ENPM677 Final Project Report

Mushty Sri Sai Kaushik - 116917094

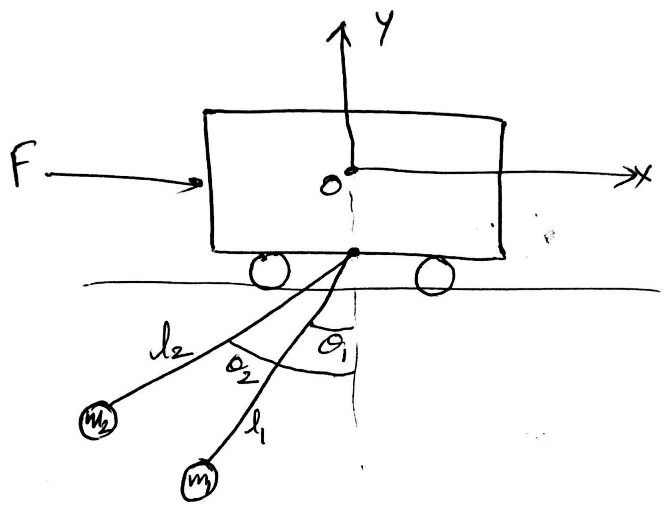
Nishanth Vanipenta - 116409605

December 16, 2019

Contents

a Equations of motion for Nonlinear State Representation

In order to obtain the dynamics of the above system, we need to obtain the states of system which are linear velocity, linear acceleration of the crane along with, angular velocity and angular acceleration of the masses of the pendulum m_1 and m_2.



We consider (X,Y) as the origin of the reference frame as shown in the system and the systems are modeled with the same consideration.

Position of m_1 w.r.t the reference frame is given by

$$x_1 = (x - (l_1)sin(\theta_1))X + (l_1)cos(\theta_1)Y \tag{1}$$

And position of m_1 w.r.t the reference frame is given by

$$x_2 = (x - (l_2)sin(\theta_2))X + (l_2)cos(\theta_2)Y \tag{2}$$

To obtain the total energy of the system, we get the Kinetic energy as follows:

$$K.E = 1/2[M(dx/dt)^2] \tag{3}$$

Now, from Eq (3), we get the equations of Kinetic energy as follows for the individual masses,

$$(K.E)_1 = 1/2[M((\dot{x} - l_1\dot{\theta}_1 cos(\theta_1)^2 + (-l_1\dot{\theta}_1 sin(\theta_1)^2))] \tag{4}$$

$$(K.E)_1 = 1/2[M(\dot{x}^2 + l_1^2\dot{\theta_1}^2 cos^2(\theta_1) - 2\dot{x}l_1\dot{\theta}_1 cos(\theta_1) + l_1^2\dot{\theta_1}^2 sin^2(\theta_1))] \tag{5}$$

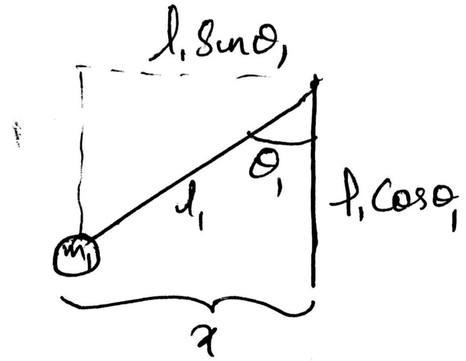
On solving,

$$(K.E)_1 = 1/2[M(\dot{x}^2 + l_1^2\dot{\theta_1}^2 - 2\dot{x}l_1\dot{\theta}_1 cos(\theta_1))] \tag{6}$$

Similarly, solving for $(K.E)_2$, we get

$$(K.E)_2 = 1/2[M(\dot{x}^2 + l_1^2\dot{\theta_2}^2 - 2\dot{x}l_2\dot{\theta_2}cos(\theta_2))] \tag{7}$$

Now, for total energy, we require Potential energy for the same.



$$(P.E)_1 = m_1gl_1 - m_1gl_1(cos(\theta_1)) \tag{8}$$

$$\Rightarrow (P.E)_1 = m_1gl_1(1 - (cos(\theta_1)) \tag{9}$$

And,

$$(P.E)_2 = m_2gl_2(1 - (cos(\theta_2)) \tag{10}$$

Now, from Euler-Lagrange Equation, we know that

$$L = K.E - P.E \tag{11}$$

Here,

$$K.E = (K.E)_{crane} + (K.E)_1 + (K.E)_2 \tag{12}$$

Similarly,

$$P.E = (P.E)_{crane} + (P.E)_1 + (P.E)_2 \tag{13}$$

$$(K.E)_{crane} = 1/2[M(\dot{x})^2] \tag{14}$$

$$(K.E)_1 = 1/2[M(\dot{x}^2 + l_1^2\dot{\theta_1}^2 - 2\dot{x}l_1\dot{\theta_1}cos(\theta_1))] \tag{15}$$

$$(K.E)_2 = 1/2[M(\dot{x}^2 + l_1^2\dot{\theta_2}^2 - 2\dot{x}l_2\dot{\theta_2}cos(\theta_2))] \tag{16}$$

And,

$$(P.E)_{crane} = 0 \tag{17}$$

Because the reference height is zero, we get h = 0.

$$(P.E)_1 = m_1gl_1(1 - (cos(\theta_1)) \tag{18}$$

And,

$$(P.E)_2 = m_2 g l_2 (1 - (cos(\theta_2)) \tag{19}$$

From Eq (11), we get,

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

From Euler-Lagrange Equation, we know that,

$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{q}}) - \frac{\partial L}{\partial q} \tag{20}$$

q = state variables

In our case, $q = x, \theta_1, \theta_2$

$$F = x\frac{d}{dt}(\frac{\partial L}{\partial \dot{x}}) - \frac{\partial L}{\partial x} \tag{21}$$

$$0 = x - \frac{\partial L}{\partial \theta_1} \tag{22}$$

$$0 = x\frac{d}{dt}(\frac{\partial L}{\partial \dot{\theta_2}}) - \frac{\partial L}{\partial \theta_2} \tag{23}$$

Now,

$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{x}}) = \ddot{x}(M + m_1 + m_2) - (m_1 l_1 \ddot{\theta_1} cos(\theta_1) + m_2 l_2 \ddot{\theta_1} cos(\theta_2) + m_1 l_1 \dot{\theta_1}^2 sin\theta_1 + m_2 l_2 \dot{\theta_2}^2 sin\theta_2 \tag{24}$$

And Thus, since there are not x terms in Eq (20), we get

$$F = \frac{d}{dt}(\frac{\partial L}{\partial \dot{x}}) = \ddot{x}(M + m_1 + m_2) - (m_1 l_1 \ddot{\theta_1} cos(\theta_1) + m_2 l_2 \ddot{\theta_1} cos(\theta_2) + m_1 l_1 \dot{\theta_1}^2 sin\theta_1 + m_2 l_2 \dot{\theta_2}^2 sin\theta_2 \tag{25}$$

Similarly,

$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{\theta_1}}) = -m_1 \ddot{x} l_1 cos(\theta_1) + m_1 l_1^2 \ddot{\theta_1} + m_1 \dot{x} l_1 \dot{\theta_1} sin(\theta_1) \tag{26}$$

And,

$$\frac{\partial L}{\partial \theta_1} = m_1 \dot{x} l_1 \dot{\theta_1} sin(\theta_1) - m_1 g l_1 sin(\theta_1) \tag{27}$$

Thus,

$$0 = \frac{d}{dt}(\frac{\partial L}{\partial \dot{\theta_1}} - \frac{\partial L}{\partial \theta_1}) = -m_1 \ddot{x} l_1 cos(\theta_1) + m_1 l_1^2 \ddot{\theta_1} + m_1 g l_1 sin(\theta_1) \tag{28}$$

We can do the same in the case of θ_2 as follows :

$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{\theta_2}}) = -m_2 \ddot{x} l_2 cos(\theta_2) + m_2 l_2^2 \ddot{\theta_2} + m_2 \dot{x} l_2 \dot{\theta_2} sin(\theta_2) \tag{29}$$

And,

$$\frac{\partial L}{\partial \theta_2} = m_2 \dot{x} l_2 \dot{\theta_2} sin(\theta_2) - m_2 g l_2 sin(\theta_2) \tag{30}$$

Thus,

$$0 = \frac{d}{dt}(\frac{\partial L}{\partial \dot{\theta_2}} - \frac{\partial L}{\partial \theta_2}) = -m_2\ddot{x}l_2cos(\theta_1) + m_2l_2^2\ddot{\theta_2} + m_2gl_2sin(\theta_2) \tag{31}$$

On equating Eq (28) and (31) and solving further,

$$\ddot{\theta_1} = \frac{m_1\ddot{x}l_1cos(\theta_1) - m_1gl_1sin(\theta_1)}{m_1l_1^2} \tag{32}$$

$$\ddot{\theta_2} = \frac{m_2\ddot{x}l_2cos(\theta_2) - m_2gl_2sin(\theta_2)}{m_2l_2^2} \tag{33}$$

On substituting in Eq(25), we can equate to get $\ddot{x}$

$$\ddot{x} \ = \ \frac{F - m_1gsin\theta_1cos\theta_1 - m_2gsin\theta_2cos\theta_2 - m_1l_1sin\theta_1\dot{\theta_1}^2 - m_2l_2sin\theta_2\dot{\theta_2}^2}{M + m_1 + m_2 - m_1cos^2\theta_1 - m_2cos^2\theta_2} \tag{34}$$

Substituting Eq (34) in Eqs. (32) and (33)

$$\ddot{\theta_1} = \frac{m_1(\frac{F - m_1gsin\theta_1cos\theta_1 - m_2gsin\theta_2cos\theta_2 - m_1l_1sin\theta_1\dot{\theta_1}^2 - m_2l_2sin\theta_2\dot{\theta_2}^2}{M + m_1 + m_2 - m_1cos^2\theta_1 - m_2cos^2\theta_2})l_1cos\theta_1 - m_1gl_1sin\theta_1}{m_1l_1^2} \tag{35}$$

$$\ddot{\theta_1} = \frac{F - m_1gsin\theta_1cos\theta_1 - m_2gsin\theta_2cos\theta_2 - m_1l_1sin\theta_1\dot{\theta_1}^2 - m_2l_2sin\theta_2\dot{\theta_2}^2}{l_2(M + m_1 + m_2 - m_1cos^2\theta_1 - m_2cos^2\theta_2)}cos\theta_2 - \frac{gsin\theta_2}{l_2} \tag{36}$$

On simplifying,

$$\ddot{\theta_1} = \frac{Fcos\theta_2 - m_1gsin\theta_1cos\theta_1cos\theta_2 - m_2gsin\theta_2cos^2\theta_2 - m_1l_1sin\theta_1cos\theta_2\dot{\theta_1}^2 - m_2l_2sin\theta_2cos\theta_2\dot{\theta_2}^2}{l_2(M + m_1sin^2\theta_1 + m_2sin^2\theta_2)} - \frac{gsin\theta_2}{l_2} \tag{37}$$

and for $\ddot{\theta_2}$,

$$\ddot{\theta_2} = \frac{Fcos\theta_1 - m_2gsin\theta_2cos\theta_1cos\theta_2 - m_1gsin\theta_1cos^2\theta_1 - m_1l_1sin\theta_1cos\theta_1\dot{\theta_1}^2 - m_2l_2sin\theta_2cos\theta_1\dot{\theta_2}^2}{l_1(M + m_1sin^2\theta_1 + m_2sin^2\theta_2)} - \frac{gsin\theta_1}{l_1} \tag{38}$$

b Linearization of the state-space representation of the system

We can achieve linearization of the system from the values we have obtained by considering the conditions of the system as:

$$sin(\theta) \approx \theta \tag{39}$$

$$cos(\theta) \approx 1 \tag{40}$$

$$sin^2(\theta) \approx 0 \tag{41}$$

$$cos^2(\theta) \approx 1 \tag{42}$$

Substituting the above in Eq. (34), we get,

$$\ddot{x} = \frac{F - m_1 g\theta_1 - m_2 g\theta_2}{M + m_1 + m_2 - m_1 - m_2} \tag{43}$$

$$\Rightarrow \ddot{x} = \frac{F - m_1 g\theta_1 - m_2 g\theta_2}{M} \tag{44}$$

Similarly for θ_1 and θ_2

$$\ddot{\theta_1} = \frac{F - m_1 g\theta_1 - m_2 g\theta_2}{M l_2} - \frac{g\theta_2}{l_2} \tag{45}$$

$$\Rightarrow \ddot{\theta_1} = \frac{F - m_1 g\theta_1 - m_2 g\theta_2 - Mg\theta_2}{M l_2} \tag{46}$$

And,

$$\ddot{\theta_2} = \frac{F - m_1 g\theta_1 - m_2 g\theta_2 - Mg\theta_1}{M l_1} \tag{47}$$

We consider the linear state space equations to be of the form

$$\dot{x_1} = Ax_1 + Bu \tag{48}$$

$$\dot{y_1} = Cx_1 + Du \tag{49}$$

Where u is given by F

Now, we can define the state variables as follows

$$x_1 = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta_1} \\ \theta_2 \\ \dot{\theta_2} \end{bmatrix}$$

On differentiating, we get,

$$\dot{x_1} = \begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \\ \dot{x_4} \\ \dot{x_5} \\ \dot{x_6} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta_1} \\ \ddot{\theta_1} \\ \dot{\theta_2} \\ \ddot{\theta_2} \end{bmatrix}$$

Now, substituting the values from Eqs. (44), (46) and (47)

$$\dot{x_1} = \begin{bmatrix} \dot{x} \\ \frac{F - m_1 g \theta_1 - m_2 g \theta_2}{M} \\ \dot{\theta}_1 \\ \frac{F - m_1 g \theta_1 - m_2 g \theta_2 - M g \theta_2}{M l_2} \\ \dot{\theta}_2 \\ \frac{F - m_1 g \theta_1 - m_2 g \theta_2 - M g \theta_1}{M l_1} \end{bmatrix}$$

Substituting in Eqs. (48) and (49),

$$\dot{x_1} = \begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta}_1 \\ \ddot{\theta}_1 \\ \dot{\theta}_2 \\ \ddot{\theta}_2 \end{bmatrix} = \mathrm{A} \begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta}_1 \\ \theta_2 \\ \dot{\theta}_2 \end{bmatrix} + B\ F$$

We can now calculate A and B as,

$$\dot{x_1} = \begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta}_1 \\ \ddot{\theta}_1 \\ \dot{\theta}_2 \\ \ddot{\theta}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -m_1 g/M & 0 & -g m_2/M & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -g(m_1+M)/Ml_1 & 0 & -g m_2/Ml_1 & \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -g m_1/Ml_2 & 0 & -g(m_2+M)/Ml_2 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta}_1 \\ \theta_2 \\ \dot{\theta}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1/M \\ 0 \\ 1/Ml_1 \\ 0 \\ 1/Ml_2 \end{bmatrix} F$$

Similarly, we get,

$$\mathrm{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \text{and } \mathrm{D} = 0$$

c Conditions for the Linear system to be Controllable

In order to test the controllability of the system,[1] we calculate the rank of 'C' Matrix. This is given by the following:

C $= \begin{bmatrix} B & AB & A^2B & A^3B & A^4B & A^5B \end{bmatrix}$

$\Rightarrow$Rank[C] = Rank $\begin{bmatrix} B & AB & A^2B & A^3B & A^4B & A^5B \end{bmatrix}$ We can calculate the value of C from the A and B matrices we have obtained previously using MATLAB and the resultant output script for the same is found to be:

From here, we can calculate the determinant of the matrix to be as follows:

$$det[C] = \frac{-(g^6{l_1}^2 - 2l_1l_2g^6 + g^6{l_2}^2)}{M^6{l_1}^6{l_2}^6} \tag{50}$$

Now, we know that the system is controllable from the general rank of the C matrix. From the determinant of the matrix we get the it is not controllable if and only if det[c] = 0.

Now, in Eq. (50) we equate the value to zero to get the condition.

$$\frac{-(g^6{l_1}^2 - 2l_1l_2g^6 + g^6{l_2}^2)}{M^6{l_1}^6{l_2}^6} = 0 \tag{51}$$

$$\Rightarrow -(g^6{l_1}^2 - 2l_1l_2g^6 + g^6{l_2}^2) = 0 \tag{52}$$

$$\Rightarrow -g^6({l_1}^2 - 2l_1l_2 + {l_2}^2) = 0 \tag{53}$$

$$\Rightarrow -g^6(l_1 - l_2)^2 = 0 \tag{54}$$

from this we get the condition that,

$$l_1 - l_2 = 0 \tag{55}$$

$$\Rightarrow l_1 = l_2 \tag{56}$$

Thus, we obtain that $l_1 = l_2$ is the condition in which the system is not controllable.

d LQR Controller Design

This section deals with the design of an Linear Quadratic Regulator and also the simulation results for the initial conditions for both the linear and non-linear conditions. This is done using the following parameters:

$M = 1000kg$

$m_1 = m_2 = 100kg$

$l_1 = 20m$, and

$l_2 = 10m$

The LQR controller design uses a cost function which then gives rise to the Riccati equation on minimization. The cost function is given by:

$$J = \int_0^\infty (x^TQx + u^TRu)dt \tag{57}$$

And the Riccati equation is written as:

$$A^TP + PA - PBR^{-1}B^TP + Q = 0 \tag{58}$$

This is used as the general equation to obtain the value of P, which then gives us the K, the controller gain in $u = -Kx$. This is done using:

$$-K = R^{-1}\ B\ P \tag{59}$$

Using the LQR function in MATLAB, we obtain the value of K. We also consider the initial conditions (in degrees) to be as:

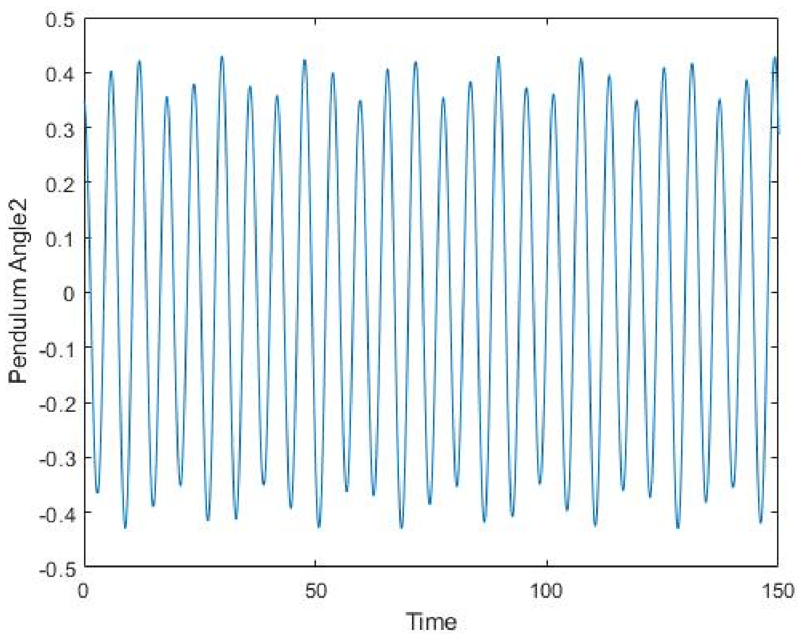
X = $\begin{bmatrix} 0 & 0 & 15 & 0 & 20 & 0 \end{bmatrix}$

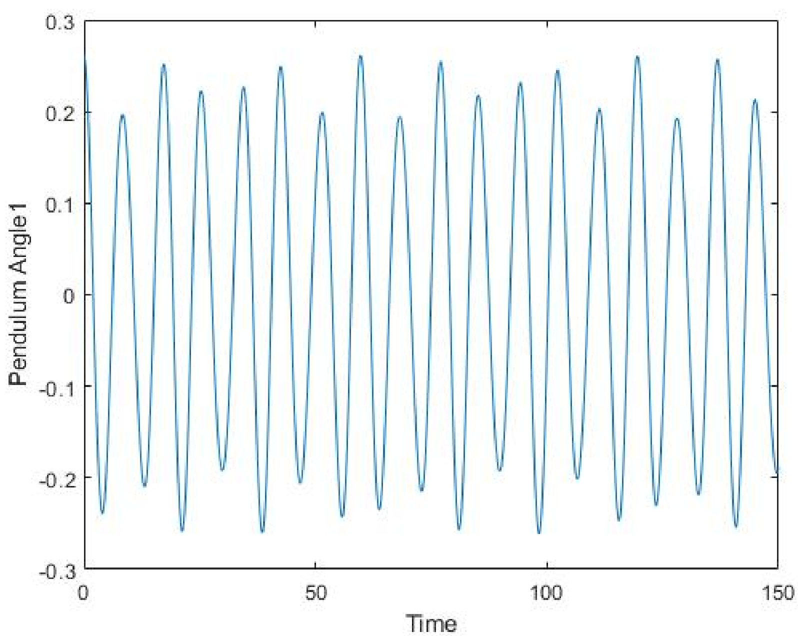
The values for the weights were assumed as:

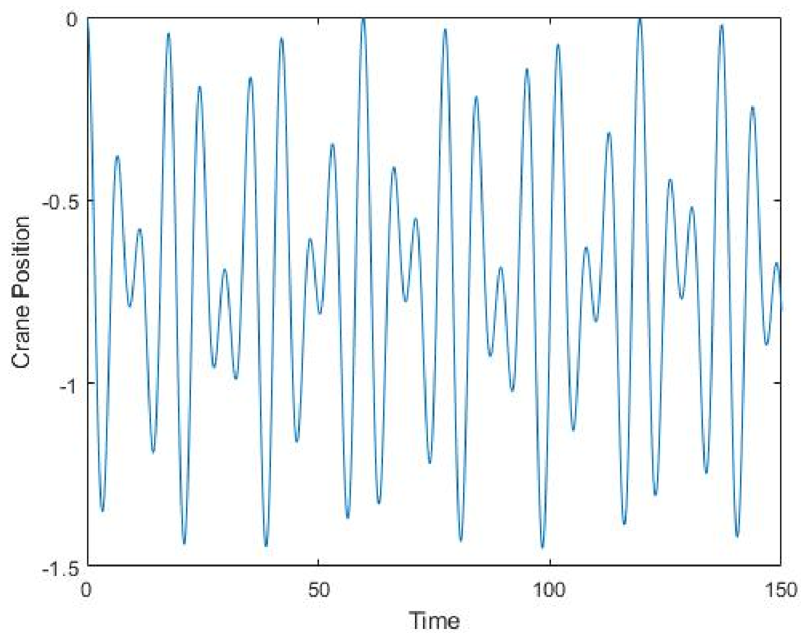
$$\begin{gathered} \text{Q(1,1)} = 50 \\ \text{Q(2,2)} = 1500 \\ \text{Q(3,3)} = 50000 \\ \text{Q(4,4)} = 120000 \\ \text{Q(5,5)} = 500000 \\ \text{Q(6,6)} = 80000 \end{gathered}$$

The MATLAB code for the simulation results and the output values are mentioned below:

(a) Response to initial conditions:






0
-0.5
-1
-1.5
Crane Position
0
50
100
150
Time

The stability of the obtained system is also checked using Lyapunov's indirect method.

This gives information about the stability of the system using the eigen values for our given A matrix. It states that if a system has N eigen values and all the values have no positive real part, then the system is locally stable.

Similarly, if any of the eigen values have a positive real part, we get that the system is unstable.

In the case of our project, we have calculated the eigen values of A to be as follows:

-0.3511 + 1.0782i

-0.3511 - 1.0782i

-0.2141 + 0.0797i

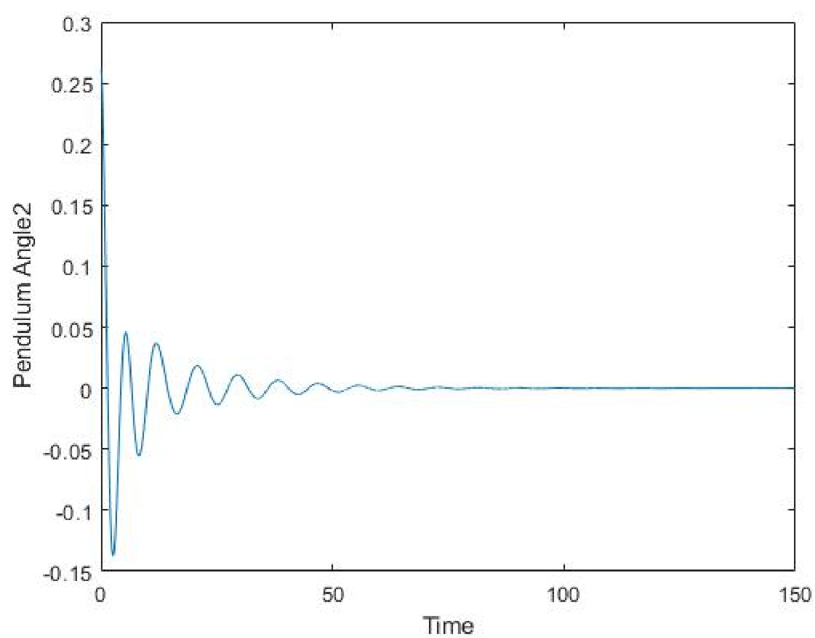
-0.2141 - 0.0797i

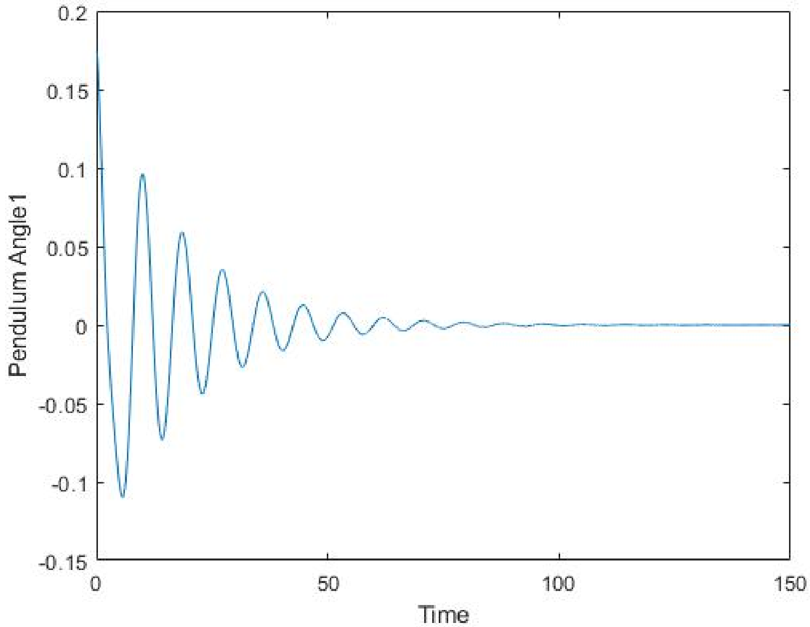
-0.0579 + 0.7236i

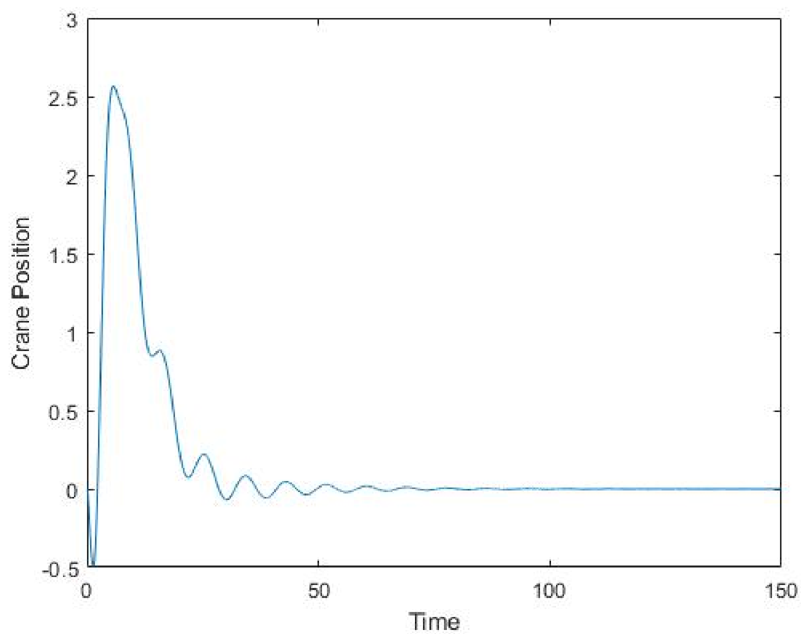
-0.0579 - 0.7236i

Since they do not have any positive real part and only negative real part, we can say that the system is locally stable.

(b) Response to LQR controller:




0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
Pendulum Angle1
0
50
100
150
Time


3
2.5
2
1.5
1
0.5
0
-0.5
Crane Position
0
50
100
150
Time

e Observability of the Linearized System for the given Output Vectors

Following the checking of the controllability and stability of the system that we have designed, we now obtain the values and simulation results of observability for the given scenarios.

The simulation results for the observability were obtained using MATLAB and are mentioned below for the various output vectors of $x(t), (\theta_1(t), \theta_2(t)), (x(t), \theta_2(t)) or (x(t), \theta_1(t), \theta_2(t))$:

```
clc
clear all

syms x1 x1_dot t1 t1_dot t2 t2_dot

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

State Space form of the System

```
%%%%%%%% X = A*x +B*u %%%%%%%%%%%
A=[0 1 0 0 0 0
    0 0 (-g*m1)/M 0 (-g*m2)/M 0
    0 0 0 1 0 0
    0 0 (-g*(m1+M))/(M*l1) 0 (-g*m2)/(M*l1) 0
    0 0 0 0 0 1
    0 0 (-g*m1)/(M*l2) 0 (-g*(m2+M))/(M*l2) 0 ];

Order_of_A=6;
```

Checks for Different Cases

```
%Case 1 for x(t)

disp('Observability check for Case-x(t)')

C1=[1 0 0 0 0 0];
% Check1=rank([C1;C1*A;C1*(A^2);C1*(A^3);C1*(A^4);C1*(A^5)])
Check1=Observability(C1,A)

disp('Observability check for Case-x(t)')

if Check1==Order_of_A
    disp('System is Observable for x(t)')
else
    disp('System is not Observable for the expected Output')
end

%Case 2 for (t1,t2)

C2=[0 0 1 0 0 0
    0 0 0 0 1 0];

Check2=Observability(C2,A)

disp('Observability check for Case-(t1,t2)')

if Check2==Order_of_A
    disp('System is Observable for (t1,t2)')
```

1

```
else
    disp('System is not Observable for the expected Output')
end

%Case 3 for (x,t2)

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

Check3=Observability(C3,A)

disp('Observability check for Case-(x,t2)')

if Check3==Order_of_A
    disp('System is Observable for (x,t2)')
else
    disp('System is not Observable for the expected Output')
end


%Case 4 for (x,t1,t2)

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

Check4=Observability(C4,A)

disp('Observability check for Case-(x,t1,t2)')

if Check4==Order_of_A
    disp('System is Observable for (x,t1,t2)')
else
    disp('System is not Observable for the expected Output')
end
```

```
Observability check for Case-x(t)

Check1 =

     6

Observability check for Case-x(t)
System is Observable for x(t)

Check2 =

     4

Observability check for Case-(t1,t2)
System is not Observable for the expected Output

Check3 =
```

```
     6

Observability check for Case-(x,t2)
System is Observable for (x,t2)

Check4 =

     6

Observability check for Case-(x,t1,t2)
System is Observable for (x,t1,t2)
```

Published with MATLAB® R2019a

f Luenberger Observer for each output vector

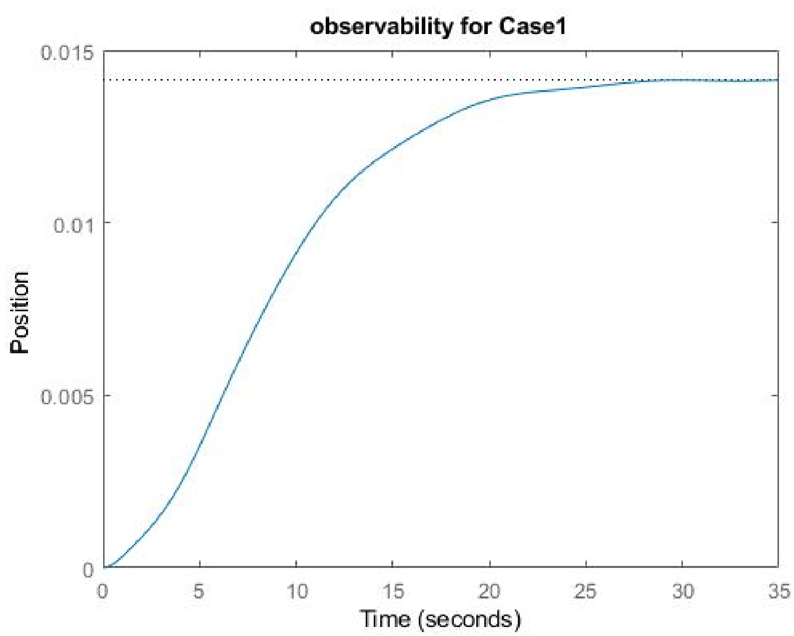
The Luenberger Observer is a state observer that provides estimates and calculates the error and corrects it with respect to the output values of the state.
It is given by the following general equation:

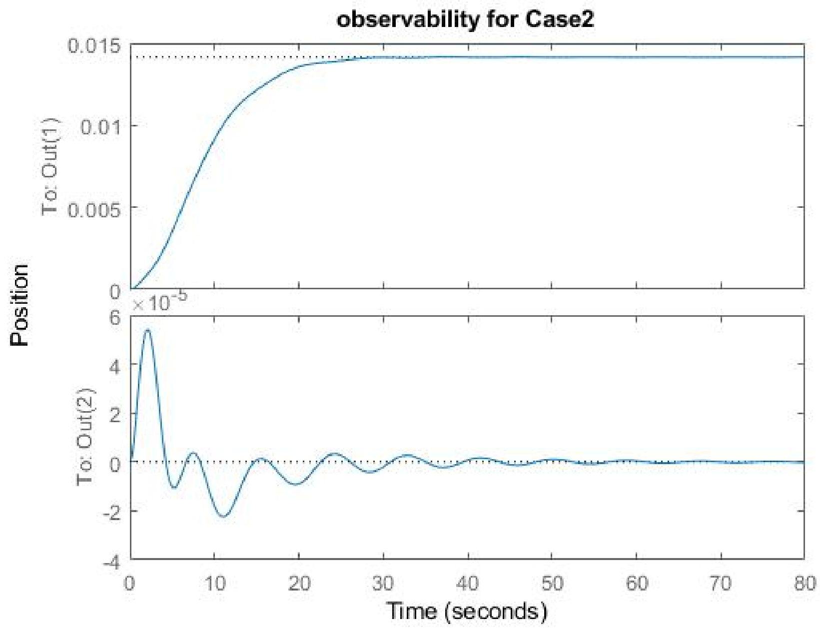
$$\dot{X} = AX + BU + LC(X - \hat{X}) \tag{60}$$

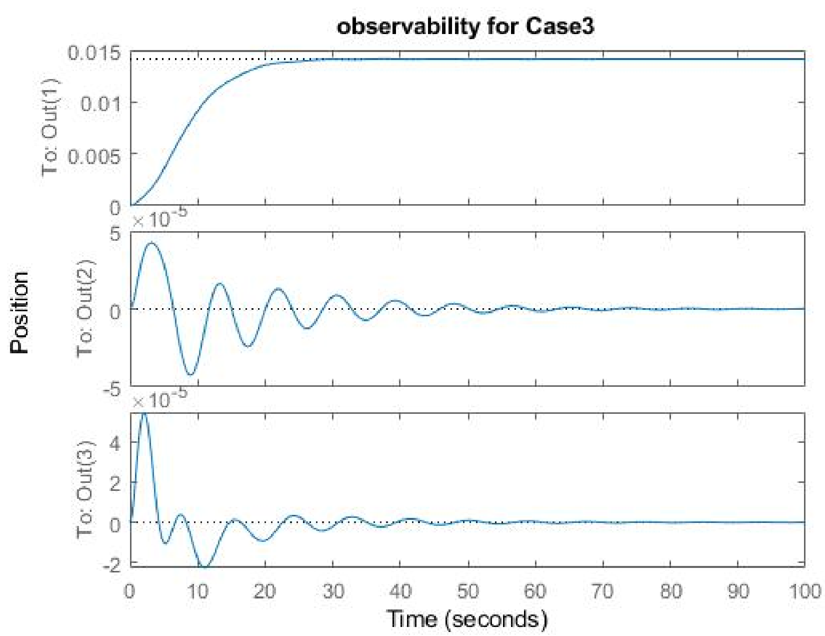
We now let $X_e = X - \hat{X}$ and rearrange to get the final equation for the error to be:

$$\dot{X}_e = (A - LC)(X_e) \tag{61}$$

Now we develop the 'best' possible Luenberger observer by simulating it using both the original nonlinear system as well as the linearized version.
Conditions for the Luenberger oberserver are:
For the (A^T, C^T) to be stabilizable, we need the them to be detectable and similarly, for controllability, they need to be observable.
Also, the matrix $(A - LC)^T$ must be stable for A-LC to be stable.
The MATLAB and Simulink results for the response to Luenberger Observer are [3]:




observability for Case2
Position
To: Out(1)
To: Out(2)
0.015
0.01
0.005
0
× 10^-5
6
4
2
0
-2
-4
0
10
20
30
40
50
60
70
80
Time (seconds)


observability for Case3
Position
To: Out(1)
To: Out(2)
To: Out(3)
0.015
0.01
0.005
0
× 10^-5
5
0
-5
× 10^-5
4
2
0
-2
0
10
20
30
40
50
60
70
80
90
100
Time (seconds)

g LQG Feedback Controller

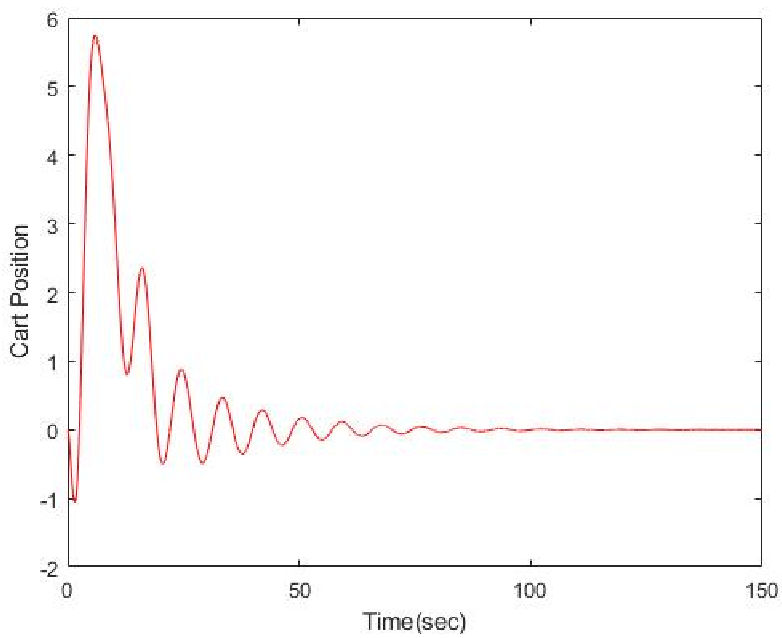
The LQG feedback controller is defined as the Linear Quadratic Gaussian which deals with a linear system driven by 'Gaussian white noise'. It is simply a combination of the LQR (Linear Quadratic Regulator) controller and the Kalman filter. LQG controller is used for both linear time-invariant systems and linear time-varying systems. the LQG controller itself is a dynamic system that controls systems have the same dimension in its state.

The state space representation for the LQG controller is given as

$$\dot{X} = AX + BU + Bw \tag{62}$$

$$Y = CX + v \tag{63}$$

Where, w is the process noise and v is the measurement noise. The response of the LQG system to the step source that is obtained from simulink is:



h Appendix

```
clc
clear all

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

syms g m1 m2 l1 l2 M
```

State Space form of the System

```
%%%%%%%% X = A*x +B*u%%%%%%%%%%%
A=[0 1 0 0 0 0
    0 0 (-g*m1)/M 0 (-g*m2)/M 0
    0 0 0 1 0 0
    0 0 (-g*(m1+M))/(M*l1) 0 (-g*m2)/(M*l1) 0
    0 0 0 0 0 1
    0 0 (-g*m1)/(M*l2) 0 (-g*(m2+M))/(M*l2) 0 ];

%size(A)
B=[ 0
    1/M
    0
    1/(M*l1)
    0
    1/(M*l2)];

C= [ 1 0 0 0 0 0
      0 0 1 0 0 0
      0 0 0 0 1 0];

 % Check for Controllability
disp('CONTROLLABILITY CHECK')

% Using the generalized form rank([B B*A B*A^2 B*A^3 B*A^4
 B*A^5])=size(A)

rank_check=([B A*B (A^2)*B (A^3)*B (A^4)*B (A^5)*B])
det(rank_check)
```

```
CONTROLLABILITY CHECK

rank_check =

[        0,      1/M,                                                  0,
         - (g*m1)/(M^2*l1) - (g*m2)/(M^2*l2),

                                              0,                ((g^2*m1*(M
 + m1))/(M^2*l1) + (g^2*m1*m2)/(M^2*l2))/(M*l1) + ((g^2*m2*(M + m2))/
(M^2*l2) + (g^2*m1*m2)/(M^2*l1))/(M*l2)]
```

1

Simulink models

```
[       1/M,         0,              - (g*m1)/(M^2*l1) - (g*m2)/(M^2*l2),
                                                        0,
  ((g^2*m1*(M + m1))/(M^2*l1) + (g^2*m1*m2)/(M^2*l2))/(M*l1) +
 ((g^2*m2*(M + m2))/(M^2*l2) + (g^2*m1*m2)/(M^2*l1))/(M*l2),

                                                             0]
[         0, 1/(M*l1),
 0, - (g*(M + m1))/(M^2*l1^2) - (g*m2)/(M^2*l1*l2),

                                                           0, ((g^2*m2*(M +
 m1))/(M^2*l1^2) + (g^2*m2*(M + m2))/(M^2*l1*l2))/(M*l2) + ((g^2*(M +
 m1)^2)/(M^2*l1^2) + (g^2*m1*m2)/(M^2*l1*l2))/(M*l1)]
[ 1/(M*l1),          0, - (g*(M + m1))/(M^2*l1^2) - (g*m2)/(M^2*l1*l2),
                                                        0, ((g^2*m2*(M + m1))/
(M^2*l1^2) + (g^2*m2*(M + m2))/(M^2*l1*l2))/(M*l2) + ((g^2*(M +
 m1)^2)/(M^2*l1^2) + (g^2*m1*m2)/(M^2*l1*l2))/(M*l1),

                                                        0]
[         0, 1/(M*l2),
 0, - (g*(M + m2))/(M^2*l2^2) - (g*m1)/(M^2*l1*l2),

                                                           0, ((g^2*m1*(M +
 m2))/(M^2*l2^2) + (g^2*m1*(M + m1))/(M^2*l1*l2))/(M*l1) + ((g^2*(M +
 m2)^2)/(M^2*l2^2) + (g^2*m1*m2)/(M^2*l1*l2))/(M*l2)]
[ 1/(M*l2),          0, - (g*(M + m2))/(M^2*l2^2) - (g*m1)/(M^2*l1*l2),
                                                        0, ((g^2*m1*(M + m2))/
(M^2*l2^2) + (g^2*m1*(M + m1))/(M^2*l1*l2))/(M*l1) + ((g^2*(M +
 m2)^2)/(M^2*l2^2) + (g^2*m1*m2)/(M^2*l1*l2))/(M*l2),

                                                        0]


ans =

-(g^6*l1^2 - 2*g^6*l1*l2 + g^6*l2^2)/(M^6*l1^6*l2^6)
```

Published with MATLAB® R2019a

Table of Contents


```
clc
clear all

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

State Space form of the System

```
%%%%%%%% X = A*x +B*u%%%%%%%%%%%
A=[0 1 0 0 0 0
    0 0 (-g*m1)/M 0 (-g*m2)/M 0
    0 0 0 1 0 0
    0 0 (-g*(m1+M))/(M*l1) 0 (-g*m2)/(M*l1) 0
    0 0 0 0 0 1
    0 0 (-g*m1)/(M*l2) 0 (-g*(m2+M))/(M*l2) 0 ];

%size(A)
B=[ 0
    1/M
    0
    1/(M*l1)
    0
    1/(M*l2)];

C= [ 1 0 0 0 0 0
      0 0 1 0 0 0
      0 0 0 0 1 0];

 % Check for Controllability
disp('CONTROLLABILITY CHECK')

% Using the generalized form rank([B B*A B*A^2 B*A^3 B*A^4
 B*A^5])=size(A)

rank_check=rank([B A*B (A^2)*B (A^3)*B (A^4)*B (A^5)*B]);

if rank_check==6
    disp('System is Controllable')
else
```

```
    disp('System is Not Controllable')
end

% DESIGN FOR LQR CONTROLLER
```

```
CONTROLLABILITY CHECK
System is Controllable
```

We Observed that the system is Controllable, So we now determine the stateFeed back (K)

```
%Adjusting Cost  'Q' until the system is controllable
%R=0.01
Q=diag([50 1500 50000 120000 500000 80000]);
R=0.01;

%State FeedBack 'K'
K=lqr(A,B,Q,R);
%size(K)
disp(K)

%New State Space System would be ''X= (A-B*K)x + B*U''

A_N=[A-B*K];
States={'x' 'x_dot' 'theta1' 'theta1_dot' 'theta2' 'theta2_dot'};
input={'F'};
Outputs = {'x'; 'alpha1'; 'alpha2'};

%Converting to a State Space Model
sys=ss(A_N,B,C,0,'statename',States,'inputname',input,'outputname',Outputs);
```

```
   1.0e+03 *

    0.0707    0.6344    0.1834    4.6043    5.6662    3.8161
```

Simulating the linearState responses to the initial conditions

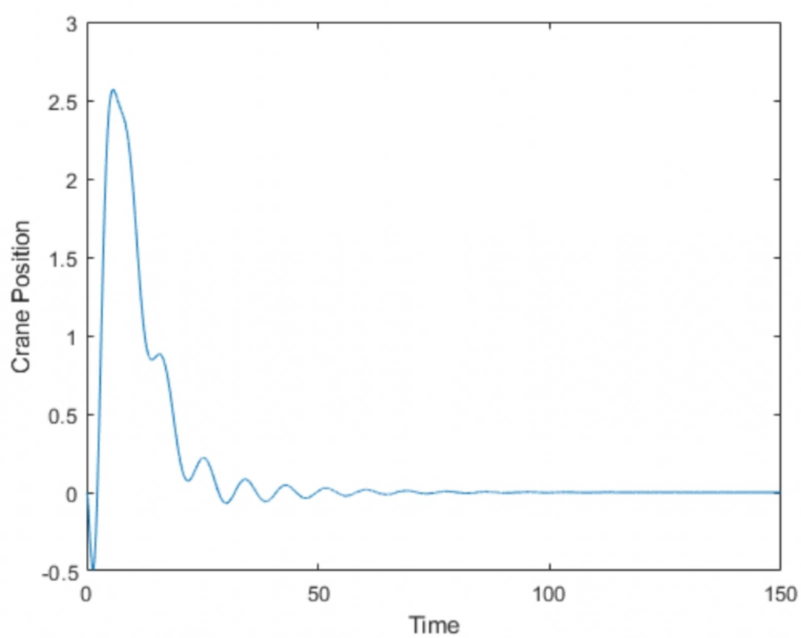
```
X=[0;0;10*pi/180;0;15*pi/180;0];
t = 0:0.01:150;
dim_t= size(t);
F = zeros(dim_t);
[Y, t_T, X_T] = lsim(sys, F, t, X);
size(Y)
```

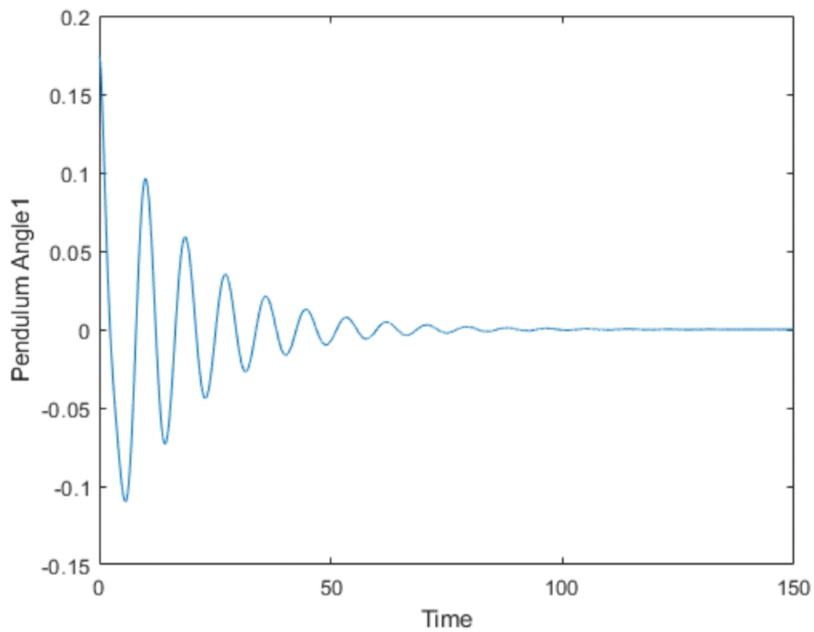
```
ans =

       15001           3
```

Visualization

```
figure,
plot(t, Y(:,1));
xlabel('Time'); ylabel('Crane Position');

figure,
plot(t, Y(:,2));
xlabel('Time'); ylabel('Pendulum Angle1');

figure,
plot(t, Y(:,3));
xlabel('Time'); ylabel('Pendulum Angle2');
% Using the Lynapunov indirect Method to Obtain Stability

%System has been Linearized
Ly_St= eig(A_N);
if real(Ly_St)<1
    disp('System is Stable')
else
    disp('System is not Stable')
end
```

System is Stable




0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
Pendulum Angle1
0
50
100
150
Time

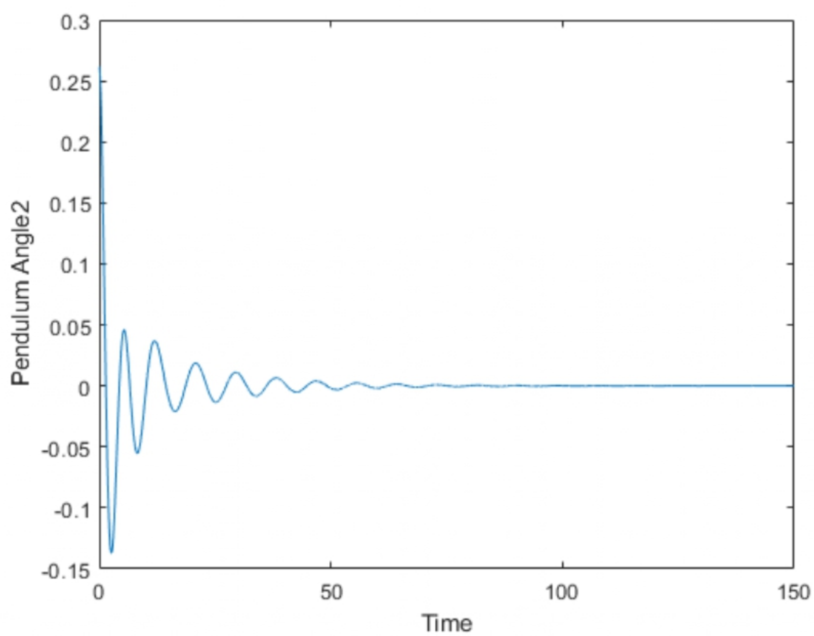

0.3
0.25
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
Pendulum Angle2
0
50
100
150
Time

Table of Contents


```
clc
clear all

% Defining the Variables
m1 = 100; m2 = 100; M = 1000;
l1 = 20; l2 = 10;
g = 9.81;
```

Substituting the Values of Different Parameters in Matrices

```
A = [0 1 0 0 0 0;
     0 0 (-(m1*g)/M) 0 (-(m2*g)/M) 0;
     0 0 0 1 0 0;
     0 0 (-(M+m1)*g/(M*l1)) 0 (-(m2*g)/(M*l1)) 0;
     0 0 0 0 0 1;
     0 0 (-(m1*g)/(M*l2)) 0 (-(M+m2)*g/(M*l2)) 0];

B = [0;
     1/M;
     0;
     1/(M*l1);
     0;
     1/(M*l2)];

C = [1 0 0 0 0 0;
     0 0 1 0 0 0;
     0 0 0 0 1 0];

D = 0;
```

Choosing the Values of Q & R

```
Q = (C')*(C);

% Assigning the Values in Q using Trial & Error Method

Q(1,1) = 70000000;
Q(3,3) = 8000000000;
Q(5,5) = 9000000000;
```

```
% Selecting the Ideal Value of R

R = 1;
```

Designing the LQR

```
% Calculating the Optimal Gain Matrix K

K = lqr(A, B, Q, R);

% Calculating the New Value of A using K

ANew = (A - (B*K));

% Creating the Observability Matrix

States = {'x' 'x_dot' 'theta1' 'theta1_dot' 'theta2' 'theta2_dot'};
Inputs = {'r'};
Outputs = {'x'; 'phi1'; 'phi2'};

% Creating the State Space Model

ClosSS = ss(ANew, B, C, D,'statename', States, 'inputname',
 Inputs, 'outputname', Outputs);
```

Initializing Conditions

```
X0 = [0;
      0;
      10*pi/180;
      0;
      15*pi/180;
      0];

t = 0:0.01:150;

Temp = size(t)
F = zeros(Temp);

% Simulating the Time Response of Dynamic System to Arbitrary Inputs

[Y, tTemp, XTemp] = lsim(ClosSS, F, t, X0);


Temp =

           1       15001
```

Plotting the Output

```
figure,
```

```
plot(t, Y(:,1));
xlabel('Time'); ylabel('Crane Position');

figure,
plot(t, Y(:,2));
xlabel('Time'); ylabel('Pendulum Angle1');

figure,
plot(t, Y(:,3));
xlabel('Time'); ylabel('Pendulum Angle2');
```

Published with MATLAB® R2019a

Table of Contents


```
clc
clear all

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

%%State Space form of the System
%%%%%%%% X = A*x +B*u%%%%%%%%%%%
A=[0 1 0 0 0 0
    0 0 (-g*m1)/M 0 (-g*m2)/M 0
    0 0 0 1 0 0
    0 0 (-g*(m1+M))/(M*l1) 0 (-g*m2)/(M*l1) 0
    0 0 0 0 0 1
    0 0 (-g*m1)/(M*l2) 0 (-g*(m2+M))/(M*l2) 0 ];

%size(A)
B=[ 0
    1/M
    0
    1/(M*l1)
    0
    1/(M*l2)];

C= [ 1 0 0 0 0 0
      0 0 1 0 0 0
      0 0 0 0 1 0];
```

The error Dynamics in the State Space Observer be (A-L*C)*e

```
  % We know that from the Checks_for_Observability the pair A,C is
 observable only for
  %1.System is Observable for x(t)
  %2.System is Observable for (x,t2)
  %3.System is Observable for (x,t1,t2)

Q=diag([50 1500 50000 120000 500000 80000]);
R=0.01;
```

```
disp(' State FeedBack')
%State FeedBack 'K'
K=lqr(A,B,Q,R);
%size(K)
disp(K)

disp('Eigen Values of (A-B*K)')
% The Pole Placement should be faster than the eigen values of (A-B*K)
Eig_Vals= eig(A-B*K);
disp(Eig_Vals)

%Arbitrary Pole Placement
Poles=[-1 -2 -3 -4 -5 -6];

%State Outputs for the system that are 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];
% for C_l=C1:C3
%     L=place(A',C_l',Poles)';
% end

L1=place(A',C1',Poles)';
L2=place(A', C3',Poles)';
L3=place(A', C4',Poles)';
size(C1)
```

```
 State FeedBack
   1.0e+03 *

    0.0707    0.6344    0.1834    4.6043    5.6662    3.8161

Eigen Values of (A-B*K)
  -0.3511 + 1.0782i
  -0.3511 - 1.0782i
  -0.2141 + 0.0797i
  -0.2141 - 0.0797i
  -0.0579 + 0.7236i
  -0.0579 - 0.7236i


ans =

     1     6
```

System Matrices for each state of the Observer

```
%For x(t)
A_L1=[(A-B*K) (B*K)
```

```
    zeros(6,6) (A-L1*C1)];
B_L1=[B
    zeros(size(B))];
C_L1=[C1 zeros(size(C1))];
%For (x(t),theta2)
A_L2=[(A-B*K) (B*K)
    zeros(6,6) (A-L2*C3)];
B_L2=[B
    zeros(size(B))];
C_L2=[C3 zeros(size(C3))];
%For (x(t),theta2, theta1)
A_L3=[(A-B*K) (B*K)
    zeros(6,6) (A-L3*C4)];
B_L3=[B
    zeros(size(B))];
C_L3=[C4 zeros(size(C4))];
```

State Space representation of the Linear system

```
SS2=ss(A_L2,B_L2,C_L2,0);
SS1=ss(A_L1,B_L1,C_L1,0)
SS3=ss(A_L3,B_L3,C_L3,0);
```

```
SS1 =

  A =
                x1          x2          x3          x4          x5
 x6
   x1            0           1           0           0           0
  0
   x2    -0.07071     -0.6344      -1.183      -4.604      -6.666
 -3.816
   x3            0           0           0           1           0
  0
   x4   -0.003536    -0.03172     -0.5592     -0.2302     -0.3333
 -0.1908
   x5            0           0           0           0           0
  1
   x6   -0.007071    -0.06344     -0.1183     -0.4604      -1.667
 -0.3816
   x7            0           0           0           0           0
  0
   x8            0           0           0           0           0
  0
   x9            0           0           0           0           0
  0
   x10           0           0           0           0           0
  0
   x11           0           0           0           0           0
  0
```

```
   x12          0          0          0          0          0
  0

                x7         x8         x9        x10        x11
  x12
   x1            0          0          0          0          0
  0
   x2      0.07071     0.6344     0.1834      4.604      5.666
  3.816
   x3            0          0          0          0          0
  0
   x4     0.003536    0.03172    0.00917     0.2302     0.2833
  0.1908
   x5            0          0          0          0          0
  0
   x6     0.007071    0.06344    0.01834     0.4604     0.5666
  0.3816
   x7          -21          1          0          0          0
  0
   x8       -173.3          0         -1          0         -1
  0
   x9         2802          0          0          1          0
  0
   x10      -105.4          0      -0.55          0      -0.05
  0
   x11       -2102          0          0          0          0
  1
   x12        1443          0       -0.1          0       -1.1
  0

  B =
            u1
   x1        0
   x2    0.001
   x3        0
   x4    5e-05
   x5        0
   x6   0.0001
   x7        0
   x8        0
   x9        0
   x10       0
   x11       0
   x12       0

  C =
        x1   x2   x3   x4   x5   x6   x7   x8   x9  x10  x11  x12
   y1    1    0    0    0    0    0    0    0    0    0    0    0

  D =
        u1
   y1    0

Continuous-time state-space model.
```

Visualizing Positions From STEP function

X=[0;0;10*pi/180;0;15*pi/180;0];

```
figure(1)
step(SS1)
title('observability for Case1')
xlabel('Time')
ylabel('Position')

figure(2)
step(SS2)
title('observability for Case2')
xlabel('Time')
ylabel('Position')

figure(3)
step(SS3)
title('observability for Case3')
xlabel('Time')
ylabel('Position')
```

Published with MATLAB® R2019a

```
clc
clear all

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

%%State Space form of the System
%%%%%%%% X = A*x +B*u%%%%%%%%%%%
A=[0 1 0 0 0 0
    0 0 (-g*m1)/M 0 (-g*m2)/M 0
    0 0 0 1 0 0
    0 0 (-g*(m1+M))/(M*l1) 0 (-g*m2)/(M*l1) 0
    0 0 0 0 0 1
    0 0 (-g*m1)/(M*l2) 0 (-g*(m2+M))/(M*l2) 0 ];

%size(A)
B=[ 0
    1/M
    0
    1/(M*l1)
    0
    1/(M*l2)];

C= [ 1 0 0 0 0 0
      0 0 1 0 0 0
      0 0 0 0 1 0];
```

Initial Value Conditions

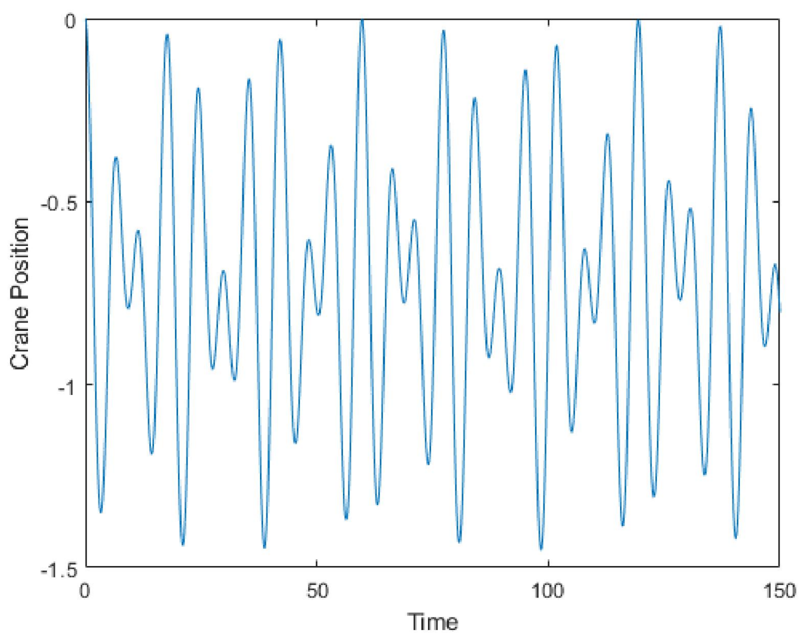
1. Initial Value Conditions are arbitrary 2. Angular Velocity and the Liner velocity and Position are Assumed to Be zero Pendulam initial positions are 15*pi/180 and %10*Pi/180

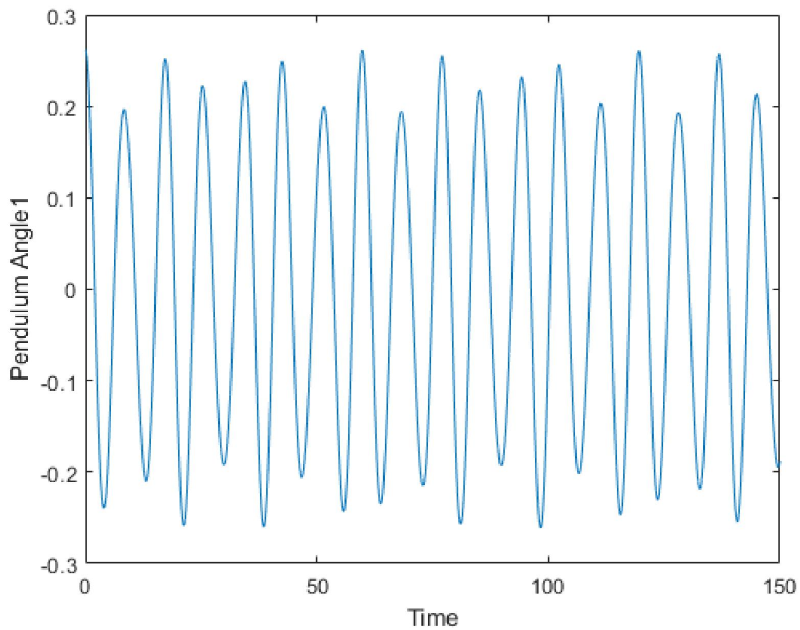
```
X=[0,0,15*pi/180,0,20*pi/180,0];
t=0:0.01:150;
dim_t= size(t);
F = zeros(dim_t);

%Defining The Parameters of the State for Visualization
States={'x' 'x_dot' 'theta1' 'theta1_dot' 'theta2' 'theta2_dot'};
input={'F'};
Outputs = {'x'; 'alpha1'; 'alpha2'};

%State

sys=ss(A,B,C,0,'statename',States,'inputname',input,'outputname',Outputs);
[Y, t_T, X_T] = lsim(sys, F, t, X);
```

Visualization

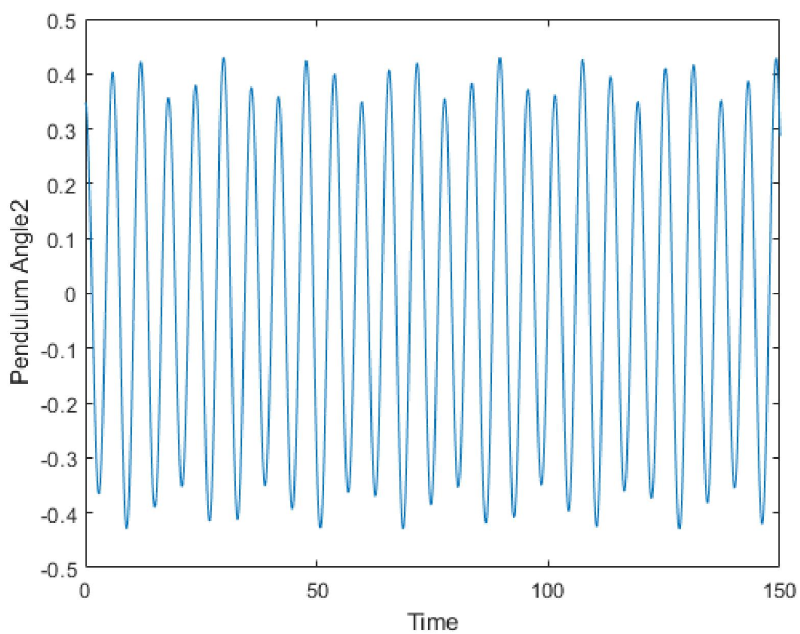
```
figure,
```

```
plot(t, Y(:,1));
xlabel('Time'); ylabel('Crane Position');

figure,
plot(t, Y(:,2));
xlabel('Time'); ylabel('Pendulum Angle1');

figure,
plot(t, Y(:,3));
xlabel('Time'); ylabel('Pendulum Angle2');
```





0.3
0.2
0.1
0
-0.1
-0.2
-0.3
Pendulum Angle1
0
50
100
150
Time



0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
Pendulum Angle2
0
50
100
150
Time

```
clc
clear all

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

%%State Space form of the System
%%%%%%%% X = A*x +B*u%%%%%%%%%%%%
A=[0 1 0 0 0 0
    0 0 (-g*m1)/M 0 (-g*m2)/M 0
    0 0 0 1 0 0
    0 0 (-g*(m1+M))/(M*l1) 0 (-g*m2)/(M*l1) 0
    0 0 0 0 0 1
    0 0 (-g*m1)/(M*l2) 0 (-g*(m2+M))/(M*l2) 0 ];

%size(A)
B=[ 0
    1/M
    0
    1/(M*l1)
    0
    1/(M*l2)];

C= [ 1 0 0 0 0 0
      0 0 1 0 0 0
      0 0 0 0 1 0];
```

Considering FeedBack Control for the State x(t)

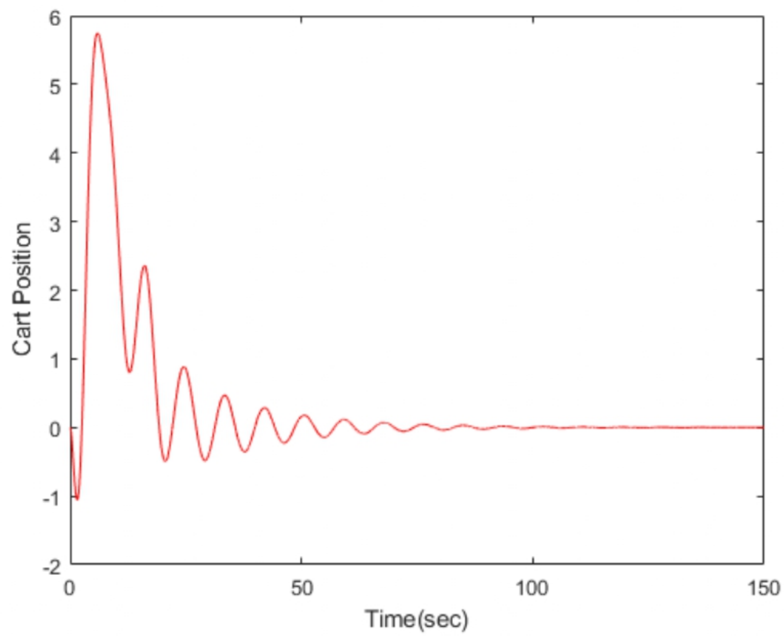
```
C1=[1 0 0 0 0 0];
Q=diag([50 1500 50000 1200 50000 80000]);
R=0.01;
disp(' State FeedBack')
%State FeedBack 'K'
K=lqr(A,B,Q,R);
%size(K)
disp(K)


Poles=[-1 -2 -3 -4 -5 -6];
L1=place(A',C1',Poles)';

%Kalman Estimator
stat_space= ss(A,[B B],C,0);
R1=0.01; Q1=0.05;
sensors=[1];
W=[1];
```

1

```
[~,L,~]=kalman(stat_space,Q1,R1,[],sensors,W);

 %Defining The Parameters of the State for Visualization
States={'x' 'x_dot' 'theta1' 'theta1_dot' 'theta2' 'theta2_dot','e_1','e_2','e_3',
input={'F'};
Outputs = {'x'};

A_L1=[(A-B*K) (B*K)
    zeros(6,6) (A-L1*C1)];
B_L1=[B
    zeros(size(B))];
C_L1=[C1 zeros(size(C1))];
stat_space2=ss(A_L1,B_L1,C_L1,0,'statename',States,'inputname',input,'outputname',


X=[0;0;45*pi/180;0;35*pi/180;0;0;0;0;0;0;0];
t = 0:0.01:150;
dim_t= size(t);
F = zeros(dim_t);
[Y, t_T, X_T] = lsim(stat_space2, F, t, X);

figure(1)
plot(t,Y(:,1),'r')
ylabel('Cart Position')
xlabel('Time(sec)')
```

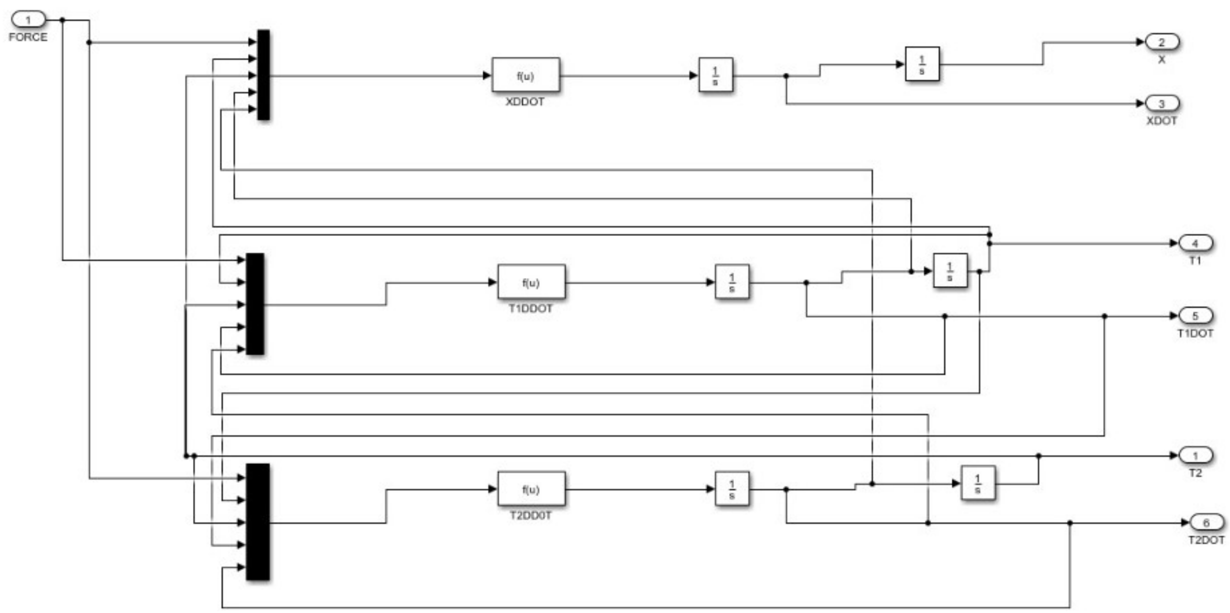
```
 State FeedBack
   1.0e+03 *

    0.0707    0.5992    0.6198    2.6216    2.1607    2.0024
```

2

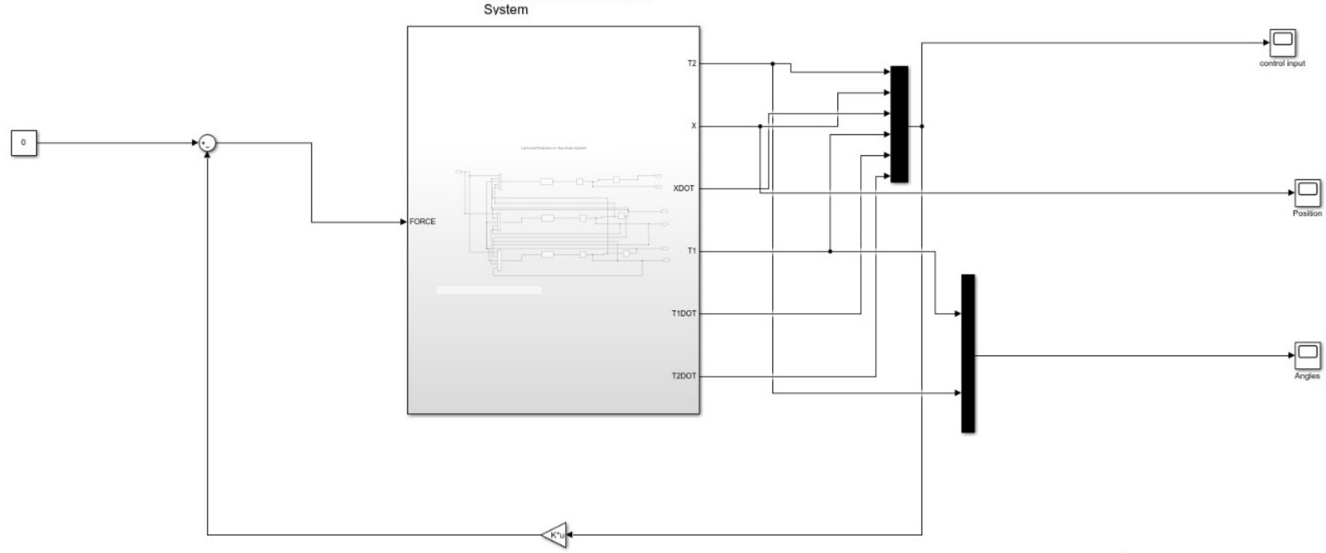


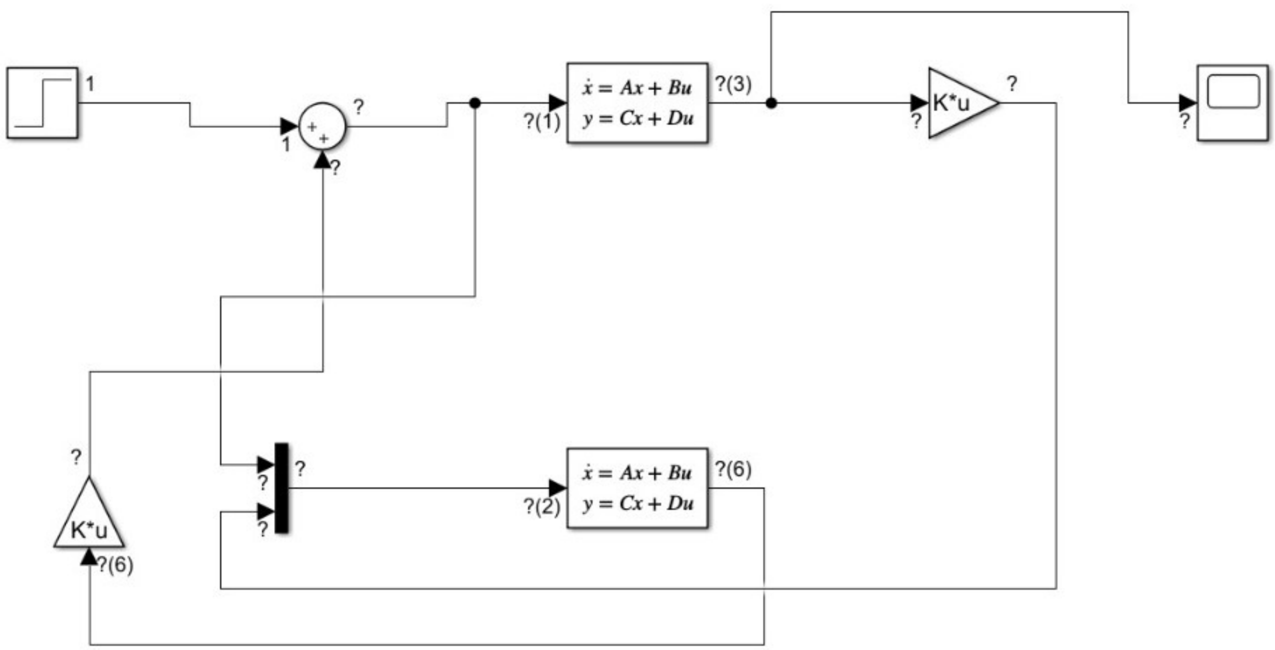
Published with MATLAB® R2019a

Cart and Pendulam of Non linear System



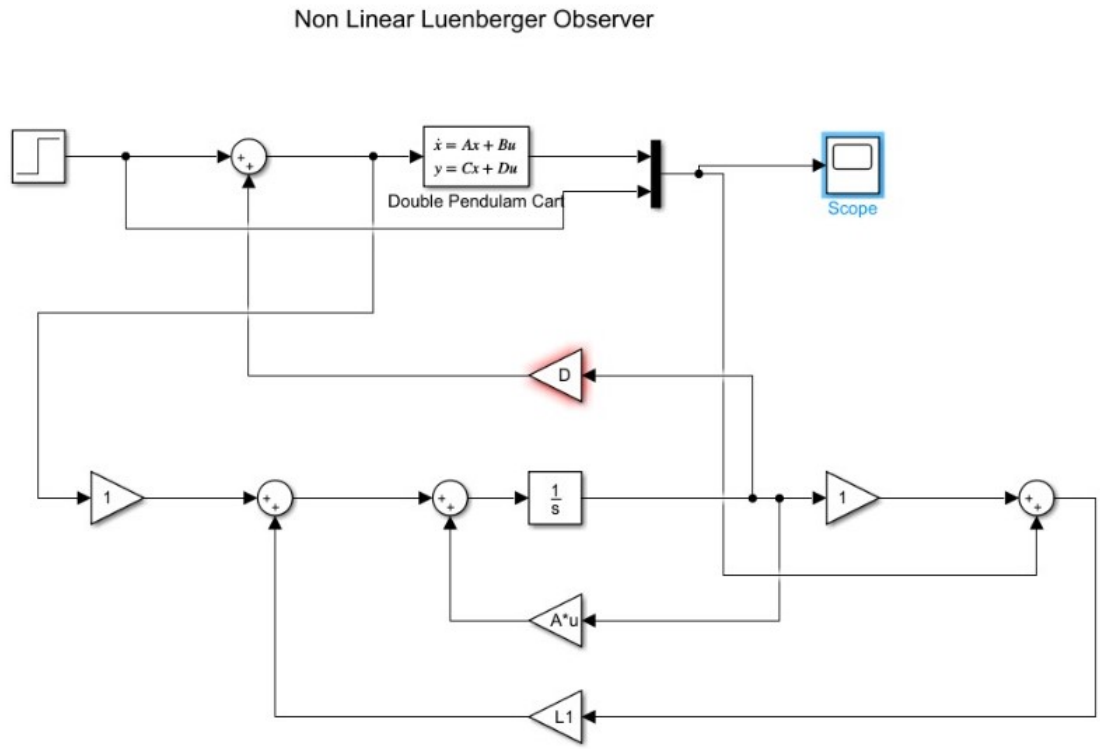
LQR Controller of a nolinear
System





Luenberger Observer

The simulink model was constructing using [2]



References

[1] Controllability and observability.

[2] Inverted pendulum: Simulink model.

[3] Luenberger model.