EE 5235 Final Project
CartBalancer: Robust Inverted Pendulum Stabilization

SAMEER SWAIN
SAHITH BABIREDDY

Final Project Report

May 8, 2023

Contents

List of Figures

1 INTRODUCTION

The inverted pendulum is a canonical example of a nonlinear system, often used to represent problems dealing with balance, such as bipedal walking [1][2][3], unicycles [4], and the Segway Personal Transporters [5]. The inherent instability of the system and its under-actuated mechanical nature, where the control inputs available are less than the degrees of freedom of the system, make it a prime candidate as a baseline study when investigating new control algorithms. However, models are never perfect, and the addition of model uncertainty, sensor noise, and other unknown exogenous disturbances can easily destabilize a seemingly stable system.

In this research project, we aim to study the inverted pendulum on a cart within the context of model uncertainty, sensor noise, and external disturbances. Our goal is to develop a robust control system, the CartBalancer, that can maintain stability under various conditions and disturbances. We will provide a comparison between two popular control methods: H∞ and Linear Quadratic Regulator (LQR) design methods as they relate to the overall system stability and performance.

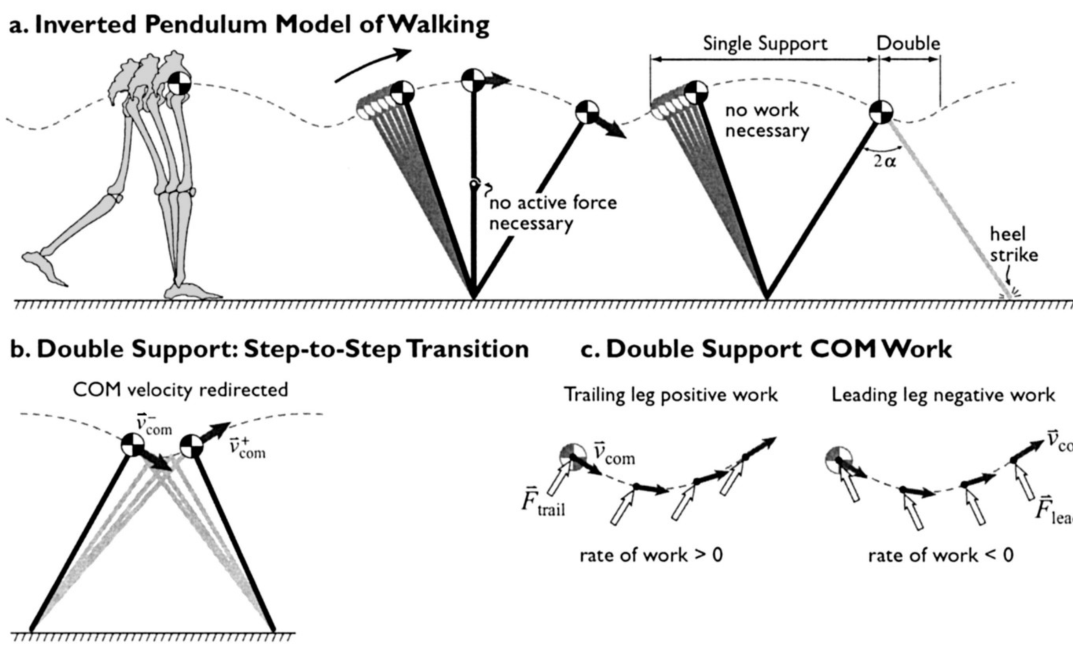

Figure 1: Schematic diagram of the simple inverted pendulum model of walking

However, no model is perfect, and systems like the inverted pendulum can be affected by model uncertainty, sensor noise, and unknown exogenous disturbances. These factors can lead to instability in a system previously considered stable [6][7]. In this context, it is crucial to develop robust control systems capable of maintaining stability under various conditions and disturbances.

The remainder of this paper is organized as follows: we provide the physical and mathematical description of the inverted pendulum model, including the error introduced due to the linearization of the nonlinear system. Discussing the idea of using two separate control algorithms and system representations for swinging up the pendulum from rest to an upright position and maintaining it in that state. Presenting the technical details of the H∞ and LQR methods used for stabilization, including performance requirements and design methodology. Analyzing the closed-loop system results using μ synthesis, comparing the success of the two methods in terms of nominal performance, robust stability, and robust performance. Finally, a conclusion summarises the key findings and potential future work.

Throughout this paper, we refer to various figures and equations to illustrate our findings. For the sake of brevity, these references are not included in the introduction but can be found in the appropriate sections.

2 PROBLEM FORMULATION AND SYSTEM DESCRIPTION

2.1 Physical System

The inverted pendulum system consists of a pendulum attached to a cart that moves along a horizontal track. The pendulum has a mass (m) located at its end, and its length is denoted by (l). The cart has a mass (M) and is subjected to a force (F) that controls its horizontal motion. The angle of the pendulum with respect to the vertical axis is represented by (θ). The goal of the control system is to apply an appropriate force (F) to the cart, such that the pendulum remains balanced in the upright position ($\theta = 0$) despite model uncertainties, sensor noise, and external disturbances. A schematic representation of the inverted pendulum system can be found in Figure 2. For this example, let's assume the following quantities:

- (M) mass of the cart 0.5 kg
- (m) mass of the pendulum 0.2 kg
- (b) coefficient of friction for cart 0.1 N/m/sec
- (l) length to pendulum center of mass 0.3 m
- (I) mass moment of inertia of the pendulum 0.006 kg·m^2
- (F) force applied to the cart
- (x) cart position coordinate
- (θ) pendulum angle from vertical (down)

2.2 Mathematical Model

To develop a control algorithm for the inverted pendulum, we first need to derive the equations of motion for the system. Using the Lagrangian method, the equations of motion can be derived as follows [8]:

$$-(M+m)\ddot{x} + ml\ddot{\theta}\cos(\theta) - ml\dot{\theta}^2\sin(\theta) = F \tag{1}$$

$$2l\ddot{\theta} + \ddot{x}\cos(\theta) - g\sin(\theta) = 0 \tag{2}$$

where $\ddot{x}$ and $\ddot{\theta}$ represent the second time derivatives of the cart position (x) and pendulum angle (θ), respectively; $\dot{\theta}$ is the first time derivative of θ; and g is the acceleration due to gravity.

To design a linear control algorithm, it is necessary to linearize the nonlinear equations of motion (1) and (2) around the equilibrium point ($x = 0, \theta = 0$). Applying the small-angle approximation ($\sin(\theta) \approx \theta$,

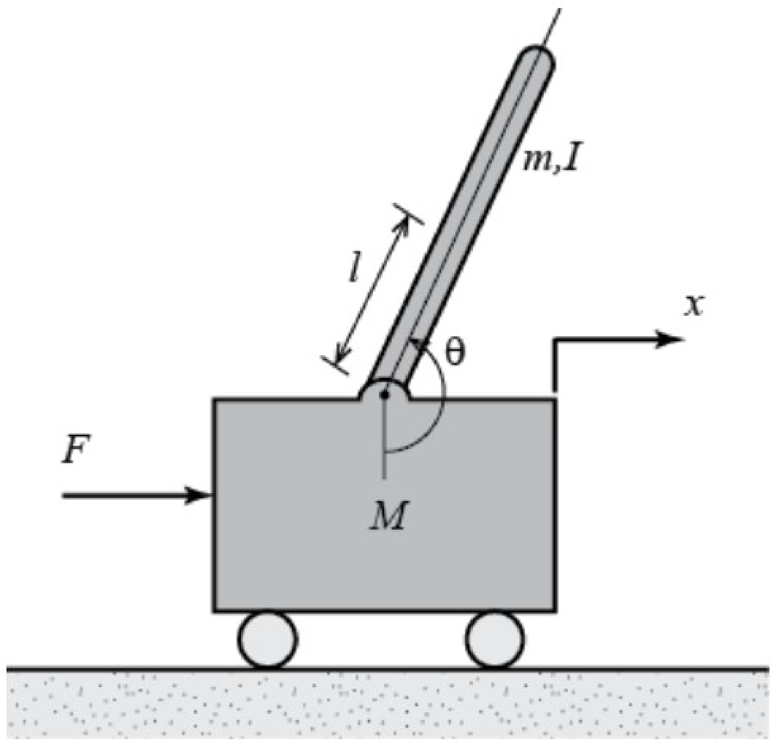

Figure 2: Inverted Pendulum Free Body Diagram

$\cos(\theta) \approx 1$) and assuming that the system is at the equilibrium point, we obtain the following linearized state-space representation:

$$\ddot{x} = \frac{mgl}{M}\theta + \frac{1}{M}F \tag{3}$$

$$\ddot{\theta} = -g\theta - \frac{ml}{M}F \tag{4}$$

Let us define the state vector as $x = [x, \dot{x}, \theta, \dot{\theta}]$ and the control input as $u = F$. We can rewrite the linearized system in matrix form as follows:

$$\dot{x} = Ax + Bu \tag{5}$$

where

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{mgl}{M} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -g & 0 \end{bmatrix}$$

and

$$B = \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ -\frac{ml}{M} \end{bmatrix}.$$

Now, the problem is to design a control law (u) that stabilizes the inverted pendulum by keeping the pendulum angle (θ) and its rate ($\dot{\theta}$) close to zero while ensuring that the cart position (x) and velocity ($\dot{x}$) are also regulated.

In the following sections, we will discuss the design and implementation of H^∞ and LQR control methods for stabilizing the CartBalancer system.

2.3 System Uncertainties and Disturbances

In real-world scenarios, various factors can introduce uncertainties into the system model and affect its performance. These factors include, but are not limited to:

- Uncertainty in the parameters of the system, such as the mass of the cart (M), the mass of the pendulum (m), and the length of the pendulum (l).
- Sensor noise, which can cause inaccuracies in the measurement of the pendulum angle (θ), cart position (x), and their respective rates ($\dot{\theta}$, $\dot{x}$).
- External disturbances, such as friction, wind, or other environmental factors can influence the motion of the cart and pendulum.

In order to design a robust control system that can maintain the stability of the inverted pendulum despite these uncertainties and disturbances, it is crucial to account for them in the control design process.

2.4 Problem Formulation

The control problem can be formulated as follows:

Given the linearized state-space representation of the inverted pendulum system (5), design a control law $u = Kx$, where K is the control gain matrix, such that:

- The closed-loop system is stable, i.e., the eigenvalues of the matrix $(A - BK)$ have negative real parts.
- The control law effectively regulates the states $x = [x, \dot{x}, \theta, \dot{\theta}]$ to zero, despite the presence of model uncertainties, sensor noise, and external disturbances.
- The control effort (u) is minimized to reduce energy consumption and wear on the actuators.

To achieve these objectives, we will employ two main control methods: H^∞ control and Linear Quadratic Regulator (LQR). The H^∞ control method aims to minimize the worst-case effects of disturbances and uncertainties on the system performance, while the LQR method seeks to minimize a quadratic cost function that balances state regulation and control effort.

We discussed the linearization of the nonlinear equations of motion for the inverted pendulum on a cart system and obtain the state-space representation. To provide more detail, we will walk through the linearization process and derive the linearized state-space equations.

Recall the nonlinear equations of motion from Section 2.2:

$$M\ddot{x} + m\ddot{x} - m\dot{\theta}^2 L \sin\theta + mL\ddot{\theta}\cos\theta = F \tag{6}$$

$$(mL^2 + I)\ddot{\theta} + mgL\sin\theta - mL\ddot{x}\cos\theta = 0 \tag{7}$$

We linearize these equations around the equilibrium point where $\theta = 0$ and $\dot{\theta} = 0$. Using the small-angle approximation, we can assume $\sin\theta \approx \theta$ and $\cos\theta \approx 1$. Substituting these approximations into the equations, we get:

$$(M + m)\ddot{x} - m\dot{\theta}^2 L + mL\ddot{\theta} = F \tag{8}$$

$$(mL^2 + I)\ddot{\theta} + mgL\theta - mL\ddot{x} = 0 \tag{9}$$

Now we can rewrite these equations in state-space form. Let the state vector $\mathbf{x} = [x, \dot{x}, \theta, \dot{\theta}]^T$. Then, the state-space representation can be written as:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u \tag{10}$$

where

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{m^2 g L^2}{I(M+m)+MmL^2} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{mgL(M+m)}{I(M+m)+MmL^2} & 0 \end{bmatrix} \tag{11}$$

and

$$\mathbf{B} = \begin{bmatrix} 0 \\ \frac{I+mL^2}{I(M+m)+MmL^2} \\ 0 \\ \frac{mL}{I(M+m)+MmL^2} \end{bmatrix} \tag{12}$$

The output equation is given by:

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}u \tag{13}$$

where

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{14}$$

and

$$\mathbf{D} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{15}$$

This linearized model provides a foundation for designing and analyzing various control strategies in the subsequent sections. The state-space representation enables the application of modern control techniques to stabilize the system and achieve desired performance objectives.

3 Open loop stability

The inverted pendulum when vertically upright with no disturbance is at an unstable equilibrium as explained above. This can also be understood if we look at the potential energy of the system when the pendulum is vertically upward. Even a small disturbance to the cart or pendulum will result in the pendulum falling down and reaching vertically downward which is the stable equilibrium position. This can also be verified by computing the open loop system poles. For the system we have chosen, the poles are as follows:

- 0
- -0.1428
- -5.6041
- 5.5651

We see that we have one positive real part for a pole which makes the system unstable without any control. The same can be observed from the individual transfer functions in the above sections. This gives rise to the control problem which are discussed in the further sections.

4 Two Control Methods: H_∞ and LQR

In this section, we discuss the use of two control methods, H_∞ and Linear Quadratic Regulator (LQR), for designing the stabilization controller. These methods will be employed after the pendulum has been swung up to the upright position by the swing-up controller.

4.1 H_∞ Control

The H_∞ control method aims to minimize the worst-case effects of disturbances and uncertainties on the system performance. Using the mixed sensitivity design approach, we design a linear controller C_∞ that meets the following requirements:

The bandwidth is approximately 250 Hz. The upper bound on $|S(j\omega)|$ should be 1.5, ensuring robustness. $|S(j\omega)|$ should have a slope of approximately 20, dB per decade below the bandwidth. The DC gain of S should be lower than $-80,\mathrm{dB}$. This is a measure of the steady-state tracking error for a step input. $|T(j\omega)|$ should be less than $-3,\mathrm{dB}$ at 500 Hz. This is to reduce the effect of noise. The upper bound on $|T(j\omega)|$ should be 1.5, ensuring robustness. $|T(j\omega)|$ should be less than $-40,\mathrm{dB}$ as $\omega \to \infty$. $|C_\infty S(j\omega)| \leq 5$ for all ω to limit the control effort.

We can use the `mixsyn` command in MATLAB to compute the H_∞ controller C_∞, given the appropriate weighting functions. To implement the H_∞ control method, we must first define appropriate weighting functions that reflect the desired performance specifications. These weighting functions are used to shape the sensitivity function $S(j\omega)$ and complementary sensitivity function $T(j\omega)$ of the system. The weighting functions can be chosen based on the performance specifications mentioned earlier, such as robustness, tracking performance, and control effort.

Once the weighting functions have been defined, we can use the `mixsyn` command in MATLAB to compute the H_∞ controller C_∞. The resulting controller is a linear, state-feedback control law that can be applied to the linearized system to achieve the desired performance and robustness specifications.

The performance and robustness of the H_∞ controller can be analyzed by examining the frequency response of the closed-loop system. In particular, the stability margins, tracking performance, and control effort can be evaluated by analyzing the sensitivity function $S(j\omega)$, complementary sensitivity function $T(j\omega)$, and control-to-output transfer function $C_\infty S(j\omega)$, respectively.

To implement the H_∞ control method, we first define appropriate weighting functions, $W_1(s)$, $W_2(s)$, and $W_3(s)$, that reflect the desired performance specifications. These weighting functions are used to shape the sensitivity function $S(j\omega)$ and complementary sensitivity function $T(j\omega)$ of the system:

$$W_1(s)S(s) = P_{11}(s)C_\infty(s) \tag{16}$$

$$W_2(s)T(s) = P_{22}(s)C_\infty(s) \tag{17}$$

$$W_2(s)T(s) = P_{22}(s)C_\infty(s) \tag{18}$$

$$W_3(s)U(s) = P_{33}(s)C_\infty(s) \tag{19}$$

$$W_3(s)U(s) = P_{33}(s)C_\infty(s) \tag{20}$$

Once the weighting functions have been defined, we can use the `mixsyn` command in MATLAB to compute the H_∞ controller C_∞ by solving the following optimization problem:

$$C_\infty = \min_{\omega} \max \left\{ \left| W_1(j\omega)S(j\omega) \right|, \left| W_2(j\omega)T(j\omega) \right|, \left| W_3(j\omega)U(j\omega) \right| \right\} \tag{21}$$

4.2 LQR Control

The Linear Quadratic Regulator (LQR) is a classical control method that seeks to minimize a quadratic cost function by balancing state regulation and control effort. Given the linearized state-space representation of the system, the LQR control problem can be formulated as follows:

Design a control law $u = -Kx$, such that the closed-loop system is stable and the cost function

$$J = \int_0^\infty (x^T Q x + u^T R u)\, dt \tag{22}$$

is minimized, where Q and R are positive definite weighting matrices that determine the relative importance of state regulation and control effort.

The LQR control gain matrix K can be computed using the Riccati equation or MATLAB's `lqr` command.

The LQR control method requires the selection of appropriate weighting matrices Q and R that determine the relative importance of state regulation and control effort. These matrices must be positive definite, and their choice significantly impacts the overall performance of the LQR controller. By adjusting the elements of Q and R, we can tune the controller to achieve desired performance characteristics such as fast settling time, minimal overshoot, and reduced control effort.

After selecting the weighting matrices Q and R, we can compute the LQR control gain matrix K using the Riccati equation or MATLAB's `lqr` command. The resulting controller is a linear, state-feedback control law that can be applied to the linearized system to achieve the desired performance specifications.

The performance of the LQR controller can be evaluated by simulating the closed-loop system and analyzing the time-domain responses of the states and control input. By comparing the performance of the LQR controller with that of the H_∞ controller, we can assess the trade-offs between these two control methods in terms of stability, robustness, tracking performance, and control effort.

The LQR control method requires the selection of appropriate weighting matrices Q and R that determine the relative importance of state regulation and control effort. The LQR control gain matrix K can be computed by solving the following algebraic Riccati equation:

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{23}$$

where A and B are the system matrices of the linearized state-space representation, and P is the positive definite solution of the Riccati equation. The LQR control gain matrix K is then given by:

$$K = R^{-1}B^T P \tag{24}$$

The resulting controller is a linear, state-feedback control law of the form:

$$u = -Kx \tag{25}$$

By adjusting the elements of Q and R, we can tune the controller to achieve desired performance characteristics such as fast settling time, minimal overshoot, and reduced control effort.

In conclusion, the combination of the swing-up controller and the stabilization controllers designed using H_∞ and LQR methods provides a comprehensive control strategy for the CartBalancer system. By implementing and comparing these two control methods, we can gain insights into the trade-offs between different control approaches and select the most suitable method for achieving the desired performance and robustness specifications in the presence of model uncertainties, sensor noise, and external disturbances.

5 Controller Design and Results

In this section we will see the controller design for the two methods specified above.

5.1 H_∞ Control

In the previous section, we have selected the appropriate weighting functions to design a robust controller using the H_∞ method. The bode plots of the weighting functions are as shown in Figure 3. The choice of the weighting functions was based on the parameters as specified in the previous section. This gives rise to the following weighting functions to use in *mixsyn* command in MATLAB.

$$W1(s) = \frac{1.5s + 0.2896}{s + 2896} \tag{26}$$

$$W2(s) = 10 \tag{27}$$

$$W3(s) = \frac{0.01s + 2337}{s + 1558} \tag{28}$$

The closed loop feedback system was generated by using the function and weights as specified above and the closed loop response of the system to a step input is as shown below in Figure 4.

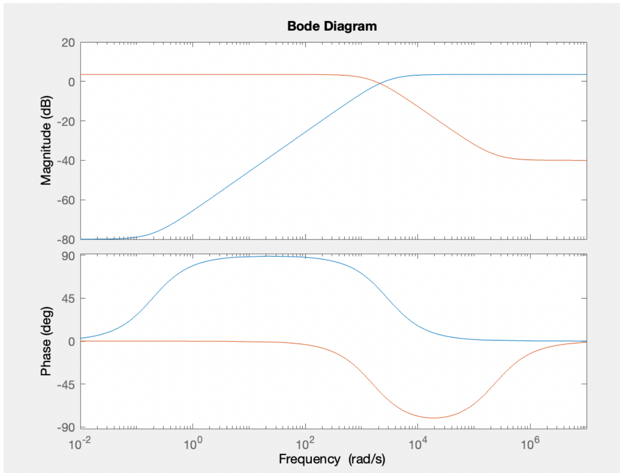

Figure 3: Bode plot of $W1$(Blue line) and $W3$ (Red line)

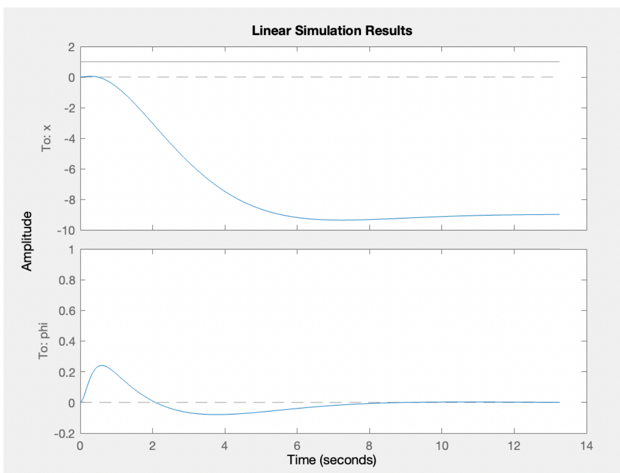

Figure 4: Step response of the closed loop system with H_∞ controller

The H_∞ controller has been robustly designed to handle noise and uncertainty in the system. The possible uncertainties and noise have been discussed in the previous section. The controller effort is as shown in the Figure 5. The settling time and controller effort can be modified by adjusting the weighting functions. The controller effort can be modified by changing the $W2(s)$. Based on further requirements, we can modify them to suit our application and meet our robustness criteria based on the noise and uncertainty we expect to see in our system. The system synthesized has been found to have high margins to be consisdered as robustly stable.

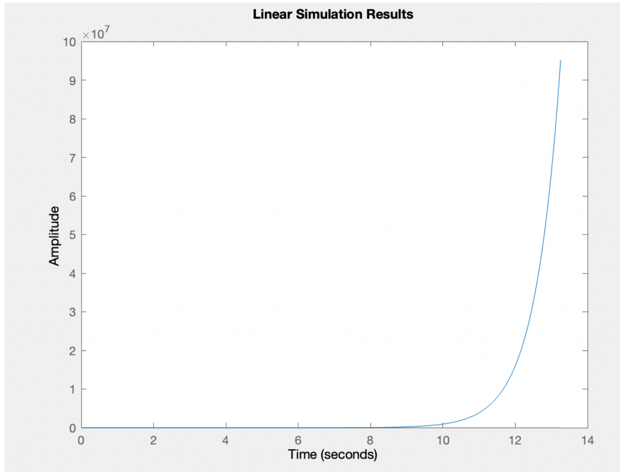

Figure 5: Controller effort

5.2 LQR Control

For the Q matrix in LQR control, we chose it be as

$$Q = C' * C \tag{29}$$

This gives us the following Q matrix:

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Additionally, R is chosen to be 1 which is the input weighting. This gives us a cheap control as the value of R is small. However, it may not be the optimal control as it causes a large change in states. If our control strategy needs smaller change in states, we need to increase the value of R accordingly. This will give us an expensive control. Using these parameters, lqr function was used in MATLAB to generate the control matrix K. The closed loop control was formulated and a stable system was designed. The step response of the system is as shown below in Figure 6.

But we see that this gives us a high rise and settling time. , we see that the pendulum moves quite a bit before stabilizing. To lower these parameters, we have chosen $Q(1,1)$ to be 3000 and $Q(3,3)$ to be 300. The first element represents weighting on the cart position and the second element represents the weighting on the pendulum angle. The modified Q matrix is as shown:

$$Q = \begin{bmatrix} 3000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 300 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The new step response is as shown below in Figure 7.

We can see that we got a much lower settling time. Further, if we have a desired rise time, settling

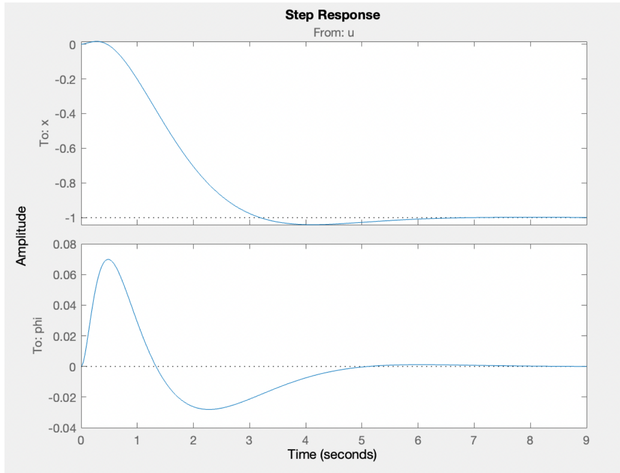

Figure 6: Step response of the closed loop system of basic LQR controller

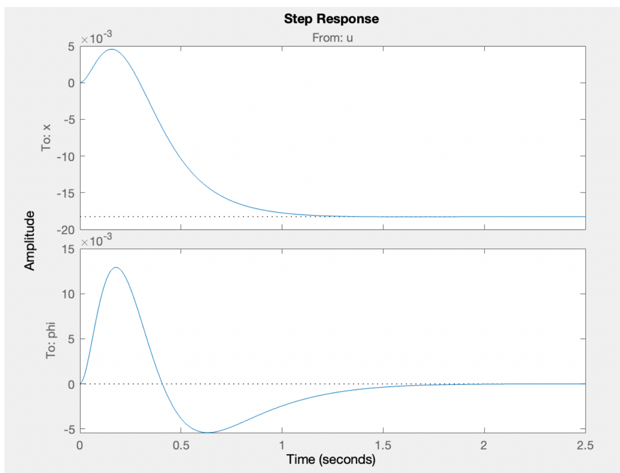

Figure 7: Step response of the closed loop system with modified Q matrix

time and or other parameter optimization, we can adjust the non-zero elements in the Q matrix to achieve the desired performance. Based on our performance objective and cost, we can chose the Q and R values appropriately.

Additionally, we see that the system is robustly stable as the margins of the system were high enough to handle uncertainty.

6 Conclusion

First, the study was focussed on creating controls for an inverted pendulum on a cart system. There were two controllers considered: the H_∞ and LQR controllers.

The H_∞ controller was developed to be resilient against system uncertainties and noise. It employed an optimization strategy to reduce the effects of disturbances on the system and ensure stability. The H_∞ controller was demonstrated to be successful at system stabilization under various levels of noise and uncertainty. It does, however, come at a cost in terms of performance as we saw with the settling time and the control effort.

The LQR controller, on the other hand, was meant to minimize a cost function that weighed the importance of the various state variables. The cost function was chosen to strike a balance between the relevance of the state variables and the work required to control them. This controller was found to be effective in system stabilization as well, but with superior settling time performance than the H_∞ controller.

Overall, the controller of choice is determined by the application and the trade-off between robustness and performance. When the system is subjected to high levels of noise and uncertainty, where robustness is critical, the H_∞ controller may be more appropriate, but the LQR controller may be better suited when performance is critical and the system is subjected to low levels of noise and uncertainty. Further the controllers can be modified by changing the appropriate parameters to achieve the desired response.

Finally, the controller architecture shown in this project is an appropriate option for stabilizing an inverted pendulum on a cart system. The study highlighted how several control approaches may be employed to achieve stability, and how the controller selection is dependent on the unique application needs. The results of the project could be further refined and applied to other systems in the future.

7 References

1. Bugeja, Marvin. (2003). Non-linear swing-up and stabilizing control of an inverted pendulum system. Computer as a Tool. 2. 437 - 441 vol.2. 10.1109/EURCON.2003.1248235.

2. A. K. Yadav, P. Gaur, A. P. Mittal and M. Anzar, "Comparative analysis of various control techniques for inverted pendulum," India International Conference on Power Electronics 2010 (IICPE2010), New Delhi, India, 2011, pp. 1-6, doi: 10.1109/IICPE.2011.5728071

3. MATLAB resources

4. Y. Xin, J. Xu, B. Xu and H. Xin, "The inverted-pendulum model with consideration of pendulum resistance and its LQR controller," Proceedings of 2011 International Conference on Electronic Mechanical Engineering and Information Technology, Harbin, China, 2011, pp. 3438-3441, doi: 10.1109/EMEIT.2011.6023822.

5. X. Jiarui, Z. Chunguang and W. Ruixin, "Single Stage Inverted Pendulum Controller Design," 2021 3rd International Conference on Applied Machine Learning (ICAML), Changsha, China, 2021, pp. 441-444, doi: 10.1109/ICAML54311.2021.00098.

A Appendix A: Code Listings

Contents

- System parameters
- Generation of state space representation
- LQR Control
- Modified LQR
- Hinf Controller
- Controller effort for Hinf

System parameters

```
M = .5;
m = 0.2;
b = 0.1;
I = 0.006;
g = 9.8;
l = 0.3;
```

Generation of state space representation

```
p = I*(M+m)+M*m*l^2;

A = [0      1              0           0;
     0 -(I+m*l^2)*b/p  (m^2*g*l^2)/p   0;
     0      0              0           1;
     0 -(m*l*b)/p       m*g*l*(M+m)/p  0];
B = [     0;
     (I+m*l^2)/p;
          0;
        m*l/p];
C = [1 0 0 0;
     0 0 1 0];
D = [0;
     0];

states = {'x' 'x_dot' 'phi' 'phi_dot'};
inputs = {'u'};
outputs = {'x'; 'phi'};

sys= ss(A,B,C,D,'inputname',inputs,'outputname',outputs);

p = pole(sys);
```

LQR Control

```
Q = [1,0,0,0;...
    0,0,0,0;...
    0,0,1,0;...
    0,0,0,0];
R = 1;

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

sysfb = ss(A-B*K,B,C,D);
sysfb= ss(sysfb.A,sysfb.B,sysfb.C,sysfb.D,'statename',states,'inputname',inputs,'outputname',outputs);
[M1,wc1]=robstab(sysfb);
step(sysfb);
```

Modified LQR

```
Q2 = [3000,0,0,0;...
    0,0,0,0;...
    0,0,300,0;...
    0,0,0,0];
R2 = 1;

[K2,S2,P2] = lqr(sys,Q2,R2);

sysfb2 = ss(A-B*K2,B,C,D);
[M2,wc2]=robstab(sysfb2);
sysfb2= ss(sysfb2.A,sysfb2.B,sysfb2.C,sysfb2.D,'inputname',inputs,'outputname',outputs);
step(sysfb2);
```

Hinf Controller

```
% Weighting function design
W1 = makeweight(10e-5,2590,1.5);
W3 = makeweight(1.5,1742,0.01);

bode(W1,W3)
[Kinf,CLinf,gammainf]=mixsyn(sys,W1,5,W3);

h11 = tf([1],[1]);
h21 = tf(0);
h12 = tf(0);
h22 = tf([1],[1]);
H=[h11 h12;h21 h22];
inputs = {'u','dist'};
```

```
sysfb3=feedback(sys*Kinf,H);
sysfb3 = ss(sysfb3.A,sysfb3.B,sysfb3.C,sysfb3.D,'inputname',inputs,'outputname',outputs);


[y1,t,x1] = step(sysfb3); %get sim time only
t=t'; %sim timestep(sysfb3);
u = [ones(size(t)); zeros(size(t))];
lsim(sysfb3,u,t);
```

Controller effort for Hinf

```
lsim(Kinf,u,t)
```