ENPM 667 Control of Robotics Systems

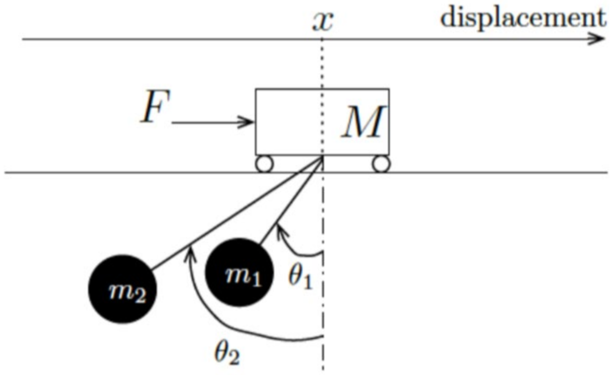
Final Project

Pradipkumar Kathiriya	Shelvin Pauly
117678345	118426556

Table Of Contents

Problem Statement

Consider a crane that moves along a one-dimensional track. It behaves as a frictionless cart with mass M actuated by an external force F that continues the input of the system. There are two loads suspended from cables attached to the crane. The loads have mass m_1 and m_2, and the lengths of the cables are l_1 and l_2, respectively. The following figure depicts the crane and associated variables used throughout this project.



Some notations throughout the project: -

- Mass of the cart: M
- Force applied on the cart: F
- Mass on the first loads: m_1
- Mass on the second loads: m_2
- Angle made by the cable for load 1 with the vertical axis: θ_1
- Angle made by the cable for load 2 with the vertical axis: θ_2
- Length of cable with mass 1: l_1
- Length of cable with mass 1: l_2

Assumptions: -

- The cables are rigid and massless.
- The system is frictionless
- The trolley moves only in the x direction
- The loads can move in the x and y directions

Part a] Equations of motion and the corresponding nonlinear state-space representation:

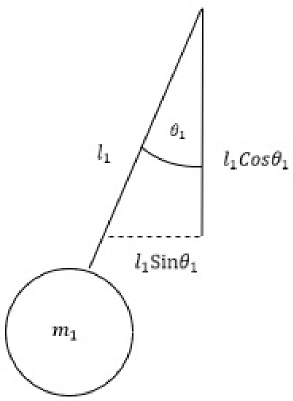
The Euler-Lagrange equations lead to a set of coupled second order equations and provides a formulation of the dynamic equations of motion equivalent Newton's second law of motion.

This problem is similar to a crane problem where we have two pendulums suspended from a cart which is allowed to move in the x direction. The masses can move in the x and y direction

Generalized Co-ordinates: -

$$q_1 = x_1 \, ; \, q_2 = x_2 \, ; \, q_3 = x_3$$

Firstly, we consider mass 1,



The position vector is given by,

$$r_1 = (x - l_1 Sin\theta_1)\hat{\imath} - l_1 Cos\theta_1 \hat{\jmath}$$

Differentiating with respect to time 't',

$$\dot{r}_1 = \left(\dot{x} - l_1 Cos\theta_1 \dot{\theta}_1\right)\hat{\imath} + l_1 Sin\theta_1 \dot{\theta}_1 \hat{\jmath}$$

The dot product $\dot{r}_1 \, . \, \dot{r}_1$ gives,

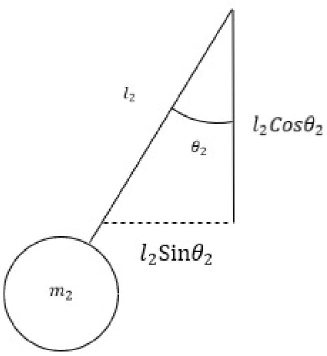
$$\dot{r}_1^2 = \left(\dot{x} - l_1 Cos\theta_1 \dot{\theta}_1\right)^2 + (l_1 Sin\theta_1 \dot{\theta}_1)^2$$

$$\dot{r}_1^2 = \dot{x}^2 - 2\dot{x} l_1 Cos\theta_1 \dot{\theta}_1 + l_1^2 \dot{\theta}_1^2$$

The vertical component is given by h_1,

$$h_1 = -l_1 Cos\theta_1$$

Now, considering mass 2: -



Similar to mass 1, we can write that,

$$\dot{r}_2^2 = \dot{x}^2 - 2\dot{x} l_2 Cos\theta_2 \dot{\theta}_2 + l_2^2 \dot{\theta}_2^2$$

$$h_2 = -l_2 Cos\theta_2$$

The Lagrangian is given by,

$$L = K.E._{system} - P.E._{system}$$

$$L = (KE)_{cart} + (KE)_{mass1} + (KE)_{mass2} - (PE)_{cart} - (PE)_{mass1} - (PE)_{mass2}$$

Kinetic Energy,

$$K.E. = \frac{1}{2} Mass(Acceleration)$$

Potential Energy,

$$P.E. = Mass\ .Acceleration\ due\ to\ gravity\ .Height$$

Now, substituting all the values, we have

$$L = \frac{1}{2}M\dot{x}^2 + \frac{1}{2}m_1\left(\dot{x}^2 - 2\dot{x}l_1Cos\theta_1\dot{\theta}_1 + l_1^2\dot{\theta}_1^2\right) + \frac{1}{2}m_2\left(\dot{x}^2 - 2\dot{x}l_2Cos\theta_2\dot{\theta}_2 + l_2^2\dot{\theta}_2^2\right) - 0 - (-l_2Cos\theta_2) - (-l_2Cos\theta_2)$$

$$L = \frac{1}{2}(M + m_1 + m_2)\dot{x}^2 - \left(m_1l_1Cos\theta_1\dot{\theta}_1 + m_2l_2Cos\theta_2\dot{\theta}_2\right)\dot{x} + \frac{1}{2}m_1l_1^2\dot{\theta}_1^2 + \frac{1}{2}m_1l_1^2\dot{\theta}_2^2 + m_1l_1Cos\theta_1g + m_2l_2Cos\theta_2g$$

$$L = \frac{1}{2}(M + m_1 + m_2)\dot{x}^2 + m_1l_1Cos\theta_1\left(g - \dot{\theta}_1\dot{x}\right) + m_2l_2Cos\theta_2\left(g - \dot{\theta}_2\dot{x}\right) + \frac{1}{2}(m_1l_1^2\dot{\theta}_1^2 + m_2l_2^2\dot{\theta}_2^2) \tag{1}$$

Euler-Lagrange's Equation of motion: -

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) - \left(\frac{\partial L}{\partial x}\right) = F \tag{2}$$

The partial differentiation of L with respect to $'\dot{x}'$,

$$\left(\frac{\partial L}{\partial \dot{x}}\right) = (M + m_1 + m_2)\dot{x} + (m_1l_1Cos\theta_1)\left(0 - \dot{\theta}_1\right) + (m_2l_2Cos\theta_2\left(0 - \dot{\theta}_2\right)$$

Differentiating with respect to time 't',

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) = (M + m_1 + m_2)\ddot{x} - (m_1l_1Cos\theta_1\ddot{\theta}_1 - m_1l_1Sin\theta_1\dot{\theta}_1^2) - (m_2l_2Cos\theta_2\ddot{\theta}_2 - m_2l_2Sin\theta_2\dot{\theta}_2^2)$$

Now the partial differentiation of L with respect to 'x'

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

Substituting the values in equation (2),

$$F = (M + m_1 + m_2)\ddot{x} - (m_1 l_1 Cos\theta_1 \ddot{\theta}_1 - m_1 l_1 Cos\theta_1 \dot{\theta}_1^2) - (m_2 l_2 Cos\theta_2 \ddot{\theta}_2 - m_2 l_2 Cos\theta_2 \dot{\theta}_2^2) \tag{3}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_1}\right) - \left(\frac{\partial L}{\partial \theta_1}\right) = 0 \tag{4}$$

The partial differentiation of L with respect to $'\,\dot{\theta}_1'$,

$$\frac{\partial L}{\partial \dot{\theta}_1} = 0 - m_1 l_1 Cos\theta_1 \big(0 - \dot{\theta}_1 \dot{x}\big) + m_1 l_1^2 \dot{\theta}_1^2$$

Differentiating with respect to time 't',

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_1}\right) = -m_1 l_1 (Cos\theta_1 \ddot{x} + \dot{x} Sin\theta_1 \dot{\theta}_1) + m_1 l_1^2 \ddot{\theta}_1$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_1}\right) = -m_1 l_1 Cos\theta_1 \ddot{x} + m_1 l_1 Sin\theta_1 \dot{x} \dot{\theta}_1 + m_1 l_1^2 \ddot{\theta}_1$$

Now the partial differentiation of L with respect to 'θ_1'

$$\frac{\partial L}{\partial \theta_1} = 0 + m_1 l_1 \big(g - \dot{\theta}_1 \dot{x}\big)(-Sin\theta_1)$$

$$\frac{\partial L}{\partial \theta_1} = -m_1 l_1 g Sin\theta_1 + m_1 l_1 Sin\theta_1 \dot{\theta}_1 \dot{x}$$

Substituting the values in equation (4),

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_1}\right) - \frac{\partial L}{\partial \theta_1} = -m_1 l_1 Cos\theta_1 \ddot{x} + m_1 l_1^2 \ddot{\theta}_1 + m_1 l_1 g Sin\theta_1$$

$$-m_1 l_1 Cos\theta_1 \ddot{x} + m_1 l_1^2 \ddot{\theta}_1 + m_1 l_1 g Sin\theta_1 = 0 \tag{5}$$

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

Similarly, as equation (5),

$$-m_2 l_2 Cos\theta_2 \ddot{x} + m_2 l_2^2 \ddot{\theta}_2 + m_2 l_2 g Sin\theta_2 = 0 \tag{6}$$

From equation (5) and (6),

$$m_1 l_1 \ddot{\theta}_1 = m_1 Cos\theta_1 \ddot{x} - m_1 g Sin\theta_1 \tag{7}$$

$$m_2 l_2 \ddot{\theta}_2 = m_2 Cos\theta_2 \ddot{x} - m_2 g Sin\theta_2 \tag{8}$$

Substitute equations (7) and (8) into equation (3),

$$F = (M + m_1 + m_2)\ddot{x} + m_1 l_1 Sin\theta_1 \dot{\theta}_1^2 - Cos\theta_1 (m_1 Cos\theta_1 \ddot{x} - m_1 g Sin\theta_1) + m_2 l_2 Sin\theta_2 \dot{\theta}_2^2 - Cos\theta_2 (m_2 Cos\theta_2 \ddot{x} - m_2 g Sin\theta_2)$$

$$F = (M + m_1 + m_2 - m_1 Cos^2\theta_1 - m_1 Cos^2\theta_2)\ddot{x} + m_1 l_1 Sin\theta_1 \dot{\theta}_1^2 + m_1 g Sin\theta_1 Cos\theta_1 + m_2 l_2 Sin\theta_2 \dot{\theta}_2^2 + m_2 g Sin\theta_2 Cos\theta_2$$

$$\ddot{x} = \frac{F - m_1 l_1 Sin\theta_1 \dot{\theta}_1^2 - m_2 l_2 Sin\theta_2 \dot{\theta}_2^2 - m_1 g Sin\theta_1 Cos\theta_1 - m_2 g Sin\theta_2 Cos\theta_2}{M + m_1 + m_2 - m_1 Cos^2\theta_1 - m_2 Cos^2\theta_2} \tag{9}$$

Substitute equation (9) into (7) and (8),

$$\ddot{\theta}_1 = \frac{Cos\theta_1}{l_1}\ddot{x} - \frac{g}{l_1} Sin\theta_1 \tag{10}$$

$$\ddot{\theta}_2 = \frac{Cos\theta_2}{l_2}\ddot{x} - \frac{g}{l_2} Sin\theta_2 \tag{11}$$

Substituting the value of $\ddot{x}$ in (10) and (11),

$$\ddot{\theta}_1 = \frac{C\theta_1}{l_1}\left[\frac{F - m_1 l_1 S\theta_1 \dot{\theta}_1^2 - m_2 l_2 S\dot{\theta}_2^2 - m_1 g S\theta_1 C\theta_1 - m_2 g S\theta_2 C\theta_2}{M + m_1 + m_2 - m_1 C^2\theta_1 - m_1 C^2\theta_2}\right] - \frac{g}{l_1} S\theta_1 \tag{12}$$

$$\ddot{\theta}_2 = \frac{C\theta_2}{l_2}\left[\frac{F - m_1 l_1 S\theta_1 \dot{\theta}_1^2 - m_2 l_2 S\dot{\theta}_2^2 - m_1 g S\theta_1 C\theta_1 - m_2 g S\theta_2 C\theta_2}{M + m_1 + m_2 - m_1 C^2\theta_1 - m_1 C^2\theta_2}\right] - \frac{g}{l_2} S\theta_2 \tag{13}$$

Now the non-linear state space representation will be

$$\dot{\vec{x}} = A\vec{x} + B\vec{U}$$

$$\dot{\vec{x}} = \begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta}_1 \\ \ddot{\theta}_1 \\ \dot{\theta}_2 \\ \ddot{\theta}_2 \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \frac{F - m_1 l_1 Sin\theta_1 \dot{\theta}_1^2 - m_2 l_2 Sin\theta_2 \dot{\theta}_2^2 - m_1 g Sin\theta_1 Cos\theta_1 - m_2 g Sin\theta_2 Cos\theta_2}{M + m_1 + m_2 - m_1 Cos^2\theta_1 - m_1 Cos^2\theta_2} \\ \dot{\theta}_1 \\ \frac{C\theta_1}{l_1}\left[\frac{F - m_1 l_1 S\theta_1 \dot{\theta}_1^2 - m_2 l_2 S\dot{\theta}_2^2 - m_1 g S\theta_1 C\theta_1 - m_2 g S\theta_2 C\theta_2}{M + m_1 + m_2 - m_1 C^2\theta_1 - m_1 C^2\theta_2}\right] - \frac{g}{l_1} S\theta_1 \\ \dot{\theta}_2 \\ \frac{C\theta_2}{l_2}\left[\frac{F - m_1 l_1 S\theta_1 \dot{\theta}_1^2 - m_2 l_2 S\dot{\theta}_2^2 - m_1 g S\theta_1 C\theta_1 - m_2 g S\theta_2 C\theta_2}{M + m_1 + m_2 - m_1 C^2\theta_1 - m_1 C^2\theta_2}\right] - \frac{g}{l_2} S\theta_2 \end{bmatrix}$$

Part b] Linearized system around the equilibrium point specified by $\boldsymbol{x} = \mathbf{0}\ \boldsymbol{and}\ \boldsymbol{\theta_1} = \mathbf{0}\ \boldsymbol{and}\ \boldsymbol{\theta_2} = \mathbf{0}$

Using Jacobian Linearization, at the equilibrium point $(x, \theta_1, \theta_2)_{(0,0,0)}$

$$A = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} & \frac{\partial f_1}{\partial x_4} & \frac{\partial f_1}{\partial x_5} & \frac{\partial f_1}{\partial x_6} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3} & \frac{\partial f_2}{\partial x_4} & \frac{\partial f_2}{\partial x_5} & \frac{\partial f_2}{\partial x_6} \\ \frac{\partial f_3}{\partial x_1} & \frac{\partial f_3}{\partial x_2} & \frac{\partial f_3}{\partial x_3} & \frac{\partial f_3}{\partial x_4} & \frac{\partial f_3}{\partial x_5} & \frac{\partial f_3}{\partial x_6} \\ \frac{\partial f_4}{\partial x_1} & \frac{\partial f_4}{\partial x_2} & \frac{\partial f_4}{\partial x_3} & \frac{\partial f_4}{\partial x_4} & \frac{\partial f_4}{\partial x_5} & \frac{\partial f_4}{\partial x_6} \\ \frac{\partial f_5}{\partial x_1} & \frac{\partial f_5}{\partial x_2} & \frac{\partial f_5}{\partial x_3} & \frac{\partial f_5}{\partial x_4} & \frac{\partial f_5}{\partial x_5} & \frac{\partial f_5}{\partial x_6} \\ \frac{\partial f_6}{\partial x_1} & \frac{\partial f_6}{\partial x_2} & \frac{\partial f_6}{\partial x_3} & \frac{\partial f_6}{\partial x_4} & \frac{\partial f_6}{\partial x_5} & \frac{\partial f_6}{\partial x_6} \end{bmatrix}_{(0,0,0)}$$

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta}_1 \\ \ddot{\theta}_1 \\ \dot{\theta}_2 \\ \ddot{\theta}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{-gm_1}{M} & 0 & \frac{-gm_2}{M} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{-g(M+m_1)}{Ml_1} & 0 & \frac{-gm_2}{Ml_1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{-gm_1}{Ml_2} & 0 & \frac{-g(M+m_2)}{Ml_2} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta}_1 \\ \theta_2 \\ \dot{\theta}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{Ml_1} \\ 0 \\ \frac{1}{Ml_2} \end{bmatrix} F$$

Part c] CONTROLLABILITY :

We will obtain the conditions for which the system is controllable. An LTI system is controllable if the controllability matrix obtained has full rank condition.

$$rank = \text{rank}\ [B\ AB\ A^2B\ \ A^3B\ A^4B\ A^5B]$$

Controllability matrix is obtained using MATLAB

MATLAB code:

```
%% Define the state space equation
syms F m1 m2 t1 td1 L2 L1 t2 td2 g M xd x
f1 = xd;
f3 = td1;
f5 = td2;
f2 = (F - m1*L1*sin(t1)*td1^2 - m2*L2*sin(t2)*td2^2 - m1*g*sin(t1)*cos(t1)
- m2*g*sin(t2)*cos(t2))/(M + m1 + m2 - m1*cos(t1)^2 - m2*cos(t2)^2);
f4 = cos(t1)*f2/L1 - g*sin(t1)/L1;
f6 = cos(t2)*f2/L2 - g*sin(t2)/L2;


%% Find the Jacobian to Linearize the system.
%% A, B, C and D matrices are found using jacobian
Ja = jacobian ([f1, f2, f3, f4, f5, f6],[x, xd, t1, td1, t2, td2]);
Jb = jacobian ([f1, f2, f3, f4, f5, f6],[F]);
Jc = jacobian ([x, t1, t2],[x, xd, t1, td1, t2, td2]);
Jd = jacobian ([x, t1, t2], [F]);



%% Find the A, B, C and D matrix around equilibrium point which is (0,0)
A = simplify((subs(Ja , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0])));
B = simplify((subs(Jb , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0])));
C = simplify((subs(Jc , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0])));
D = simplify((subs(Jd , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0])));



%% Find the rank of the controllability matrix of the linearized system
around (0,0)
%% A, B, C and D matrix are symbolically represented during this rank
calculation
E = [B, A*B, A^2*B, A^3*B, A^4*B, A^5*B];
disp("Rank of the controllability matrix of the linearized system: ")
rank(E)



%%Check the condition when the system is not controllable
%% when and L1 = L2, controllability matrix loose its rank, hence system is
not controllable
%% Other than that, system is controllable for all other the conditions
E1 = subs(E, [m1 m2 L1 L2], [100 160 50 50]);
disp("Rank of the controllability matrix of the linearized system when
length are equal: ")
rank(E1)
disp("Since rank is less than 6, system is not controllable. ")
```

MATLAB output:

Rank of the controllability matrix of the linearized system:

ans =

6

Rank of the controllability matrix of the linearized system when length are equal:

ans =

4

Since rank is less than 6, system is not controllable.

Part d) LQR CONTROLLER

CHECKING CONTROLLABILITY:

Take M = 1000 Kg, m1=m2=100 Kg, L1 = 20 m, L2 = 10m. We will check the controllability of the system using MATLAB.

MATLAB code to check the controllability of the LQR controller with given parameter.

MATLAB code:

```
%% check that the system is controllable when M = 1000, m1 = m2 = 100, L1 =
20, L2 = 10
E1 = subs(E, [M m1 m2 L1 L2], [1000 100 100 20 10]);
disp("Rank of the controllability matrix: ")
rank(E1)
```

MATLAB output:

Rank of the controllability matrix:

ans =

6

LQR CONTROLLER IN LINEAR SYSTEM:

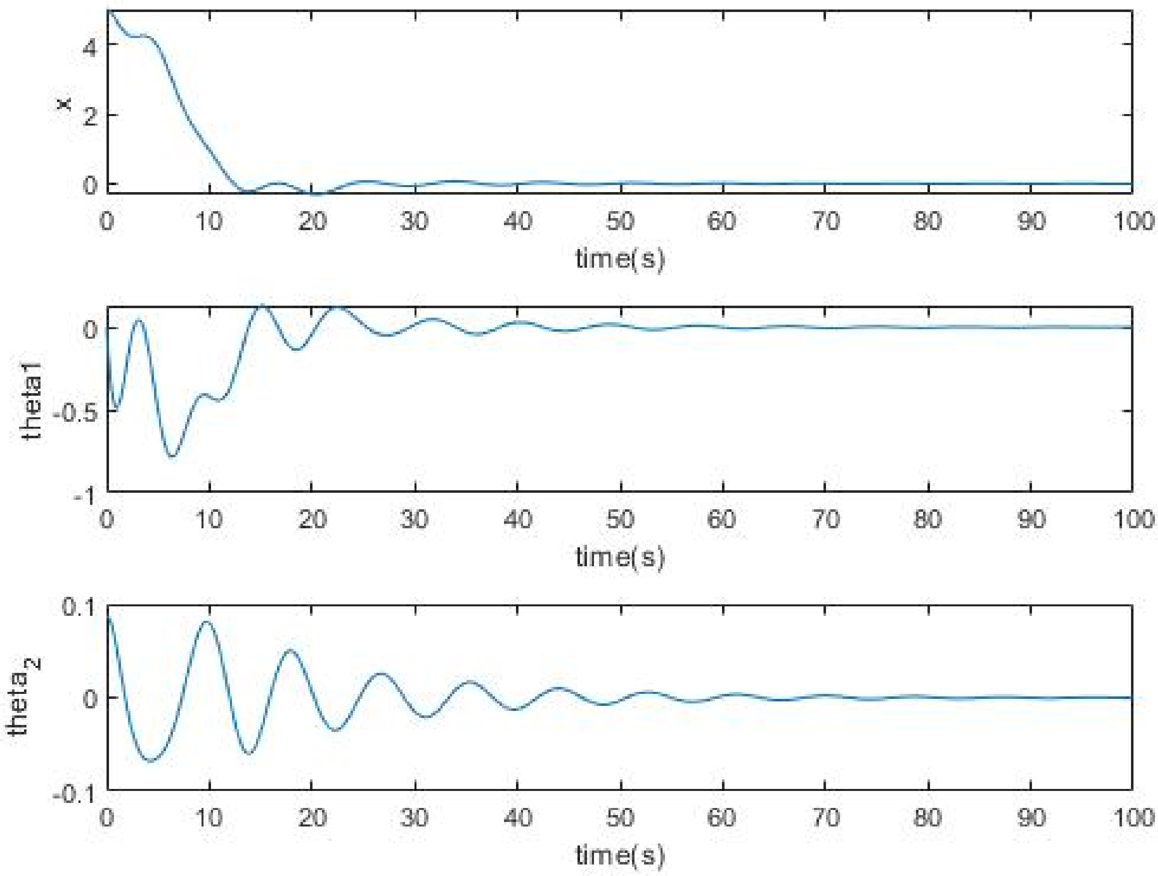
MATLAB code:

```
%% LQR design

%%LQR without gain tuning
Q = eye(6,6);
R = 1;
A0 = double(subs(A,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));
B0 = double(subs(B,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));
C0 = double(subs(C,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));
```

```
D0 = double(subs(D,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));
K1 = lqr(A0, B0, Q, R);

sys = ss((A0-B0*K1),B0, C0, D0);
figure(1)
step(sys)

%% LQR after gain tuning
Q(4,4) = 1000;
Q(6,6) = 1000;
R = 0.0001;
A1 = double(subs(A,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));
B1 = double(subs(B,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));
C1 = double(subs(C,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));
D1 = double(subs(D,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));
K2 = lqr(A1, B1, Q, R);

tspan = 0:0.1:100;
X0 = [5; 0; deg2rad(5); 0; deg2rad(10); 0];  %% intial condition of the
system
Xw = [0; 0; 0; 0; 0; 0];  %% Final condition of the system
u = @(X) -K2*(X - Xw);     %% Control input
[t,X] = ode45(@(t,X)system(X, A1, B1, u(X)), tspan, X0);

title('LQR on the linear controller')
figure(2)
subplot(3,1,1);
plot(t, X(:,1))
xlabel('time(s)');
ylabel('x');
subplot(3,1,2);
plot(t, X(:,2))
xlabel('time(s)');
ylabel('theta1');
subplot(3,1,3);
plot(t, X(:,3))
xlabel('time(s)')
ylabel('theta_2')

function dX = system(X, A1, B1, u)
    dX = A1*X + B1*u;
end
```

MATLAB output:

Linear system's behaviour after applying LQR controller:



LYAPUNOV'S INDIRECT METHOD TO CHECK STABILITY:

To check the stability of the closed loop system using Lyapunov's indirect method, we have written MATLAB code shown below:

MATLAB code:

```
%% Lyapunov indirect method to certify stability of the close loop system
%% Eigen value of the close loop system is calculated
%% since all of the eigen value has a negative real part, system is stable
Ac = A1 - B1*K2;
disp("eigen value of the closed loop matrix:")
ei = eig(Ac)
disp("since all of the eigen value has negative real part, system is
controllable")
```

MATLAB output:

eigen value of the closed loop matrix:

ei =

-0.2062 + 0.2031i

-0.2062 - 0.2031i

-0.1654 + 1.0277i

-0.1654 - 1.0277i

-0.0592 + 0.7249i

-0.0592 - 0.7249i

since all of the eigen value has negative real part, system is controllable

Part e] OBSERVABILITY OF LINEAR SYSTEM:

To check the observability of the system when the output vector is Observability $x(t), (\theta_1(t), \theta_2(t)), (x(t)\theta_2(t))\ or\ (x(t), \theta_1(t), \theta_2(t)$, we have written a MATLAB code as shown below:

MATLAB code:

```
%%check the observality of the system when output is x
Jc2 = jacobian ([x, 0, 0],[x, xd, t1, td1, t2, td2]);
C2 = double(simplify((subs(Jc2 , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0]))))
O1 = [C2 ; C2*A ; C2*A^2 ; C2*A^3 ; C2*A^4 ; C2*A^5];
disp("Rank of the observability matrix when output vector is x : ")
rank(O1)
disp("since the rank is 6, system is observable.")

%%check the observality of the system when output is t1 and t2
Jc3 = jacobian ([0, t1, t2],[x, xd, t1, td1, t2, td2]);
C3 = simplify((subs(Jc3 , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0])));
O2 = [C3 ; C3*A ; C3*A^2 ; C3*A^3 ; C3*A^4 ; C3*A^5];
disp("Rank of the observability matrix when output vector is theta1 and
theta2")
rank(O2)
disp("since the rank is less than 6, system is not observable.")

%%check the observality of the system when output is x and t2
Jc3 = jacobian ([x, 0, t2],[x, xd, t1, td1, t2, td2]);
C3 = double(simplify((subs(Jc3 , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0]))));
O3 = [C3 ; C3*A ; C3*A^2 ; C3*A^3 ; C3*A^4 ; C3*A^5];
disp("Rank of the observability matrix when output vector is x and theta2")
rank(O3)
disp("since the rank is 6, system is observable.")

%%check the observality of the system when output is x t1 and t2
Jc4 = jacobian ([x, t1, t2],[x, xd, t1, td1, t2, td2]);
C4 = double(simplify((subs(Jc4 , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0]))));
```

```
O4 = [C4 ; C4*A ; C4*A^2 ; C4*A^3 ; C4*A^4 ; C4*A^5];
disp("Rank of the observability matrix when output vector is x theta1 and
theta2")
rank(O4)
disp("since the rank is 6, system is observable.")
```

MATLAB output:

Rank of the observability matrix when output vector is x :

ans =

6

since the rank is 6, system is observable.

Rank of the observability matrix when output vector is theta1 and theta2

ans =

4

since the rank is less than 6, system is not observable.

Rank of the observability matrix when output vector is x and theta2

ans =

6

since the rank is 6, system is observable.

Rank of the observability matrix when output vector is x theta1 and theta2

ans =

6

since the rank is 6, system is observable.

Part f) Leunberger Observer:

We have simulated the response of the “best” Luenberger observer for each one of the output vectors for which the system is observable and simulated the response to initial condition and unit step input. The MATLAB code and output is as below:

MATLAB code:

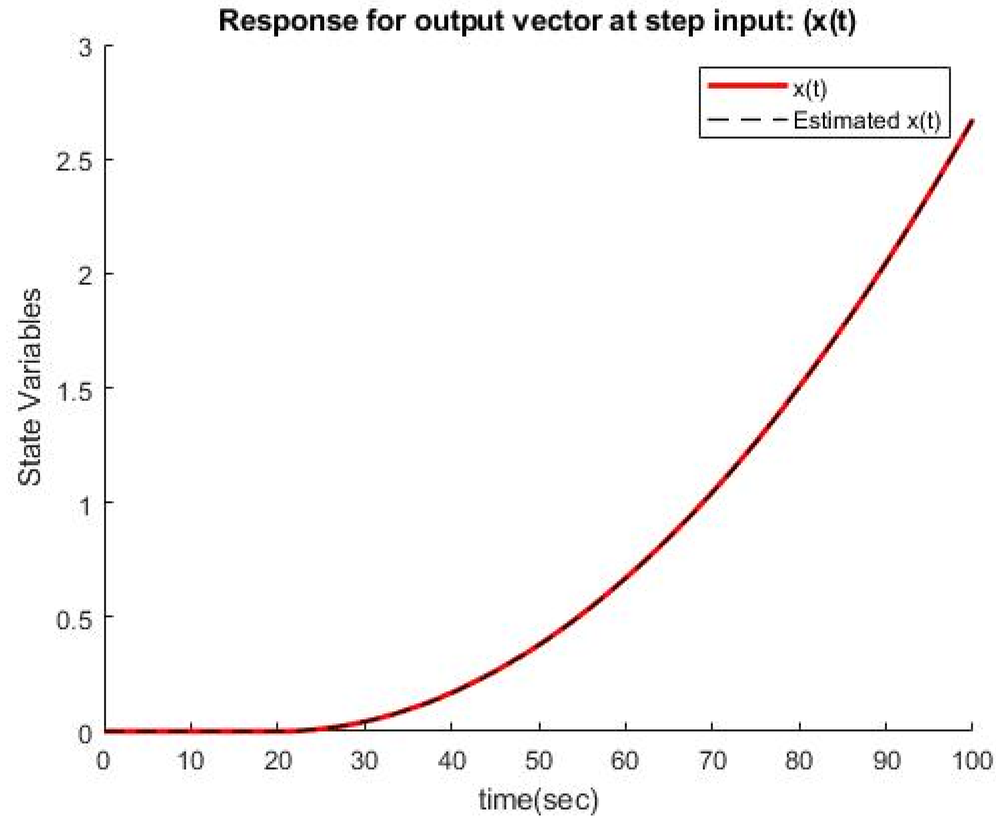
```
%% Define the state space equation
syms F m1 m2 t1 td1 L2 L1 t2 td2 g M xd x
f1 = xd;
f3 = td1;
f5 = td2;
```

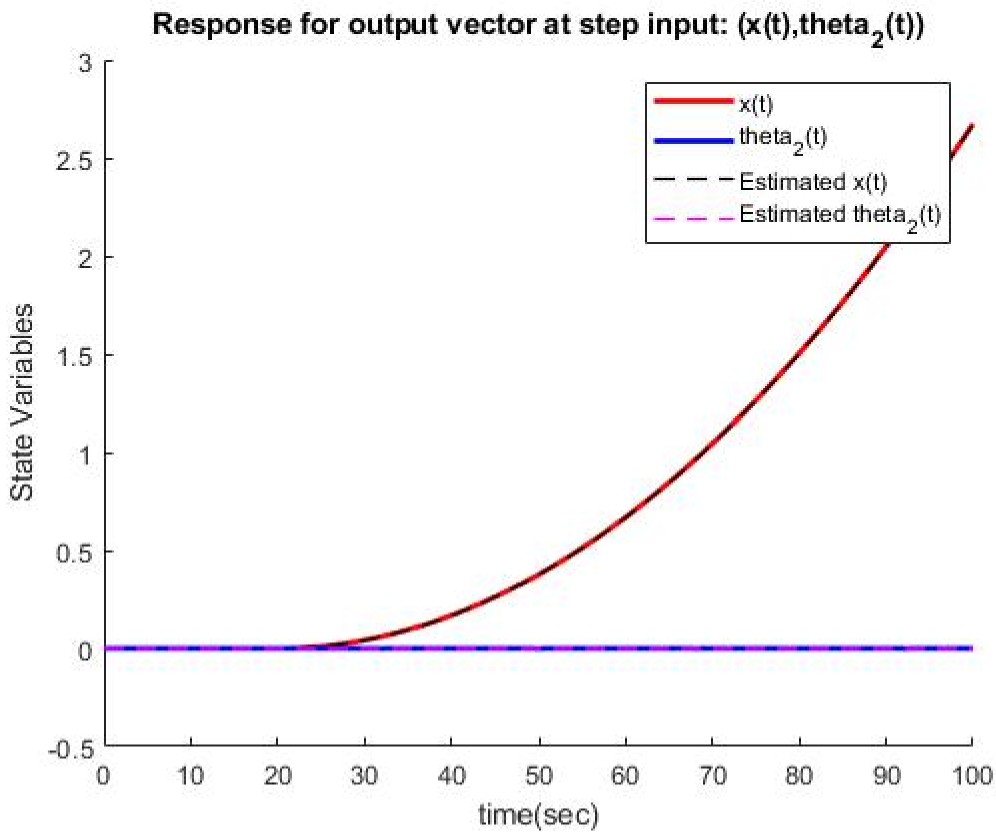
```
f2 = (F - m1*L1*sin(t1)*td1^2 - m2*L2*sin(t2)*td2^2 - m1*g*sin(t1)*cos(t1)
- m2*g*sin(t2)*cos(t2))/(M + m1 + m2 - m1*cos(t1)^2 - m2*cos(t2)^2);
f4 = cos(t1)*f2/L1 - g*sin(t1)/L1;
f6 = cos(t2)*f2/L2 - g*sin(t2)/L2;
X0 = [5; 0; deg2rad(5); 0; deg2rad(10); 0]

%% Find the Jacobian to Linearize the system.
%% A, B, C and D matrices are found using jacobian
Ja = jacobian ([f1, f2, f3, f4, f5, f6],[x, xd, t1, td1, t2, td2]);
Jb = jacobian ([f1, f2, f3, f4, f5, f6],[F]);
Jc = jacobian ([x, t1, t2],[x, xd, t1, td1, t2, td2]);
Jd = jacobian ([x, t1, t2], [F]);



%% Find the A, B, C and D matrix around equilibrium point which is (0,0)
A = simplify((subs(Ja , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0])))
B = simplify((subs(Jb , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0])));
C = simplify((subs(Jc , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0])));
D = simplify((subs(Jd , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0])));



%%
Q = eye(6,6);
R = 1;
A0 = double(subs(A,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));
B0 = double(subs(B,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));
C0 = double(subs(C,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));
D0 = double(subs(D,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));
K1 = lqr(A0, B0, Q, R);

%%check the observality of the system when output is x
Jc2 = jacobian ([x, 0, 0],[x, xd, t1, td1, t2, td2]);
C2 = double(simplify((subs(Jc2 , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0]))))

%%check the observality of the system when output is x and t2
Jc3 = jacobian ([x, 0, t2],[x, xd, t1, td1, t2, td2]);
C3 = double(simplify((subs(Jc3 , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0]))));

%%check the observality of the system when output is x t1 and t2
Jc4 = jacobian ([x, t1, t2],[x, xd, t1, td1, t2, td2]);
C4 = double(simplify((subs(Jc4 , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0]))));

sys2 = ss(A0, B0, C2, D0);
sys3 = ss(A0, B0, C3, D0);
sys4 = ss(A0, B0, C4, D0);


%% kalman estimation
Bd = 0.1*eye(6,6);
Vt = 0.01*eye(3,3);
[L2,P, E] = lqe(A0, Bd, C2, Bd,Vt);
[L3,P, E] = lqe(A0, Bd, C3, Bd,Vt);
[L4,P, E] = lqe(A0, Bd, C4, Bd,Vt);

Ac2 = A0 - L2*C2;
Ac3 = A0 - L3*C3;
Ac4 = A0 - L4*C4;
```

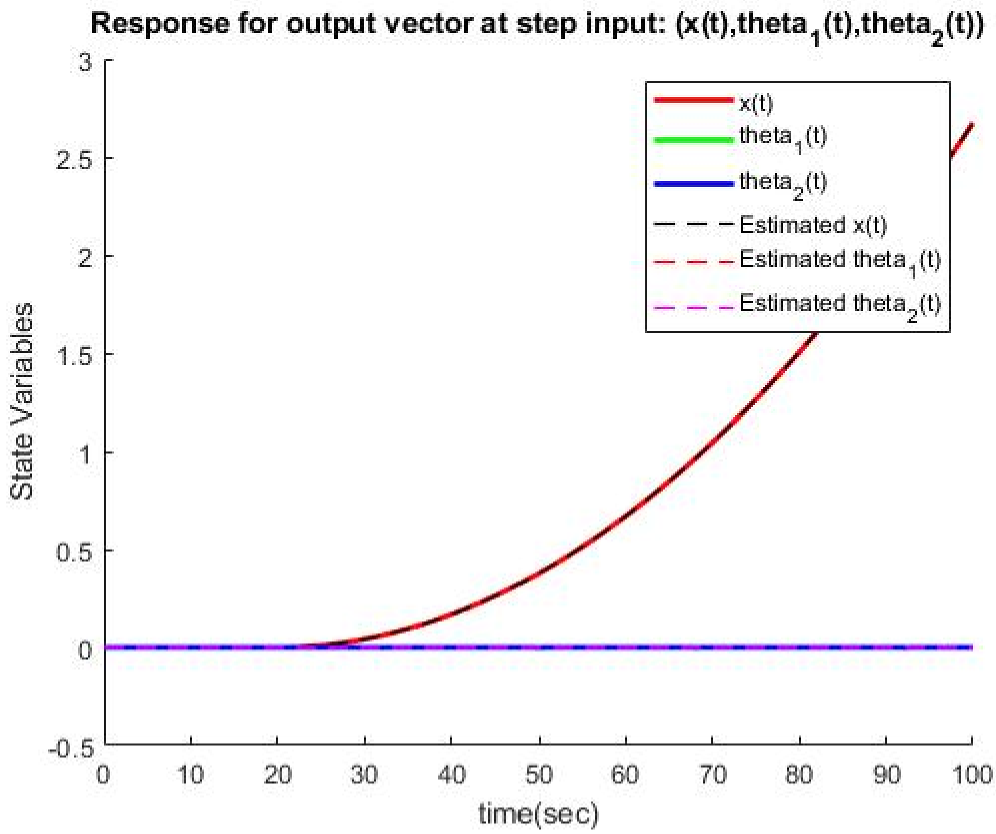
```
e_sys2 = ss(Ac2, [B0 L2], C2, 0);
e_sys3 = ss(Ac3, [B0 L3], C3, 0);
e_sys4 = ss(Ac4, [B0 L4], C4, 0);
%% step input response
tspan = 0:0.1:100;
unitStep = 0*tspan;
unitStep(200:length(tspan)) = 1;

[Y2,t] = lsim(sys2,unitStep, tspan);
[X2,t] = lsim(e_sys2,[unitStep;Y2'],tspan);

[Y3,t] = lsim(sys3,unitStep, tspan);
[X3,t] = lsim(e_sys3,[unitStep;Y3'],tspan);

[Y4,t] = lsim(sys4,unitStep, tspan);
[X4,t] = lsim(e_sys2,[unitStep;Y4'],tspan);

figure();
hold on
plot(t,Y2(:,1),'r','Linewidth',2)
plot(t,X2(:,1),'k--','Linewidth',1)
ylabel('State Variables')
xlabel('time(sec)')
legend('x(t)','Estimated x(t)')
title('Response for output vector at step input: (x(t)')
hold off

figure();
hold on
plot(t,Y3(:,1),'r','Linewidth',2)
plot(t,Y3(:,3),'b','Linewidth',2)
plot(t,X3(:,1),'k--','Linewidth',1)
plot(t,X3(:,3),'m--','Linewidth',1)
ylabel('State Variables')
xlabel('time(sec)')
legend('x(t)','theta_2(t)','Estimated x(t)','Estimated theta_2(t)')
title('Response for output vector at step input: (x(t),theta_2(t))')
hold off

figure();
hold on
plot(t,Y4(:,1),'r','Linewidth',2)
plot(t,Y4(:,2),'g','Linewidth',2)
plot(t,Y4(:,3),'b','Linewidth',2)
plot(t,Y4(:,1),'k--','Linewidth',1)
plot(t,X4(:,2),'r--','Linewidth',1)
plot(t,X4(:,3),'m--','Linewidth',1)
ylabel('State Variables')
xlabel('time(sec)')
legend('x(t)','theta_1(t)','theta_2(t)','Estimated x(t)','Estimated
theta_1(t)','Estimated theta_2(t)')
title('Response for output vector at step input:
(x(t),theta_1(t),theta_2(t))')
hold off

[t,x2] = ode45(@(t,x)linear2(t, x  ,L2, A0, B0, C2),tspan,X0);
figure();
hold on
plot(t,x2(:,1))
```

```
ylabel('state variables')
xlabel('time (sec)')
title('Linear system Observer when output vector: x(t)')
legend('x')
hold off

[t,x3] = ode45(@(t,x)linear3(t, x, L3, A0, B0, C3),tspan,X0);
figure();
hold on
plot(t,x3(:,1))
plot(t,x3(:,5))
ylabel('state variables')
xlabel('time (s)')
title('Linear system Observer when output vector: (x(t),theta_2(t))')
legend('x','theta_2')
hold off

[t,x4] = ode45(@(t,x)linear4(t, x, L4, A0, B0, C4),tspan,X0);
figure();
hold on
plot(t,x4(:,1))
plot(t,x4(:,3))
plot(t,x4(:,5))
ylabel('state variables')
xlabel('time (s)')
title('Linear system Observer when output vector:
(x(t),theta_1(t),theta_2(t))')
legend('x','theta_1','theta_2')
hold off

function dX = linear2(t,x,L,A,B,C)
    y = [x(1); 0; 0];
    K = 1;
    dX = (A+B*K)*x + L*(y - C*x);
end

function dX = linear3(t,x,L,A,B,C)
    y = [x(1); 0; x(5)];
    K = 1;
    dX = (A+B*K)*x + L*(y - C*x);
end
function dX = linear4(t,x,L,A,B,C)
    y = [x(1); x(3); x(5)];
    K = 1;
    dX = (A+B*K)*x + L*(y - C*x);
end
```

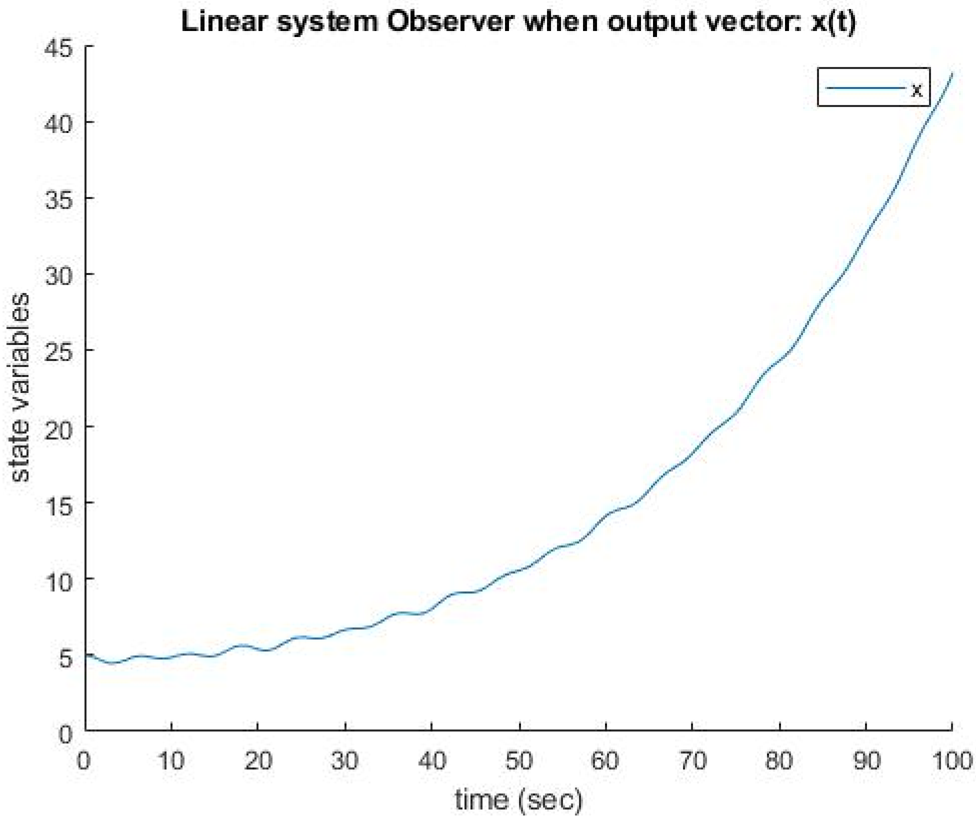
MATLAB output:



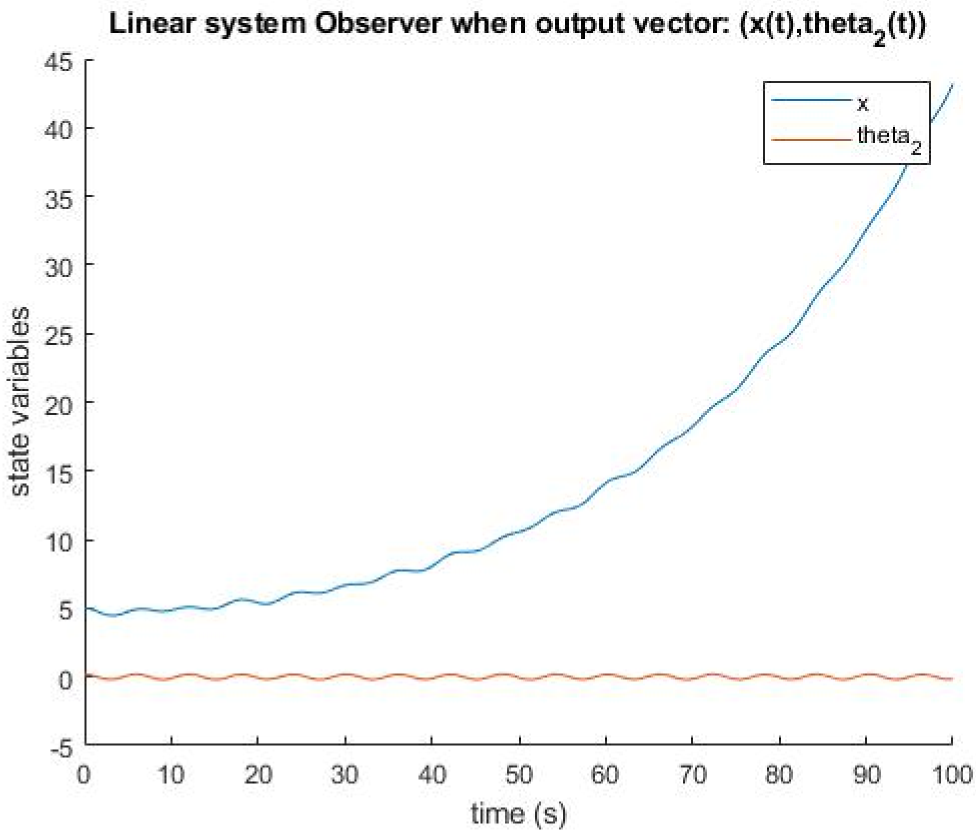


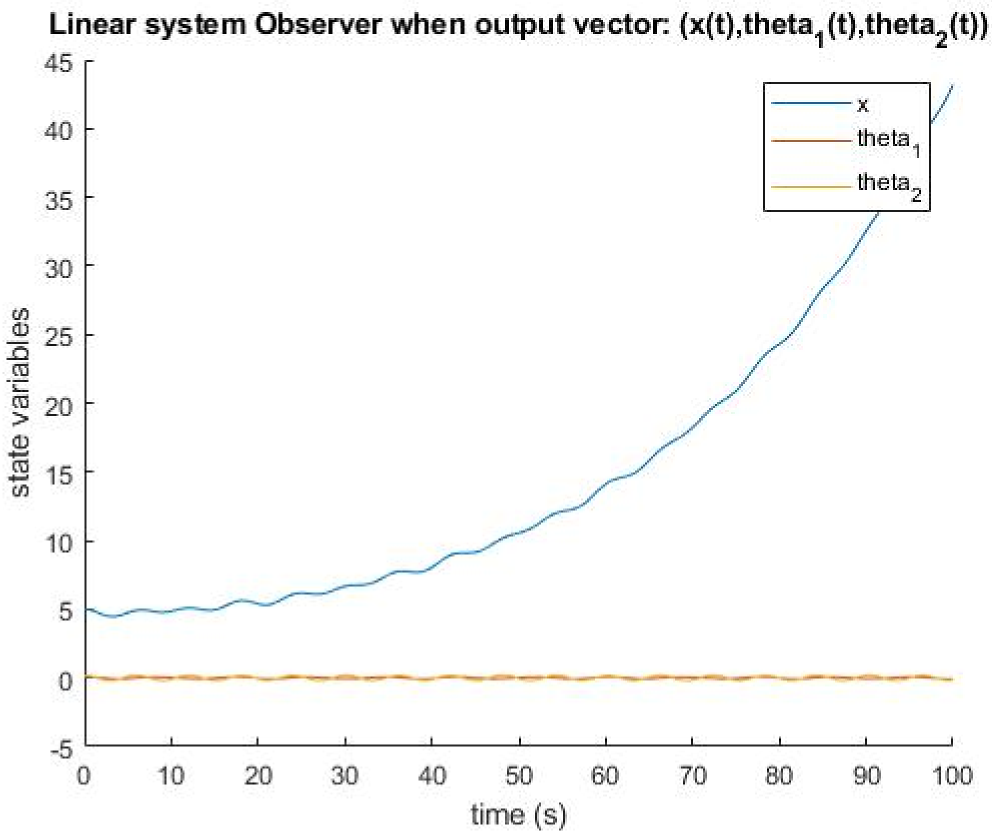


Response for output vector at step input: (x(t),theta_1(t),theta_2(t))
x(t)
theta_1(t)
theta_2(t)
Estimated x(t)
Estimated theta_1(t)
Estimated theta_2(t)
State Variables
time(sec)



Linear system Observer when output vector: x(t)
x
state variables
time (sec)


Linear system Observer when output vector: (x(t),theta2(t))
x
theta2
state variables
time (s)
45
40
35
30
25
20
15
10
5
0
-5
0
10
20
30
40
50
60
70
80
90
100


Linear system Observer when output vector: (x(t),theta1(t),theta2(t))
x
theta1
theta2
state variables
time (s)
45
40
35
30
25
20
15
10
5
0
-5
0
10
20
30
40
50
60
70
80
90
100

Part g) LQG Controller:

With R = 0.3, Q = 1 and noise Bd = 0.5 and Vn = 0.05, we are getting following output.

Question: How would you reconfigure your controller to asymptotically track a constant reference on x ?
Answer : We will reconfigure our controller by providing a desired X and tune LQR controller to get the feedback according to the desired output.

Question: Will your design reject constant force disturbances applied on the cart ?
Answer : Yes. LQR controller is strong enough to stabilise the position of cart x(t), if disturbance noise is increase in the Kalman Filter.

MATLAB code:

```
%% Define the state space equation
syms F m1 m2 t1 td1 L2 L1 t2 td2 g M xd x
f1 = xd;
f3 = td1;
f5 = td2;
f2 = (F - m1*L1*sin(t1)*td1^2 - m2*L2*sin(t2)*td2^2 - m1*g*sin(t1)*cos(t1)
- m2*g*sin(t2)*cos(t2))/(M + m1 + m2 - m1*cos(t1)^2 - m2*cos(t2)^2);
f4 = cos(t1)*f2/L1 - g*sin(t1)/L1;
f6 = cos(t2)*f2/L2 - g*sin(t2)/L2;
tspan = 0:0.1:100;
q0 = [5 0 deg2rad(0) 0 deg2rad(0) 0];


%% Find the Jacobian to Linearize the system.
%% A, B, C and D matrices are found using jacobian
Ja = jacobian ([f1, f2, f3, f4, f5, f6],[x, xd, t1, td1, t2, td2]);
Jb = jacobian ([f1, f2, f3, f4, f5, f6],[F]);
Jc = jacobian ([x, t1, t2],[x, xd, t1, td1, t2, td2]);
Jd = jacobian ([x, t1, t2], [F]);



%% Find the A, B, C and D matrix around equilibrium point which is (0,0)
A = simplify((subs(Ja , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0])))
B = simplify((subs(Jb , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0])));
C = simplify((subs(Jc , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0])));
D = simplify((subs(Jd , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0])));

%% Linearized Model
A0 = double(subs(A,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));
B0 = double(subs(B,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));
C0 = double(subs(C,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));
D0 = double(subs(D,[M m1 m2 L1 L2 g], [1000 100 100 20 10 9.8]));


%%check the observality of the system when output is x
Jc2 = jacobian ([x, 0, 0],[x, xd, t1, td1, t2, td2]);
C2 = double(simplify((subs(Jc2 , [t1 td1 t2 td2 xd x], [0 0 0 0 0 0]))))
```

```
sys2 = ss(A0, B0, C2, D0);

Q = [1 0 0 0 0 0; 0 0 0 0 0 0; 0 0 0 0 0 0; 0 0 0 0 0 0; 0 0 0 0 0 0; 0 0 0
0 0 0];
R = 0.3;
[K1,S,P] = lqr(A0, B0, Q, R);

sys = ss(A0-B0*K1,B0,C2,D0);

%% Kalman Estimator Design
Bd = 0.5*eye(6);                %Process Noise
Vn = 0.05;                      %Measurement Noise
[L2,P,E] = lqe(A0,Bd,C2,Bd,Vn*eye(3)); %Considering vector output: x(t)
Ac1 = A0-(L2*C2);
e_sys1 = ss(Ac1,[B0 L2],C2,0);

%% Non-linear Model LQG Response
[t,q1] = ode45(@(t,q)nonLinear(t,q,-K1*q,L2),tspan,q0);
figure();
hold on
plot(t,q1(:,1))
ylabel('state variable')
xlabel('time (sec)')
title('Non-Linear System LQG for output vector: x(t)')
legend('x')
hold off

function dQ = nonLinear(t,y,F,Lue1)
    m1 = 100; m2 = 100; M = 1000; L1 = 20; L2 = 10; g = 9.81;
    x = y(1);
    dx = y(2);
    t1 = y(3);
    dt1 = y(4);
    t2 = y(5);
    dt2 = y(6);
    dQ=zeros(6,1);
    y1 = [x; 0; 0];
    c1 = [1 0 0 0 0 0; 0 0 0 0 0 0; 0 0 0 0 0 0];
    sum = Lue1*(y1-c1*y);
    dQ(1) = dx + sum(1);
    dQ(2) = (F-((m1*sin(t1)*cos(t1))+(m2*sin(t2)*cos(t2)))*g -
(L1*m1*(dQ(3)^2)*sin(t1)) - (L2*m2*(dQ(5)^2)*sin(t2)))/(m1+m2+M-
(m1*(cos(t1)^2))-(m2*(cos(t2)^2)))+sum(2);
    dQ(3) = dt1+sum(3);
    dQ(4) = ((cos(t1)*dQ(2)-g*sin(t1))/L1) + sum(4);
    dQ(5) = dt2 + sum(5);
    dQ(6) = (cos(t2)*dQ(2)-g*sin(t2))/L2 + sum(6);
end
```

MATLAB output: