A values in decimal:

$$A_{vals} = \begin{bmatrix} 0 & 1.0000 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.9800 & 0 & -0.9800 & 0 \\ 0 & 0 & 0 & 1.0000 & 0 & 0 \\ 0 & 0 & -0.5390 & 0 & -0.0490 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0000 \\ 0 & 0 & -0.0980 & 0 & -1.0780 & 0 \end{bmatrix} \tag{39}$$

```
% Initial Eigen Values of the Linearized System
eigen_values = double(eig(A_vals))
```

Eigen Values:

$$eigenvalues = \begin{bmatrix} 0.0000 + 0.0000i \\ 0.0000 + 0.0000i \\ -0.0000 + 0.7282i \\ -0.0000 - 0.7282i \\ 0.0000 + 1.0425i \\ 0.0000 - 1.0425i \end{bmatrix} \tag{40}$$

```
C = eye(6);
```

C matrix:

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{41}$$

```
D = 0;
```

D matrix:

$$D = 0 \tag{42}$$

```
initial_state = [2,0.5,10,0.5,20,1.2];
% Initial State Response
state_space = ss(A_vals,B,C,D);
figure
initial(state_space,initial_state);
```

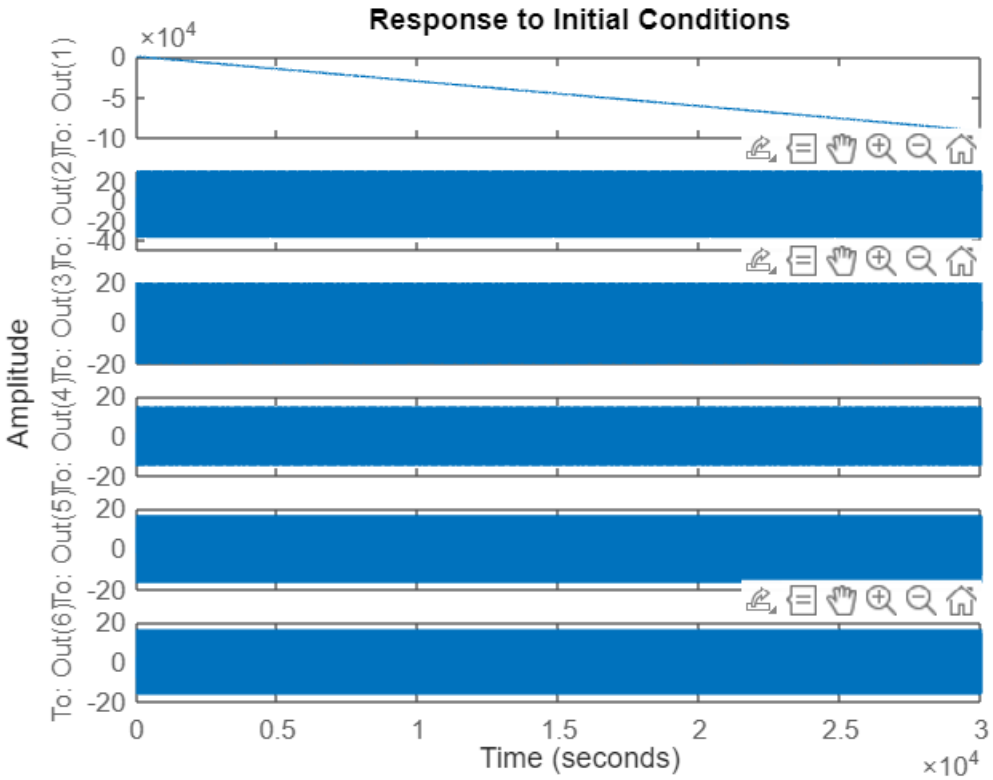

Figure 2: Response to initial condition.

```
Q = [10000 0 0 0 0 0;0 10000 0 0 0 0;0 0 10000 0 0 0;0 0 0 10000 0 0;0
     0 0 0 10000 0;0 0 0 0 0 10000];
R = 0.0001;

% Obtaining the Gain matrix K
figure
[K,P,Poles] = lqr(A_vals,B,Q,R);
eig(A_vals-(B*K))
```

New Eigenvalues:

$$Eigenvalues = \begin{bmatrix} -9.9962 + 0.0000i \\ -1.0060 + 0.0000i \\ -0.0490 + 0.9865i \\ -0.0490 - 0.9865i \\ -0.0174 + 0.6994i \\ -0.0174 - 0.6994i \end{bmatrix} \tag{43}$$

```
lqr_ss = ss(A_vals - (B * K), zeros(6, 1), C, D);
initial(lqr_ss, initial_state);
title('LQR Response to Initial Conditions')
xlabel('Time')
```

```
ylabel('Output')
```

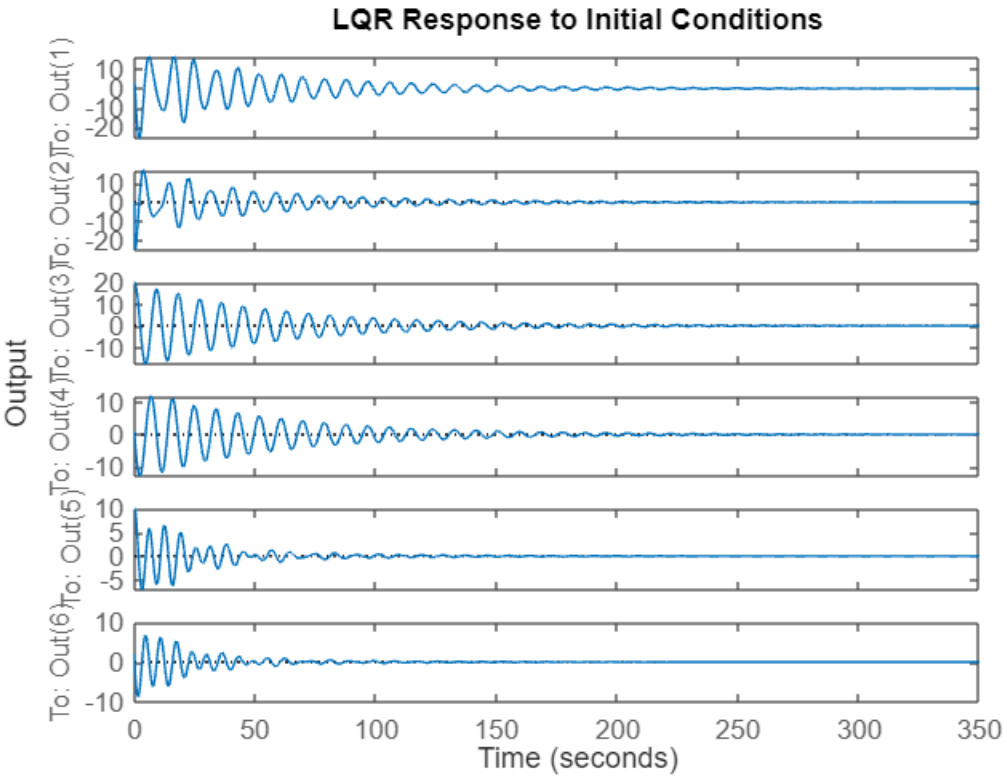

Figure 3: LQR response to initial conditions

```
Poles;
```

$$Poles = \begin{bmatrix} -0.0174 + 0.6994i \\ -0.0174 - 0.6994i \\ -0.0490 + 0.9865i \\ -0.0490 - 0.9865i \\ -1.0060 + 0.0000i \\ -9.9962 + 0.0000i \end{bmatrix} \tag{44}$$

```
disp("Eigen Values of the P Matrix is: ")
eig(P)
```

$$P = \begin{bmatrix} 990.3823 \\ 1.1000 \times 10^4 \\ 2.0291 \times 10^5 \\ 2.1829 \times 10^5 \\ 4.3930 \times 10^5 \\ 9.1418 \times 10^5 \end{bmatrix} \tag{45}$$

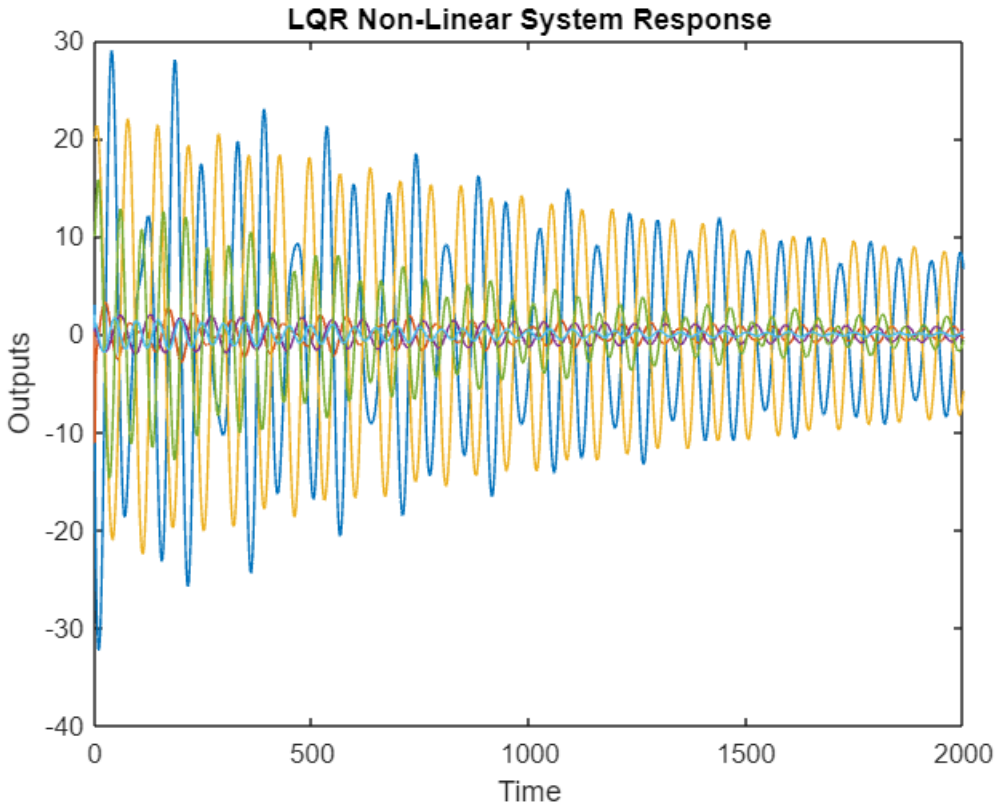

Figure 4: LQR Nonlinear System Response.

```
% Non-Linear Model:
time_span = 0:1:2000;
[time, output] = ode45(@lqr_func, time_span, initial_state);
figure
plot(time, output)
title('LQR Non-Linear System Response')
xlabel('Time')
ylabel('Outputs')
```

The function used to compute the differential equation:

```
function output = lqr_func(t, x)
    output = zeros(6, 1);
    Q = [10000 0 0 0 0 0; 0 10000 0 0 0 0; 0 0 10000 0 0 0; 0 0 0
        10000 0 0; 0 0 0 0 10000 0; 0 0 0 0 0 10000];
    R = 0.0001;
    A = [0 1 0 0 0 0; 0 0 -0.98 0 -0.98 0; 0 0 0 1 0 0; 0 0 -0.539 0
        -0.049 0; 0 0 0 0 0 1; 0 0 -0.098 0 -1.078 0];
    B = [0; 0.001; 0; 0.00005; 0; 0.0001];
    [K, S, P] = lqr(A, B, Q, R);
    % Force = U and U = -KX
    F = -K * x;
    M = 1000;
    m1 = 100;
    m2 = 100;
    theta1 = x(3);
    theta2 = x(5);
    theta1_dot = x(4);
    theta2_dot = x(6);
    l1 = 20;
    l2 = 10;
    g = 9.8;
    x_ddot = (F - (m1 * sind(theta1)) * (g * cosd(theta1) + l1 *
        theta1_dot * theta1_dot) - m2 * sind(theta2) * (g * cosd(
        theta2) + l2 * theta2_dot * theta2_dot)) / (M + m1 * sind(
        theta1) * sind(theta1) + m2 * sind(theta2) * sind(theta2));
    theta1_ddot = (x_ddot * cosd(theta1) - g * sind(theta1)) / l1;
    theta2_ddot = (x_ddot * cosd(theta2) - g * sind(theta2)) / l2;

    % derivative of the state vector X_dot
    output(1) = x(2);
    output(2) = x_ddot;
    output(3) = x(4);
    output(4) = theta1_ddot;
    output(5) = x(6);
    output(6) = theta2_ddot;
end
```

3.3 Observability

The observability of the system can be checked using the observability matrix:

$$O(A,C) = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix} \tag{46}$$

```
% Observability of the system
clc;
close all;
syms x x_dot theta1 theta1_dot theta2 theta2_dot F M m1 m2 g l1 l2
x_ddot = (F-(m1*sin(theta1))*(g*cos(theta1)+l1*theta1_dot*theta1_dot)-
    m2*sin(theta2)*(g*cos(theta2)+l2*theta2_dot*theta2_dot))/(M+m1*sin
    (theta1)*sin(theta1)+m2*sin(theta2)*sin(theta2));
theta1_ddot = (x_ddot*cos(theta1)-g*sin(theta1))/l1;
theta2_ddot = (x_ddot*cos(theta2)-g*sin(theta2))/l2;

J = [diff(x_dot,x) diff(x_dot,x_dot) diff(x_dot,theta1) diff(x_dot,
    theta1_dot) diff(x_dot,theta2) diff(x_dot,theta2_dot);
     diff(x_ddot,x) diff(x_ddot,x_dot) diff(x_ddot,theta1) diff(x_ddot,
         theta1_dot) diff(x_ddot,theta2) diff(x_ddot,theta2_dot);
     diff(theta1_dot,x) diff(theta1_dot,x_dot) diff(theta1_dot,theta1)
         diff(theta1_dot,theta1_dot) diff(theta1_dot,theta2) diff(
         theta1_dot,theta2_dot);
     diff(theta1_ddot,x) diff(theta1_ddot,x_dot) diff(theta1_ddot,
         theta1) diff(theta1_ddot,theta1_dot) diff(theta1_ddot,theta2)
         diff(theta1_ddot,theta2_dot);
     diff(theta2_dot,x) diff(theta2_dot,x_dot) diff(theta2_dot,theta1)
         diff(theta2_dot,theta1_dot) diff(theta2_dot,theta2) diff(
         theta2_dot,theta2_dot);
     diff(theta2_ddot,x) diff(theta2_ddot,x_dot) diff(theta2_ddot,
         theta1) diff(theta2_ddot,theta1_dot) diff(theta2_ddot,theta2)
         diff(theta2_ddot,theta2_dot);
     ];
A = subs(J,[x,x_dot,theta1,theta1_dot,theta2,theta2_dot
    ],[0,0,0,0,0,0]);
```

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{g\,m_1}{M} & 0 & -\frac{g\,m_2}{M} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{g+\frac{g\,m_1}{M}}{l_1} & 0 & -\frac{g\,m_2}{M\,l_1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{g\,m_1}{M\,l_2} & 0 & -\frac{g+\frac{g\,m_2}{M}}{l_2} & 0 \end{pmatrix}$$

```
B = [x_dot;x_ddot;theta1_dot;theta1_ddot;theta2_dot;theta2_ddot];
B = subs(B,[x,x_dot,theta1,theta1_dot,theta2,theta2_dot,F
    ],[0,0,0,0,0,0,1]);
```

$$B = \begin{pmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{M\,l_1} \\ 0 \\ \frac{1}{M\,l_2} \end{pmatrix}$$

```
% From the given problem statement, the output vectors are
% x,(theta1,theta2),(x,theta2),(x,theta1,theta2)
C1 = [1 0 0 0 0 0];
C2 = [0 0 1 0 0 0;
      0 0 0 0 1 0];
C3 = [1 0 0 0 0 0;
      0 0 0 0 1 0];
C4 = [1 0 0 0 0 0;
      0 0 1 0 0 0;
      0 0 0 0 1 0];
% Observability matrix O
O1 = [C1' A'*C1' A'*A'*C1' A'*A'*A'*C1' A'*A'*A'*A'*C1' A'*A'*A'*A'*A
    '*C1'];
O2 = [C2' A'*C2' A'*A'*C2' A'*A'*A'*C2' A'*A'*A'*A'*C2' A'*A'*A'*A'*A
    '*C2'];
O3 = [C3' A'*C3' A'*A'*C3' A'*A'*A'*C3' A'*A'*A'*A'*C3' A'*A'*A'*A'*A
    '*C3'];
O4 = [C4' A'*C4' A'*A'*C4' A'*A'*A'*C4' A'*A'*A'*A'*C4' A'*A'*A'*A'*A
    '*C4'];
% We consider the system to be Observable when the rank of the
% observability is = 6
disp('Rank of output vector 1(x):');
```

Rank of output vector 1(x):

```
rank(O1)
```

$ans = 6$

```
disp('Rank of output vector 2(theta1,theta2):');
```

Rank of output vector 2(theta1,theta2):

```
rank(O2)
```

$ans = 4$

```
disp('Rank of output vector 3(x,theta2):');
```

Rank of output vector 3(x,theta2):

```
rank(O3)
```

$ans = 6$

```
disp('Rank of output vector 4(x,theta1,theta2):');
```

Rank of output vector 4(x,theta1,theta2):

```
rank(O4)
```

$ans = 6$

As we can see using the Output vectors $C1$, $C3$, and $C4$ makes the system observable with full rank, whereas $C2$ is not the output vector to use to observe the system as it does not have full rank $= 6$.

3.4 Luenberger Observer

One important tool in control theory for estimating unmeasured states in dynamic systems is the Luenberger Observer. It is a popular option because of its simplicity and ease of implementation, particularly in situations where not all states can be measured directly. An important observer gain matrix affects performance. The observer uses output measurements and a simplified model to estimate unmeasured states through a set of differential equations. In real-world applications, the Luenberger Observer is still useful for tracking and managing systems despite its susceptibility to modeling errors.

Here Luenberger observer state space equation is given by:

$$\dot{\hat{\mathbf{X}}}(t) = A\hat{\mathbf{X}}(t) + B_K\tilde{\mathbf{U}_K}(t) + L(\tilde{\mathbf{Y}}(t) - C\hat{\mathbf{X}}(t)) \tag{47}$$

Where L is the Luenberger Observer gain matrix.
Here we choose the desired poles as:

$$\text{Desired Poles} = [-10, -20, -30, -50, -60, -70] \tag{48}$$

```
% Luenberger Observer for the System
clc;
close all;
A = [0 1 0 0 0 0; 0 0 -0.98 0 -0.98 0; 0 0 0 1 0 0;0 0 -0.539 0 -0.049
    0;0 0 0 0 0 1;0 0 -0.098 0 -1.078 0];
B = [0;0.001;0;0.00005;0;0.0001];
% From the observability section we saw that C1, C3, and C4 were
   observable
C1 = [1 0 0 0 0 0];
C3 = [1 0 0 0 0 0;
      0 0 0 0 1 0];
C4 = [1 0 0 0 0 0;
      0 0 1 0 0 0;
      0 0 0 0 1 0];
eig(A)
```

$$\text{eigen values} = \begin{bmatrix} 0.0000 + 0.0000i \\ 0.0000 + 0.0000i \\ -0.0000 + 0.7282i \\ -0.0000 - 0.7282i \\ 0.0000 + 1.0425i \\ 0.0000 - 1.0425i \end{bmatrix} \tag{49}$$

```
% We have chosen these poles as the desired poles.
desired_poles = [-10;-20;-30;-50;-60;-70];

% Luenberger Observer 1
L1 = place(A',C1',desired_poles)';
```

$$L1 = \begin{bmatrix} 239.9998 \\ 2.2598 \times 10^{4} \\ -6.1366 \times 10^{8} \\ -2.5979 \times 10^{9} \\ 6.1259 \times 10^{8} \\ 2.5720 \times 10^{9} \end{bmatrix} \tag{50}$$

```
% Luenberger Observer 3
L3 = place(A',C3',desired_poles)';
```

$$L3 = \begin{bmatrix} 147.3259 & 23.7330 \\ 7.0985 \times 10^3 & 2.3281 \times 10^3 \\ -1.2849 \times 10^5 & -6.2514 \times 10^4 \\ -6.9734 \times 10^5 & -4.0342 \times 10^5 \\ 8.4887 & 92.6741 \\ 679.9085 & 2.0480 \times 10^3 \end{bmatrix} \tag{51}$$

```
% Luenberger Observer 4
L4 = place(A',C4',desired_poles)';
```

$$L4 = \begin{bmatrix} 102.1195 & -14.6026 & 0.0220 \\ 2.4875 \times 10^3 & -827.3797 & 0.2473 \\ -14.8408 & 107.8805 & -0.0272 \\ -842.3600 & 2.8120 \times 10^3 & -1.5622 \\ 0.0021 & -0.0027 & 30 \\ 0.0330 & -0.1423 & 198.9220 \end{bmatrix} \tag{52}$$

```
% LQR parameters Q and R
Q = [10000 0 0 0 0 0;0 10000 0 0 0 0;0 0 10000 0 0 0;0 0 0 10000 0 0;0
     0 0 0 10000 0;0 0 0 0 0 10000];
R = 0.0001;

[K,~,~] = lqr(A,B,Q,R); % Gain matrix using LQR controller

% Generating the Luenberger observer closed-loop state equation
    Parameters
% For Observer 1
L1_A = [(A-B*K) B*K;zeros(size(A)) (A-L1*C1)];
L1_B = [B;zeros(size(B))];
L1_C = [C1 zeros(size(C1))];
L1_D = 0;

% For Observer 3
L3_A = [(A-B*K) B*K;zeros(size(A)) (A-L3*C3)];
L3_B = [B;zeros(size(B))];
L3_C = [C3 zeros(size(C3))];
L3_D = 0;

% For Observer 4
L4_A = [(A-B*K) B*K;zeros(size(A)) (A-L4*C4)];
L4_B = [B;zeros(size(B))];
L4_C = [C4 zeros(size(C4))];
L4_D = 0;
```

```
% The initial state of the Observer is all zeroes
initial_state = [2,0.5,10,0.5,20,1.2,0,0,0,0,0,0];

% State space for observer 1
L1_ss = ss(L1_A,L1_B,L1_C,L1_D);

% State space for observer 3
L3_ss = ss(L3_A,L3_B,L3_C,L3_D);

% State space for observer 4
L4_ss = ss(L4_A,L4_B,L4_C,L4_D);

% State space 1 step response
figure
step(L1_ss)
title('Step Response for Observer 1')
xlabel('Time')
ylabel('Output States')
```

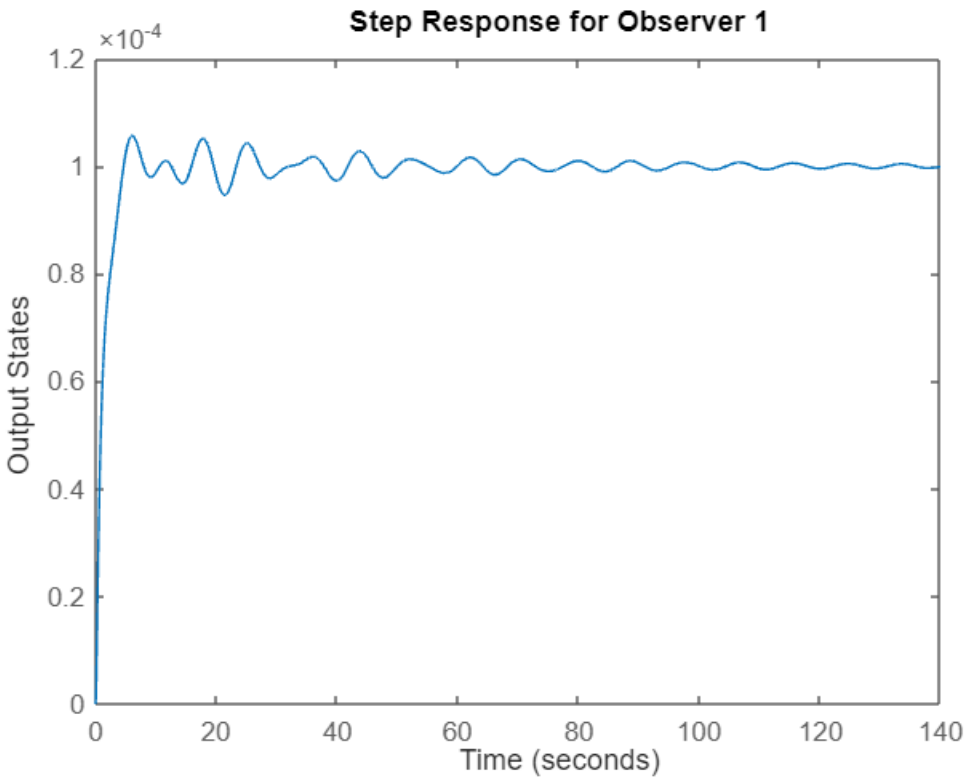

Figure 5: Step Response for Observer 1

```
% State space 1 initial state response
figure
initial(L1_ss, initial_state)
```

```
title('Initial Response for Observer 1')
xlabel('Time')
ylabel('Output States')
```

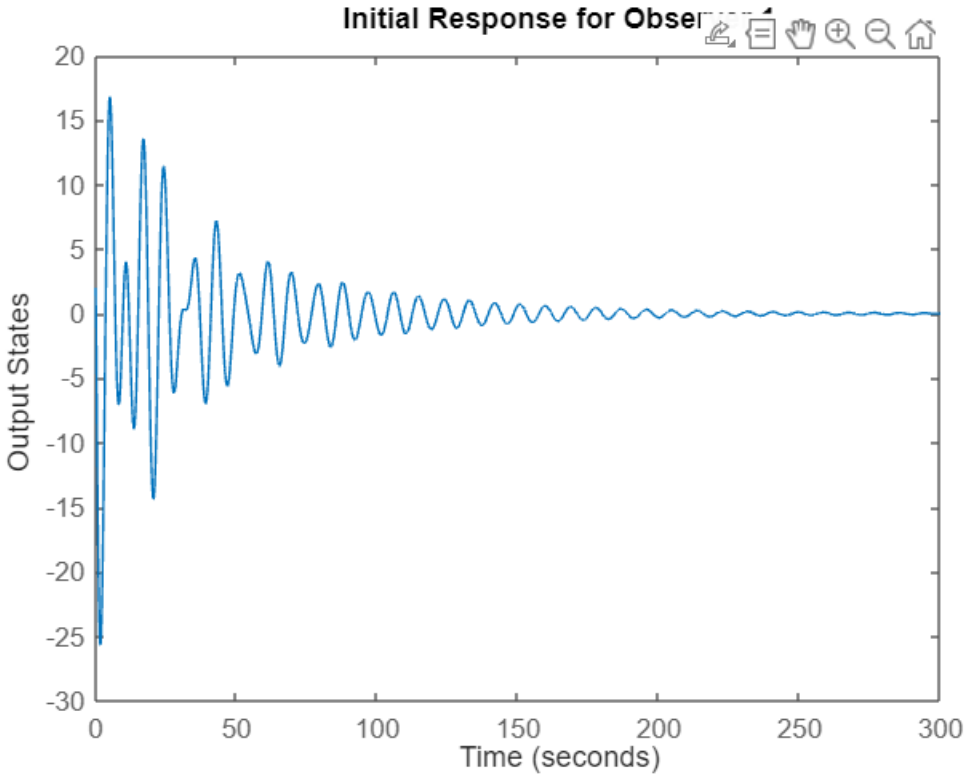

Figure 6: Initial Response for Observer 1.

```
%State space 3 step response
figure
step(L3_ss)
title('Step Response for Observer 3')
xlabel('Time')
ylabel('Output States')
```

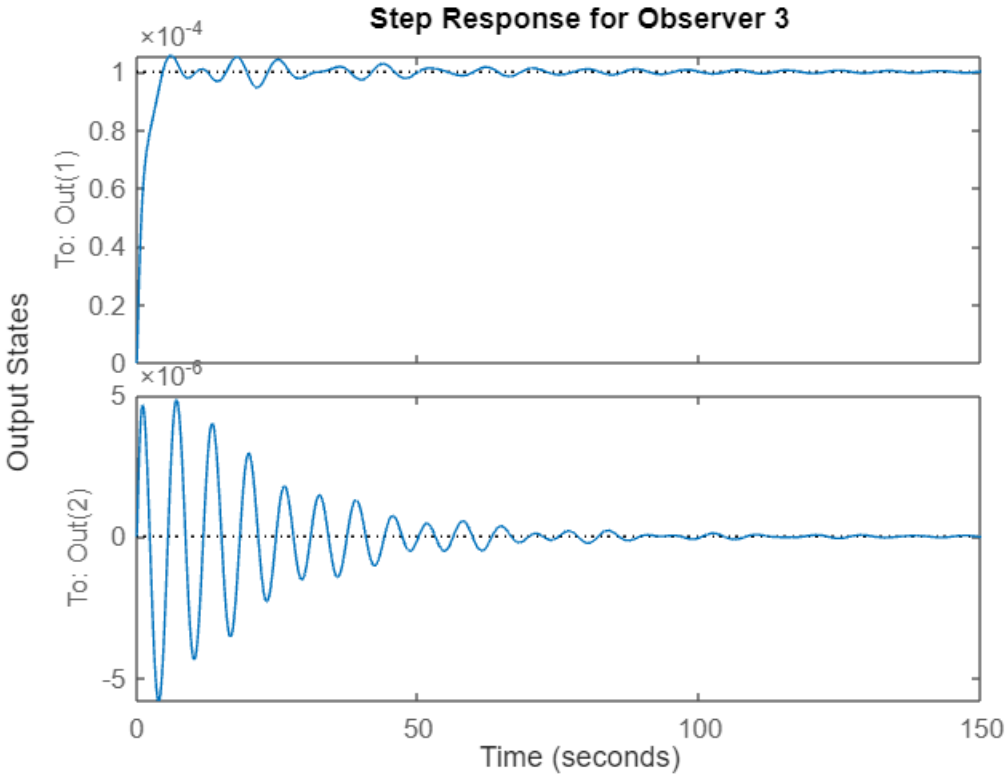

Figure 7: Initial Response for Observer 3.

```
%State space 3 initial state response
figure
initial(L3_ss,initial_state)
title('Initial Response for Observer 3')
xlabel('Time')
ylabel('Output States')
```

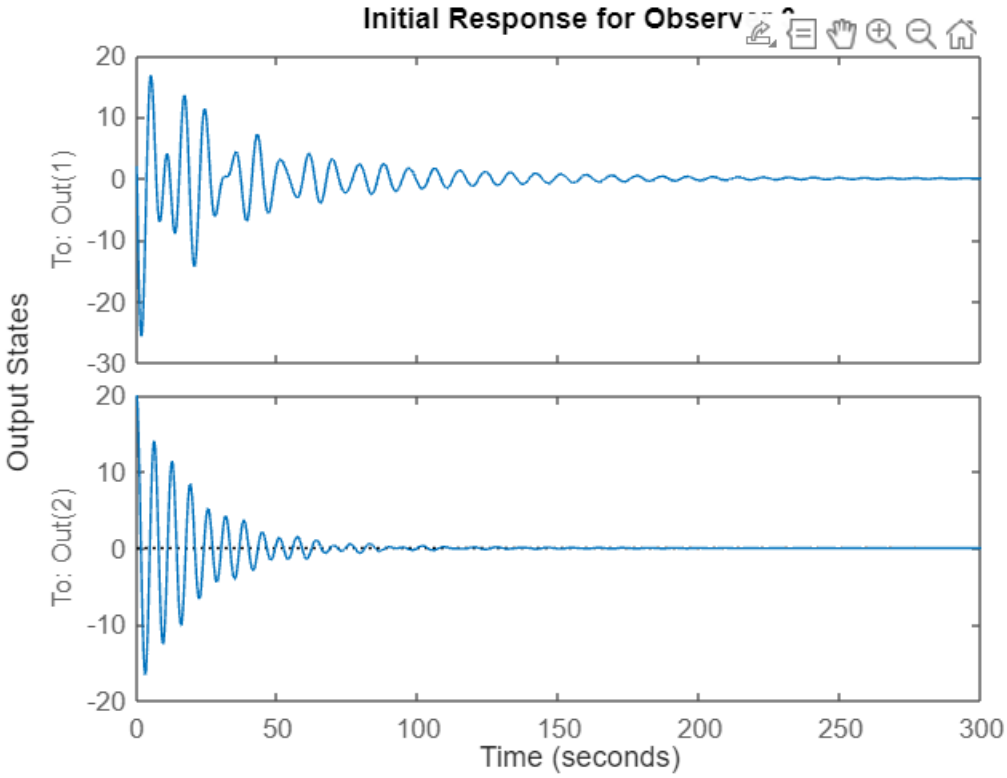

Figure 8: Initial Response for Observer 3.

```
%State space 4 step response
figure
step(L4_ss)
title('Step Response for Observer 4')
xlabel('Time')
ylabel('Output States')
```

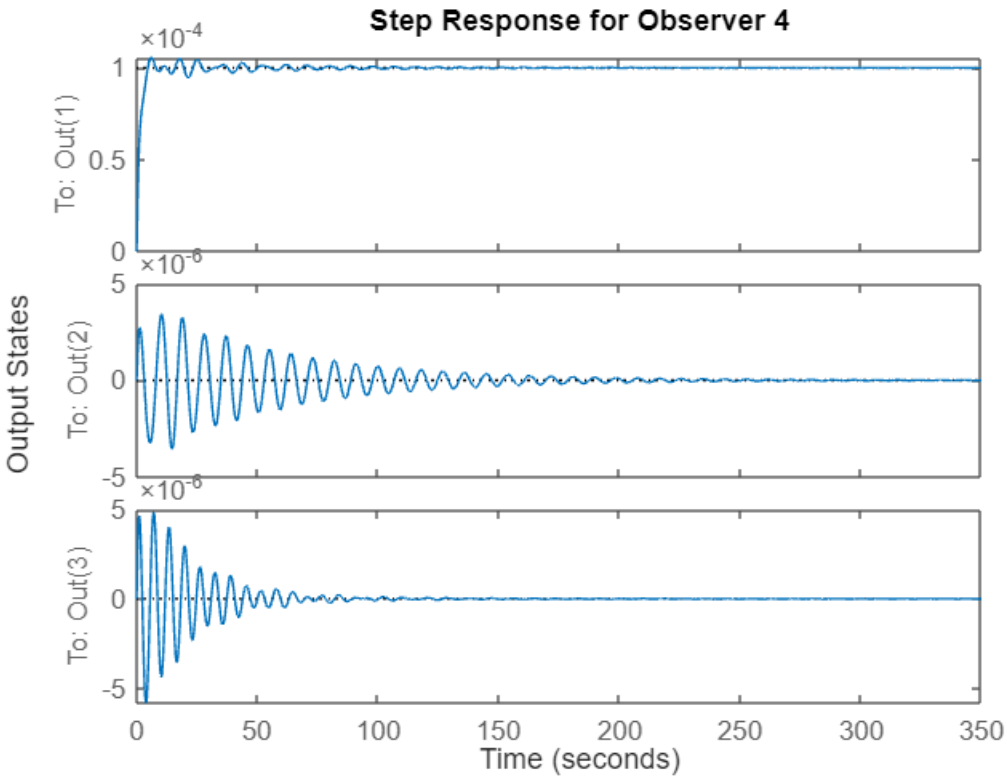

Figure 9: Step Response for Observer 4.

```
%State space 4 initial state response
figure
initial(L4_ss,initial_state)
title('Initial Response for Observer 4')
xlabel('Time')
ylabel('Output States')
```

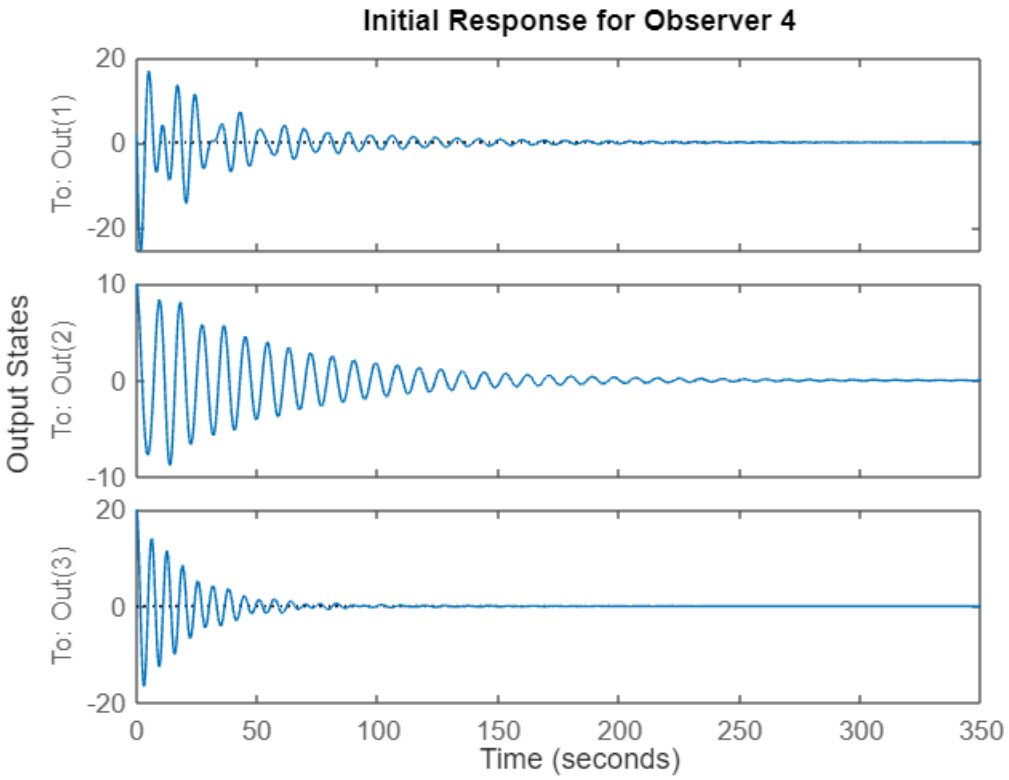

Figure 10: Initial Response of Observer 4.

```
%Luenberger Observer for Non Linear System Model:
time_span = 0:1:1000;
[time,out] = ode45(@L1,time_span,initial_state);
figure
plot(time,out)
title('Luenberger Observer 1 for Non Linear System')
xlabel('Time')
ylabel('Output States')
```

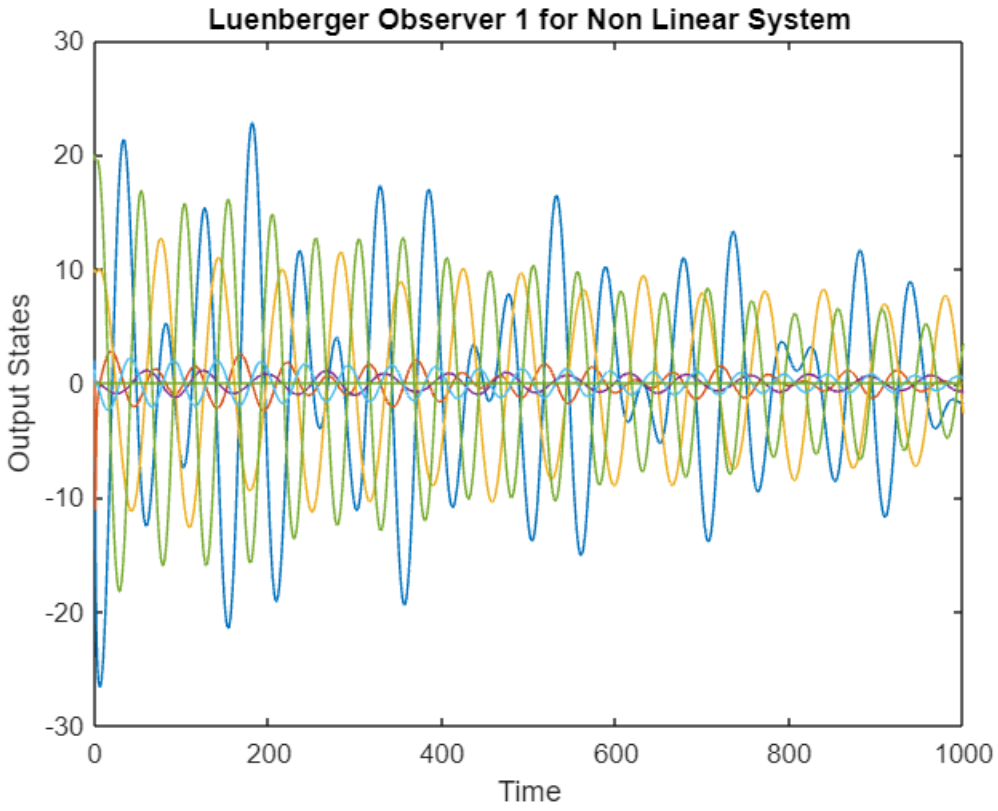

Figure 11: Luenberger Observer 1 for Nonlinear System.

```
[time,out] = ode45(@L3,time_span,initial_state);
figure
plot(time,out)
title('Luenberger Observer 3 for Non Linear System')
xlabel('Time')
ylabel('Output States')
```

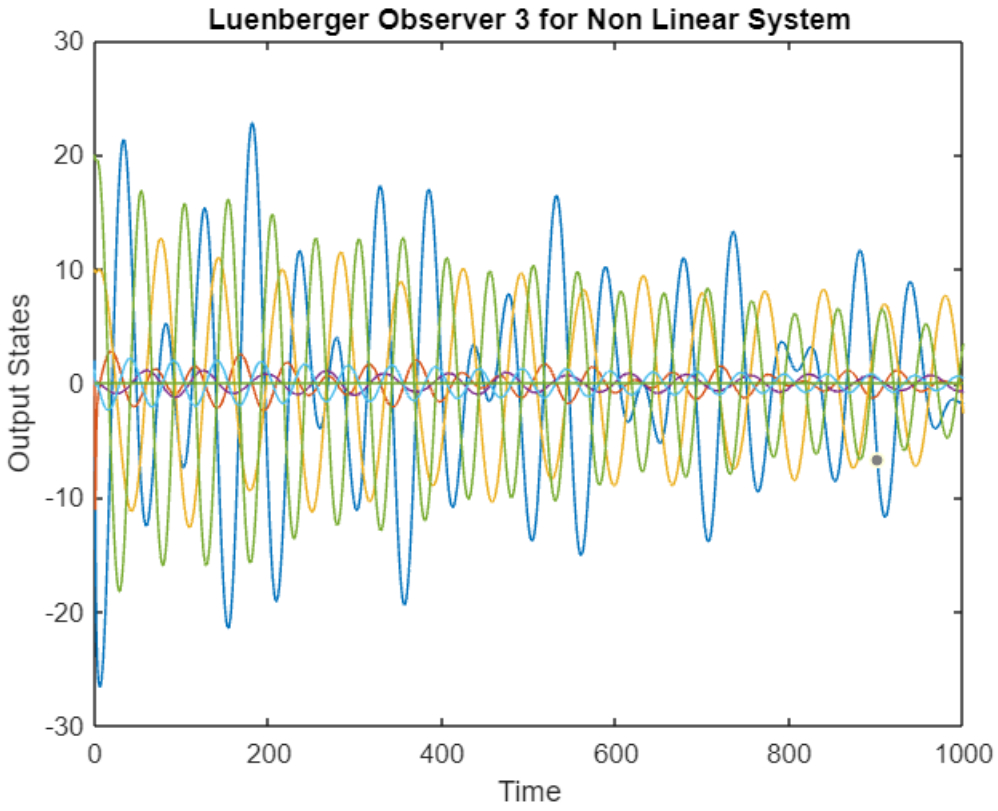

Figure 12: Luenberger Observer 3 for Nonlinear System.

```
figure
[time,out] = ode45(@L4,time_span,initial_state);
plot(time,out)
title('Luenberger Observer 4 for Non Linear System')
xlabel('Time')
ylabel('Output States')
```

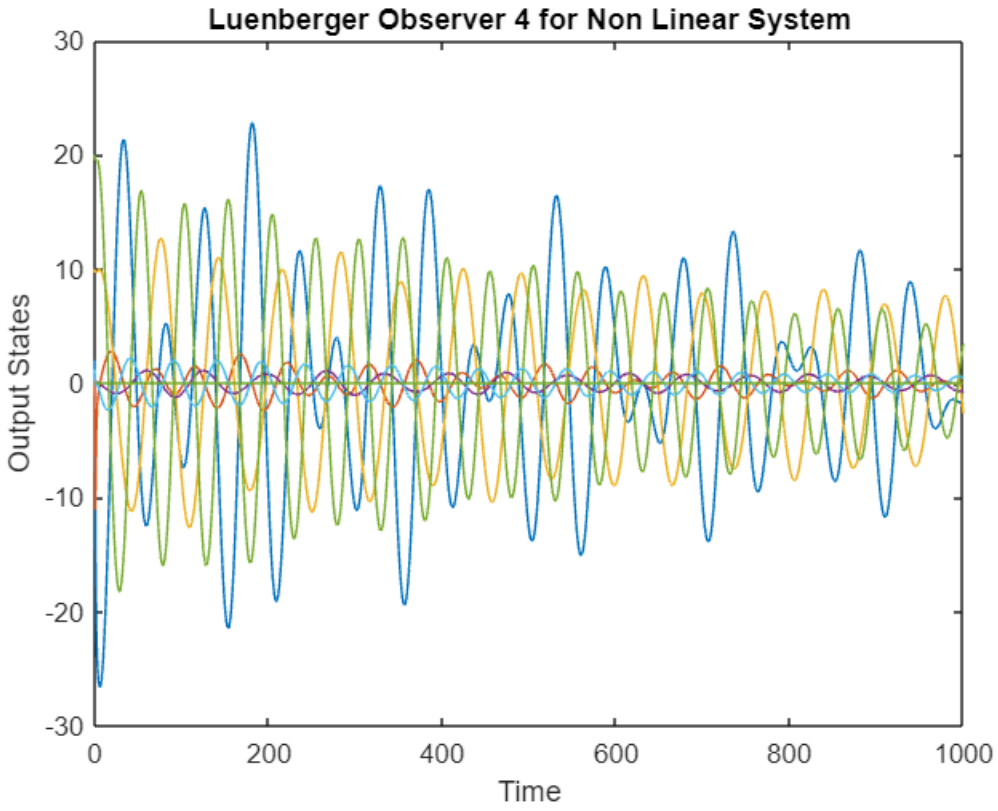

Figure 13: Luenberger Observer 4 for Nonlinear System

3.5 Functions used for solving the Differential Equations for the Observer of Nonlinear Systems

This is the code of the function L1 used for the Luenberger Observer 1 for the Nonlinear System.

```
function states = L1(t,x)
theta1 = x(3);
theta2 = x(5);
theta1_dot = x(4);
theta2_dot = x(6);

l1 = 20;
l2 = 10;
g = 9.8;
M = 1000;
m1 = 100;
m2 = 100;

A = [0 1 0 0 0 0; 0 0 -0.98 0 -0.98 0; 0 0 0 1 0 0;0 0 -0.539 0 -0.049
     0;0 0 0 0 0 1;0 0 -0.098 0 -1.078 0];
B = [0;0.001;0;0.00005;0;0.0001];
```

```
C1 = [1 0 0 0 0 0];

Q = [10000 0 0 0 0 0;0 10000 0 0 0 0;0 0 10000 0 0 0;0 0 0 10000 0 0;0
     0 0 0 10000 0;0 0 0 0 0 10000];
R = 0.0001;
[K,~,~] = lqr(A,B,Q,R);
F= -K*x(1:6); % taking only the first six elements as the initial
    state of the observer is zero

states = zeros(12,1);

x_ddot = (F-(m1*sind(theta1))*(g*cosd(theta1)+l1*theta1_dot*theta1_dot
    )-m2*sind(theta2)*(g*cosd(theta2)+l2*theta2_dot*theta2_dot))/(M+m1
    *sind(theta1)*sind(theta1)+m2*sind(theta2)*sind(theta2));
theta1_ddot = (x_ddot*cosd(theta1)-g*sind(theta1))/l1;
theta2_ddot = (x_ddot*cosd(theta2)-g*sind(theta2))/l2;

desired_poles = [-10;-20;-30;-50;-60;-70];
L1 = place(A',C1',desired_poles)';

lu_states = (A-L1*C1)*x(7:12);
% output states
states(1) = x(2); %because initial state has x,x_dot,theta1,theta1_dot
    ,theta2,theta2_dot,lu_statesimation states (6)
states(2) = x_ddot;
states(3) = x(4);
states(4) = theta1_ddot;
states(5) = x(6);
states(6) = theta2_ddot;
states(7) = lu_states(1);
states(8) = lu_states(2);
states(9) = lu_states(3);
states(10) = lu_states(4);
states(11) = lu_states(5);
states(12) = lu_states(6);
end
```

This is the code of the function L3 used for the Luenberger Observer 3 for the Nonlinear System.

```
function states = L3(t,x)

theta1 = x(3);
theta2 = x(5);
theta1_dot = x(4);
theta2_dot = x(6);
l1 = 20;
l2 = 10;
M = 1000;
m1 = 100;
m2 = 100;
g = 9.8;

A = [0 1 0 0 0 0; 0 0 -0.98 0 -0.98 0; 0 0 0 1 0 0;0 0 -0.539 0 -0.049
     0;0 0 0 0 0 1;0 0 -0.098 0 -1.078 0];
B = [0;0.001;0;0.00005;0;0.0001];

C3 = [1 0 0 0 0 0;
        0 0 0 0 1 0];

Q = [10000 0 0 0 0 0;0 10000 0 0 0 0;0 0 10000 0 0 0;0 0 0 10000 0 0;0
     0 0 0 10000 0;0 0 0 0 0 10000];
R = 0.0001;
[K,~,~] = lqr(A,B,Q,R);
F= -K*x(1:6); % taking only the first six elements as the initial
    state of the observer is zero

states = zeros(12,1);

x_ddot = (F-(m1*sind(theta1))*(g*cosd(theta1)+l1*theta1_dot*theta1_dot
    )-m2*sind(theta2)*(g*cosd(theta2)+l2*theta2_dot*theta2_dot))/(M+m1
    *sind(theta1)*sind(theta1)+m2*sind(theta2)*sind(theta2));
theta1_ddot = (x_ddot*cosd(theta1)-g*sind(theta1))/l1;
theta2_ddot = (x_ddot*cosd(theta2)-g*sind(theta2))/l2;


desired_poles = [-10;-20;-30;-50;-60;-70];
L3 = place(A',C3',desired_poles)';

lu_states = (A-L3*C3)*x(7:12);

% output states
states(1) = x(2);
states(2) = x_ddot;
states(3) = x(4);
states(4) = theta1_ddot;
states(5) = x(6);
```

```
states(6) = theta2_ddot;
states(7) = lu_states(1);
states(8) = lu_states(2);
states(9) = lu_states(3);
states(10) = lu_states(4);
states(11) = lu_states(5);
states(12) = lu_states(6);
end
```

This is the code of the function L4 used for the Luenberger Observer 4 for the Nonlinear System.

```
function states = L4(t,x)

theta1 = x(3);
theta2 = x(5);
theta1_dot = x(4);
theta2_dot = x(6);

M = 1000;
m1 = 100;
m2 = 100;
l1 = 20;
l2 = 10;
g = 9.8;

A = [0 1 0 0 0 0; 0 0 -0.98 0 -0.98 0; 0 0 0 1 0 0;0 0 -0.539 0 -0.049
     0;0 0 0 0 0 1;0 0 -0.098 0 -1.078 0];
B = [0;0.001;0;0.00005;0;0.0001];

C4 = [1 0 0 0 0 0;
       0 0 1 0 0 0;
       0 0 0 0 1 0];

Q = [10000 0 0 0 0 0;0 10000 0 0 0 0;0 0 10000 0 0 0;0 0 0 10000 0 0;0
     0 0 0 10000 0;0 0 0 0 0 10000];
R = 0.0001;
[K,~,~] = lqr(A,B,Q,R);
F= -K*x(1:6); % taking only the first six elements as the initial
    state of the observer is zero

states = zeros(12,1);

x_ddot = (F-(m1*sind(theta1))*(g*cosd(theta1)+l1*theta1_dot*theta1_dot
    )-m2*sind(theta2)*(g*cosd(theta2)+l2*theta2_dot*theta2_dot))/(M+m1
    *sind(theta1)*sind(theta1)+m2*sind(theta2)*sind(theta2));
theta1_ddot = (x_ddot*cosd(theta1)-g*sind(theta1))/l1;
theta2_ddot = (x_ddot*cosd(theta2)-g*sind(theta2))/l2;


desired_poles = [-10;-20;-30;-50;-60;-70];
L4 = place(A',C4',desired_poles)';


lu_states = (A-L4*C4)*x(7:12);
% output states
states(1) = x(2); %because initial state has x,x_dot,theta1,theta1_dot
    ,theta2,theta2_dot,lu_statesimation states (6)
states(2) = x_ddot;
```

```
states(3) = x(4);
states(4) = theta1_ddot;
states(5) = x(6);
states(6) = theta2_ddot;
states(7) = lu_states(1);
states(8) = lu_states(2);
states(9) = lu_states(3);
states(10) = lu_states(4);
states(11) = lu_states(5);
states(12) = lu_states(6);
end
```

3.6 LQG controller

In control theory, the Linear-Quadratic-Gaussian (LQG) control paradigm is a key idea that is specifically used to optimize the control of linear systems impacted by additive white Gaussian noise. Formulating an output feedback law that minimizes the expected value of a quadratic cost criterion is the fundamental task of the LQG control problem. A linear–quadratic regulator (LQR) and a Kalman filter (which functions as a linear–quadratic state estimator, or LQE) combine to form the unique and widely used LQG controller. Especially in model predictive control scenarios, this integrated controller proves to be highly useful. The ability to independently design the state estimator and state feedback via the separation principle is what makes the LQG approach special. The best solution follows a conventional output feedback setup with a Luenberger observer. The MATLAB implementation of the LQG approach is crucial as it utilizes measurement noise and weighted process noise to produce the necessary state-space representation of the system. It is noteworthy that the response of the system varies with each LQG controller code execution because distinct noise terms are introduced through the use of white Gaussian noise, which is enabled by the *wgn* function and dependent on the Communication Toolbox.

QXU is one of the weighting matrices, it is the cost function designed as a 7x1 Identity matrix. The size of the identity matrix is determined by the number of states and control inputs in the system. The cost is equally weighted for all state variables.
White Gaussian noise is generated using the *wgn* function. w is the state noise, and v is the measurement noise. The QWV is the other weighting matrix constructed by stacking these vectors of noise and forming a covariance matrix.

```
% LQG For Linear Model
clc;
close all;

% The state-space matrices
A = [0 1 0 0 0 0; 0 0 -0.98 0 -0.98 0; 0 0 0 1 0 0;0 0 -0.539 0 -0.049
     0;0 0 0 0 0 1;0 0 -0.098 0 -1.078 0];
B = [0;0.001;0;0.00005;0;0.0001];

C1 = [1 0 0 0 0 0]; % smallest output vector

Q = [10000 0 0 0 0 0;0 10000 0 0 0 0;0 0 10000 0 0 0;0 0 0 10000 0 0;0
     0 0 0 10000 0;0 0 0 0 0 10000];
R = 0.0001;
D=0;

% Weighting matrix
QXU = eye(7);

% State noise vector with power 3dbW
w = wgn(6,1,3)
```

$$\text{ans} = \begin{bmatrix} 2.0318 \\ 0.4593 \\ -1.0664 \\ 1.9356 \\ -2.4176 \\ -0.1444 \end{bmatrix} \tag{53}$$

```
% Measurement noise vector with power 3dbW
v = wgn(1,1,3)
```

v = -0.3411

```
%Covariance matrix
QWV = [w;v]*[w' v']
```

$$\text{ans} = \begin{bmatrix} 4.1281 & 0.9333 & -2.1666 & 3.9327 & -4.9120 & -0.2934 & -0.6929 \\ 0.9333 & 0.2110 & -0.4898 & 0.8891 & -1.1105 & -0.0663 & -0.1567 \\ -2.1666 & -0.4898 & 1.1371 & -2.0641 & 2.5780 & 0.1540 & 0.3637 \\ 3.9327 & 0.8891 & -2.0641 & 3.7465 & -4.6795 & -0.2795 & -0.6601 \\ -4.9120 & -1.1105 & 2.5780 & -4.6795 & 5.8447 & 0.3492 & 0.8245 \\ -0.2934 & -0.0663 & 0.1540 & -0.2795 & 0.3492 & 0.0209 & 0.0493 \\ -0.6929 & -0.1567 & 0.3637 & -0.6601 & 0.8245 & 0.0493 & 0.1163 \end{bmatrix} \tag{54}$$

```
% Generates the LQG regulator for the given state space and the
    weighted
% matrices
lqg_ss = lqg(ss(A,B,C1,D),QXU,QWV)
```

```
lqg_ss =

  A =
             x1_e      x2_e      x3_e      x4_e      x5_e      x6_e
   x1_e    -1.502         1         0         0         0         0
   x2_e    -1.373    -0.049   -0.9785   0.07827   -0.9792   0.03919
   x3_e   0.09679         0         0         1         0         0
   x4_e   -0.5954  -0.00245   -0.5389  0.003913  -0.04896  0.001959
   x5_e     0.479         0         0         0         0         1
   x6_e   -0.2798   -0.0049  -0.09785  0.007827    -1.078  0.003919
```

Figure 14:

```
B =
             y1
 x1_e     1.502
 x2_e     1.372
 x3_e  -0.09679
 x4_e    0.5954
 x5_e    -0.479
 x6_e    0.2797

C =
       x1_e    x2_e    x3_e    x4_e    x5_e    x6_e
 u1      -1     -49   1.532   78.27  0.7671   39.19
```

Figure 15:

```
C =
       x1_e    x2_e    x3_e    x4_e    x5_e    x6_e
 u1      -1     -49   1.532   78.27  0.7671   39.19

D =
     y1
 u1   0
```

Figure 16:

```
%Taking the initial state response of the system
initial_state = [2,0.5,10,0.5,20,1.2];
figure
initial(lqg_ss,initial_state)
title('LQG Controller for Linear System')
xlabel('Time')
ylabel('Output')
```

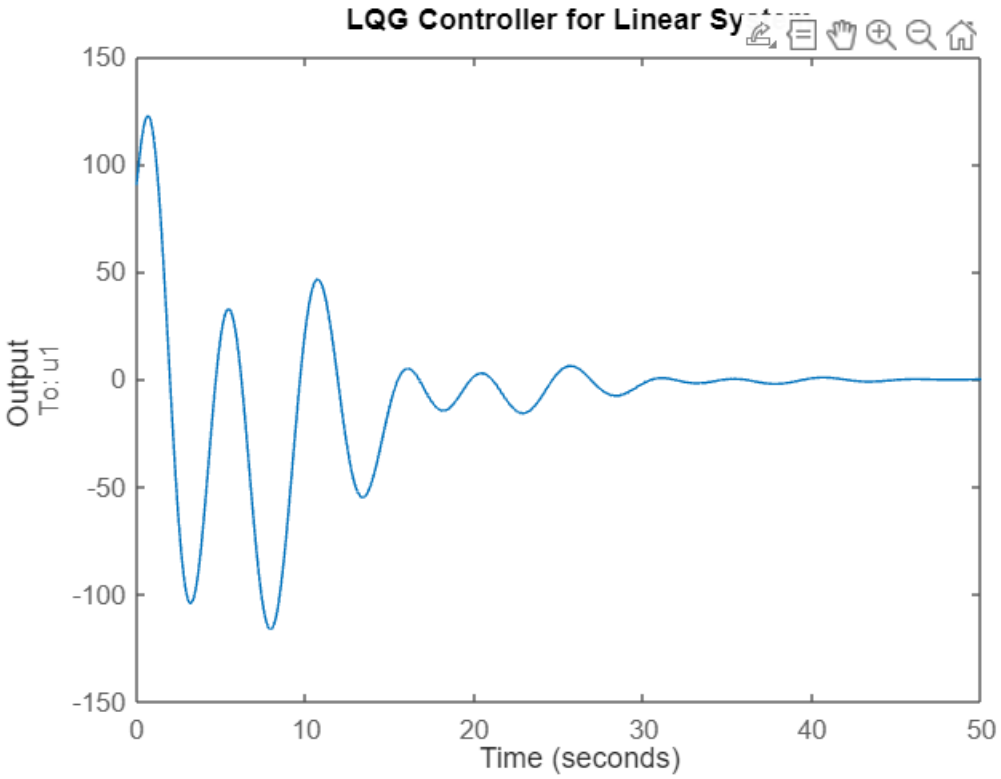

Figure 17: LQG controller for linear system

```
%LQG for non-linear system model
initial_state = [2,0.5,10,0.5,20,1.2,0,0,0,0,0,0];
time_span = 0:1:2000;
[time,out] = ode45(@ode45_lqg,time_span,initial_state);
figure
plot(time,out)
title('LQG Controller for Non Linear Model')
xlabel('Time')
ylabel('State')
```

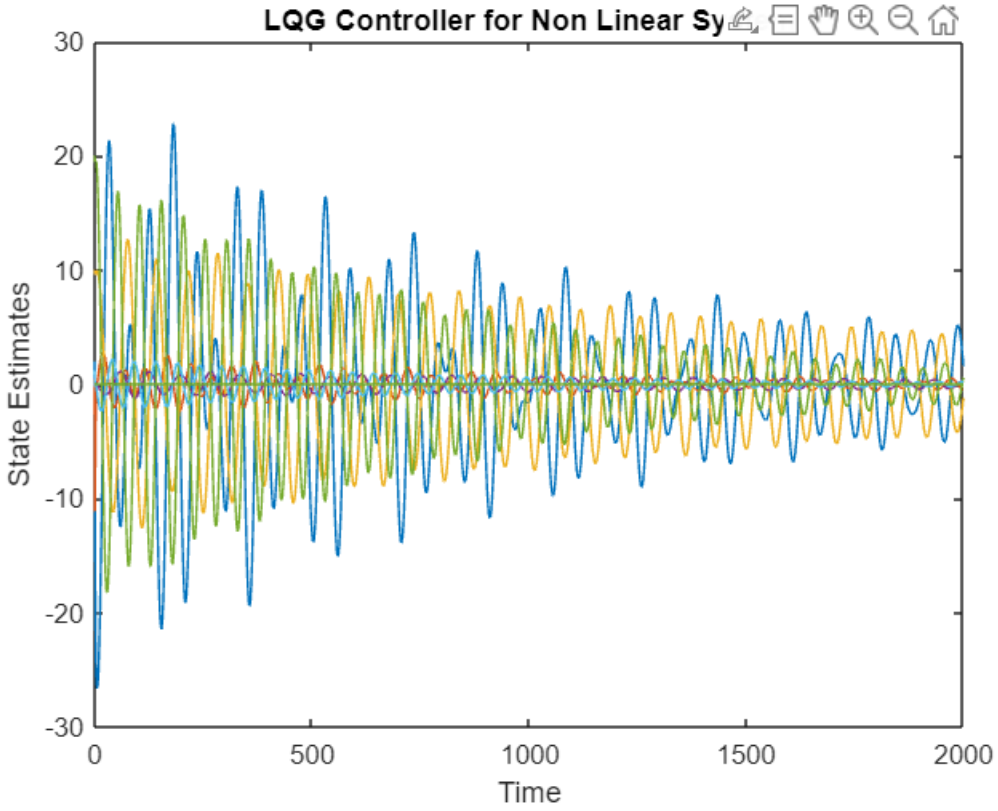

Figure 18: LQG controller for non linear system.

3.7 Function to implement LQG controller for Non linear system:

```
function states = lqg_func(t,x)

states = zeros(12,1);

theta1 = x(3);
theta2 = x(5);
theta_dot_1 = x(4);
theta_dot_2 = x(6);

l1 = 20;
l2 = 10;
g = 9.8;
M = 1000;
m1 = 100;
m2 = 100;

A = [0 1 0 0 0 0; 0 0 -0.98 0 -0.98 0; 0 0 0 1 0 0;0 0 -0.539 0 -0.049
     0;0 0 0 0 0 1;0 0 -0.098 0 -1.078 0];
B = [0;0.001;0;0.00005;0;0.0001];

C1 = [1 0 0 0 0 0]; % smallest output vector

Q = [10000 0 0 0 0 0;0 10000 0 0 0 0;0 0 10000 0 0 0;0 0 0 10000 0 0;0
     0 0 0 10000 0;0 0 0 0 0 10000];
R = 0.0001;
D=0;

[K,~,~] = lqr(A,B,Q,R);
F= -K*x(1:6);



x_ddot = (F-(m1*sind(theta1))*(g*cosd(theta1)+l1*theta_dot_1*
    theta_dot_1)-m2*sind(theta2)*(g*cosd(theta2)+l2*theta_dot_2*
    theta_dot_2))/(M+m1*sind(theta1)*sind(theta1)+m2*sind(theta2)*sind
    (theta2));
theta1_ddot = (x_ddot*cosd(theta1)-g*sind(theta1))/l1;
theta2_ddot = (x_ddot*cosd(theta2)-g*sind(theta2))/l2;


n1 = eye(6);
n2 = 1;
k = lqr(A',C1',n1,n2)';

lqg_states = (A-k*C1)*x(7:12);

states(1) = x(2);
```

```
states(2) = x_ddot;
states(3) = x(4);
states(4) = theta1_ddot;
states(5) = x(6);
states(6) = theta2_ddot;
states(7) = lqg_states(1);
states(8) = lqg_states(2);
states(9) = lqg_states(3);
states(10) = lqg_states(4);
states(11) = lqg_states(5);
states(12) = lqg_states(6);
end
```

4 Assumptions

- The cart moves with no resistance between the ground and the wheels.
- The Luenberger observer's poles are positioned far from the system's poles. Comments are included in the code wherever they are deemed important for comprehension.
- Estimates rather than state variable errors are displayed in the plots for the non-linear system simulations.
- It is evident that the estimates are damping as well, indicating that if more time is allowed, the system will reach equilibrium.
- It is assumed that the acceleration caused by gravity is $9.8\,\mathrm{m/s^2}$.
- The two loads do not collide with one another.

References

[1] What is linearization? https://www.youtube.com/watch?v=5gEattuH3tI

[2] Practical Linearization https://www.youtube.com/watch?v=K1oMF4684OQ

[3] Control Bootcamp: Full-State Estimation https://www.youtube.com/watch?v=MZJMi-6_4UU

[4] Control Bootcamp: Linear Quadratic Gaussian (LQG) https://www.youtube.com/watch?v=H4_hFazBGxU

[5] What is Linear Quadratic Regulator (LQR) Optimal Control? — State Space, Part 4 https://www.youtube.com/watch?v=E_RDCFOlJx4

[6] *Robot Modeling and Control, 2nd Edition*, Mark W. Spong, Seth Hutchinson, M. Vidyasagar

[7] LQG Toolbox https://www.mathworks.com/help/control/ref/ss.lqg.html?s_tid=doc_ta

[8] LQR Toolbox https://www.mathworks.com/help/control/ref/lti.lqr.html?s_tid=doc_ta

[9] Linear–quadratic regulator (Wikipedia) https://en.wikipedia.org/wiki/Linear%E2%80%93quadratic_regulator

[10] MIT OpenCourseWare - Maneuvering and Control of Surface and Underwater Vehicles https://ocw.mit.edu/courses/2-154-maneuvering-and-control-of-surface-and-underwater-vehicles-13-49-fall-2004/2d5318503b8f97d5a5255596fcf2e3f1_lec19.pdf

[11] Stanford EE363 - Lecture on Linear Quadratic Regulator https://stanford.edu/class/ee363/lectures/clqr.pdf

[12] MathWorks - Design an LQG Regulator https://www.mathworks.com/help/control/getstart/design-an-lqg-regulator.html?searchHighlight=lqg&s_tid=srchtitle_support_results_6_lqg

[13] Math Stack Exchange - Dynamical Systems Luenberger Observers: How to Calculate the Gain Matrix and A Priori Estimates https://math.stackexchange.com/questions/3782397/dynamical-systems-luenberger-observers-how-to-calculate-the-gain-matrix-and-ap

5 Conclusion

With MATLAB, the provided set of problems was successfully implemented. The system's LQG (Linear Quadratic Gaussian) controller includes noise terms to improve its resilience. The LQG controller can demonstrate its ability to reject such disturbances as part of its design by effectively treating situations where a persistent disturbance impacts the input as an external disturbance. A constant reference can be tracked using the state equation of the system, and the reference itself can be tracked using the same system. An alternate method of reference tracking is made possible by adding a new state variable as an integral term of the error in the state equation of the LQR (Linear Quadratic Regulator) controller. Common names for this variation include residual controller and linear quadratic integral (LQI) controller. The LQI controller incorporates a crucial element to mitigate steady-state errors and enhance the system's tracking efficacy gradually.