A. James Clark
SCHOOL OF ENGINEERING

University of Maryland, College Park

Control of Robotic Systems

Design & Analysis of LQR, LQG Controllers

Members:	*UID:*
Prasheel Renkuntla	116925570

Contents

Acknowledgement

I would like to show my deepest gratitude to my Professor, Dr. Waseem Malik, Control of Robot Systems, University of Maryland, College Park, whose constant and apt guidance on the project helped me overcome the challenges faced. I would also like to thank my friends, family, fellow students, and faculty who were involved directly or indirectly towards the completion of this project.

Abstract

This project deals with stabilizing and controlling a non-linear double crane system. There are two components to the given problem statement for this project. In the first component, the equations of motion for the system are determined. The presented non-linear system is then linearised and an LQR controller is designed to stabilize the obtained linearized system. It is then tuned to stabilize the initial non-linear system.
In the second component of this project, for each of the observable output vectors, a Luenberger Observer is designed for the linear system with the given initial conditions and a step response. Finally, an LQG controller is designed to stabilise the non linear system using a Kalman filter along with the existing LQR control.

List of Figures

1 Introduction

In the first component of the project the equations of motion of the system are determined. The system is represented in Non Linear State Space Representation. The non-linear system is then linearised using Jacobean linearisation about an equilibrium point and the Linear State Space representation is found. The conditions for controllability are defined in this report and the controllability of the linearized system is checked. An LQR[1] controller is designed to stabilize the obtained linearized system. It is then tuned to stabilize the initial non-linear state space equations. The Lyapunov indirect method is used to find stability of the closed-loop system.
In the second component of this project the observability of different output vectors is verified. For each of these observable vectors, a Luenberger Observer[4] is designed for the linear system with the given initial conditions and a step response. Finally, an LQG[3] controller is designed to stabilise the non linear system using a Kalman filter and the existing LQR control.

1.1 System Description

The system consists of a Crane which moves in only one-dimensional track. There is no friction for the body with mass M moving along the track which is actuated by an external Force F. This force F makes the input for the system. Additionally, There are two loads suspended from cables attached to the crane. The loads have mass m1 and m2, and the lengths of the cables are l1 and l2, respectively. The following figure depicts the crane and associated variables that will be used throughout the project.

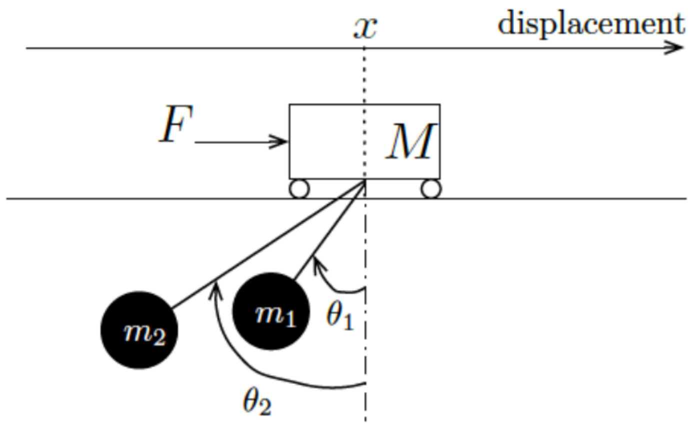

Figure 1: Double Crane System

1.2 Model Assumptions

- No friction between the cart and its track.
- All masses have uniform density
- The cart movement has only 1 degree of freedom
- The center of mass for each body is located at their geometric centers
- The system is modelled in 2D not considering the mass and thickness of rod
- The center of mass of the cart is located at the point of suspension.

2 Design and Analysis of the system

2.1 Defining the Model

First, we will find the energy related to each mass m1, m2, M in the system and then we will use the Euler-Lagrangian machine to find the dynamic model. First, for mass m_1, The position vector is given by,

$$r_1(t) = (x - l_1 sin\theta_1)\hat{i} + (l_1 cos\theta_1)\hat{j}$$
$$\dot{r}_1(t) = (\dot{x} - l_1\dot{\theta}_1 cos\theta_1)\hat{i} + (-l_1\dot{\theta}_1 sin\theta_1)\hat{j}$$

$$T_1 = K.E = \frac{1}{2}m[\dot{x}_1^2 + \dot{y}_1^2] = \frac{1}{2}m[(\dot{x} - l_1\dot{\theta}_1 cos\theta_1)^2 + (l_1\dot{\theta}_1 sin\theta_1)^2]$$
$$V_1 = P.E_1 = -m_1 g l_1 cos\theta_1$$

For mass m_2,
The position vector is given by,

$$r_2(t) = (x - l_2 sin\theta_2)\hat{i} + (l_2 cos\theta_2)\hat{j}$$
$$\dot{r}_2(t) = (\dot{x} - l_2\dot{\theta}_2 cos\theta_2)\hat{i} + (-l_2\dot{\theta}_2 sin\theta_2)\hat{j}$$

$$T_2 = K.E = \frac{1}{2}m[\dot{x}_1^2 + \dot{y}_1^2] = \frac{1}{2}m[(\dot{x} - l_2\dot{\theta}_2 cos\theta_2)^2 + (l_2\dot{\theta}_2 sin\theta_2)^2]$$
$$V_2 = P.E_2 = -m_2 g 2_2 cos\theta_2$$

For main body with mass M,

$$T_3 = K.E = \frac{1}{2}M\dot{x}^2$$
$$V_3 = P.E_3 = 0$$

So, the total energy of the system is given by,

$$L = (T_1 + T_2 + T_3) - (V_1 + V_2 + V_3)$$

$$L = \frac{1}{2}[M\dot{x}^2 + m_1(\dot{x} - l_1\dot{\theta_1}cos\theta_1)^2 + m_1(-l_1\dot{\theta_1}sin\theta_1)^2 + m_2(\dot{x} - l_2\dot{\theta_2}cos\theta_2)^2 \\ +m_2(-l_2\dot{\theta_2}sin\theta_2)^2] - [-m_1gl_1cos\theta_1 - m_2gl_2cos\theta_2] \tag{1}$$

The Lagrange Crank for the three variables are,

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

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta_1}} - \frac{\partial L}{\partial \theta_1} = 0 \tag{3}$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta_2}} - \frac{\partial L}{\partial \theta_2} = 0 \tag{4}$$

Using the Euler-Lagrangian, We will solve for x, θ_1, θ_2.
For x,

$$\frac{\partial L}{\partial \dot{x}} = M\ddot{x} + m_1(\dot{x} - l_1\dot{\theta_1}\cos\theta_1) + m_2(\dot{x} - l_2\dot{\theta_2}\cos\theta_2)$$

$$\frac{d}{dt}\frac{\partial L}{\partial \ddot{x}} = M\ddot{x} + m_1(\ddot{x} - (l_1\ddot{\theta_1} - l_1\dot{\theta}_1^2\sin\theta_1) + m_2(\ddot{x} - (l_2\ddot{\theta_2} - l_2\dot{\theta}_2^2\sin\theta_2)) \tag{5}$$

$$\frac{\partial L}{\partial x} = 0 \tag{6}$$

Using eq.(2) Euler-Lagrangian equation (for x),

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

$$F = \ddot{x}(M + m_1 + m_2) - m_1l_1\ddot{\theta_1}\cos\theta_1 - m_2l_2\ddot{\theta_2}\cos\theta_2 \\ +m_1l_1\dot{\theta}_1^2\sin\theta_1 + m_2l_2\dot{\theta}_2^2\sin\theta_2 \tag{7}$$

For θ_1,

$$\frac{\partial L}{\partial \dot{\theta_1}} = m_1(\dot{x} - l_1\dot{\theta_1}\cos\theta_1)(-l_1\cos\theta_1) + m_1l_1\dot{\theta_1}\sin\theta_1(l_1\sin\theta_1)$$

$$\frac{\partial L}{\partial \dot{\theta_1}} = m_1l_1^2\dot{\theta_1}\cos^2\theta_1 - m_1\dot{x}l_1\cos\theta_1 + m_1l_1^2\dot{\theta_1}\sin^2\theta_1$$

$$\frac{\partial L}{\partial \dot{\theta_1}} = m_1l_1^2\dot{\theta_1} - m_1\dot{x}l_1\cos\theta_1$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta_1}} = m_1l_1^2\ddot{\theta_1} - m_1\ddot{x}l_1\cos\theta_1 + m_1\dot{x}l_1\dot{\theta_1}\sin\theta_1 \tag{8}$$

$$\frac{\partial L}{\partial \theta_1} = m_1l_1\dot{x}\dot{\theta_1}\sin\theta_1 - m_1l_1^2\dot{\theta}_1^2\cos\theta_1\sin\theta_1 + m_1l_1^2\dot{\theta}_1^2\cos\theta_1\sin\theta_1 - m_1gl_1\sin\theta_1$$

$$\frac{\partial L}{\partial \theta_1} = m_1\dot{x}l_1\dot{\theta_1}\sin\theta_1 - m_1gl_1\sin\theta_1 \tag{9}$$

Using eq.(3) Euler-Lagrangian equation (for θ_1),

$$0 = m_1 l_1^2 \ddot{\theta}_1^2 - m_1 \ddot{x} l_1 \cos\theta_1 + m_1 \dot{x} l_1 \dot{\theta}_1 \sin\theta_1 - m_1 \dot{x} l_1 \dot{\theta}_1 \sin\theta_1 + m_1 g l_1 \sin\theta_1$$

$$\ddot{\theta}_1 = \frac{m_1(\ddot{x} l_1 \cos\theta_1 - g l_1 \sin\theta_1)}{m_1 l_1^2}$$

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

Similarly, using the Euler-Lagrangian equation(4) to compute for θ_2, we get,

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

Using the above equations for θ_1 and θ_2, we substitute in F, to compute $\ddot{x}$, i.e,

$$F = \ddot{x}(M + m_1 + m_2) - m_1 l_1 \cos\theta_1 \left(\frac{\ddot{x}\cos\theta_1}{l_1} - \frac{g\sin\theta_1}{l_1}\right) - m_2 l_2 \cos\theta_2 \left(\frac{\ddot{x}\cos\theta_2}{l_2} - \frac{g\sin\theta_2}{l_2}\right)$$
$$+ m_1 l_1 \dot{\theta}_1^2 \sin\theta_1 + m_2 l_2 \dot{\theta}_2^2 \sin\theta_2$$

$$F = \ddot{x}(M + m_1(1 - \cos\theta_1^2) + m_2(1 - \cos\theta_2^2)) + m_1 g \sin\theta_1 \cos\theta_1$$
$$+ m_2 g \sin\theta_2 \cos\theta_2 + m_1 l_1 \dot{\theta}_1^2 \sin\theta_1 + m_2 l_2 \dot{\theta}_2^2 \sin\theta_2$$

$$\ddot{x} = \frac{F - m_1 g \sin\theta_1 \cos\theta_1 - m_2 g \sin\theta_2 \cos\theta_2 - m_1 l_1 \dot{\theta}_1^2 \sin\theta_1 - m_2 l_2 \dot{\theta}_2^2 \sin\theta_2}{M + m_1 \sin\theta_1^2 + m_2 \sin\theta_2^2} \tag{12}$$

2.2 Non-Linear Representation

The non-linear state space representation of a system is-

$$\begin{aligned} \dot{X}(t) &= F(X, U) \\ Y(t) &= H(X, U) \end{aligned} \tag{13}$$

where F is a function of the state X and input U. For the given crane system, the non linear state-space is given by-

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta}_1 \\ \ddot{\theta}_1 \\ \dot{\theta}_2 \\ \ddot{\theta}_2 \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \frac{F - m_1 g \sin\theta_1 \cos\theta_1 - m_2 g \sin\theta_2 \cos\theta_2 - m_1 l_1 \dot{\theta}_1^2 \sin\theta_1 - m_2 l_2 \dot{\theta}_2^2 \sin\theta_2}{M + m_1 \sin\theta_1^2 + m_2 \sin\theta_2^2} \\ \dot{\theta}_1 \\ \frac{\ddot{x}\cos\theta_1}{l_1} - \frac{g\sin\theta_1}{l_1} \dot{\theta}_2 \\ \frac{\ddot{x}\cos\theta_2}{l_2} - \frac{g\sin\theta_2}{l_2} \end{bmatrix} \tag{14}$$

2.3 Linearisation

For the system, we can either do a small angle linearisation[2] or a Jacobian linearisation. The resulting state space must be linear in either methods. In Jacobian Linearisation, we will find A_F and B_F by using the below equation-

$$\begin{aligned} A_F &= \nabla_{\vec{X}} F(\overrightarrow{X(t)}, \overrightarrow{U(t)}) \\ B_F &= \nabla_{\vec{U}} F(\overrightarrow{X(t)}, \overrightarrow{U(t)}) \\ C_H &= \nabla_{\vec{X}} H(\overrightarrow{X(t)}, \overrightarrow{U(t)}) \\ D_H &= \nabla_{\vec{U}} H(\overrightarrow{X(t)}, \overrightarrow{U(t)}) \end{aligned} \tag{15}$$

Here, we calculate the Jacobian matrix at equilibrium point $(X_0, \theta_{1_0}, \theta_{2_0}) = (0, 0, 0)$ and given, the corresponding first derivatives $(\dot{X}, \dot{\theta_1}, \dot{\theta_2}) = (0, 0, 0)$ Here, A_F is given by,

$$A_F = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} & \frac{\partial f_1}{\partial x_4} & \frac{\partial f_1}{\partial x_5} & \frac{\partial f_1}{\partial x_6} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3} & \frac{\partial f_2}{\partial x_4} & \frac{\partial f_2}{\partial x_5} & \frac{\partial f_2}{\partial x_6} \\ \frac{\partial f_3}{\partial x_1} & \frac{\partial f_3}{\partial x_2} & \frac{\partial f_3}{\partial x_3} & \frac{\partial f_3}{\partial x_4} & \frac{\partial f_3}{\partial x_5} & \frac{\partial f_3}{\partial x_6} \\ \frac{\partial f_4}{\partial x_1} & \frac{\partial f_4}{\partial x_2} & \frac{\partial f_4}{\partial x_3} & \frac{\partial f_4}{\partial x_4} & \frac{\partial f_4}{\partial x_5} & \frac{\partial f_4}{\partial x_6} \\ \frac{\partial f_5}{\partial x_1} & \frac{\partial f_5}{\partial x_2} & \frac{\partial f_5}{\partial x_3} & \frac{\partial f_5}{\partial x_4} & \frac{\partial f_5}{\partial x_5} & \frac{\partial f_5}{\partial x_6} \\ \frac{\partial f_6}{\partial x_1} & \frac{\partial f_6}{\partial x_2} & \frac{\partial f_6}{\partial x_3} & \frac{\partial f_6}{\partial x_4} & \frac{\partial f_6}{\partial x_5} & \frac{\partial f_6}{\partial x_6} \end{bmatrix}$$

where functions $f_1, f_2, f_3, f_4, f_5, f_6$ are the individual elements of the vector in the right hand side of equation (14) Similarly, B_F for input U = F, is given by-

$$B_F = \begin{bmatrix} \frac{\partial f_1}{\partial U} & \frac{\partial f_2}{\partial U} & \frac{\partial f_3}{\partial U} & \frac{\partial f_4}{\partial U} & \frac{\partial f_5}{\partial U} & \frac{\partial f_6}{\partial U} \end{bmatrix}^T$$

Computing the individual elements, we get the following final matices.

$$A_F = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{-(m_1 g)}{M} & 0 & \frac{-(m_2 g)}{M} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{-((m_1+M)g)}{Ml_1} & 0 & \frac{-(m_2 g)}{Ml_1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{-(m_1 g)}{Ml_2} & 0 & \frac{-((m_2+M)g)}{Ml_2} & 0 \end{bmatrix} B_F = \begin{bmatrix} 0 \\ 1/M \\ 0 \\ 1/Ml_1 \\ 0 \\ 1/Ml_2 \end{bmatrix}$$
$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} D = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{16}$$

The Linearised State Space equation is given by,

$$\begin{aligned} \dot{X} &= A_F X + B_F U \\ Y &= CX + DU \end{aligned} \tag{17}$$

Here, $X = [X\dot{X}\theta_1\dot{\theta}_1\theta_2\dot{\theta}_2]^T$ and the input will be, $U = F$. Hence, the Linearised state space for the given system is,

$$\begin{bmatrix}\dot{X}\\ \ddot{X}\\ \dot{\theta}_1\\ \ddot{\theta}_1\\ \dot{\theta}_2\\ \ddot{\theta}_2\end{bmatrix} = \begin{bmatrix}0 & 1 & 0 & 0 & 0 & 0\\ 0 & 0 & \frac{-(m_1 g)}{M} & 0 & \frac{-(m_2 g)}{M} & 0\\ 0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & \frac{-((m_1+M)g)}{Ml_1} & 0 & \frac{-(m_2 g)}{Ml_1} & 0\\ 0 & 0 & 0 & 0 & 0 & 1\\ 0 & 0 & \frac{-(m_1 g)}{Ml_2} & 0 & \frac{-((m_2+M)g)}{Ml_2} & 0\end{bmatrix}\begin{bmatrix}X\\ \dot{X}\\ \theta_1\\ \dot{\theta}_1\\ \theta_2\\ \dot{\theta}_2\end{bmatrix} + \begin{bmatrix}0\\ 1/M\\ 0\\ 1/Ml_1\\ 0\\ 1/Ml_2\end{bmatrix} F \tag{18}$$

2.4 Controllability

The controllability for a linear system can be found using the following matrix-

$$C = \begin{bmatrix}B & AB & A^2B & A^3B & A^4B & A^5B\end{bmatrix} \tag{19}$$

here, A and B are matrices computed from the linearised system A_F and B_F. The controllability is found using the *ctrb* function in MATLAB. Also, we can use the below individual matrix computation method to find the controllability.
Here, the final matrix C is computed with values of $B, AB, A^2B, A^3B, A^4B, A^5B$ which are given by,

$$B = \begin{bmatrix}0\\ 1/M\\ 0\\ 1/(Ml_1)\\ 0\\ 1/(Ml_2)\end{bmatrix} AB = \begin{bmatrix}1/M\\ 0\\ 1/(Ml_1)\\ 0\\ 1/(Ml_2)\\ 0\end{bmatrix} A^2B = \begin{bmatrix}0\\ -(gm_1)/(M^2l_1) - (gm_2)/(M^2l_2)\\ 0\\ -(g(M+m_1))/(M^2l_1^2) - (gm_2)/(M^2l_1l_2)\\ 0\\ -(g(M+m_2))/(M^2l_2^2) - (gm_1)/(M^2l_1l_2)\end{bmatrix}$$

$$A^3B = \begin{bmatrix}-(gm_1)/(M^2l_1) - (gm_2)/(M^2l_2)\\ 0\\ -(g(M+m_1))/(M^2l_1^2) - (gm_2)/(M^2l_1l_2)\\ 0\\ -(g(M+m_2))/(M^2l_2^2) - (gm_1)/(M^2l_1l_2)\\ 0\end{bmatrix}$$

$$A^4B = \begin{bmatrix}0\\ (gm_1((g(M+m_1))/(M^2l_1^2) + (gm_2)/(M^2l_1l_2)))/M + (gm_2((g(M+m_2))/(M^2l_2^2) + (gm_1)/(M^2l_1l_2)))/M\\ 0\\ (g(M+m_1)((g(M+m_1))/(M^2l_1^2) + (gm_2)/(M^2l_1l_2)))/(Ml_1) + (gm_2((g(M+m_2))/(M^2l_2^2) + (gm_1)/(M^2l_1l_2)))/(Ml_1)\\ 0\\ (g(M+m_2)((g(M+m_2))/(M^2l_2^2) + (gm_1)/(M^2l_1l_2)))/(Ml_2) + (gm_1((g(M+m_1))/(M^2l_1^2) + (gm_2)/(M^2l_1 * l_2)))/(Ml_2)\end{bmatrix}$$

$$A^5B = \begin{bmatrix}(gm_1((g(M+m_1))/(M^2l_1^2) + (gm_2)/(M^2l_1l_2)))/M + (gm_2((g(M+m_2))/(M^2l_2^2) + (gm_1)/(M^2l_1l_2)))/M\\ 0\\ (g(M+m_1)((g(M+m_1))/(M^2l_1^2) + (gm_2)/(M^2l_1l_2)))/(Ml_1) + (gm_2((g(M+m_2))/(M^2l_2^2) + (gm_1)/(M^2l_1l_2)))/(Ml_1)\\ 0\\ (g(M+m_2)((g(M+m_2))/(M^2l_2^2) + (gm_1)/(M^2l_1l_2)))/(Ml_2) + (gm_1((g(M+m_1))/(M^2l_1^2) + (gm_2)/(M^2l_1l_2)))/(Ml_2)\\ 0\end{bmatrix}$$

After the final matrix is found using the column matrices from above. The rank of Controllability matrix is 6 which is equal to the rank of the system(A). Also, The controllability for the following conditions-

- $l_1 = l_2 = k$
  When l1 and l2 are equal to some scalar quantity (say k), then the rank reduces to 4, which is not equal to the rank of the system. The System is not Controllable.
- $m_1 = m_2 = k$
  When m1 and m2 are equal to some scalar quantity (say k), then the rank remains 6, which is equal to the rank of the system. The System is Controllable.
- $M = m_1 = m_2 = k$ When M, m1 and m2 are equal to some scalar quantity (say k), then the rank remains 6, which is equal to the rank of the system. Hence, the System is Controllable.

Please refer to the Appendix to find the MATLAB code ControllabilityCheck.m for the above computations.

2.5 Implemenation of LQR Controller

2.5.1 Linearised System

Using the matrices from eq. (16), we design an LQR controller in MATLAB for the configuration, M = 1000kg, m1 = 100kg, m2 = 100kg, l1 = 20m, l2 = 10m. The controllability matrix C from eq. (19) was used to first find the controllability of the matrix for this configuration of the system. The Controllability matrix is given by-

$$C = \begin{bmatrix} 0 & 1.0000 & 0 & -0.1470 & 0 & 0.1417 \\ 1.0000 & 0 & -0.1470 & 0 & 0.1417 & 0 \\ 0 & 0.0500 & 0 & -0.0319 & 0 & 0.0227 \\ 0.0500 & 0 & -0.0319 & 0 & 0.0227 & 0 \\ 0 & 0.1000 & 0 & -0.1127 & 0 & 0.1246 \\ 0.1000 & 0 & -0.1127 & 0 & 0.1246 & 0 \end{bmatrix} \tag{20}$$

The rank of the above controllability matrix is equal to the dimensions of the system(A) i.e. 6. Hence, the linearised system is controllable.
The given system is now given subjected to initial conditions $x = [3\ 0\ 0.5\ 0\ 0.5\ 0]^T$. The following response was plotted.

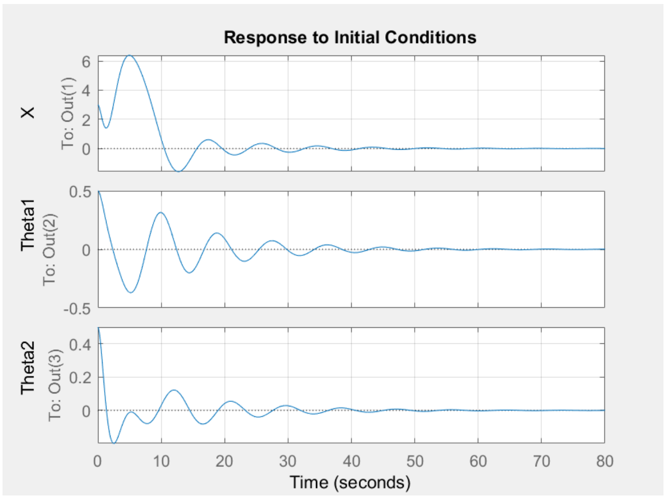

Figure 2: Linearised system subject to initial Conditions

Now, the LQR system was given a step input, the following response was obtained.

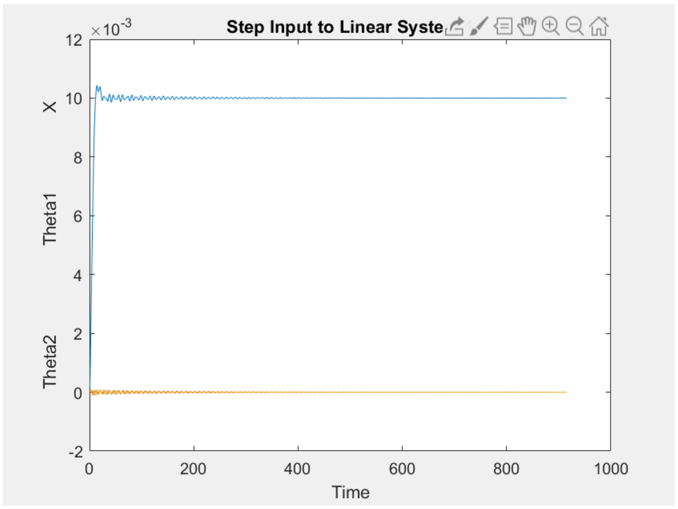

Figure 3: Linearised system subject to Step Input

The LQR controller was tuned to get a better output with the given step input. The plot below shows the updated response.

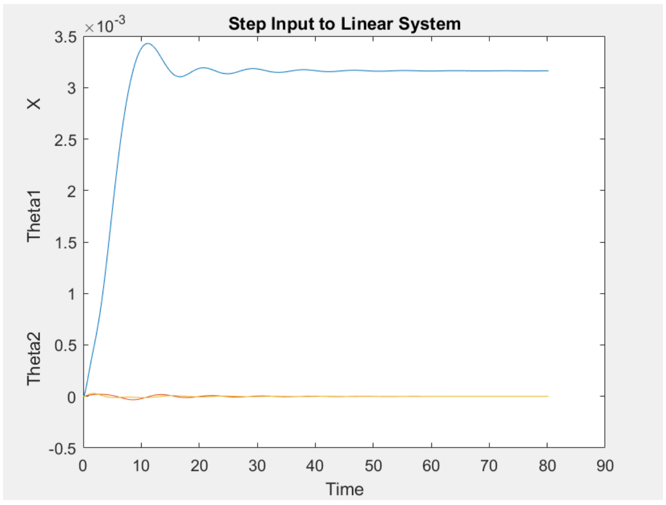

Figure 4: LQR response for Tuned system

The Matlab code for the above simulations can be accessed through the Appendix - LQR-ControlLinearised.m.

2.5.2 Non Linear System

This tuned LQR controller was also applied on the original Non linear system represented in section 2.2. The Non linear system is first computed using the *ode*45 function in MATLAB to find the output states. Using these, the tuned LQR control is applied on this system. The following graph is obtained.

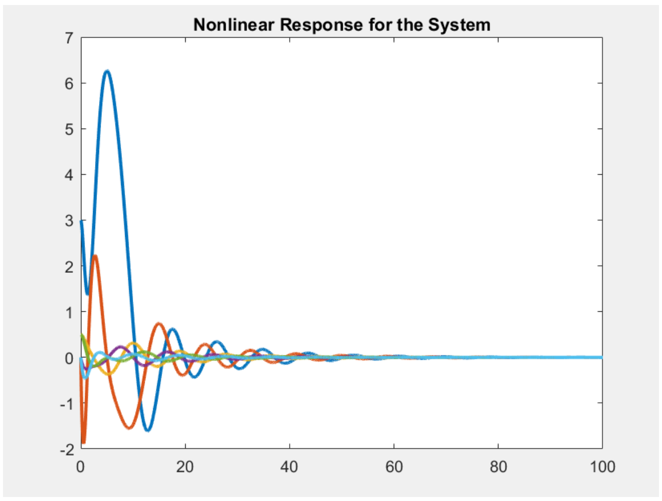

Figure 5: LQR Control on Non Linear system

The Matlab code for the above simulations can be accessed through the Appendix - LQR-ControlNonLinear.m.

2.5.3 Lyapunov's Stability test - Indirect Method

The Lyapunov's Indirect method determines the stability using the eigen values of the A_F. For the given system, the eigen values are negative. This means that the system is atleast locally stable. The final A-BK matrix eigen values are identical poles lying on the imaginary axis which means the indirect method is inconclusive.

2.6 Observability

The Observability for a linear system can be found using the following matrix-

$$O = \begin{bmatrix} C & CA & CA^2 & CA^3 & CA^4 & CA^5 \end{bmatrix}^T \tag{21}$$

here, A is a matrix computed from the linearised system A_F. The observability is found for different set of output vectors which are:

- $x(t)$
$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{22}$$

- $\theta_1(t)\ and\ \theta_2(t)$
$$C = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{23}$$

- $x(t)\ and\ \theta_2(t)$
$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{24}$$

- $x(t), \theta_1(t)\ and\ \theta_2(t)$
$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{25}$$

The observability is found using the *obsv* function in MATLAB. The final observability matrices for each of these output vectors is given below -

- $x(t)$ - The rank for the resultant O matrix matches the dimensions of the system(A), hence it is an Observable output.
- $\theta_1(t)\ and\ \theta_2(t)$ - The rank of this O matrix is less than the dimensions of the system, hence this output vector is not Observable.

- $x(t)$ *and* $\theta_2(t)$ - The rank for the resultant O matrix matches the dimensions of the system(A), hence it is an Observable output.
- $x(t), \theta_1(t)$ *and* $\theta_2(t)$ - The rank for the resultant O matrix matches the dimensions of the system(A), hence it is an Observable output.

The MATLAB code for the above computations can be found in Appendix - ObservabilityCheck.m

2.7 Luenberger Observer

From the above observable states, we will design the best Luenberger observer to determine the observable states when subject to initial conditions and unit step response.

2.7.1 Linearised System

There are three output vectors for which the system observer can be designed. We will use the Pole placement method to find the states.

- For $x(t)$ -
  These poles were chosen arbitrarily - $[-1.2 - 2.4 - 3.6 - 4.8 - 6 - 7.2]$
  Following graph shows the response to initial conditions.

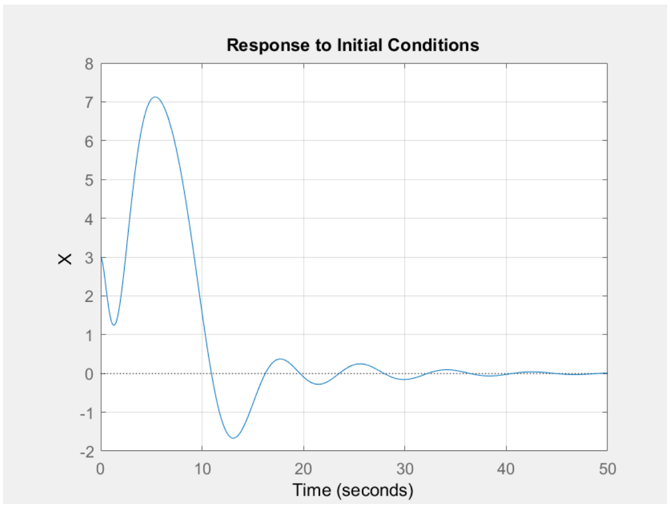

Figure 6: Luenberger Observer for $x(t)$ - Initial Conditions

  The graph below shows the response to unit step input.
  The MATLAB code can be found in Appendix - LuenbergerObsvX.m.

- For $x(t)$ *and* $\theta_2(t)$ -
  These poles were chosen arbitrarily - $[-0.1 - 0.2 - 0.3 - 0.4 - 0.5 - 0.6]$

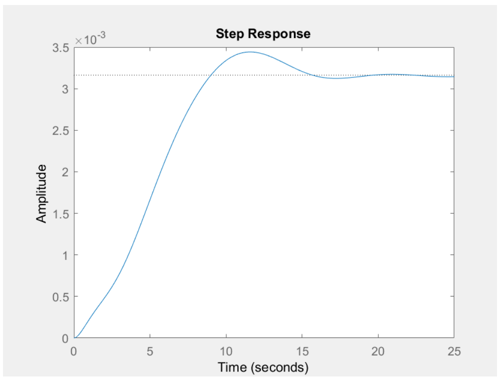

Figure 7: Luenberger Observer for $x(t)$ - Unit Step

Following graph shows the response to initial conditions.

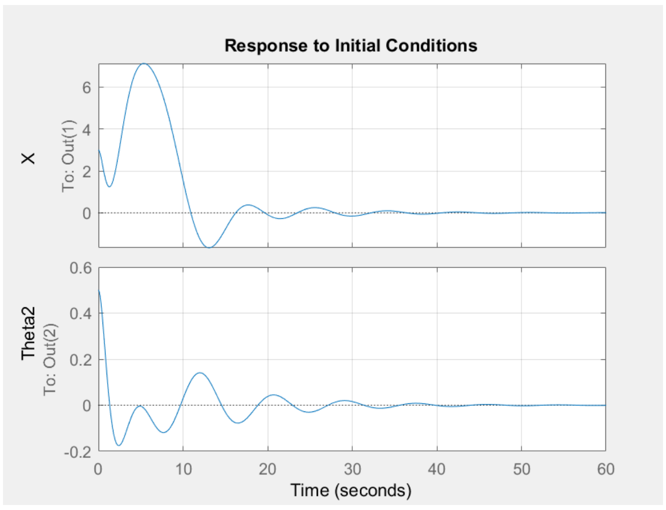

Figure 8: Luenberger Observer for $x(t)$ *and* $\theta_2(t)$ - Initial Conditions

The graph below shows the response to unit step input.

The MATLAB code can be found in Appendix - LuenbergerObsvXT2.m.

- For $x(t), \theta_1(t)$ *and* $\theta_2(t)$ -
  These poles were chosen arbitrarily - $[-1.2 - 2.4 - 3.6 - 4.8 - 6 - 7.2]$
  Following graph shows the response to initial conditions.

  The graph below shows the response to unit step input.

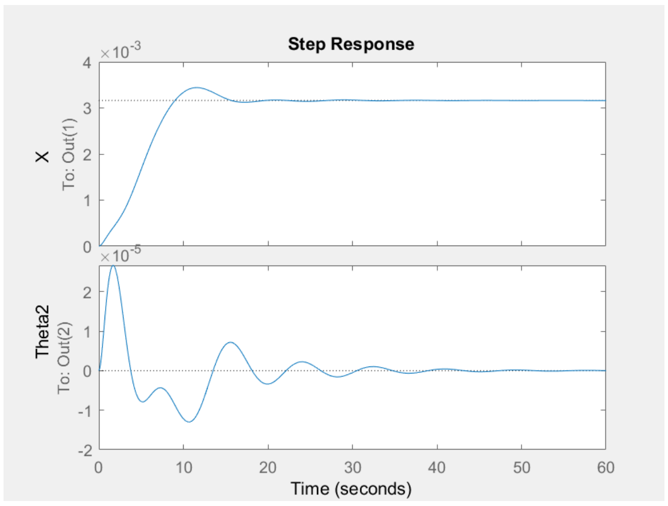

Figure 9: Luenberger Observer for $x(t)\ and\ \theta_2(t)$ - Unit Step

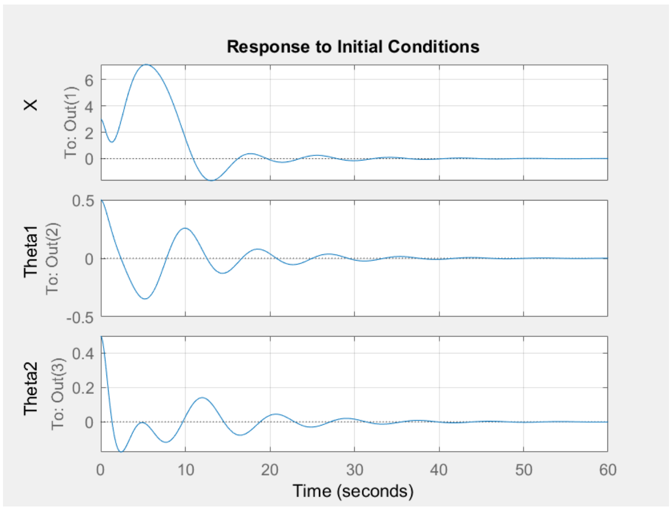

Figure 10: Luenberger Observer for $x(t),\ \theta_1(t) and\ \theta_2(t)$ - Initial Conditions

The MATLAB code can be found in Appendix - LuenbergerObsvXT12.m.

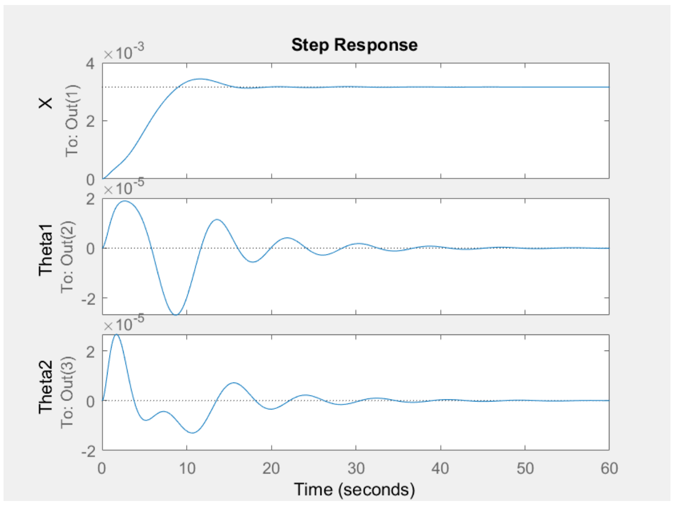

Figure 11: Luenberger Observer for $x(t),\ \theta_1(t) and\ \theta_2(t)$ - Unit Step

2.7.2 Non Linear System

The Non Linear system was designed on SIMULINK. The block diagram below shows the non linear system.

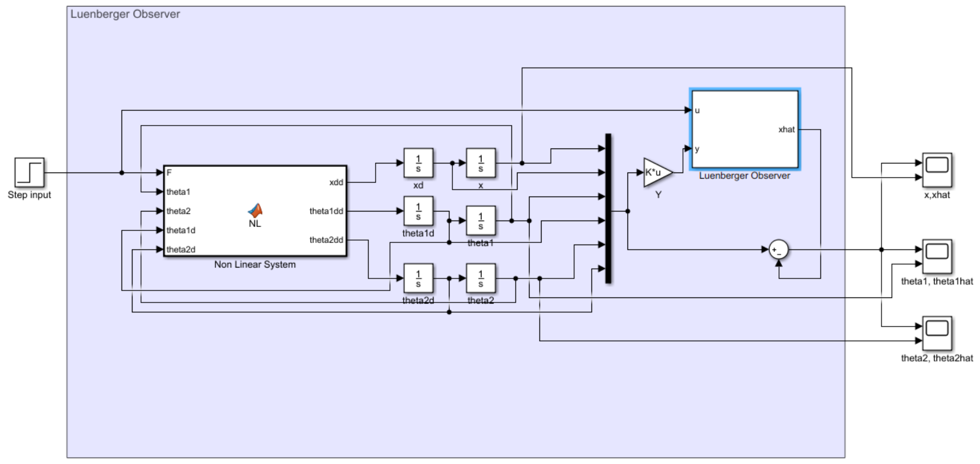

Figure 12: Luenberger Observer Block diagram

For each of the observer states, the luenberger observer was designed. The below graph shows the observer output for X-

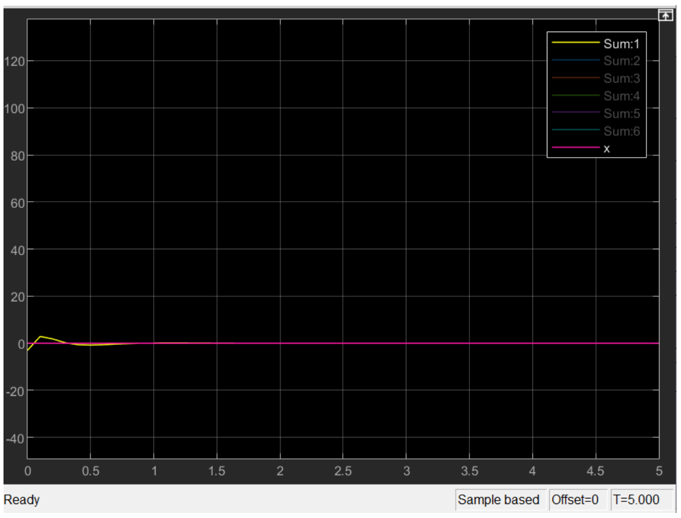

Figure 13: Luenberger Observer for $x(t)$

The following graphs show the observer output for X, Theta2-

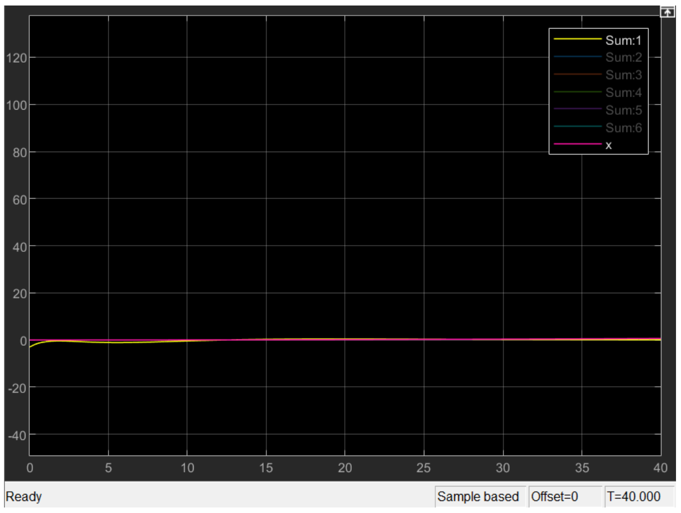

Figure 14: Luenberger Observer for $x(t), \theta_2(t)$

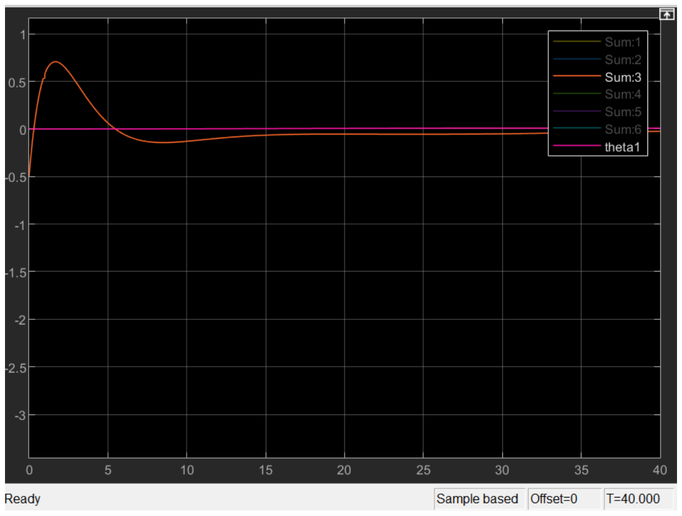

Figure 15: Luenberger Observer for $x(t), \theta_2(t)$

Finally, these graphs show the observer output for X, Theta1 ,Theta2-

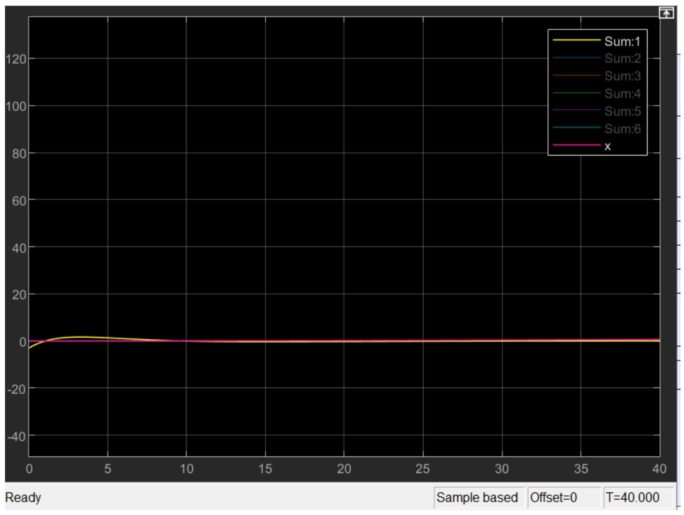

Figure 16: Luenberger Observer for $x(t),\ \theta_1(t)\ and\theta_1(t)$

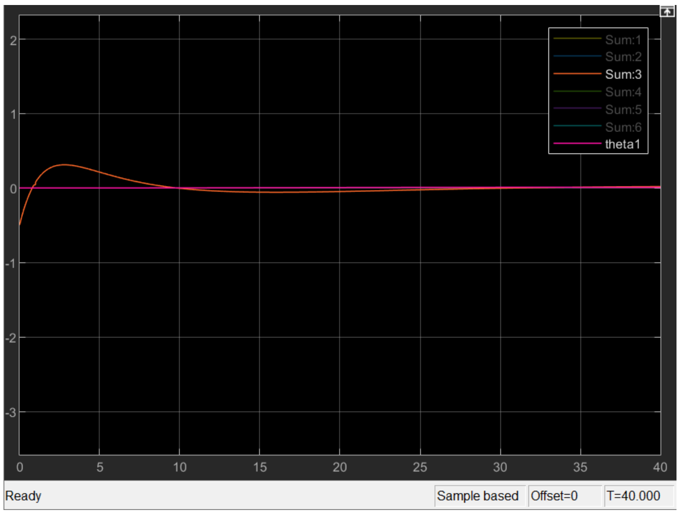

Figure 17: Luenberger Observer for $x(t),\ \theta_1(t)\ and \theta_1(t)$

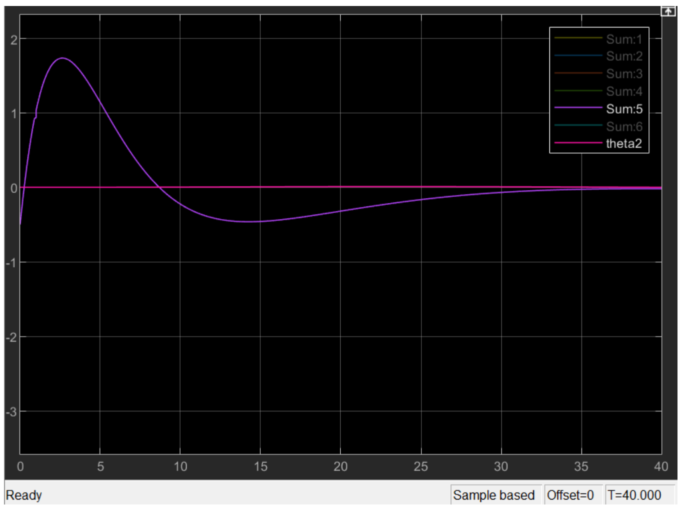

Figure 18: Luenberger Observer for $x(t),\ \theta_1(t)\ and \theta_2(t)$

2.8 Implementation of LQG Controller

The LQG[3] Controller is a combination of the Kalman filter (a Linear Quadratic Estimator) together with the LQR (Linear Quadratic Regulator) controller. The smallest observable output vector for the system will consist of only $x(t)$. The LQG is designed with only x component present in the C matrix for the system. The LQG non linear system was implemented in Simulink. The following block diagram shows the LQG Controller-

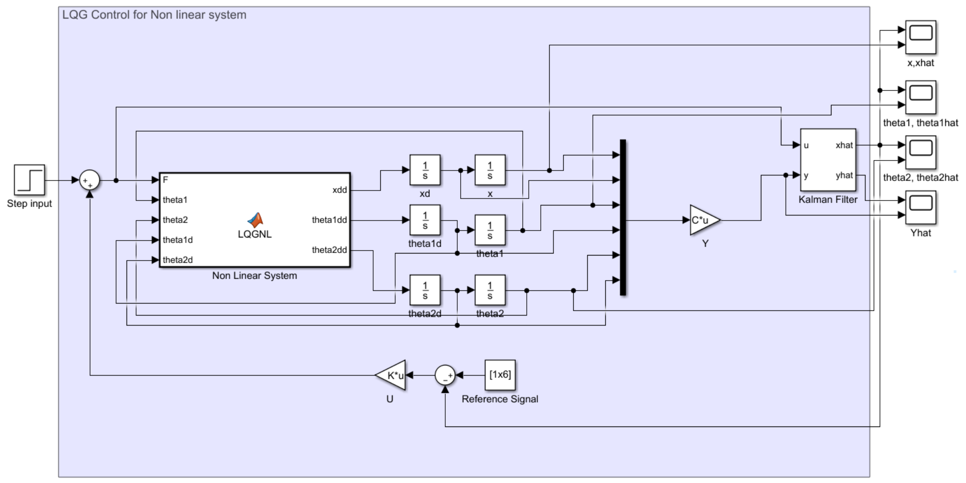

Figure 19: LQG Controller

The following graph shows the LQG on Linear as well as non linear system to show the individual outputs for $x(t),\ \theta_1(t)\ and\ \theta_2(t)$

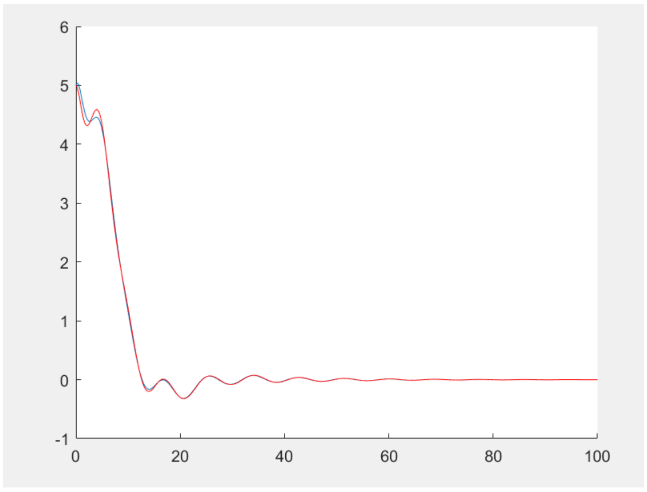

Figure 20: LQG Controller - Linear System$X$

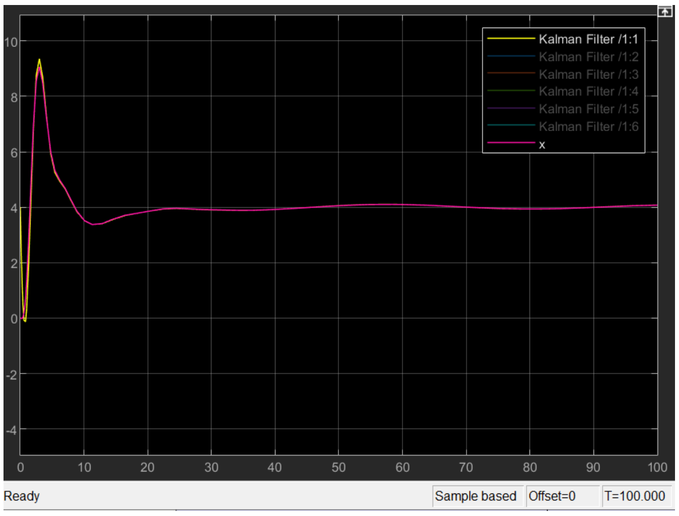

Figure 21: LQG Controller - Non Linear System$X$

The LQG controller thus designed will always have a steady state error in reference tracking since we include noise and disturbances while desiging the LQG Controller. Such a problem can be avoided by pre multiplying a gain with the reference signal.

3 Conclusion

The Double crane system was designed with the LQR and LQG controller implemented. These controllers were able to stabilise the non linear system and show relevant analysis.

References

[1] E. V. Kumar and J. Jerome. Robust lqr controller design for stabilizing and trajectory tracking of inverted pendulum. *Procedia Engineering*, 64:169–178, 2013.

[2] Wikipedia.org. Linearisation. `http://en.wikipedia.org/wiki/Linearization` Accessed: 2019-12-15.

[3] Wikipedia.org. Lqg controller. `https://en.wikipedia.org/wiki/Linear%E2%80%93quadratic%E2%80%93Gaussian_control` Accessed: 2019-12-15.

[4] Wikipedia.org. Luenberger observer. `http://en.wikipedia.org/wiki/State_observer` Accessed: 2019-12-15.

A Appendix

A.1 MATLAB Code

A.1.1 Lyapunov Indirect Method

```
%———————— Define Matrices————————%
syms m1;
syms m2;
syms M;
syms l1;
syms l2;
syms g;
syms k;
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 -(g*(m1+M))/(l1*M) 0 -m2*g/(l1*M) 0;
     0 0 0 0 0 1;
     0 0 -m1*g/(l2*M) 0 -(g*(m2+M))/(l2*M) 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 = zeros(1,1);
% D = 0;



%——————— Check Controllability for given initial conditions
    ————————%
M = 1000;
m1 = 100;
m2 = 100;
l1 = 20;
l2 = 10;
g =9.8;
AL = subs(A);
BL = subs(B);
%For the system to be controllable, the rank must be equal to
```

```
    matrix A rank
if length(AL) == rank(ctrb(AL, BL))
    disp('The system is Controllable at Eq. Point');
else
    disp("System is not Controllable at Eq. Point");
end

xInitial = [3; 0; 0.5; 0; 0.5; 0];
R = 0.00001;
% QLI = 20000000 * transpose(C) * C;
QLI = [1  0  0  0  0  0;
       0  1  0  0  0  0;
       0  0  100  0  0  0;
       0  0  0  500  0  0;
       0  0  0  0  250  0;
       0  0  0  0  0  2000];

ALI = [  0,  1,          0,  0,          0,  0;
  0,  0,     -49/50,  0,     -49/50,  0;
  0,  0,          0,  1,          0,  0;
  0,  0,  -539/1000,  0,  -49/1000,  0;
  0,  0,          0,  0,          0,  1;
  0,  0,    -49/500,  0,  -539/500,  0];
BLI = [0;
   1/1000;
        0;
  1/20000;
        0;
  1/10000];

[KLI,SLI,PLI] = lqr(ALI,BLI,QLI,R);
LIN_SYS = ss(ALI - BLI*KLI, BLI, C, D);
[y,t] = step(LIN_SYS);
plot(t,y);
title('Step Input to Linear System')
xlabel('Time')
ylabel('Theta2                    Theta1                    X')

%——————————————— Lyapunov's Indirect Method
    ————————————————————%
lambda = eig(ALI - BLI * KLI)
disp('The system is atleast locally stable as it has poles on the
    LHP');
```

```
disp(' ');
```

A.1.2 Controllability Check

```
%—————————— Define Matrices———————————%
syms m1;
syms m2;
syms M;
syms l1;
syms l2;
syms g;
syms k;
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 -(g*(m1+M))/(l1*M) 0 -m2*g/(l1*M) 0;
     0 0 0 0 0 1;
     0 0 -m1*g/(l2*M) 0 -(g*(m2+M))/(l2*M) 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 = zeros(3,1);
% D = 0;


%—————————— Check Controllability for different conditions
    ——————————%
%—————————— For l1 = l2 = k———————%
l1 = k;
l2 = k;
CT1 = subs(B);
CT2 = subs(A*B);
CT3 = subs(A*CT2);
CT4 = subs(A*CT3);
CT5 = subs(A*CT4);
CT6 = subs(A*CT5);
CTCOND1 = [CT1 CT2 CT3 CT4 CT5 CT6];
if length(A) == rank(CTCOND1)
    disp('The system is Controllable for l1 = l2');
```

```
else
    disp("System is not Controllable for l1 = l2");
end
%------------- For m1 = m2 = k--------%
m1 = k;
m2 = k;
syms l1;
syms l2;
CTM1 = subs(B);
CTM2 = subs(A*B);
CTM3 = subs(A*CTM2);
CTM4 = subs(A*CTM3);
CTM5 = subs(A*CTM4);
CTM6 = subs(A*CTM5);
CTCOND2 = [CTM1 CTM2 CTM3 CTM4 CTM5 CTM6];
if length(A) == rank(CTCOND2)
    disp('The system is Controllable for m1 = m2');
else
    disp("System is not Controllable for m1 = m2");
end
%--------------- For M = m1 = m2 = k ------------%
M = k;
m1 = k;
m2 = k;
CTMM1 = subs(B);
CTMM2 = subs(A*B);
CTMM3 = subs(A*CTMM2);
CTMM4 = subs(A*CTMM3);
CTMM5 = subs(A*CTMM4);
CTMM6 = subs(A*CTMM5);
CTCOND3 = [CTMM1 CTMM2 CTMM3 CTMM4 CTMM5 CTMM6];
if length(A) == rank(CTCOND3)
    disp('The system is Controllable for M = m1 = m2');
else
    disp("System is not Controllable for M = m1 = m2");
end
disp(' ');
```

A.1.3 LQR Control for Linearised System

```
%------------- Define Matrices--------------%
syms m1;
syms m2;
syms M;
syms l1;
syms l2;
syms g;
syms k;
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 -(g*(m1+M))/(l1*M) 0 -m2*g/(l1*M) 0;
      0 0 0 0 0 1;
      0 0 -m1*g/(l2*M) 0 -(g*(m2+M))/(l2*M) 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 = zeros(1,1);
% D = 0;



%---------- Check Controllability for given initial conditions
    ------------%
M = 1000;
m1 = 100;
m2 = 100;
l1 = 20;
l2 = 10;
g =9.8;
AL = subs(A);
BL = subs(B);
%For the system to be controllable, the rank must be equal to
    matrix A rank
if length(AL) == rank(ctrb(AL, BL))
     disp('The system is Controllable at Eq. Point');
else
     disp("System is not Controllable at Eq. Point");
end

xInitial = [3; 0; 0.5; 0; 0.5; 0];
R = 0.00001;
```

```
% QLI = 20000000 * transpose(C) * C;
QLI = [1  0  0  0  0  0;
       0  1  0  0  0  0;
       0  0  100  0  0  0;
       0  0  0  500  0  0;
       0  0  0  0  250  0;
       0  0  0  0  0  2000];

ALI = [  0,  1,           0,  0,          0,  0;
 0,  0,      -49/50,  0,     -49/50,  0;
 0,  0,           0,  1,          0,  0;
 0,  0,  -539/1000,  0,  -49/1000,  0;
 0,  0,           0,  0,          0,  1;
 0,  0,     -49/500,  0,  -539/500,  0];
BLI = [0;
  1/1000;
       0;
 1/20000;
       0;
 1/10000];

[KLI,SLI,PLI] = lqr(ALI,BLI,QLI,R);
LIN_SYS = ss(ALI - BLI*KLI, BLI, C, D);
initial(LIN_SYS, xInitial)
xlabel('Time')
ylabel('Theta2                    Theta1                    X')
grid


figure(2);
[y,t] = step(LIN_SYS);
plot(t,y);
title('Step Input to Linear System')
xlabel('Time')
ylabel('Theta2                    Theta1                    X')



R1 = 0.01;
QLI1 = [100  0  0  0  0  0;
       0  100  0  0  0  0;
       0  0  50  0  0  0;
       0  0  0  50  0  0;
```

```
        0  0  0  0  200  0;
        0  0  0  0  0  200];
[KLI1,SLI1,PLI1] = lqr(ALI,BLI,QLI1,R1);
LIN_SYS1 = ss(ALI - BLI*KLI1, BLI, C, D);
[y1,t1] = step(LIN_SYS1);
figure(3);
plot(t1,y1);
title('Step Input to Linear System 1')
xlabel('Time')
ylabel('Theta2                    Theta1                    X')


% hold on
% R2 = 0.0002;
% QLI2 = [
%         100  0  0  0  0  0;
%         0  200  0  0  0  0;
%         0  0  500  0  0  0;
%         0  0  0  500  0  0;
%         0  0  0  0  200  0;
%         0  0  0  0  0  100
%     ];
% [KLI2,SLI2,PLI2] = lqr(ALI,BLI,QLI2,R2);
% LIN_SYS2 = ss(ALI - BLI*KLI2, BLI, C, D);
% [y2,t2] = step(LIN_SYS2);
% figure(4);
% plot(t2,y2);
% title('Step Input to Linear System 2')
% xlabel('Time')
% ylabel('Theta2                    Theta1                    X')
% hold off
```

A.1.4 LQR Control for Non Linear System

```
%----------- Define same initial Conditions ------------%
m1=100;
m2=100;
M = 1000;
g = 9.8;
l1  =20;
l2  =10;
wInitial= [3; 0; 0.5; 0; 0.5; 0];
timeStep=0:0.01:100;

```

```
[timeStep,w_state]=ode45(@findWState, timeStep, wInitial);
plot(timeStep, w_state, 'linewidth', 2);
title('Nonlinear Response for the System');

function dw_state= findWState(t,w_state)
m1=100;
m2=100;
M = 1000;
g = 9.8;
l1 =20;
l2 =10;
A_NLI= [0 1 0 0 0 0 ; 0 0 -(m1*g/M) 0 -(m2*g/M) 0 ; 0 0 0 1 0 0 ;
    0 0 -(g*(M+m1))/(M*l1) 0 -(m2*g)/(M*l1) 0 ;0 0 0 0 0 1; 0 0
    -(m1*g)/(M*l2) 0 -(g*(M+m2))/(M*l2) 0];
B_NLI= [ 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];
R = 0.00001;
QNLI = [1 0 0 0 0 0;
        0 1 0 0 0 0;
        0 0 100 0 0 0;
        0 0 0 500 0 0;
        0 0 0 0 250 0;
        0 0 0 0 0 2000];
K_NLI=lqr(A_NLI,B_NLI,QNLI,R);
eig(A_NLI-B_NLI * K_NLI);
u = -K_NLI * w_state;
dw_state = zeros(6,1);
dw_state(1)= w_state(2);
dw_state(2)= -((-u) + m1*l1*w_state(4)^2*sin(w_state(3)) + m1*g*
    sin(w_state(3))*cos(w_state(3))+ m2*l2*w_state(6)^2*sin(
    w_state(5))+m2*g*sin(w_state(5))*cos(w_state(5)))/(M+m1*sin(
    w_state(3)^2)+ m2*sin(w_state(5)^2));
dw_state(3)= w_state(4);
dw_state(4)= -((-u) +(M+m1)*g*sin(w_state(3)) + m1*l1*w_state(4)
    ^2*sin(w_state(3))*cos(w_state(3)) + m2*l2*w_state(6)^2*sin(
    w_state(5))*cos(w_state(3)) + m2*g*sin(w_state(5))*cos(w_state
    (3)-w_state(5)))/(( M+ m1*sin(w_state(3)^2)+ m2*sin(w_state(5)
    ^2))*l1);
dw_state(5)= w_state(6);
dw_state(6)= -((-u) + m1*l1*w_state(4)^2*sin(w_state(3))*cos(
    w_state(5)) +m1*g*sin(w_state(3))*cos(w_state(3)-w_state(5)) +
     (M+m1)*g*sin(w_state(5))+m2*l2*w_state(6)^2*sin(w_state(5))*
    cos(w_state(5)))/(( M+ m1*sin(w_state(3)^2)+ m2*sin(w_state(5)
    ^2))*l2);
end
```

A.1.5 Observability Check

```
%---------------- Linearised System Matrices -----------------%
ALI = [ 0, 1,           0, 0,          0, 0;
 0, 0,      -49/50, 0,     -49/50, 0;
 0, 0,           0, 1,          0, 0;
 0, 0, -539/1000, 0, -49/1000, 0;
 0, 0,           0, 0,          0, 1;
 0, 0,     -49/500, 0, -539/500, 0];
BLI = [0;
   1/1000;
         0;
  1/20000;
         0;
  1/10000];

%---------------- Observability for x(t) -----------------%
CX = [1  0  0  0  0  0;
        0  0  0  0  0  0;
        0  0  0  0  0  0;
      ];
%For the system to be controllable, the rank must be equal to
    matrix A rank
if length(ALI) == rank(obsv(ALI, CX))
     disp('The system is Observable for x(t)');
else
     disp("System is not Observable for x(t)");
end
%---------------- Observability for theta1(t), theta2(t)
    -----------------%
CT12 = [0  0  0  0  0  0;
          0  0  1  0  0  0;
          0  0  0  0  1  0;
      ];
%For the system to be controllable, the rank must be equal to
    matrix A rank
if length(ALI) == rank(obsv(ALI, CT12))
```

```
    disp('The system is Observable for theta1(t) and theta2(t)');
else
    disp("System is not Observable for theta1(t) and theta2(t)");
end
%———————— Observability for x(t), theta2(t) ————————%
CXT1 = [1 0 0 0 0 0;
        0 0 0 0 0 0;
        0 0 0 0 1 0;
    ];
%For the system to be controllable, the rank must be equal to
    matrix A rank
if length(ALI) == rank(obsv(ALI, CXT1))
    disp('The system is Observable for x(t) and theta1(t)');
else
    disp("System is not Observable for x(t) and theta1(t)");
end
%———————— Observability for x(t), theta1(t), theta2(t)
    ————————%
CXT12 = [1 0 0 0 0 0;
         0 0 1 0 0 0;
         0 0 0 0 1 0;
    ];
%For the system to be controllable, the rank must be equal to
    matrix A rank
if length(ALI) == rank(obsv(ALI, CXT12))
    disp('The system is Observable for x(t), theta1(t) and theta2
       (t)');
else
    disp("System is not Observable for x(t), theta1(t) and theta2
       (t)");
end
disp(' ');




%———————— Luenberger observer ————————%
% [LO, PO, EO] = lqe(ALI,BLI,CX,QLI',R);
```

A.1.6 Luenberger Observer for $x(t)$

```
%———————— Linearised System Matrices ————————%
A_OBX = [ 0, 1,          0, 0,         0, 0;
 0, 0,     -49/50, 0,    -49/50, 0;
 0, 0,          0, 1,         0, 0;
 0, 0, -539/1000, 0, -49/1000, 0;
 0, 0,          0, 0,         0, 1;
 0, 0,   -49/500, 0, -539/500, 0];
B_OBX = [0;
   1/1000;
        0;
 1/20000;
        0;
 1/10000];

%------------ Observability for x(t) ---------------%
CX = [1 0 0 0 0 0;
%      0 0 0 0 0 0;
%      0 0 0 0 0 0;
    ];
%For the system to be controllable, the rank must be equal to
    matrix A rank
if length(A_OBX) == rank(obsv(A_OBX, CX))
    disp('The system is Observable for x(t)');
else
    disp("System is not Observable for x(t)");
end

%------------- Observer -----------------------%
R_OBX = 0.00001;
Q_OBX = [1 0 0 0 0 0;
         0 1 0 0 0 0;
         0 0 500 0 0 0;
         0 0 0 700 0 0;
         0 0 0 0 100 0;
         0 0 0 0 0 2500];
K_OBX = lqr(A_OBX, B_OBX, Q_OBX, R_OBX);
xInitial = [3; 0; 0.5; 0; 0.5; 0; 0; 0; 0; 0; 0; 0];
POLES = [-1.2 -2.4 -3.6 -4.8 -6 -7.2];
L = place(transpose(A_OBX), transpose(CX), POLES);
L = transpose(L);
EIGEN_VALUES = eig(A_OBX - L*CX);
Ac = [(A_OBX-B_OBX*K_OBX) (B_OBX*K_OBX);(zeros(size(A_OBX))) (
    A_OBX - L*CX)];
Bc = [B_OBX; zeros(size(B_OBX))];
Cc = [CX zeros(size(CX))];
Dc = 0;
```

```
OBS_SYS = ss(Ac, Bc, Cc, Dc)
initial(OBS_SYS, xInitial);
xlabel('Time')
ylabel('X')
grid

hold on
figure(2);
step(OBS_SYS)
hold off
```

A.1.7 Luenberger Observer for $x(t)$ $and\theta_2(t)$

```
%—————————— Linearised System Matrices ————————————%
A_OBXT2 = [ 0, 1,          0, 0,         0, 0;
 0, 0,     -49/50, 0,    -49/50, 0;
 0, 0,          0, 1,         0, 0;
 0, 0, -539/1000, 0, -49/1000, 0;
 0, 0,          0, 0,         0, 1;
 0, 0,    -49/500, 0, -539/500, 0];
B_OBXT2 = [0;
   1/1000;
        0;
 1/20000;
        0;
 1/10000];

%—————————— Observability for theta1(t), theta2(t)
   ————————————%
CXT2 = [1 0 0 0 0 0;
        0 0 0 0 1 0;
     ];
%For the system to be controllable, the rank must be equal to
    matrix A rank
if length(A_OBXT2) == rank(obsv(A_OBXT2, CXT2))
    disp('The system is Observable for theta1(t) and theta2(t)');
else
    disp("System is not Observable for theta1(t) and theta2(t)");
end

%——————————— Observer ———————————————%
R_OBXT2 = 0.00001;
Q_OBXT2 = [1 0 0 0 0 0;
           0 1 0 0 0 0;
```

```
            0  0  500  0  0  0;
            0  0  0  700  0  0;
            0  0  0  0  100  0;
            0  0  0  0  0  2500];
K_OBXT2 = lqr(A_OBXT2, B_OBXT2, Q_OBXT2, R_OBXT2);
xInitial = [3; 0; 0.5; 0; 0.5; 0; 0; 0; 0; 0; 0; 0];
POLES = [-0.1 -0.2 -0.3 -0.4 -0.5 -0.6];
L = place(transpose(A_OBXT2), transpose(CXT2), POLES);
L = transpose(L);
EIGEN_VALUES = eig(A_OBXT2 - L*CXT2);
Ac = [(A_OBXT2-B_OBXT2*K_OBXT2) (B_OBXT2*K_OBXT2);(zeros(size(
    A_OBXT2))) (A_OBXT2 - L*CXT2)];
Bc = [B_OBXT2; zeros(size(B_OBXT2))];
Cc = [CXT2 zeros(size(CXT2))];
Dc = 0;
OBS_SYS = ss(Ac, Bc, Cc, Dc)
initial(OBS_SYS, xInitial);
xlabel('Time')
ylabel(' Theta2                              X')
grid

hold on
figure(2);
step(OBS_SYS)
xlabel('Time')
ylabel(' Theta2                              X')
hold off
```

A.1.8 Luenberger Observer for $x(t)$, $\theta_1(t)$ $and\theta_2(t)$

```
%------------ Linearised System Matrices ------------%
A_OBXT12 = [ 0, 1,          0, 0,          0, 0;
 0, 0,     -49/50, 0,     -49/50, 0;
 0, 0,          0, 1,          0, 0;
 0, 0, -539/1000, 0, -49/1000, 0;
 0, 0,          0, 0,          0, 1;
 0, 0,    -49/500, 0, -539/500, 0];
B_OBXT12 = [0;
   1/1000;
        0;
 1/20000;
        0;
 1/10000];

```

```
%—————————— Observability for theta1(t), theta2(t)
    ————————————%
CT12 = [1  0  0  0  0  0
          0  0  1  0  0  0;
          0  0  0  0  1  0;
      ];
%For the system to be controllable, the rank must be equal to
    matrix A rank
if length(ALI) == rank(obsv(ALI, CT12))
    disp('The system is Observable for theta1(t) and theta2(t)');
else
    disp("System is not Observable for theta1(t) and theta2(t)");
end

%——————————— Observer ——————————————%
R_OBXT12 = 0.00001;
Q_OBXT12 = [1  0  0  0  0  0;
            0  1  0  0  0  0;
            0  0  500  0  0  0;
            0  0  0  700  0  0;
            0  0  0  0  100  0;
            0  0  0  0  0  2500];
K_OBXT12 = lqr(A_OBXT12, B_OBXT12, Q_OBXT12, R_OBXT12);
xInitial = [3; 0; 0.5; 0; 0.5; 0; 0; 0; 0; 0; 0; 0];
POLES = [-1.2 -2.4 -3.6 -4.8 -6 -7.2];
L = place(transpose(A_OBXT12), transpose(CT12), POLES);
L = transpose(L);
EIGEN_VALUES = eig(A_OBXT12 - L*CT12);
Ac = [(A_OBXT12-B_OBXT12*K_OBXT12) (B_OBXT12*K_OBXT12);(zeros(
    size(A_OBXT12))) (A_OBXT12 - L*CT12)];
Bc = [B_OBXT12; zeros(size(B_OBXT12))];
Cc = [CT12 zeros(size(CT12))];
Dc = 0;
OBS_SYS = ss(Ac, Bc, Cc, Dc)
initial(OBS_SYS, xInitial);
xlabel('Time')
ylabel(' Theta2                    Theta1                    X')
grid

hold on
figure(2);
step(OBS_SYS)
xlabel('Time')
```

```
ylabel(' Theta2                          Theta1                          X')
hold off
```

A.1.9 Lyapunov Indirect Method

```
%------------- Define Matrices---------------%
syms m1;
syms m2;
syms M;
syms l1;
syms l2;
syms g;
syms k;
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 -(g*(m1+M))/(l1*M) 0 -m2*g/(l1*M) 0;
      0 0 0 0 0 1;
      0 0 -m1*g/(l2*M) 0 -(g*(m2+M))/(l2*M) 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 = zeros(1,1);
% D = 0;



%---------- Check Controllability for given initial conditions
    ------------%
M = 1000;
m1 = 100;
m2 = 100;
l1 = 20;
l2 = 10;
g =9.8;
AL = subs(A);
BL = subs(B);
%For the system to be controllable, the rank must be equal to
    matrix A rank
```

```
if length(AL) == rank(ctrb(AL, BL))
    disp('The system is Controllable at Eq. Point');
else
    disp("System is not Controllable at Eq. Point");
end

xInitial = [3; 0; 0.5; 0; 0.5; 0];
R = 0.00001;
% QLI = 20000000 * transpose(C) * C;
QLI = [1  0  0  0  0  0;
       0  1  0  0  0  0;
       0  0  100  0  0  0;
       0  0  0  500  0  0;
       0  0  0  0  250  0;
       0  0  0  0  0  2000];

ALI = [ 0, 1,          0, 0,         0, 0;
 0, 0,     -49/50, 0,    -49/50, 0;
 0, 0,          0, 1,         0, 0;
 0, 0, -539/1000, 0, -49/1000, 0;
 0, 0,          0, 0,         0, 1;
 0, 0,    -49/500, 0, -539/500, 0];
BLI = [0;
  1/1000;
       0;
 1/20000;
       0;
 1/10000];

[KLI,SLI,PLI] = lqr(ALI,BLI,QLI,R);
LIN_SYS = ss(ALI - BLI*KLI, BLI, C, D);
[y,t] = step(LIN_SYS);
plot(t,y);
title('Step Input to Linear System')
xlabel('Time')
ylabel('Theta2                    Theta1                    X')

%------------------ Lyapunov's Indirect Method
   -------------------------%
lambda = eig(ALI - BLI * KLI)
disp('The system is atleast locally stable as it has poles on the
    LHP');
disp(' ');
```