ENPM667 Project 2

December 20, 2024

Tim Sweeney
tsweene1@umd.edu

Grayson Gilbert
ggilbert@umd.edu

1 Introduction

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 constitutes 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.

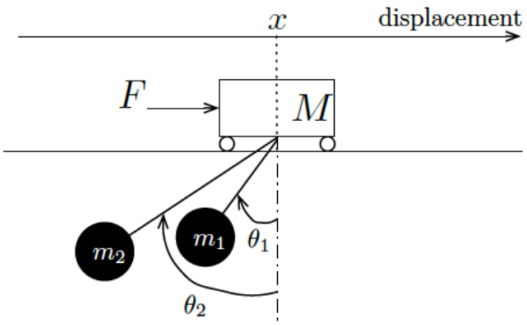

Figure 1: Diagram of the crane and mass assembly.

Known Variables:
External force: $\boldsymbol{F}$
Mass of the frictionless cart: $\boldsymbol{M}$
Load mass 1: $\boldsymbol{m_1}$
Load mass 2: $\boldsymbol{m_2}$
Cable length 1: $\boldsymbol{l_1}$
Cable length 2: $\boldsymbol{l_2}$

2 Obtaining Equations of Motion for the System

The given problem is a non-linear system. One way to obtain the equations of motion of a non-linear system is by leveraging the Euler-Lagrange method. This method uses kinetic and potential energies within the system to derive the appropriate equations of motion.

The Euler-Lagrange equation is as follows.

$$\frac{\partial \mathcal{L}}{\partial q_i} - \frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i}\right) = f, \tag{1}$$

For the non-linear crane mass system, the kinetic energy can be defined as

$$T = \frac{1}{2}M\dot{x}^2 + \frac{1}{2}m_1\left(\dot{x}_1^2 + \dot{y}_1^2\right) + \frac{1}{2}m_2\left(\dot{x}_2^2 + \dot{y}_2^2\right), \tag{2}$$

where

$$x_i = x - l_i\sin(\theta_i), \quad y_i = l_i(1-\cos(\theta_i)), \quad \text{for } i = 1, 2$$

$$\dot{x}_i = \dot{x} - l_i\dot{\theta}_i\cos(\theta_i), \quad \dot{y}_i = l_i\dot{\theta}_i\sin(\theta_i).$$

Using these values in 2, T can be rewritten as follows.

$$T = \frac{1}{2}(M + m_1 + m_2)\dot{x}^2 + \frac{1}{2}m_1l_1^2\dot{\theta}_1^2 + \frac{1}{2}m_2l_2^2\dot{\theta}_2^2 - m_1l_1\cos(\theta_1)\dot{\theta}_1\dot{x} - m_2l_2\cos(\theta_2)\dot{\theta}_2\dot{x} \tag{3}$$

The potential energy for the system, V, is defined as

$$V = m_1gl_1(1-\cos(\theta_1)) + m_2gl_2(1-\cos(\theta_2)). \tag{4}$$

The Lagrangian, $\mathcal{L}$, for a system is defined as

$$\mathcal{L} = T - V \tag{5}$$

The following equations will define the nonlinear dynamics of this specific system

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{x}}\right) - \frac{\partial \mathcal{L}}{\partial x} = F, \tag{6}$$

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\theta}_1}\right) - \frac{\partial \mathcal{L}}{\partial \theta_1} = 0, \tag{7}$$

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\theta}_2}\right) - \frac{\partial \mathcal{L}}{\partial \theta_2} = 0. \tag{8}$$

Solving the individual components of 6 gives the following

$$\frac{\partial \mathcal{L}}{\partial x} = 0, \tag{9}$$

$$\frac{\partial \mathcal{L}}{\partial \dot{x}} = (M + m_1 + m_2)\dot{x} - m_1l_1\cos(\theta_1)\dot{\theta}_1 - m_2l_2\cos(\theta_2)\dot{\theta}_2, \tag{10}$$

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{x}}\right) = (M+m_1+m_2)\ddot{x}+m_1 l_1 \sin(\theta_1)\dot{\theta}_1^2+m_2 l_2 \sin(\theta_2)\dot{\theta}_2^2-m_1 l_1 \cos(\theta_1)\ddot{\theta}_1-m_2 l_2 \cos(\theta_2)\ddot{\theta}_2 \quad (11)$$

Next, solving the individual components of 7 gives the following

$$\frac{\partial \mathcal{L}}{\partial \theta_1} = m_1 l_1 \sin(\theta_1)\dot{\theta}_1 \dot{x} - m_1 g l_1 \sin(\theta_1), \quad (12)$$

$$\frac{\partial \mathcal{L}}{\partial \dot{\theta}_1} = m_1 l_1^2 \dot{\theta}_1 - m_1 l_1 \cos(\theta_1)\dot{x}, \quad (13)$$

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\theta}_1}\right) = m_1 l_1^2 \ddot{\theta}_1 + m_1 l_1 \sin(\theta_1)\dot{\theta}_1 \dot{x} - m_1 l_1 \cos(\theta_1)\ddot{x} \quad (14)$$

Finally, solving the individual components of 8 gives the following

$$\frac{\partial \mathcal{L}}{\partial \theta_2} = m_2 l_2 \sin(\theta_2)\dot{\theta}_2 \dot{x} - m_2 g l_2 \sin(\theta_2), \quad (15)$$

$$\frac{\partial \mathcal{L}}{\partial \dot{\theta}_2} = m_2 l_2^2 \dot{\theta}_2 - m_2 l_2 \cos(\theta_2)\dot{x}, \quad (16)$$

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\theta}_2}\right) = m_2 l_2^2 \ddot{\theta}_2 + m_2 l_2 \sin(\theta_2)\dot{\theta}_2 \dot{x} - m_2 l_2 \cos(\theta_2)\ddot{x} \quad (17)$$

Now it is time to put it all together. Plugging in the individual components of 6, 7, and 8 into their respective equations will yield the following

$$(M+m_1+m_2)\ddot{x} + m_1 l_1 \sin(\theta_1)\dot{\theta}_1^2 + m_2 l_2 \sin(\theta_2)\dot{\theta}_2^2 - m_1 l_1 \cos(\theta_1)\ddot{\theta}_1 - m_2 l_2 \cos(\theta_2)\ddot{\theta}_2 = F \quad (18)$$

$$l_1 \ddot{\theta_1} - \cos(\theta_1)\ddot{x} + g \sin(\theta_1) = 0 \quad (19)$$

$$l_2 \ddot{\theta}_2 \ddot{x} + g \sin(\theta_2) = 0 \quad (20)$$

Then, we can rearrange equations 18, 19, and 20, which will define the equations of motion for the nonlinear system.

$$\begin{aligned}\ddot{x} = \frac{1}{M + m_1 \sin^2(\theta_1) + m_2 \sin^2(\theta_2)} \\ \Big[F - m_1 l_1 \sin(\theta_1)\dot{\theta}_1^2 - m_2 l_2 \sin(\theta_2)\dot{\theta}_2^2 \\ -m_1 g \cos(\theta_1)\sin(\theta_1) - m_2 g \cos(\theta_2)\sin(\theta_2)\Big],\end{aligned} \quad (21)$$

$$\ddot{\theta}_1 = \frac{1}{l_1}\left[\cos(\theta_1)\ddot{x} - g \sin(\theta_1)\right], \quad (22)$$

$$\ddot{\theta}_2 = \frac{1}{l_2}\left[\cos(\theta_2)\ddot{x} - g \sin(\theta_2)\right]. \quad (23)$$

The next step is to substitute $\ddot{x}$ into the non-linear equations of motion for 22 and 23. This will give the following equations.

$$\ddot{\theta}_1 = \frac{\cos(\theta_1)}{l_1}\left[\frac{F - m_1 l_1 \sin(\theta_1)\dot{\theta}_1^2 - m_2 l_2 \sin(\theta_2)\dot{\theta}_2^2 - m_1 g \cos(\theta_1)\sin(\theta_1) - m_2 g \cos(\theta_2)\sin(\theta_2)}{M + m_1 \sin^2(\theta_1) + m_2 \sin^2(\theta_2)}\right] - \frac{g \sin(\theta_1)}{l_1} \tag{24}$$

$$\ddot{\theta}_2 = \frac{\cos(\theta_2)}{l_2}\left[\frac{F - m_1 l_1 \sin(\theta_1)\dot{\theta}_1^2 - m_2 l_2 \sin(\theta_2)\dot{\theta}_2^2 - m_1 g \cos(\theta_1)\sin(\theta_1) - m_2 g \cos(\theta_2)\sin(\theta_2)}{M + m_1 \sin^2(\theta_1) + m_2 \sin^2(\theta_2)}\right] - \frac{g \sin(\theta_2)}{l_2} \tag{25}$$

We have now reached the point where the non-linear dynamics of the system are appropriately defined. The system can then be represented in the State Space Form using the State Space equation.

$$\dot{X} = AX + BU \tag{26}$$

The State Space Representation of the crane mass system is defined as

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

3 Linearize the System about the Equilibrium

The state space equations for this nonlinear system are written as $\dot{X} = f(X(t), U(t))$ where

$$X(t) = \begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta_1} \\ \theta_2 \\ \dot{\theta_2} \end{bmatrix}, \quad U(t) = \begin{bmatrix} F \end{bmatrix}, \quad \dot{X}(t) = \begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta_1} \\ \ddot{\theta_1} \\ \dot{\theta_2} \\ \ddot{\theta_2} \end{bmatrix} \tag{28}$$

The equilibrium point of this system is ($x = 0,\ \dot{x} = 0,\ \theta_1 = 0,\ \dot{\theta_1} = 0,\ \theta_2 = 0,\ \dot{\theta_2} = 0$). In any other configuration the masses are in an unstable state, and will begin moving if not already moving. We linearize the system about this equilibrium point using **Lyapunov's Indirect Method**.

First we take the Jacobian of f.

$$A_F = \nabla_x | f(x,u), \quad B_F = \nabla_u | f(x,u) \tag{29}$$

$$A_f = \begin{bmatrix} \frac{\delta f_1}{x_1} & \frac{\delta f_1}{x_2} & \frac{\delta f_1}{x_3} & \frac{\delta f_1}{x_4} & \frac{\delta f_1}{x_5} & \frac{\delta f_1}{x_6} \\ \frac{\delta f_2}{x_1} & \frac{\delta f_2}{x_2} & \frac{\delta f_2}{x_3} & \frac{\delta f_2}{x_4} & \frac{\delta f_2}{x_5} & \frac{\delta f_2}{x_6} \\ \frac{\delta f_3}{x_1} & \frac{\delta f_3}{x_2} & \frac{\delta f_3}{x_3} & \frac{\delta f_3}{x_4} & \frac{\delta f_3}{x_5} & \frac{\delta f_3}{x_6} \\ \frac{\delta f_4}{x_1} & \frac{\delta f_4}{x_2} & \frac{\delta f_4}{x_3} & \frac{\delta f_4}{x_4} & \frac{\delta f_4}{x_5} & \frac{\delta f_4}{x_6} \\ \frac{\delta f_5}{x_1} & \frac{\delta f_5}{x_2} & \frac{\delta f_5}{x_3} & \frac{\delta f_5}{x_4} & \frac{\delta f_5}{x_5} & \frac{\delta f_5}{x_6} \\ \frac{\delta f_6}{x_1} & \frac{\delta f_6}{x_2} & \frac{\delta f_6}{x_3} & \frac{\delta f_6}{x_4} & \frac{\delta f_6}{x_5} & \frac{\delta f_6}{x_6} \end{bmatrix} \tag{30}$$

The equations for the Jacobian are too long to be described here. However they can be generated using the code in the Appendix: Part B. After finding the partial derivatives of the Jacobian, we then evaluate them at the equilibrium point ($x = 0,\ \dot{x} = 0,\ \theta_1 = 0,\ \dot{\theta}_1 = 0,\ \theta_2 = 0,\ \dot{\theta}_2 = 0$) to give us the linearized state space equations.

$$A_f = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{gm_1}{M} & 0 & \frac{-gm_2}{M} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{-g(M+m_1)}{Ml_1} & 0 & \frac{-gm_2}{Ml_1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{-gm_1}{Ml_2} & 0 & \frac{-g(M+m_2)}{Ml_2} & 0 \end{bmatrix}, B_f = \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{Ml_1} \\ 0 \\ \frac{1}{Ml_2} \end{bmatrix} \tag{31}$$

4 Obtain the conditions of M, m_1, m_2, l_1, l_2 for which the linearized system is controllable

Now that we have linearized the crane mass system, we will next check for controllability. In the previous section, the matrices of A and B are now linear time invariant matrices. This allows for the controllability matrix to be found using

$$\mathrm{C} = \begin{bmatrix} B & AB & A^2B & A^3B & A^4B & A^5B \end{bmatrix} \tag{32}$$

$$\mathrm{rank}[\mathrm{C}] = n \tag{33}$$

where, C, is defined as the controllability matrix of the system, and n is the number of state variables in the system. In our case, there are six state variables. The matrix B has dimensions 6x1, which means the following matrices AB, A^2B, etc. also have dimensions 6x1. Combining the 6 individual matrices together will form C, which has dimensions 6x6. C is a square matrix, so we can determine its rank by calculating the determinant. If the determinant of C is **not** equal to 0, then the matrix is considered full rank and equal to n. The following calculations will be preformed using MATLAB, and the code will be provide at the end of the report.

$$B = \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{Ml_1} \\ 0 \\ \frac{1}{Ml_2} \end{bmatrix}, \quad AB = \begin{bmatrix} \frac{1}{M} \\ 0 \\ \frac{1}{Ml_1} \\ 0 \\ \frac{1}{Ml_2} \\ 0 \end{bmatrix}, \quad A^2B = \begin{bmatrix} 0 \\ -\frac{gm_1}{M^2l_1} - \frac{gm_2}{M^2l_2} \\ 0 \\ -\frac{g(M+m_1)}{M^2l_1^2} - \frac{gm_2}{M^2l_1l_2} \\ 0 \\ -\frac{g(M+m_2)}{M^2l_2^2} - \frac{gm_1}{M^2l_1l_2} \end{bmatrix} \tag{34}$$

$$A^3B = \begin{bmatrix} -\frac{gm_1}{M^2l_1} - \frac{gm_2}{M^2l_2} \\ 0 \\ -\frac{g(M+m_1)}{M^2l_1^2} - \frac{gm_2}{M^2l_1l_2} \\ 0 \\ -\frac{g(M+m_2)}{M^2l_2^2} - \frac{gm_1}{M^2l_1l_2} \\ 0 \end{bmatrix}, \quad A^4B = \begin{bmatrix} 0 \\ \frac{gm_1\left(\frac{g(M+m_1)}{M^2l_1^2} + \frac{gm_2}{M^2l_1l_2}\right)}{M} + \frac{gm_2\left(\frac{g(M+m_2)}{M^2l_2^2} + \frac{gm_1}{M^2l_1l_2}\right)}{M} \\ 0 \\ \frac{g(M+m_1)\left(\frac{g(M+m_1)}{M^2l_1^2} + \frac{gm_2}{M^2l_1l_2}\right)}{Ml_1} + \frac{gm_2\left(\frac{g(M+m_2)}{M^2l_2^2} + \frac{gm_1}{M^2l_1l_2}\right)}{Ml_1} \\ 0 \\ \frac{g(M+m_2)\left(\frac{g(M+m_2)}{M^2l_2^2} + \frac{gm_1}{M^2l_1l_2}\right)}{Ml_2} + \frac{gm_1\left(\frac{g(M+m_1)}{M^2l_1^2} + \frac{gm_2}{M^2l_1l_2}\right)}{Ml_2} \end{bmatrix} \tag{35}$$

$$A^5B = \begin{bmatrix} \frac{gm_1\left(\frac{g(M+m_1)}{M^2l_1^2}+\frac{gm_2}{M^2l_1l_2}\right)}{M} + \frac{gm_2\left(\frac{g(M+m_2)}{M^2l_2^2}+\frac{gm_1}{M^2l_1l_2}\right)}{M} \\ 0 \\ \frac{g(M+m_1)\left(\frac{g(M+m_1)}{M^2l_1^2}+\frac{gm_2}{M^2l_1l_2}\right)}{Ml_1} + \frac{gm_2\left(\frac{g(M+m_2)}{M^2l_2^2}+\frac{gm_1}{M^2l_1l_2}\right)}{Ml_1} \\ 0 \\ \frac{g(M+m_2)\left(\frac{g(M+m_2)}{M^2l_2^2}+\frac{gm_1}{M^2l_1l_2}\right)}{Ml_2} + \frac{gm_1\left(\frac{g(M+m_1)}{M^2l_1^2}+\frac{gm_2}{M^2l_1l_2}\right)}{Ml_2} \\ 0 \end{bmatrix} \tag{36}$$

Combining all of these individual 6x1 matrices forms the C matrix. Calculating its determinant would be difficult by hand, however with MATLAB it is quite simple. The determinant of C is as follows.

$$-\frac{g^6l_1^2 - 2g^6l_1l_2 + g^6l_2^2}{M^6l_1^6l_2^6} \tag{37}$$

Rearranging this equation, we get

$$\frac{-g^6(l_1 - l_2)^2}{M^6l_1^6l_2^6} \tag{38}$$

Based on this result, the determinant of C matrix will not equal 0 as long as

$$l_1 \neq l_2$$

This means that the system is controllable when the lengths of cables l_1 and l_2 are not equal to each other.

5 Create an LQR controller for Given Values

We now select values for the system for evaluation, and use an LQR controller to control the system. The benefit of the LQR controller is that it is guaranteed to stabilize the system in the optimal way, and it provides an easy way to trade off speed of response with control effort. We select the system parameters to be $M = 1000Kg,\ m_1 = m_2 = 100Kg,\ l_1 = 20m,\ l_2 = 10m$. Substituting these values into our A_f and B_f matrices we get the following linearized state space equations.

$$A_f = \begin{bmatrix} 0 & 1.0000 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.9800 & 0 & -0.9800 & 0 \\ 0 & 0 & 0 & 1.0000 & 0 & 0 \\ 0 & 0 & -0.5390 & 0 & -0.0490 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0000 \\ 0 & 0 & -0.0980 & 0 & -1.0780 & 0 \end{bmatrix}, B_f = \begin{bmatrix} 0 \\ 0.0010 \\ 0 \\ 0.00005 \\ 0.0010 \end{bmatrix} \tag{39}$$

After calculating the controllability matrix $C = [B\ AB\ A^2B\ A^3B\ A^4B\ A^5B]$, we determine the rank of this matrix and see that it is 6, which means given the system parameters defined above, the system is controllable.

Since the pair (A, B_K) is controllable, we can look for a K that minimizes the following cost function:

$$J(K, x(0)) = \int_0^\infty x^T(t)Qx(t) + u_k^T(t)RU_K(t)\ dt \tag{40}$$

Where J is the cost, x is the state vector, and u is the input vector. The optimal K value that minimizes J is given by the following equation:

$$K = \frac{1}{R} B_K P \tag{41}$$

Where P is the unique symmetric positive definite $n \times n$ solution to the algebraic Riccati equation:

$$A^T P + PA - PBR^{-1}B^T P = -Q \tag{42}$$

In this the equation Q and R are the cost matrices for our state and input parameters. A higher weighting value for a parameter in Q or R will work to prioritize the minimization of that parameter. Q must be a positive semidefinite matrix, and R must be a positive definite matrix. The simplest way to ensure this is to make the Q and R matrices diagonal.

We evaluated this linearized system in Matlab, which has a built in function `lqr(A, B, Q, R)` which solves the Riccati equation and returns the optimal K vector. The weighting values we chose and the resulting control vector are as follows

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 100 & 0 & 0 \\ 0 & 0 & 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \end{bmatrix}, R = \begin{bmatrix} 0.00001 \end{bmatrix} \tag{43}$$

Which gives us our LQR control constants of.

$$K = 0.001 * \begin{bmatrix} 0.3162 & 1.1136 & 2.3936 & 3.0234 & 3.5128 & 0.7663 \end{bmatrix} \tag{44}$$

Before implementing the feedback control, we wanted to see that our system will oscillate indefinitely without any input, since there is no friction in the system. This is supported by the fact that all of the Eigenvalues of A are on the imaginary axis. This also means that the stability of the system about the equilibrium point by Lyapunov's indirect method is inconclusive. The uncontrolled response of the system can be seen in Figure 2.

$$eig(A) = \begin{bmatrix} 0 \\ 0 \\ 0.7282i \\ -0.7282i \\ 1.0425i \\ -1.0425i \end{bmatrix} \tag{45}$$

With the initial conditions

$$x(0) = \begin{bmatrix} 1 \\ 0 \\ \pi/2 \\ 0 \\ -\pi/10 \\ 0 \end{bmatrix} \tag{46}$$

and no control we see that the system oscillates indefinitely

We then introduce the state feedback control to the system by using the closed loop state feedback equations

$$\dot{X}(t) = (A + B_K K)X(t) + B_D U_D(t) \tag{47}$$

$$Y(t) = (C + DK)X(t) \tag{48}$$

Which produces the response seen in Figure 3

In this diagram we see that the system is controlled and reaches equilibrium after about 50 seconds. We then modeled our nonlinear system in Simulink and applied the same optimal control gain to the

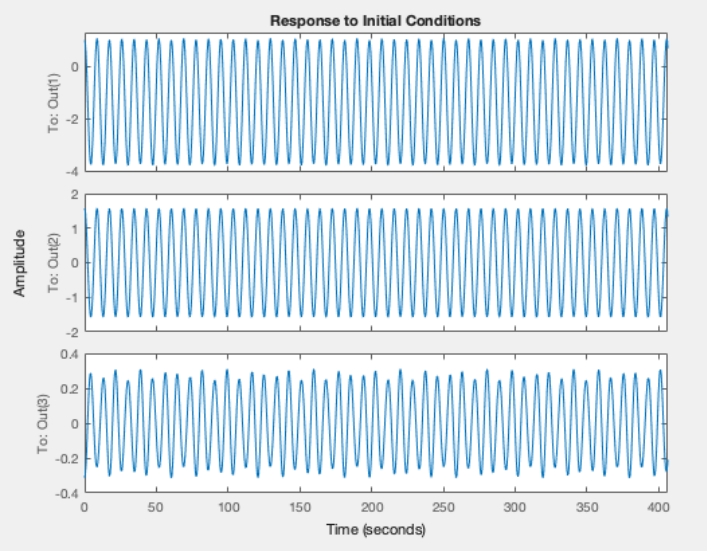

Figure 2: Plot of Uncontrolled State Response.

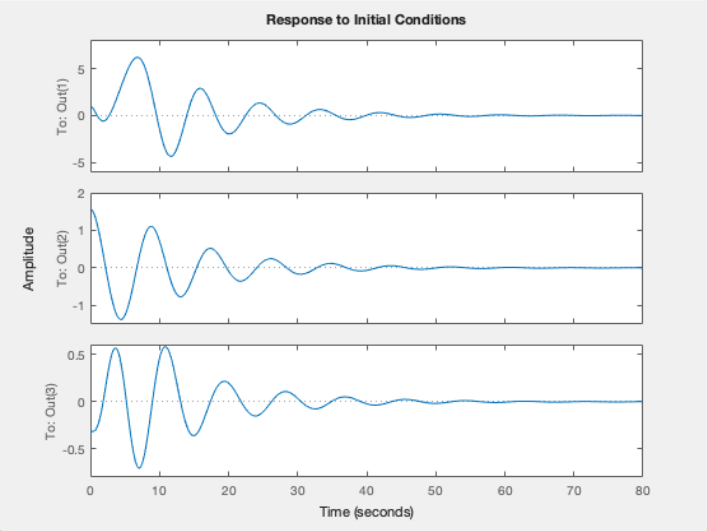

Figure 3: Plot of Controlled Linearized State Response.

system to see the response on a system that has not been linearized. The nonlinear response was modeled in Simulink, the block diagram for which can be seen in Figure 4

We see that the linearized equations closely match the response of the nonlinear system, and the control input acting on the nonlinear system brings the system to equilibrium.

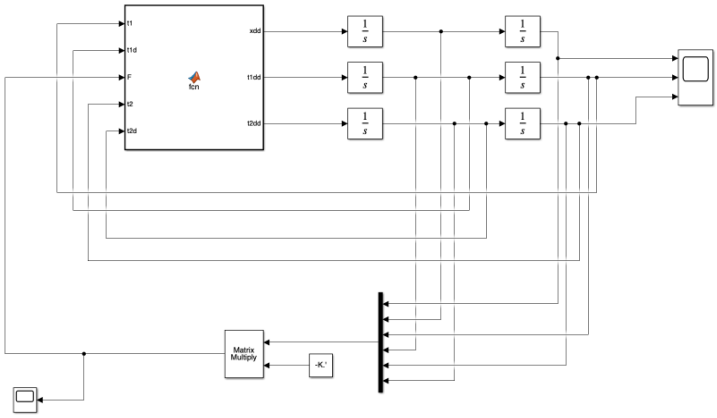

Figure 4: Simulink Model of Nonlinear System.

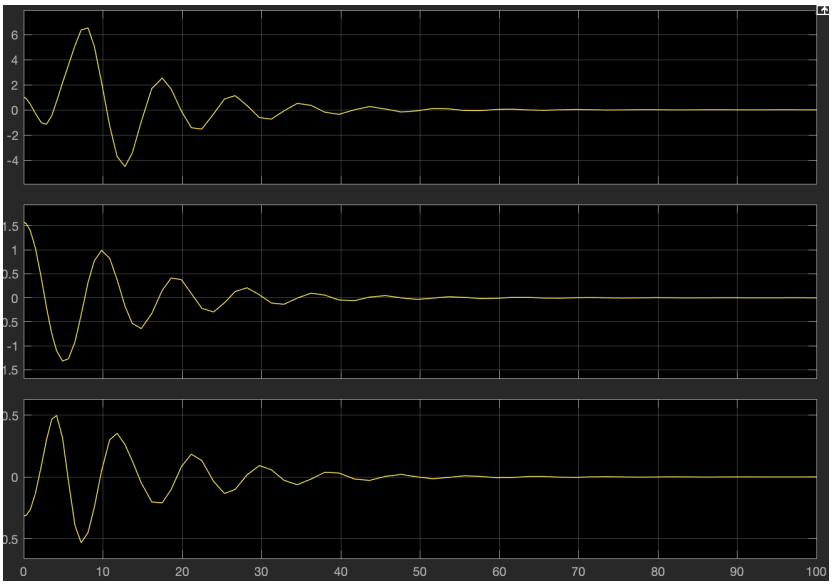

Figure 5: Nonlinear System Control Response.

6 Determine which output vectors of the linearized system are observable

We can use the Observability matrix, O, to determine if the linearized system is observable given the following four potential output vectors

$$C_1 = [x(t)] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{49}$$

$$C_2 = [\theta_1(t), \theta_2(t)] = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{50}$$

$$C_3 = [x(t), \theta_2(t)] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{51}$$

$$C_4 = [x(t), \theta_1(t), \theta_2(t)] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{52}$$

To calculate the observability matrix for each vector, we use the A_f matrix found in 31 and the C_i matrices described above. The equation for O_i is as follows

$$O_i = \begin{bmatrix} C_i \\ C_i A_f \\ C_i A_f^2 \\ \vdots \\ C_i A_f^{n-1} \end{bmatrix} \tag{53}$$

Similarly to the test for Controllability, this is a Linear Time Invariant system, so the Observability can be determined by taking the rank of O_i for each vector output.

$$\text{rank}(O_i) = n \tag{54}$$

Once again there are 6 states, so an Observable system will have a full rank of $n = 6$. The Observability matrices for each vector were computed in MATLAB, and the results are shown below. The full code for this section will be included at the end of the report.

```
>> part_E
rank of output vector [x(t)]:
     6

rank of output vector [theta1(t), theta2(t)]:
     4

rank of output vector [x(t), theta2(t)]:
     6

rank of output vector [x(t), theta1(t), theta2(t)]:
     6
```

Figure 6: Output of the Observability check MATLAB script.

As you can see in 6, the output of the MATLAB script shows that vectors C_1, C_3, and C_4 are Observable with a full rank of $n = 6$ for the linearized system, however vector C_2, is not Observable because it does not have a full rank.

7 Design the Luenberger Observer for the Observable output vectors

The Luenberger Observer, is a state observer concept that involves using the current and past values of plant input and output signals to generate an estimate of a current state. The equation for the Luenberger is as follows.

$$\begin{aligned} &\dot{\hat{X}}(t) = A\hat{X}(t) + B_K U_K(t) + L\left(Y(t) - \hat{C}X(t)\right), \\ &X(0) = 0 \end{aligned} \tag{55}$$

In the previous equation, $\hat{X}(t)$ is considered to be state estimator, L is the observer gain matrix, and $Y(t) - C\hat{X}(t)$ is the correction term. The initial condition $\hat{X}(0)$ is also set equal to 0. As presented in lecture, $X_e(t)$ has the following state space form.

$$\dot{X}_e(t) = \dot{X}(t) - \dot{\hat{X}}(t), \tag{56}$$

$$\dot{X}_e(t) = AX_e(t) - L\big(Y(t) - C\hat{X}(t)\big) + B_d U_d(t). \tag{57}$$

If we assume that the disturbance of the system is 0, then

$$Y(t) = Cx(t). \tag{58}$$

Plugging this into the $\dot{X}_e$ equation yields

$$\dot{X}_e(t) = (A - LC)X_e(t) + B_d U_d(t). \tag{59}$$

This is the Luenberger Observer equation that will be used to create the observer for the crane mass system.

The work for the Luenberger Observer for the crane mass system was completed in MATLAB. All code for these simulations can be found at the bottom of the document. The results for the Luenberger Observer for linearized system is as follows. The next three figures will show the system response as well as the observer estimate for the linearized system with only initial conditions as input. The initial conditions for these three simulations were $x(0) = [1; 0; \pi/2; 0; -\pi/10; 0]$.

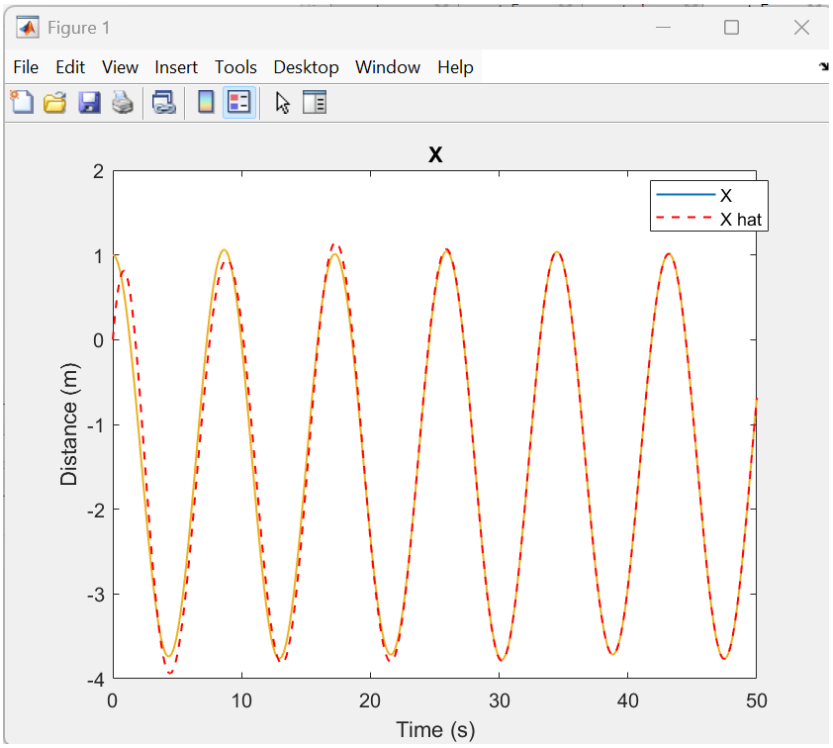

Figure 7: Linearized system response and observation for the [x(t)] output vector with only initial conditions as input.

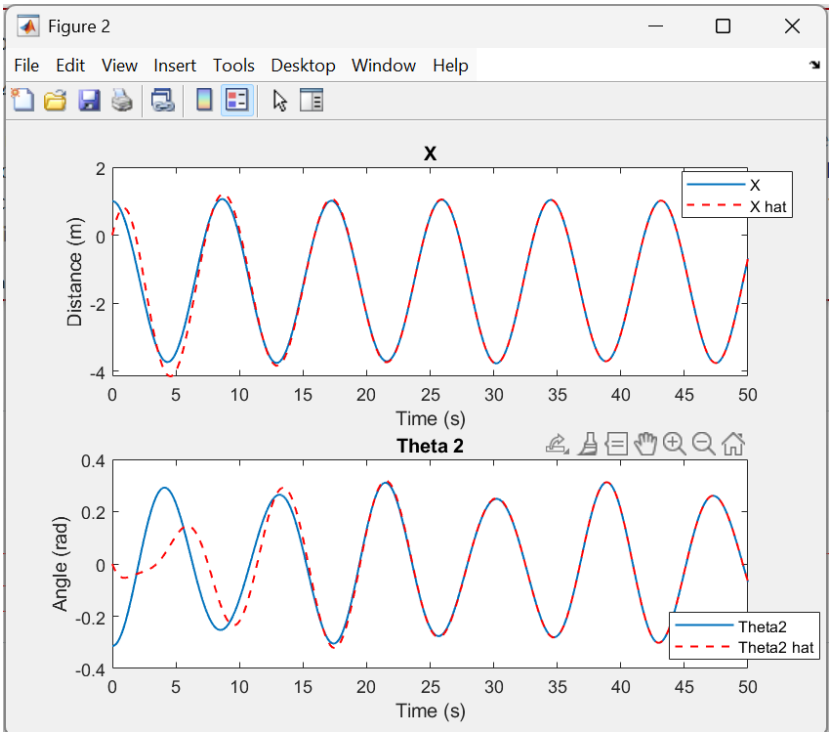

Figure 8: Linearized system response and observation for the [x(t), $\theta_2(t)$] output vector with only initial conditions as input.

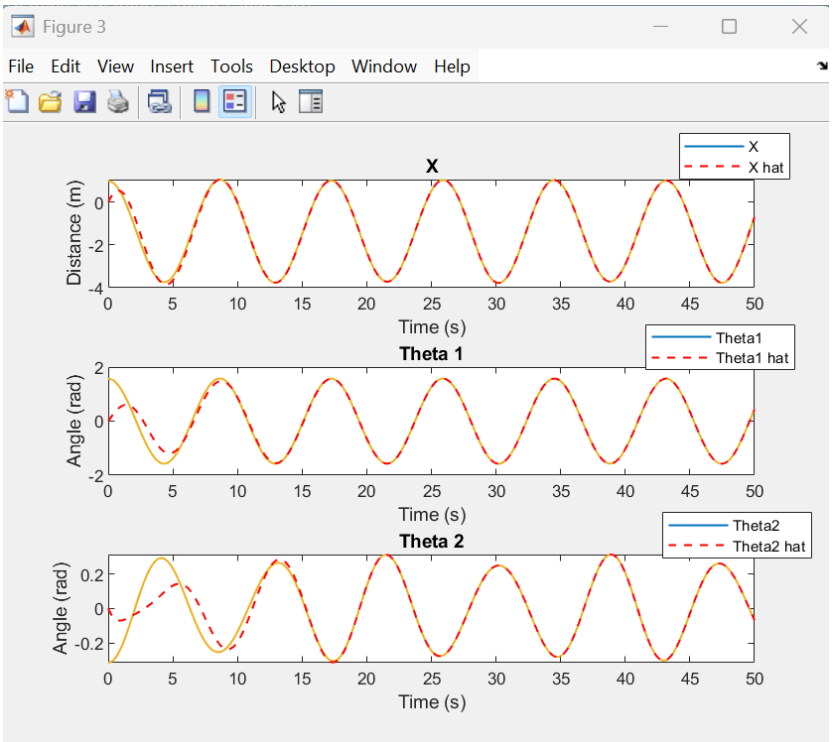

Figure 9: Linearized system response and observation for the [x(t), $\theta_1(t)$, $\theta_2(t)$] output vector with only initial conditions as input.

The next set of figures are also for the linearized system, however this time the initial conditions are set to 0 and the system experiences a step input at $t = 0$.

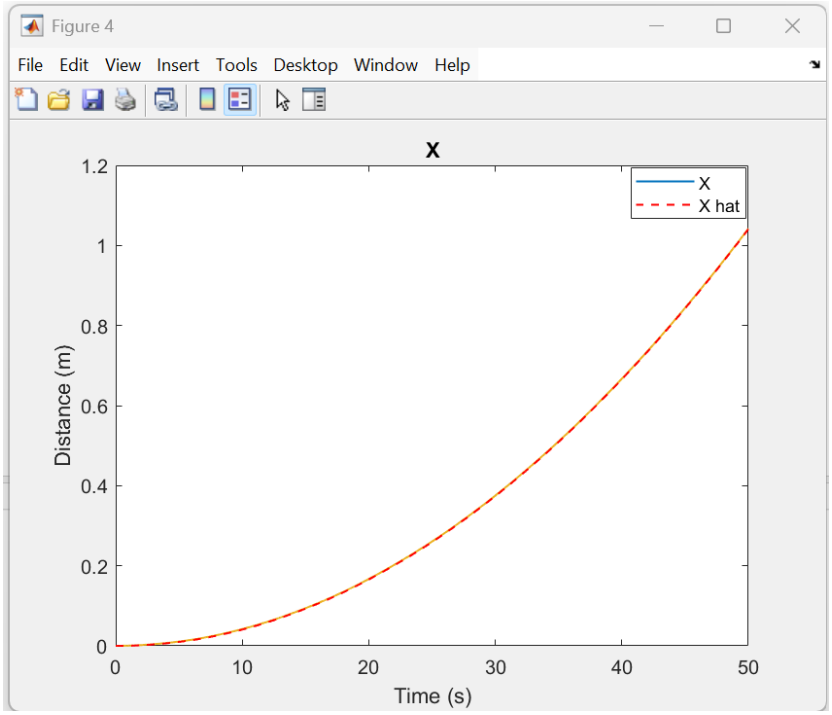

Figure 10: Linearized system response and observation for the [x(t)] output vector with step input at $t = 0$.

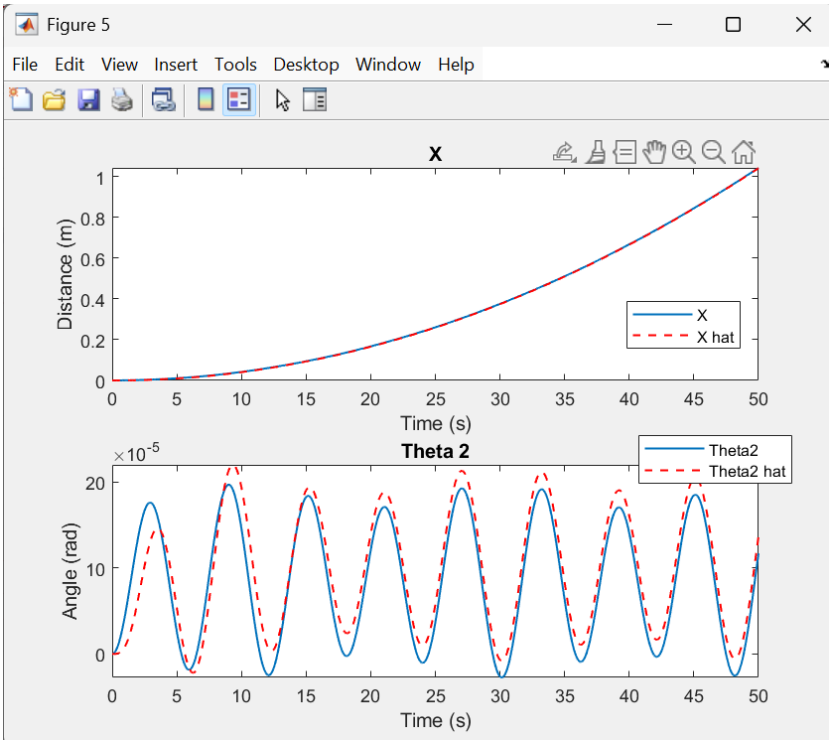

Figure 11: Linearized system response and observation for the [x(t), $\theta_2(t)$] output vector with step input at $t = 0$.

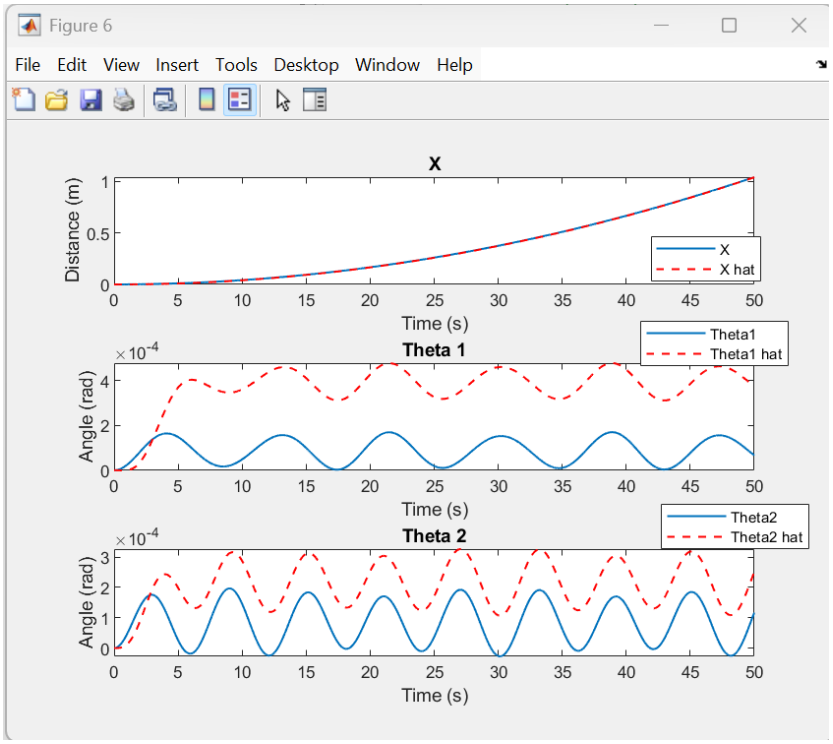

Figure 12: Linearized system response and observation for the [x(t), $\theta_1(t)$, $\theta_2(t)$] output vector with step input at $t = 0$. Note the small magnitude of the observation error for θ_1, and θ_2

.

In addition to creating the Luenberger Observer for the linearized system model, we also applied the observer to the nonlinear plant model. Instead of writing up a MATLAB script like we did for the linearized system, the nonlinear plant developed in the LQR control section was reused. Connected to this plant, was a MATLAB function block for the Luenberger Observer. An image of the Simulink model is shown below.

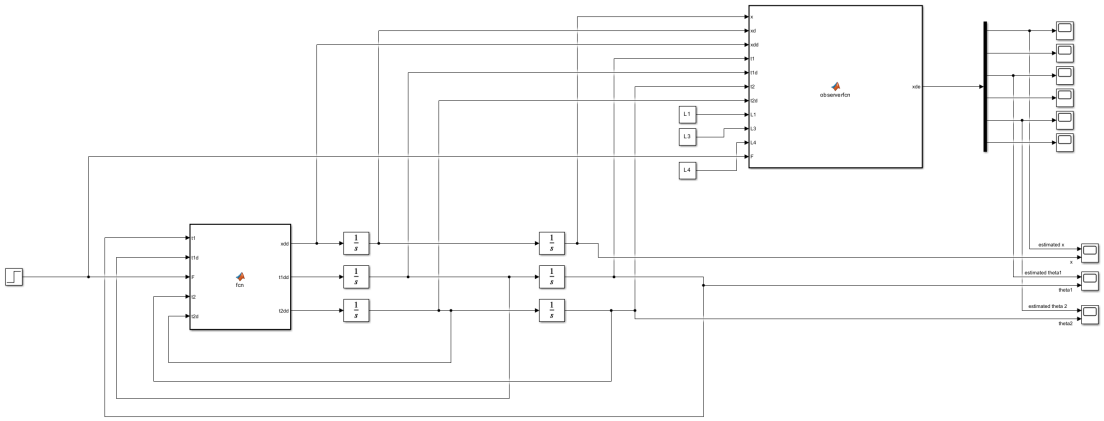

Figure 13: The Nonlinear plant model with the Luenberger Observer. The step function block was commented out during simulations with only initial conditions

The simulations for each output vector were then repeated on the nonlinear plant model. They are as follows.

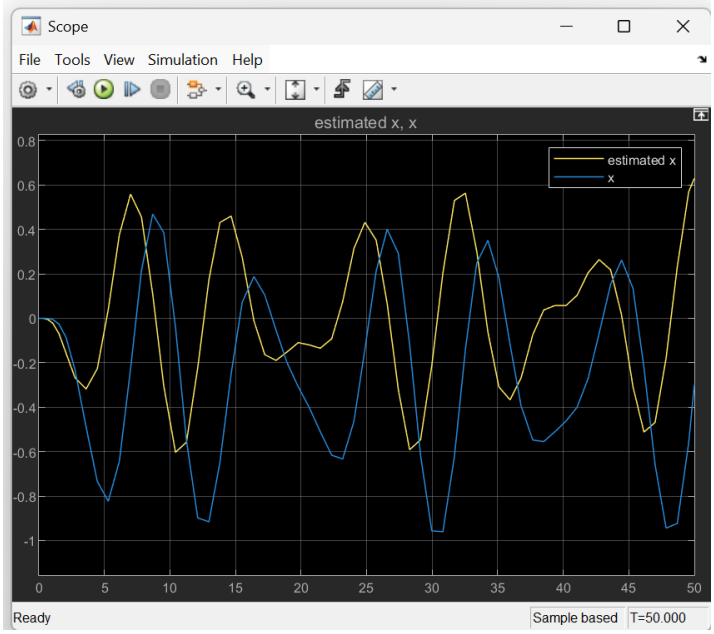

Figure 14: Nonlinear system response and observation for the [x(t)] output vector with with only initial conditions as input.

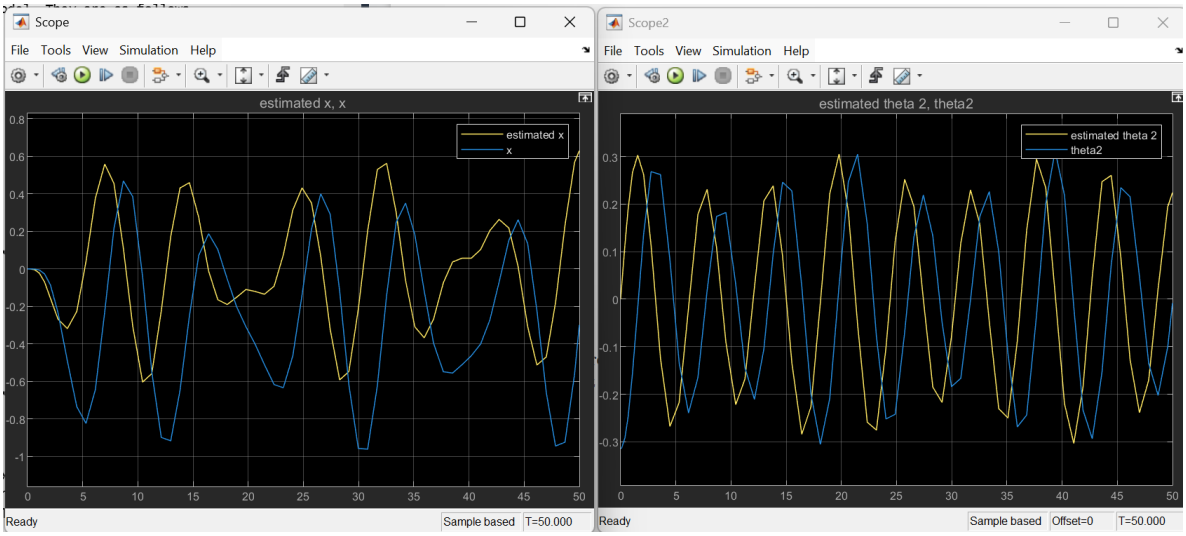

Figure 15: Nonlinear system response and observation for the [x(t), $\theta_2(t)$] output vector with only initial conditions as input.

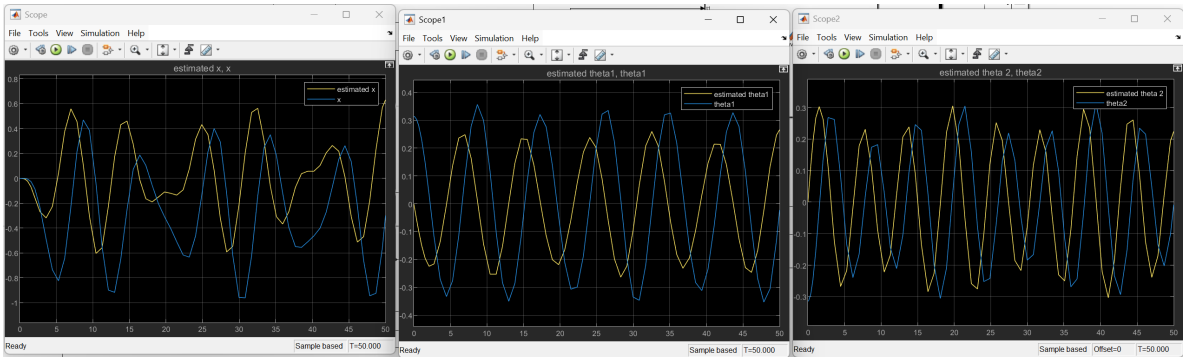

Figure 16: Nonlinear system response and observation for the [x(t), $\theta_1(t)$, $\theta_2(t)$] output vector with only initial conditions as input.

As you can see, the Luenberger Observer is able to track the nonlinear plant fairly well when subject to only initial conditions. There is larger, but constant, estimation error for the nonlinear observer compared to the linearized system. Next, we will show the observer when faced with a step function applied to the nonlinear plant model.

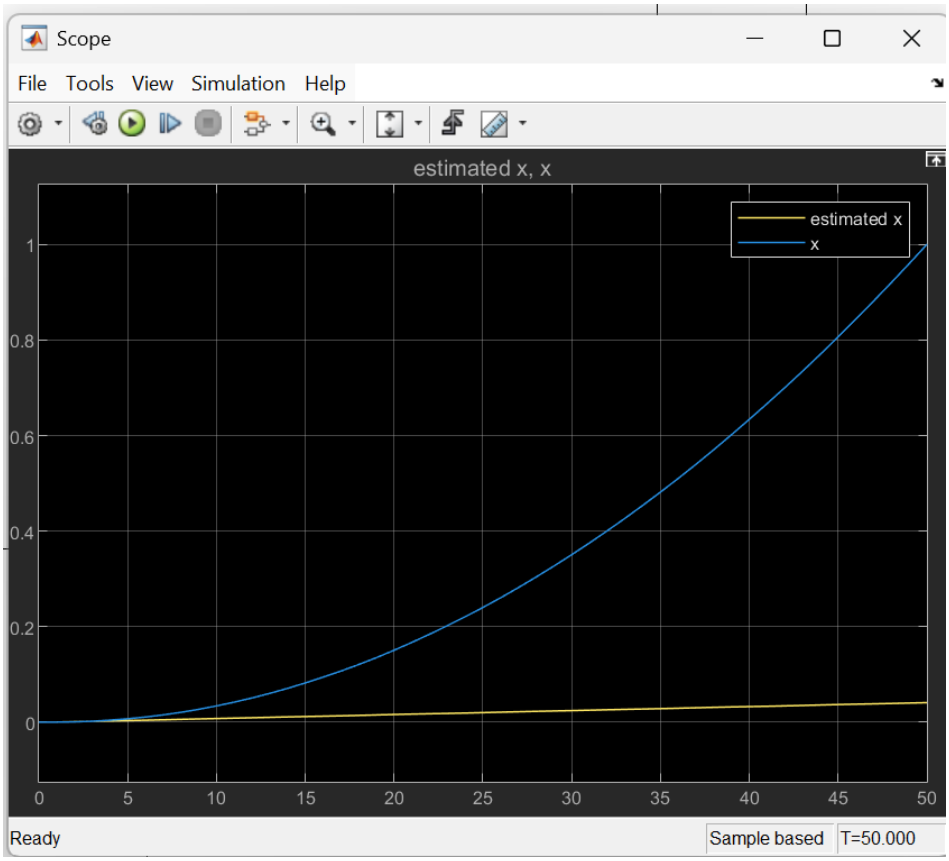

Figure 17: Nonlinear system response and observation for the [x(t)] output vector with step input at $t = 0$.

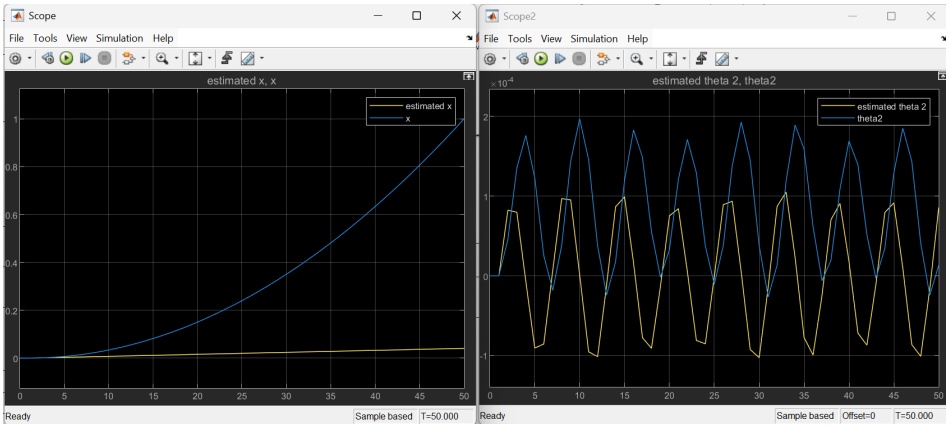

Figure 18: Nonlinear system response and observation for the [x(t), $\theta_2(t)$] output vector with step input at $t = 0$.

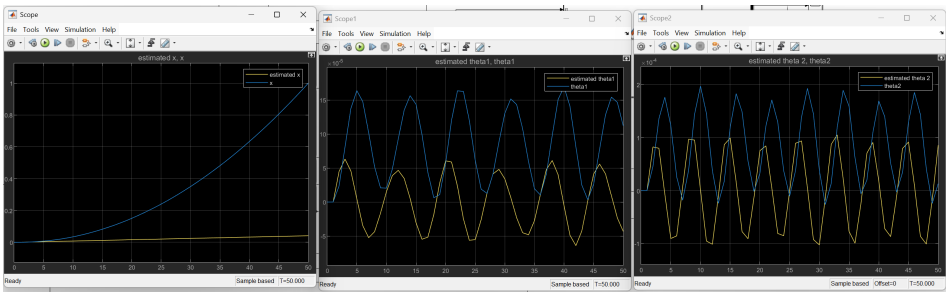

Figure 19: Nonlinear system response and observation for the [x(t), $\theta_1(t)$, $\theta_2(t)$] output vector with step input at $t = 0$.

As the figures show, the observer struggled to estimate the nonlinear system when faced with a step input. The response for the x value is increasing, although it is significantly behind the true x state value. The estimations for the θ_1 and θ_2 were more responsive, but they had a decent estimation error compared to the true state as well. One potential solution to address this could be update the L values for the system. Using a different gain matrix could improve system response time for the x state, and reduce some of the estimation error in the θ_1 and θ_2 states. The most effective solution, as shown in the earlier simulations, is to linearize the nonlinear system. This method proved to deliver great results with precise and consistent state estimations. As you will see in the next section, yet another way to improve that state estimation, is to use a Kalman filter.

8 LQG Controller Design

In the final section of this paper we use the combination of an LQR optimal controller with a Kalman filter observer to form the LQG controller. This controller uses the state estimation from the Kalman filter to generate the optimal response for the system. This design is based on the separation principal of control, which states that observers and controllers can be designed separately and then combined to form a stable system.

We implemented the LQG controller in Simulink, the block diagram for which can be seen below.

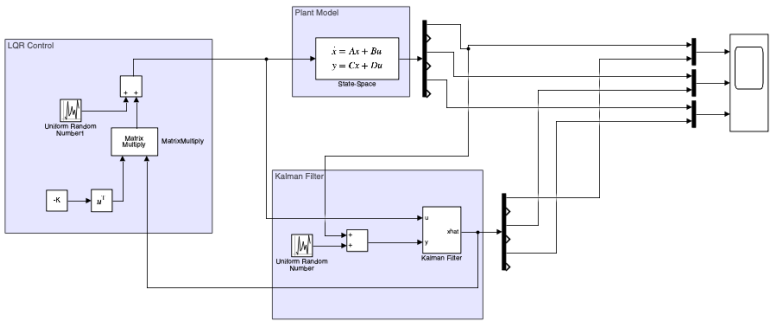

Figure 20: Simulink Model of LQG Controller.

The covariance values we selected are

$$Q = 0.1,\ R = 0.0001 \tag{60}$$

And the starting state parameters are

$$x(0) = \begin{bmatrix} 1 \\ 0 \\ \pi/2 \\ 0 \\ -\pi/10 \\ 0 \end{bmatrix}, \hat{x}(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{61}$$

From these output graphs we can see that the system is stable and settles to the equilibrium point in about 60 seconds. The yellow lines show the true system states while the blue lines show the estimated states from the Kalman filter. In our Simulink design we introduce white Gaussian noise to our control input and to our state output, which model the inaccuracies of a real word system. The result of this added noise can be seen in the variability of the Kalman state estimation. This still produces a stable system.

In our system the output vector ($x(t)$), which only contains 1 state variable is enough to produce an observability matrix which is full rank. So for our LQG design the state observation/estimation is based only on the output $x(t)$.

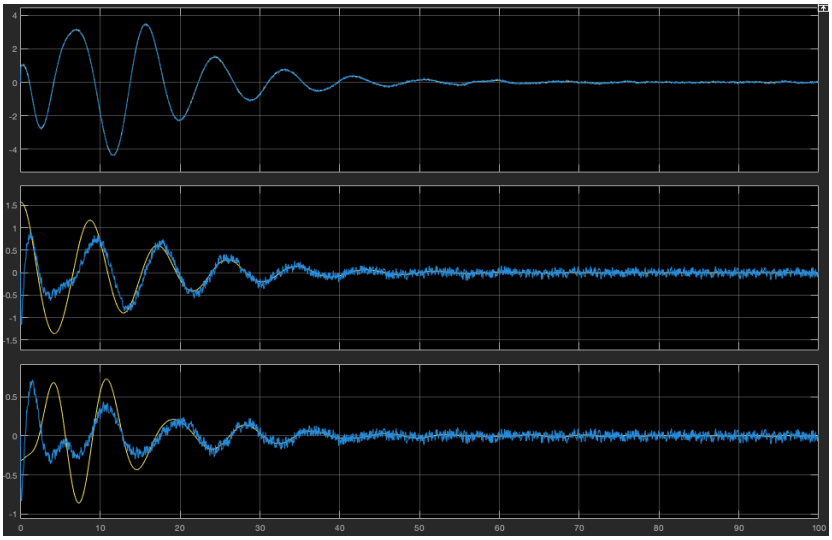

Figure 21: System response to LQG Control.

If we wanted to modify our design to track an input reference, instead of regulate to equilibrium, we would need to make updates to the state space equation and control law. These equations would become

$$\dot{x}(t) = Ax(t) + Bu(t) + B_r r(t) \tag{62}$$

$$u(t) = -Kx(t) + K_r r(t) \tag{63}$$

Our system also rejects constant force disturbances on the system. We tested this by adding a sine function to the state input equation after the LQR control. We see that this does not affect the performance of the system.

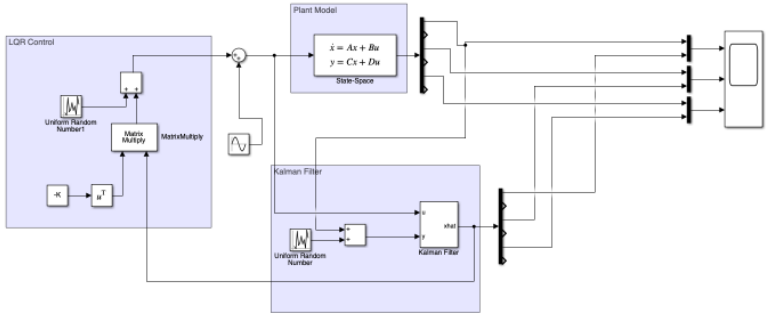

Figure 22: System Design with Constant Disturbance.

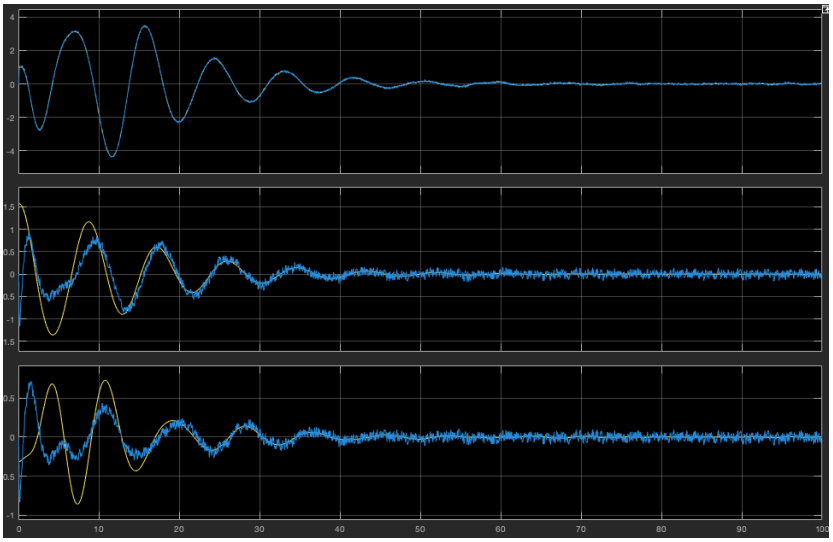

Figure 23: System response to LQG Control with Constant Input Disturbance.

9 Appendix

9.1 Code for Part B

```
    clear

% Define the variables of the system
x = sym('x');
xDot = sym('xdot');
theta1 = sym("theta1");
theta1Dot = sym("theta1Dot");
theta2 = sym("theta2");
theta2Dot = sym("theta2Dot");
F = sym("F");
M = sym("M");
m1 = sym("m1");
m2 = sym("m2");
l1 = sym("l1");
l2 = sym("l2");
g = sym("g");

% Define first equation of motion
eq2 = (F - ...
    m1*l1*sin(theta1)*theta1Dot^2 - ...
    m2*l2*sin(theta2)*theta2Dot^2 - ...
    m1*g*cos(theta1)*sin(theta1) - ...
    m2*g*cos(theta2)*sin(theta2) ) / ...
    (M+m1*sin(theta1)^2+m2*sin(theta2)^2);

% Define second equation of motion
eq4 = (cos(theta1)/l1) * ...
      ((F - m1*l1*sin(theta1)*theta1Dot^2 - m2*l2*sin(theta2)*theta2Dot^2 - ...
      m1*g*cos(theta1)*sin(theta1) - m2*g*cos(theta2)*sin(theta2))  / ...
      (M+m1*(sin(theta1))^2 + m2*(sin(theta2))^2))  - ...
      (g*sin(theta1)/l1);

% Define third equation of motion
eq6 = (cos(theta2)/l2) * ...
      ((F - m1*l1*sin(theta1)*theta1Dot^2 - m2*l2*sin(theta2)*theta2Dot^2 - ...
      m1*g*cos(theta1)*sin(theta1) - m2*g*cos(theta2)*sin(theta2))  / ...
      (M+m1*(sin(theta1))^2 + m2*sin((theta2))^2)) - ...
      (g*sin(theta2)/l2);


% Compute the partial derivative of an equation with respect to a given
% variable.  Do this for all equations and all variables to compute the
% jacobian
deriv = diff(eq6,theta1)

% Define equilibrium point
theta1 = 0;
theta1Dot = 0;
theta2 = 0;
theta2Dot = 0;

% Evaluate partial derivative at the equilibrium
subs(deriv)
```

9.2 Code for Part C

```
    % PART C TEST FOR CONTROLLABILITY

clear

x = sym('x');
xDot = sym('xdot');
theta1 = sym("theta1");
theta1Dot = sym("theta1Dot");
theta2 = sym("theta2");
theta2Dot = sym("theta2Dot");
F = sym("F");
M = sym("M");
m1 = sym("m1");
m2 = sym("m2");
l1 = sym("l1");
l2 = sym("l2");
g = sym("g");


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

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

AB = A*B;

A2B = A * AB;

A3B = A * A2B;

A4B = A * A3B;

A5B = A * A4B;

C = [B, AB, A2B, A3B, A4B, A5B];

rank(C);
simplify(det(C))
```

9.3 Code for Part D

```
    % PART D LQR

clear

x = sym('x');
```

```
xDot = sym('xdot');
theta1 = sym("theta1");
theta1Dot = sym("theta1Dot");
theta2 = sym("theta2");
theta2Dot = sym("theta2Dot");
F = sym("F");

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


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

eig(A)

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

% Define our outputs as x, t1, and t2
C = [1, 0, 0, 0, 0, 0;
     0, 0, 1, 0, 0, 0;
     0, 0, 0, 0, 1, 0];


AB = A*B;

A2B = A * AB;

A3B = A * A2B;

A4B = A * A3B;

A5B = A * A4B;

C_AB = [B, AB, A2B, A3B, A4B, A5B];

rank(C_AB)
det(C_AB)

Q = [1, 0, 0, 0, 0, 0;
     0, 1, 0, 0, 0, 0;
     0, 0, 100, 0, 0, 0;
     0, 0, 0, 100, 0, 0;
```

```
     0, 0, 0, 0, 100, 0;
     0, 0, 0, 0, 0, 100;];

R = [0.00001];

[K, S, P] = lqr(A, B, Q, R);
K

% create the state space representation for the linearized system
% sys = ss(A, B, C, [])    % uncontrolled system
sys = ss(A-B*K, B, C, []); % controlled system


%     x, xd, t1,  t1d,  t2,  t2d
x0 = [1; 0; pi/2; 0; -pi/10; 0;];

% Simulate and plot the initial condition response
initial(sys, x0)
```

9.4 Code for part E

```
    %PART E CHECK FOR OBSERVABILITY

clear

x = sym('x');
xDot = sym('xdot');
theta1 = sym("theta1");
theta1Dot = sym("theta1Dot");
theta2 = sym("theta2");
theta2Dot = sym("theta2Dot");
F = sym("F");
M = sym("M");
m1 = sym("m1");
m2 = sym("m2");
l1 = sym("l1");
l2 = sym("l2");
g = sym("g");


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

% First Output Vector

C1 = [1, 0, 0, 0, 0, 0];

C1A = C1 * A;

C1_2_A = C1A * A;

C1_3_A = C1_2_A * A;
```

```
C1_4_A = C1_3_A * A;

C1_5_A = C1_4_A * A;

O1 = [C1;
      C1A;
      C1_2_A;
      C1_3_A;
      C1_4_A;
      C1_5_A];

disp('rank of output vector [x(t)]:')
disp(rank(O1))

% Second Output Vector

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

C2A = C2 * A;

C2_2_A = C2A * A;

C2_3_A = C2_2_A * A;

C2_4_A = C2_3_A * A;

C2_5_A = C2_4_A * A;

O2 = [C2;
      C2A;
      C2_2_A;
      C2_3_A;
      C2_4_A;
      C2_5_A];

disp('rank of output vector [theta1(t), theta2(t)]:')
disp(rank(O2))

% Third Output Vector

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

C3A = C3 * A;

C3_2_A = C3A * A;

C3_3_A = C3_2_A * A;

C3_4_A = C3_3_A * A;

C3_5_A = C3_4_A * A;

O3 = [C3;
      C3A;
```

```
      C3_2_A;
      C3_3_A;
      C3_4_A;
      C3_5_A];

disp('rank of output vector [x(t), theta2(t)]:')
disp(rank(O3))

% Fourth Output Vector

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

C4A = C4 * A;

C4_2_A = C4A * A;

C4_3_A = C4_2_A * A;

C4_4_A = C4_3_A * A;

C4_5_A = C4_4_A * A;

O4 = [C4;
      C4A;
      C4_2_A;
      C4_3_A;
      C4_4_A;
      C4_5_A];

disp('rank of output vector [x(t), theta1(t), theta2(t)]:')
disp(rank(O4))
```

9.5 Code for part F

```
    % PART F: LUENBERGER OBSERVER

clear

x = sym('x');
xDot = sym('xdot');
theta1 = sym("theta1");
theta1Dot = sym("theta1Dot");
theta2 = sym("theta2");
theta2Dot = sym("theta2Dot");
F = sym("F");

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

q0 = [2 0 deg2rad(17) 0 deg2rad(30) 0];
```

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

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


% Define our outputs as x
C1 = [1, 0, 0, 0, 0, 0];

% Define our outputs as x and t2
C3 = [1, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 1, 0];

% Define our outputs as x, t1, and t2
C4 = [1, 0, 0, 0, 0, 0;
     0, 0, 1, 0, 0, 0;
     0, 0, 0, 0, 1, 0];

% Create the state space representation for the linearized system
sys1 = ss(A, B, C1, []);
sys3 = ss(A, B, C3, [0;0]);
sys4 = ss(A, B, C4, [0;0;0]);

% Setting up simulation time initial conditions
t = 0:0.01:50;

% real state i.c.
x0 = [1; 0; pi/2; 0; -pi/10; 0;];
% estimated state i.c.
xhat0 = [0; 0; 0; 0; 0; 0];

% Using lqe to calculate L for observer gain
% Process noise covariance
Q1 = 0.1 * eye(6);
Q3 = 0.1 * eye(6);
Q4 = 0.1 * eye(6);

 % Measurement noise covariance
R1 = 0.01 * eye(1);
R3 = 0.01 * eye(2);
R4 = 0.01 * eye(3);

% Compute observer gain
[L1, P1, E1] = lqe(A, Q1, C1, Q1, R1);
[L3, P3, E3] = lqe(A, Q3, C3, Q3, R3);
[L4, P4, E4] = lqe(A, Q4, C4, Q4, R4);
```

```
% Display results
disp('Observer Gain Matrix L1:');
disp(L1);

disp('Observer Gain Matrix L3:');
disp(L3);

disp('Observer Gain Matrix L4:');
disp(L4);


disp('Eigenvalues of A - L1C (error dynamics):');
disp(E1);

disp('Eigenvalues of A - L3C (error dynamics):');
disp(E3);

disp('Eigenvalues of A - L4C (error dynamics):');
disp(E4);


% Constructing Observer state-space models for Observable output vectors

% Vector 1 [x(t)]
A1_obs = A - (L1 * C1);
B1_obs = [B, L1];
D1_obs = 0;

obs_sys1 = ss(A1_obs, B1_obs, C1, D1_obs);

% Vector 3 [x(t), theta2(t)]
A3_obs = A - (L3 * C3);
B3_obs = [B, L3];
D3_obs = 0;

obs_sys3 = ss(A3_obs, B3_obs, C3, D3_obs);

% Vector 4 [x(t), theta_1(t), theta2(t)]
A4_obs = A - (L4 * C4);
B4_obs = [B, L4];
D4_obs = 0;

obs_sys4 = ss(A4_obs, B4_obs, C4, D4_obs);


% Creating U for intial condition simualtions (no control input)
u1 = zeros(size(t));
u3 = zeros(size(t));
u4 = zeros(size(t));


% SIMULATION 1: OUTPUT VECTOR 1, WITH ONLY INITIAL CONDITIONS

[y1, t] = initial(sys1, x0, t);
[y1_est, t_est] = lsim(obs_sys1,[u1; y1'], t);
```

```
% Plot multiple outputs
figure(1);

plot(t, y1(:, 1), 'LineWidth', 1);  % First output
hold on;
plot(t_est, y1_est(:, 1), 'r--', 'LineWidth', 1);  % First output
title('X')
xlabel('Time (s)');
ylabel('Distance (m)');
legend('X', 'X hat', 'Location', 'best');



% SIMULATION 2: OUTPUT VECTOR 3, WITH ONLY INITIAL CONDITIONS

[y3, t] = initial(sys3, x0, t);
[y3_est, t_est] = lsim(obs_sys3,[u3; y3'], t);


% Plot multiple outputs
figure(2);

subplot(2, 1, 1)
plot(t, y3(:, 1), 'LineWidth', 1);  % First output
hold on;
plot(t_est, y3_est(:, 1), 'r--', 'LineWidth', 1);  % First output
title('X')
xlabel('Time (s)');
ylabel('Distance (m)');
legend('X', 'X hat', 'Location', 'best');

subplot(2, 1, 2)
plot(t, y3(:, 2), 'LineWidth', 1);  % Second output
hold on;
plot(t_est, y3_est(:, 2), 'r--', 'LineWidth', 1);  % Second output
title('Theta 1')
xlabel('Time (s)');
ylabel('Angle (rad)');
legend('Theta1', 'Theta1 hat', 'Location', 'best');

% SIMULATION 3: OUTPUT VECTOR 1, WITH ONLY INITIAL CONDITIONS

[y4, t] = initial(sys4, x0, t);
[y4_est, t_est] = lsim(obs_sys4,[u4; y4'], t);


% Plot multiple outputs
figure(3);

subplot(3, 1, 1)
plot(t, y4(:, 1), 'LineWidth', 1);  % First output
hold on;
plot(t_est, y4_est(:, 1), 'r--', 'LineWidth', 1);  % First output
title('X')
xlabel('Time (s)');
```

```
ylabel('Distance (m)');
legend('X', 'X hat', 'Location', 'best');

subplot(3, 1, 2)
plot(t, y4(:, 2), 'LineWidth', 1);  % Second output
hold on;
plot(t_est, y4_est(:, 2), 'r--', 'LineWidth', 1);  % Second output
title('Theta 1')
xlabel('Time (s)');
ylabel('Angle (rad)');
legend('Theta1', 'Theta1 hat', 'Location', 'best');

subplot(3, 1, 3)
plot(t, y4(:, 3), 'LineWidth', 1);  % Third Output
hold on;
plot(t_est, y4_est(:, 3), 'r--', 'LineWidth', 1);  % Second output
title('Theta 2')
xlabel('Time (s)');
ylabel('Angle (rad)');
legend('Theta2', 'Theta2 hat', 'Location', 'best');


% SIMULATION 4: OUTPUT VECTOR 1, WITH STEP

[y1, t] = step(sys1, t);
[y1_est, t_est] = lsim(obs_sys1,[u1; y1'], t);


% Plot multiple outputs
figure(4);

plot(t, y1(:, 1), 'LineWidth', 1);  % First output
hold on;
plot(t_est, y1_est(:, 1), 'r--', 'LineWidth', 1);  % First output
title('X')
xlabel('Time (s)');
ylabel('Distance (m)');
legend('X', 'X hat', 'Location', 'best');



% SIMULATION 5: OUTPUT VECTOR 3, WITH STEP

[y3, t] = step(sys3, t);
[y3_est, t_est] = lsim(obs_sys3,[u3; y3'], t);

% Plot multiple outputs
figure(5);

subplot(2, 1, 1)
plot(t, y3(:, 1), 'LineWidth', 1);  % First output
hold on;
plot(t_est, y3_est(:, 1), 'r--', 'LineWidth', 1);  % First output
title('X')
xlabel('Time (s)');
ylabel('Distance (m)');
```

```
legend('X', 'X hat', 'Location', 'best');

subplot(2, 1, 2)
plot(t, y3(:, 2), 'LineWidth', 1);  % Second output
hold on;
plot(t_est, y3_est(:, 2), 'r--', 'LineWidth', 1);  % Second output
title('Theta 1')
xlabel('Time (s)');
ylabel('Angle (rad)');
legend('Theta1', 'Theta1 hat', 'Location', 'best');

% SIMULATION 6: OUTPUT VECTOR 1, WITH ONLY INITIAL CONDITIONS

[y4, t] = step(sys4, t);
[y4_est, t_est] = lsim(obs_sys4,[u4; y4'], t);


% Plot multiple outputs
figure(6);

subplot(3, 1, 1)
plot(t, y4(:, 1), 'LineWidth', 1);  % First output
hold on;
plot(t_est, y4_est(:, 1), 'r--', 'LineWidth', 1);  % First output
title('X')
xlabel('Time (s)');
ylabel('Distance (m)');
legend('X', 'X hat', 'Location', 'best');

subplot(3, 1, 2)
plot(t, y4(:, 2), 'LineWidth', 1);  % Second output
hold on;
plot(t_est, y4_est(:, 2), 'r--', 'LineWidth', 1);  % Second output
title('Theta 1')
xlabel('Time (s)');
ylabel('Angle (rad)');
legend('Theta1', 'Theta1 hat', 'Location', 'best');

subplot(3, 1, 3)
plot(t, y4(:, 3), 'LineWidth', 1);  % Third Output
hold on;
plot(t_est, y4_est(:, 3), 'r--', 'LineWidth', 1);  % Second output
title('Theta 2')
xlabel('Time (s)');
ylabel('Angle (rad)');
legend('Theta2', 'Theta2 hat', 'Location', 'best');
```

9.6 Code for part G

```
    % PART D LQR

clear

g = 9.8;
M = 1000;
m1 = 100;
m2 = 100;
```

```
l1 = 20;
l2 = 10;


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

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

% Get all of our states from the system
% only x will be used for kalman filtering/control
% other states used for graphing
C_all = [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;];
% create a C matrix for the Kalman filter state model
C_obs = [1, 0, 0, 0, 0, 0;];

D_all = [0; 0; 0; 0; 0; 0];
D_obs = [0];


Q = [1, 0, 0, 0, 0, 0;
     0, 1, 0, 0, 0, 0;
     0, 0, 100, 0, 0, 0;
     0, 0, 0, 100, 0, 0;
     0, 0, 0, 0, 100, 0;
     0, 0, 0, 0, 0, 100;];

R = [0.00001];

[K, S, P] = lqr(A, B, Q, R);


% Define our covariance matrices for process noise (Q) and measurement
% noise (R). (N) is correlation between the two, which we set to 0. For
% SISO system we have scalers instead of matrices for (Q) and (R)
Q = 0.1;
R = 0.0001;
N = 0;


% Initial conditions
x0 = [1; 0; pi/2; 0; -pi/10; 0;];      % Initial system state
```

```
x_hat0 = [0; 0; 0; 0; 0; 0;];  % Initial state estimate
```

9.7 Code from Matlab Function Blocks

```
*****Nonlinear Function Block*****

    function [xdd, t1dd, t2dd] = fcn( t1, t1d, F, t2, t2d, m1, m2, l1, l2, M, g)

xdd = (F - m1*l1*sin(t1)*t1d^2 - m2*l2*sin(t2)*t2d^2 - m1*g*cos(t1)*sin(t1) - m2*g*cos(t2)*sin(t2)) .
       / ...
      (M + m1*sin(t1)^2 + m2*sin(t2)^2);
t1dd = (1/l1) * (cos(t1)*xdd - g*sin(t1));
t2dd = (1/l2) * (cos(t2)*xdd - g*sin(t2));


*****Luenberger Observer Function Block*****

function xde  = observerfcn( x, xd, xdd, t1, t1d, t2, t2d, L1, L3, L4, F)

g = 9.81;
m1 = 100;
m2 = 100;
M = 1000;
l1 = 20;
l2 = 10;
xde = zeros(6,1);

current_y = [x; xd; t1; t1d; t2; t2d];

% Uncomment lines of code based on which output vector you are observing

%y1 = [x];
%c1 = [1, 0, 0, 0, 0, 0];
%est = L1 * (y1 - (c1*current_y));

%y3 = [x; t2];
%c3 = [1, 0, 0, 0, 0, 0; 0, 0, 0, 0, 1, 0];
%est = L3 * (y3 - (c3*current_y));

y4 = [x; t1; t2];
c4 = [1, 0, 0, 0, 0, 0; 0, 0, 1, 0, 0, 0; 0, 0, 0, 0, 1, 0];
est = L4 * (y4 - (c4*current_y));

xde(1) = xd + est(1);
xde(2) = ((F - m1*l1*sin(t1)*t1d^2 - m2*l2*sin(t2)*t2d^2 - m1*g*cos(t1)*sin(t1) - m2*g*cos(t2)*sin(t2
       / ...
      (M + m1*sin(t1)^2 + m2*sin(t2)^2)) + est(2);
xde(3) = t1d + est(3);
xde(4) = (1/l1) * (cos(t1)*xdd - g*sin(t1)) + est(4);
xde(5) = t2d + est(5);
xde(6) = (1/l2) * (cos(t2)*xdd - g*sin(t2)) + est(6);
```