Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Control Engineering

Robust Control for the Two-Wheeled-Legged Robot Sk8o

Master's thesis

Author: Petr Kuchař

Supervisor: Zdeněk Hurák

Academic Year: 2023/2024

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Kuchař Petr** Personal ID number: **516109**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Control Engineering**

Study program: **Cybernetics and Robotics**

II. Master’s thesis details

Master’s thesis title in English:

Robust control for the two-wheeled-legged robot Sk8o

Master’s thesis title in Czech:

Robustní řízení pro dvounohého robota na kolečkách Sk8o

Guidelines:

Build (or acquire) a simple mathematical model of the Sk8o robot suitable for model-based control design. Build (or acquire) also a more detailed (high-fidelity) model of the same robot suitable for verification purposes.
Since the bending of the knee (actually caused by the movement of the hip) causes a change in the height of the centre of gravity of the load of the imaginary Segway robot, one way to simplify (and linearize) the full 3D nonlinear model of the Sk8o robot is to use a simple inverted pendulum model a varying length of the pendulum. There are now two ways how to handle this varying parameter: either regard it as uncertain, that is, only known to be within some interval, or regard it indeed as varying in time with a measurement (or an estimate) of it.
Should it be useful, determine which other physical parameters (or even parts of the dynamics) are significantly uncertain and should be modelled as such. Provide then suitable model (or bounds) on these uncertainties.
Demonstrate accuracy of the model including the uncertainty using experimental data.
For existing and already deployed controllers such as LQR/LQG evaluate their robustness with respect to the "length of the pendulum". Does it give some guidance as for the nominal value of this parameter?
Design some robust controller(s) for the robot, that is, take the uncertainty explicitly into consideration during the computational design of controllers.
Design also a controller that takes advantage of being supplied with a measurement (or an estimate of the) "length of the pendulum" and thus can reparameterize. Explore the framework of Linear Parameter Varying (LPV) control.
Demonstrate the functionality of your controller(s) using a high-fidelity model, and perhaps even experiments with the real robot, if possible.

Bibliography / sources:

[1] F. Amato, Robust Control of Linear Systems Subject to Uncertain Time-Varying Parameters. in Lecture Notes in Control and Information Sciences, no. 325. Berlin: Springer, 2006.
[2] C. Hoffmann and H. Werner, ‘A Survey of Linear Parameter-Varying Control Applications Validated by Experiments or High-Fidelity Simulations’, IEEE Transactions on Control Systems Technology, vol. 23, no. 2, pp. 416–433, Mar. 2015.
[3] J. Mohammadpour and C. W. Scherer, Eds., Control of Linear Parameter Varying Systems with Applications. New York: Springer, 2012.
[4] O. Sename, P. Gaspar, and J. Bokor, Eds., Robust Control and Linear Parameter Varying Approaches: Application to Vehicle Dynamics. in Lecture Notes in Control and Information Sciences, no. 437. Berlin, Heidelberg: Springer, 2013.

Name and workplace of master's thesis supervisor:

doc. Ing. Zdeněk Hurák, Ph.D. Department of Control Engineering FEE

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **01.02.2024** Deadline for master's thesis submission: **24.05.2024**

Assignment valid until: **21.09.2025**

doc. Ing. Zdeněk Hurák, Ph.D. Supervisor's signature	prof. Ing. Michael Šebek, DrSc. Head of department's signature	prof. Mgr. Petr Páta, Ph.D. Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student's signature

DECLARATION

I hereby declare that I have independently authored the master's thesis, utilizing solely professional literature and sources, with a comprehensive list provided within.

Prague, May 24, 2024

. . . . . .
Petr Kuchař

ACKNOWLEDGEMENTS

I would like to thank my supervisor Zdeněk Hurák for his guidance, valuable advice, suggestions and dedicated time. I would also like to thank Krištof Pučejdl for his tips during the simulator creation and controller implementation. Furthermore, I thank Petr Brož for introducing me to the robot platform and software. Last and foremost, I would like to express my deepest gratitude to my parents for their love and support. I would not have made it this far without them.

ABSTRACT

This master's thesis focuses on analyzing and designing robust controllers for a two-wheeled-legged robot on wheels named Sk8o. The objective is to determine the optimal extension length of the robot's legs and subsequently design robust controllers. One controller will be designed based on the identified nominal length, while the other will benefit from measuring the robot's height. Therefore, the thesis first addresses system modeling. With the help of the model, an analysis is conducted based on which the controllers are designed. Finally, these controllers are tested and compared in simulations and on the real robot.

Keywords: linear quadratic regulator, robot, robust control, uncertainty, gain-scheduling, $\mathcal{H}_\infty$, loop-shaping

ABSTRAKT

Tato diplomová práce se zabývá analýzou a návrhem robustních regulátorů pro dvounohého robota na kolečkách jménem Sk8o. Cílem je nalézt optimální délku natažení nohou robota a následně navrhnout robustní regulátory. Jeden regulátor bude navrhnut na základě nalezené nominální délky, zatímco druhý bude benefitovat z měření výšky robota. Nejdříve se proto práce věnuje modelování systému. S pomocí modelu se provede analýza na základě čehož se navrhnou regulátory. Na závěr jsou tyto regulátory testovány a provnávány v simulacích a na reálném robotovi.

Klíčová slova: lineární kvadratický regulátor, robot, robustní řízení, neurčitost, gain-scheduling, $\mathcal{H}_\infty$, tvarování frekvenční charakteristiky

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Goals

This thesis aims to analyze and design robust controllers for the two-wheeled bipedal robot named Sk8o. Special attention will be paid to the problem of stretching and shortening its legs, which changes the robot's height. Notably, the analysis will be performed on the simplified wheeled inverted pendulum, whose rod length will represent the equivalent of the robot's height. Based on this analysis, two robust controllers will be designed to ensure stability and performance across all possible heights of the robot. One controller will utilize measurements of the imaginary rod length, while the other will assume the rod length falls within a specified range.

1.2 Motivation

The motivation for starting to analyze and design new controllers was that the robot oscillates at its maximum height with the currently deployed controller. This controller was designed at nearly its lowest height because it was intuitively considered that the height where it is the hardest to stabilize the robot is in its bent legs. It was shown that this idea was not the best, and from observation, it was seen that the performance was not ideal.

1.3 Outline

In Chapter 2 the robot Sk8o will be introduced. It will present its structure together with the components used in the robot.

Chapter 3 will present two used models. One simplified model is used for analysis and designing new controllers, together with a full rigid body simulation model, whose purpose is to test controllers before they are deployed to the real robot. The connection between these two models and the real robot will be shown. How to measure the robot's height will also be explained.

Next, Chapter 4 will present the structure of the linearized simplified model together with analysis, which leads to finding the optimal prolongation of the legs. Also, an analysis of the already deployed controller will be performed to find the length for which the original controller could be designed to ensure good performance across all heights of the robot.

In Chapter 5, two types of controllers will be introduced. One robust controller benefiting from the analysis in the previous chapter, while the second introduced controller will benefit from the measurement from which we can estimate the robot's height.

In the following two chapters, the experiments will be performed. In Chapter 6, the experiments in the simulator will be shown. The comparison between the controllers will be evaluated. The same holds for Chapter 7, where the experiments will be done on the real robot. Also, this chapter will compare the results achieved with those achieved in the simulator.

CHAPTER 2

SK8O ROBOT

This chapter will introduce the Sk8o, a two-wheeled bipedal robot, emphasizing its structure and components. The robot was originally developed in 2021 by Krištof Pučejdl and Martin Gurtner in the Advanced Algorithms for Control and Communications (AA4CC) group at the Faculty of Electrical Engineering (FEE) Czech Technical University in Prague (CTU). It was inspired by the robot Ascento [2], originally developed at ETH Zurich.

2.1 Related Works

Since the first development of the robot, much work, especially in the form of a thesis, was done. For this thesis, it is important to acknowledge the works that provided the foundational information and insights. The most significant source of information was gained from the thesis by Dominik Hodan [13]. Also, as a source of information, theses by Adam Kollarcik [9], and Petr Brož [21] were used. The robot Sk8o is captured in Figure 2.1.

Figure 2.1: Sk8o robot

2.2 Description

The robot has two legs with a closed kinematic chain ending with actuated wheels. These legs are attached to the body at the hip. In the body, all the electronics, except motors, are concerned. Each leg can be extended and stretched independently by controlling the corresponding motor installed in the hip. A rapid extension of both legs allows the robot to jump. Moreover, the joint is situated between the hip and wheels, referred to as the knee. The knees contain a torsion spring that partially counteracts gravity. Most of the parts were created using a 3D printer.

2.2.1 Actuators

The robot movement is handled by four eX8108 105KV brushless DC motors, two placed at the wheels and the other at the hips. They are controlled by Ben Katz's 3-phase motor controller [1]. Both position and torque control modes are available for all four motors, with a frequency of operation set at 40 kHz. Position control governs the extension of the legs at the hips, while the motor torque mode control is used for the wheels. The software imposes a maximum torque limit of 0.7 N m at the wheels. Additionally, the actuators at the hips are geared up by a

[1] `https://github.com/bgkatz/3phase_integrated`

factor of 16.5 to ensure adequate torque delivery.

2.2.2 Sensors

The robot has an Inertial Measurement Unit (IMU) to measure accelerations, angular rates, and Euler angles. More precisely, it utilizes the ICM-42688 sensor, featuring a built-in anti-aliasing filter, and is configured with a bandwidth of 1051 Hz. Moreover, each motor controller monitors and reports the motor's position, velocity, and torque. For completeness, the robot can be integrated with a RealSense camera for extended functionality. Its inclusion is not important to the focus of this thesis.

2.2.3 Computers

There are currently two computational units in the robot, namely the Teensy 4.0 board and the Odroid N2+ computer. Teensy performs periodic tasks, such as collecting measurements from the motor controllers and IMU, as well as passing the required actions to the motor controllers. Subsequently, this measurement data is transmitted to the Odroid computer on demand. Programmed in the C++ language, the Teensy operates with its main loop running at a frequency of 1 KHz. The Odroid board facilitates remote interaction with the robot through Wi-Fi connectivity and/or an Xbox controller. Operating on Ubuntu Linux, this board organizes its functionalities as services programmed in C++ or Python. The schematic of components is in Figure 2.2.

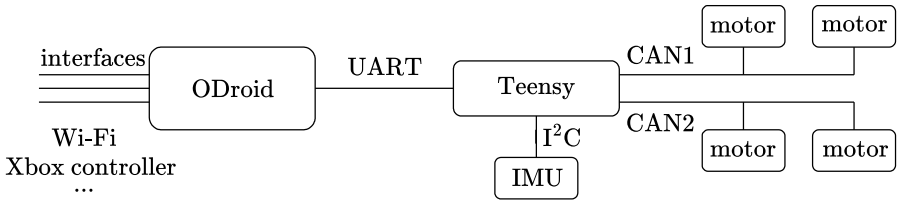

Figure 2.2: Electronics diagram. Reproduced from [13]

CHAPTER 3

MODELING SYSTEM DYNAMICS

This chapter introduces two types of models, each serving specific purposes. The simple control-oriented model, primarily used for some analysis and design controllers. The second model, known as the high-fidelity, is intended to verify the designed controllers before they are deployed to the real robot. This high-fidelity model closely replicates the full dynamics of the robot. Additionally, the chapter will draw analogies between these two models.

3.1 Simple Control-Oriented Model

For analysis and control design, the Segway-like model (a wheeled inverted pendulum) representing a simplified model of the Sk8o robot will be used. Notably, when the robot's legs are fixed, its behavior resembles that of a Segway-like model. This model can be seen in Figure 3.1 and was obtained from previous work by Dominik Hodan [13] and [18]. The model will be presented in the form of state-space equations, along with its linearized approximation. Unlike previous works, this model will incorporate dependence on the rod length in its linear approximation.

The generalized coordinates of the model are position in x coordinate, pitch angle ϕ, and yaw angle ψ. These coordinates are concentrated to a single vector $\mathbf{q}_1$. Its derivatives are also concentrated to a vector, this time to $\mathbf{q}_2$. By stacking these two vectors, we got the final $\mathbf{q}$ state vector. The model has two torque inputs, one u_L from the left wheel and the second u_R from the right wheel, concentrated to input vector $\mathbf{u}$. Finally, the state space representation is

$$\dot{\mathbf{q}} = \begin{bmatrix} \mathbf{q}_1 \\ \mathbf{M}^{-1}\left(\mathbf{B}\mathbf{u} - \mathbf{C} - \mathbf{D}\mathbf{q}_2 - \mathbf{G}\right) \end{bmatrix} = \mathbf{f}(\mathbf{q}, \mathbf{u}), \tag{3.1.1}$$

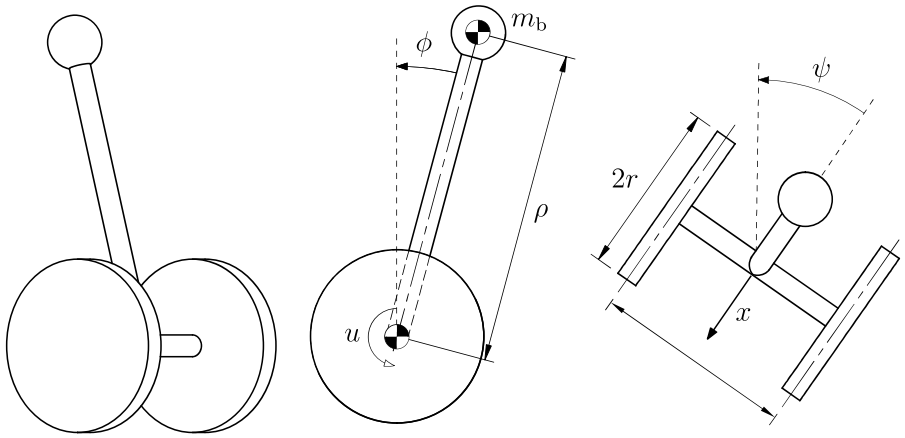

Figure 3.1: Segway-like model. Reproduced from [13]

where matrices are in the following form

$$\mathbf{M} = \begin{bmatrix} m_{11} & m_{12} & 0 \\ m_{21} & m_{22} & 0 \\ 0 & 0 & m_{33} \end{bmatrix}, \mathbf{B} = \begin{bmatrix} \frac{1}{r} & \frac{1}{r} \\ -1 & -1 \\ -\frac{w}{r} & -\frac{w}{r} \end{bmatrix}, \mathbf{C} = \begin{bmatrix} 0 & c_{12} & c_{13} \\ 0 & 0 & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix},$$

$$\mathbf{D} = \begin{bmatrix} d_{11} & d_{12} & 0 \\ d_{21} & d_{22} & 0 \\ 0 & 0 & d_{33} \end{bmatrix}, \mathbf{G} = \begin{bmatrix} 0 \\ -mg\rho\cos\phi \\ 0 \end{bmatrix}, \tag{3.1.2}$$

with the following elements of matrices

$$m_{11} = m_\mathrm{b} + 2m_\mathrm{w} + 2\frac{J}{r^2}, \quad m_{12} = m_{21} = m_\mathrm{b}\rho\cos\phi, \quad m_{22} = I_{by} + m_\mathrm{b}\rho^2,$$

$$m_{33} = I_{bz} + 2K + \frac{w^2}{2}\left(m_\mathrm{w} + \frac{J}{r^2}\right) - \left(I_{bz} - I_{bx} - m_\mathrm{b}\rho^2\right)\sin^2\phi,$$

$$c_{12} = -m_\mathrm{b}\rho\dot{\phi}\sin\phi, \quad c_{13} = -c_{31} = m_\mathrm{b}\rho\dot{\psi}\sin\phi,$$

$$c_{23} = -c_{32} = \left(I_{bz} - I_{bx} - m_\mathrm{b}\rho^2\right)\dot{\psi}\sin\phi\cos\phi,$$

$$c_{23} = -\left(I_{bz} - I_{bx} - m_\mathrm{b}\rho^2\right)\dot{\phi}\sin\phi\cos\phi,$$

$$d_{11} = \frac{2b}{r^2}, \quad d_{12} = d_{21} = -\frac{2b}{r}, \quad d_{22} = 2b, \quad d_{33} = \frac{w^2 b}{2r^2},$$

where the values of these parameters are listed in table 3.1. It is important to note that the rod length of the Segway model is not a fixed value but varies due to the adjustable height of the actual robot.

Symbol	Parameter	Value	Unit
w	distance between wheels	0.29	m
ρ	rod length	0.26 - 0.43	m
r	wheel radius	0.08	m
m_{b}	body mass	4	kg
m_{w}	wheel mass	0.3	kg
J	wheel moment of inertia about turning axis	735×10^{-6}	kg m^2
K	wheel moment of inertia about vertical axis	39×10^{-5}	kg m^2
I_{bx}	roll moment of inertia	484×10^{-4}	kg m^2
I_{by}	pitch moment of inertia	377×10^{-4}	kg m^2
I_{bz}	yaw moment of inertia	406×10^{-4}	kg m^2
b	wheel damping	0.01	N m s rad^{-1}
g	gravity constant	9.81	m s^{-2}

Table 3.1: List of parameters for Segway-like model

3.1.1 Linearized model

Since all methods and analyses in this thesis require a linear model, a linear approximation of the model must be performed. The linear approximation about its equilibrium $\mathbf{q} = \mathbf{0}, \mathbf{u} = \mathbf{0}$ is computed by the Jacobians of the state space model 3.1.1 as follows

$$\mathbf{A}(\rho) = \left.\frac{\partial \mathbf{f}(\mathbf{q},\mathbf{u})}{\partial \mathbf{q}}\right|_{\mathbf{q}=\mathbf{0},\mathbf{u}=\mathbf{0}}, \mathbf{B}(\rho) = \left.\frac{\partial \mathbf{f}(\mathbf{q},\mathbf{u})}{\partial \mathbf{u}}\right|_{\mathbf{q}=\mathbf{0},\mathbf{u}=\mathbf{0}}. \tag{3.1.3}$$

Because the real robot can change its height, both the state matrix $\mathbf{A}(\rho)$ and input matrix $\mathbf{B}(\rho)$ depend on the rod length. The position and orientation will not affect further analysis and control. That is why the states x and ψ are removed from state vector $\mathbf{q}$. From now on, if the vector $\mathbf{q}$ will be mentioned, we mean the reduced version. The same holds for the state and input matrices.

3.1.2 Models for control design

To address the differences between the real robot and the simulation model, which will be detailed in section 3.2.3, we first reorder the states and matrices using a transformation matrix. This reordering simplifies the subsequent transformations between the real robot states and the states used in the simulator. The

transformation matrix is defined as

$$\mathbf{T} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.1.4}$$

The transformation is then performed in the following manner

$$\mathbf{q} = \mathbf{T}\mathbf{q}, \quad \mathbf{A}(\rho) = \mathbf{T}\mathbf{A}(\rho)\mathbf{T}^{-1}, \quad \mathbf{B}(\rho) = \mathbf{T}\mathbf{B}(\rho). \tag{3.1.5}$$

In this thesis, two different types of controllers will be designed. For this purpose, we introduce two different structures of the system. One is in continuous time, which serves for dynamic controller design, and one is in discrete time for state feedback with integral actions.

System for dynamic controller

This structure of the system will not depend on the rod length. It will be set at its nominal value found in 4.3. We can measure all states there, so the output matrix $\mathbf{C}$ will be the identity matrix, and the feedthrough matrix $\mathbf{D}$ will be zero. Note that this system will sometimes be called a plant during the text. The minimal state space realization of this system will be marked as

$$\mathbf{G} = (\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{0}). \tag{3.1.6}$$

The controller developed based on this model will then be discretized for deployment to the real robot.

System for state feedback

Therefore, for deployment onto the physical robot, the state feedback controller must operate within a discrete time. For this purpose, the model must be discretized. This can be done simply by zero-order hold discretization as follows

$$\mathbf{A}_\mathrm{d}(\rho) = e^{\mathbf{A}(\rho)h}, \quad \mathbf{B}_\mathrm{d}(\rho) = \int_0^h e^{\mathbf{A}(\rho)h}\, dt \mathbf{B}(\rho), \tag{3.1.7}$$

where h is the sampling period. This controller aims to stabilize the system in the upper position and allow tracking of the velocity $\dot{x}$ and yaw rate $\dot{\psi}$. Since, for

this system, the controller will be state feedback, it is necessary to add integral action. The extended system by this integral action is as follows

$$\underbrace{\begin{bmatrix}\mathbf{q}_{k+1}\\ \boldsymbol{\epsilon}_{k+1}\end{bmatrix}}_{\mathbf{z}(k+1)} = \underbrace{\begin{bmatrix}\mathbf{A}_{\mathrm{d}}(\rho) & \mathbf{0}\\ -\mathbf{L} & \mathbf{I}\end{bmatrix}}_{\mathbf{A}_{\mathrm{i}}(\rho)}\underbrace{\begin{bmatrix}\mathbf{q}_k\\ \boldsymbol{\epsilon}_k\end{bmatrix}}_{\mathbf{z}(k)} + \underbrace{\begin{bmatrix}\mathbf{B}(\rho)\\ \mathbf{0}\end{bmatrix}}_{\mathbf{B}_{\mathrm{i}}(\rho)}\mathbf{u}_k + \underbrace{\begin{bmatrix}\mathbf{0}\\ \mathbf{I}\end{bmatrix}}_{\mathbf{B}_{\mathrm{r}}}\mathbf{r}_k \tag{3.1.8}$$

where $\mathbf{r}_k$ is the reference signal, $\mathbf{I}$ is an identity matrix, and $\mathbf{L}$ is a matrix that defines what states we want to track. In our case, this matrix has the form

$$\mathbf{L} = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\end{bmatrix}. \tag{3.1.9}$$

Then, the state feedback gain $\mathbf{K}_{\mathrm{i}}$ can be found for this augmented system.

3.2 High Fidelity Model

For testing the new controllers, a simulation model that is as accurate as possible to that of the real robot is needed. In previous work by Dominik Hodan, this model was developed, but in MuJoCo [26]. Because this thesis uses MATLAB [17] for analysis and developing new controllers, it would be demanding to rewrite each controller to this model. For this purpose, as part of this thesis, it was also developing this model in MATLAB Simscape [3]. In this environment, the entire model is depicted through the interconnection of physical components represented as blocks. Unlike classical manipulators, which have fixed bases, this model type falls into the category of floating base systems [22]. This means the robot's body is represented as a moving base, and the end effector is, in this situation, the robot's wheels. The model is, therefore, modeled from body to wheels.

3.2.1 Modelling

To facilitate easy handling of the model, it is advisable to introduce the coordinate frames of the robot first. To accomplish this, it is necessary to derive the appropriate transformations between each frame. The resulting frames for the right leg are shown in Figure 3.2. An additional view of the robot is then in Figure 3.3. All corresponding parameters are reproduced from [13]. The moments of inertia were also reproduced, except for the body, which was updated. These parameters can be seen in the Table 3.2 and moment of inertia in 3.3. The transformation

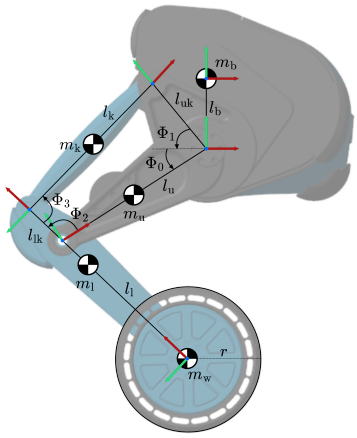

Figure 3.2: Right leg description

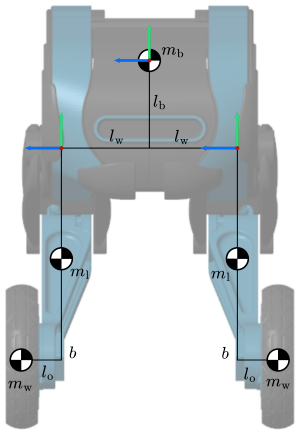

Figure 3.3: Front robot description

Body	Symbol	Value	Unit
body	m_b	3	kg
	l_b	100	mm
	l_{uk}	93	mm
	l_w	92.5	mm
upper leg	m_u	0.2	kg
	l_u	188	mm
lower leg	m_l	0.2	kg
	l_l	190	mm
	l_{lk}	50	mm
	b_k	0.15	N m s rad^{-1}
kinematic loop	m_k	0.1	kg
	l_k	193	mm
wheel	m_w	0.3	kg
	l_o	25.6	mm
	r	80	mm
-	b	0.01	N m s rad^{-1}

Table 3.2: Parameters of the full model

between each frame was defined as rotation $\mathbf{R}_{ij}$ followed by translation $\mathbf{t}_{ij}$. The

Body	I_{xx} [kg m^2]	I_{yy} [kg m^2]	I_{zz} [kg m^2]
body	4.84×10^{-2}	4.06×10^{-2}	3.77×10^{-2}
upper leg	6.04×10^{-4}	1.67×10^{-5}	5.91×10^{-4}
lower leg	9.75×10^{-4}	1.67×10^{-5}	9.62×10^{-4}
kinematic loop	6.36×10^{-4}	1.67×10^{-5}	6.22×10^{-4}
wheel	3.90×10^{-4}	3.90×10^{-4}	7.35×10^{-4}

Table 3.3: Inertia parameters of different parts

structure of the rotation and translation is defined as follows

$$\mathbf{R}_{ij} = \begin{bmatrix} \cos\Phi_i & -\sin\Phi_i & 0 \\ \sin\Phi_i & \cos\Phi_i & 0 \\ 0 & 0 & 1 \end{bmatrix}, \mathbf{t}_{ij} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{3.2.1}$$

As can be seen, the rotation is considered only around the z-axis with angle Φ_i and translation in x, y, and z-axis. The described transformation for the right leg is listed in Table 3.4. The corresponding transformation for the left leg is defined similarly.

(i,j)	Φ_i	t_x	t_y	t_z
$(0,1)$	0	0	$-l_{\mathrm{b}}$	l_{w}
$(1,2)$	Φ_1	$-l_{\mathrm{u}}$	0	0
$(2,3)$	Φ_2	l_{lk}	0	0
$(3,4)$	$-\Phi_3$	l_{k}	0	0
$(2,5)$	Φ_2	$-l_{\mathrm{l}}$	0	0
$(0,5)$	$-\Phi_0$	$-l_{\mathrm{uk}}$	0	0

Table 3.4: List of variables for right leg

Then, the model based on these parameters was designed. First, the typical blocks, such as solver configuration, world (reference) frame, and mechanism configuration blocks, were created. The interconnection of these blocks is evident in [4]. Subsequently, the appropriate transformation establishes the connection between the reference frame, the body, and the ground. The robot's body is represented as the 6-DOF joint [1]. This joint is then followed by transformations defined in Table 3.4, wherein each frame is the Revolute Joint [6]. Because this system is modeled as a floating base system, the contacts between the wheels and the floor at least had to be defined. These contacts are modeled by block Spatial Contact Force block [7]. These contacts are essential to address the interaction

between the robot, particularly the wheels and the ground. These contacts then seem to influence the behavior of the robot. However, in this thesis, the default parameters were used. The developed model [1] in MATLAB Simscape is in Figure 3.4.

Figure 3.4: Modeled the Sk8o robot in the Simscape environment

3.2.2 Connection to simple control-oriented model

All previously developed controllers, including the newly developed controllers in this thesis, utilize the states derived from an imaginary Segway-like model. For this purpose, these states must be extracted from the high-fidelity model. The forward velocity $\dot{x}$ is measured from wheel velocities as

$$\dot{x} = \frac{r}{2}(\dot{\theta}_{\mathrm{R}} + \dot{\theta}_{\mathrm{L}}), \tag{3.2.2}$$

where $\dot{\theta}_{\mathrm{R}}$ is velocity from right wheel and $\dot{\theta}_{\mathrm{L}}$ from left wheel. Similarly the yaw rate $\dot{\psi}$ is computed as

$$\dot{\psi} = \frac{r}{2(l_{\mathrm{w}} + l_{\mathrm{o}})}(\dot{\theta}_{\mathrm{R}} - \dot{\theta}_{\mathrm{L}}). \tag{3.2.3}$$

[1] `https://gitlab.fel.cvut.cz/aa4cc/sk8o/rocond`

The pitch rate $\dot{\phi}$ and pitch angle ϕ are then measured using Transform Sensor block [8]. This block measures states between the body and the world frame. Subsequently, the pitch angle is calculated as follows

$$\phi = \text{atan2}(-r_{31}, \sqrt{r_{32}^2 + r_{33}^2}), \tag{3.2.4}$$

where r_{ij} is the position index in the rotation matrix provided by the transform sensor. The computation of pitch angle is a little bit complicated. First, the skew-symmetric matrix is computed as

$$\begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} = \dot{\mathbf{R}}\mathbf{R}^\top \tag{3.2.5}$$

where $\dot{\mathbf{R}}$ is the time derivative of the rotation matrix the Transform Sensor provides. This gives us angular velocities around each axis in the world frame. Therefore, another transformation is necessary as we require the angular velocities in the body frame. After un-skewing the skew-symmetric matrix, the transformation can be written as

$$\begin{bmatrix} \omega_z \\ \omega_y \\ \omega_z \end{bmatrix} = \mathbf{R}^\top \begin{bmatrix} \omega_z \\ \omega_y \\ \omega_z \end{bmatrix}. \tag{3.2.6}$$

Then, finally, the pitch rate is given as $\dot{\phi} = -\omega_z$.

Rod length Computation

Next, it is important to connect the rod length of the imaginary Segway-like model to the height of the full robot. The change of the height of the full robot is controlled by the angle Φ_1 at the hips. The rod length can be computed from this angle based on the figure 3.5 in the following manner. Note that this computation is nothing more than using the law of cosine. The diagonal length d_1 in the kinematic loop is computed as

$$d_1 = \sqrt{l_\text{u}^2 + l_\text{uk}^2 + 2l_\text{u}l_\text{uk}\cos(\Phi_1 + \Phi_0)} \tag{3.2.7}$$

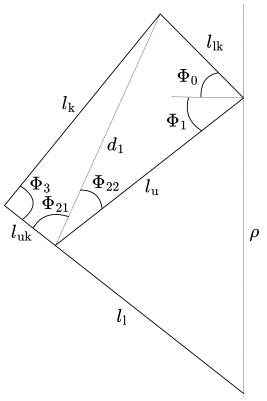

Figure 3.5: Leg description

Then, the angles Φ_{12} and Φ_{22}, where its sum is angle Φ_2 can be derived as

$$\Phi_{12} = \arccos\left(\frac{l_u^2 + d_1^2 - l_{uk}^2}{2 l_u d}\right), \tag{3.2.8}$$

$$\Phi_{22} = \arccos\left(\frac{l_{lk}^2 + d_1^2 - l_k^2}{2 l_{lk} d}\right). \tag{3.2.9}$$

Finally, the rod length approximating the actual height of the robots is given by

$$\rho = \sqrt{l_u^2 + l_l^2 + 2 l_u l_l \cos(\pi - (\Phi_{12} + \Phi_{22}))} + l_b. \tag{3.2.10}$$

Joint angles initialization

If we want to initialize the angles in joints based on the rod length, we can compute the angle Φ_2 at the knee as

$$\Phi_2 = \pi - \arccos\left(\frac{l_u^2 + l_l^2 - (\rho - l_b)^2}{2 l_u l_l}\right). \tag{3.2.11}$$

Then, the d_2 could be computed as

$$d_2 = \sqrt{l_u^2 + l_{uk}^2 + 2 l_u l_{uk} \cos(\Phi_2)}. \tag{3.2.12}$$

Based on the d_2 we can now derive angles Φ_{11} and Φ_{22} in the following manner

$$\Phi_{11} = \arccos\left(\frac{d_2^2 + l_{lk}^2 - l_k^2}{2 d_2 l_{lk}}\right) - \Phi_0, \tag{3.2.13}$$

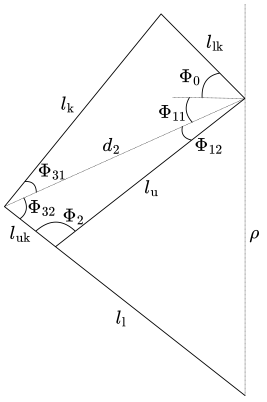

Figure 3.6: Leg description

$$\Phi_{12} = \arccos\left(\frac{d_2^2 + l_u^2 - l_k^2}{2 d_2 l_u}\right). \tag{3.2.14}$$

The sum of these two angles forms the Φ_1. Last angle Φ_3 which consist of the sum of Φ_{31} and Φ_{32} can be given by

$$\Phi_3 = \arccos\left(\frac{d_2^2 + l_k^2 - l_{lk}^2}{2 d_2 l_k}\right) + \arccos\left(\frac{d_2^2 + l_{uk}^2 - l_u^2}{2 d_2 l_{uk}}\right). \tag{3.2.15}$$

3.2.3 Differences to the real robot

This model should be as close to the real robot as possible, but still, some differences from the real robot were made. After all, it is indeed just a simplification of the real world, but still, it closely imitates the system's dynamic. The difference was made in how the states of the imaginary Segway-like model introduced in 3.2.2 are compared to how it is measured in the real robot.

The states are not the same as introduced in 3.2.2. The states of the Segway-like model in the real robot are measured as

$$\dot{x} = \dot{\theta}_{\mathrm{L}} + \dot{\theta}_{\mathrm{R}} \tag{3.2.16}$$

$$\dot{\psi} = \dot{\theta}_{\mathrm{L}} - \dot{\theta}_{\mathrm{R}}. \tag{3.2.17}$$

The pitch rate $\dot{\phi}$ and pitch angle ϕ are computed from the IMU sensor, so there is no difference. Thanks to the state reorganization introduced in 3.1.2, the next transformations will have a diagonal structure. The transformation matrices then have the following form

$$\mathbf{T_q} = \begin{bmatrix} -\frac{r}{w} & 0 & 0 & 0 \\ 0 & -\frac{r}{2} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{T_r} \begin{bmatrix} -\frac{r}{w} & 0 \\ 0 & -\frac{r}{2} \end{bmatrix} \tag{3.2.18}$$

where $\mathbf{T_q}$ transforms states introduced in 3.2.2 to the real robot states and the $\mathbf{T_r}$ transform the references. A state feedback controller was also introduced in this thesis. This controller can be easily transformed to a real robot as

$$\mathbf{K}_{\mathrm{i}} = \mathbf{K}_{\mathrm{i}} \underbrace{\begin{bmatrix} \mathbf{T_q} & \\ & \mathbf{T_r} \end{bmatrix}}_{\mathbf{T_{qr}}}, \tag{3.2.19}$$

and by multiplying the controller again but now with $\mathbf{T}_{\mathbf{qr}}^{-1}$ we get the controller back to the simulation model.

Another difference is in the hip angle references and measurement. Because one of the controllers will benefit from this measurement, feeding him correct hip angles is crucial. The equation what do this transformation is

$$\Phi_{\mathrm{s}} = \alpha + \frac{\Phi_{\mathrm{r}}}{16.5}, \tag{3.2.20}$$

where Φ_{s} is simulator hip angle, which also controller expect and Φ_{r} is real robot hip angle. The $\alpha = 66.5$ degrees is an angle constant.

CHAPTER 4

UNCERTAINITY MODELLING

This chapter focuses on system properties determined mostly by the rod length of a Segway-like model. First, the structure of the Segway-like model, depending on the rod length, will be shown. Then, the rod length will be taken as the uncertain parameter, which can vary between its maximal and minimal value. Based on this, the nominal value of some criterion will be found.

4.1 Parameter Dependence Structure

For further analysis, we need to know how the already linearized system depends on the rod length. For this purpose, the state matrix $\mathbf{A}(\rho)$ and input matrix $\mathbf{B}(\rho)$ dependent on this parameter with some rounding up for a better view is shown below

$$\mathbf{A}(\rho) = \begin{bmatrix} \frac{-0.08l^2-0.0064l-0.0002}{0.021l^2+0.00037} & \frac{0.0064l^2+0.0005l}{0.021l^2+0.00037} & 0 & \frac{-1.0045l^2}{0.021l^2+0.00037} \\ \frac{0.0064l+0.0006}{0.0017l^2+2.9\times10^{-5}} & \frac{-0.0005l}{0.0017l^2+2.9\times10^{-5}} & 0 & \frac{0.097l}{0.0017l^2+2.9\times10^{-5}} \\ 0 & 0 & -2.18 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \tag{4.1.1}$$

$$\mathbf{B}(\rho) = \begin{bmatrix} \frac{-0.32l^2-0.0256l-0.00095}{0.021l^2+0.00037} & \frac{-0.32l^2-0.0256l-0.00095}{0.021l^2+0.00037} \\ \frac{0.0256l+0.00247}{0.0017l^2+2.9\times10^{-5}} & \frac{0.0256l+0.00247}{0.0017l^2+2.9\times10^{-5}} \\ 60.24 & -60.24 \\ 0 & 0 \end{bmatrix}. \tag{4.1.2}$$

The fraction in these matrices is caused by the inverse of the mass matrix $\mathbf{M}$ in the equation 3.1.1. We can see that the yaw rate $\dot{\psi}$ does not depend on the rod length. If we plot each of the elements of state matrix $\mathbf{A}(\rho)$, which depends on

the rod length, we will see that the development is smooth and seems to tend to be quadratic. This shows the Figure 4.1. The development of the elements of

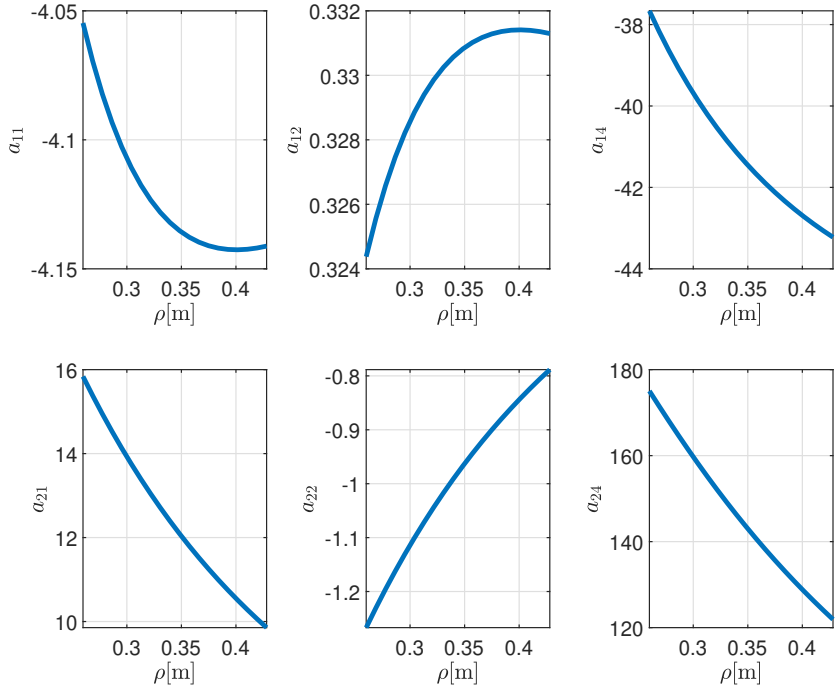

Figure 4.1: Depence of element in state matrix

the input matrix $\mathbf{B}(\rho)$ will be similar to how elements develop based on the rod length.

4.1.1 Depence of eigenvalues on the rod length

Above, we found that the development of elements of the matrices is smooth. A similar holds for the development of the eigenvalues, which is not surprising. Because the dependence of eigenvalues on the rod length is long and hence it is impossible to show it, at least the plot of the dependence based on the rod length is shown in Figure 4.2. Considering that the eigenvalues lack an imaginary part, they are represented without reference to the complex plane. Moreover, the eigenvalue λ_3 is omitted as it remains constant. We can see that with the higher rod lengths, the eigenvalues tend to go to the origin both for stable and

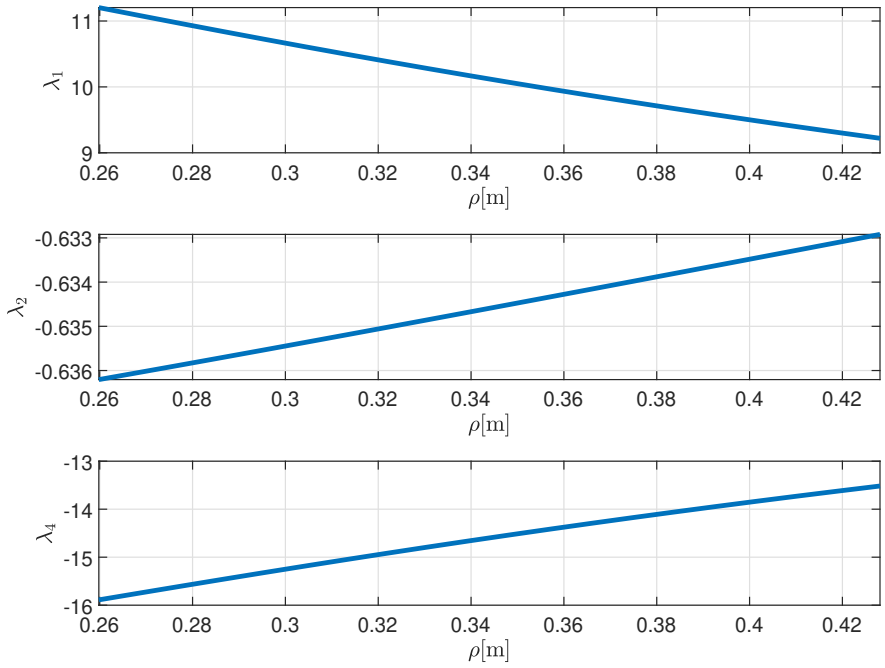

Figure 4.2: Eigenvalues rod length dependence

unstable eigenvalues. This analysis will have the impact of the one-designed controller, which uses measurement of this imaginary rod length, more precisely of the stretch of the legs in the full robot. See 5.2 for more details.

4.2 Analysis for Already Deployed LQ Controller

In the current state of this thesis, the primary controller implemented in the real robot is the Linear Quadratic (LQ) controller. It was originally developed for the rod length $\rho = 0.2907$ meters, slightly lower than the middle of the possible rod length value. The measurements and observations showed that the full robot behaves more oscillatory if the legs are stretched. This section aims to analyze the robust stability and performance of this controller. Also, to determine at which rod length the LQ controller should be roughly designed. We switch to the discrete time for this analysis because the controller is discrete, and we use the same system structure introduced in 3.1.2 for state feedback. The controller

was first transformed using inversion in the 3.2.19 to the state of the Segway-like model.

4.2.1 Robust stability

Above, the system was said to be stable at each rod length. Indeed, if we visualize the eigenvalues at each rod length within the possible value range, all will be inside the unit circle. From the analysis done in 4.1, we can expect that if we grid the discrete-time system finely, the trajectory of the development of the eigenvalues will be fully captured. The eigenvalues in the complex plane are shown in the Figure 4.3. This figure confirms that truly all eigenvalues lie inside a unit circle. Also, it is worth noting that the eigenvalue, which has seen the largest

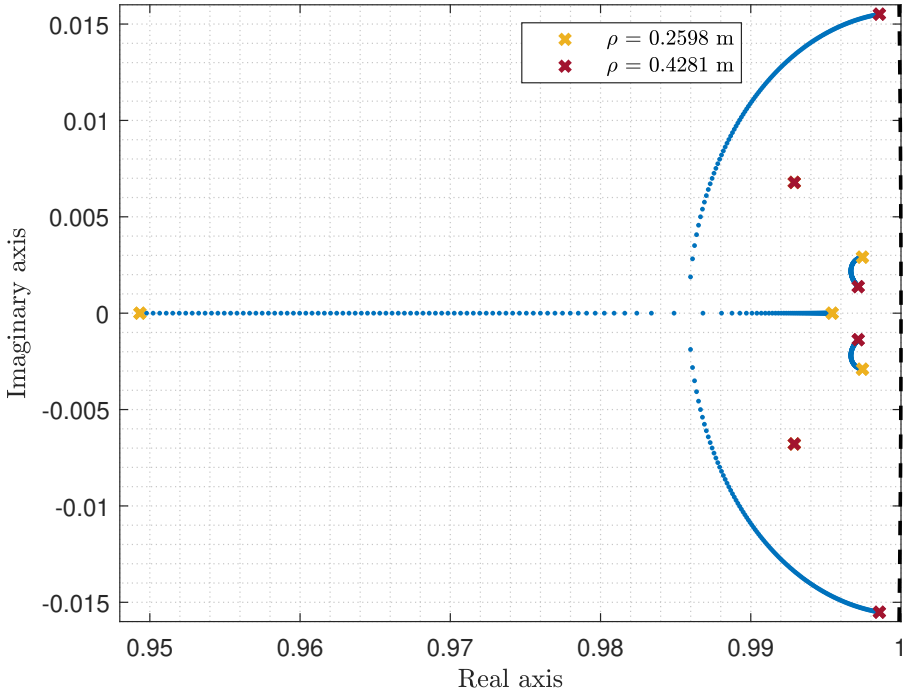

Figure 4.3: Dependence of eigenvalues on rod length

development, tends to go outside the unit circle with a higher rod length. Indeed, if we adjust the rod length, we will get an unstable system at some point.

4.2.2 Robust performance

Now that we know the system is stable at each rod length, we can analyze robust performance. However, the responses to unit references will be shown before we start the analysis. Figure 4.4 captures the responses at different rod lengths. We

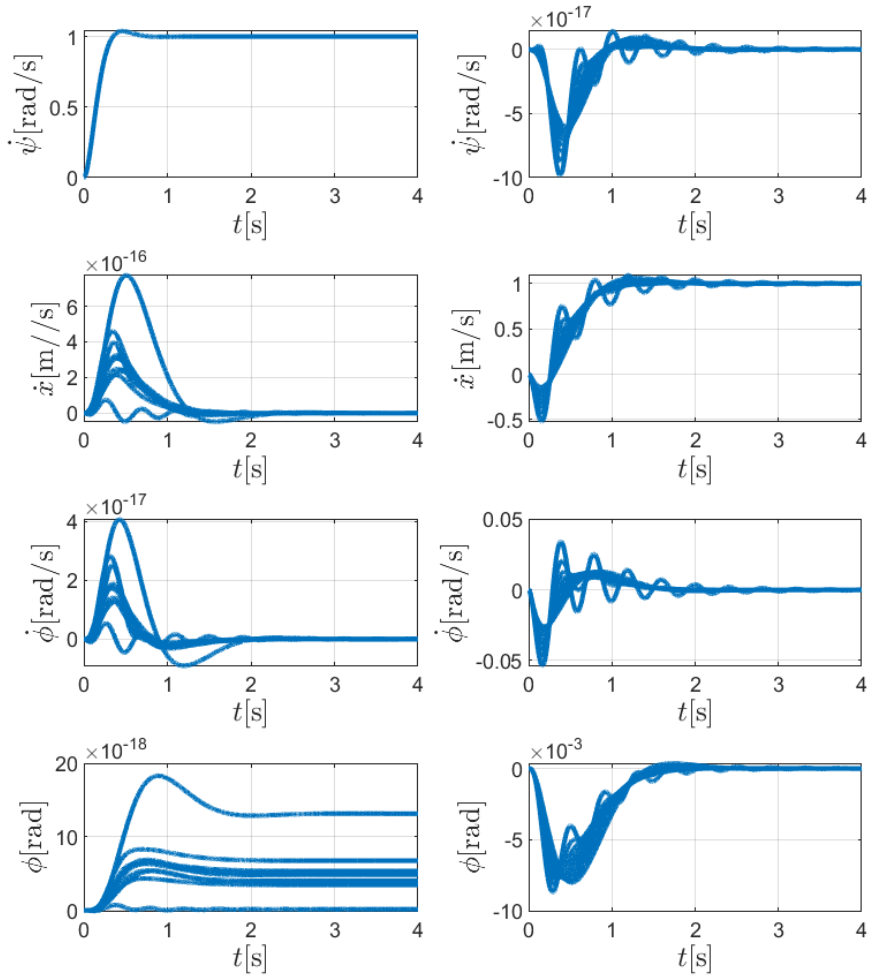

Figure 4.4: Responses to unit references

can conclude that the responses are relatively pretty fast. The velocity converges

to its reference within approximately one second, while the yaw rate achieves its reference in nearly half a second.

One possible way to analyze robust performance is to compute $\mathcal{H}_\infty$ norm. This norm is defined as the peak gain of the system $\mathbf{G}$ across all frequencies and all input directions [5]. Along this, we plot the system's singular values for better insight. This $\mathcal{H}_\infty$ norm can be computed by solving the following optimization problem outlined in [11] as

$$\begin{aligned} &\min_{\mathbf{P},\gamma} \quad \gamma \\ &\text{s.t.} \quad \begin{bmatrix} -\mathbf{P} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{\text{cl}}{}^\top \mathbf{P} \mathbf{A}_{\text{cl}} - \mathbf{P} & \mathbf{A}_{\text{cl}}{}^\top \mathbf{P} \mathbf{B}_{\text{r}} & \mathbf{C}^\top \\ \mathbf{0} & \mathbf{B}_{\text{r}}{}^\top \mathbf{P}^\top \mathbf{A}_{\text{cl}} & \mathbf{B}_{\text{r}}{}^\top \mathbf{P} \mathbf{B}_{\text{r}} - \gamma \mathbf{I} & \mathbf{D}^\top \\ \mathbf{0} & \mathbf{C} & \mathbf{D} & -\gamma \mathbf{I} \end{bmatrix} < 0, \end{aligned} \tag{4.2.1}$$

where $\mathbf{P}$ is the symmetric matrix solution. This optimization problem can be solved by [16]. By solving this optimization problem above, we obtain minimal γ, which means that all singular values of the state space realization $\mathbf{G}_{\text{cl}} = (\mathbf{A}_{\text{cl}}, \mathbf{B}_{\text{r}}, \mathbf{C}, \mathbf{D})$ is below this γ value. If we solve this optimization problem for systems at different rod lengths, we get the set of these γ values. The maximum found is $\gamma = 3.3$, corresponding to the maximum possible rod length. This statement supports Figure 4.5, which shows the singular values at different rod lengths. We can see that the maximum γ found corresponds to the maximum possible rod length. This result supports the observation that the system oscillates slightly at about its maximum height. In the next section, we will look at what happens when the controller is designed at near the maximum rod length.

4.2.3 Optimal rod length LQ controller design

Based on the analyses above, we can say that the controller could be designed near the maximum to get the peak low at this height. Again, the eigenvalues for this will be shown together with the $\mathcal{H}_\infty$ norm and singular values to support this statement. Also, the responses to the references will be shown at the end. We know how fast the dynamic should be from the 4.2.2. Based on this, the controller was designed for the rod length with value $\rho = 0.4$ meters. The dependence of eigenvalues on the rod length is in Figure 4.6, which shows that the system is also stable at each rod length. Examining Figure 4.7, depicting the singular values, reveals the absence of resonance peaks, with the maximum γ at just 1.08. These findings indicate a significant improvement in robust performance. Lastly, the responses to the unit yaw rate reference $\dot{\psi}_{\text{ref}}$ are in the figure 4.8 together with

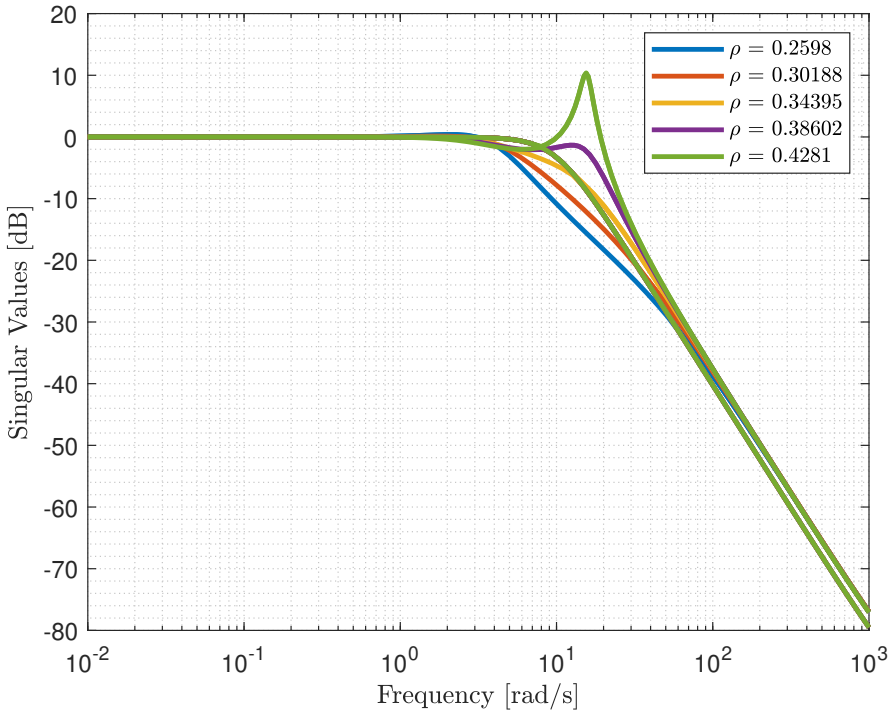

Figure 4.5: Singular values at different rod lengths

the responses to the unit velocity reference $\dot{x}_{\text{ref}}$. We can see no oscillation there, which means the responses are also better at this rod length, as expected. The responses at different rod lengths are similar to each other.

From this analysis, we can say that the LQ controller should be designed in the neighborhood of its maximum value for similar performance at each rod length. However, because it could be hard to determine the exact optimal rod length and this analysis will not be very important for designing the controller, the exact optimal rod length will not be provided. Lastly, from some experimenting, we can at least provide the range where the performance is still good, and also, we will not see much difference between performance in this range. The optimal rod length ρ is between 0.38 and 0.43 meters.

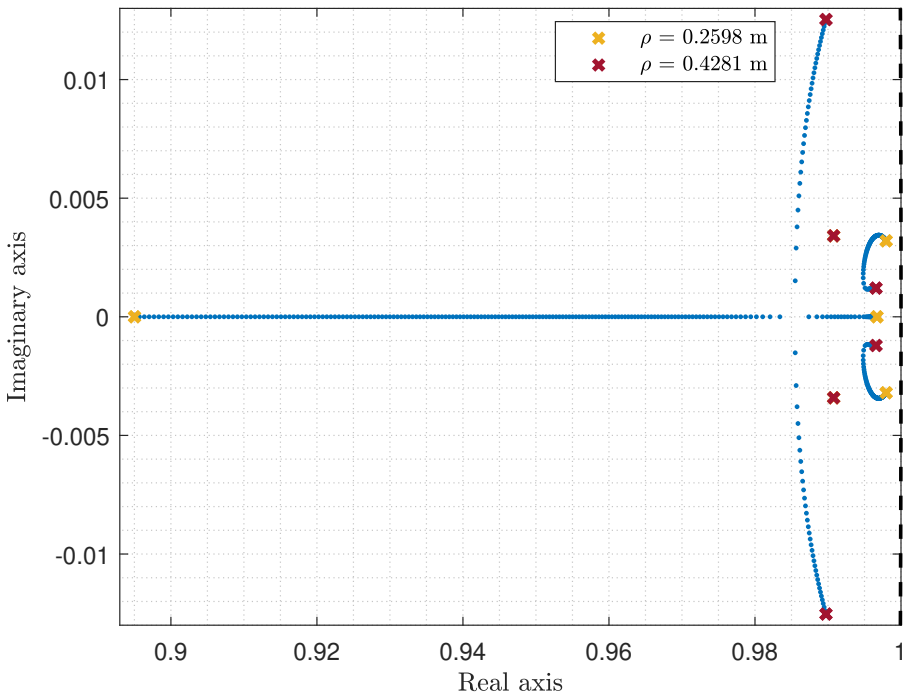

Figure 4.6: Dependence of eigenvalues on rod length in the complex plane

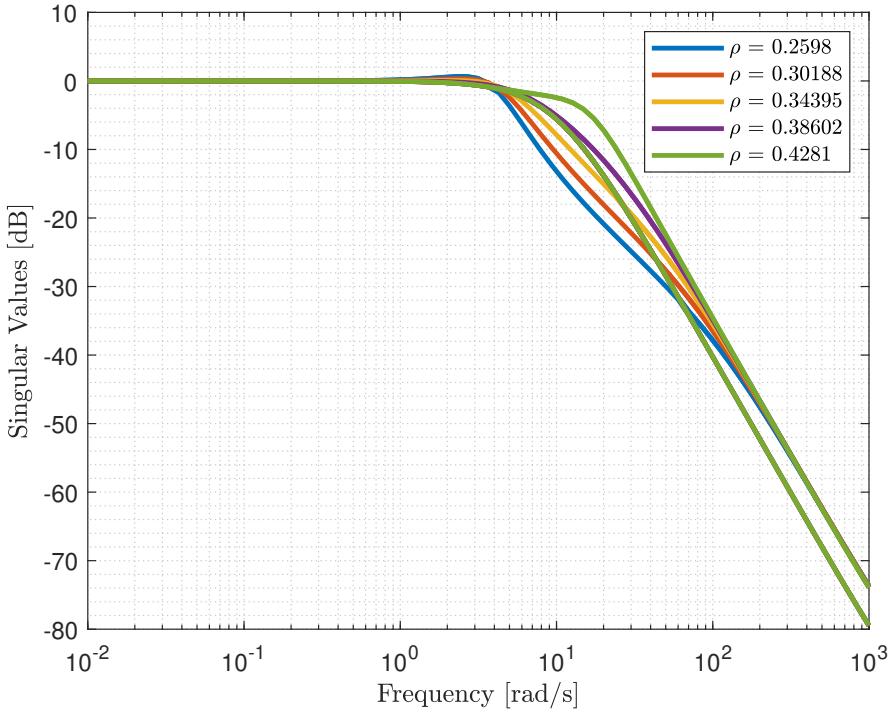

Figure 4.7: Singular values at different rod lengths

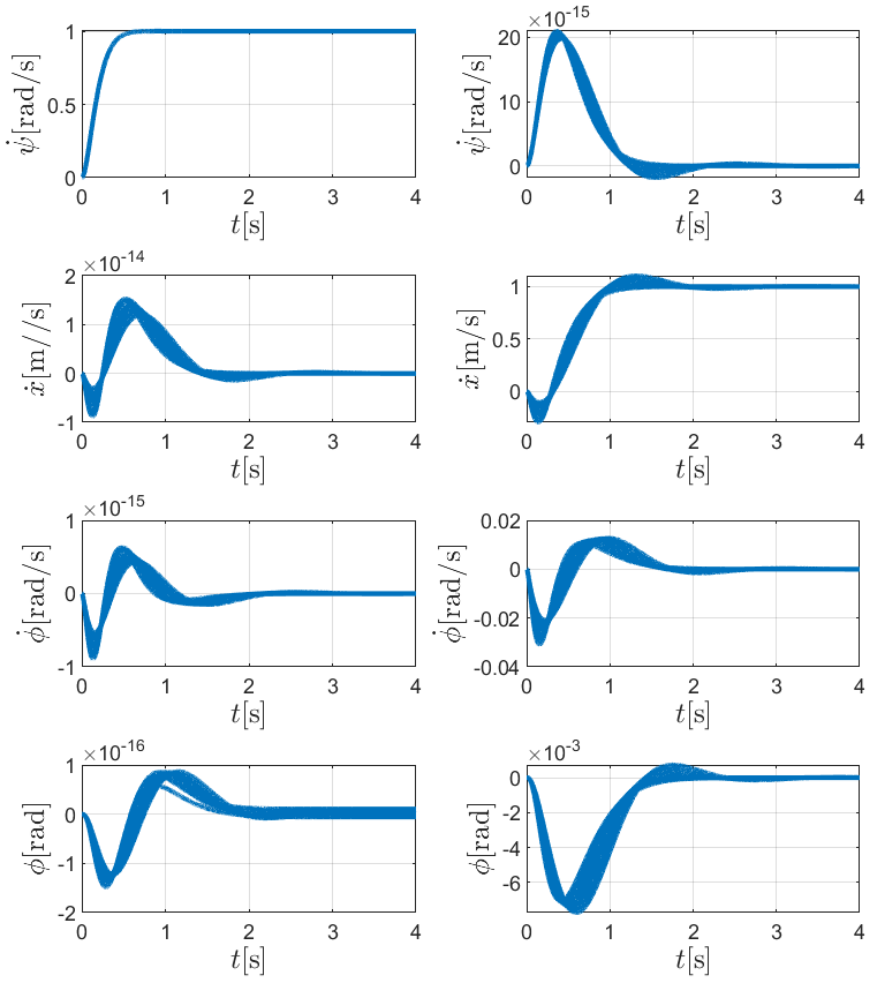

Figure 4.8: Responses to unit references

4.3 Nominal Model Search

This section aims to determine the nominal model for the robot. The following approach was used to derive the nominal value for the rod length, which represents the actual leg lengths of the real robot. First, the model was linearly sampled between a minimal and maximal possible value of rod length. These sampled models now serve as potential representations of the nominal model, denoted as G_0. For better visualization, only five possible nominal models were generated. Because testing these nominal models against a larger set of possible models is beneficial, another set of models G_i was generated. Then, the multiplicative (relative) uncertainty to each of the possible nominal models was applied

$$e(j\omega) = \max_{\sigma(G_i)\in\mathbb{S}} \left| \frac{G_i(j\omega) - G_0(j\omega)}{G_0(j\omega)} \right|. \tag{4.3.1}$$

The $e(j\omega)$ now forms the upper bound of the relative uncertainty for each nominal model. The example of the upper bound for one of the nominal models is shown in Figure 4.9. Then, the upper bound for each of the generated nominal models shows the Figure 4.10.

This figure shows that if we choose any of these models as nominal, there is no crossover frequency. So, one of the possible criteria to quantify the nominal model is the lowest $e(j\omega)$, specifically at low frequencies. This criterium was chosen, and it can be seen that the nominal model lies around $\rho_{\text{nominal}} = 0.33$. Also, it can be seen that the more the nominal value is further from the actual nominal value with the lowest uncertainty, the bigger the uncertainty is. Based on this observation, we can shrink the radius of the possible nominal models around the nominal model with the lowest uncertainty. This idea of shrinking the possible models can be done iteratively, and we can find the exact value of the nominal model with the lowest uncertainty. The result of the nominal model finding can be seen in Figure 4.11. Based on our earlier criteria, the figure suggests that the nominal model aligns with a rod length of $\rho_{\text{nominal}} = 0.32$ meters. Now, this rod length will serve as the nominal value for which the robust controller in the next sections will be designed.

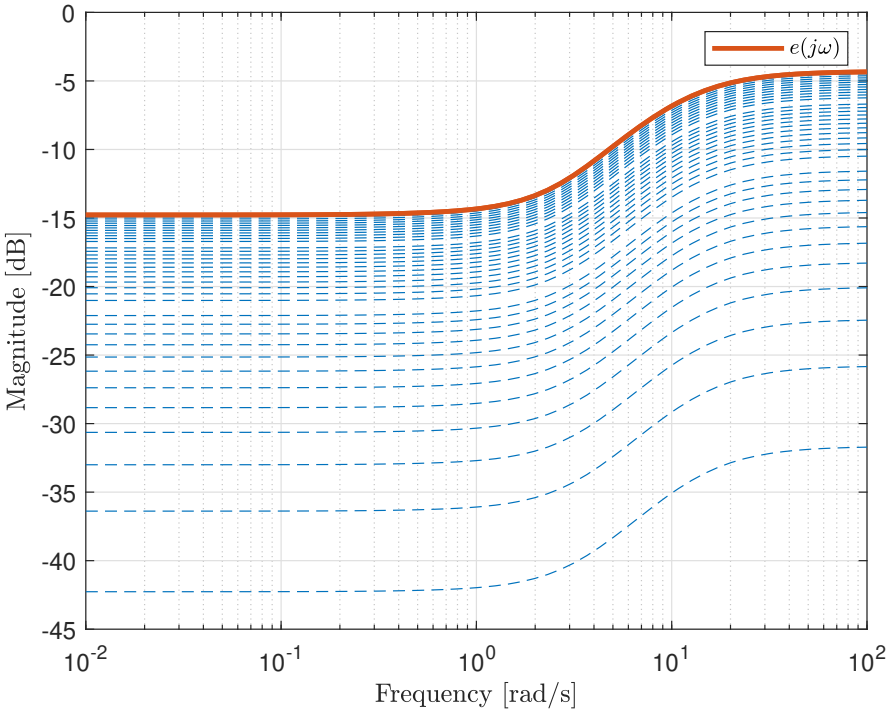

Figure 4.9: Example of found upper bound

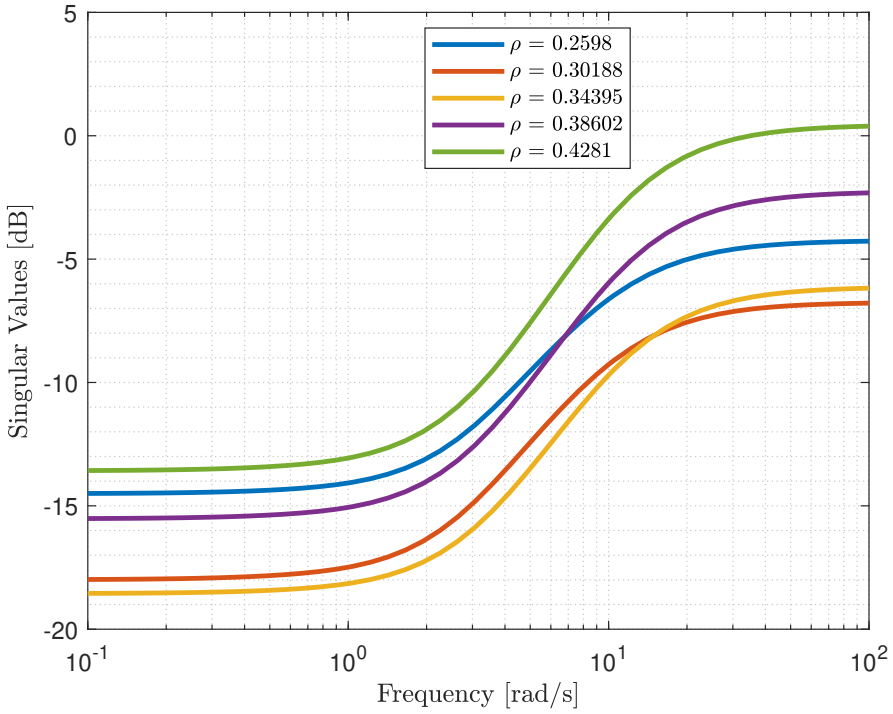

Figure 4.10: Upper bounds of uncertainty of possible nominal models

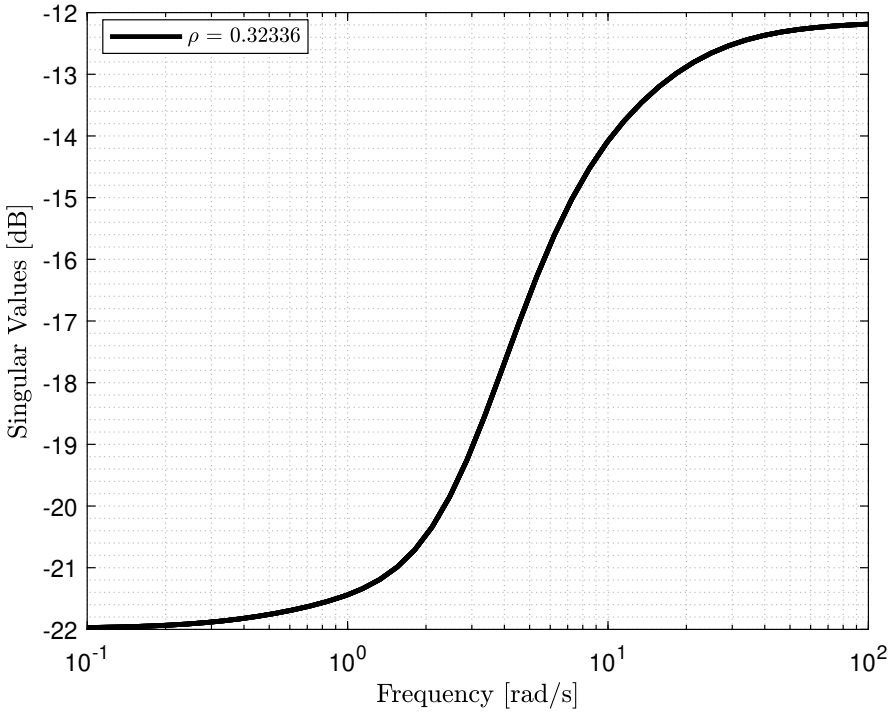

Figure 4.11: Nominal model found

CHAPTER 5

CONTROLLER DESIGN

In this chapter, two different types of controllers will be designed. One approach assumes that the rod length of the Segway-like model is constant, evaluated at a nominal value, while the second approach benefits from the estimation of the robot height. Initially, the robust controller, which cannot estimate the robot's height, will be designed. Then, gain scheduling will be discussed. The gain scheduling approach estimates the robot's height to update its gains based on this estimate in real-time.

5.1 Robust Controller

This type of controller will not benefit from estimating the robot's height. Instead, we will assume that the robot height is uncertain within some range. This approach will benefit from the analysis in a section about searching for the nominal model in section 4.3. Based on the criterion described in this section, it was found that the nominal model is for the robot height of $\rho_{\text{nominal}} = 0.32$ meters, which is a little under half of its maximum and minimum height. This section will describe the $\mathcal{H}_\infty$ Loop Shaping method, which consists of loop shaping and robust stabilization. In this design method, we switch to a continuous time system defined in the equation 3.1.2, because it is easier than in discrete time. Then, the controller is discretized. For the discrete-time case definition, see [12].

5.1.1 $\mathcal{H}_\infty$ loop-shaping

This design procedure is based on $\mathcal{H}_\infty$ robust stabilization, combined with the loop shaping method explained in [25] and originally proposed by McFarlane

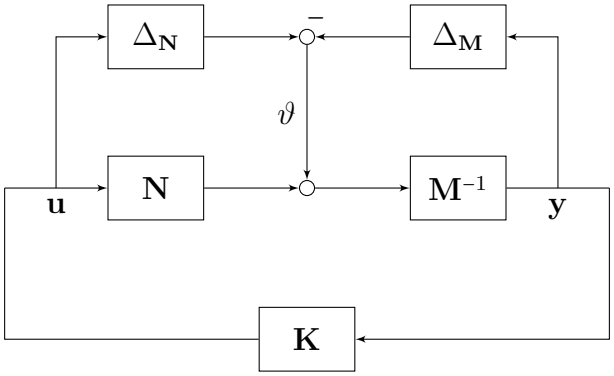

Figure 5.1: $\mathcal{H}_\infty$ robust stabilization

and Glover. This method can be divided into two stages. The first stage involves shaping the open-loop plant using pre-compensator $\mathbf{W}_1$ and post-compensator $\mathbf{W}_2$ to achieve the desired shape to the singular values of the open-loop frequency response. The second stage is concerned with robustly stabilizing this shaped plant. We note that robust stabilization can be done without shaping the plant. However, the shaping procedure is crucial because we want to get some system performance, so we will consider only this option.

Robust Stabilization

The shaped plant is robustly stabilized using $\mathcal{H}_\infty$ optimization. An advantage of this method is that we do not need to know these uncertainties explicitly. This approach directly addresses uncertainty using two stable perturbations in the normalized coprime factorization. This can be expressed in either its right or left form. Here, we will assume the left form, defined as follows

$$\mathbf{G} = \mathbf{M}^{-1}\mathbf{N}, \tag{5.1.1}$$

where $\mathbf{M}$ and $\mathbf{N}$ are stable transfer function. Then, the perturbed plant is defined as

$$\mathbf{G}_\mathrm{p} = (\mathbf{M} + \Delta_\mathbf{M})^{-1}(\mathbf{N} + \Delta_\mathbf{N}), \tag{5.1.2}$$

where the $\Delta_\mathbf{M}$ and $\Delta_\mathbf{N}$ are this time unknown stable transfer functions. These transfer functions represent the uncertainty in the nominal system. Then, the goal of robust stabilization is to stabilize the whole family of systems

$$\mathbf{G}_\mathrm{p} = \left\{(\mathbf{M} + \Delta_\mathbf{M})^{-1}(\mathbf{N} + \Delta_\mathbf{N}) : \left\|\begin{bmatrix}\Delta_\mathbf{M} & \Delta_\mathbf{N}\end{bmatrix}\right\|_\infty < \varepsilon\right\}, \tag{5.1.3}$$

using the dynamic controller feedback controller $\mathbf{K}$, where $\varepsilon > 0$ is stability margin. We can say that this perturbed system will be stable with the dynamic feedback controller $\mathbf{K}$ if the following holds

$$\left\| \begin{bmatrix} \mathbf{K} \\ \mathbf{I} \end{bmatrix} (\mathbf{I} - \mathbf{G}\mathbf{K})^{-1} \mathbf{M}^{-1} \right\|_{\infty} \leq \frac{1}{\varepsilon}. \tag{5.1.4}$$

The maximum achievable stability margin ε is given by [14] as

$$\varepsilon_{\max}^{-1} = \left(1 + \varrho(\mathbf{X}\mathbf{Z})\right)^{\frac{1}{2}}, \tag{5.1.5}$$

where $\varrho(\cdot)$ denotes the spectral radius, which means maximum eigenvalue. For the strictly proper minimal state space realization of the system $\mathbf{G}$, the $\mathbf{Z}$ and $\mathbf{X}$ are the positive, unique solutions of the following algebraic Riccati equations

$$\mathbf{A}\mathbf{Z} + \mathbf{Z}\mathbf{A}^{\top} - \mathbf{Z}\mathbf{C}^{\top}\mathbf{C}\mathbf{Z} + \mathbf{B}\mathbf{B}^{\top} = \mathbf{0}, \tag{5.1.6}$$

$$\mathbf{A}^{\top}\mathbf{X} + \mathbf{X}\mathbf{A} - \mathbf{X}\mathbf{B}\mathbf{B}^{\top}\mathbf{X} + \mathbf{C}^{\top}\mathbf{C} = \mathbf{0}. \tag{5.1.7}$$

Then, the controller $\mathbf{K}$ which guarantees 5.1.4 for specified $\varepsilon^{-1} < \varepsilon_{\max}^{-1}$ is given by

$$\mathbf{K}_s = \begin{bmatrix} \mathbf{A} + \mathbf{B}\mathbf{B}^{\top}\mathbf{X} + \varepsilon^{-2}\mathbf{L}^{-\top}\mathbf{Z}\mathbf{C}^{\top}(\mathbf{C} + \mathbf{D}\mathbf{B}^{\top}\mathbf{X}) & \varepsilon^{-2}\mathbf{L}^{-\top}\mathbf{Z}\mathbf{C}^{\top} \\ \mathbf{B}^{\top}\mathbf{X} & \mathbf{0} \end{bmatrix}, \tag{5.1.8}$$

where the matrix $\mathbf{L}$ is defined as

$$\mathbf{L} = (1 - \varepsilon^{-2})\mathbf{I} + \mathbf{X}\mathbf{Z}. \tag{5.1.9}$$

It is important to note that these equations hold only for strictly proper systems, which our system is. For the case where the system is not strictly proper (matrix $\mathbf{D}$ is not zero), the corresponding equations can be found in the [25].

Loop Shaping Design

As mentioned, the plant should first be shaped to change the system's performance. This can be done, for example, by using some already designed controller that ensures performance for the nominal model. During this thesis, there were some attempts to design controllers, which initially ensured some good performance, but these controllers were of high order, and the performance was farther from that described in 4.2.2. Another option is to use the classical loop shaping method, where we shape the singular values to ensure performance. Because we have no phase information, robust stabilization is ensured in the second stage as explained

in [15]. That is why we are just considering shaping the singular values of the open loop frequency response. Also, in [25], it was mentioned that in real applications, generally, little effort can be made by following some simple rules to ensure good performance.

A general approach was involved in this design procedure. Our requirements align with typical rules, seeking high gain at low frequencies to ensure good tracking and disturbance rejection while maintaining low gain at higher frequencies for effective noise attenuation. Also, the slope should be -1 at the bandwidth frequency. We will first consider the pre-compensator $\mathbf{W}_1$. Based on these rules, similarly as in [10], a Proportional-Integral (PI) controller was used and set to the diagonal of the pre-compensator $\mathbf{W}_1$ to ensure zero steady-state error. After several trials of tuning the PI controller parameters, the final structure of the pre-compensator $\mathbf{W}_1$ was determined to provide sufficient performance. The post-compensator $\mathbf{W}_2$ was selected as an identity matrix. The pre-compensator has the following structure

$$\mathbf{W_1} = \begin{bmatrix} \frac{1.5s+0.075}{s} & \\ & \frac{1.5s+0.075}{s} \end{bmatrix}, \tag{5.1.10}$$

and the shaped plant is then given as

$$\mathbf{G}_\mathrm{s} = \mathbf{W}_2\mathbf{G}\mathbf{W}_1. \tag{5.1.11}$$

Then, the robust stabilization was performed on this shaped plant, resulting in the controller $\mathbf{K}_\mathrm{s}$. The maximal stability margin achieved was $\varepsilon_{\max} = 0.33$, meaning the nominal system can take up 33% of coprime uncertainty. This is still higher than the minimum recommended in [25]. The shaped controller can be expressed for the plant $\mathbf{G}$ as

$$\mathbf{K} = \mathbf{W}_1\mathbf{K}_\mathrm{s}\mathbf{W}_2. \tag{5.1.12}$$

The shaped singular values of the frequency response, together with that robustly stabilized, are shown in Figure 5.2 We observe that at lower frequencies, the singular values are squeezed together. At higher frequencies, the gain decreases, and the slope is steeper. Additionally, the gain increases around the bandwidth. Figure 5.3. shows the resulting controller structure, where the $\mathbf{K}_\mathrm{s}(0)$ and $\mathbf{W}_2(0)$ are the dc gains. This controller structure was selected over the classical approach to avoid directly exciting the dynamics of $\mathbf{K}_\mathrm{s}$, which leads to the undesirable effects of the classical derivative kick [25]. On the other hand, we got the initial opposite kick in the responses on reference tracking. This is likely because the system is a non-minimum phase, and the controller $\mathbf{K}_\mathrm{s}$ is not directly excited with the change of reference.

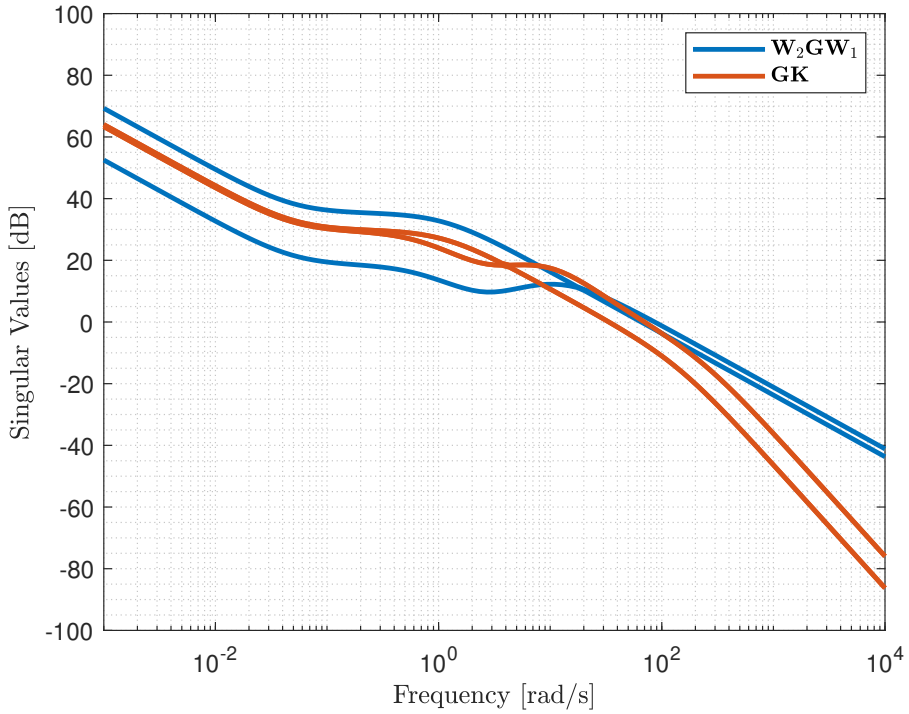

Figure 5.2: Initially shaped and robustified plant

Now, we switch back to the time domain, showing the responses to unit references. The responses are shown in Figure 5.4. We can see that with a comparison to unit responses in 4.2.2, the response to the velocity $\dot{x}_\mathrm{r}$ is a little bit slower than that for the already deployed controller, approximately 0.5 seconds slower. On the other hand, the response to $\dot{\psi}_\mathrm{r}$ is around 0.4 seconds faster. We can also observe the initial kicks, as was mentioned before. If we also compare the pitch rate $\dot{\phi}$ and pitch ϕ, the deviation from zeros is bigger than in the case of the already deployed controller. A better comparison will be seen in Chapter 6, where all developed controllers in this thesis will be compared with the already deployed one. Chapter 7 will then compare all controllers in a real robot.

Anti-windup

Another advantage of using this controller structure is incorporating the anti-windup. Because we used in the pre-compensator $\mathbf{W}_1$ the integrator, and the real system has some saturation, it is required to use anti-windup. The anti-windup can

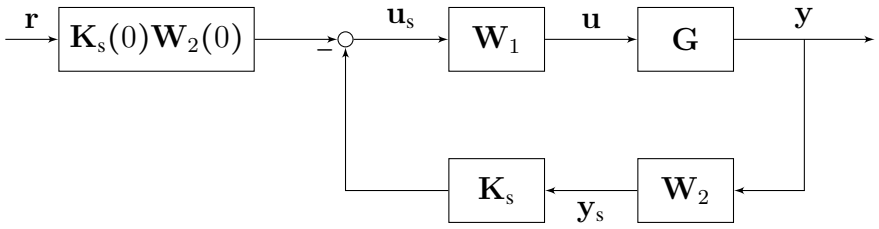

Figure 5.3: Controller structure

be implemented by introducing the pre-compensator with its minimal state-space representation $\mathbf{W}_1 = (\mathbf{A}_w, \mathbf{B}_w, \mathbf{C}_w, \mathbf{D}_w)$ in Hanus form [24] as

$$\mathbf{u} = \left[\begin{array}{c|cc} \mathbf{A}_w - \mathbf{B}_w\mathbf{D}_w^{-1}\mathbf{C}_w & \mathbf{0} & \mathbf{B}_w\mathbf{D}_w^{-1} \\ \hline \mathbf{C}_w & \mathbf{D}_w & \mathbf{0} \end{array}\right] \begin{bmatrix} \mathbf{u}_s \\ \mathbf{u}_a \end{bmatrix}, \tag{5.1.13}$$

where $\mathbf{u}_s$ is the output from the controller $\mathbf{K}_s$ and $\mathbf{u}_a$ represents the output from the actuator saturation. Every experiment made in Chapters 6 and 7 was performed with implemented anti-windup.

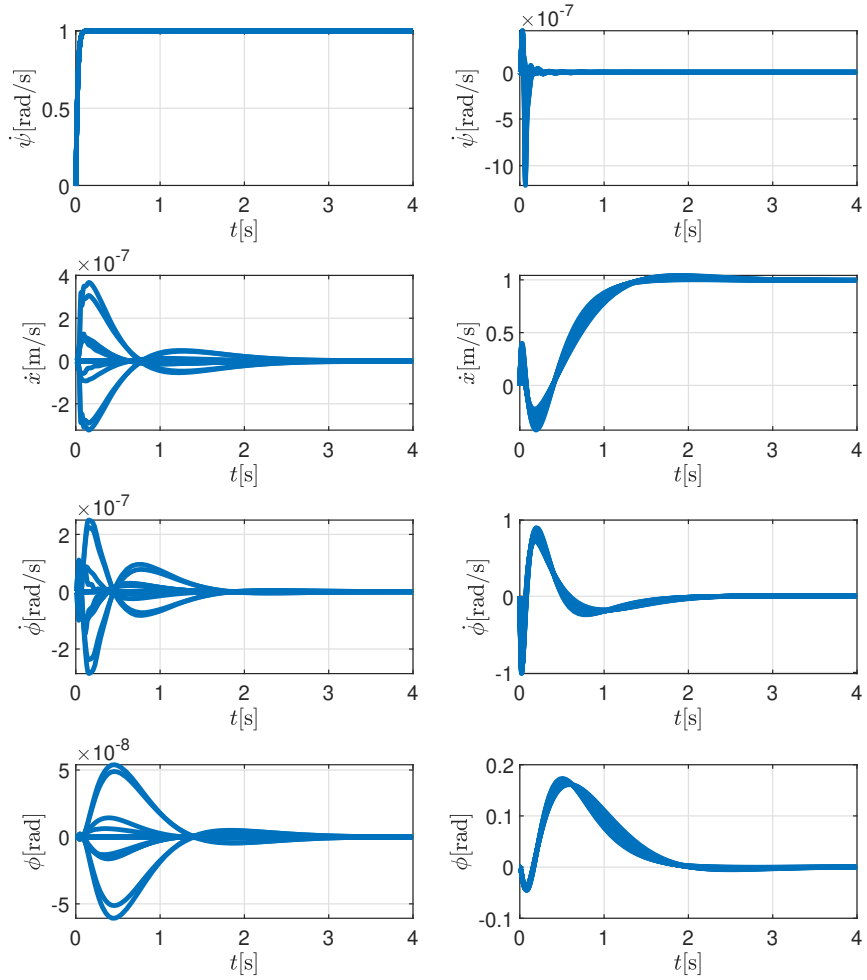

Figure 5.4: Responses to unit references

5.2 Gain Scheduled Controller

In this section, we explore the implementation of a gain-scheduled controller that leverages the measurement of the hip angles to estimate the robot's height. Gain scheduling is a control strategy that adapts its parameters, or gains, in response to changes in the system dynamics. This approach is particularly useful for

managing nonlinear systems whose dynamics vary across different operating conditions. The typical methodology involves linearizing the system around various operating points and designing a separate controller for each linearized model. These controllers are then implemented through a lookup table or by fitting the parameters (gains) to a spline function, allowing smooth transitions between different operating points. This enables the controller to maintain optimal performance across various conditions.

In section 3.2.2, it was demonstrated that rod length can be expressed using the hip angle, which is measurable. The objective was to design a controller that ensures consistent system dynamics across varying robot heights. To ensure this, the following approach was used. First, the discrete-time Linear Quadratic (LQ) controller was designed for a specific rod length. The used rod length was its nominal value obtained in 4.3. But it could be chosen any other.

5.2.1 LQ controller design

This controller design approach aims to determine the stabilizable state feedback gain $\mathbf{K}_i$ with integral action for tracking, which minimizes the following quadratic cost function.

$$J(\mathbf{z}, \mathbf{u}) = \sum_{k=0}^{\infty} \mathbf{z}_k^\top \mathbf{Q} \mathbf{z}_k + \mathbf{u}_k^\top \mathbf{R} \mathbf{u}_k \tag{5.2.1}$$

for a given linear discrete-time state space model

$$\mathbf{z}_k = \mathbf{A}_i \mathbf{z}_k + \mathbf{B}_i \mathbf{u}_k, \tag{5.2.2}$$

where $\mathbf{Q} = \mathbf{Q}^\top \succeq \mathbf{0}$ and $\mathbf{R} = \mathbf{R}^\top \succ \mathbf{0}$ are cost matrices for states and control inputs. The optimal solution to this problem can be achieved by solving the infinite time horizon LQ problem, expressed as

$$\mathbf{S} = \mathbf{Q} + \mathbf{A}_i^\top \mathbf{S} \mathbf{A}_i - (\mathbf{A}_i^\top \mathbf{S} \mathbf{B}_i)(\mathbf{R} + \mathbf{B}_i^\top \mathbf{S} \mathbf{B}_i)^{-1} (\mathbf{B}_i^\top \mathbf{S} \mathbf{A}_i), \tag{5.2.3}$$

where $\mathbf{S} = \mathbf{S}^\top \succeq \mathbf{0}$ represents the solution. This equation is known as the discrete-time Algebraic Riccati equation. The optimal state feedback gain matrix $\mathbf{K}_i$ is then given by

$$\mathbf{K}_i = (\mathbf{B}_i^\top \mathbf{S} \mathbf{B}_i + \mathbf{R})^{-1} (\mathbf{B}_i^\top \mathbf{S} \mathbf{A}_i). \tag{5.2.4}$$

The closed-loop system state matrix at this rod length is expressed as

$$\mathbf{A}_{cl} = (\mathbf{A}_i - \mathbf{B}_i \mathbf{K}_i) \tag{5.2.5}$$

5.2.2 Eigen structure assigment

To ensure stability and maintain identical responses across different rod lengths, one approach is to assign each rod length with the characteristics of the closed-loop matrix $\mathbf{A}_{cl}$. This effectively harmonizes the system's behavior regardless of the specific rod length. The closed-loop matrix $\mathbf{A}_{cl}$ exhibits similarity to its Jordan form $\mathbf{J}$. The closed-loop matrix can be likened to its Jordan form in the following manner

$$\mathbf{A}_{cl} = (\mathbf{A}_i - \mathbf{B}_i\mathbf{K}_i) \sim \mathbf{J}. \tag{5.2.6}$$

Utilizing the transformation matrix $\mathbf{X}$, the above similarity relation can be equivalently expressed as

$$(\mathbf{A}_i - \mathbf{B}_i\mathbf{K}) = \mathbf{X}\mathbf{J}\mathbf{X}^{-1}. \tag{5.2.7}$$

Upon rearranging terms, the equation transforms as in [23] into

$$\mathbf{A}_i\mathbf{X} - \mathbf{X}\mathbf{J} - \mathbf{B}_i\mathbf{K}_i\mathbf{X} = \mathbf{0}. \tag{5.2.8}$$

We can further parameterize the equation using the matrix $\mathbf{H}$, defined as

$$\mathbf{H} = \mathbf{K}_i\mathbf{X}. \tag{5.2.9}$$

Because we already know for the used nominal system, these matrice $\mathbf{K}$ and $\mathbf{X}$, the parametrization matrix $\mathbf{H}$ is known, and the equation 5.2.8 can be rewritten as follows

$$\mathbf{A}_i\mathbf{X} - \mathbf{X}\mathbf{J} - \mathbf{B}_i\mathbf{H} = \mathbf{0}, \tag{5.2.10}$$

which will result in the Sylvester equation. From [20], we can say that if the matrices $\mathbf{A}_i$ and $\mathbf{J}$ have no same eigenvalues, then the solution $\mathbf{X}$ exists. The controller is then finally retrieved as

$$\mathbf{K}_i = -\mathbf{H}^{-1}\mathbf{X}. \tag{5.2.11}$$

Based on this approach, we can assign the Jordan form $\mathbf{J}$ determined by the LQ controller for a nominal rod length to systems with different rod lengths.

After assigning the Jordan form to a set of grided rod lengths between its maximal and minimal possible value, we can see from Figure 5.5 that the gain dependence on the rod length is in the worst approximately quadratic. This is not surprising because section 4.1 shows that the system dependence on the rod length is approximately quadratic. Moreover, the eigenvalues have the property of changing only in one direction, also in the quadratic form. The quadratic gain dependence then holds for any nominal design. This fact is used, and the

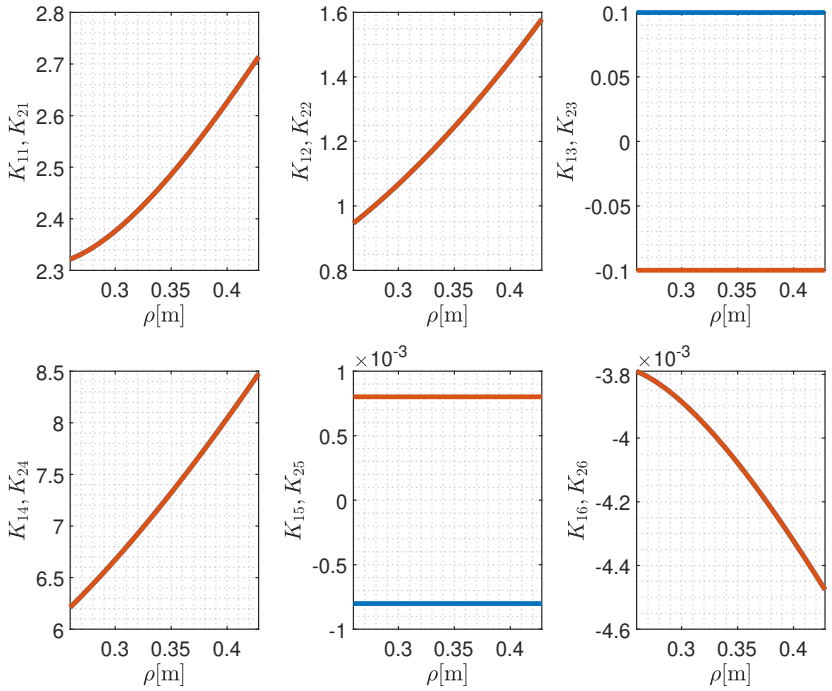

Figure 5.5: The dependence of individual gains on rod length

controller gains are fitted to a second-order polynomial. The final form of the controller is then

$$\mathbf{u} = -\mathbf{K}_{\mathrm{i}}(\rho)\mathbf{z}, \tag{5.2.12}$$

where (ρ) indicates parameter depence. Finally, the step response on the unit references of the yaw rate $\dot{\psi}_{\mathrm{ref}}$ and velocity $\dot{x}_{\mathrm{ref}}$ are in Figure 5.6. From this figure,

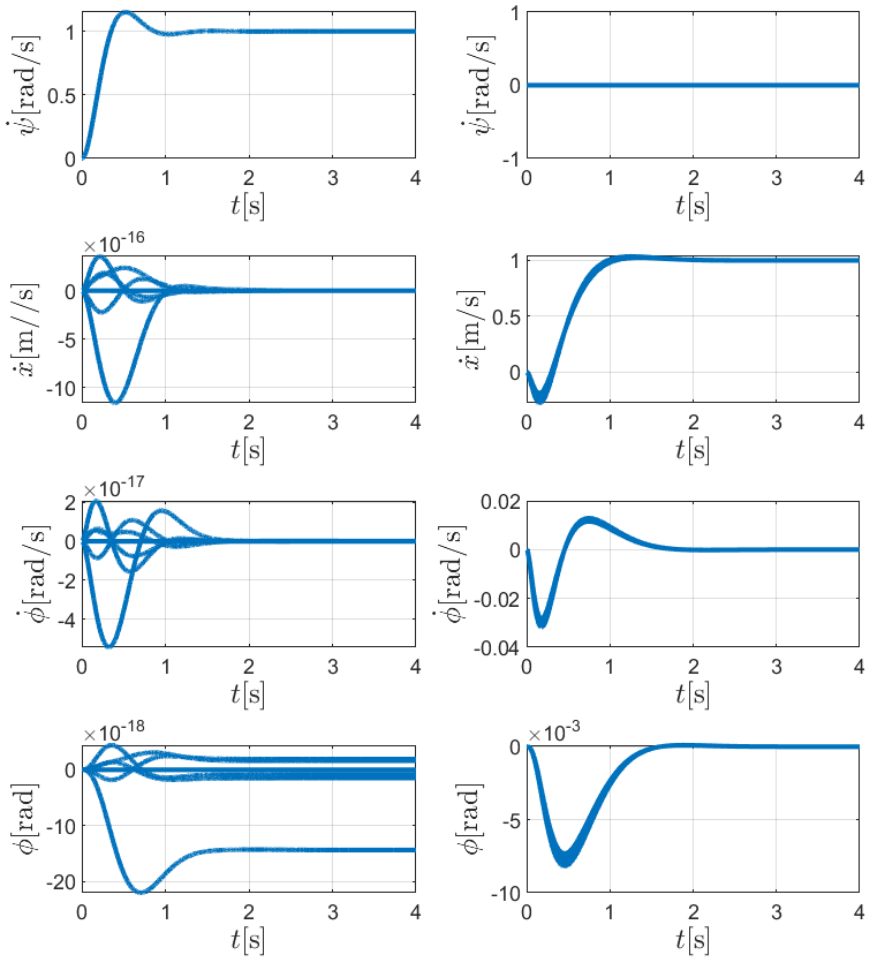

Figure 5.6: Response on the yaw rate reference

we can conclude that the response is almost the same at every rod length and

much or less the same as for the already deployed controller in the rod length for which it was designed. In the next two chapters, more detailed comparison will be performed. It is worth mentioning that this approach can be used for any already created state feedback controller if we know the gains of the matrix and the rod length for which the controller was designed.

CHAPTER 6

SIMULATION EXPERIMENTS

In this chapter, the experiments in the simulator will be performed. First, a comparison between old and newly developed controllers will be made. A simple balancing experiment will be done at different heights of the robot. Next, we generate a general trajectory tracking incorporating velocity and yaw rate tracking. Also, the robot height will change when tracking these references.

6.1 Balancing

The two developed controllers, gain scheduled and $\mathcal{H}_\infty$ controller, will be compared with the old LQ controller in these experiments. The comparison will be performed at three different heights of the robot, in its low, middle, and maximum height. The comparison will be made in terms of balancing performance. The robot behavior at its lowest height is in Figure 6.1. We can see that the robot has a similar balancing performance using the LQ and gain-scheduled controller at this height. The robot performs poorly with the $\mathcal{H}_\infty$ controller. A robot with this controller tends to do little balancing, whereas the robot is still in the simulation with the other controllers.

Following, we analyze the robot's behavior when both legs are in the middle position. The comparison of the robot's behavior in this leg configuration is shown in Figure 6.2. Again, we see the same behavior of the robot with LQ and gain-scheduled controller. This is not surprising since the initial design of the gain-scheduled controller was done at approximately this height. The same holds for the LQ controller. The robot balance is much or less the same for both controllers as for the lowest height. With the $\mathcal{H}_\infty$ controller, the robot behaves similarly as in the shortest legs configuration.

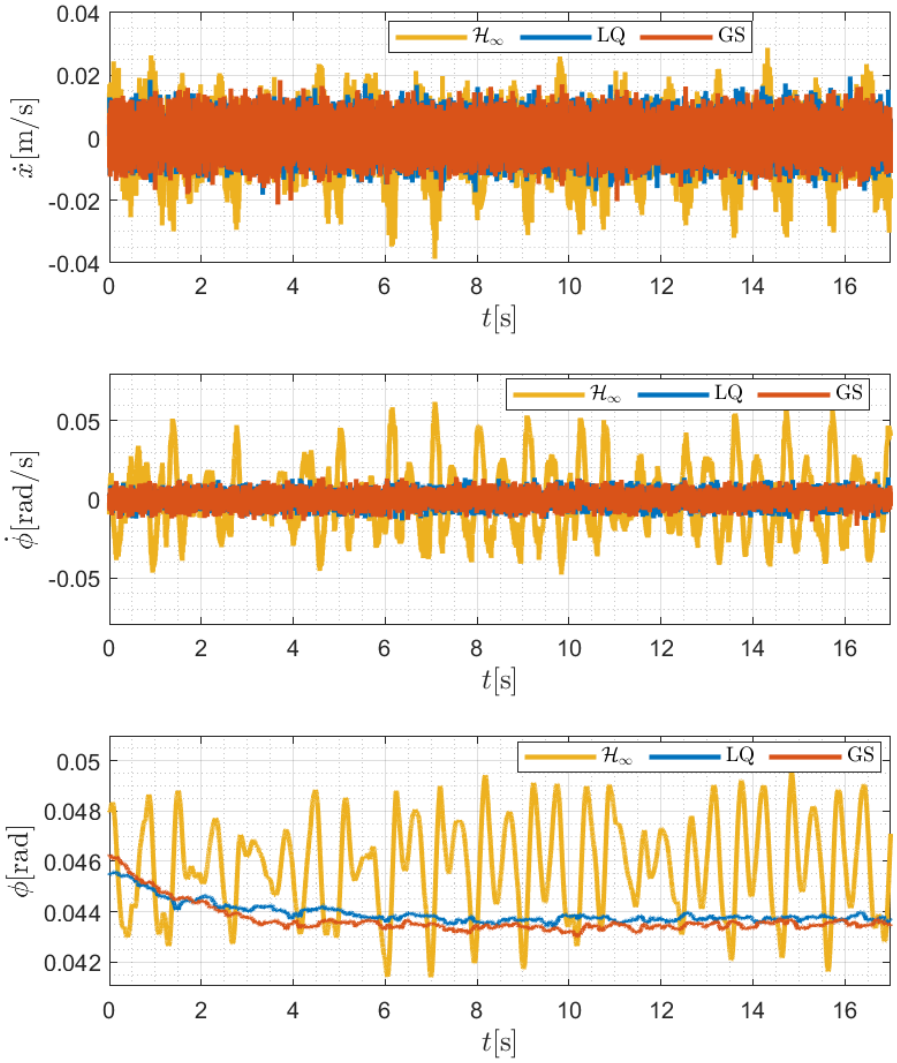

Figure 6.1: Balancing of the robot for shortest legs

Lastly, we will look at the robot's behavior for the stretched legs. This behavior is shown in the Figure 6.3. As expected, the robot with the LQ controller had the worst balancing performance. With the gain-scheduled controller, the robot behaved best. Again, the performance is similar to the other's height, which

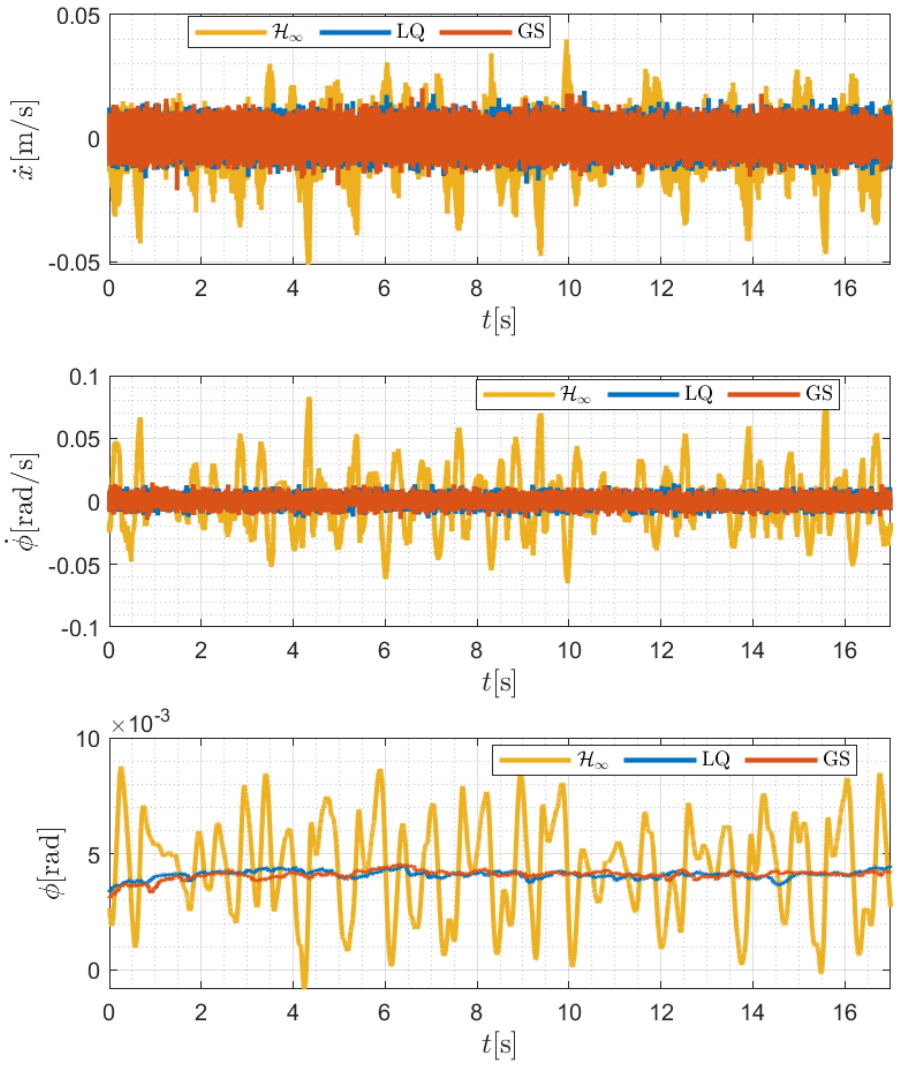

Figure 6.2: Balancing of the robot at mid-legs configuration

was wanted. This time, the $\mathcal{H}_\infty$ beats at least the LQ controller in balancing. Also, as for the gain-scheduled controller, the behavior with this controller is the same at each height.

Based on these three experiments, we can conclude that similar behaviors at

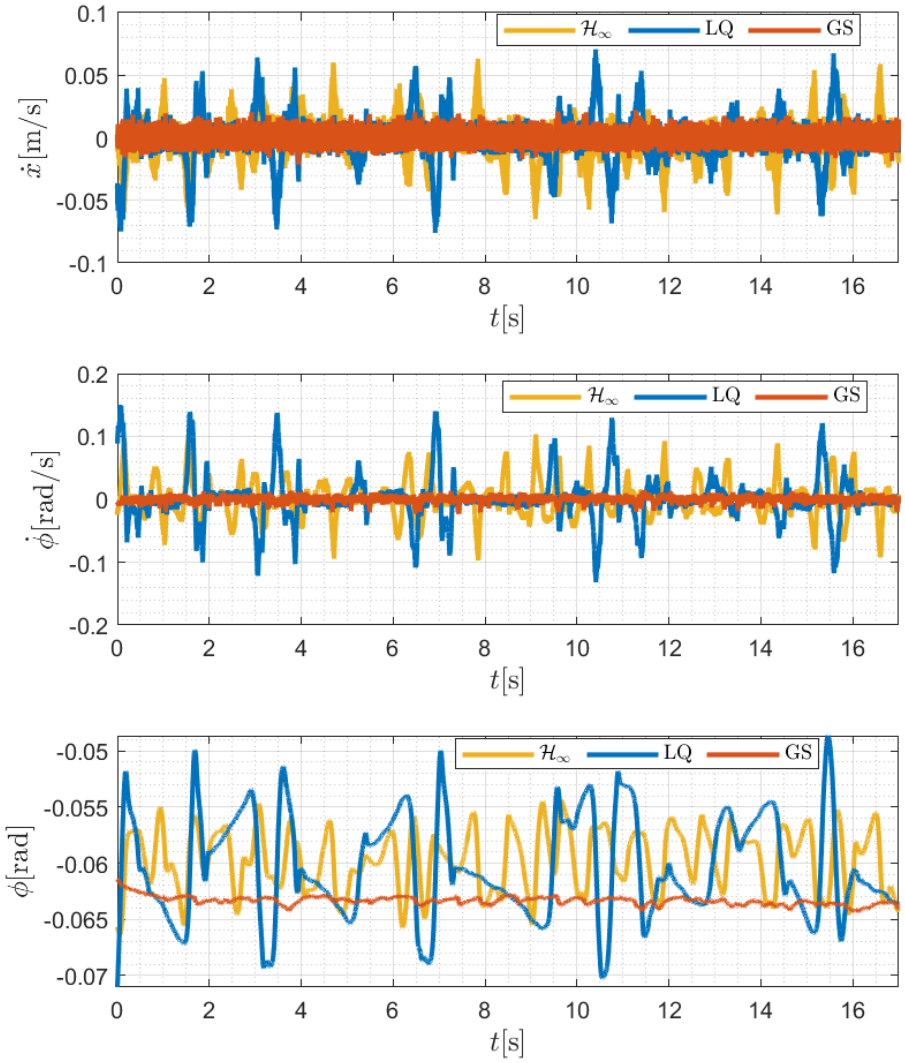

Figure 6.3: Balancing of the robot for the stretched legs

each height have gain-scheduled and $\mathcal{H}_\infty$ controllers. We especially wanted this behavior with the gain-scheduled controller. Although in simulations, the $\mathcal{H}_\infty$ produced worse behavior than the LQ controller, at least it kept consistent robot behavior at each height. In the end, we can say that some improvements were

made with these two newly developed controllers. We will see slightly different behaviors in Chapter 7.

6.2 Reference Tracking

Now, the performance of the newly developed controllers in terms of tracking will be evaluated. The tracking of the yaw rate $\dot{\psi}$ and velocity $\dot{x}$ during the robot's height change will be applied.

First, we will look at the robot's behavior with $\mathcal{H}_\infty$ controller. The responses, applied torques, and hip angles are shown in Figure 6.4. We can observe overshoots in the velocity $\dot{x}$ due to a change in the robot height during tracking the references. Some undershoots and overshoots in all measurements with the changes of references appeared. The yaw rate $\dot{\psi}$ is pretty fast because the yaw rate reference $\dot{\psi}_{\mathrm{ref}}$ is settled immediately. Looking at the actions, except for the reference changes, the actions are small. We can also see the right hip angle $\Phi_{1\mathrm{L}}$ and the left hip angle $\Phi_{1\mathrm{R}}$ changing over time. Some influence of the varying height is obvious.

If we switch to using the gain-scheduled controller, we can observe the tracking of references in Figure 6.5. We observe better velocity reference $\dot{x}_{\mathrm{ref}}$ tracking. The deviation of velocity $\dot{x}$ from its reference during the robot's height variation is insignificant. Also, the pitch rate $\dot{\psi}$ and pitch angle ϕ are not influenced by the varying height of the robot. The yaw rate $\dot{\psi}$ settles its reference approximately after half of a second with little overshoot. This is slower than using $\mathcal{H}_\infty$ controller.

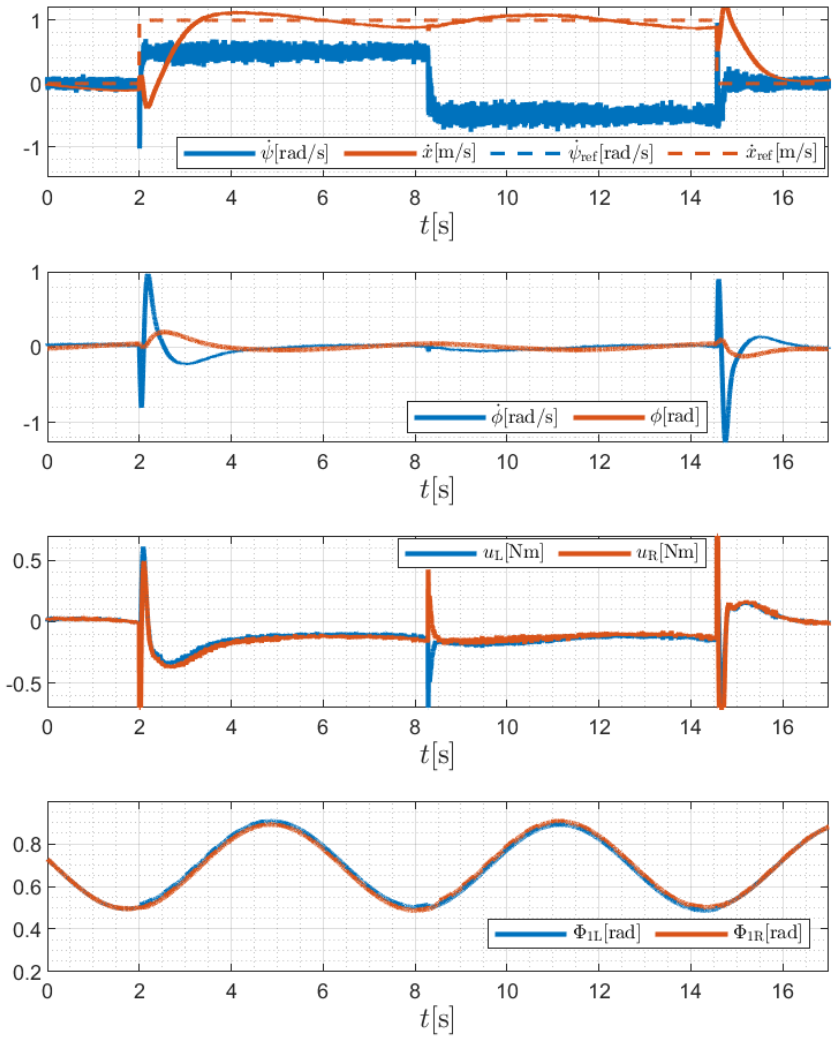

Figure 6.4: Reference tracking with varying height using $\mathcal{H}_\infty$ controller

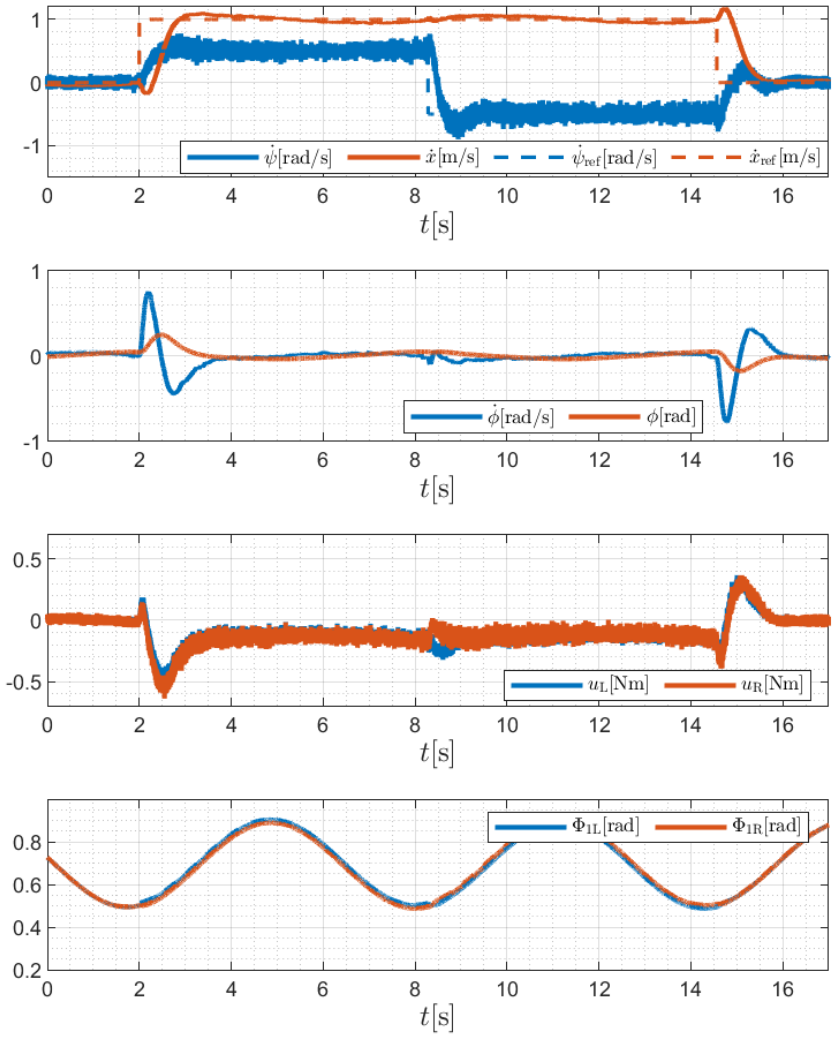

Figure 6.5: Velocity reference tracking with varying height using gain-scheduled controller

CHAPTER 7

REAL ROBOT EXPERIMENTS

In this chapter, the same experiments as in 3.2.2 will be performed. First, a simple balancing experiment will be performed at different heights. This experiment will demonstrate that the newly developed controllers can improve the system's behavior at different heights, especially in the extended legs where the robot for the original LQ controller oscillates. The next experiment will demonstrate that the robot with these two newly developed controllers can follow a simple trajectory during varying heights.

7.1 Balancing

This balancing experiment was performed on three different robot heights. Notably at its lowest height, in the middle, and at its maximum height. The comparison at the lowest height can be seen in the figure 7.1. We can see that the best behavior in terms of balancing has the $\mathcal{H}_\infty$ controller. There, we can see the pattern in the velocity $\dot{x}$, where to ensure a low pitch rate $\dot{\phi}$, the robot slightly moves forward and backward. The gain-scheduled controller holds that the robot behaves slightly worse than using the original LQ controller. This can be because the old controller has higher gains at this height (see the dependence of gains on the rod length in Figure 5.5).

Now, we compare the robot's behavior using different controllers at its middle height. This behavior is shown in Figure 7.2. Again, the best behavior was achieved with $\mathcal{H}_\infty$ controller. We can observe that the pitch rate $\dot{\phi}$ and pitch angle ϕ are closest to the origin of the whole experiment. The robot with a gain-scheduled controller has more or less the same behavior, which is unsurprising. The initial design for the gain-scheduled controller was made somewhere around

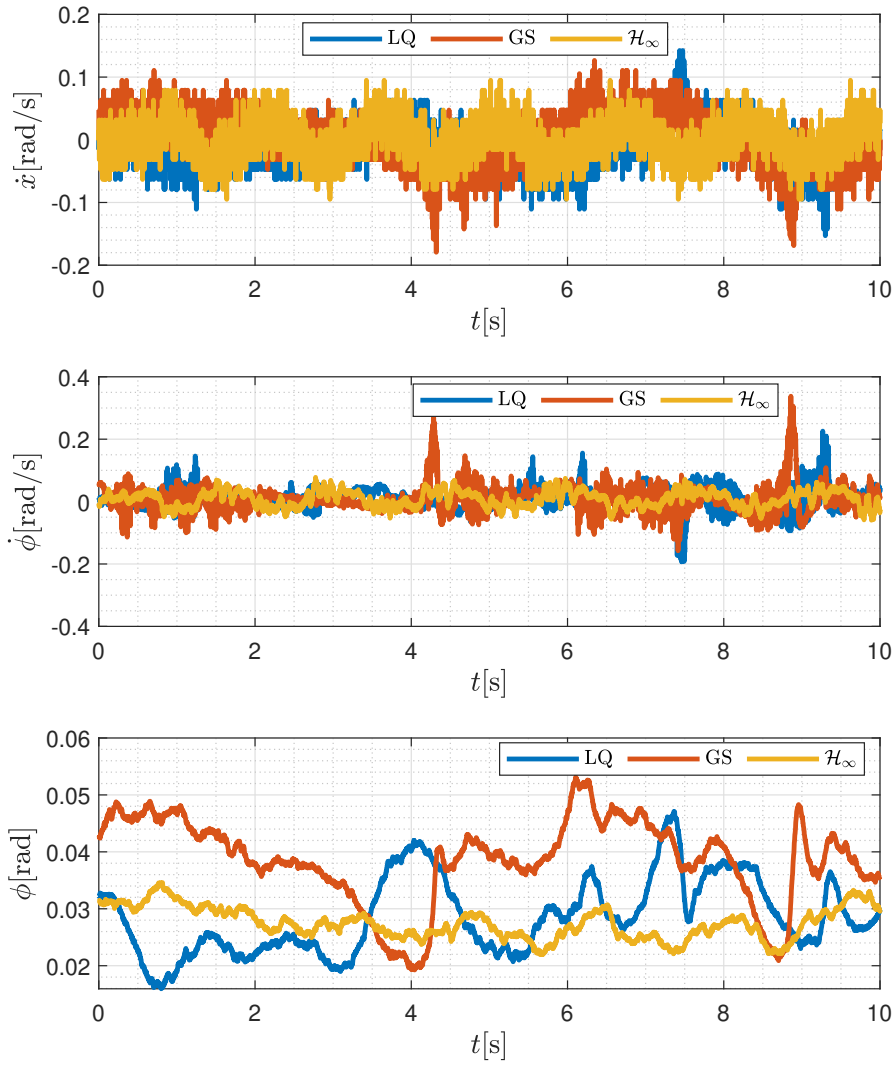

Figure 7.1: Balancing of the robot at its minimum height

this height. The same is applied to the LQ controller.

The last balancing experiment was made at its maximum height. There, we state that the robot oscillates for the LQ controller. Indeed, this fact is shown in Figure 7.3, where the robot's behavior with different controllers is also shown.

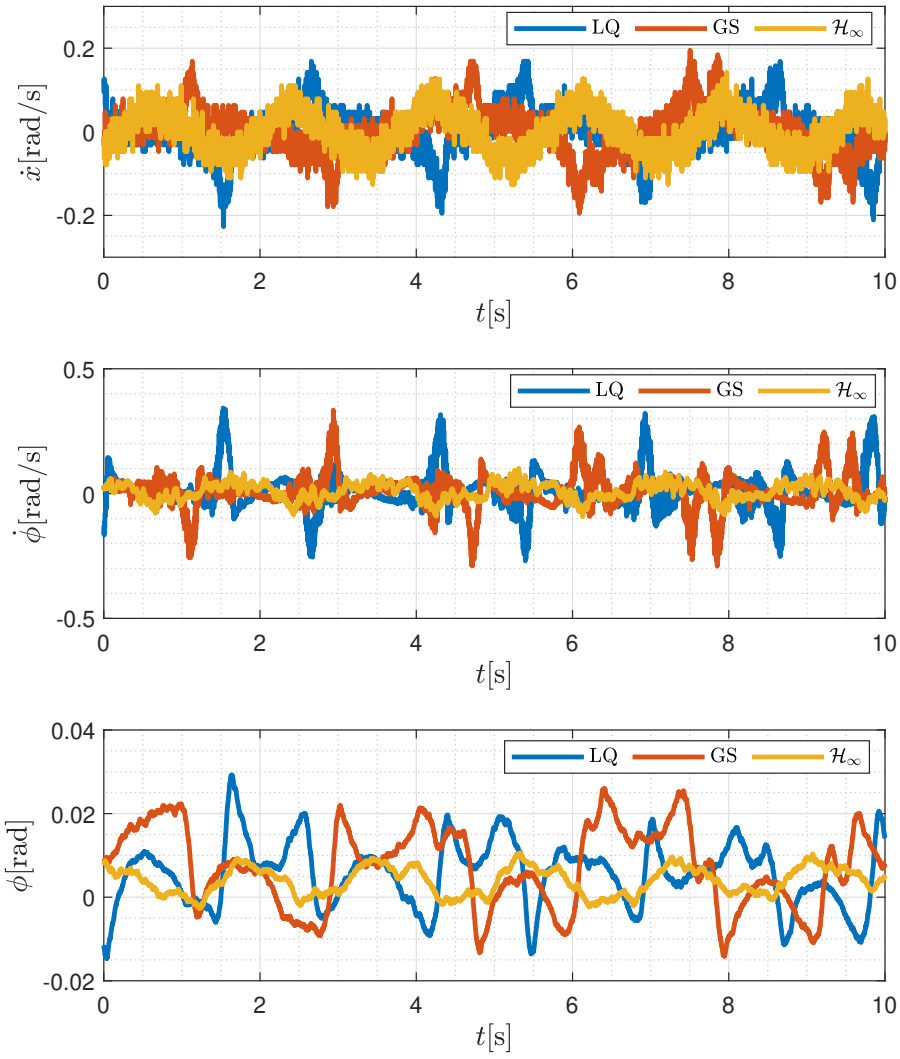

Figure 7.2: Balancing of the robot at its middle height

We can observe the improvement in the robot's behavior using newly developed controllers. Again, the best performance was achieved with $\mathcal{H}_\infty$ controller. The gain-scheduled controller improved the balancing behavior at this height from using the LQ controller. The behavior is again similar to other heights.

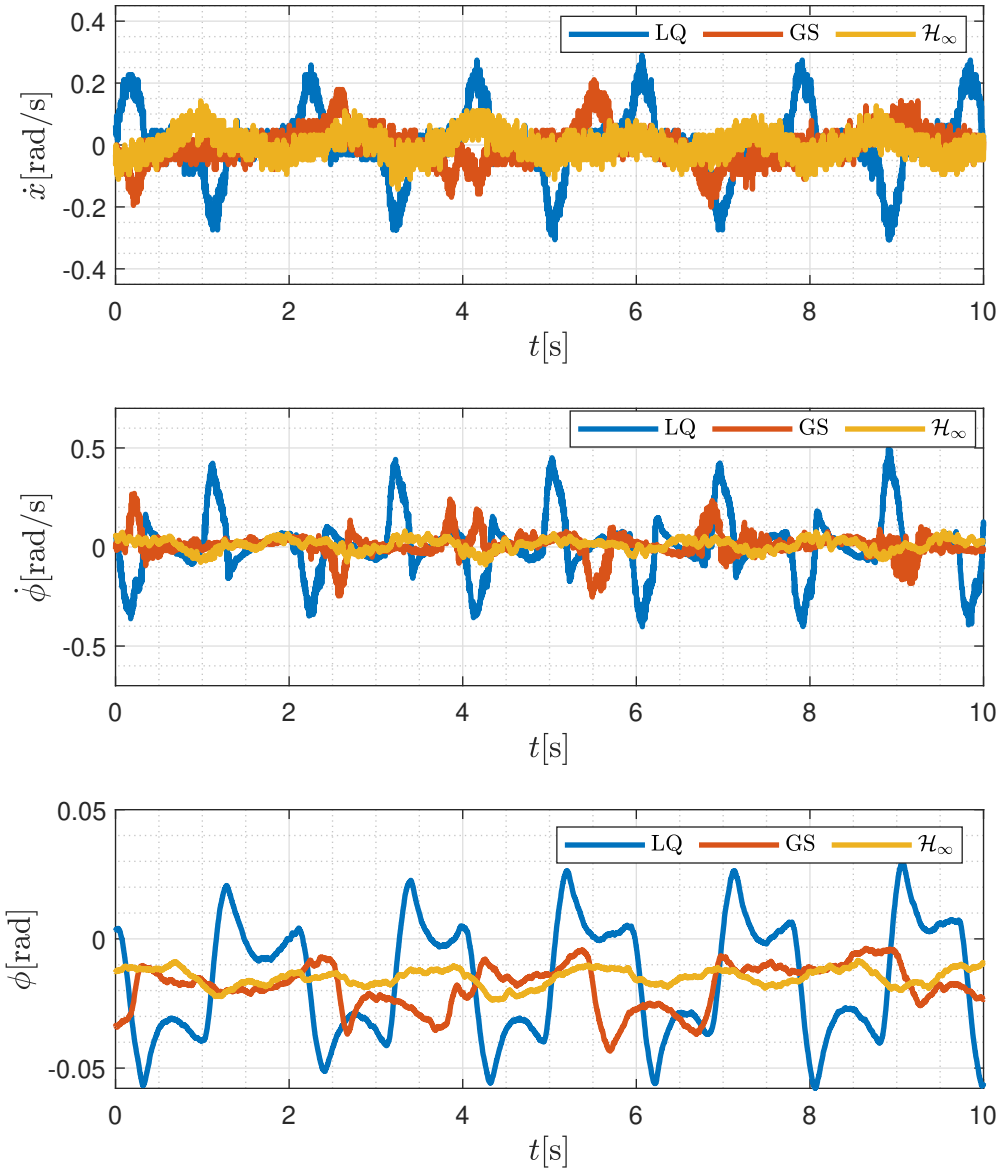

Figure 7.3: Balancing of the robot at its maximum height

We can conclude that in terms of balancing, the $\mathcal{H}_\infty$ controller beat both the LQ and the gain-scheduled controller. With this controller, the deviation from the origin was small, especially in pitch rate $\dot{\phi}$ and pitch angle ϕ. This result differs from in 6.1, where we saw that $\mathcal{H}_\infty$ controller does not produce the best behavior across different heights. This can be because the parameters in this simulator are not precise. Also, the contact forces modeled in the simulator cannot be very accurate.

7.2 Reference tracking

This section demonstrates that the robot with the newly developed controllers can track references while changing its height. A comparison with the original LQ controller was not made because the response with the gain-scheduled controller will be much or less the same.

We will look first at how well the robot could track references with $\mathcal{H}_\infty$ controller. The result of the experiment can be seen in the Figure 7.4. We can see that the velocity reference $\dot{x}_{\text{ref}}$ is settled after approximately one and a half seconds. The velocity $\dot{x}$ at 11 seconds goes slightly lower (the robot began to slow down). At this time, the hip angles were almost minimal (in robot measurement, a higher value means a lower hip angle, see 3.2.3). In the second half of references tracking, where the yaw rate reference $\dot{\psi}_{\text{ref}}$ change from 0.5 rad/s to -0.5 rad/s, the velocity was again settled and almost precisely tracked. We can observe the rough deviation from the origin in the pitch rate $\dot{\phi}$ with the velocity reference $\dot{x}_{\text{ref}}$ change. The system overreacts every time. This is probably due to how the controller is implemented (see controller structure described in 5.1.1). This overreaction can be seen in the left wheel action u_{L} and right wheel action u_{R}. The response to the yaw rate reference $\dot{\psi}_{\text{ref}}$ is settled almost immediately.

Now, we look at reference tracking using a gain-scheduled controller. The responses are in the Figure 7.5. Now we see the faster response to the velocity reference $\dot{x}_{\text{ref}}$, traced in one second. After settling the reference, velocity $\dot{x}$ remains at this reference. Unlike the previous $\mathcal{H}_\infty$ controller, we do not observe any overreaction. The response of the yaw rate $\dot{\psi}$ to its reference is slower than that with using $\mathcal{H}_\infty$ controller, but still pretty fast with settling time after 0.5 seconds. We can say that the varying heights of the robot during tracking do not have almost any influence on using this gain-scheduled controller.

If we should compare the gain-scheduled with $\mathcal{H}_\infty$ controller regarding reference tracking, we can conclude that velocity tracking is better with gain-scheduled controller. On the other hand, the yaw rate reference tracking is better with $\mathcal{H}_\infty$

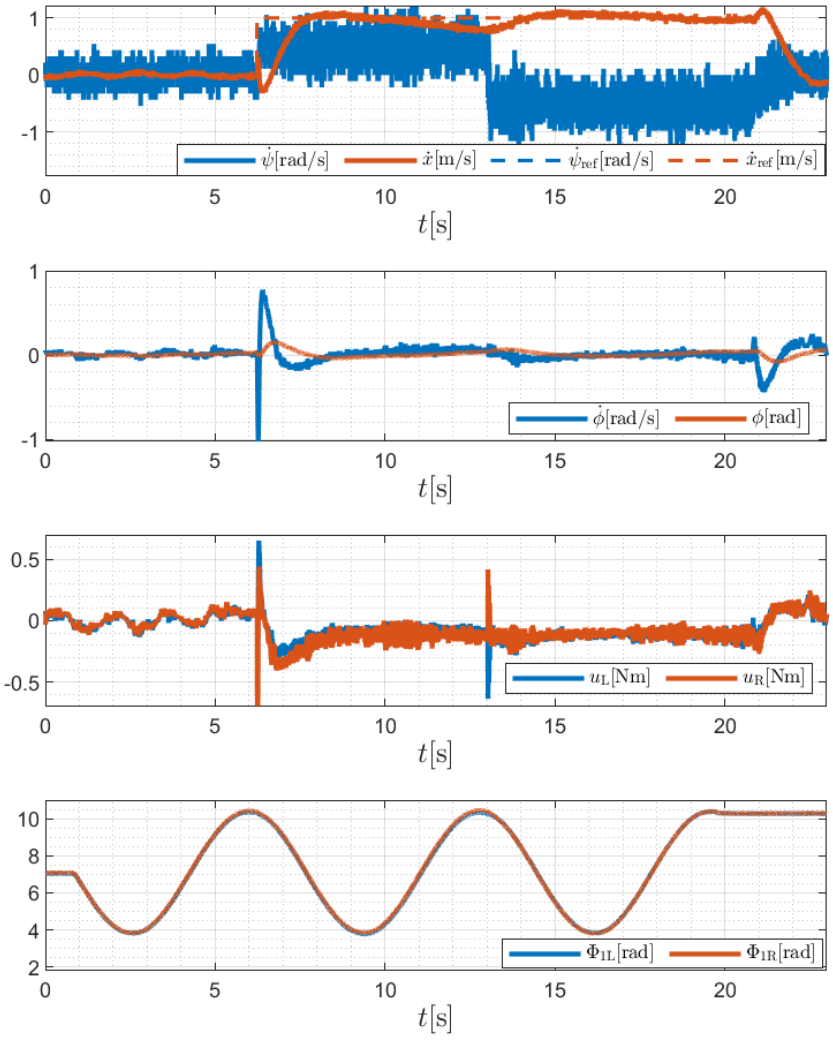

Figure 7.4: Reference tracking with varying height using $\mathcal{H}_\infty$ controller

controller. Also, the initial kick with reference change using $\mathcal{H}_\infty$ is not the most attractive.

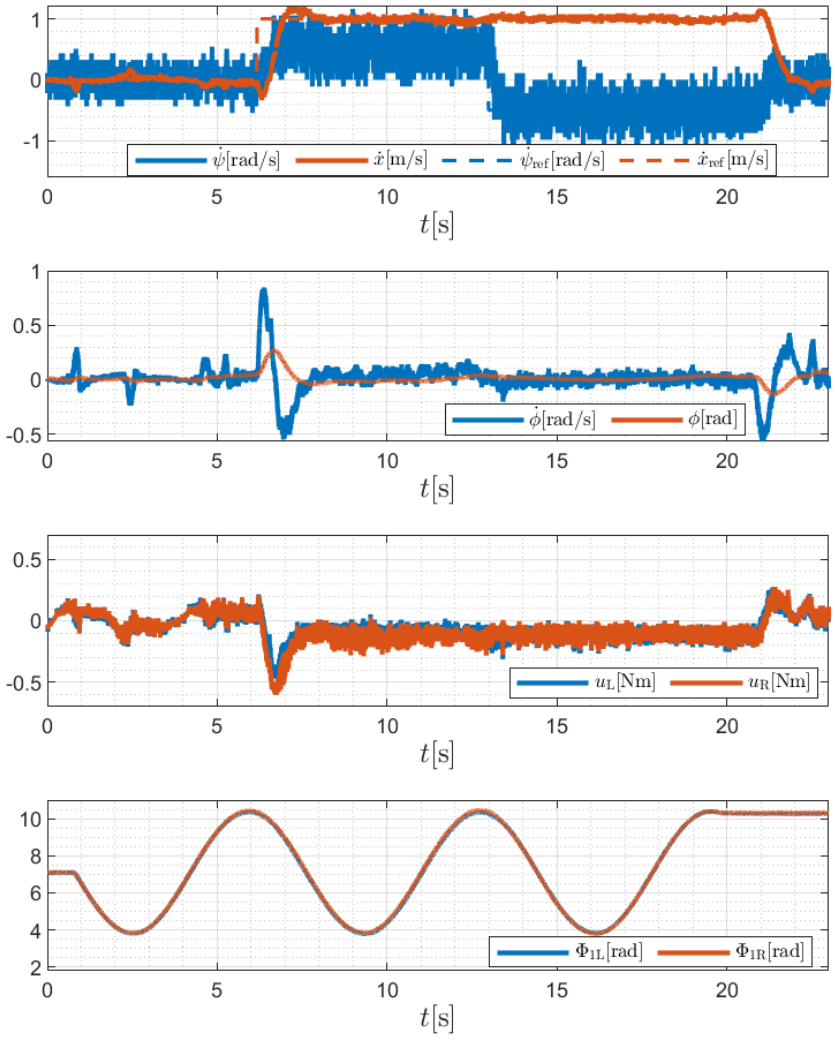

Figure 7.5: Reference tracking with varying height using gain-scheduling

7.3 Comparison with Simulator Experiments

In the comparison with the simulator, some changes in all experiments are observable. In section 6.1, we saw different results for the $\mathcal{H}_\infty$ controller as in

7.1. Unlike in the simulator, the reference tracking with varying heights had almost no effect in the real robot experiment with both controllers. Despite some efforts to specify the model parameters (for example, a moment of inertia of the body), which led to better similarity between the simulator and real robot, some differences are still observable. Based on the experiment, we can conclude that the simulator is imperfect. On the other hand, it is worth noting that every controller successfully tested in the simulator also functioned correctly on the real robot, which cannot be said for the simple control-oriented model. Ultimately, we can conclude that the simulator fulfilled its main purpose.

CHAPTER 8

CONCLUSION

The primary objective of this thesis was to design a controller that ensures optimal performance at various robot heights, beginning with an analysis based on a simplified model.

To achieve this objective, Chapter 3 introduces two models. The simplified model was used to analyze and design the controllers, while the high-fidelity model was employed to test the controllers before deploying them on the robot. Next, in Chapter 4 the optimal extension length for the robot's legs was found.

Based on this analysis, two types of controllers were designed in Chapter 5. One robust controller was designed for optimal robot legs and the second one benefits from measuring the corresponding robot's height.

In the following Chapter 6, the newly developed controller was compared with the old one in the simulator. Also, the robot's behavior with the new controllers at varying heights was observed. Subsequently, we saw in Chapter 7 that the best balancing performance in a real robot was achieved with $\mathcal{H}_\infty$ controller. On the other hand, tracking performance at varying heights was achieved with the gain-scheduled controller.

In conclusion, we can say that better performance than the old controller was achieved. Especially at the height where the old one has the worst performance.

In recent years, whole-body control has gained lots of popularity. One example is a similar robot Ascento mentioned at the beginning. The next step would be to design a full rigid body model and perform whole-body control. In [19], they achieved better performance by using whole-body control for the Ascento robot.

BIBLIOGRAPHY

[1] 6-DOF Joint. https://www.mathworks.com/help/sm/ref/6dofjoint.html.

[2] Ascento. https://www.ascento.ai/.

[3] MATLAB Simscape. https://www.mathworks.com/products/simscape.html.

[4] Model Template. https://www.mathworks.com/help/sm/gs/open-the-simscape-multibody-model-template.html.

[5] Norms and Singular Values. https://www.mathworks.com/help/robust/gs/norms-and-singular-values.html.

[6] Revolute Joint. https://www.mathworks.com/help/sm/ref/revolutejoint.html.

[7] Spatial Contact Force. https://www.mathworks.com/help/sm/ref/spatialcontactforce.html.

[8] Transform Sensor. https://www.mathworks.com/help/sm/ref/transformsensor.html.

[9] Kollarčík Adam. Modeling and control of two-legged wheeled robot. Diploma thesis, Czech Technical University in Prague, 2021.

[10] Bo Bernharsson and Karl Johan Åström. Robust Control, $\mathcal{H}_\infty$, ν and Glover-McFarlane, 2016.

[11] Ryan James Caverly and James Richard Forbes. Lmi properties and applications in systems, stability, and control theory, 2021.

[12] Mihail M Konstantinov Da-Wei Gu, Petko H. Petkov. *Robust Control Design with MATLAB*. 2013.

[13] Hodan Dominik. Reinforcement learning-based control system for the sk8o robot. Diploma thesis, Czech Technical University in Prague, 2023.

[14] Keith Glover Duncan C. McFarlane. *Robust Controller Design Using Normalized Coprime Factor Plant Descriptions*. 1990.

[15] K. Glover, J. Sefton, and D.C. Mcfarlane. A tutorial on loop shaping using h-infinity robust stabilization. *IFAC Proceedings Volumes*, 23(8, Part 3):117–126, 1990. 11th IFAC World Congress on Automatic Control, Tallinn, 1990 - Volume 3, Tallinn, Finland.

[16] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `https://cvxr.com/cvx`, March 2014.

[17] The MathWorks Inc. Matlab version: 23.2.0.2391609 (r2023b) update 2, 2023.

[18] Sangtae Kim and SangJoo Kwon. Dynamic modeling of a two-wheeled inverted pendulum balancing mobile robot. *International Journal of Control, Automation and Systems*, 13, 08 2015.

[19] Victor Klemm, Alessandro Morra, Lionel Gulich, Dominik Mannhart, David Rohr, Mina Kamel, Yvain de Viragh, and Roland Siegwart. LQR-Assisted Whole-Body Control of a Wheeled Bipedal Robot With Kinematic Loops. *IEEE Robotics and Automation Letters*, 5(2).

[20] Jana Königsmarková and Milos Schlegel. Parametric jordan form assignment by state-derivative feedback. pages 19–24, 06 2015.

[21] Brož Petr. Predictive control for the sk8o robot. Diploma thesis, Czech Technical University in Prague, 2024.

[22] ETH Zurich Robotic Systems Lab. *Robot Dynamics Lecture Notes*. 2017.

[23] Miloš Schlegel and Jana Königsmarková. Parametric jordan form assignment revisited. *Asian Journal of Control*, 16(2):409–420, 2014.

[24] Paul Schwerdtner, Scott Bortoff, Claus Danielson, and Stefano Di Cairano. Projection-based anti-windup for multivariable control with heat pump application. 06 2019.

[25] Sigur Skogestad and Ian Postlethwaite. *Multivariable Feedback Control: Analysis and Design, Second Edition*. 2005.

[26] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.