EEMEE 6602 Modern Control Theory, Spring 2025
© Columbia University

Balancing a Unicycle Robot

Anthony Chen (zc2812)

Mechanical Engineer, Columbia University
New York, USA
anthony.c@columbia.edu

Abstract

This report is for the course project of EE6602: Modern Control Theory, Spring 2025. In this paper, we analyze the dynamic model of the Unicycle Robot on a slope, which is more complicated than the previous research. We linearize the original nonlinear model and use different linear control methods to achieve stability of the closed-loop system. Not only do we simulate the linearized model in MATLAB, but we also implement the linear controllers in the original nonlinear system and still manage to stabilize it. The MATLAB code and the Simulink environment setup are also posted on Github: **https://github.com/Anthony-ZC/Columbia-EE6602-Modern-Control-Theory-Project**.

Keywords: Unicycle Robot, Control, Lyapunov, $\mathcal{H}_2$ optimal control, $\mathcal{H}_\infty$ optimal control

1 Introduction

As a student learning robotics, I am interested in Humanoid Robot control and balancing. However, in the preliminary investigation, I found that Humanoid robot models like SLIP systems are hard to linearize since they usually do not work around equilibrium points and might encounter switching dynamic models during operation. Therefore, I chose a Unicycle Robot since its desired working point is an unstable equilibrium. Also, a Unicycle Robot is actually an "inverted pendulum", which is a famous and fundamental model in Humanoid Robot, on a rolling wheel. As a result, I think learning it might provide me with some of the basics for my study of nonlinear humanoid robots.

As we know, balancing an inverted pendulum is a relatively straightforward control problem when the base is fixed or moves in a constrained direction. However, the problem becomes significantly more complex when the pendulum is mounted on a rolling wheel, forming a unicycle model. In this case, the system must not only maintain the upright stability of the pendulum but also account for the dynamics of the wheel motion. This introduces strong nonlinearities and coupling between translational and rotational dynamics, making the control problem much more challenging. Moreover, I also want to investigate balancing the Unicycle on a slope with a given angle rather than just on a flat plane.

In the following part of the report, we will discuss the following:

- **Section 3**: Analysis of the dynamic model of the Unicycle model on a slope, with the different proposed linearized models.
- **Section 4**: Formulate the problem in a standard closed-loop description, introduce the linear and nonlinear simulation in MATLAB and Simulink, and state the goals of these paper.
- **Section 5**: Linear controller designs using the Lyapunov equation, $\mathcal{H}_2$ optimal control, $\mathcal{H}_\infty$ optimal control. In each sub-section, we will show the behavior of the closed-loop linearized model, the closed-loop nonlinear model under the linear controller, and their behavior under disturbance.
- **Section 6**: Balancing the nonlinear Unicycle Robot on time-variant slope angles using different controllers.

.

2 Literature Review

There has been a lot of research on how to achieve balance on unicycles. Some 3D version model analysis is appealing, but too complicated to reproduce [1][2]. So, I focus on a rather simple 2D version model, which models the Unicycle Robot as an inverted pendulum mounted to a wheel with state variables $\alpha, \dot{\alpha}, \beta, \dot{\beta}$ implying the angle and the angular velocity of the pendulum and the wheels [3][4].

Both these 2 papers [3][4] showed that a Unicycle Robot can be balanced on **a flat plane** using this simplified model with an LQR controller. However, they do not consider the non-flat slope case, nor the potential input disturbance to the linearized model.

3 System Modeling and Analysis

3.1 System Overview

I refine the model used in [3][4] to an on-slope version. As shown in **Figure 1**, the inclination of the slope is θ, the mass and inertia of the wheel are M_w and I_w, the radius of the wheel is R_w, the mass and inertia of the inverted pendulum are M_p and I_p, the length of the inverted pendulum is L, d denotes the distance that unicycle move on the slope, β denotes the angular of the wheel and α denotes the angular deviation (tilt angle) from plumb line. We assume that the mass of the inverted pendulum is concentrated at the end and that there is no slippage between the wheel and the ground. In this report, I use the physical parameters from the [3], which are shown in **Table 1**.

Hence, in this system model, the input is the voltage u of the DC motor of the wheel and the inclination of the slope θ, which is the actuator, the states of the system are $[\alpha, \dot{\alpha}, \beta, \dot{\beta},]^T$. For the system output, we assume that we have accurate encoders to measure the tilt angle α, wheel angle β and wheel velocity $\dot{\beta}$, but we can not measure the velocity of the tilt angle $\dot{\alpha}$, especially when the slope angle is changing over time.

Figure 1: Unicycle Model

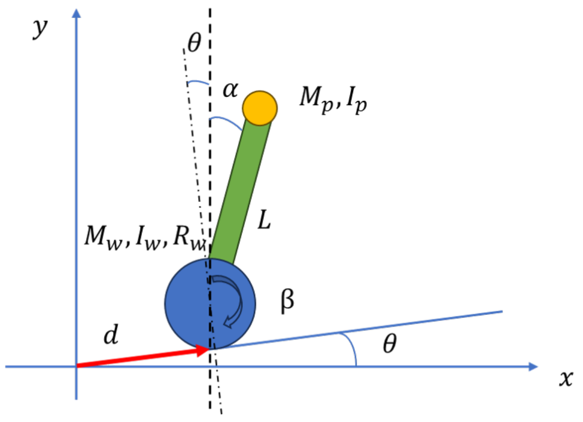


Table 1: System Parameters

Symbol	Parameter	Value
g	Gravity	$9.8\,\mathrm{m/s^2}$
M_p	Pendulum mass	$15.63\,\mathrm{kg}$
M_w	Wheel mass	$31.28\,\mathrm{kg}$
I_p	Pendulum inertia	$3.55\,\mathrm{kg\,m^2}$
I_w	Wheel inertia	$1.40\,\mathrm{kg\,m^2}$
R_w	Wheel radius	$0.31\,\mathrm{m}$
L	Distance between CG (wheel to body)	$0.155\,\mathrm{m}$
n	Gear ratio	19.56
k_T	Motor Torque constant	$0.050\,\mathrm{Nm}$
R	Motor Resistance	$1.2\,\Omega$
k_e	EMF constant	$0.0534\,\mathrm{V/(rad\,s^{-1})}$

3.2 Dynamic Model and Lagrangian Analysis

The translational and angular kinetic energy of this system is:

$$\begin{aligned} K &= \frac{1}{2}M_w V_w^2 + \frac{1}{2}M_p V_p^2 + \frac{1}{2}I_w\dot{\beta}^2 + \frac{1}{2}I_p\dot{\alpha}^2 \\ &= \frac{1}{2}M_w(\dot{d})^2 + \frac{1}{2}M_p\left[(\dot{d} + \dot{\alpha}L\cos(\alpha+\theta))^2 + (\dot{\alpha}L\sin(\alpha+\theta))^2\right] + \frac{1}{2}I_w\frac{\dot{d}^2}{R_2^2} + \frac{1}{2}I_p\dot{\alpha}^2 \end{aligned} \tag{1}$$

$$P = M_w g(R_w + d\sin\theta) + M_p g\left(R_w + d\sin\theta + \frac{1}{2}L\cos\alpha\right) \tag{2}$$

Using the Lagrangian equation

$$\begin{gathered} L = K - P \\ \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \left(\frac{\partial L}{\partial q}\right) = Q \end{gathered} \tag{3}$$

Where $q = [\alpha, d]$, and Q is the motor torque.

$$(M_p L^2 + I_p)\ddot{\alpha} + \ddot{d}M_p L\cos(\alpha+\theta) - M_p L g\sin(\alpha+\theta) = -\tau \tag{4}$$

$$\left(M_w + M_p + \frac{I_w}{R_w^2}\right)\ddot{d} + \ddot{\alpha}M_p L\cos(\alpha+\theta) - \dot{\alpha}^2 M_p L\sin(\alpha+\theta) + (M_w + M_p)g\sin\theta = \frac{\tau}{R_w} \tag{5}$$

Using a model of the DC motor mentioned in [3]:

$$\tau = \frac{k_T n}{R} u - \frac{k_T k_e n^2}{R} \dot{\beta} \tag{6}$$

where u denotes the input voltage of the motor, n denotes motor transmission gear ratio, k_T denotes motor torque constant, k_e denotes motor damping constant and R denotes motor resistance.
With above motor model and the constraints $\dot{d} = R_w \dot{\theta}, \ddot{d} = R_w \ddot{\theta}$, we can get the final dynamic equations:

$$(M_p L^2 + I_p)\ddot{\alpha} + (M_p R_w L \cos(\alpha + \theta))\ddot{\beta} = \frac{n^2 k_T k_e}{R} \dot{\beta} - \frac{n k_T}{R} u + M_p L g \sin(\alpha + \theta) \tag{7}$$

$$(M_p R_w L \cos(\alpha+\theta))\ddot{\alpha} + (M_w R_w^2 + M_p R_w^2 + I_w)\ddot{\beta} = -\frac{n^2 k_T k_e}{R} \dot{\beta} + \frac{n k_T}{R} u - (M_w + M_p) R_w g \sin\theta + \dot{\alpha}^2 M_p R_w L \sin(\alpha+\theta) \tag{8}$$

3.3 Equilibrium Analysis

Given the nonlinear dynamics equations, for equilibrium, we should have $\dot{\alpha} = 0, \ddot{\alpha} = 0, \ddot{\beta} = 0, \dot{\beta} = 0$ and limited β, then:

$$\begin{aligned} 0 &= -\frac{n k_T}{R} u + M_p L g \sin(\alpha + \theta) \\ 0 &= \frac{n k_T}{R} u - (M_w + M_p) R_w g \sin\theta \end{aligned} \tag{9}$$

Then we have 2 cases:

$$\begin{cases} -\frac{n k_T}{R} u + M_p L g \sin(\alpha + \theta) = \frac{n k_T}{R} u - (M_w + M_p) R_w g \sin\theta & \textbf{case1} \\ -\frac{n k_T}{R} u + M_p L g \sin(\alpha + \theta) = -\frac{n k_T}{R} u + (M_w + M_p) R_w g \sin\theta & \textbf{case2} \end{cases}$$

For case 1, things are hard to analyze, because we will get:

$$\alpha^* = \arcsin\left(\frac{2 n k_T}{R} u - \frac{(M_w + M_p) R_w}{M_p L} \sin\theta\right) - \theta \tag{10}$$

in which the equilibrium is dependent on the input.
Hence, in the result of the report, we only focus on case 2, we have:

$$\alpha^* = \arcsin\left(\frac{(M_w + M_p) R_w}{M_p L} \sin\theta\right) - \theta \tag{11}$$

in which the equilibrium is independent of the input, a much easier case.
Using parameters in **Table 1**, we have

$$\alpha = \arcsin(6.0026 \sin\theta) - \theta \tag{12}$$

$$\Rightarrow -1 \leq 6.0026 \sin\theta \leq 1 \equiv -0.1674 \leq \theta \leq 0.1674 \tag{13}$$

which gives us a maximum and minimum boundary of the slope angle.

3.4 Linearization

Previous research assumed a flat plane $\theta = 0$. So, it is naturally to linearized around $\alpha^* = 0$, which satisfies both $\alpha^* = -\theta$ and α^* is also an equilibrium. In the slope case, these 2 cases are separated, and we can come up with different linearized models. We will later show that both these linearized models can be stabilized, and they can be used to stabilize the original nonlinear Unicycle Robot, though giving different performance.

3.4.1 Linearize around $\alpha = -\theta$

For any θ,we can linearize around $\alpha = -\theta$, then $\sin(\alpha + \theta) \approx (\alpha + \theta), \cos(\alpha + \theta) \approx 1, \dot{\alpha}^2 \approx 0$ and we have:

$$(M_p L^2 + I_p)\ddot{\alpha} + (M_p R_w L)\ddot{\beta} = \frac{n^2 k_T k_e}{R} \dot{\beta} - \frac{n k_T}{R} u + M_p L g(\alpha + \theta) \tag{14}$$

$$(M_p R_w L)\ddot{\alpha} + (M_w R_w^2 + M_p R_w^2 + I_w)\ddot{\beta} = -\frac{n^2 k_T k_e}{R} \dot{\beta} + \frac{n k_T}{R} u - (M_w + M_p) R_w g \sin\theta \tag{15}$$

$$\begin{bmatrix}\dot\alpha\\ \ddot\alpha\\ \dot\beta\\ \ddot\beta\end{bmatrix} = \underbrace{\begin{bmatrix}0 & 1 & 0 & 0\\ \frac{P_1P_4}{P_1P_3-P_2^2} & 0 & 0 & \frac{(P_1+P_2)P_5nk_e}{P_1P_3-P_2^2}\\ 0 & 0 & 0 & 1\\ -\frac{P_2P_4}{P_1P_3-P_2^2} & 0 & 0 & -\frac{(P_2+P_3)P_5nk_e}{P_1P_3-P_2^2}\end{bmatrix}}_{A^{-\theta}} \begin{bmatrix}\alpha\\ \dot\alpha\\ \beta\\ \dot\beta\end{bmatrix} + \underbrace{\begin{bmatrix}0\\ -\frac{(P_1+P_2)P_5}{P_1P_3-P_2^2}\\ 0\\ \frac{(P_2+P_3)P_5}{P_1P_3-P_2^2}\end{bmatrix}}_{B^{-\theta}} u + \underbrace{\begin{bmatrix}0\\ \frac{P_1P_4\theta+P_2P_6\sin\theta}{P_1P_3-P_2^2}\\ 0\\ -\frac{P_2P_4\theta+P_3P_6\sin\theta}{P_1P_3-P_2^2}\end{bmatrix}}_{H^{-\theta}(\theta)} \tag{16}$$

where:

$$P_1 = M_wR_w^2 + M_pR_w^2 + I_w, P_2 = M_pR_wL, P_3 = M_pL^2 + I_p, P_4 = M_pLg, P_5 = \frac{nk_T}{R}, P_6 = (M_w + M_p)R_w$$

Noticing that although we use the notation that $A^{-\theta}, B^{-\theta}$, they are **independent** of θ, which is different from the next linearization. Also, we have a nonlinear part $H^{-\theta}(\theta)$, which depends on θ. Let's just ignore it for simplification, and we will later show that ignoring it will not affect the steady-state performance when stabilizing the nonlinear system.
Using the parameter mentioned in **Table 1** we have

$$A^{-\theta} = \begin{bmatrix}0 & 1 & 0 & 0\\ 6.1989 & 0 & 0 & 0.2505\\ 0 & 0 & 0 & 1\\ -0.788 & 0 & 0 & -0.1759\end{bmatrix}, B^{-\theta} = \begin{bmatrix}0\\ -0.2398\\ 0\\ 0.1684\end{bmatrix} \tag{17}$$

Checking the eigenvalues of the $A^{-\theta}$, which are $\lambda = [0, 2.4748, -2.5067 - 0.1440]$, we notice that the linearized system is inherently unstable.
It can be certified that this system is fully controllable:

$$C_{AB}^{-\theta} = \begin{bmatrix}B^{-\theta} & A^{-\theta}B^{-\theta} & A^{-\theta^2}B^{-\theta} & A^{-\theta^3}B^{-\theta}\end{bmatrix}, \mathbf{rank}(C_{AB}^{-\theta}) = 4$$

Recalling that we assume we can only observe $\alpha.\beta, \dot\beta$, which give us $C = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}$,we can also certified that this system is fully observable:

$$O_{AC}^{-\theta} = \begin{bmatrix}C^{-\theta}\\ A^{-\theta}C_2^{-\theta}\\ A^{-\theta^2}C^{-\theta}\\ A^{-\theta^3}C^{-\theta}\end{bmatrix}, \mathbf{rank}(O_{AB}^{-\theta}) = 4$$

3.4.2 Linearize around equilibrium $\alpha = \arcsin\left(\frac{(M_w+M_p)R_w}{M_pL}\sin\theta\right) - \theta$

For any θ,we can linearize around $\alpha^* = \arcsin\left(\frac{(M_w+M_p)R_w}{M_pL}\sin\theta\right) - \theta$, and using first order Taylor expansion, we have:

$$\begin{gathered}\sin(\alpha+\theta) \approx \sin(\alpha^*+\theta) + \cos(\alpha^*+\theta)(\alpha-\alpha^*)\\ \cos(\alpha+\theta)\ddot\beta \approx \cos(\alpha^*+\theta)(\ddot\beta-\ddot\beta^*) - \sin(\alpha^*+\theta)\ddot\beta^*(\alpha-\alpha^*) = \cos(\alpha^*+\theta)\ddot\beta \Leftarrow \ddot\beta^* = 0\\ \cos(\alpha+\theta)\ddot\alpha \approx \cos(\alpha^*+\theta)(\ddot\alpha-\ddot\alpha^*) - \sin(\alpha^*+\theta)\ddot\alpha^*(\alpha-\alpha^*) = \cos(\alpha^*+\theta)\ddot\alpha \Leftarrow \ddot\alpha^* = 0\\ \dot\alpha^2 \approx 0\end{gathered}$$

Then we have:

$$(M_pL^2+I_p)\ddot\alpha + (M_pR_wL\cos(\alpha^*+\theta))\ddot\beta = \frac{n^2k_Tk_e}{R}\dot\beta - \frac{nk_T}{R}u + M_pLg\cos(\alpha^*+\theta)(\alpha-\alpha^*) + M_pLg(\sin(\alpha^*+\theta)) \tag{18}$$

$$(M_pR_wL\cos(\alpha^*+\theta))\ddot\alpha + (M_wR_w^2 + M_pR_w^2 + I_w)\ddot\beta = -\frac{n^2k_Tk_e}{R}\dot\beta + \frac{nk_T}{R}u - (M_w+M_p)R_wg\sin\theta \tag{19}$$

Here, we can actually come up with 2 models, in the first model, we set the state variables as $x = [\alpha-\alpha^*, \dot\alpha, \beta, \dot\beta]^T$, which gives us:

$$\begin{bmatrix}\dot\alpha\\ \ddot\alpha\\ \dot\beta\\ \ddot\beta\end{bmatrix} = \underbrace{\begin{bmatrix}0 & 1 & 0 & 0\\ \frac{P_1P_4'}{P_1P_3-P_2'^2} & 0 & 0 & \frac{(P_1+P_2')P_5nk_e}{P_1P_3-P_2'^2}\\ 0 & 0 & 0 & 1\\ -\frac{P_2'P_4'}{P_1P_3-P_2'^2} & 0 & 0 & -\frac{(P_2'+P_3)P_5nk_e}{P_1P_3-P_2'^2}\end{bmatrix}}_{A^{eq}(\theta)} \begin{bmatrix}\alpha-\alpha^*\\ \dot\alpha\\ \beta\\ \dot\beta\end{bmatrix} + \underbrace{\begin{bmatrix}0\\ -\frac{(P_1+P_2')P_5}{P_1P_3-P_2'^2}\\ 0\\ \frac{(P_2'+P_3)P_5}{P_1P_3-P_2'^2}\end{bmatrix}}_{B^{eq}(\theta)} u + \underbrace{\begin{bmatrix}0\\ \frac{P_1P_7+P_2'P_6\sin\theta}{P_1P_3-P_2'^2}\\ 0\\ -\frac{P_2'P_7+P_3P_6\sin\theta}{P_1P_3-P_2'^2}\end{bmatrix}}_{H^{eq}(\theta)} \tag{20}$$

where:

$$P_1 = M_w R_w^2 + M_p R_w^2 + I_w, P_2' = M_p R_w L \cos(\alpha^* + \theta), P_3 = M_p L^2 + I_p$$

$$P_4' = M_p L g \cos(\alpha^* + \theta), P_5 = \frac{nk_T}{R}, P_6 = (M_w + M_p) R_w, P_7 = M_p L g \sin(\alpha^* + \theta)$$

In the second model, we set the state variables as $x = [\alpha, \dot{\alpha}, \beta, \dot{\beta}]^T$, which gives us:

$$\begin{bmatrix} \dot{\alpha} \\ \ddot{\alpha} \\ \dot{\beta} \\ \ddot{\beta} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{P_1 P_4}{P_1 P_3 - P_2'^2} & 0 & 0 & \frac{(P_1 + P_2') P_5 n k_e}{P_1 P_3 - P_2'^2} \\ 0 & 0 & 0 & 1 \\ -\frac{P_2' P_4}{P_1 P_3 - P_2'^2} & 0 & 0 & -\frac{(P_2' + P_3) P_5 n k_e}{P_1 P_3 - P_2'^2} \end{bmatrix}}_{A^{eq}(\theta)} \begin{bmatrix} \alpha \\ \dot{\alpha} \\ \beta \\ \dot{\beta} \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ -\frac{(P_1 + P_2') P_5}{P_1 P_3 - P_2'^2} \\ 0 \\ \frac{(P_2' + P_3) P_5}{P_1 P_3 - P_2'^2} \end{bmatrix}}_{B^{eq}(\theta)} u + \underbrace{\begin{bmatrix} 0 \\ \frac{P_1 P_7' + P_2 P_6 \sin\theta}{P_1 P_3 - P_2'^2} \\ 0 \\ -\frac{P_2 P_7' + P_3 P_6 \sin\theta}{P_1 P_3 - P_2'^2} \end{bmatrix}}_{H^{eq2}(\theta)} \tag{21}$$

where:

$$P_1 = M_w R_w^2 + M_p R_w^2 + I_w, P_2' = M_p R_w L \cos(\alpha^* + \theta), P_3 = M_p L^2 + I_p$$

$$P_4' = M_p L g \cos(\alpha^* + \theta), P_5 = \frac{nk_T}{R}, P_6 = (M_w + M_p) R_w, P_7' = M_p L g (\sin(\alpha^* + \theta) - \cos(\alpha^* + \theta)\alpha^*)$$

Noticing that $A^{eq}(\theta), B^{eq}(\theta)$ **depend** on θ. Also, changing the state variables' representation only affects the nonlinear part $H^{eq1/2}(\theta)$, but not the linearized model. Luckily, by performing a grid check through $\theta \in [-0.1674, 0.1674]$, we can ensure that $\forall \theta \in [-0.1674, 0.1674]$, we have:

- $A^{eq}(\theta)$ has at least one positive eigenvalue, the linearized system is inherently unstable.
- $\mathbf{rank}(C_{AB}^{eq}(\theta)) = 4$, the linearized system is controllable.
- Given $C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$, $\mathbf{rank}(O_{AC}^{eq}(\theta)) = 4$, the linearized system is observable.

Noticing that if $\theta = 0$, we have $A^{-\theta} = A^{eq}(0), B^{-\theta} = B^{eq}(0), H^{-\theta}(0) = H^{eq1}(0) = H^{eq2}(0) = 0$.

4 Problem statement

4.1 Closed-loop Formulation

Knowing that both linearized open-loop models are inherently unstable, we need to design a controller to achieve closed-loop stability.

As shown in **Figure 2**, we assume the open-loop plant G of the linearized Unicycle Robot system has the following output map:

$$\dot{x} = Ax(t) + \begin{bmatrix} B_1 & B_2 \end{bmatrix} \begin{bmatrix} w(t) \\ u(t) \end{bmatrix} \tag{22}$$

$$\begin{bmatrix} z(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} x(t) + \begin{bmatrix} D_{11} & D_{12} \\ D21 & D22 \end{bmatrix} \begin{bmatrix} w(t) \\ u(t) \end{bmatrix} \tag{23}$$

where, $A = A^{-\theta}/A^{eq}(\theta), B_2 = B^{-\theta}/B^{eq}(\theta)$. The output matrix $C_2 = I_{4\times4}$ for the state feedback case and $C_2 = C$ for the output feedback case.
We want to regularize all the state variables, which means $C_1 = I_{4\times4}$.
We assume that the exogenous input (disturbance) w is due to the potential motor voltage vibration, which means $B_1 = B_2$.
We assume that no feedforward loop in the system, which means $D_{11} = D_{12} = D_{21} = D_{22} = 0$.
We make the following notations:

- $G_{-\theta}$ denotes the linearized model around $\alpha = -\theta$ with state variables $x = [\alpha, \dot{\alpha}, \beta, \dot{\beta}]^T$ (**Equation 16**), which does not depend on θ.
- $G_{eq1}(\theta)$ denotes the linearized model around equilibrium with state variables $x = [\alpha - \alpha^*, \dot{\alpha}, \beta, \dot{\beta}]^T$ (**Equation 20**), which depends on θ.
- $G_{eq2}(\theta)$ denotes the linearized model around equilibrium with state variables $x = [\alpha, \dot{\alpha}, \beta, \dot{\beta}]^T$ (**Equation 21**), which depends on θ.

The **MATLAB Code 1 and 2** for defining the above systems are included in the Appendix.

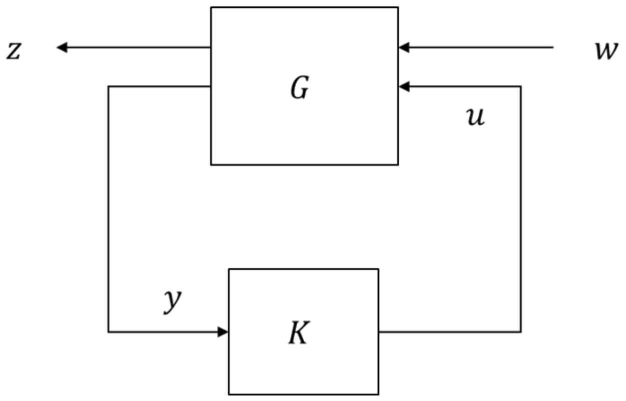


Figure 2: Closed-loop Diagram

The closed-loop controller $K = \left[\begin{array}{c|c} A_K & B_K \\ \hline C_K & D_K \end{array}\right]$ is dynamic:

$$\begin{aligned} \dot{x}_K(t) &= A_K x_K(t) + B_K y(t) \\ u(t) &= C_K x_K(t) + D_K y(t) \end{aligned} \tag{24}$$

4.2 Simulation pre-setting

Unless stated otherwise, the following simulation settings will be used throughout the rest of the paper, including nonlinear simulation:

- Initial condition for the plant and the dynamic controller$x_0 = [0.1, -0.1, 0, 0.1]^T, x_{K0} = [0, 0, 0, 0]^T$
- $\theta = 0.1 \Rightarrow \alpha^* \approx 0.5426$ for $G_{eq1}(\theta), G_{eq2}(\theta)$ and the nonlinear simulation
- α has a natural limitation that $-\pi/2 - \theta \leq \alpha \leq \pi/2 - \theta$ ($\approx [-1.67, 1.47]$ for $\theta = 0.1$), indicating that the pendulum cannot hit the slope
- we simulate 30s in the linearized and nonlinear models by default

4.3 Linearized Model Simulation in MATLAB

We can do the open-loop simulation with the above non-zero initial conditions for the $G_{-\theta}, G_{eq1/2}(\theta = 0.1)$ using *lsim* in MATLAB, which is implemented through **MATLAB Code 3**. Recalling that $G_{eq1}(\theta), G_{eq2}(\theta)$ share the same state matrices. Hence, we only do the linear simulation once. The results are shown in **Figure 3, 4**. We can see all state variables explode, which is fitted with the previous analysis.

For the linear closed-loop simulation, with the open-loop plant and the controller, we can get the mathematical formulation shown below:

$$\mathcal{F}_l(G, K) = \left[\begin{array}{c|c} \mathcal{A}_{cl} & \mathcal{B}_{cl} \\ \hline \mathcal{C}_{cl} & \mathcal{D}_{cl} \end{array}\right] =$$

$$\left[\begin{array}{c|c} \begin{bmatrix} A & 0 \\ 0 & A_K \end{bmatrix} + \begin{bmatrix} B_2 & 0 \\ 0 & B_K \end{bmatrix} \begin{bmatrix} I & -D_{22} \\ -D^K & I \end{bmatrix}^{-1} \begin{bmatrix} 0 & C_K \\ C_2 & 0 \end{bmatrix} & \begin{array}{c} B_1 + B_2 D_K Q D_{21} \\ B_K Q D_{21} \end{array} \\ \hline \begin{bmatrix} C_1 & 0 \end{bmatrix} + \begin{bmatrix} D_{12} & 0 \end{bmatrix} \begin{bmatrix} I & -D_{22} \\ -D^K & I \end{bmatrix}^{-1} \begin{bmatrix} 0 & C_K \\ C_2 & 0 \end{bmatrix} & D_{11} + D_{12} D_K Q D_{21} \end{array}\right] \tag{25}$$

where $Q = (I - D_K D22)^{-1}$, the input is disturbance w, the state variables is $x_{cl} = [x, x_K]^T$, output is z. Whenever we get a controller $K = \left[\begin{array}{c|c} A_K & B_K \\ \hline C_K & D_K \end{array}\right]$, we can use the **Equation 25** to establish to closed-loop system and use *lsim* in MATLAB to do the simulation as open-loop.

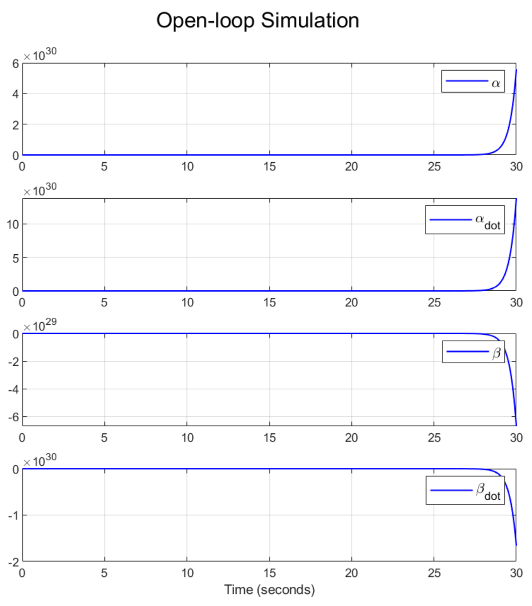


Figure 3: Open-loop Simulation for $G_{-\theta}$

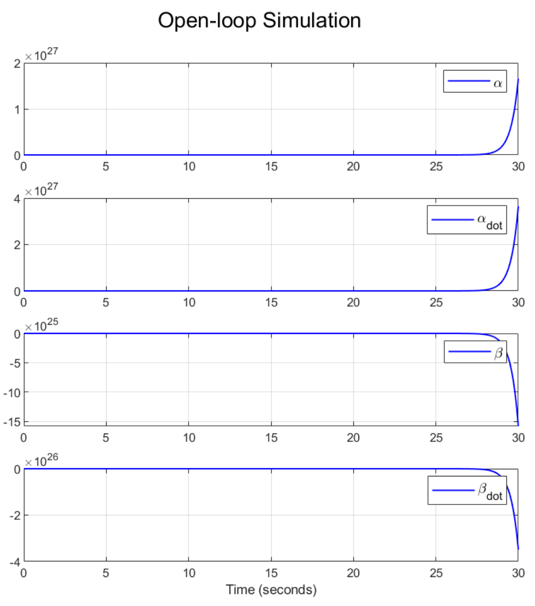


Figure 4: Open-loop Simulation for $G_{eq1/2}(\theta = 0.1)$

4.4 Nonlinear Model Simulation in Simulink

Besides the linear simulation, we also want to see whether the linear controllers we obtain in the linearized model can stabilize the original **nonlinear** Unicycle Robot. Hence, I built the simulation environment in Simulink, which is shown in Figure 5.

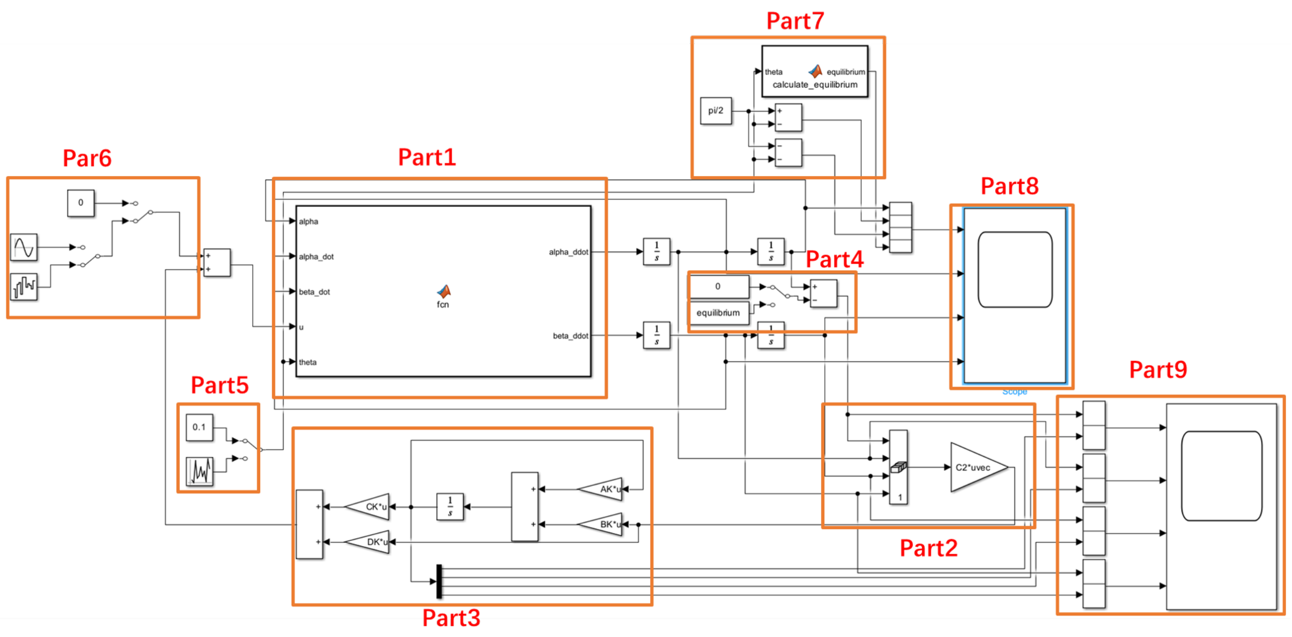


Figure 5: Nonlinear Simulation of the Unicycle Robot in Simulink

The simulation configuration mainly consists of the following parts:

- **Part1:** The nonlinear dynamic equations of the Unicycle Robot (**Equation 7, 8**), which is implemented by **MATLAB Code 4** in the Appendix.
- **Part2:** The measured output $y = C_2 x$.
- **Part3:** The linear feedback controller/observer $K = \left[\begin{array}{c|c} A_K & B_K \\ \hline C_K & D_K \end{array}\right]$

- **Part4:** Observed state adjustment, recalling that for $G_{-\theta}, G_{eq2}(\theta)$, we have state variable $\alpha - 0$, so α is directly feed into the controller; for $G_{eq1}(\theta)$, we have state variable $\alpha - \alpha^*$, so α subtract by the equilibrium is feed into the controller
- **Part5:** The slope input for the nonlinear dynamics, allowing us to select a constant slope angle or a uniformly sampled slope angle, which will be used in **Section 6**
- **Part6:** The exogenous input w is introduced to the input as a disturbance, allowing us to select 0, sine wave, and Gaussian noise.
- **Part7:** The assistance plot of the natural limits for α, also the assistance plot of equilibrium.
- **Part8:** The plot of the **physical variables** of the Unicycle Robot $[\alpha, \dot{\alpha}, \beta, \dot{\beta}]^T$
- **Part9:** The plot of the real state variables x and the estimated states x_K observed by the observer.

Similar to the linear closed-loop system's notation $\mathcal{F}_l(G, K)$, we will use notation $\mathcal{F}_{nl}(G, K)$ **for the nonlinear closed-loop system**, which means we perform a linear feedback controller K designed through the linearized model G on the nonlinear open-loop Unicycle Robot.

We can also do the open-loop simulation (zero input), and the result is shown in **Figure 6**, which also shows that the open-loop system is unstable.

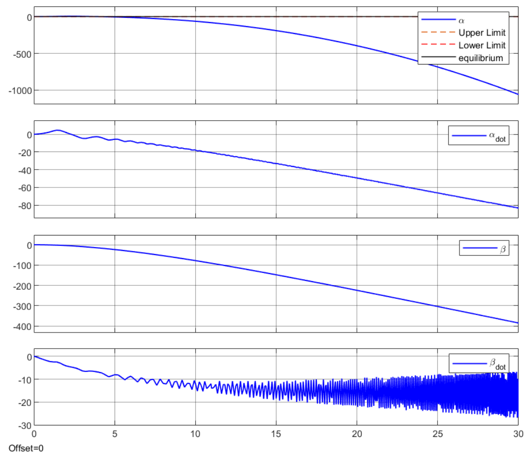


Figure 6: Open-loop Nonlinear Model Simulation

4.5 Design Goals

Knowing all the open-loop linearized systems and the original nonlinear system are inherently unstable, the goals of this report are:

1. Design a controller that stabilizes the linearized models.
2. Stabilize the nonlinear system of the Unicycle Robot using a linear controller.
3. Investigate which linearized model better describes the nonlinear model (based on the second goal).
4. On top of that, design a controller that can reject the input disturbance, for example, Gaussian noise, for the linearized and non-linear systems.
5. Balance the Unicycle Robot on a time-variant (changing angle) slope using a linear controller.

5 Results of Controller Design and Correspondent Simulation

5.1 Output Feedback Controller using Lyapunov Equation

5.1.1 LMI Formulation

At the very beginning, we want to deploy a fundamental controller, an observer-based output feedback controller by solving the Lyapunov Equation. Since we have already ensured that both linearized models $G_{-\theta}, G_{eq1/2}(\theta)$ are controllable and observable, we can use the Separation Principle to design the controller and observer separately.

Since we assume a non-full-state measurement, we need to design an observer. Here, I use the Luenberger Observer taught in the course:

$$\dot{\hat{x}}(t) = (A + LC)\hat{x}(t) - Ly(t) + (B + LD)u(t) \tag{26}$$

where the observer gain L is to make sure the stability of the observer, which can be obtained by solving the Lyapunov Equation-based LMI:

$$\begin{gathered} W \succ 0, V \in \mathbb{R}^{p_y \times n} \\ A^T W + WA + C_2^T V + V^T C_2 \prec 0 \end{gathered} \tag{27}$$

where $L = W^{-1}V^T$.

For the controller, we have $u = F\hat{x}$, which can be also obtained by solving an LMI:

$$\begin{gathered} P \succ 0, Z \in \mathbb{R}^{m \times n} \\ AP + PA^T + B_2 Z + Z^T B^T \prec 0 \end{gathered} \tag{28}$$

where $F = ZP^{-1}$.

After obtaining L and K by solving LMI **27, 28** through CVX, we can rewrite this Luenberger Observer-based output feedback controller in a standard form (since $D_{11} = D_{12} = D_{21} = D_{22} = 0$, we do not include them):

$$K_1 = \left[\begin{array}{c|c} A + LC_2 + B_2 F & -L \\ \hline F & 0 \end{array}\right] \tag{29}$$

5.1.2 Simulation Results

Having obtained the controller K_1, we can establish the closed-loop system through **Formula 25**, then do the simulation in MATLAB using *lsim*. The whole designing process and the simulation are implemented by **MATLAB Code 5** in the Appendix.

First, we look at the linear simulation results of $\mathcal{F}_l(G_{-\theta}, K_1)$ and $\mathcal{F}_l(G_{eq1/2}(\theta = 0.1), K_1)$, which are shown in **Figure 7, 8**. We can see that in both linearized models, the designed Lyapunov-based output feedback controller can stabilize the system, and the observer manages to track all the state variables perfectly (red-dash lines denote the states estimated by the observer), which meets the theorem. Finally, for $\mathcal{F}_l(G_{-\theta}, K_1)$ and $\mathcal{F}_l(G_{eq1}(\theta), K_1)$, $-0.2 < \alpha < 0.1$ is in the safe region $[-1.67, 1.47]$, same for $\mathcal{F}_l(G_{eq1/2}(\theta), K_1)$ that $-0.2 < \alpha - \alpha^* < 0.1 \Rightarrow -0.3426 < \alpha < 0.6426$.

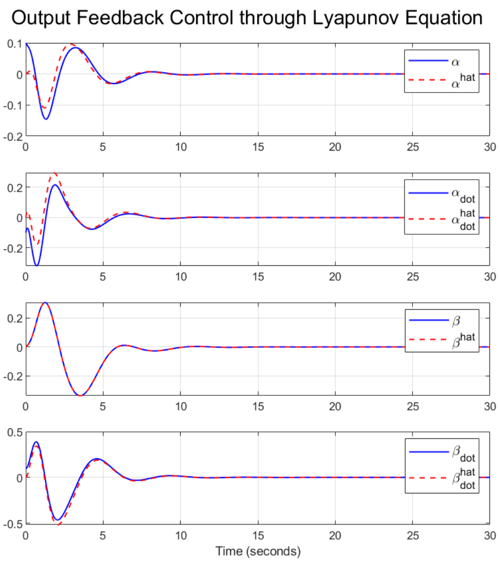


Figure 7: Linear Simulation of $\mathcal{F}_l(G_{-\theta}, K_1)$

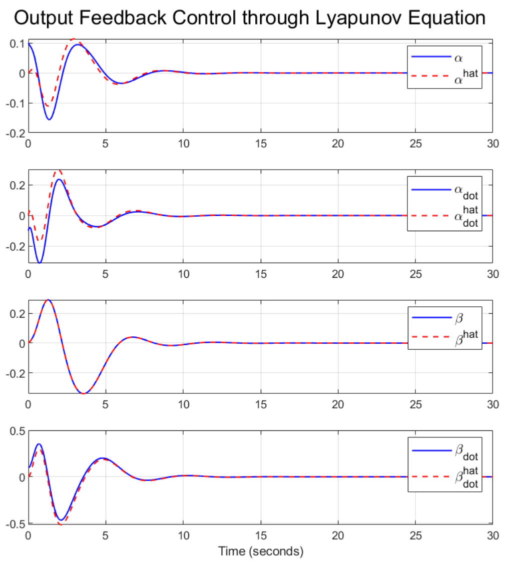


Figure 8: Linear Simulation of $\mathcal{F}_l(G_{eq1/2}(\theta), K_1)$

However, if we implement this linear controller in the nonlinear model in Simulink and do the nonlinear simulation, which are shown in **Figure 9, 10, 11**, things get different. Although all 3 controllers manage to stabilize the system to the equilibrium ($\alpha = \alpha^* \approx 0.5426, \dot{\alpha} = 0, \dot{\beta} = 0$), the transient behaviors of the nonlinear system are quite different. It is more clear to see the different when overlapping the trajectories of 4 physical variables $\alpha, \dot{\alpha}, \beta, \dot{\beta}$ under 3 different linearized models' controllers in **Figure 12**, in which blue lines refer to $\mathcal{F}_{nl}(G_{-\theta}, K_1)$, cyan dash lines refer to $\mathcal{F}_{nl}(G_{eq1}(\theta), K_1)$ and orange dot-dashed lines refer to $\mathcal{F}_{nl}(G_{eq2}(\theta), K_1)$.

We can see that over 3 linear models' controllers, the controller from $G_{eq1}(\theta)$ gives the worst performance, although modeling state variable as $\alpha - \alpha^*$ seems more reasonable than α, the controller makes multiple physical variables oscillate violently before stabilizing. What makes things worse is that α crosses the limitation during this process, which means the pendulum hits the slope, so it is unacceptable.

For $\mathcal{F}_{nl}(G_{-\theta}, K_1)$ and $\mathcal{F}_{nl}(G_{eq2}(\theta), K_1)$, we can see that the controller designed using $G_{eq2}(\theta)$ has better performance, not only does it make the nonlinear system response quicker with smaller overshot over all 4 physical variables, but also it gives the smallest β deviation from the zero point. Although achieving the balance of the Unicycle Robot does not put a limit on how far the robot moves before stabilizing, we still want to make its moving distance as small as possible. Hence, we can somehow say the linearized model $G_{eq2}(\theta)$ can better describe the nonlinear system if the slope angle is known.

As a result, in the rest of the report, we will only use $G_{eq2}(\theta)$ as the only linearized model for the report to be more concise, and we give the simplified notation that $G_l(\theta) = G_{eq2}(\theta)$.

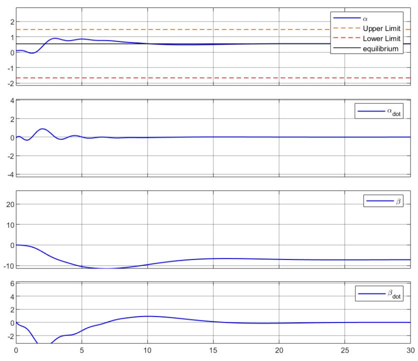


Figure 9: Nonlinear Simulation of $\mathcal{F}_{nl}(G_{-\theta}, K)$

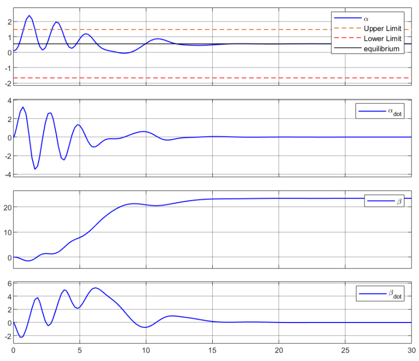


Figure 10: Nonlinear Simulation of $\mathcal{F}_{nl}(G_{eq1}(\theta), K_1)$

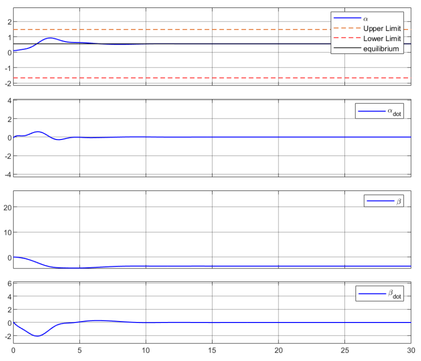


Figure 11: Nonlinear Simulation of $\mathcal{F}_{nl}(G_{eq2}(\theta), K_1)$

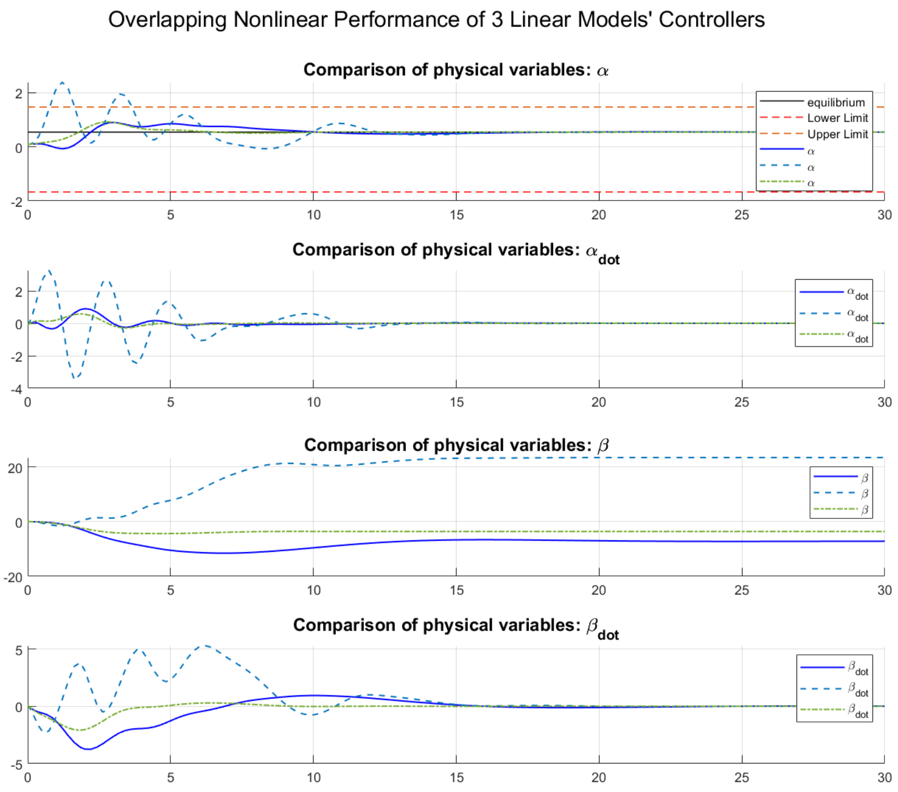


Figure 12: Closed-loop Nonlinear Simulation Overlapping Comparison

5.1.3 Observer Refinement

Although compared to $G_{-\theta}$ and $G_{eq1}(\theta)$, the controller design using $G_l = G_{eq2}(\theta)$ is much better, there are still some small problems. If we look at the observer behavior in the nonlinear system, which is shown in **Figure 14**, there are noticeable steady-state errors between estimated states and real states. Though it further demonstrates the effectiveness of the linear controller, we still want to investigate whether we can make the error smaller.

One possible way is that we can place the poles of the observer further to the left of the left plane. This idea is taken from the simple assumption that placing the poles further to the left increases the feedback gain of the observer, and that in linear systems with steady-state errors, increasing the feedback gain tends to reduce the steady-state error. We therefore hope that this approach will also work in nonlinear systems.

To achieve that, we can use another theorem learned from the course Assignment 3, given $a, b > 0$, by rewriting **LMI 27** as:

$$\begin{gathered} W \succ 0, V \in \mathbb{R}^{p_y \times n} \\ A^T W + WA + C_2^T V + V^T C_2 + 2aW \prec 0 \\ A^T W + WA + C_2^T V + V^T C_2 + 2bW \succ 0 \end{gathered} \tag{30}$$

we can place the poles of the observer in the region $\mathcal{D} = \{x + jy, -b \leq x \leq -a\}$.

Heence, I set $a = 2$ and do not set b, which indicates $\mathcal{D} = \{x + jy, x \leq -2\}$,and solve the new LMI to get the new L'. We can see that the eigenvalues of the observer ($A^{eq}(0.1) + LC_2 \to A^{eq}(0.1) + L'C_2$) change from $\lambda = \begin{bmatrix} -0.4529 + 1.1617i \\ -0.4529 - 1.1617i \\ -0.4691 + 0.0000i \\ -0.5000 + 0.0000i \end{bmatrix}$ to $\lambda' = \begin{bmatrix} -4.6748 + 3.0500i \\ -4.6748 - 3.0500i \\ -2.5197 + 0.0000i \\ -2.5000 + 0.0000i \end{bmatrix}$, which meets the theorem. If we do the linear simulation again with the refined output feedback controller K_1', which is shown in **Figure 13**. We can see that, compared to **Figure 8**, not only does the observer track the state variables faster, but also the whole linear system has a faster response.

In the nonlinear simulation, which is shown in **Figure 15**, we can also see that the estimation error has been reduced a lot, and the system does respond a little bit faster. However, we get a larger β deviation from the zero point, which is the pay off under the nonlinearity.

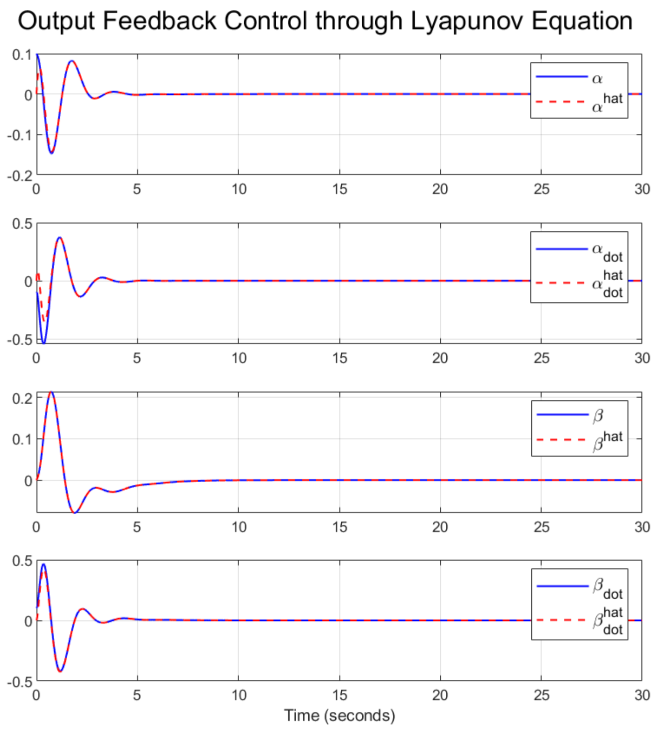


Figure 13: Refined Linear Simulation for $\mathcal{F}_l(G_l(\theta), K_1')$

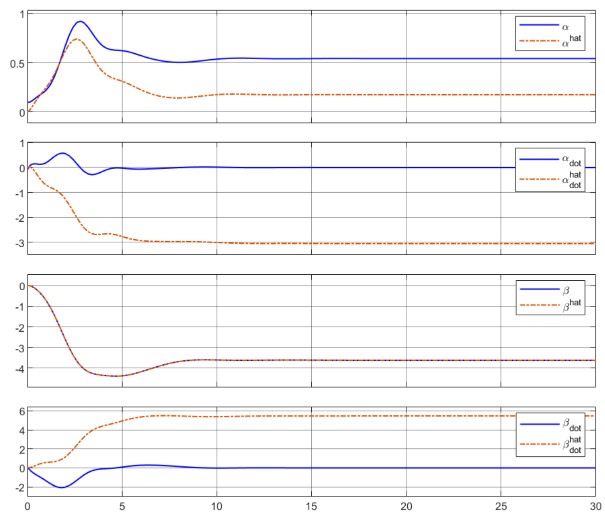


Figure 14: Orignal Estimated/Real States Comparison in Nonlinear Simulation for $\mathcal{F}_{nl}(G_l(\theta), K_1)$

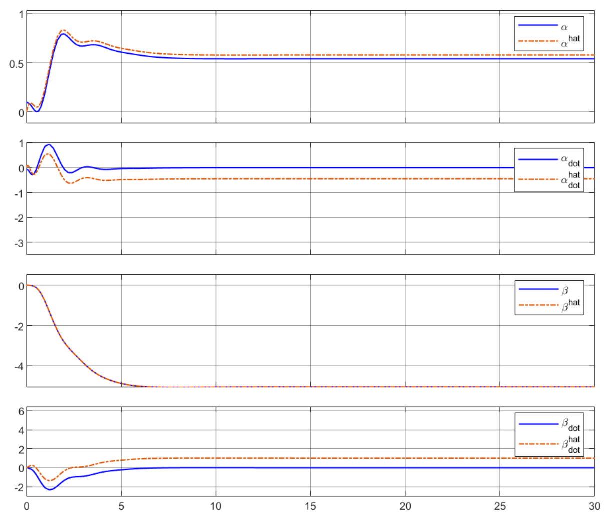


Figure 15: Refined Estimated/Real States Comparison in Nonlinear Simulation for $\mathcal{F}_{nl}(G_l(\theta), K'_1)$

5.1.4 Behavior under Disturbance

At the end of this sub-section, we want to test the behavior under disturbance of the closed-loop system under this simple controller. We introduce 2 disturbances, a Gaussian noise $w = \mathcal{N}(0, 1)$ and a Sine wave $w = \sin(t)$.

Applying these 2 disturbances to the closed-loop linear system $\mathcal{F}_l(G_l(\theta), K'_1)$ and nonlinear system $\mathcal{F}_{nl}(G_l(\theta), K'_1)$, we can get the results shown in **Figure 16, 17, 18, 19**. We can see that using this simple controller, the closed-loop system cannot reject the disturbance at all, which is what we are going to focus on in the next few sections.

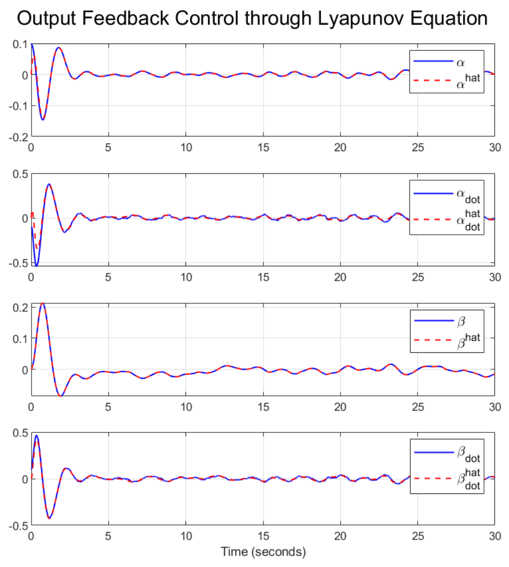


Figure 16: Linear Simulation of $\mathcal{F}_l(G_l(\theta), K'_1)$ under Gaussian Noise Disturbance

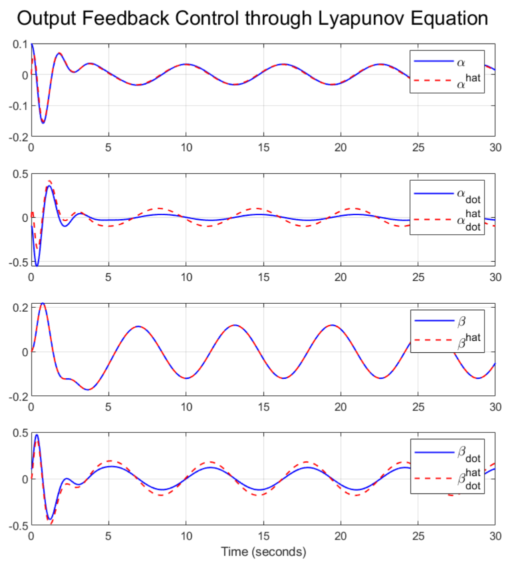


Figure 17: Linear Simulation of $\mathcal{F}_l(G_l(\theta), K'_1)$ under Sine Wave Disturbance

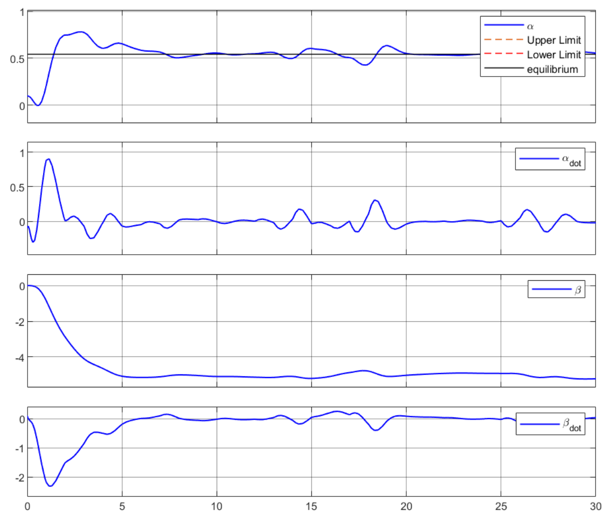


Figure 18: Nonlinear Simulation of $\mathcal{F}_{nl}(G_l(\theta), K_1')$ under Gaussian Noise Disturbance

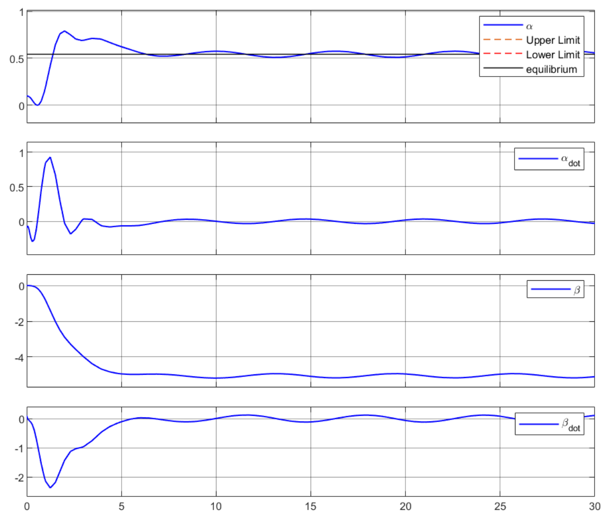


Figure 19: Nonlinear Simulation of $\mathcal{F}_{nl}(G_l(\theta), K_1')$ under Sine Wave Disturbance

5.2 State Feedback Controller using $\mathcal{H}_2$ Optimal Control

5.2.1 LMI Formulation

To suppress the influence of the disturbance, we need to come up with some more advanced controllers. First, we look at the Gaussian noise disturbance. As we know in the course, for a linear system, its $\mathcal{H}_2$ norm is interpreted as the "average energy" of the output z when w is white noise:

$$\|G\|_{\mathcal{H}_2} = \mathbb{E}\int_0^\infty \|z(t)\|_2^2 = \mathbb{E}\|z\|_{L_2[0,\infty)} \tag{31}$$

or the average energy of the output response over all frequencies.

Hence, we would like to apply $\mathcal{H}_2$ optimal control to minimize the effect produced by the Gaussian noise disturbance.

For the $\mathcal{H}_2$ optimal control, we assume a full-information control problem, which denotes state feedback and $C_2 = I$ in this sub-section, and the standard form of the state feedback controller is:

$$K_2 = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & F_{\mathcal{H}_2} \end{array}\right] \tag{32}$$

where $F_{\mathcal{H}_2}$ is the state feedback gain.

The theorem tells us that there exists a state feedback control law $u = F_{\mathcal{H}_2}x$ that satisfies $\|\mathcal{F}_l(G, K_2)\|_{\mathcal{H}_2} = (\textbf{trace}(W))^{\frac{1}{2}} \leq \gamma$, which can be obtained through solving the following convex optimization problem with LMI constraints:

$$\begin{gathered} \min_{\gamma,P,Z,W} \gamma \\ P \succ 0, Z \in \mathbb{R}^{m\times n}, W \in \mathbb{R}^{p_z \times p_z} \\ \begin{bmatrix} A & B_2 \end{bmatrix}\begin{bmatrix} P \\ Z \end{bmatrix} + \begin{bmatrix} P & Z^T \end{bmatrix}\begin{bmatrix} A^T \\ B_2^T \end{bmatrix} + B_1B_1^T \prec 0 \\ \begin{bmatrix} W & (C_1P + D_{12}^TZ) \\ C_1P + D_{12}^TZ & P \end{bmatrix} \succ 0 \\ \textbf{trace}(W) \leq \gamma \end{gathered} \tag{33}$$

where $F_{\mathcal{H}_2} = ZP^{-1}$ and $\gamma = 7.7 \times 10^{-5} \Rightarrow \|\mathcal{F}_l(G, K_2)\|_{\mathcal{H}_2} \leq 0.0088$. So theoretically, we can suppress the Gaussian noise disturbance by over 100 times.

5.2.2 Simulation Results

By implementing the design process in **MATLAB Code 6**, we can simulate the closed-loop system $\mathcal{F}_l(G_l(\theta = 0.1), K_2)$ with no disturbance input and a Gaussian noise disturbance $w = \mathcal{N}(0, 1)$, which is shown in **Figure 20, 21**. We can see that the trajectories of the state variables in the 2 plots are nearly identical, and all states go to zero without hitting the limitation. The same thing happen in the nonlinear simulation, whose results are shown in **Figure 22, 23**, indicating that the $\mathcal{H}_2$ state feedback controller manages to reject the Gaussian disturbance for linearized model and original nonlinear model.

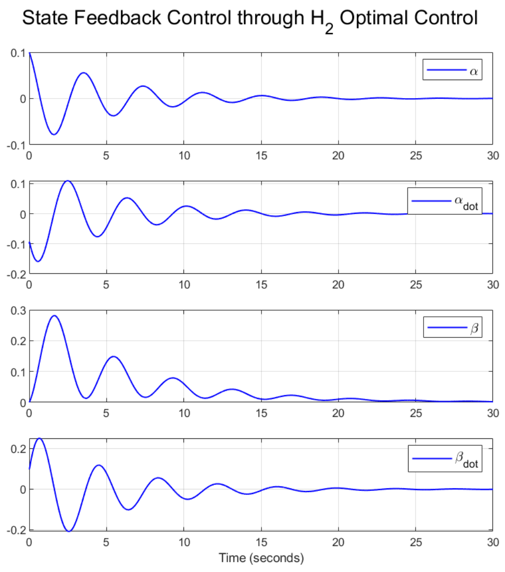


Figure 20: Linear Simulation of $\mathcal{F}_l(G_l(\theta), K_2)$ under no Disturbance

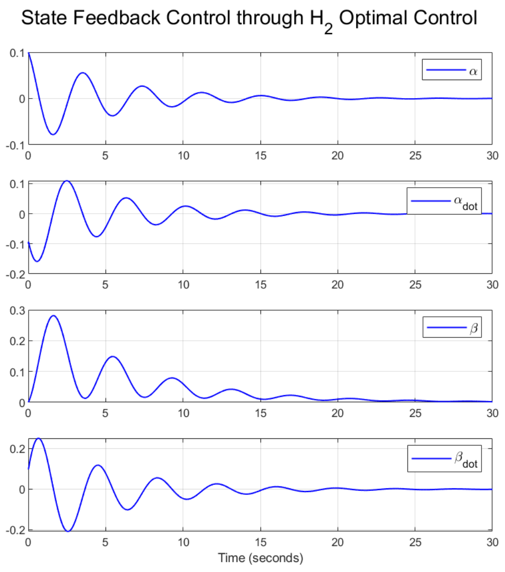


Figure 21: Linear Simulation of $\mathcal{F}_l(G_l(\theta), K_2)$ under Gaussian Disturbance

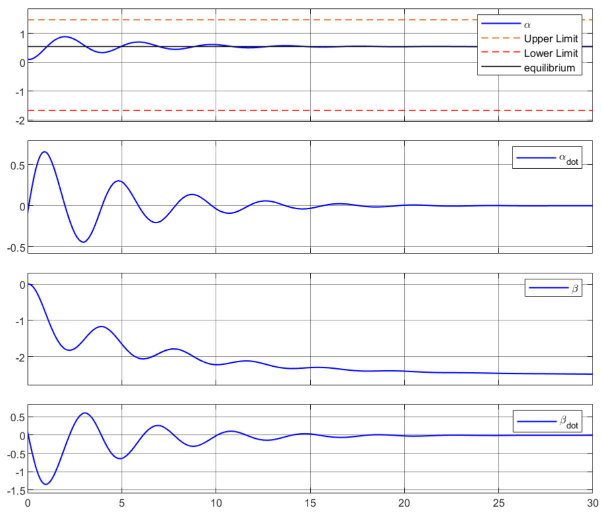


Figure 22: Nonlinear Simulation of $\mathcal{F}_{nl}(G_l(\theta), K_2)$ under no Disturbance

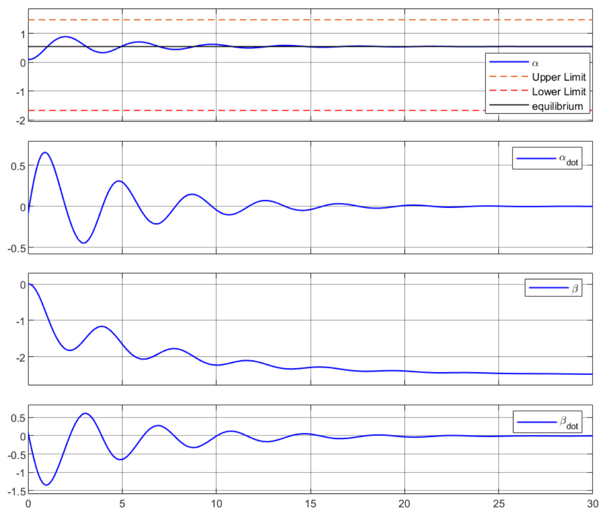


Figure 23: Nonlinear Simulation of $\mathcal{F}_{nl}(G_l(\theta), K_2)$ under Gaussian Disturbance

5.2.3 Controller Refinement

If we compare the results of $\mathcal{F}_{nl}(G_l(\theta), K_2)$ under $\mathcal{H}_2$ controller to $\mathcal{F}_{nl}(G_l(\theta), K_1)$ under Lyapunov controller, we will find that $\mathcal{F}_{nl}(G_l(\theta), K_2)$ has a longer response time for both linear and nonlinear systems. If we check the eigenvalues of the closed-loop system $(A^{eq}(0.1) + B^{eq}(0.1)F_{\mathcal{H}_2})$, we will find that $\lambda = \begin{bmatrix} -1.7705 \times 10^5 \\ -0.1907 - 1.6353i \\ -0.1907 + 1.6353i \\ -0.1448 \end{bmatrix}$, in which 3 of them might be too close to the y-axis. To improve the performance of the controller, we can place the poles further to the left of the left plane.

Hence, we use the theorem learned from the course Assignment 3 again, given $a = 0.5$, by rewriting the LMI constraint in **Problem 33** as:

$$\begin{gathered}
\min_{\gamma,P,Z,W} \gamma \\
P \succ 0, Z \in \mathbb{R}^{m\times n}, W \in \mathbb{R}^{p_z \times p_z} \\
\begin{bmatrix} A & B_2 \end{bmatrix} \begin{bmatrix} P \\ Z \end{bmatrix} + \begin{bmatrix} P & Z^T \end{bmatrix} \begin{bmatrix} A^T \\ B_2^T \end{bmatrix} + B_1 B_1^T + \mathbf{2aP} \prec 0 \\
\begin{bmatrix} W & (C_1 P + D_{12}^T Z) \\ C_1 P + D_{12}^T Z & P \end{bmatrix} \succ 0 \\
\mathbf{trace}(W) \leq \gamma
\end{gathered} \tag{34}$$

where refined feedback control gain $F'_{\mathcal{H}_2} = ZP^{-1}$. We can ensure all poles of the closed-loop system are within the region $\mathcal{D} = \{x + jy, x \leq -0.5\}$, which can be also verified by looking at the eigenvalues of the closed-loop system $(A^{eq}(0.1) + B^{eq}(0.1)F'_{\mathcal{H}_2})$ that $\lambda = \begin{bmatrix} -1.1856 \times 10^5 \\ -0.6248 - 1.6638i \\ -0.6248 + 1.6638i \\ -0.5917 \end{bmatrix}$.

Still, we don't want this refinement to weaken the disturbance suppression a lot. The solution to **Optimization Problem 34** gives us $\gamma = 1.7243 \times 10^{-4} \Rightarrow \|\mathcal{F}_l(G, K_2)\|_{\mathcal{H}_2} \leq 0.0131$, which means we are still able to suppress the Gaussian noise disturbance by over 75 times.

Shwon in **Figure 24, 25**, the closed-loop simulation of $\mathcal{F}_l(G_l(\theta), K_2')$ and $\mathcal{F}_{nl}(G_l(\theta), K_2')$ show that not only we maintain to reject the Gaussian noise disturbance, but also we successfully achiever much faster system response.

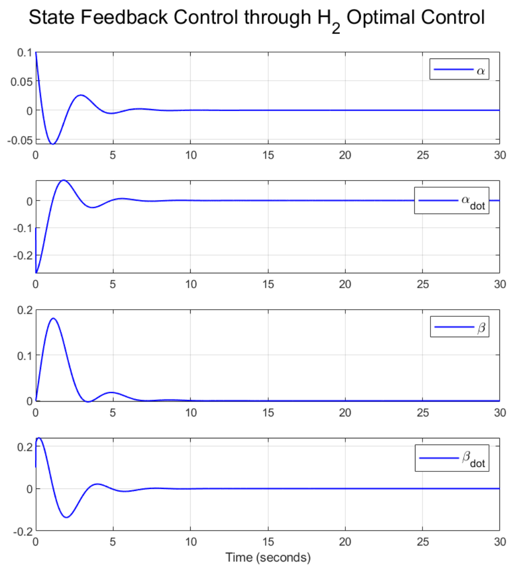


Figure 24: Linear Simulation of Refined $\mathcal{F}_l(G_l(\theta), K_2')$ under Gaussian Disturbance

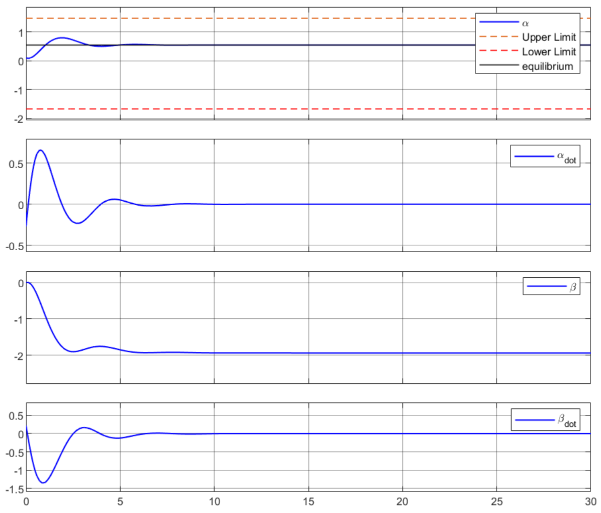


Figure 25: Nonlinear Simulation of Refined $\mathcal{F}_{nl}(G_l(\theta), K_2')$ under Gaussian Disturbance

5.3 State Feedback Controller using $\mathcal{H}_\infty$ Optimal Control

5.3.1 LMI Formulation

After dealing with the Gaussian noise disturbance, we come to the Sine wave disturbance, which leads us to the $\mathcal{H}_\infty$ Optimal Control since it minimizes the maximum singular value of the transfer matrix over all frequencies $\|\hat{G}\|_{\mathcal{H}_\infty} := \sup_{\omega\in\mathbb{R}} \sigma_1[\hat{F}(j\omega)]$, the $\mathcal{H}_\infty$ norm of the system. In this sub-section, we still assume state feedback ($C_2 = I$). But, we will get back to the output feedback in the next sub-section.

The theorem tells us that $\|\hat{G}\|_{\mathcal{H}_\infty}$ is numerically equal to $\|G\|_{\mathcal{H}_\infty} := \sup_{u\neq 0, u\in L_2[0,\infty]} \frac{\|Gu|\_L_2[0,\infty]}{\|u\|_{L_2[0,\infty]}}$, which is the induced norm of the system and the "peak energy" of the output. Hence, we can use the "dilated" version of the KYP Lemma to minimize $\mathcal{H}_\infty$ norm of the closed-loop system by formulating the following convex optimization problem with LMI constraints:

$$\min_{\gamma,Y,Z} \gamma$$
$$Y \succ 0, Z \in \mathbb{R}^{m\times n}$$
$$\begin{bmatrix} YA^T + AY + Z^TB_2^T + B_2Z & B_1 & YC_1^T + Z^TD_{12}^T \\ B_1^T & -\gamma I & D_{12}^T \\ C_1Y + D_{12}Z & D_{11} & -\gamma I \end{bmatrix} \prec 0 \tag{35}$$

The KYP Lemma tells us that under the feedback control law $u = F_{\mathcal{H}_\infty}x$, hence the state feedback controller:

$$K_3 = \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & F_{\mathcal{H}_\infty} \end{array}\right] \tag{36}$$

the closed-loop system $\mathcal{F}_l(G, K_3)$ satisfies $\|\mathcal{F}_l(G, K_3)\|_{\mathcal{H}_\infty} \leq \gamma$, in which $F_{\mathcal{H}_\infty} = ZY^{-1}$ and γ is the solution of **Problem 35**. Here, we get $\gamma = 1.904 \times 10^{-5}$, which means we can theoretically suppress disturbance over all frequencies by over 10 thousand times.

5.3.2 Simulation Results

By implementing the design process in **MATLAB Code 7**, we can simulate the closed-loop system $\mathcal{F}_l(G, K_3)$ with no disturbance input and a Sine wave disturbance $w = \sin(t)$. which is shown in **Figure 26, 27**. We can see that the trajectories of the state variables in the 2 plots are nearly identical, and all states go to zero without hitting the limitation. Also, for the nonlinear simulation, results shown in **Figure 28, 29** indicate that the $\mathcal{H}_\infty$ state feedback controller manages to reject the Sine wave disturbance for both the linearized model and the original nonlinear model. Hence, we achieve the goal of rejecting disturbance.

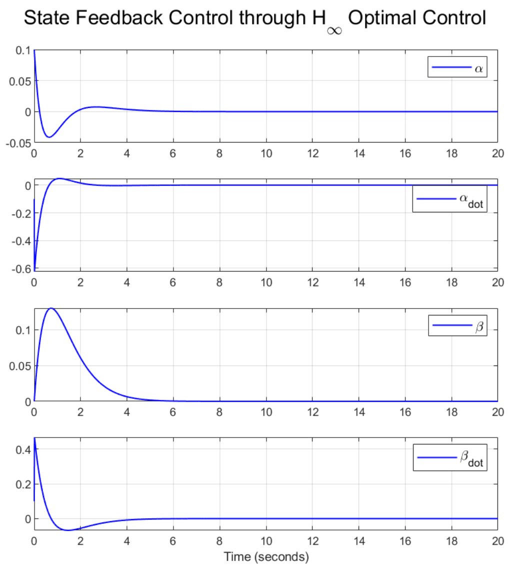


Figure 26: Linear Simulation of $\mathcal{F}_l(G_l(\theta), K_3)$ under no Disturbance

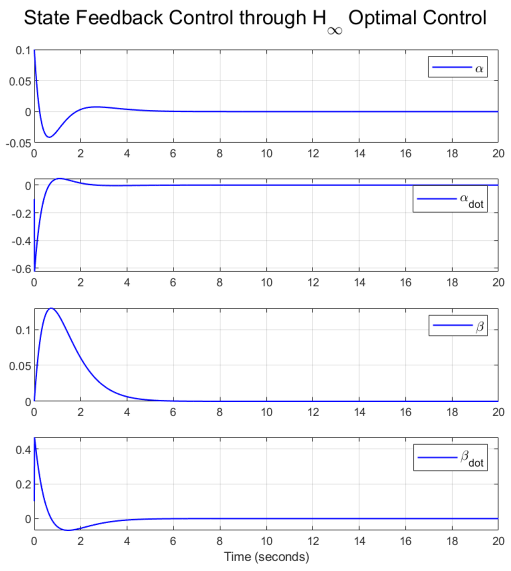


Figure 27: Linear Simulation of $\mathcal{F}_l(G_l(\theta), K_3)$ under Sine Wave Disturbance

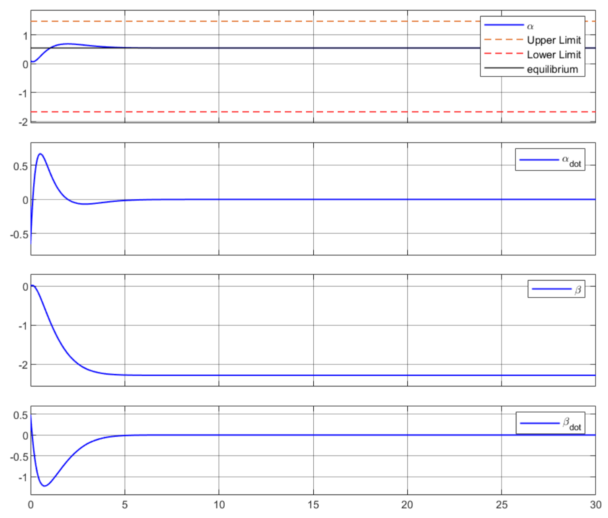


Figure 28: Nonlinear Simulation of $\mathcal{F}_{nl}(G_l(\theta), K_3)$ under no Disturbance

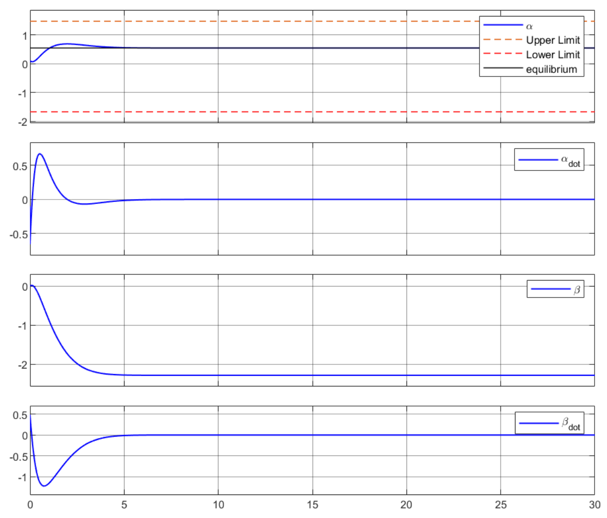


Figure 29: Nonlinear Simulation of $\mathcal{F}_{nl}(G_l(\theta), K_3)$ under Sine Wave Disturbance

5.4 Output Feedback Controller using $\mathcal{H}_\infty$ Optimal Control

5.4.1 LMI Formulation

Finally, we come back to the $\mathcal{H}_\infty$ output feedback control ($C_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$), the **SOTA** control method we learned from the course.

Without going into details, if apply M.I.L to the closed-loop system described in **Equation 25**, do the changing of matrix variables, do the congruent transformation with the Transformation Lemma, and use a congruent transformation to the closed-loop system, finally we can come up the optimal $\mathcal{H}_\infty$ output feedback controller using the following process:

1. Solve the convex optimization problem with LMI constraints:

$$\min_{\gamma, A_n, B_n, C_n, D_n, X_1, Y_1} \gamma$$
$$\begin{bmatrix} AY_1 + Y_1A^T + B_2C_n + C_n^TB_2^T & * & * & * \\ A^T + A_n + (B_2D_nC_2)^T & X_1A + A^TX_1 + B_nC_2 + C_2^TB_n^T & * & * \\ (B_1 + B_2D_nD_{21})^T & (X_1B_1 + B_nD_{21})^T & -\gamma I & * \\ C_1Y_1 + D_{12}C_n & C_1 + D_{12}D_nC_2 & D_{11} + D_{12}D_nD_{21} & -\gamma I \end{bmatrix} \prec 0 \tag{37}$$

2. Constructing $X_2 = 1 - X_1Y_1, Y_2 = I$ so that $X_2Y_2^T = 1 - X_1Y_1$ is satisfied.

3. Use X_2 and Y_2 from step 2, and the solution to the optimization problem to construct:

$$\begin{bmatrix} A_{K2} & B_{K2} \\ C_{K2} & D_{K2} \end{bmatrix} = \begin{bmatrix} X_2 & X_1B_2 \\ 0 & I \end{bmatrix}^{-1} \left(\begin{bmatrix} A_n & B_n \\ C_n & D_n \end{bmatrix} - \begin{bmatrix} X_1AY_1 & 0 \\ 0 & 0 \end{bmatrix} \right) \begin{bmatrix} Y_2^T & 0 \\ C_2Y_1 & I \end{bmatrix}^{-1} \tag{38}$$

4. Use $A_{K2}, B_{K2}, C_{K2}, D_{K2}$ to construct the controller $K_4 = \left[\begin{array}{c|c} A_K & B_K \\ \hline C_K & D_K \end{array}\right]$ where (solving in order):

$$\begin{aligned} D_K &= (I + D_{K2}D_{22})^{-1}D_{K2} \\ B_K &= B_{K2}(I - D_{22}D_K) \\ C_K &= I - D_{22}D_k \\ A_K &= A_{K2} - B_K(I - D_{22}D_K)^{-1}D_{22}C_K \end{aligned} \tag{39}$$

Since we have $D_{22} = 0$, $I - D_{22}D_K$ is ensured to be invertible and the closed-loop system is well-posed.

The theorem tells us that with the controller K_4, the closed-loop system $\mathcal{F}_l(G, K_4)$ satisfies $\|\mathcal{F}_l(G, K_4)\|_{\mathcal{H}_\infty} \leq \gamma$.

When solving **Problem 37** by CVX, I encounter an issue that CVX fails to give an answer. However, if we add an extra constraint:

$$\gamma \geq \epsilon \tag{40}$$

where ϵ is a manually picked small number, CVX might return a solution and $\gamma = \epsilon$ if ϵ is big enough. So, there might be some numerical problem in CVX here.

To get the minimum γ, I do a bisection and grid search over $[10^{-6}, 10^{-2}]$ and get $\gamma = 2.2 \times 10^{-5}$, which means we can theoretically suppress disturbance over all frequencies by nearly 10 thousand times.

5.4.2 Simulation Results

By implementing the design process in **MATLAB Code 8**, we can simulate the closed-loop system $\mathcal{F}_l(G, K_4)$ with no disturbance input and a Sine wave disturbance $w = \sin(t)$. Looking at **Figure 30, 31**, we can see that the system response is almost identical, which means the $\mathcal{H}_\infty$ output feedback controller successfully reject the Sine wave disturbance. Zooming out to the estimated states under the Sine wave disturbance, which is shown in **Figure 32**, we see that the estimated states are oscillating for the disturbance compensation.

Interestingly, the estimated states actually take quite a long time to track the real states ($\sim 80s$, so I have to increase the simulation time from 30s to 80s). However, the controller still managed to stabilize the system way before ($\sim 20s$) the correct estimated states, which might indicate the power of this controller or just a coincidence of the system's properties.

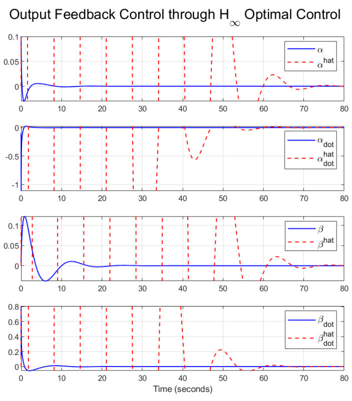


Figure 30: Linear Simulation of $\mathcal{F}_l(G_l(\theta), K_4), \gamma = 2.2 \times 10^{-5}$ under no Disturbance

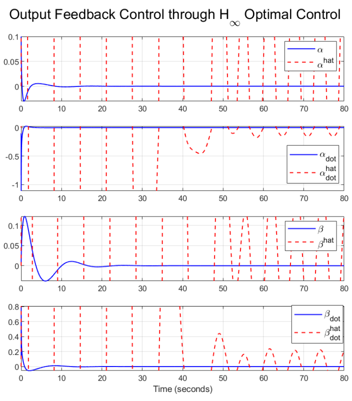


Figure 31: Linear Simulation of $\mathcal{F}_l(G_l(\theta), K_4), \gamma = 2.2 \times 10^{-5}$ under Sine Wave Disturbance

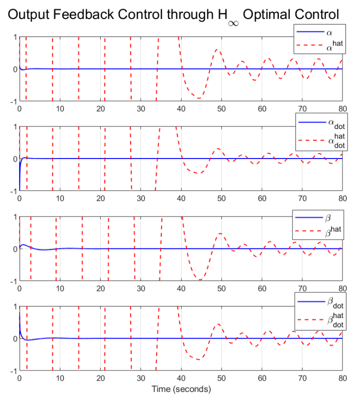


Figure 32: Linear Simulation of $\mathcal{F}_l(G_l(\theta), K_4), \gamma = 2.2 \times 10^{-5}$ under Sine Wave Disturbance (Zoomed Out)

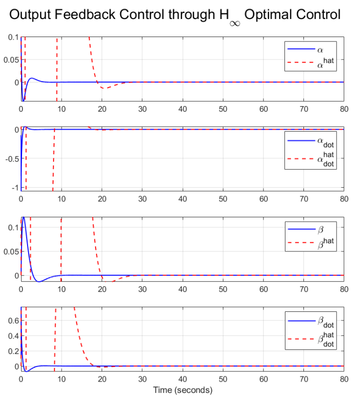


Figure 33: Linear Simulation of $\mathcal{F}_l(G_l(\theta), K_4), \gamma = 2.2 \times 10^{-4}$ under no Disturbance

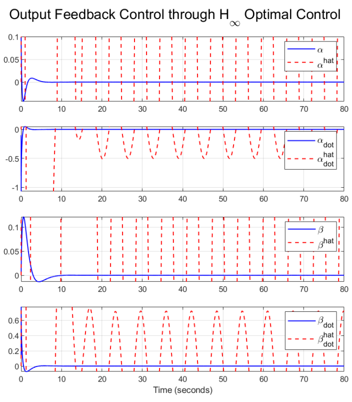


Figure 34: Linear Simulation of $\mathcal{F}_l(G_l(\theta), K_4), \gamma = 2.2 \times 10^{-4}$ under Sine Wave Disturbance

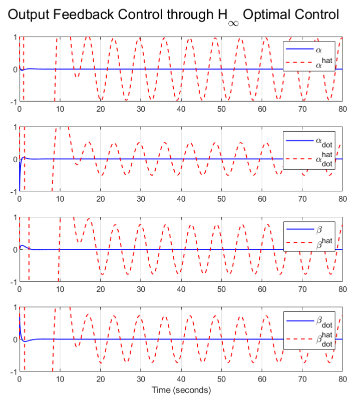


Figure 35: Linear Simulation of $\mathcal{F}_l(G_l(\theta), K_4), \gamma = 2.2 \times 10^{-4}$ under Sine Wave Disturbance (Zoomed Out)

Moreover, if we increase the γ a little, say $2.2 \times 10^{-5} \Rightarrow= 2.2 \times 10^{-4}$, so still a strong suppression factor, not only can we make the estimated states track the real states faster ($\sim 30s$, shown in **Figure 33**)and rejecting Sine wave disturbance (shown in **Figure 34**), but also actually make the system response faster (from $\sim 20s$ to ($\sim 10s$)). Another thing is that the estimated states are now oscillating with bigger magnitudes (shown in **Figure 35**).

For nonlinear simulation, the same thing happens to the system response speed if we increase the γ. Shown in **Figure 36, 37**, we can see that system stabilization time decreases from $\sim 30s$ to $\sim 15s$. Also, some weird jitters in $\dot{\alpha}$ happens in $gamma = 2.2 \times 10^{-5}$(second subplot of **Figure 36**) disappears when $\gamma = 2.2 \times 10^{-4}$.

Strangely, shown in **Figure 38, 39**, the state estimations seem to completely break down in the nonlinear simulation when $\theta \neq 0$, but if we set $\theta = 0$, the state estimations are perfect just like the linear case, which is shown in **Figure 40**. Moreover, the system still manages to stabilize under the controller under bad estimation. I don't quite understand what's happening here, perhaps it's the strong power of the controller, some inherent good properties in the model, or just the mystery of the nonlinearity.

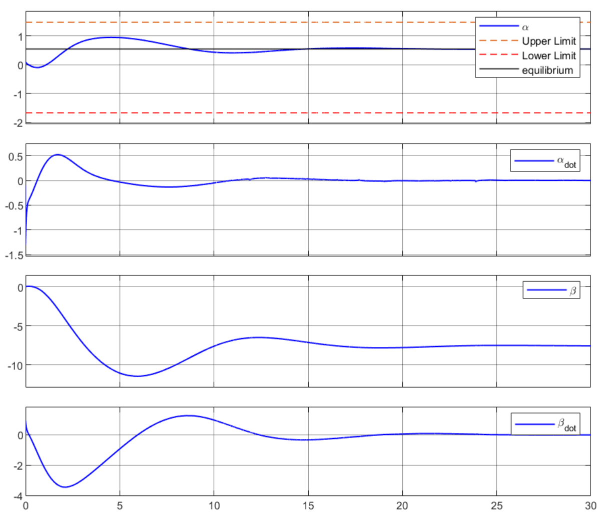


Figure 36: Nonlinear Simulation of $\mathcal{F}_l(G_l(\theta), K_4), \gamma = 2.2 \times 10^{-5}$ under Sine Wave Disturbance

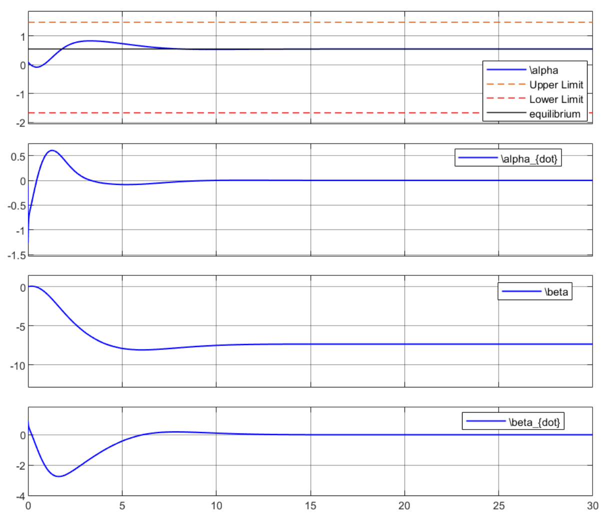
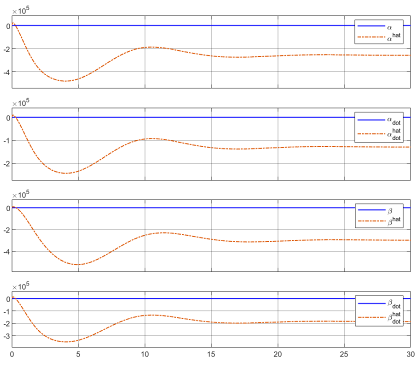
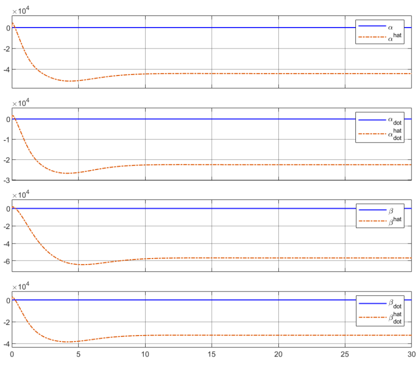


Figure 37: Nonlinear Simulation of $\mathcal{F}_l(G_l(\theta), K_4), \gamma = 2.2 \times 10^{-4}$ under Sine Wave Disturbance





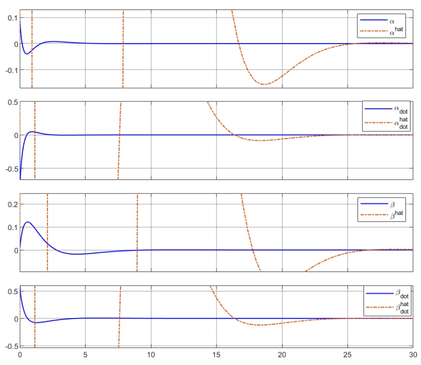


Figure 38: Estimated/Real States Comparison in Nonlinear Simulation of $\mathcal{F}_l(G_l(0.1), K_4), \gamma = 2.2 \times 10^{-5}$ under Sine Wave Disturbance

Figure 39: Estimated/Real States Comparison in Nonlinear Simulation of $\mathcal{F}_l(G_l(0.1), K_4), \gamma = 2.2 \times 10^{-4}$ under Sine Wave Disturbance

Figure 40: Estimated/Real States Comparison in Nonlinear Simulation of $\mathcal{F}_l(G_l(0), K_4), \gamma = 2.2 \times 10^{-4}$ under Sine Wave Disturbance

6 Balancing the Unicycle Robot on the Time-Variant Slope

In the last section, I want to test whether we can stabilize the Unicycle when the slope angle is changing. We set a uniform sampler mentioned in the function descriptions of **Figure 5** to sample the slope angle from -0.1 to 0.1 radius every $10s$, then simulate the system for $100s$ (10 different slope angles). Since we cannot know the actual angle of the slope, we just keep using $\theta = 0.1$ for linearization.

We do the simulation for all 4 types of controllers we mentioned in the report. For the Lyapunov-based controller K_1', we add no noise. For the $\mathcal{H}_2$ state feedback controller K_2', we add a standard Gaussian noise disturbance $w = \mathcal{N}(0, 1)$. For the $\mathcal{H}_\infty$ state feedback controller K_3 and $\mathcal{H}_\infty$ output feedback controller $K_4, \gamma = 2.2 \times 10^{-4}$, we add a Sine wave disturbance $w = \sin(t)$.

Shown in the **Figure 41, 42, 43, 44**, we can see that all controllers successfully make the Unicycle Robot balance on different slope angles without hitting any physical limitations for α, even if we are using a fixed linearized model to design the linear controllers! Among them, I will say the $\mathcal{H}_\infty$ state feedback controller gives the best performance since it reacts fastest with the least oscillation. It is quite reasonable because $\mathcal{H}_\infty$ control is somehow better than the Lyapunov method and $\mathcal{H}_2$ control generally, and state feedback control should work better than output feedback control.

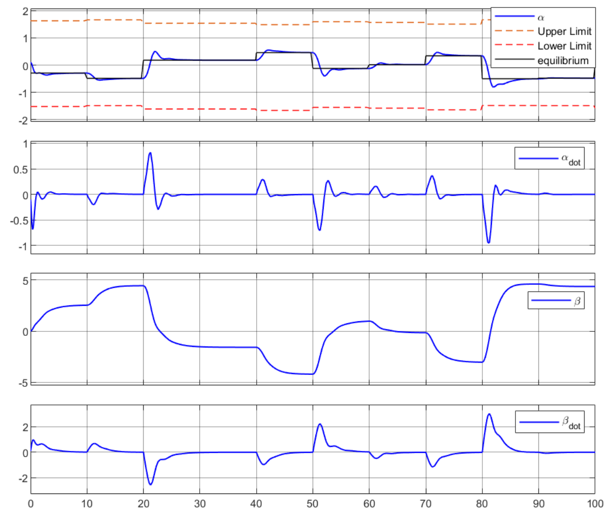


Figure 41: Simulation of $\mathcal{F}_l(G_l(0.1), K_1')$, on Time-Variant Slope (No Disturbance)

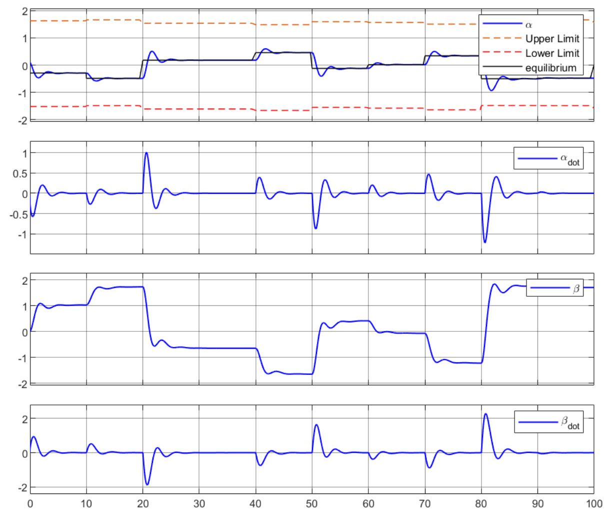


Figure 42: Simulation of $\mathcal{F}_l(G_l(0.1), K_2')$, on Time-Variant Slope (Gaussian Noise Disturbance)

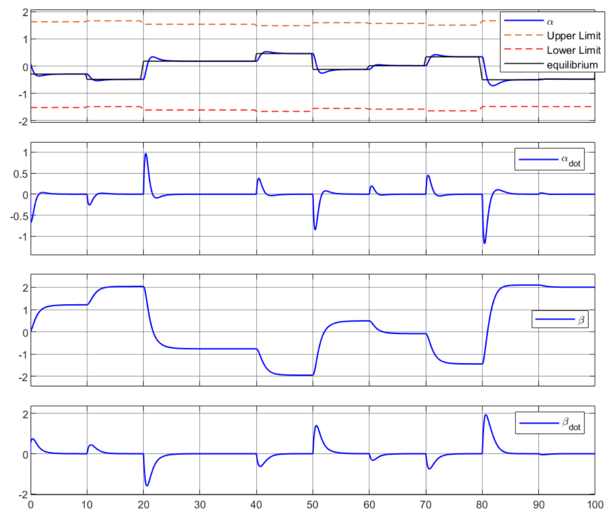


Figure 43: Simulation of $\mathcal{F}_l(G_l(0.1), K_3)$, on Time-Variant Slope (Sine Wave Disturbance)

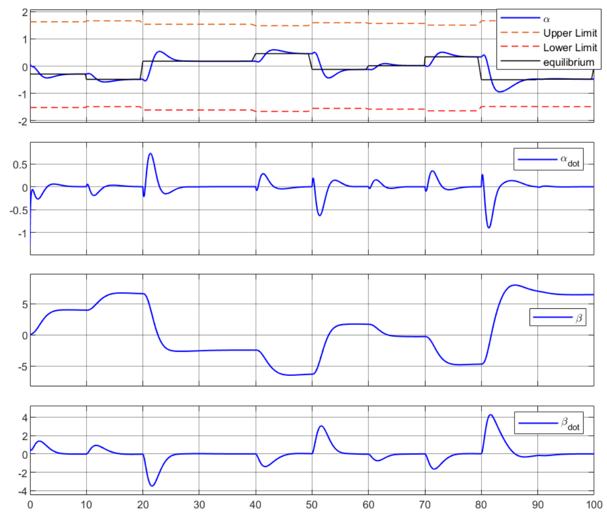


Figure 44: Simulation of $\mathcal{F}_l(G_l(0.1), K_4), \gamma = 2.2 \times 10^{-4}$ on Time-Variant Slope (Sine Wave Disturbance)

7 Conclusion and Potential Future Work

7.1 Conclusion

In this report, we look inside how to balance a Unicycle Robot on a slope. We analyze the nonlinear dynamics of the Unicycle Robot and do several linearizations and simplifications. We successfully designed the controllers for the linearized model using Lyapunov output feedback control, optimal $\mathcal{H}_2$ state feedback control, optimal $\mathcal{H}_\infty$ state feedback control, and optimal $\mathcal{H}_\infty$ output feedback control. We also implemented these linear controllers to the nonlinear system of the Unicycle Robot in Simulink and successfully achieved the balance as well, even under certain disturbances. Finally, we designed a slope over time scenario for testing the final performance of the controllers, which was successfully achieved by all controllers.

7.2 Potential Work

During the process of analyzing, experimenting, and writing the report, I still have many questions and ideas that I may continue to research in the future:

1. According to **Equation 10**, there seems to be a chance to assign the equilibrium by the input voltage, but I cannot figure it out during the experiments, so it is a good angle.
2. In the output feedback control based on the Lyapunov Equation and $\mathcal{H}_\infty$ optimal control, although there are errors (even a very huge one) between the estimated states and the real states in the nonlinear simulation, the controller still manages to stabilize the nonlinear system, which is quite interesting and I would like to do more investigation.
3. I am interested in whether I can implement these controllers in a more realistic environment, like physical simulators (Mujoco, Isaac Lab), or even a real Unicycle Robot.

In closing, I am deeply grateful for this course. At the start of the semester, I questioned the relevance of linear system control to my focus in robotics, given the inherently nonlinear nature of most robotic systems. However, I soon came to appreciate that linear approximations—through local linearization—can effectively capture the essential behavior of many nonlinear systems around their operating points. This insight, along with the opportunity to study the advanced control methods, Linear Matrix Inequalities (LMIs) in depth, has significantly enriched my understanding and broadened my toolkit as a robotics student.

References

[1] Seong I Han, Jang M Lee, Balancing and velocity control of a unicycle robot based on the dynamic model, *IEEE Transactions on Industrial Electronics*, **62**(1):405–413, 2014.

[2] M Anfa'ur Rosyidi, Eko Henfri Binugroho, S Ekti Radin Charel, R Sanggar Dewanto, Dadet Pramadihanto, Speed and balancing control for unicycle robot, in *2016 International Electronics Symposium (IES)*. year, IEEE, 2016.

[3] Sudarshan M Samarasinghe, Manukid Parnichkun, Pitch control of an active omni-wheeled unicycle using lqr, in *2019 First International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics (ICA-SYMP)*. year, IEEE, 2019.

[4] Lele Wei, Wei Yao, Design and implement of lqr controller for a self-balancing unicycle robot, in *2015 ieee international conference on information and automation*. year, IEEE, 2015.

A MATLAB Realization

Runnable codes and Simulink set up can also be found on Github:
https://github.com/Anthony-ZC/Columbia-EE6602-Modern-Control-Theory-Project

```
clear;
g = 9.8;
M_p = 15.63;
M_w = 31.28;
I_p = 3.55;
```

```
I_w = 1.40;
R_w = 0.31;
L = 0.155;
n = 19.56;
k_T = 0.05;
R = 1.2;
k_e = 0.0534;

equilibrium = 0;
P1 = M_w*R_w^2 + M_p*R_w^2 + I_w;
P2 = M_p*R_w*L;
P3 = M_p*L^2 + I_p;
P4 = M_p*L*g;
P5 = (n*k_T)/R;
P6 = (M_w + M_p)*R_w;


A = [0 1 0 0;
    (P1*P4)/(P1*P3-P2^2) 0 0 ((P1+P2)*P5*n*k_e)/(P1*P3-P2^2);
    0 0 0 1;
    -(P2*P4)/(P1*P3-P2^2) 0 0 -((P2+P3)*P5*n*k_e)/(P1*P3-P2^2)];
eig(A)
B2 = [0;
    -(P1+P2)*P5/(P1*P3-P2^2);
    0;
    (P2+P3)*P5/(P1*P3-P2^2)];

B1 = B2;

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

n = size(A,1);
m = size(B2,2);
p_z = size(C1,1);

D11 = zeros(p_z,1);
D12 = zeros(p_z,1);
```

Listing 1: Matlab Code for Defining $G_{-\theta}$

```
clear;
g = 9.8;
M_p = 15.63;
M_w = 31.28;
I_p = 3.55;
I_w = 1.40;
R_w = 0.31;
L = 0.155;
n = 19.56;
k_T = 0.05;
R = 1.2;
k_e = 0.0534;

theta = 0.1;
equilibrium = asin((M_w+M_p)*R_w/(M_p*L)*sin(theta))-theta;
```

```
P1 = M_w*R_w^2 + M_p*R_w^2 + I_w;
P2 = M_p*R_w*L*cos(equilibrium+theta);
P3 = M_p*L^2 + I_p;
P4 = M_p*L*g*cos(equilibrium+theta);
P5 = (n*k_T)/R;
P6 = (M_w + M_p)*R_w;

A = [0 1 0 0;
    (P1*P4)/(P1*P3-P2^2) 0 0 ((P1+P2)*P5*n*k_e)/(P1*P3-P2^2);
    0 0 0 1;
    -(P2*P4)/(P1*P3-P2^2) 0 0 -((P2+P3)*P5*n*k_e)/(P1*P3-P2^2)];
eig(A)
B2 = [0;
    -(P1+P2)*P5/(P1*P3-P2^2);
    0;
    (P2+P3)*P5/(P1*P3-P2^2)];

B1=B2;

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

n = size(A,1);
m = size(B2,2);
p_z = size(C1,1);

D11 = zeros(p_z,1);
D12 = zeros(p_z,1);
```

Listing 2: Matlab Code for Defining $G_{eq1/2}(\theta)$, here $\theta = 0.1$

```
sys_ol= ss(A,B2,C1,0);

x0 = [0.1 ; -0.1; 0; 0.1];
t = 0:0.01:30;
w = [zeros(1,length(t))];
[y, t, x] = lsim(sys_ol, w, t,x0);

figure('Position', [100, 100, 600, 600]);
sgtitle('Open-loop Simulation')
state_variables = ["\alpha","\alpha_{dot}","\beta","\beta_{dot}"];
for i=1:4
    subplot(4,1,i);
    plot(t, x(:,i), 'b','LineWidth',1);
    legend( sprintf('%s', state_variables(i)),'FontSize',10);
    grid on;
end
xlabel('Time (seconds)')
```

Listing 3: Linear Model Open-loop Simulation in MATLAB

```
function [alpha_ddot, beta_ddot] = fcn(alpha, alpha_dot, beta_dot,u,theta)
    g = 9.8;
    M_p = 15.63;
    M_w = 31.28;
    I_p = 3.55;
    I_w = 1.40;
    R_w = 0.31;
    L = 0.155;
    n = 19.56;
    k_T = 0.05;
    R = 1.2;
    k_e = 0.0534;

    tau = -(n^2 * k_T *k_e / R )* beta_dot + (n * k_T / R)*u;

    a = M_p* L^2 + I_p;
    b = M_p * R_w * L * cos(alpha+theta);
    d = M_w * R_w^2 + M_p * R_w^2 + I_w;

    c = -tau + M_p * L * g * sin(alpha + theta);
    e = tau - (M_p + M_w) * R_w * g * sin(theta) + alpha_dot^2 * M_p * R_w * L * sin(alpha
     ↪  + theta);

    alpha_ddot = (c*d - b*e)/(a*d - b*b);
    beta_ddot = (a*e - b*c)/(a*d - b*b);
end
```

Listing 4: Nonlinear Dynamics for Unicycle Robot in Simulink

```
clear;
close;
cvx_clear;
run report_model_equilibrium.m

C2 = [1 0 0 0;
    0 0 1 0;
    0 0 0 1];
p_y = size(C2,1);
D21 = zeros(p_y,1);
D22 = zeros(p_y,1);

cvx_begin sdp
    variable W(n,n) symmetric
    variable V(p_y,n)

    subject to
        W >= 1e-6*eye(n);
        A'*W + W*A +C2'*V + V'*C2 <= -1e-6*eye(n);
cvx_end
L_gain = W\V';

cvx_begin sdp
    variable P(n,n) symmetric
    variable Z(m,n)

    subject to
        P >= 1e-6*eye(n);
        A*P + P*A' + B2*Z + Z'*B2' <= -1e-6*eye(n);
cvx_end
K_gain = Z/P;

AK = A+L_gain*C2+B2*K_gain;
BK = -L_gain;
CK = K_gain;
```

```
DK = zeros(m,p_y);

Q = inv(eye(m)-DK*D22);
Assistant_matrix = [eye(m) -DK;-D22 eye(p_y)]\[zeros(m,n) CK;C2 zeros(p_y,n)];

A_cl = blkdiag(A,AK) +blkdiag(B2,BK)*Assistant_matrix;
B_cl = [B1+B2*DK*Q*D21; BK*Q*D21];
C_cl = [C1 zeros(p_z,n)] + [D12 zeros(p_z,p_y)]*Assistant_matrix;
D_cl = D11+D12*DK*Q*D21;

sys_cl= ss(A_cl,B_cl,C_cl,D_cl);

x0 = [0.1 ; -0.1; 0; 0.1; 0; 0; 0; 0];
t = 0:0.01:30;
w = [zeros(1,length(t))];
% w = randn(1, length(t));
% w= sin(t);
[y, t, x] = lsim(sys_cl, w, t,x0);

figure('Position', [100, 100, 600, 600]);
sgtitle('Output Feedback Control through Lyapunov Equation')
state_variables = ["\alpha","\alpha_{dot}","\beta","\beta_{dot}"];
for i=1:4
    subplot(4,1,i);
    plot(t, x(:,i), 'b', t, x(:,i+n), 'r--','LineWidth',1);
    legend( sprintf('%s', state_variables(i)), sprintf('%s^{hat}',
    ↪  state_variables(i)),'FontSize',10);
    grid on;
end
xlabel('Time (seconds)')
```

Listing 5: Designing Output Feedback Controller using Lyapunov Equation

```
clear;
close;
cvx_clear;
run report_model_equilibrium.m

C2 = [1 0 0 0;
    0 1 0 0;
    0 0 1 0;
    0 0 0 1];
p_y = size(C2,1);
D21 = zeros(p_y,1);
D22 = zeros(p_y,1);

cvx_begin sdp
    variable P(n,n) symmetric
    variable Z(m,n)
    variable G(p_z,p_z) symmetric

    minimize(trace(G))
    subject to
        P >= 1e-6*eye(n);
        A*P + B2*Z + P*A' +Z'*B2' + B1*B1'<= -1e-6*eye(n);
        [G (C1*P+D12*Z)';C1*P+D12*Z  P] >= 1e-6*eye(p_z+n);
cvx_end
K_gain = Z/P;
trace(G)
```

```
AK = zeros(n,n);
BK = zeros(n,p_y);
CK = zeros(m,n);
DK = K_gain;

Q = inv(eye(m)-DK*D22);
Assistant_matrix = [eye(m) -DK;-D22 eye(p_y)]\[zeros(m,n) CK;C2 zeros(p_y,n)];

A_cl = blkdiag(A,AK) +blkdiag(B2,BK)*Assistant_matrix;
B_cl = [B1+B2*DK*Q*D21; BK*Q*D21];
C_cl = [C1 zeros(p_z,n)] + [D12 zeros(p_z,p_y)]*Assistant_matrix;
D_cl = D11+D12*DK*Q*D21;

sys_cl= ss(A_cl,B_cl,C_cl,D_cl);
sigma(sys_cl)

x0 = [0.1 ; -0.1; 0; 0.1; 0; 0; 0; 0];
t = 0:0.01:30;
w = [zeros(1,length(t))];
w = randn(1, length(t));
[y, t, x] = lsim(sys_cl, w, t,x0);

figure('Position', [100, 100, 600, 600]);
sgtitle('State Feedback Control through H_2 Optimal Control')
state_variables = ["\alpha","\alpha_{dot}","\beta","\beta_{dot}"];
for i=1:4
    subplot(4,1,i);
    plot(t, x(:,i), 'b','LineWidth',1);
    legend( sprintf('%s', state_variables(i)),'FontSize',10);
    grid on;
end
xlabel('Time (seconds)')
```

Listing 6: Designing State Feedback Controller using $\mathcal{H}_2$ Optimal Control

```
clear;
close;
cvx_clear;
run report_model_equilibrium.m

C2 = [1 0 0 0;
    0 1 0 0;
    0 0 1 0;
    0 0 0 1];
p_y = size(C2,1);
D21 = zeros(p_y,1);
D22 = zeros(p_y,1);

cvx_begin sdp
    variable Y(n,n) symmetric
    variable Z(m,n)
    variable gamm

    minimize(gamm)

    subject to
        Y >= 1e-6*eye(n);
        [Y*A'+A*Y+Z'*B2'+B2*Z B1 Y*C1'+Z'*D12';
```

```
            B1' -gamm*eye(m) D11';
            C1*Y+D12*Z D11 -gamm*eye(p_z)] <= -1e-6*eye(n+m+p_z);
cvx_end
K_gain=Z/Y;
disp(gamm)

AK = zeros(n,n);
BK = zeros(n,p_y);
CK = zeros(m,n);
DK = K_gain;

Q = inv(eye(m)-DK*D22);
Assistant_matrix = [eye(m) -DK;-D22 eye(p_y)]\[zeros(m,n) CK;C2 zeros(p_y,n)];

A_cl = blkdiag(A,AK) +blkdiag(B2,BK)*Assistant_matrix;
B_cl = [B1+B2*DK*Q*D21; BK*Q*D21];
C_cl = [C1 zeros(p_z,n)] + [D12 zeros(p_z,p_y)]*Assistant_matrix;
D_cl = D11+D12*DK*Q*D21;

sys_cl= ss(A_cl,B_cl,C_cl,D_cl);

x0 = [0.1 ; -0.1; 0; 0.1; 0; 0; 0; 0];
t = 0:0.01:20;
w = [zeros(1,length(t))];
% w = sin(t);
[y, t, x] = lsim(sys_cl, w, t,x0);

figure('Position', [100, 100, 600, 600]);
sgtitle('State Feedback Control through H_{\infty} Optimal Control')
state_variables = ["\alpha","\alpha_{dot}","\beta","\beta_{dot}"];
for i=1:4
    subplot(4,1,i);
    plot(t, x(:,i), 'b','LineWidth',1);
    legend( sprintf('%s', state_variables(i)),'FontSize',10);
    grid on;
end
```

Listing 7: Designing State Feedback Controller using $\mathcal{H}_\infty$ Optimal Control

```
clear;
close;
cvx_clear;
run report_model_equilibrium.m

C2 = [1 0 0 0;
    0 0 1 0;
    0 0 0 1];
p_y = size(C2,1);
D21 = zeros(p_y,1);
D22 = zeros(p_y,1);

cvx_begin sdp
    variable gamm
    variable Y1(n,n) symmetric
    variable X1(n,n) symmetric
    variable An(n,n)
    variable Bn(n,p_y)
    variable Cn(m,n)
    variable Dn(m,p_y)
    minimize(gamm)

    M11 = A*Y1+Y1*A'+B2*Cn+Cn'*B2';
    M21 = A'+An+(B2*Dn*C2)';
    M22 = X1*A+A'*X1+Bn*C2+C2'*Bn';
    M31 = (B1+B2*Dn*D21)';
    M32 = (X1*B1+Bn*D21)';
    M41 = C1*Y1+D12*Cn;
    M42 = C1+D12*Dn*C2;
    M43 = D11+D12*Dn*D21;

    gamm>=2.2*1e-5;
    [M11 M21' M31' M41';
     M21 M22 M32' M42';
     M31 M32 -gamm*eye(m) M43';
     M41 M42 M43 -gamm*eye(p_z)] <= -1e-6*eye(n+n+m+p_z);

    [Y1 eye(n);
     eye(n) X1] >= 1e-6*eye(n+n);

cvx_end
disp(gamm)

Y2 = eye(n);
X2 = eye(n)-X1*Y1;

M_K2 = [X2 X1*B2; zeros(m,n) eye(m)]\([An-X1*A*Y1 Bn;Cn Dn])/[Y2' zeros(n,p_y);C2*Y1
↪  eye(p_y)];

AK2 = M_K2(1:n,1:n);
BK2 = M_K2(1:n,n+1:end);
CK2 = M_K2(n+1:end,1:n);
DK2 = M_K2(n+1:end,n+1:end);

DK = (eye(m)+DK2*D22)\DK2;
BK = BK2*(eye(p_y)-D22*DK);
CK = (eye(m)-DK*D22)*CK2;
AK = AK2 - BK/(eye(p_y)-D22*DK)*D22*CK;

Q = inv(eye(m)-DK*D22);
Assistant_matrix = [eye(m) -DK;-D22 eye(p_y)]\[zeros(m,n) CK;C2 zeros(p_y,n)];

A_cl = blkdiag(A,AK) +blkdiag(B2,BK)*Assistant_matrix;
B_cl = [B1+B2*DK*Q*D21; BK*Q*D21];
C_cl = [C1 zeros(p_z,n)] + [D12 zeros(p_z,p_y)]*Assistant_matrix;
D_cl = D11+D12*DK*Q*D21;

sys_cl= ss(A_cl,B_cl,C_cl,D_cl);

x0 = [0.1 ; -0.1; 0; 0.1; 0; 0; 0; 0];
t = 0:0.01:30;
w = [zeros(1,length(t))];
% w = sin(t);
[y, t, x] = lsim(sys_cl, w, t,x0);

figure('Position', [100, 100, 600, 600]);
sgtitle('Output Feedback Control through H_{\infty} Optimal Control')
state_variables = ["\alpha","\alpha_{dot}","\beta","\beta_{dot}"];
for i=1:4
```

```
    subplot(4,1,i);
    plot(t, x(:,i), 'b', t, x(:,i+n), 'r--','LineWidth',1);
    legend( sprintf('%s', state_variables(i)), sprintf('%s^{hat}',
    ↪  state_variables(i)),'FontSize',10);
    grid on;
    ylim([min(x(:,i)),max(x(:,i))])
end
xlabel('Time (seconds)')

figure('Position', [100, 100, 600, 600]);
sgtitle('Output Feedback Control through H_{\infty} Optimal Control')
state_variables = ["\alpha","\alpha_{dot}","\beta","\beta_{dot}"];
for i=1:4
    subplot(4,1,i);
    plot(t, x(:,i), 'b', t, x(:,i+n), 'r--','LineWidth',1);
    legend( sprintf('%s', state_variables(i)), sprintf('%s^{hat}',
    ↪  state_variables(i)),'FontSize',10);
    grid on;
    ylim([-1,1])
end
xlabel('Time (seconds)')
```

Listing 8: Designing Output Feedback Controller using $\mathcal{H}_\infty$ Optimal Control