STRONGLY STABILIZING CONTROLLERS FOR A TWO-LINK ROBOTIC SYSTEM

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF

MASTER OF SCIENCE

IN

ELECTRICAL AND ELECTRONICS ENGINEERING

By

Adile Merve Andaç

January 2024

Strongly Stabilizing Controllers For A Two-Link Robotic System
By Adile Merve Andaç
January 2024

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Hitay Özbay(Advisor)

Ömer Morgül

İsmail Uyanık

Approved for the Graduate School of Engineering and Science:

Orhan Arıkan
Director of the Graduate School

ABSTRACT

STRONGLY STABILIZING CONTROLLERS FOR A TWO-LINK ROBOTIC SYSTEM

Adile Merve Andaç
M.S. in Electrical and Electronics Engineering
Advisor: Hitay Özbay
January 2024

Strong stabilization is defined as finding a stable controller that stabilizes the feedback system for a given plant. This study addresses the strong stabilization of a robotic system called the "acrobot", which is a two-linked underactuated planar robot system. The linearized system is fourth-order, with two poles and one zero in the right half-plane. In this thesis, stable second-order controllers designed with different methods have been investigated for this system. The stability margins are analyzed with respect to various free parameters. In addition, the time-domain transient response analysis is illustrated through simulations. Furthermore, the effects of nonlinearities are studied by estimating the region of attraction for each linear controller considered.

Keywords: Strong stabilization, stability margins, acrobot, underactuated robotics, second order controller, reduced order controller.

ÖZET

İKİ BAĞLANTILI ROBOTİK SİSTEMLER İÇİN GÜÇLÜ STABİLİZASYON SAĞLAYAN KONTROLCÜ TASARIMI

Adile Merve Andaç
Elektrik ve Elektronik Mühendisliği, Yüksek Lisans
Tez Danışmanı: Hitay Özbay
Ocak 2024

Güçlü stabilizasyon, verilen bir tesis (plant) için geri besleme sistemini kararlı hale getiren kendisi de kararlı olan kontrolcüler bulmak olarak tanımlanır. Bu çalışmada iki bağlantılı "acrobot" adı verilen bir robotik sisteminin güçlü stabilizasyonu ele alınmaktadır. Doğrusallaştırılmış sistem dördüncü derecedendir, iki adet kutubu ve bir adet sıfırı sağ düzlemdedir. Tezde, bu sistem için farklı yöntemlerle tasarlanan ikinci dereceden kararlı denetleyiciler incelenmiştir. Farklı serbest değişkenlerin kararlılık marjlarına etkileri gösterilmiştir. Bununla beraber, simülasyonlar yardımıyla zaman bazlı geçici tepki analizleri yapılmıştır. Ayrıca, üzerinde çalışılan her bir lineer denetleyici için çekim bölgeleri hesaplanarak doğrusal olmayan etkiler incelenmiştir.

Anahtar sözcükler: Güçlü stabilization, kararlılık marjları, akrobot, eksik eyleyicili robotlar, ikinci dereceden kontrolcüler, indirgenmiş kontrolcüler.

Acknowledgement

I would like to express my sincere appreciation to Prof. Hitay Özbay, my supervisor, for igniting my passion for systems and control theory. His patience, guidance, suggestions, and support throughout my Master's Degree were invaluable.

I am grateful to Prof. Ömer Morgül and Assoc. Prof. İsmail Uyanık for agreeing to be the jury members for my M.Sc. thesis defense and for their valuable feedback.

I would like to extend my special thanks to Emre Geyik for our productive and inspiring conversations. Together, we have accomplished a great deal. I would also like to express my gratitude to Caner Odabaş and Veysel Yücetürk for being supportive friends and colleagues who have helped pave the way for me during my M.Sc. journey.

I would also like to acknowledge the exceptional teaching quality of the Bilkent University Electrical and Electronics Engineering Department and the opportunity provided by ASELSAN to complete my M.Sc. degree. Additionally, I would like to thank my colleagues at ASELSAN for their understanding and cooperation.

Lastly, I am deeply grateful to my parents, Bülent and Nuray, and my brother, Batuhan, for their unwavering love and endless support throughout my life. I consider myself incredibly fortunate to be part of such a wonderful family.

To my parents and my brother, with love and gratitude ...

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Underactuated mechanical systems are defined as systems with fewer actuators than the degrees of freedom that systems possess [1]. Such systems can be found in the rapid and unconscious movements of the human body. These systems can achieve the desired motions more efficiently than may be done with proper actuation [2]. This paper aims to design a strong and reduced-order controller for underactuated mechanical systems because of the convenience and cost of implementation.

1.1 Motivation

In this thesis, strong stabilization applications on underactuated robots with two links are studied. Strong stabilization is defined as finding stable controllers that stabilize the feedback systems for a given plant. In recent decades, numerous research studies have shown how these stable controllers are essential in application domains [3]. Strong stabilization provides less sensitivity to the other unexpected effects on the system and also provides more predictable reactions from the system [4]. This predictability brings greater safety and robustness to the system [5,6].

When the feedback mechanism of a closed-loop system is disrupted, unstable controllers may cause unstable responses due to open-loop instability. In Figure 1.1; r, v and P denote reference, disturbance, and plant, respectively. Finally, C represents the transfer function of the stable controller to be designed.

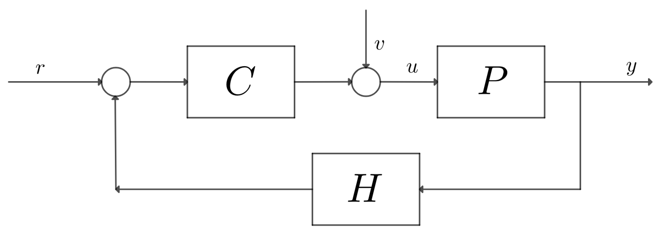

Figure 1.1: Unity Feedback Control System Blok Diagram: $H = 1$ sensor operational, $H = 0$ sensor inactive

Underactuated mechanical systems include a subset of open-loop, two-linked planar robots where one of the joints is not actuated, which makes the control problem more challenging. *"Acrobot"* and *"pendubot"* systems are principal examples of open-loop two-link mechanisms, both of which have only a single actuator to manipulate the system. Since the degree of freedom of these systems, i.e., the number of defining positional parameters is two, manipulation with a single actuator results in underactuation. Control strategies for such a system require feedback stabilization, as a generalized underactuated two-link robot system is both unstable and nonlinear within broader operating regions distant from equilibrium points. Such systems have a wide possibility of applications in recently emerging robotic fields and can be used in various robotic systems that require demanding dynamic properties. For example, in assistant robot applications where the motions of various animals are imitated, Acrobot modeling for active limbs is used. Hence, achieving better control performance becomes essential.

This study focuses on the concept of strong stabilization, which involves the identification of stable controllers capable of stabilizing the feedback system for a specified plant. The specific robotic system under consideration is the "acrobot", a two-linked underactuated planar robot. The linearized representation of this system is fourth-order, featuring two poles and one zero located in the right half-plane. Furthermore, the relative degree of the linearized plant transfer function is two (that means the system has two zeros at $+\infty$).

Many controller models, such as linear quadratic Gaussian and H_∞ controllers, have the same dimension as the plant model. If the plant is in high order, the controller is too. However, higher-order controllers are challenging to implement. On the other hand, their test and debug process is also inconvenient, and higher-order controllers are expensive. Due to these lucubratory conditions, reduced-order controllers are preferable [7].

The investigation in this study revolves around second-order stable controllers designed for the acrobot system. The analysis delves into examining the stability margins associated with different free parameters, settling time, and their respective impacts on the positions of the closed-loop poles. Through this exploration, the study aims to contribute insights into the strong stabilization of the acrobot and enhance our understanding of control strategies for underactuated planar robots with two links.

In terms of problem definition, the swing-up control problem could be an example in which the robot is to be moved from its stable downward position to its upright equilibrium position and balanced about its vertical axis [8].

1.2 Related Work

Controllers that provide strong stabilization for different plant models can be designed using various methods [3,5,6,9–21]. A recent article that comprehensively reviews the literature in this regard is [22].

In [22], it is argued that a minimum-phase controller is a possible method to obtain a strongly stabilizing property. A minimum-phase transfer function is defined as having neither poles nor zeros in the right half-s-plane. This case brings the condition that for systems with the same magnitude characteristic, the range in phase angle of the minimum-phase transfer function is minimum among all such systems. Static output feedback (SOF) stabilization can be considered as the first method that comes to mind when designing a stabilizing controller. However, when the number of inputs and outputs is small (such as in SISO systems), it may not be possible to obtain a SOF that stabilizes the plant.

In [5], a necessary and sufficient condition is derived, called the parity interlacing property (PIP), for the existence of a strongly stabilizing controller for a given plant. PIP is satisfied if the plant has an even number of real axis poles between each pair of consecutive real axis zeros in the s-plane (including $+\infty$).

Strong stabilization is crucial to maintain balance and achieve stable motion in robotic systems. In this context, the paper [19] presents an interesting controller design method to achieve strong stabilization in *underactuated planar robots (acrobot, pendubot)*. Although various controllers for acrobots and pendubots have been designed in the literature (for example, [23–25]), the focus of this study is on the strong stabilization of such systems. In this sense, the paper [19] will serve as a reference for experiments to be conducted in the scope of this study.

Two-link underactuated planar robots have a two-joint arm structure, with the motion of one link dependent on the other. The fundamental characteristic of these robots is that they operate two movable joints with a single sensor and a single actuator, implying underactuation.

Several previous studies have examined underactuated mechanical systems in the scope of control theory. The “Acrobot” term comes from the first studies by Murray and Hauser [26] where the controllability properties of such systems are addressed. Later, studies that focused on applying non-linear control to “Acrobot” systems gained attention, such as [27].

Bortoff [28] performed the first experimental studies for Acrobot as a Ph.D. thesis. The technique of local linearization of non-linear plant dynamics was utilized to design observers and controllers to balance the underactuated mechanism near its marginally stable equilibrium configuration. Another mechanism with similar characteristics was studied in [29] with very effective results. The control algorithms of this study were relatively new and innovative for a recently developing field of underactuated robotic control. In addition to the studies mentioned, [30] uses the energy of the system as a controlling strategy and is a notable exception among various studies that utilize more classical strategies.

Several other studies in the field of space robotics can also be mentioned, such as the papers by Papadopoulos and Dubowsky [31–33]. In these studies, the existence of so-called dynamic singularities was proven within task space control, which yields great complications in the control problems required to be solved in that field.

In conjunction with studies in the literature, this thesis designs second-order strong stabilizing controllers for a robotic system called 'acrobot' and investigates the closed-loop system through simulations by using the benchmark designs provided in [11, 19, 34].

1.3 Contributions

In this thesis, a number of second-order strongly stabilizing controller designs for the linearized acrobot model will be considered. Together with the strong stabilization design for the acrobot control problem, different methods to conduct analysis and comparison of proposed controllers are discussed. In this way, it is intended to demonstrate the performance of the designed controllers in different specifications.

In addition to designing strongly stabilizing controllers for the linearized acrobot model, the region of attraction for the nonlinear system is analyzed by using time domain simulations under different initial conditions.

1.4 Organization of Thesis

The general flow of the contents presented in this thesis begins with a mathematical model of the system given in Chapter 2. This chapter discusses calculation procedures for obtaining plant transfer functions and provides dynamic equations for the acrobot system to be controlled. The second-order strong stabilizing controllers proposed for this system are given in Chapter 3. First, the feedback loop of the Acrobot system is analyzed and the existence of stabilizing controllers is demonstrated. Then, the design procedures for such stabilizing controllers are discussed. Later, details regarding the MATLAB SIMULINK model used to analyze and simulate the feedback system with different controller types have been discussed in Chapter 4. After that, the performance criteria of the closed-loop system are explained in Chapter 5, together with the analysis results that are categorized and presented for each controller type design. Finally, in Chapter 6, overall evaluations are made, and possible future work propositions are presented.

Chapter 2

Mathematical Model of the Acrobot

The mathematical representation of any plant for any control scheme plays a very prominent role in control theory. In this chapter, the transfer function of the systems concerned, that is, the acrobot or pendubot systems, will be derived by the utilization of an analytical model of the physical plant.

The modeling problem of underactuated non-linear Lagrangian systems is defined as a challenging task. There are several approaches to handle the modeling problem. One of the important points for modeling the Lagrangian system is the system's total energy and its passivity relation, according to Kolesnichenko and Shiriaev [35]. In this chapter, energy-based swing-up control analysis for the acrobot is used to achieve a linearized model and transfer function [36].

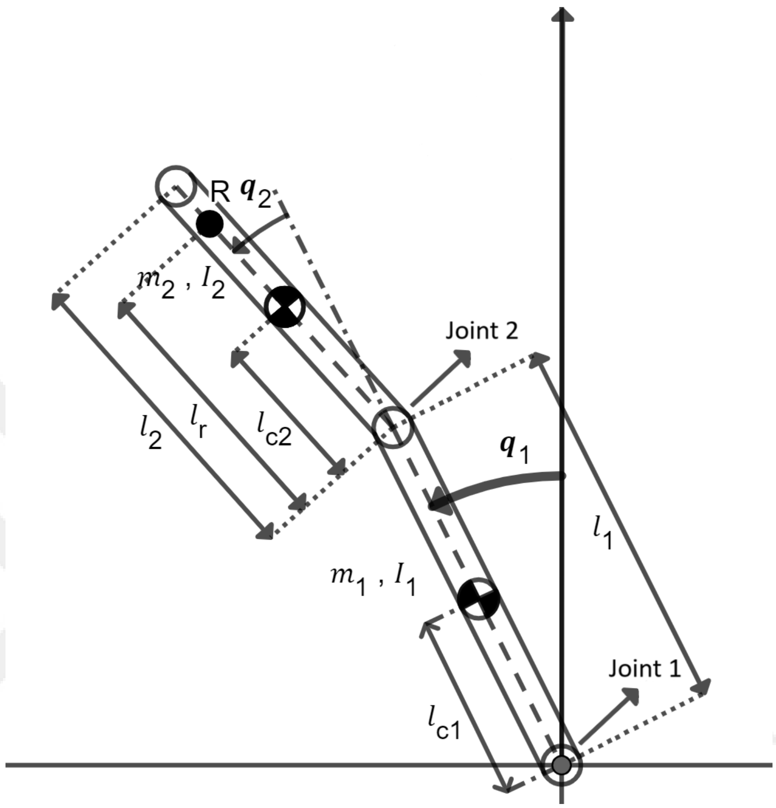

Figure 2.1: Two Link Underactuated Robot

Consider a two-link planar mechanism illustrated in Figure 2.1, m_1 and m_2 denote the mass values while I_1 and I_2 denote the centroidal moments of inertia of the first and second links, respectively; l_1 and l_2 are the lengths of the links, and l_{c1} and l_{c2} are the distances from the first and second joints to their center of mass. Parameters q_1 and q_2 represent the positional angle variables of the first and second links. The angle of the first link q_1 is measured between the link itself and the fixed vertical axis. The second angle q_2 is the angle variable between the second link itself and its proximal link, which is the first link. The equation of motion of the mechanism is derived from the Euler-Lagrange equations. The derivation in the following is taken from [37] and [36].

$$M(q)\ddot{q} + H(q, \dot{q}) + G(q) = \begin{bmatrix} 0 \\ \tau \end{bmatrix} \tag{2.1}$$

In (2.1), q is defined as the following vector, $q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}$. Furthermore, τ represents the torque applied to the joint between links 1 and 2. The following expressions define the matrices M,H, and G:

$$M(q) = \begin{bmatrix} \alpha_1 + \alpha_2 + 2\alpha_3 \cos q_2 & \alpha_2 + \alpha_3 \cos q_2 \\ \alpha_2 + \alpha_3 \cos q_2 & \alpha_2 \end{bmatrix} \tag{2.2}$$

$$H(q, \dot{q}) = \alpha_3 \begin{bmatrix} -2\dot{q}_1\dot{q}_2 - \dot{q_2}^2 \\ \dot{q_1}^2 \end{bmatrix} \sin q_2 \tag{2.3}$$

$$G(q) = \begin{bmatrix} -\beta_1 \sin q_1 - \beta_2 \sin (q_1 + q_2) \\ -\beta_2 \sin (q_1 + q_2) \end{bmatrix} \tag{2.4}$$

where g is the gravity acceleration and,

$$\begin{cases} \alpha_1 = m_1 l_{c1}^2 + m_2 l_1^2 + I_1 \\ \alpha_2 = m_2 l_{c2}^2 + I_2 \\ \alpha_3 = m_2 l_1 l_{c2} \\ \beta_1 = (m_1 l_{c1} + m_2 l_1) g \\ \beta_2 = m_2 l_{c2} g \end{cases} \tag{2.5}$$

The mechanical parameters used in (2.5) are defined in [36]. These parameters need to satisfy the following relations, according to [19].

$$\alpha_2\beta_1 > \alpha_3\beta_2 \tag{2.6}$$

$$\alpha_3\beta_1 \geq \alpha_1\beta_2 \tag{2.7}$$

$$\rho = (\alpha_2 + \alpha_3\beta_1 - (\alpha_1 + \alpha_3)\beta_2 > 0 \tag{2.8}$$

The conditions described in (2.6),(2.7), and (2.8) are necessary to be able to transform a nominally multi-output system with two different angular position states into a single output system through a position measurement as described in (2.9). Note that y is a scalar output. This would allow for the description of the Acrobot system as a Single Input-Single Output system from τ to y. For this purpose, the projection of a point along link 2 in the horizontal direction can be used. Let this point be defined as R and assume that this point is l_r meters from the joint between two links of the Acrobot along link 2. In that case, the horizontal projection of this point is calculated as;

$$y = l_1 \sin q_1 + l_r \sin (q_1 + q_2) \tag{2.9}$$

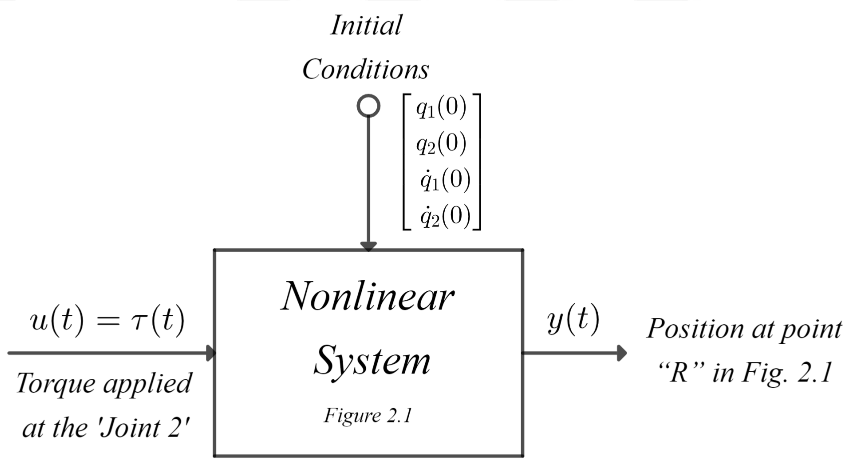

Figure 2.2: System Schema With Input and Output Definition

In summary, Figure 2.2 explains the overall configuration of the plant. The initial conditions are included to study the effect of nonlinearity around the equilibrium point (they can be considered as a disturbance to the LTI SISO system). The notations used on the two-link underactuated robots are summarized in Table 2.1.

After the above calculations, the system needs to be linearized around an equilibrium point. Since angular displacements are measured from the upright equilibrium point (UEP), in linearization we may assume $sin(\theta) = \theta$. Thus, the above-mentioned relations can be transformed into state-space representation in terms of A, B and C matrices shown in (2.12),(2.13) and (2.14), as linearization operation is done around UEP. The state space representation of the linearized

Table 2.1: Notation for Two-Link Underactuated Robot

Symbol	Unit	Description
g	m/s^2	The acceleration of gravity
l_1	m	Length of the first stick
l_2	m	Length of the second stick
l_{c1}	m	The distance from joint 1 to its center of mass
l_{c2}	m	The distance from joint 2 to its center of mass
m_1	kg	Mass of first stick
m_2	kg	Mass of second stick
J_1, I_1	$kg\ m^2$	The moment of inertia
J_2, I_2	$kg\ m^2$	The moment of inertia
l_r	m	The displacement from joint 2 to point R

model of the Acrobot is as shown below:

$$\dot{x} = Ax + Bu \tag{2.10}$$

$$y = Cx. \tag{2.11}$$

The state x and control input u are defined as $x = [q_1\ q_2\ \dot{q}_1\ \dot{q}_2]$ and $u = \tau$, at the upright equilibrium point $x_{eq} = 0$ and $\tau_{eq} = 0$. Then, the linearized model is defined as $\dot{x} = Ax + Bu$, where $\Delta = \alpha_1\alpha_2 - \alpha_3^2$ and ρ are defined in (2.8). In that case, the State Space representation of the Acrobot system in Figure 2.1 would have the following system matrices:

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{\alpha_2\beta_1 - \alpha_3\beta_2}{\Delta} & \frac{-\alpha_3\beta_2}{\Delta} & 0 & 0 \\ \frac{-\rho}{\Delta} & \frac{(\alpha_1+\alpha_3)\beta_2}{\Delta} & 0 & 0 \end{bmatrix} \tag{2.12}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ \frac{(\omega_1-\omega_2)\alpha_2 - \omega_2\alpha_3}{\Delta} \\ \frac{\omega_2\alpha_1 - (\omega_1-\omega_2)\alpha_2 - (\omega_1 - 2\omega_2)\alpha_3}{\Delta} \end{bmatrix} \tag{2.13}$$

The matrix C obtained in the light of (2.9) is shown below:

$$C = \begin{bmatrix} l_1 + l_r, l_r, 0, 0 \end{bmatrix} \tag{2.14}$$

In order to acquire the transfer function of the Acrobot system, Laplace transform of (2.10) and (2.11) is taken as:

$$sX(s) = AX(s) + BU(s) \tag{2.15}$$

$$Y(s) = CX(s) \tag{2.16}$$

which yields,

$$\frac{Y(s)}{U(s)} = C(sI - A)^{-1}B \tag{2.17}$$

Note that the transfer function of the plant $P(s)$ is defined as the division of $\frac{Y(s)}{U(s)}$ found in (2.17). After matrices given in (2.12), (2.13) and (2.14) are substituted, the following is found:

$$P(s) = \frac{\eta s^2 - \delta}{(\alpha_1\alpha_2 - \alpha_3^2)s^4 - (\alpha_2\beta_1 + \alpha_1\beta_2)s^2 + \beta_1\beta_2} \tag{2.18}$$

where

$$\eta = (\omega_2\alpha_1 - (\omega_1 - \omega_2)\alpha_3)l_r + ((\omega_1 - \omega_2)\alpha_2 - \omega_2\alpha_3)l_1 \tag{2.19}$$

and,

$$\delta = \omega_2\beta_1 l_r + (\omega_1 - \omega_2)\beta_2 l_1 \tag{2.20}$$

As described above, l_r is present in the plant function and appears as a determining factor affecting the location of zeros. The poles and zeros of the transfer function are described as follows:

$$p_{1,2} = \sqrt{\frac{(\alpha_1\beta_2 + \alpha_2\beta_1) \pm \sqrt{(\alpha_1\beta_2 - \alpha_2\beta_1)^2 + 4\alpha_3^2\beta_1\beta_2}}{2\Delta}} \tag{2.21}$$

Additionally, zeros of $P(s)$ calculated as described in [19]:

$$z = \sqrt{\delta/\eta} \tag{2.22}$$

The zeros and poles defined in (2.22) and (2.21) are all symmetric roots, therefore $P(s)$ has four poles; $\pm p_1$ and $\pm p_2$, and two zeros $\pm z$. For simplicity, the transfer function of the plant can be written in a more common form which illustrates poles and zeros, as given below:

$$P(s) = \frac{k_g(s^2 - z^2)}{(s^2 - p_1^2)(s^2 - p_2^2)} \tag{2.23}$$

where,

$$k_g = \frac{\eta}{\alpha_1\alpha_2 - \alpha_3^2} \tag{2.24}$$

For the plant given in (2.23), we will consider stable controllers and compare their performances and stability properties. The parameters chosen for acrobot are taken from [19] as given in Table 2.2.

Table 2.2: Mechanical Model Parameters of Acrobot

Symbol	m_1	m_2	l_1	l_2	l_{c1}	l_{c2}	l_r	J_1	J_2	g
Unit	kg		m					$kg \cdot m^2$		m/s^2
Value	1	1	1	2	0.5	1	0.55	0.083	0.33	9.81

Chapter 3

Strongly Stabilizing Controllers for Acrobot

3.1 Feedback System for The Linearized Model

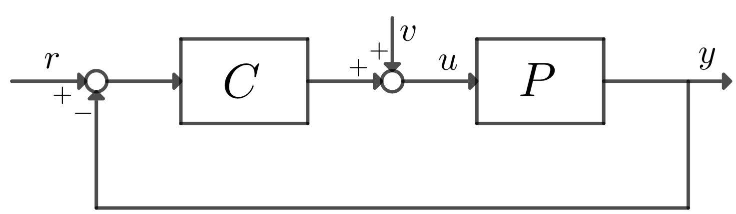

Figure 3.1: Unity Feedback Control System Block Diagram

The linearized plant model, which is used in the simulations, is designed according to the mechanical parameters of [19] for the acrobot. These mechanical parameters are given in Table 2.2.

Note that the plant satisfies the parity interlacing property (PIP) if and only if

$$z < \min\{p_1, p_2\}, \quad \text{or} \quad z > \max\{p_1, p_2\}. \tag{3.1}$$

The parameters given for the system of interest are as follows: $k_g = -1.3545$,

$p_2 = 6.101$, $p_1 = 2.24$, and $z = 1.281$, and satisfy the PIP. The resulting plant transfer function is

$$P(s) = \frac{-1.3545(s-1.281)(s+1.281)}{(s-6.101)(s-2.24)(s+2.24)(s+6.101)} \tag{3.2}$$

The controller C must first be designed by considering the closed-loop stability condition of the feedback system shown in Figure 3.1. Second, we require that C itself be stable. Among the strongly stabilizing controllers, comparisons are made using stability margins that indicate the relative stability of the system. These margins include phase, gain, and delay margins. Proper stability margins protect the designer against variations in the system parameters [38].

3.2 Controller Design with Various Objectives

In this section, the design criteria are discussed for the controllers providing strong stabilization for the linearized model of the 'Acrobot System' whose transfer function is depicted in (3.2). In this context, different design approaches are presented. The objective of the first method is to minimize the spectral abscissa, i.e. the real part of the rightmost closed-loop system pole. Another method uses optimization of the phase margin among all possible controllers. Later, an additional method based on vector margin optimization is used to overcome concurrent problems that arise while applying the aforementioned optimizations. These methods are applied to four different types of reduced-order controllers. In [34], Xin *et al.* demonstrate that for plants that have similar transfer functions in the form of (2.23), there exist no first-order stabilizing controllers. Together with this conclusion, [19] points out that there are also second-order controllers that can stabilize the system.

Second-order controllers designed in previous studies for these systems are denoted as Type 0 in Table 3.1 [11, 19]. Possible other second-order controllers, named as Type 1, Type 2, and Type 3, are also defined in Table 3.1. Accordingly, in Type 1 controllers, one of the controller zeros is assigned as a free variable rather than being one of the open-loop poles of the system. In Type 2 controllers, the controller zeros are determined by the values of the open-loop pole pair of the system, but both poles of the controller are assigned as free variables. Finally, in Type 3 controllers, except for assigning one of the zeros of the controller as the smallest open-loop pole of the system, the remaining zero and the other two poles of the controller are assigned as free variables.

Table 3.1: Second Order Controller Types

Controller Type	Transfer Function	Free Variables
Type 0	$\frac{k_c(s+p_1)(s+p_2)}{(s+a)(s+z)}$	k_c, a
Type 1	$\frac{k_c(s+b)(s+p_2)}{(s+a)(s+z)}$	k_c, a, b
Type 2	$\frac{k_c(s+p_1)(s+p_2)}{(s+a)(s+c)}$	k_c, a, c
Type 3	$\frac{k_c(s+b)(s+p_2)}{(s+a)(s+c)}$	k_c, a, b, c

3.2.1 Optimization of the Spectral Abscissa

Definition: For a given linear time invariant (LTI) and stable feedback system, the real part of the rightmost pole of the closed-loop system is defined as the *spectral abscissa*.

In this section, we investigate the spectral abscissae for different types of strongly stabilizing second-order controllers.

3.2.1.1 Controller Type 0

In this section, second-order controller designs, which are defined in [19] and [11], are considered. The first selected controller comes from [19], which discusses the definition, transfer function, and simulation objectives of the selected plant. In (3.3), this controller is given as follows:

$$C_X(s) = \frac{-131.4411(s+6.101)(s+2.24)}{(s+19.01)(s+1.281)} \tag{3.3}$$

In the design procedure of [19], controller $C_X(s)$ was chosen by designing a proper second-order stable controller for an appropriate zero of the plant. In a later study, [11] brings a new design approach to second-order controllers for the same plant model, which has been represented in Example 4.2 of [11]. This example was solved by the proposed method for strictly proper plants with one positive zero. The resulting Type 0 controller is defined below:

$$C_H(s) = \frac{-75.7487(s+6.101)(s+2.24)}{(s+10.4206)(s+1.281)} \tag{3.4}$$

The authors of [19] and [11] used different methods to obtain a strongly stabilizing controller of Type 0 for the plant (3.2). In this part, an alternative controller with the same structure is searched. According to Table 3.1, the variable a together with the gain value are the free variables that can be used to create many different controller alternatives. After comparing an adequate number of controllers that are within the search space, which is made up of controllers with different combinations of the aforementioned free variables, each a value in the search range is presented with the resulting pole locations for *Type 0* controllers. This effect of the parameter a, which chooses the leftmost spectral abscissa, can be observed in Figure 3.2.

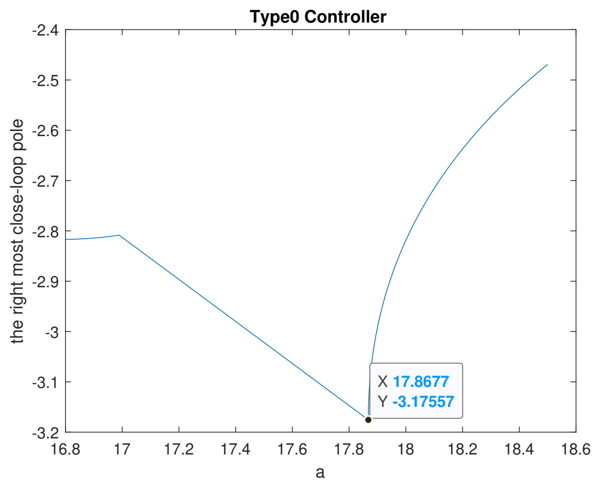

Figure 3.2: Spectral Abscissae According to the a Value

Furthermore, the selected a value for the *Type 0* controller has the *root-locus* plot shown in Figure 3.3. There are three branches that have a meeting point at -3.17 on the real axis; detailed view is demonstrated in Figure 3.4. The meeting point is observed to yield the best result and hence, the gain value of this point is chosen to determine the controller. In [19], the most left spectral abscissa was found as -1.28.

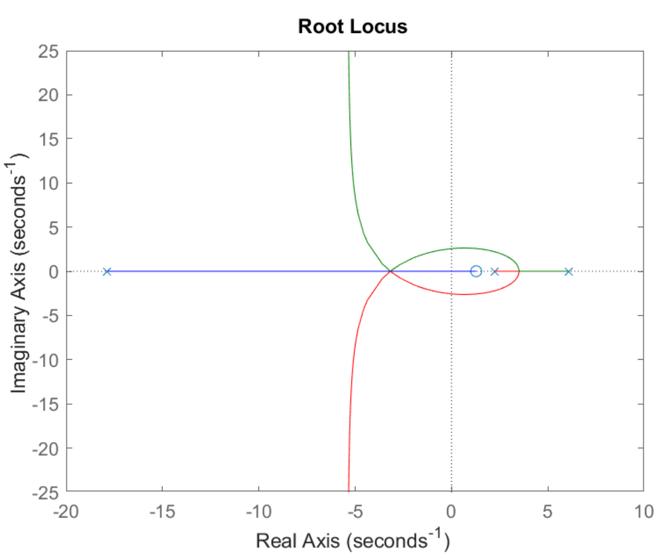

Figure 3.3: Root Locus Graph For the Selected a Value for Controller Type 0

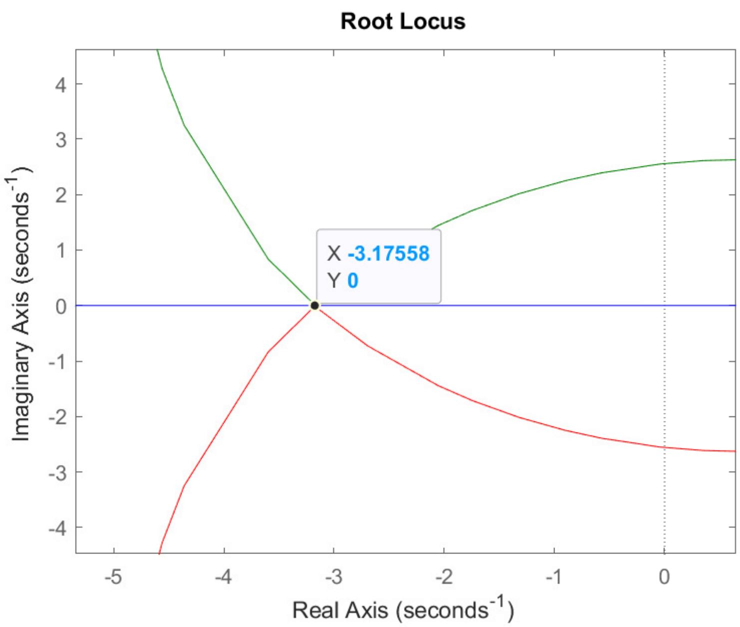

Figure 3.4: Root Locus Graph Meeting Point for Controller Type 0

Resulting *Type 0* controller is given below:

$$C_0(s) = \frac{-122.27(s + 6.101)(s + 2.24)}{(s + 17.87)(s + 1.281)} \tag{3.5}$$

3.2.1.2 Controller Type 1

In controller Type 1, the variables a, b, and the gain value are free variables according to Table 3.1. In this type, one of the two zeros of the controller has been made to be a free variable in addition to those of Type 0 controllers. A similar approach is conducted with Section 3.2.1.1.

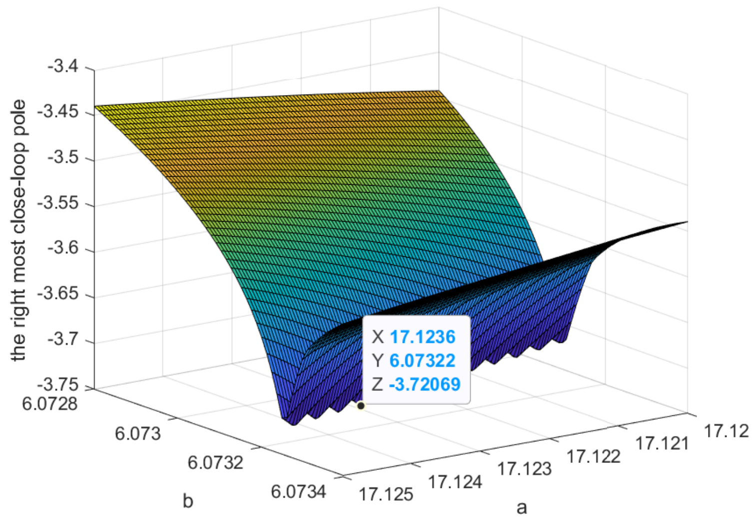

Figure 3.5: Spectral Abscissa According to the a and b Values

The values of a and b determine the leftmost spectral abscissa for controllers of Type 1. The effect of these free variables on the spectral abscissae is illustrated in Figure 3.5. For selected values of a and b, the plot of the root locus is shown in Figure 3.6.

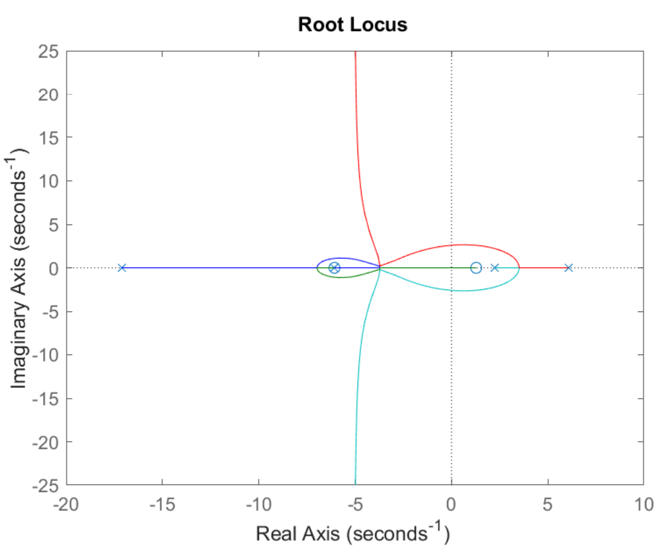

Figure 3.6: Root Locus Graph for the Selected a and b Values for Controller Type 1

The resulting controller that has the left-most spectral abscissa is given below:

$$C_1(s) = \frac{-117.19(s + 6.073)(s + 2.24)}{(s + 17.12)(s + 1.281)} \tag{3.6}$$

3.2.1.3 Controller Type 2

For controller Type 2, the parameters a, c, and the gain value are free variables according to Table 3.1. The same systematic design method applies as in Section 3.2.1.1. The free variables are adjusted to create different controllers. These controllers are then compared, and the one with the leftmost spectral abscissa is chosen. In Figure 3.7, the minimum locations of roots are shown with respect to the variation of the values of a and c. In Figure 3.8, root locus for selected controller of Type 2 is shown.

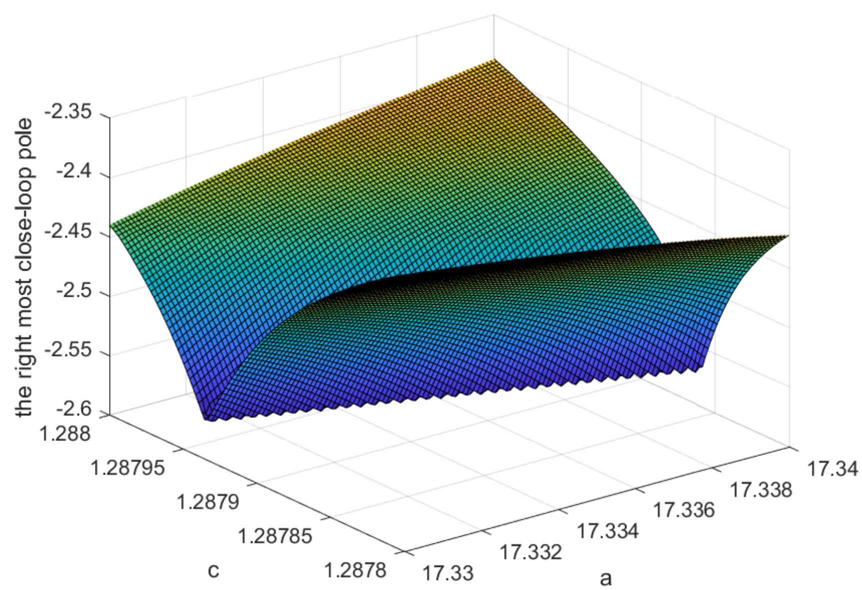

Figure 3.7: The Spectral Abscissae According to The a and c Values

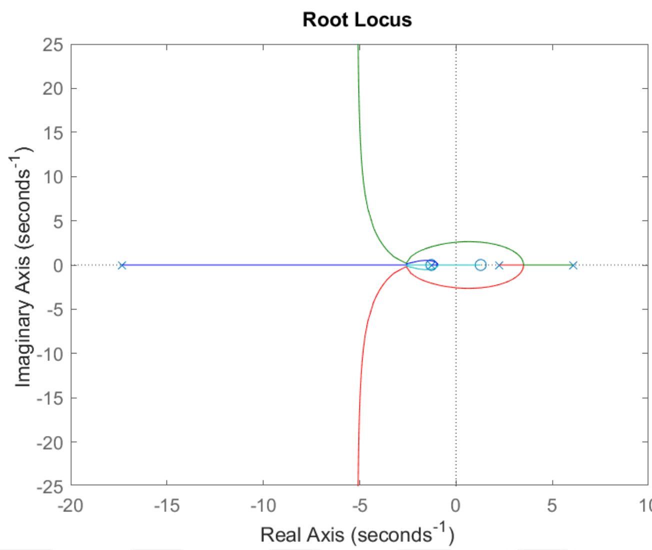

Figure 3.8: Root Locus Graph For the Selected a and c Values for Controller Type 2

The resulting controller that has the left-most spectral abscissa is given below:

$$C_2(s) = \frac{-117.47(s + 6.101)(s + 2.24)}{(s + 17.34)(s + 1.288)} \tag{3.7}$$

3.2.1.4 Controller Type 3

For controller Type 3, except the plant pole p_2, all the remaining parameters, that is, the parameters of a, b, c, and the gain value are free variables according to Table 3.1. If we follow a similar approach to that in Section 3.2.1.1, the resulting controller of Type 3 is found as in (3.8). The root locus of selected controller is given in Figure 3.9.

$$C_3(s) = \frac{-107.84(s + 6.04)(s + 2.24)}{(s + 15.94)(s + 1.292)} \tag{3.8}$$

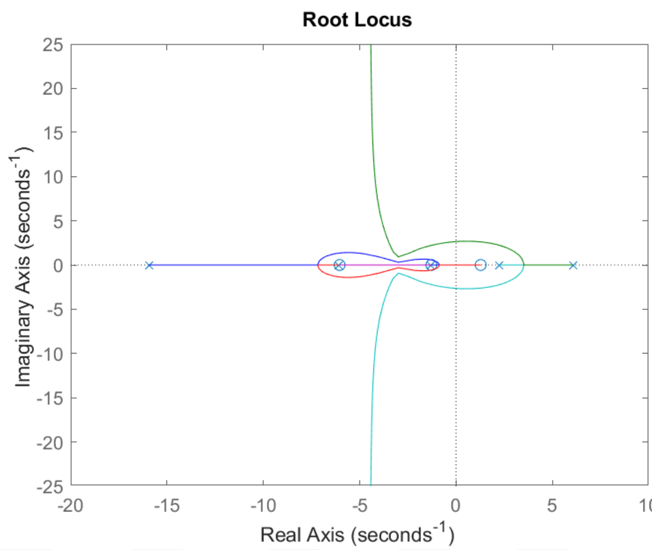

Figure 3.9: Root Locus Graph For the Selected a, b and c Values for Controller Type 3

3.2.2 Phase Margin Optimization

In this part, the controller is designed according to the maximum phase margin. The phase margin is defined as the amount of additional phase lag at the gain crossover frequency with which the system reaches the limit of instability. The selected final controllers have the maximum amount of phase margins possible among other stabilizing controllers of their respective types. Since the plant is unstable and nonminimum phase, it would be more convenient to use *Nyquist plot* (rather than the Bode plot) to observe the phase margin. Because the open-loop system transfer function has two right-hand side poles, the designed feedback system should have two counterclockwise encirclements of the critical point -1 in the Nyquist plot. In Figure 3.10, the Nyquist plot for the open-loop transfer function $G(s)$ with unity gain is shown.

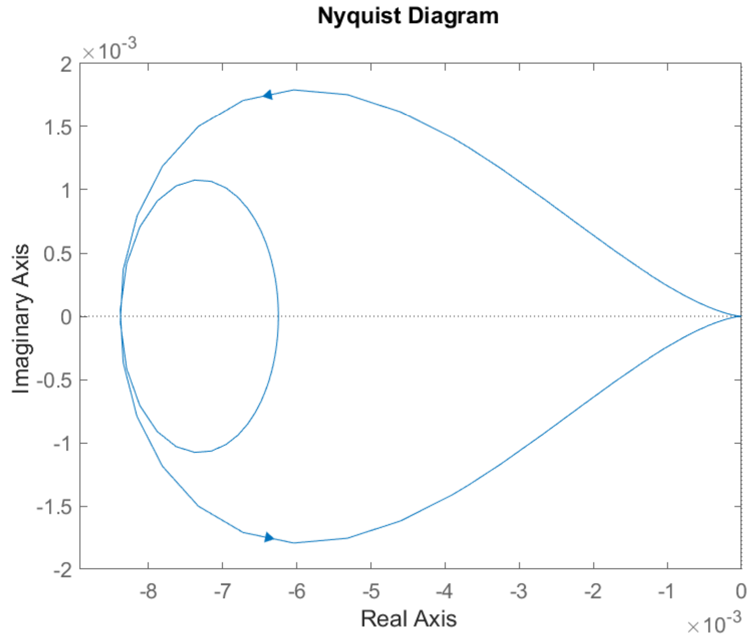

Figure 3.10: Nyquist Diagram of $G(s)$

For a correct interpretation of the stability margins, careful study of the Nyquist diagram is required. Notice that the critical point -1 in Figure 3.10 is not encircled twice, and therefore it is obviously unstable. Changing the gain value of $G(s)$ scales the shape of the Nyquist plot, and by this way it becomes possible to make adjustments for stability. For the phase margin based controller design, the limits of gain values for closed-loop stability must be calculated. In this study, the range of stabilizing gain values can be found using the mathematical relationship given in (3.9).

According to the Nyquist result, there are lower and upper bounds for the gain values, which can be defined as the inequality of (3.9). The inequality includes the magnitude of the response of the system at the phase crossover frequency ω_p. The phase crossover frequency ω_p is defined as the frequency at which the phase of the open-loop transfer function crosses -180° phase shift for the first time.

$$\frac{1}{|G(0)|} > K_g > \frac{1}{|G(j\omega_p)|} \tag{3.9}$$

The magnitude of $G(s)$ at the phase crossover frequency can be found using the *Bode plot*. It is important to note that the open-loop transfer function of the acrobot mechanical plant has one right-hand side zero on the s-plane. This phenomenon is called as a non-minimum phase system. In this case, the system behaves in a non-classical way in the frequency domain. For this reason, stability margins, especially the phase margin of the system, must be carefully examined by using Nyquist plots. In this study, it is observed that maximazing the phase margin using a classical Bode approach does not yield consistent results due to the aforementioned non-minimum phase behavior. However, it should also be noted that examination of some Nyquist plots belonging to phase margin optimized controllers has shown that maximizing the phase margin may cause minimization of the gain margin. This phenomenon is addressed in detail in Section 3.2.3.

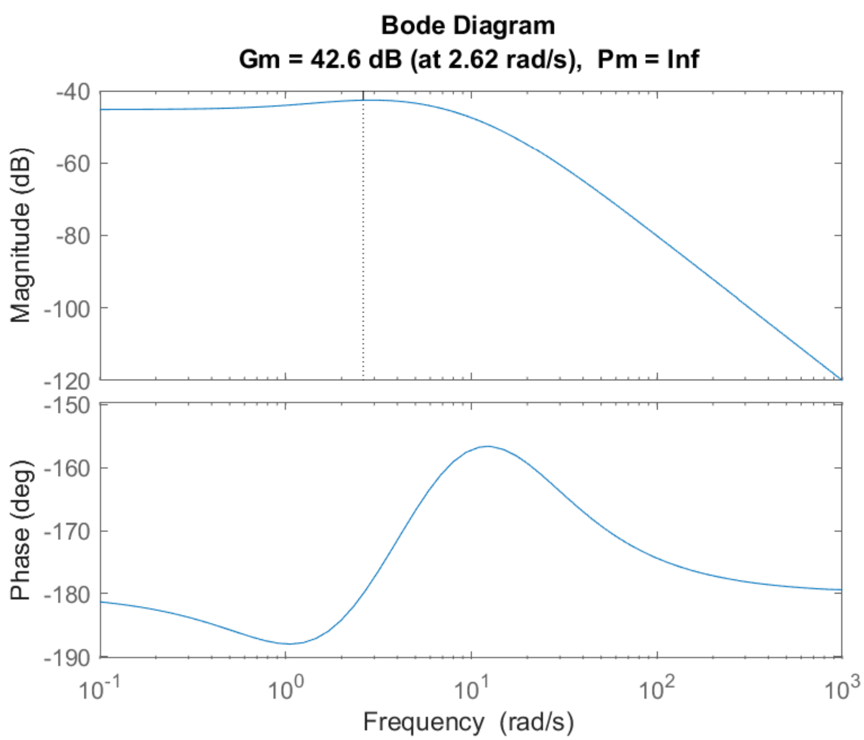

Figure 3.11: Bode Plots of G(s)

3.2.2.1 Controller Type 0

The motivation is described in the previous pages, and $G(0)$ is calculated for the generalized form of the controller Type 0. It is the upper bound of the gain values that stabilize the system. The transfer function is given at (3.10).

$$G(s) = \frac{(s-z)}{(s+a)(s-p_1)(s-p_2)} \tag{3.10}$$

Then calculate $|G(0)|$,

$$|G(0)| = \left|\frac{(-z)}{(+a)(-p_1)(-p_2)}\right| \tag{3.11}$$

The final inequality can be defined as inequality (3.12)

$$\left|\frac{a\ p_1\ p_2}{z}\right| \quad > \quad K_g \quad > \quad \frac{1}{|G(j\omega_p)|} \tag{3.12}$$

Therefore, for each fixed a value, there is a different range of gain (K_g) that stabilizes the system. Each range has different phase margin results for each different K_g value, which is observable in Figure 3.12.

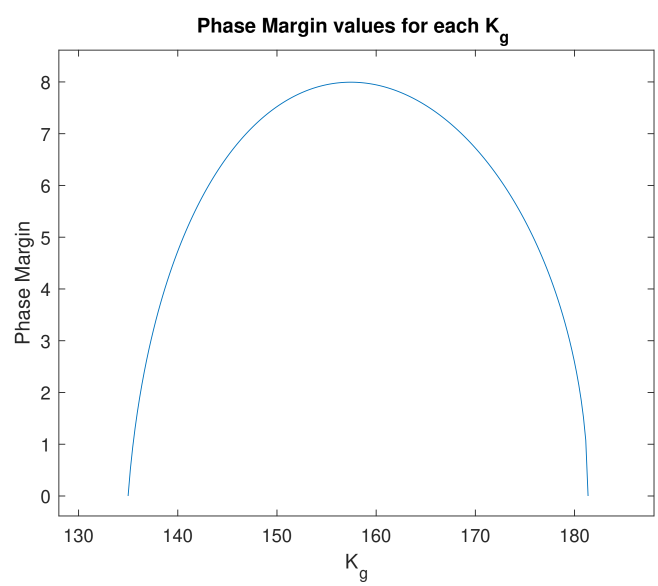

Figure 3.12: Phase Margin Values of K_g with an Arbitrary Controller Type 0

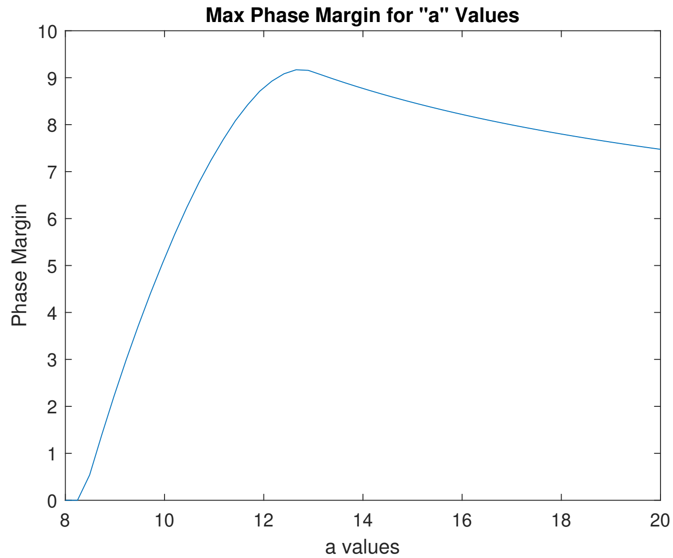

Figure 3.13: Phase Margin Values of a with Controller Type 0

For phase margin optimized controller of Type 0, transfer function is given in (3.13):

$$C_{0PM} = \frac{-87.183(s + 6.101)(s + 2.24)}{(s + 12.74)(s + 1.281)} \tag{3.13}$$

The phase margin of the system with controller C_{0PM} is 9.1782.

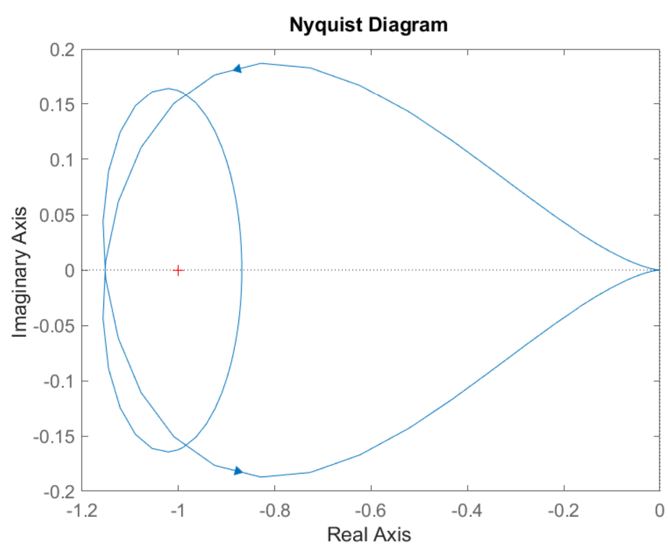

Figure 3.14: Nyquist Plot of C_{0PM}

3.2.2.2 Controller Type 1

According to Table 3.1, controller definition yiels following openloop transfer function:

$$G(s) = \frac{(s+b)(s-z)}{(s+a)(s-p_1)(s+p_2)(s-p_2)} \tag{3.14}$$

Then calculate $|G(0)|$,

$$|G(0)| = \left| \frac{(b)(-z)}{(a)(-p_1)(p_2)(-p_2)} \right| \tag{3.15}$$

The same approach with (3.9), stable gain values range is following inequality:

$$\left| \frac{a\ p_1\ p_2^2}{b\ z} \right| \quad > \quad K_g \quad > \quad \frac{1}{|G(j\omega_p)|} \tag{3.16}$$

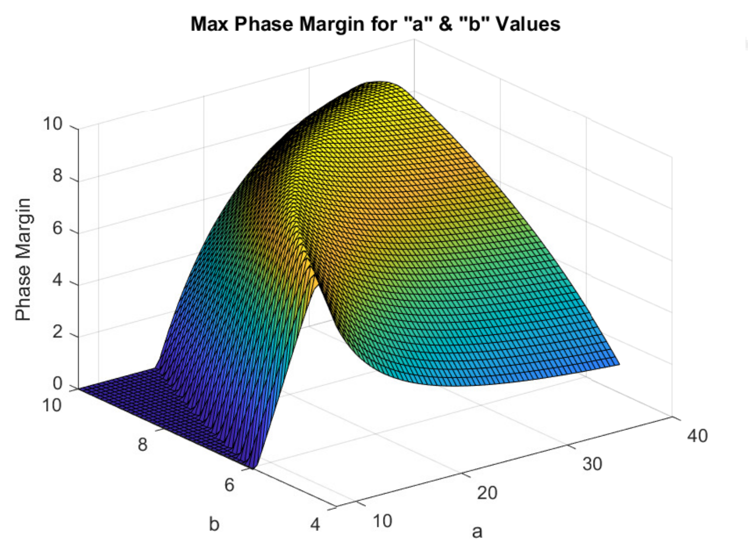

Figure 3.15: Phase Margin of a and b with Controller Type 1

According to the result of the calculations, the novel transfer function is defined as (3.17)

$$C_{1PM} = \frac{-141.21(s+8.76)(s+2.24)}{(s+1.281)(s+29.31)} \tag{3.17}$$

The phase margin of the system with C_{1PM} is 9.4.

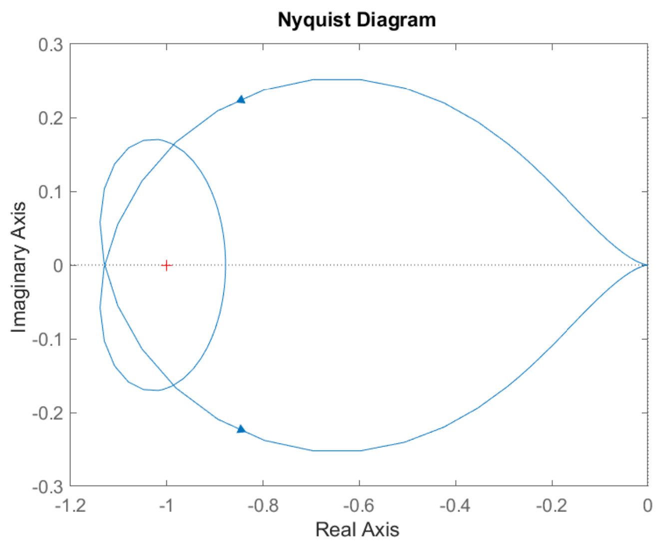

Figure 3.16: Nyquist Plot of C_{1PM}

3.2.2.3 Controller Type 2

According to Table 3.1, controller definition yiels following openloop transfer function:

$$G(s) = \frac{(s+z)(s-z)}{(s+a)(s+c)(s-p_1)(s-p_2)} \tag{3.18}$$

Then calculate $|G(0)|$,

$$|G(0)| = \left| \frac{(z)(-z)}{(a)(c)(-p_1)(-p_2)} \right| \tag{3.19}$$

With the same approach as in (3.9), range for stabilizing gain values is the inequality given in (3.20):

$$\left| \frac{a\ c\ p_1\ p_2}{z^2} \right| \quad > \quad K_g \quad > \quad \frac{1}{|G(j\omega_p)|} \tag{3.20}$$

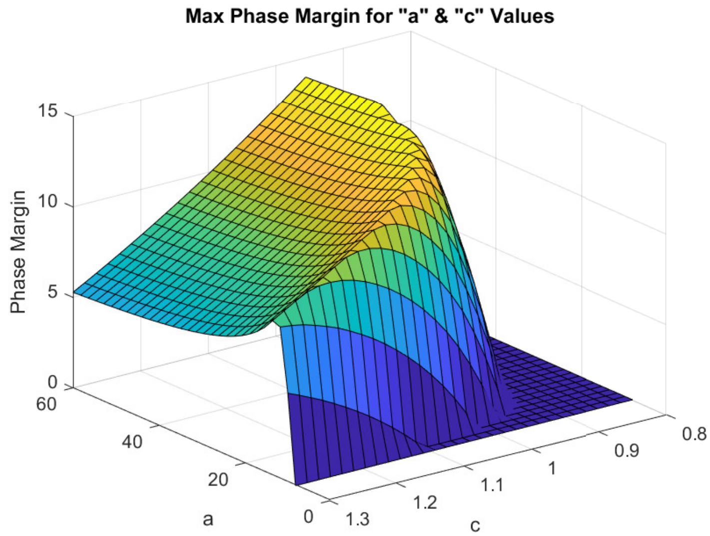

Figure 3.17: Phase Margin of a and c with Controller Type 2

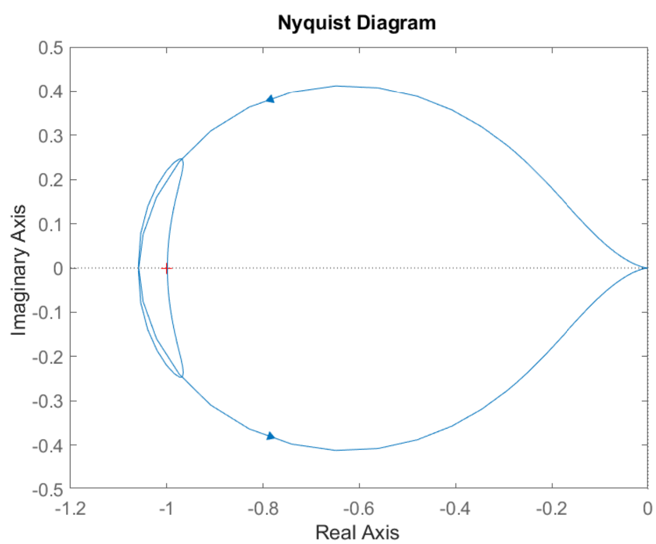

Figure 3.18: Nyquist Plot of C_{2PM}

According to the result of the calculations, the novel transfer function is defined as (3.21)

$$C_{2PM} = \frac{-240.27(s+6.101)(s+2.24)}{(s+40.84)(s+0.96)} \tag{3.21}$$

The phase margin of the system with C_{2PM} is 14.2674.

3.2.2.4 Controller Type 3

According to Table 3.1, controller definition yiels following openloop transfer function:

$$G(s) = \frac{(s+b)(s-z)}{(s+a)(s+c)(s-p_1)(s-p_2)} \tag{3.22}$$

Then calculate $|G(0)|$,

$$|G(0)| = \left|\frac{(b)(-z)}{(a)(c)(-p_1)(-p_2)}\right| \tag{3.23}$$

The same approach with (3.9), stable gain values range is following inequality:

$$\left|\frac{a\ c\ p_1\ p_2}{b\ z}\right| \quad > \quad K_g \quad > \quad \frac{1}{|G(j\omega_p)|} \tag{3.24}$$

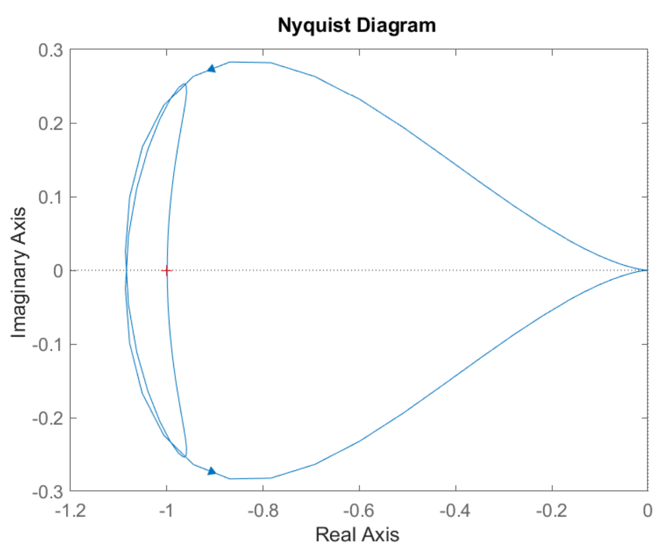

Figure 3.19: Nyquist Plot of C_{3PM}

According to the result of the calculations, the novel transfer function is defined as (3.25)

$$C_{3PM} = \frac{-101.93(s+4.5)(s+2.24)}{(s+13.11)(s+0.93)} \tag{3.25}$$

The phase margin of the system with C_{3PM} is 14.6124.

3.2.3 Vector Margin Optimization

In this section, controllers are designed to maximize the vector margin. The vector margin is the minimum distance from the polar plot $G(j\omega)$ to the point $-1 + j0$. The mathematical definition of the vector margin is given in (3.26).

$$VM = \beta = \min_{\omega} |G(j\omega) - (-1)| \tag{3.26}$$

In [39], it is denoted that vector margin is the inverse of the H_∞ norm of the sensitivity function. Recall that the sensitivity function $S(s)$ is given in (3.27).

$$S(s) = \frac{1}{1 + G(s)} \tag{3.27}$$

This yields an alternative definition for vector margin as shown in (3.28).

$$VM = \beta^{-1} = \max_{\omega} |S(j\omega)| = \|S\|_\infty \tag{3.28}$$

The motivation of this section is that phase margin maximization may cause very small gain margins in some cases as observed in previous section. For example, the controller C_{2PM} has a very small gain margin, while having a considerably improved phase margin. In Figure 3.20 this phenomenon can be seen, as the phase margin of C_{2PM} increases *(direction 1)*, the inner loop of the polar plot converges to the point $-1 + j0$ through the real axis *(direction 2)*. Therefore, it is more acceptable to perform a method that optimizes gain and phase margins while both staying in an acceptable range. For this reason, in this section we use vector margin optimization.

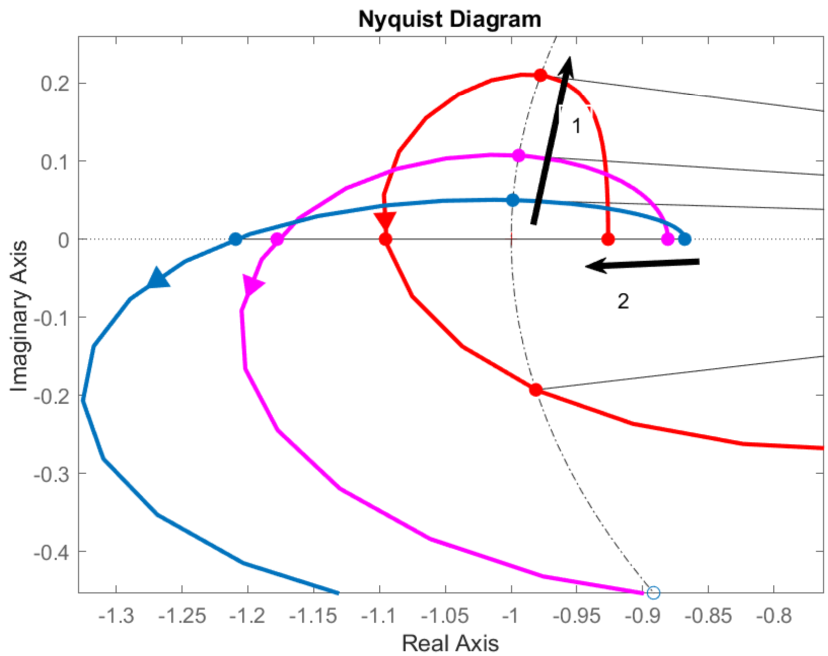

Figure 3.20: Decreasing Gain Margin due to Phase Margin Optimization Only

Similar to Section 3.2.2, the range of stabilizing gain values is calculated again by using *Nyquist plot* in this part. Later, the controller that yields the best vector margin for its respective type is found.

3.2.3.1 Controller Type 0

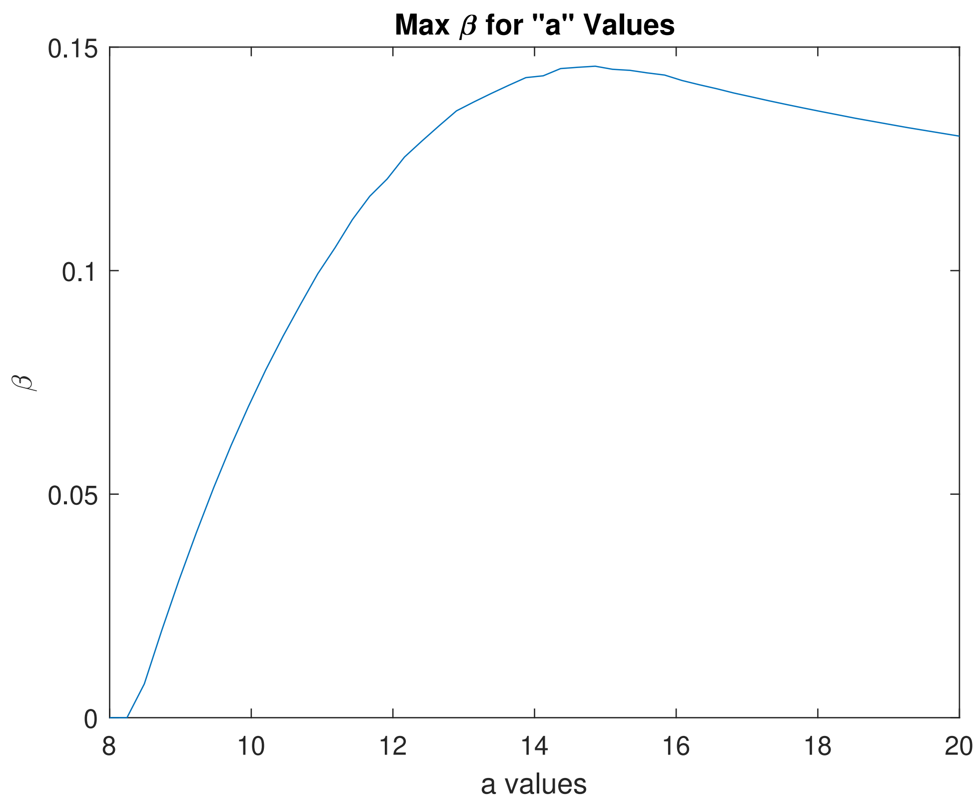

Figure 3.21: Vector Margin of a with Controller Type 0

For Controller Type 0, vector margin optimized transfer function is given in (3.29):

$$C_{0VM} = \frac{-101.37(s + 6.101)(s + 2.24)}{(s + 15.07)(s + 1.281)} \tag{3.29}$$

The vector margin of the system with controller C_{0VM} is 0.1459. Nyquist plot for C_{0VM} is shown in Figure 3.22.

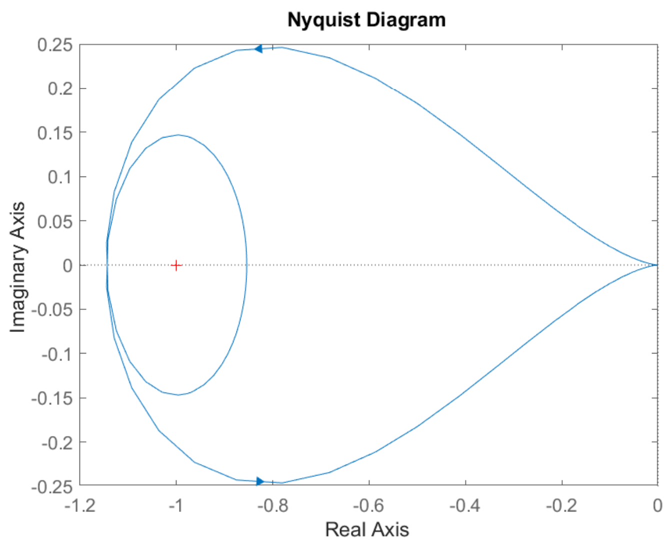

Figure 3.22: Nyquist Plot of C_{0VM}

3.2.3.2 Controller Type 1

The controller C_{1VM} has one more free variable and therefore has the three-dimensional plot for the vector margin variation shown in Figure 3.23.

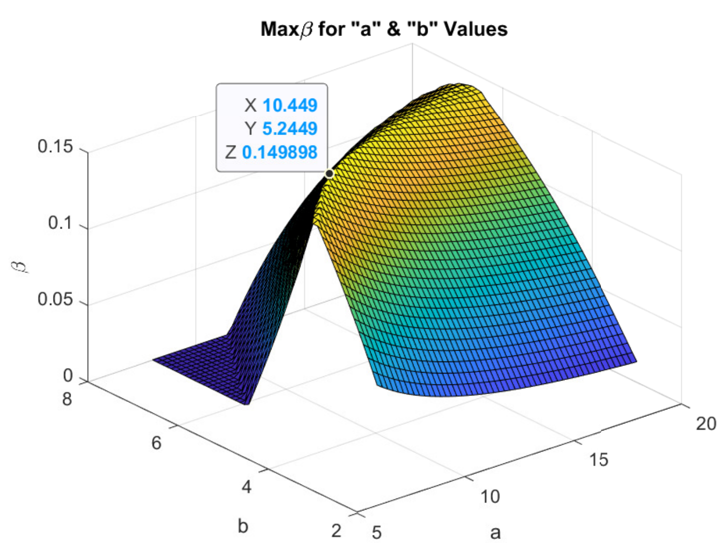

Figure 3.23: Vector Margin of a and b with Controller Type 1

According to the result of the calculations, the transfer function for the controller of Type C_{1VM} is defined as (3.30)

$$C_{1VM} = \frac{-81.382(s + 5.245)(s + 2.24)}{(s + 1.281)(s + 10.45)} \tag{3.30}$$

The vector margin of the system with C_{1VM} is 0.1499. Nyquist plot for C_{1VM} is given in Figure 3.24.

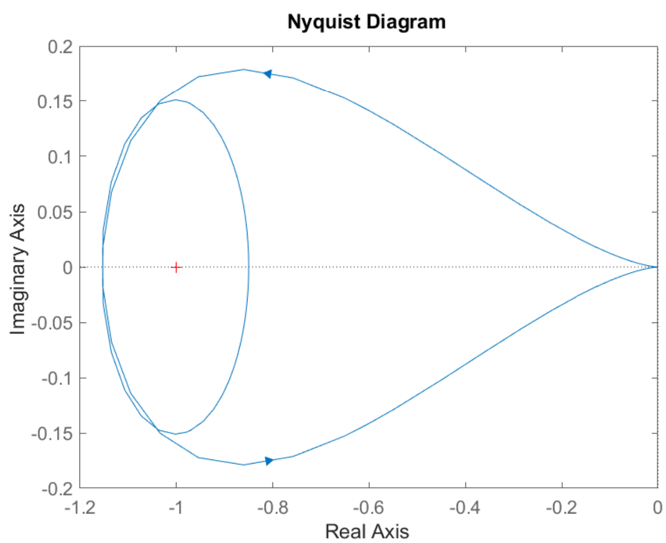

Figure 3.24: Nyquist Plot of C_{1VM}

3.2.3.3 Controller Type 2

Calculations for controller Type C_{2VM} result in the transfer function given in (3.31). In Figure 3.25, the vector margin graph is shown for changing parameters a and c.

$$C_{2VM} = \frac{-97.499(s + 6.101)(s + 2.24)}{(s + 14.45)(s + 1.29)} \tag{3.31}$$

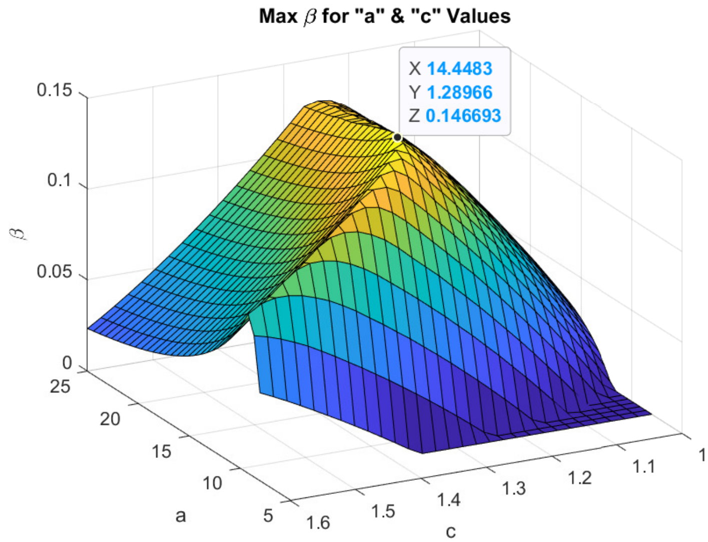

Figure 3.25: Vector Margin of a and c with Controller Type 2

The vector margin of the system with C_{2VM} is 0.1467. Nyquist plot for the vector optimized Type 2 controller is shown in Figure 3.26.

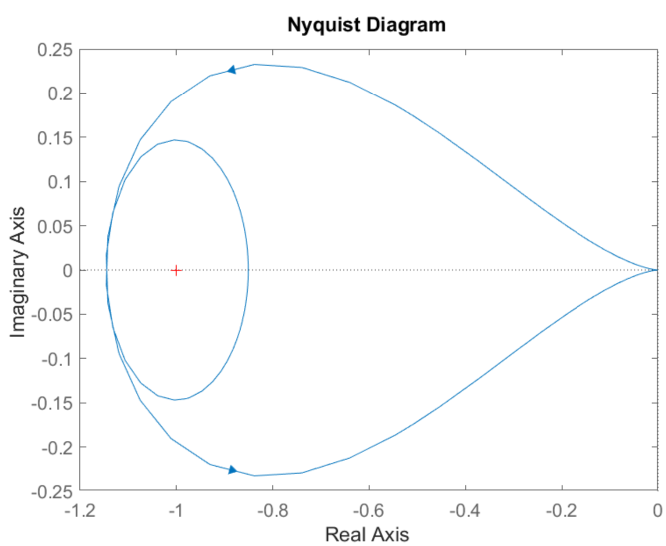

Figure 3.26: Nyquist Plot of C_{2VM}

3.2.3.4 Controller Type 3

The last controller type for vector margin optimization, C_{3VM}, has the transfer function given in (3.32).

$$C_{3VM} = \frac{-79.651(s+5.056)(s+2.24)}{(s+9.889)(s+1.278)} \tag{3.32}$$

The vector margin of the system with C_{3VM} is 0.1502. Nyquist plot for C_{3VM} is given in Figure 3.27.

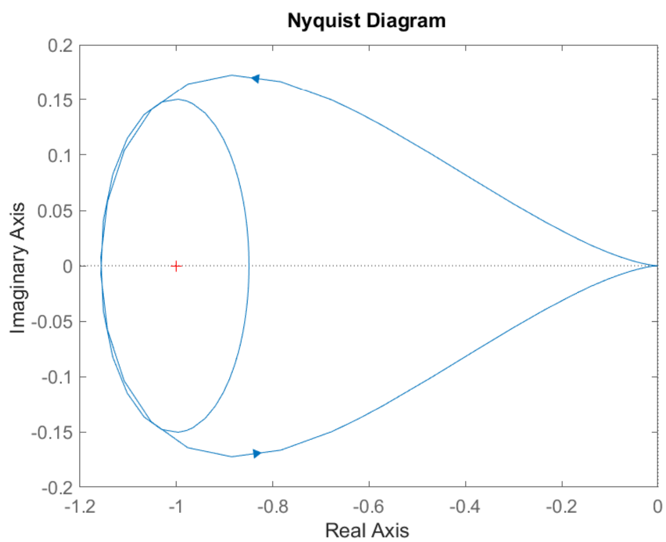

Figure 3.27: Nyquist Plot of C_{3VM}

Chapter 4

Simulation Studies

In the context of this research, the intricate dynamics of a mechanical system have been brought to life through the application of three-dimensional animation and visualization techniques, facilitated by the Simscape Multibody tool embedded in MATLAB. Simscape Multibody is a powerful tool within the Simscape family of products designed to facilitate the modeling and simulation of multidomain physical systems with a focus on mechanical components and their interactions.

Utilizing Simscape Multibody has enabled the examination and testing of stability controllers in the developed model. This phase of the study provides valuable insight into the robustness and effectiveness of stability control mechanisms within the considered mechanical system.

The Acrobot system of interest is analyzed and simulated in a MATLAB SIMULINK environment. A closed-loop system with a specified controller is also simulated with the plant. The overall layout of the system can be seen in Figure 4.1.

In Figure 4.1, the block *'myController'* refers to the controller that will be analyzed and simulated. The block *'Acrobot Plant'* refers to a physical plant that takes torque from the joint between Acrobot links as input and gives five different outputs. The first output, *'Plant output'*, refers to the mathematical

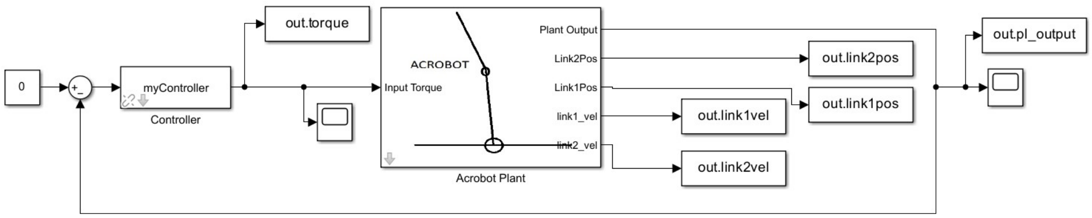

Figure 4.1: System Layout

expression that is taken as singularized plant output given in Chapter 2 (2.9). The signals *‘Link1Pos’* and *‘Link2Pos’* refer to the angular position of the first link to the ground and the angular position of the second link to the first link, respectively. ‘$link1_{vel}$’ and ‘$link2_{vel}$’ are the angular speeds of these positional variables. The *‘Plant Output’* is fed back to the control loop and compared with the reference signal, which is zero. Then, the calculated error is forwarded to the controller, and hence the feedback loop is completed.

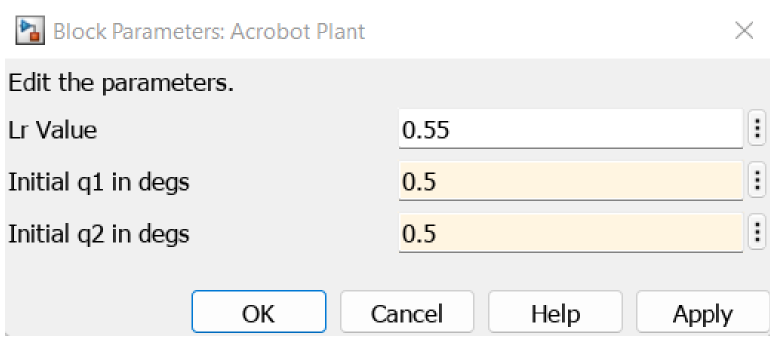

Figure 4.2: Block Parameters of The Physical Plant

In Figure 4.2, three parameters that are needed to complete the definition of the system can be seen. ‘Lr Value’ refers to the parameter ‘l_r’ in Chapter 2 to define the plant output properly. ‘Initial q_1 in degs’ and ‘Initial q_2 in degs’ parameters refer to the initial conditions of the Acrobot in terms of q_1 and q_2.

The structure of the block ‘Acrobot Plant’ can be seen in Figure 4.3. This intermediate block level is added to form the ‘Plant Output’ signal, which depends on the selection of ‘l_r’. The remaining input-output relations are identical to the ‘Acrobot Plant’ mentioned above.

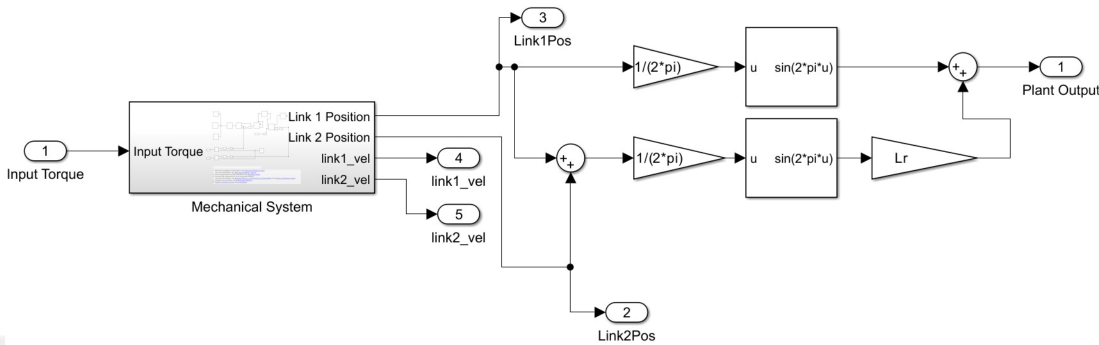

Figure 4.3: Pyhsical System Layout Under The Mask

Finally, in Figure 4.4, mechanical modeling of the physical system itself can be seen. At this step, the SIMULINK Simscape Multibody module has been used. This module can be used for mathematical simulation of physical multibody systems as well as to visualize the behavior of the system. An example can be seen in Figure 4.5.

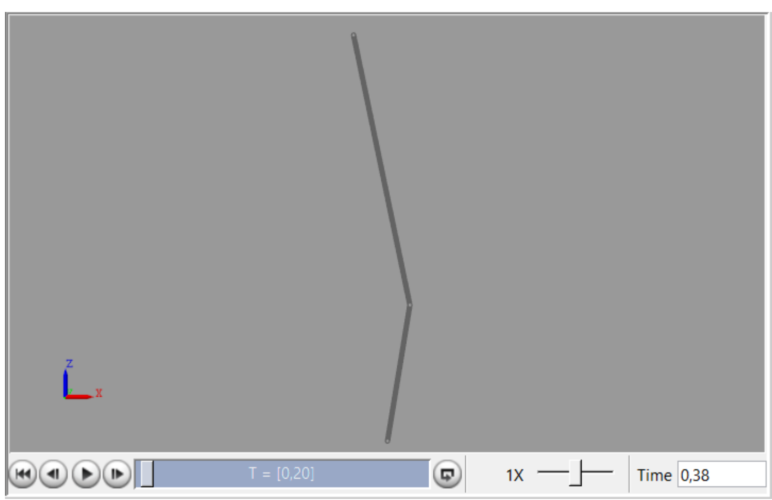

Figure 4.5: Visualization of The Physical System Using Simscape Multibody

Orientations and setup parameters to construct the Acrobot system are defined according to a reference axis that lies vertically with respect to the ground, towards which standard earth gravity is applied. In blocks '$LINK$ 1' and '$LINK$ 2', CAD models of the links 1 and 2 are provided with related reference coordinate

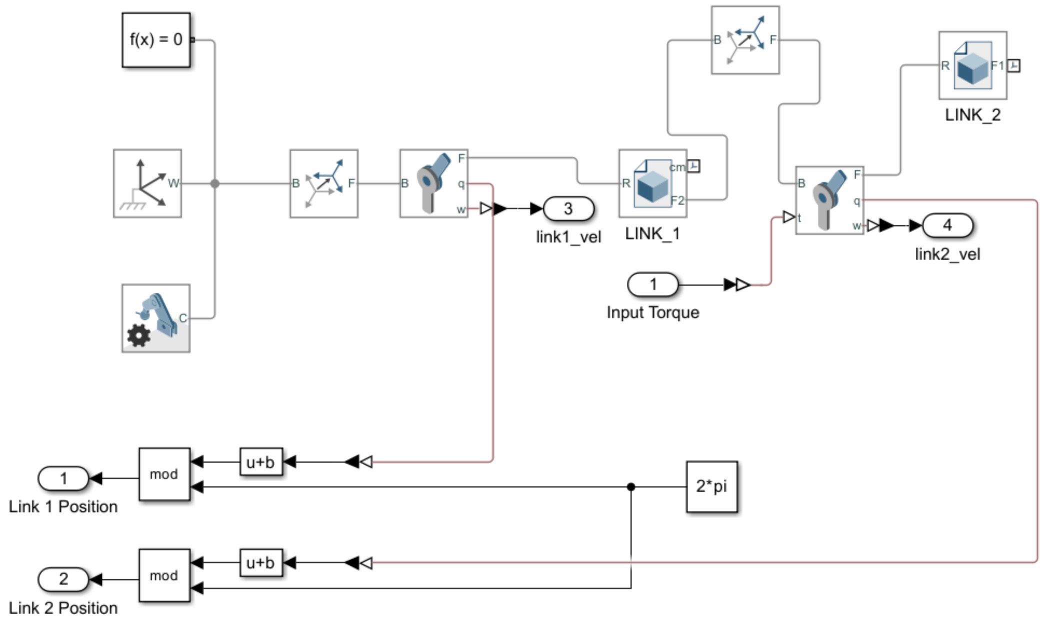

Figure 4.4: Mechanical System Layout

systems. One of the CAD models can be seen in Figure 4.6. These solid-body models include necessary mechanical interfaces, which constitute the revolute joints and have dimensional parameters given in Chapter 2.

It is important to note that the parameters in Chapter 2 are not necessarily in accordance with real world realities, as such, the parameters found in studies [19], [11]. However, the Simscape Multibody environment is capable of simulating real material properties, and hence can be further detailed in order to simulate physically realizable systems.

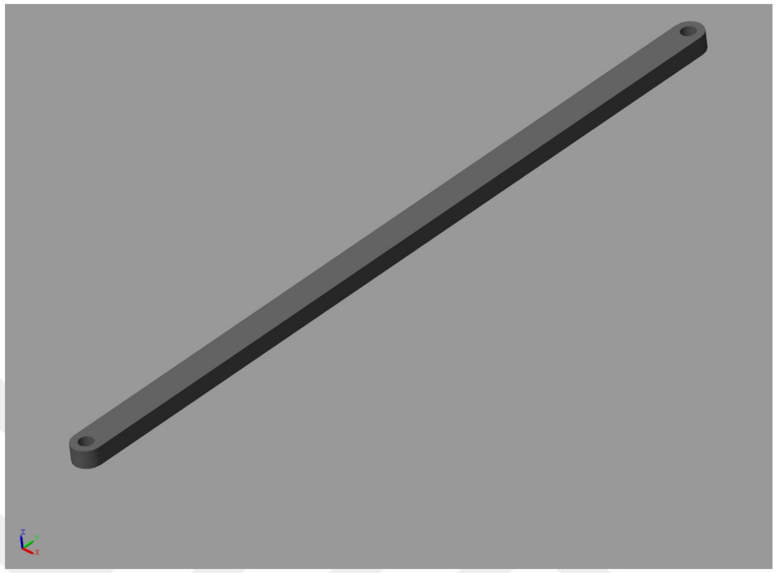

Figure 4.6: CAD Model of One of The Links

In summary, the above-mentioned SIMULINK model provides a viable simulation environment where different types of controllers can be tested and analyzed, especially with time domain responses. In the above simulations, we use the `ode4 (Runge-Kutta)` solver using a fixed step size of 0.0001 sec; this will make sure that the effects of sampling delay are negligible. Using the outputs of this simulation, settling time and total energy consumption values are recorded for different controllers for comparison purposes and also to analyze the convergence to UEP under different initial conditions.

Chapter 5

Comparison of Strongly Stabilizing Controllers

It is imperative to employ a range of performance metrics to facilitate a comprehensive comparison among the suggested controller types. In the course of this study, a meticulous selection has yielded a total of four distinct performance metrics. Subsequent explanations provide a detailed overview of these metrics, shedding light on their individual significance in evaluating and discerning the efficacy of the proposed controller types. By carefully examining these metrics, we can gain a detailed understanding of how the controllers compare in performance. This insight provides valuable information to optimize and select controllers.

5.1 Time Domain Responses

In control theory, the responses of any given feedback system in the time domain constitute an important assessment for the characterization of the system. Well-known metrics for evaluating time response data include, but are not limited to, percent overshoot, settling time, rise time, and peak time. In the case of this study, aside from the settling time metric, total energy consumption until reaching the settling time becomes prominent rather than overshoot-related metrics such as percent overshoot or rise time. Because of that reason, time-domain responses will include total energy consumption and settling time characteristics.

The main reason for not picking overshoot-related metrics is that the acrobot mechanism starts moving from an initial state that is far from the equilibrium point and later oscillates around it. Initial overshootings of these oscillations are not directly related to controller performance, since the system is underactuated and nonlinear. This leaves time and energy related properties much more convenient to assess controller performance.

	Set 1	Set 2	Set 3	Set 4	Set 5
Initial q_1 (deg)	1	1	1	1	2
Initial q_2 (deg)	1	-3	2	3	-2

Table 5.1: Initial Condition Sets

For evaluating the time domain responses of the system, different sets of initial conditions are used. These initial conditions are defined in Table 5.1.

5.1.1 Total Energy Consumption

During the developing phases of this study, attention is drawn to the potential metric of the energy expended until the controlled system is brought to its equilibrium point, primarily due to the lack of stability in the system. Consequently, this metric offers valuable insights, particularly in underactuated systems like the acrobot or pendubot, reaching their 'Upright Equilibrium Point' (UEP), which signifies stability. The energy expended by the controller up to this point provides a holistic understanding of the system's overall oscillatory behavior.

In contrast to the conventional 'percent overshoot' metric commonly employed in classical control systems, the use of total expended energy as a metric introduces a more practical approach for calculating and interpreting excessive oscillations during the transition of acrobot systems. This transition involves moving from an initially unstable state to the stable upright equilibrium point. In this context, the product of the torque applied within a given time step and the resulting angular displacement during that time step serve as a measure of the energy expended in that unit of time.

This approach not only offers a more practical calculation but also enhances the interpretative capability concerning the system's dynamic behavior during these critical transitions. The accumulated energies, considered over the period until the system achieves settlement, contribute to the computation of the total energy expended during the entire process. This metric proves to be particularly insightful in understanding and characterizing the energy dynamics of the system until it reaches a stable state.

$$E_{tot} = \sum_{k=0}^{k=\frac{t_{settling}}{t_{time\,step}}} \tau(k) \cdot (q_2(k+1) - q_2(k)) \tag{5.1}$$

In (5.1), $\tau(k)$ refers to the torque value that the motor applies to the joint between links 1 and 2 during the k^{th} time step. Since the variable q_2 measures the angular displacement between links 1 and 2, the expression $q_2(k+1) - q_2(k)$ infers the value of the angular displacement that had occurred under the motor torque applied between these two links, and hence the product expression of

$$\tau(k) \cdot (q_2(k+1) - q_2(k))$$

amounts to the average mechanical energy output of the motor during the time step k^{th}.

For each set of initial conditions, which are given in Table 5.1, energy consumption values are calculated with (5.1). The simulation setup is explained in detail in Chapter 4. According to the simulation results, the minimal energy consumption values are highlighted in Table 5.2.

Controller Types	E_{tot} for IC Set 1 (J)	E_{tot} for IC Set 2 (J)	E_{tot} for IC Set 3 (J)	E_{tot} for IC Set 4 (J)	E_{tot} for IC Set 5 (J)
C_X	**4.5322**	**0.0941**	**8.3227**	**13.7835**	14.1759
C_H	19.9937	1.5319	30.5846	–	20.3251
C_0	5.0211	0.1020	9.0878	16.6936	14.2460
C_1	5.3360	0.1060	9.5484	20.8025	14.3028
C_2	5.3557	0.5280	9.7168	23.5236	14.2925
C_3	5.8738	0.4926	13.5931	39.5343	14.3402
C_{0PM}	7.2127	0.9762	22.0674	–	15.2078
C_{1PM}	9.6930	0.7540	21.2149	–	12.8782
C_{2PM}	8.7348	0.3654	–	–	**12.4955**
C_{3PM}	9.3388	0.5510	28.3768	–	15.7573
C_{0VM}	6.2872	0.6417	15.4996	–	14.5790
C_{1VM}	7.3319	0.9213	20.0619	–	15.4320
C_{2VM}	6.4938	0.7489	16.8584	–	14.7384
C_{3VM}	7.4220	0.9607	20.2090	–	15.5547

Table 5.2: Simulation Results for Total Energy Consumption

5.1.2 Settling Time

The settling time, commonly used in classical control systems, has also been considered as an evaluation metric in this study. It represents the time taken by the system to reach and stay within a certain threshold value after reaching the steady-state equilibrium point, indicating the absence of oscillations above a specified threshold. In this study, we define the settling time as the duration of the period after which the system does not oscillate beyond *0.01 meters* from the equilibrium point [38].

For each set of initial conditions, the minimum settling time values are highlighted in Table 5.3.

As can be seen in Table 5.2 and Table 5.3, controllers designed with respect to spectral abscissa optimization of the system are observed to have a quicker settling time response together with less energy consumption until being stabilized at UEP. On the other hand, controllers that are designed with respect to phase margin optimization demonstrate slower and less efficient responses.

Controller Types	t_s for IC_1 (sec)	t_s for IC_2 (sec)	t_s for IC_3 (sec)	t_s for IC_4 (sec)	t_s for IC_5 (sec)
C_X	2.8110	0.8691	2.8876	**2.6506**	2.5266
C_H	3.9235	1.2762	5.2904	–	2.6779
C_0	2.6623	0.8699	2.6115	3.2009	2.4511
C_1	2.5864	**0.8673**	2.5077	3.3790	2.40828
C_2	2.5610	1.8580	**2.4810**	3.5987	2.4333
C_3	2.4070	1.9731	3.2638	4.7368	2.3479
C_{0PM}	3.0229	1.5392	3.4612	–	**1.8421**
C_{1PM}	3.4857	1.7237	4.4617	–	1.9361
C_{2PM}	7.8153	2.9948	–	–	6.4575
C_{3PM}	5.1816	1.9019	8.4931	–	4.0795
C_{0VM}	2.2400	1.7977	3.0357	–	2.1700
C_{1VM}	2.0491	1.5729	2.9221	–	1.9737
C_{2VM}	2.1900	1.7586	3.0164	–	2.1234
C_{3VM}	**2.0466**	1.5248	2.9076	–	1.9777

Table 5.3: Simulation Results for Settling Time

5.2 Stability Margins

This part of the study delves into the examination of stability margins, which play a pivotal role in determining the physical feasibility of the controllers. In this context, the investigation encompasses the delay margin, the gain margin, and the phase margin, which are essential parameters that characterize the robustness and stability of control systems.

The comparative analysis of these stability margins among different types of controllers provides valuable insight into their respective physical applicability. By assessing how well each type of controller performs in terms of delay, gain, and phase margins, researchers and engineers can make informed decisions about the suitability of these controllers for practical implementation in diverse real-world scenarios. Especially for acrobot systems, which are very keen to become unstable under uncertain conditions, stability margins indicate the safety factor of the selected controller.

5.2.1 Gain Margin

The gain margin represents the measure of the robustness of a control system against gain variations. It quantifies the amount by which the system's gain can be increased before instability occurs. A higher gain margin indicates a more robust system that can withstand greater variations in gain. For satisfactory performance, it is recommended to obtain a conciderable margin for the gain value [38].

Controller Types	Gain Margin(dB)
C_X	1.1314
C_H	0.6968
C_0	1.2211
C_1	1.2603
C_2	1.3553
C_3	**1.4797**
C_{0PM}	1.2246
C_{1PM}	1.1290
C_{2PM}	0.0099
C_{3PM}	0.0070
C_{0VM}	1.3698
C_{1VM}	1.4016
C_{2VM}	1.4012
C_{3VM}	1.4163

Table 5.4: Simulation Results for Gain Margin

According to Table 5.4, the controller C_3 has the widest gain margin.

5.2.2 Phase Margin

Phase margin is a crucial parameter indicating the system's stability in the frequency domain. It measures the difference in phase between the system's output and the input at the frequency where the gain is unity. A higher phase margin signifies increased stability and a better ability to maintain performance in the presence of disturbances. In [38], Ogata proposes to have the phase margin in between 30 and 60 degrees in order to obtain reliable controller performance for minimum-phase systems. However, in many underactuated cases, the state of the system must move rapidly away from its origin in order to reach the origin later on. If examined further, it would be revealed that linearized closed-loop dynamics of acrobots are in fact non-minimum phase since acrobot has one right-half zero. Thus, it is not convenient to treat acrobot plants in the manner that is for minimum-phase LTI systems. As a result of this fact, the Nyquist plot would be a better tool for understanding the phase margin.

Controller Types	Phase Margin (deg)
C_X	7.6177
C_H	8.7968
C_0	7.8242
C_1	7.9219
C_2	7.7402
C_3	7.7872
C_{0PM}	9.1782
C_{1PM}	9.400
C_{2PM}	14.2674
C_{3PM}	**14.6124**
C_{0VM}	8.4314
C_{1VM}	8.6496
C_{2VM}	8.4229
C_{3VM}	8.6229

Table 5.5: Simulation Results for Phase Margin

According to Table 5.5, the controller C_{3PM} has the widest phase margin. Notice that even the controller with the best phase margin lacks the standard requirements used in linear applications, as presented in [38]. This is mostly due to the non-minimum phase and unstable nature of acrobot systems. Since the controller C_{3PM} has been designed for the best phase margin, it is not surprising to observe that it has the widest phase margin.

5.2.3 Vector Margin

In 5.6, a comparison of controllers is shown according to the vector margin. Unsurprisingly, controllers C_{0VM}, C_{1VM}, C_{2VM}, and C_{3VM} emerge with the best vector margins. Controllers with optimized vector margins perform on average in all other metrics. However, they also do not have inapplicable or risky design characteristics. Therefore, controllers with optimized vector margins can be useful when good performance is desired for all relative stability criteria.

Controller Types	Vector Margin
C_X	0.1221
C_H	0.0771
C_0	0.1311
C_1	0.1351
C_2	0.1346
C_3	0.1315
C_{0PM}	0.1315
C_{1PM}	0.1153
C_{2PM}	5.5017e-04
C_{3PM}	8.0327e-04
C_{0VM}	0.1459
C_{1VM}	0.1499
C_{2VM}	0.1467
C_{3VM}	**0.1502**

Table 5.6: Simulation Results for Vector Margin

5.2.4 Delay Margin

Delay margin refers to the maximum time delay that a control system can tolerate without causing instability. It provides insights into the system's ability to handle delays in its feedback loop, a critical consideration for real-world applications. In reality, hardware systems for real-life applications naturally have a delay. For this reason, the delay margin is an important consideration. In cases where an acrobot mechanism is to be controlled by a time-delayed hardware system, this margin value has to be checked for conformity.

Controller Types	Delay Margin (sec)
C_X	0.0541
C_H	0.0167
C_0	0.0509
C_1	0.0490
C_2	0.0493
C_3	0.0454
C_{0PM}	0.0300
C_{1PM}	0.0341
C_{2PM}	**0.0579**
C_{3PM}	0.0484
C_{0VM}	0.0407
C_{1VM}	0.0303
C_{2VM}	0.0384
C_{3VM}	0.0294

Table 5.7: Simulation Results for Delay Margin

According to Table 5.7, the controller C_{2PM} has the best delay margin. The delay margins given in Table 5.7 have been verified by simulations made in the SIMULINK model that included a variable delay.

5.3 Closed-Loop Pole Locations

The positions of the closed-loop poles, measured by their distance to the imaginary axis, have been considered as an evaluation metric due to their impact on the overall performance and stability of the system. The controllers C_0 to C_3 provided in Chapter 3.2.1 have been optimized with respect to this metric.

According to Table 5.8, the controller C_1 has the maximum spectral abscissa. Note that the search space for the controller C_3 covers the controller C_1, however, the search for C_3 did not yield a closed-loop system with a better spectral abscissa.

Controller Types	Pole Location
C_X	-2.0564
C_H	-0.3825
C_0	-3.176
C_1	**-3.7207**
C_2	-2.5713
C_3	-2.9926
C_{0PM}	-1.0438
C_{1PM}	-0.8088
C_{2PM}	-0.0025
C_{3PM}	-0.0017
C_{0VM}	-1.8470
C_{1VM}	-2.1189
C_{2VM}	-1.3684
C_{3VM}	-1.1900

Table 5.8: Simulation Results for Spectral Abscissa

During the optimization search, the values of c may not coincide with the exact value of z. For this reason, Type 3 controllers did not have pole-zero cancelation. This condition resulted in one more pole having to be moved further to the left, which is harder to achieve. Therefore, controller C_3 could not have a better spectral abscissa.

5.4 Domain of Attraction

In this section, the domains of attraction of different controller types have been examined. The domain of attraction is defined as a collection of initial states, which can satisfy the condition that all possible trajectories starting from each and every member of this collection converge to the equilibrium point. Simulations start with q_1 and q_2 non-zero initial angular positions, and initial angular velocities are taken to be zero. Convergence to the equilibrium is numerically assumed to be the condition given in (5.2).

$$|y(t)| < 0.01 \text{ m} \quad \text{and} \quad |\dot{q}_1(t)| + |\dot{q}_2(t)| < 0.1 \text{ rad/sec} \tag{5.2}$$

The result in this chapter indicates controllers with spectral abscissa based optimization have wider domain of attraction.

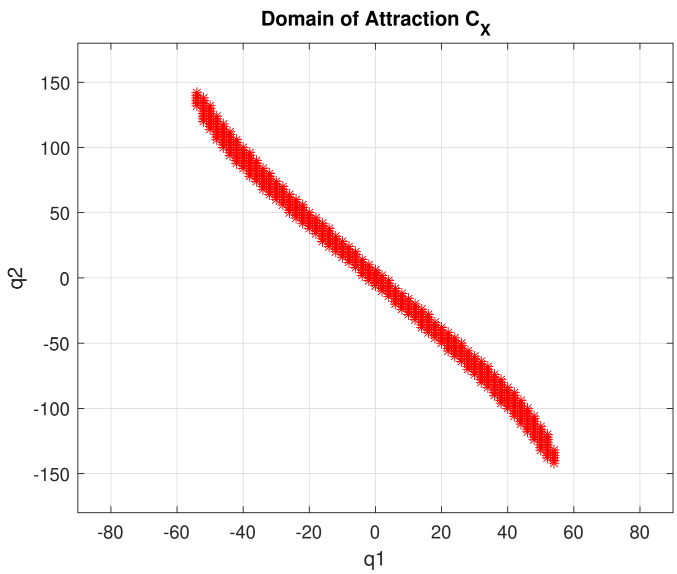

Figure 5.1: Domain of Attraction for C_X

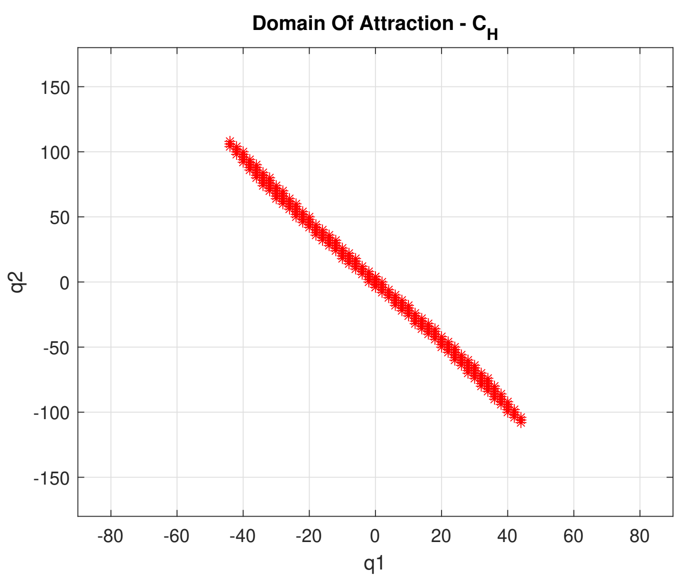

Figure 5.2: Domain of Attraction for C_H

Figures 5.1 and 5.2 are the result of controllers of previous studies. The controller C_X has wider result than controller C_H.

Figures 5.3, 5.4,5.5 and 5.6 are the domain of attraction results for controllers based on spectral abscissa optimization.

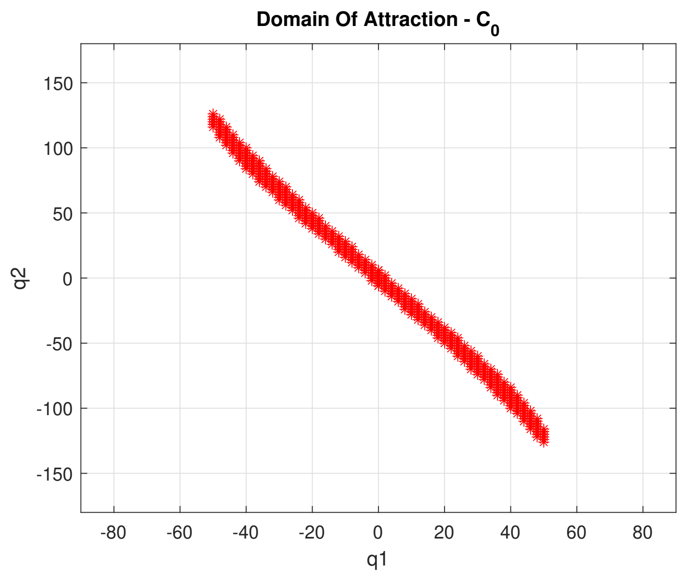

Figure 5.3: Domain of Attraction for C_0

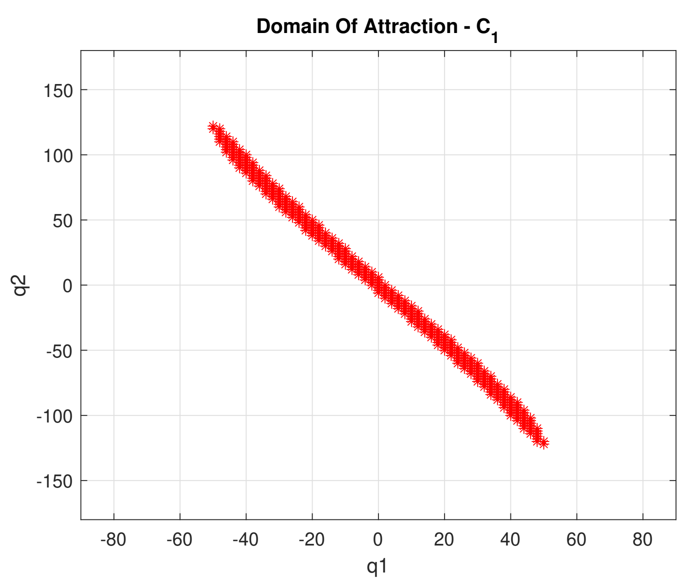

Figure 5.4: Domain of Attraction for C_1

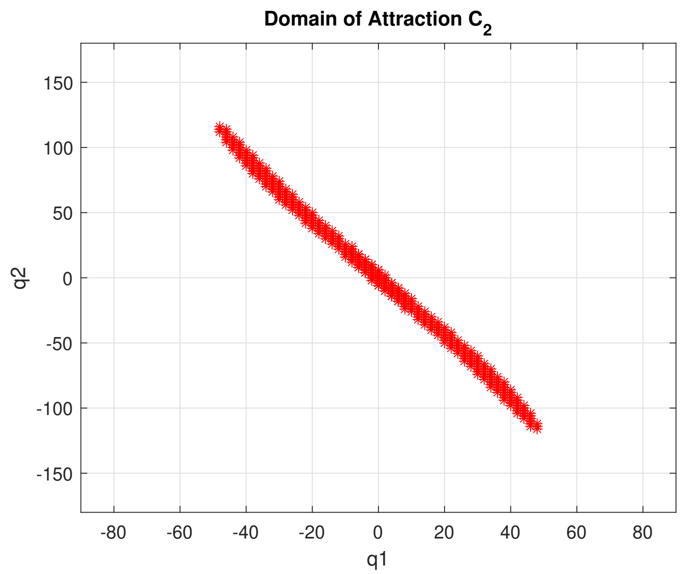

Figure 5.5: Domain of Attraction for C_2

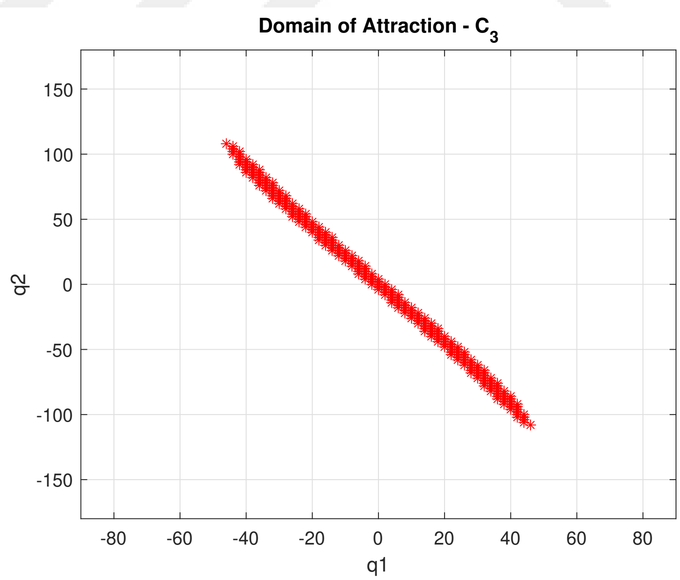

Figure 5.6: Domain of Attraction for C_3

Figures 5.7, 5.8,5.9 and 5.10 are the domain of attraction results for controllers based on phase margin optimization.

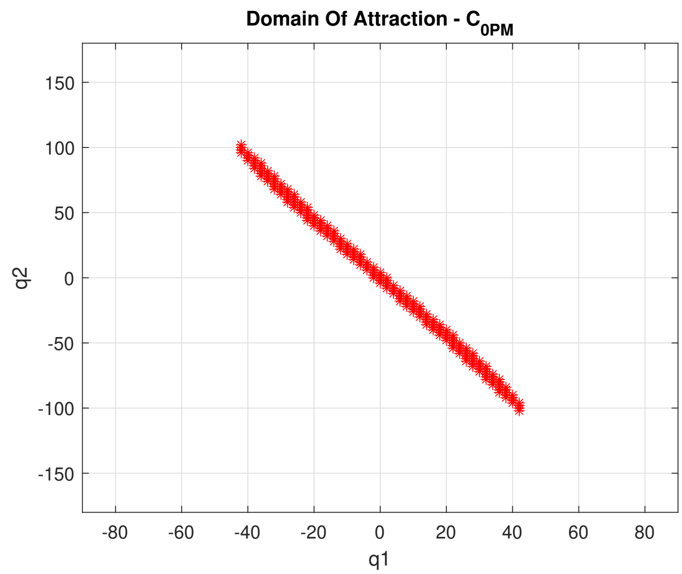

Figure 5.7: Domain of Attraction for C_{0PM}

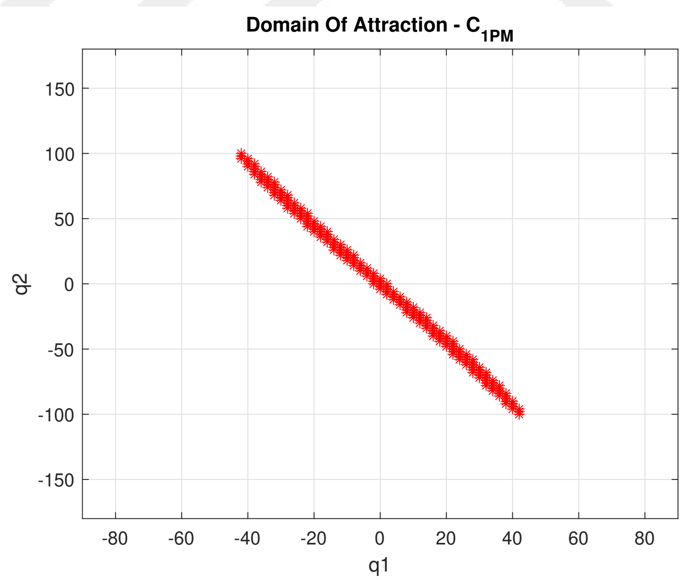

Figure 5.8: Domain of Attraction for C_{1PM}

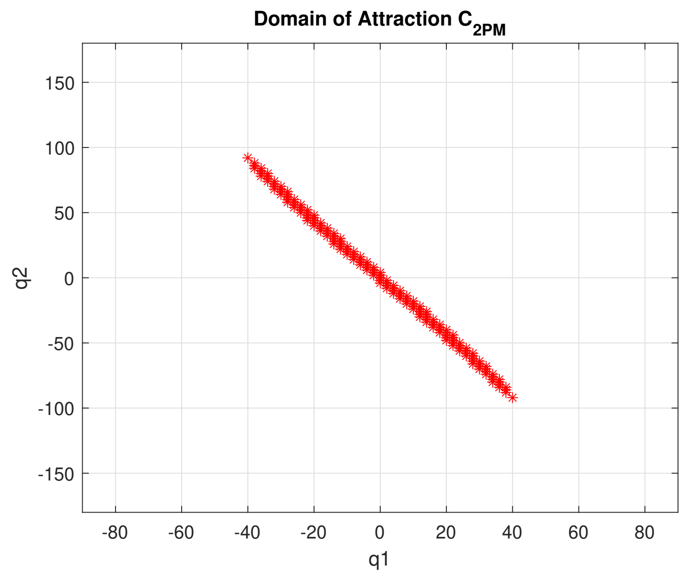

Figure 5.9: Domain of Attraction for C_{2PM}

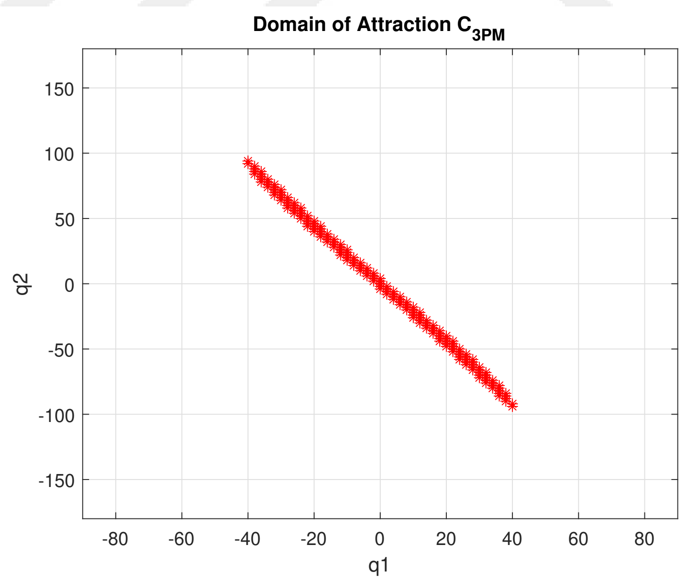

Figure 5.10: Domain of Attraction for C_{3PM}

Figures 5.11, 5.12,5.13 and 5.14 are the domain of attraction results for controllers based on vector margin optimization.

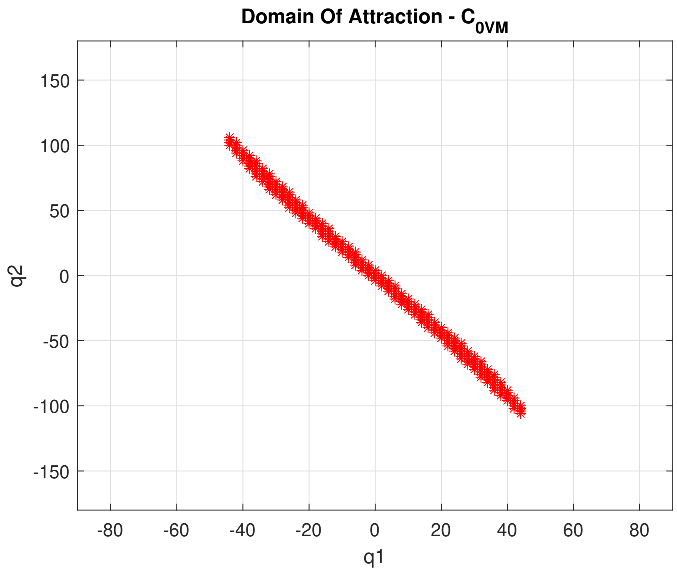

Figure 5.11: Domain of Attraction for C_{0VM}

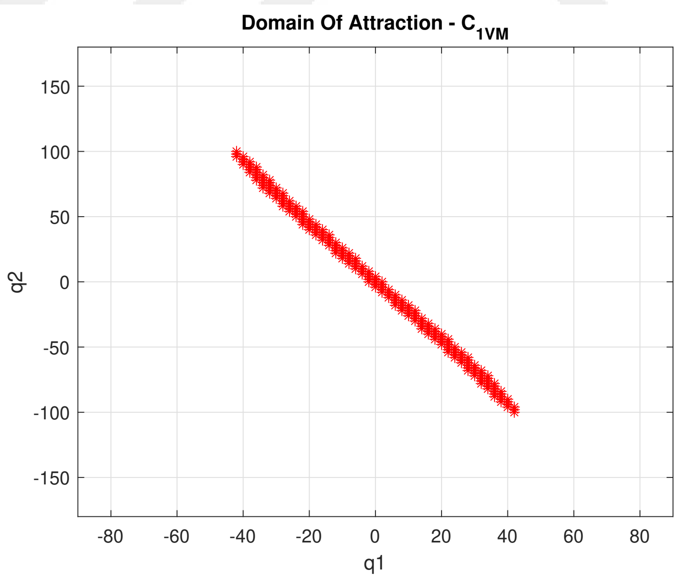

Figure 5.12: Domain of Attraction for C_{1VM}

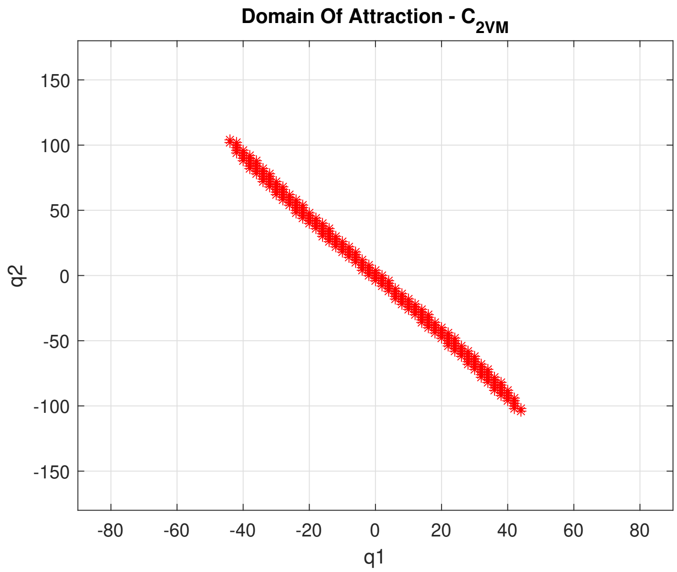

Figure 5.13: Domain of Attraction for C_{2VM}

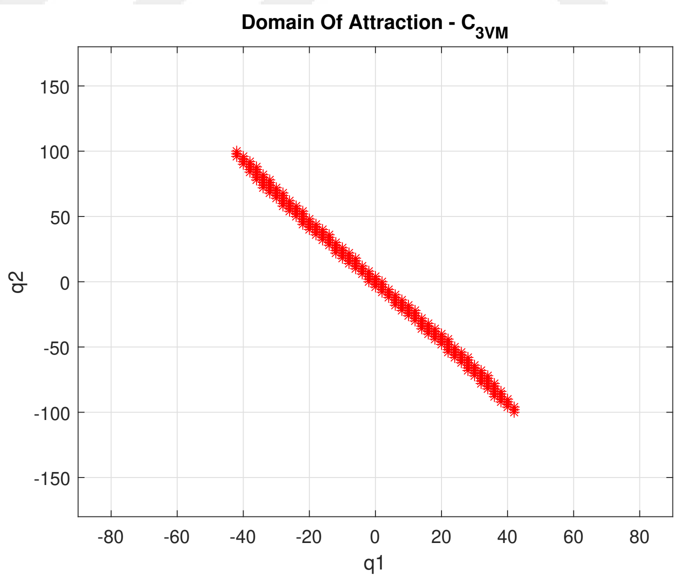

Figure 5.14: Domain of Attraction for C_{3VM}

Chapter 6

Conclusion

In this study, the strong stabilization of a fourth-order plant obtained by linearizing the acrobot system around its UEP balance using second-order controllers has been investigated. The plant has two poles and one zero in the right half-plane, and its relative degree is two. Numerical values indicating that this system satisfies the necessary Parity Interlacing Property (PIP) condition for strong stabilization are obtained from [19]. The first design metric was the spectral abscissa (placing the right-most closed-loop pole as far left as possible). In addition to this method, controller selection is also performed on the basis of phase margin optimization. Later, undesired gain margin variations in the phase margin optimization method revealed the need for an alternative optimization method that can conserve the minimum required values for both stability margins. As a solution to this problem, the vector margin optimization method is proposed and applied. Finally, convergence to the upright equilibrium is analyzed under different initial conditions (investigation of the domain of attraction for the nonlinear model) using simulations.

6.1 Evaluation of Study Results

The results presented in Chapter 5 show that optimization of controllers based on phase margin for acrobot systems produces a longer settling time and less aggressive controller handling. However, this approach has limitations such as diminishing gain margin and narrower domain of attraction. On the other hand, phase differences can be mitigated in a better way. But this improvement on phase and delay margins may not be worth waiving considerable performance on settling time and energy.

Controllers that are designed by optimization of spectral abscissa demonstrate better time-domain responses and domains of attraction with slightly less phase and delay margin than controllers that have optimized phase margins. In all cases, including the works of [19] and [11], the delay margins are no more than $60ms$ and the phase margins are no more than $15degrees$. These values are not ideal for control systems in the classical approach. According to [38], a 'good' control system should have at least 6dB of gain margin and 30 degrees of phase margin. However, it seems to be the case that controlling acrobot systems with reduced-order strongly stabilizing controllers do not reach such large margins (the reason being these are open-loop unstable non-minimum phase systems). All optimizations, whether in the time domain or in stability margins, have to be made within very narrow bands and with great caution. For the purpose of designing a reduced order strongly stabilizing controller for such a system, it is advisable to determine minimum stability margins that will allow safe operation and then perform time-domain optimizations by improving spectral abscissa.

6.2 Future Work

In the future, there are certain topics that can be beneficial for research in the field of underactuated robotics. It is worthy of recommendation to conduct studies that use non-linear control algorithms to stabilize two-link plants similar to

Acrobot systems, as distinct from this study in which local linearization of plant dynamics has been performed. In this way, it is possible to evaluate the differences between linear and non-linear control algorithms.

In addition to non-linear control methods, it is possible to materialize adaptive control algorithms for underactuated robotics. Especially mechanisms similar to Acrobot systems are amenable for improvement by adaptive control techniques, since plant dynamics are relying on defining parameters that continuously change.

Another suggestion could be the application of swing-up problems that can be derived by combining the different control methods mentioned in this study or studies that have been pointed out in related work.

In this study, the delay margin is used for analysis, but not for design purposes. Therefore, a controller design based on vector margin optimization for the acrobot system that includes a certain time delay may be performed.

In terms of initial conditions for the domain of attraction graphs, we have used only position variables with nonzero parameters, while all velocity variables were taken as zero. For future work domain of attraction analysis may include non-zero velocity parameters.

As a final suggestion for possible future work, an experimental hardware-based Acrobot setup for the implementation of different controllers can be built to acquire reliable physical application results that can be examined further.

Bibliography

[1] M. W. Spong, "The swing-up control problem for the acrobot," *IEEE Control Systems Magazine*, vol. 15, no. 1, pp. 49–55, 1995.

[2] V. Yücesoy and H. Özbay, "Design of robustly stabilizing low order, stable controllers for two-link underactuated planar robots," in *2017 IEEE Conference on Control Technology and Applications (CCTA)*, pp. 150–154, IEEE, 2017.

[3] H. Özbay and S. Garg, "Stable H_∞ controller design for the longitudinal dynamics of an aircraft," tech. rep., NASA Technical Memorandum, 1995.

[4] G. Stein, "Respect the unstable," *IEEE Control Systems Magazine*, vol. 23, no. 4, pp. 12–25, 2003.

[5] D. Youla, J. Bongiorno Jr, and C. Lu, "Single-loop feedback-stabilization of linear multivariable dynamical plants," *Automatica*, vol. 10, no. 2, pp. 159–173, 1974.

[6] M. Vidyasagar, *Control System Synthesis: A factorization approach.* MIT Press, 1985.

[7] E. G. Collins, A. Sivaprasad, and M. F. Selekwa, "Insights on reduced order, H_2 optimal controller design methods," in *Proceedings of the 2003 American Control Conference, 2003.*, vol. 6, pp. 5357–5362, IEEE, 2003.

[8] M. W. Spong, "Swing up control of the acrobot using partial feedback linearization," *IFAC Proceedings Volumes*, vol. 27, no. 14, pp. 833–838, 1994.

[9] J. C. Doyle, B. A. Francis, and A. R. Tannenbaum, *Feedback control theory*. Courier Corporation, 2013.

[10] S. Gumussoy and H. Ozbay, "Remarks on strong stabilization and stable H_∞ controller design," *IEEE Transactions on Automatic Control*, vol. 50, no. 12, pp. 2083–2087, 2005.

[11] A. N. Gündeş and H. Özbay, "Strong stabilization of high order plants," *Automatica*, vol. 140, p. 110256, 2022.

[12] D. U. Campos-Delgado and K. Zhou, "A parametric optimization approach to H_∞ and H_2 strong stabilization," *Automatica*, vol. 39, no. 7, pp. 1205–1211, 2003.

[13] P. Cheng, Y.-Y. Cao, and Y. Sun, "Some new results on strong γ_k-γ_{cl} H_∞ stabilization," *IEEE Transactions on Automatic Control*, vol. 53, no. 5, pp. 1268–1273, 2008.

[14] H. Özbay, "Robust control of infinite dimensional systems," in *Encyclopedia of Systems and Control*, pp. 1207–1214, Springer, 2014.

[15] M. C. Smith and K. Sondergeld, "On the order of stable compensators," *Automatica*, vol. 22, no. 1, pp. 127–129, 1986.

[16] H. U. Ünal and A. İftar, "On strong stabilizability of mimo infinite-dimensional systems," *Automatica*, vol. 121, p. 109178, 2020.

[17] H. U. Ünal, "On stable H_∞ controller design for plants with infinitely many unstable zeros and poles," *Automatica*, vol. 138, p. 110036, 2022.

[18] Z. Wang, X. Xin, and Y. Liu, "Strong stabilization of two-link underactuated planar robots," in *2022 China Automation Congress (CAC)*, pp. 4125–4129, IEEE, 2022.

[19] X. Xin and Y. Liu, "Reduced-order stable controllers for two-link underactuated planar robots," *Automatica*, vol. 49, no. 7, pp. 2176–2183, 2013.

[20] M. Zeren and H. Ozbay, "On the synthesis of stable H_∞ controllers," *IEEE Transactions on Automatic Control*, vol. 44, no. 2, pp. 431–435, 1999.

[21] M. Zeren and H. Özbay, "On the strong stabilization and stable H_∞ -controller design problems for mimo systems," *Automatica*, vol. 36, no. 11, pp. 1675–1684, 2000.

[22] J. D. Stefanovski, "Interpolation with strongly f-positive real matrix and application to strong stabilization," *Automatica*, vol. 154, p. 111093, 2023.

[23] L. Wiklendt, S. Chalup, and R. Middleton, "A small spiking neural network with LQR control applied to the acrobot," *Neural Computing and Applications*, vol. 18, pp. 369–375, 2009.

[24] S. C. Brown and K. M. Passino, "Intelligent control for an acrobot," *Journal of Intelligent and Robotic Systems*, vol. 18, pp. 209–248, 1997.

[25] L. Li, Y. Miao, A. H. Qureshi, and M. C. Yip, "Mpc-mpnet: Model-predictive motion planning networks for fast, near-optimal planning under kinodynamic constraints," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4496–4503, 2021.

[26] R. M. Murray and J. E. Hauser, *A case study in approximate linearization: The acrobat example.* Memorandum No. UCB/ERL M91/46, Electronics Research Laboratory College of Engineering, University of California, Berkeley, 1991.

[27] M. D. Berkemeier and R. S. Fearing, "Control of a two-link robot to achieve sliding and hopping gaits.," in *ICRA*, pp. 286–291, 1992.

[28] S. A. Bortoff, *Pseudolinearization Using spline functions with application to the acrobot.* University of Illinois at Urbana-Champaign, 1992.

[29] F. Saito, T. Fukuda, and F. Arai, "Swing and locomotion control for a two-link brachiation robot," *IEEE Control Systems Magazine*, vol. 14, no. 1, pp. 5–12, 1994.

[30] M. Wiklund, A. Kristenson, and K. J. Aström, "A new strategy for swinging up an inverted pendulum," *IFAC Proceedings Volumes*, vol. 26, no. 2, pp. 757–760, 1993.

[31] S. Dubowsky and E. Papadopoulos, "The kinematics, dynamics, and control of free-flying and free-floating space robotic systems," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 5, pp. 531–543, 1993.

[32] E. Papadopoulos and S. Dubowsky, "Coordinated manipulator/spacecraft motion control for space robotic systems.," in *ICRA*, pp. 1696–1701, 1991.

[33] E. Papadopoulos and S. Dubowsky, "Dynamic singularities in free-floating space manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 115, no. 1, pp. 44–52, 1993.

[34] X. Xin, Z. Wang, and Y. Liu, "Parameterization of minimal order strongly stabilizing controllers for two-link underactuated planar robot with single sensor," *Automatica*, vol. 158, p. 111280, 2023.

[35] O. Kolesnichenko and A. S. Shiriaev, "Partial stabilization of underactuated euler–lagrange systems via a class of feedback transformations," *Systems & Control Letters*, vol. 45, no. 2, pp. 121–132, 2002.

[36] X. Xin and M. Kaneda, "Analysis of the energy-based swing-up control of the acrobot," *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal*, vol. 17, no. 16, pp. 1503–1524, 2007.

[37] M. W. Spong and M. Vidyasagar, *Robot dynamics and control.* New York: John Wiley & Sons, January 1991.

[38] K. Ogata, "Modern control engineering," *Book Reviews*, vol. 35, no. 1181, p. 1184, 1999.

[39] H. Ozbay, *Introduction to Feedback Control Theory.* CRC Press, Inc., 1999.

Appendix A

Code

A.1 Optimization of the Spectral Abscissa

A.1.1 Controller Type 0

```
%% CT 0 - CL Pole Optimization
clc; clear all;
%%
% Plant from XinXin, 2013


lambda = -1.3545;
z = 1.281;
p1 = 2.24;
p2 = 6.101;

%       kc(s+p1)(s+p2)              kp(s-z)(s+z)
% C = -----------------  P= -------------------------
%        (s+z)(s+a)            (s+p1)(s+p2)(s-p1)(s-p2)

```

```
%           kg(s-z)
% G = -----------------
%       (s+a)(s-p1)(s-p2)



a_min = 16.8;
a_max = 18.5;
a_iter = 1000;

f_cnt = 1;

a_list = linspace(a_min,a_max,a_iter);
Ka = zeros(length(a_list));

for a_cnt = 1:length(a_list)
    a = a_list(a_cnt);
    GN = conv([1 -z],[1]);
    GD = conv([1 a],conv([1 -p2],[1 -p1]));
    G = tf(GN,GD);
    [R,K] = rlocus(G);

    for g = 1:length(K)
        RR(g) = max(real(R(:,g)));
        if RR(g) >=0
            RR(g) = 0;
        end
    end

    [RR_best,RR_best_ind] =min(RR);
    Ka(a_cnt) = K(RR_best_ind);
    RRa(a_cnt) = RR_best;
end

```

```
figure
plot(a_list,RRa)

[min_val,min_ind] = min(RRa);

sa = a_list(min_ind);
sk = Ka(min_ind);
kcm = sk/lambda;

myControllerN = kcm*conv([1 p1],[1 p2]);
myControllerD = conv([1 z],[1 sa]);
myController = tf(myControllerN,myControllerD);

GN = conv([1 -z],[1]);
GD = conv(conv([1 -p1],[1 sa]),[1 -p2]);
G = sk*tf(GN,GD);

mar = allmargin(G);

figure
rlocus(G)

pole(feedback(G,1))
```

A.1.2 Controller Type 1

```
%% CT 1 - CL Pole Optimization
clc; clear all;
%%
% Plant from XinXin, 2013

```

```
lambda = -1.3545;
z = 1.281;
p1 = 2.24;
p2 = 6.101;

%       kc(s+p1)(s+b)            kp(s-z)(s+z)
% C = -----------------  P= -------------------------
%         (s+z)(s+a)        (s+p1)(s+p2)(s-p1)(s-p2)

%       kg(s-z)(s+b)
% G = -----------------------
%      (s+p2)(s+a)(s-p1)(s-p2)

b_min = 6.0728;
b_max = 6.0734;
a_min = 17.12;
a_max = 17.125;
b_iter = 50;
a_iter = 100;

f_cnt = 1;

b_list = linspace(b_min,b_max,b_iter);
a_list = linspace(a_min,a_max,a_iter);

Kab = zeros(length(a_list),length(b_list));

for b_cnt = 1:length(b_list)
    b = b_list(b_cnt);
    for a_cnt = 1:length(a_list)
        a = a_list(a_cnt);
        GN = conv([1 -z],[1 b]);
```

```
        GD = conv(conv([1 -p1],[1 a]),conv([1 -p2],[1
           p2]));
        G = tf(GN,GD);
        [R,K] = rlocus(G);

        for g=1:length(K)
            RR(g) = max(real(R(:,g)));
            if RR(g) >= 0
                RR(g) = 0;
            end
        end

        [RR_best,RR_best_ind] = min(RR);
        Kab(b_cnt,a_cnt) = K(RR_best_ind);
        RRab(b_cnt,a_cnt) = RR_best;
    end
end


figure
surf(a_list,b_list,RRab)
xlabel('a')
ylabel('b')

[val_list, val_index_list] = min(RRab);
[min_val,min_val_ind_a] = min(val_list);
min_val_ind_b = val_index_list(min_val_ind_a);

final_a = a_list(min_val_ind_a);
final_b = b_list(min_val_ind_b);
final_K = Kab(min_val_ind_b,min_val_ind_a);

sk = final_K;
```

```
kcm= final_K/lambda;
sa = final_a;
sb = final_b;

myControllerN = kcm*conv([1 p1],[1 sb]);
myControllerD = conv([1 z],[1 sa]);
myController = tf(myControllerN,myControllerD);

GN = conv([1 -z],[1 sb]);
GD = conv(conv([1 -p1],[1 sa]),conv([1 p2],[1 -p2]));
G = sk*tf(GN,GD);

mar = allmargin(G);
figure
rlocus(G)
```

A.1.3 Controller Type 2

```
%% CT 2 - CL Pole Optimization
clc; clear all;
%%
% Plant from XinXin, 2013
%Controller Type 2

lambda = -1.3545;
z = 1.281;
p1 = 2.24;
p2 = 6.101;

%       kc(s+p1)(s+p2)              kp(s-z)(s+z)
% C = -----------------  P= --------------------------
%        (s+c)(s+a)            (s+p1)(s+p2)(s-p1)(s-p2)
```

```
%       kg(s-z)(s+z)
% G = ----------------------
%      (s+a)(s+c)(s-p1)(s-p2)

a_min = 17.33;
a_max = 17.341;
c_min = 1.28785;
c_max = 1.288;
c_iter = 110;
a_iter = 200;

f_cnt = 1;

c_list = linspace(c_min,c_max,c_iter);
a_list = linspace(a_min,a_max,a_iter);

Kac = zeros(length(a_list),length(c_list));

for c_cnt = 1:length(c_list)
    c = c_list(c_cnt);
    for a_cnt = 1:length(a_list)
        a = a_list(a_cnt);
        GN = conv([1 -z],[1 z]);
        GD = conv(conv([1 a],[1 c]),conv([1 -p1],[1 -p2
           ]));
        G = tf(GN,GD);
        %G = minreal(G,0.01);
        [R,K] = rlocus(G);

        for g=1:length(K)
            RR(g) = max(real(R(:,g)));
            if RR(g) >= 0
```

```
                RR(g) = 0;
            end
        end

        [RR_best,RR_best_ind] = min(RR);
        Kac(c_cnt,a_cnt) = K(RR_best_ind);
        RRac(c_cnt,a_cnt) = RR_best;

    end
end


figure
surf(a_list,c_list,RRac)
xlabel('a')
ylabel('c')

[val_list, val_index_list] = min(RRac);
[min_val,min_val_ind_a] = min(val_list);
min_val_ind_c = val_index_list(min_val_ind_a);

final_a = a_list(min_val_ind_a);
final_c = c_list(min_val_ind_c);
final_K = Kac(min_val_ind_c,min_val_ind_a);

sk = final_K;
kcm= final_K/lambda;
sa = final_a;
sc = final_c;

myControllerN = kcm*conv([1 p1],[1 p2]);
myControllerD = conv([1 sc],[1 sa]);

```

```
myController = tf(myControllerN,myControllerD);
zpk(myController)

GN = conv([1 -z],[1 z]);
GD = conv(conv([1 sc],[1 sa]),conv([1 -p1],[1 -p2]));
G = sk*tf(GN,GD);

mar = allmargin(G);
figure
rlocus(G)
pole(feedback(G,1))
```

A.1.4 Controller Type 3

```
%% CT 3 - CL Pole Optimization
clc; clear all;
%%
% Plant from XinXin, 2013
%Controller Type 3

lambda = -1.3545;
z = 1.281;
p1 = 2.24;
p2 = 6.101;

%       kc(s+p1)(s+b)            kp(s-z)(s+z)
% C = -----------------  P= -------------------------
%         (s+a)(s+c)           (s+p1)(s+p2)(s-p1)(s-p2)

%       kg(s-z)(s+z)(s+b)
% G = -----------------------
%      (s+a)(s+c)(s+p2)(s-p1)(s-p2)
```

```
c_min = 1.2;
c_max = 1.3;

b_min = 6.01;
b_max = 6.05;

a_min = 15.1;
a_max = 16;

c_iter = 100;
b_iter = 100;
a_iter = 100;


c_list = linspace(c_min,c_max,c_iter);
b_list = linspace(b_min,b_max,b_iter);
a_list = linspace(a_min,a_max,a_iter);

Kcba = zeros(length(c_list),length(b_list),length(
    a_list));

for c_cnt = 1:length(c_list)
    c = c_list(c_cnt);
    for b_cnt = 1:length(b_list)
        b = b_list(b_cnt);
        for a_cnt = 1:length(a_list)
            a = a_list(a_cnt);

            GN = conv(conv([1 -z],[1 z]),[1 b]);
            GD = conv(conv(conv([1 a],[1 c]),conv([1 -
                p1],[1 -p2])),[1 p2]);
```

```
            G = tf(GN,GD);

            [R,K] = rlocus(G);

            for g=1:length(K)
                RR(g) = max(real(R(:,g)));
                if RR(g) >= 0
                    RR(g) = 0;
                end
            end

            [RR_best,RR_best_ind] = min(RR);
            Kcba(c_cnt,b_cnt,a_cnt) = K(RR_best_ind);
            RRcba(c_cnt,b_cnt,a_cnt) = RR_best;
        end
    end
end


[val,ind] = min(min(min(RRcba)));
ind_a = ind;
[val2,ind_b] = min(min(RRcba(:,:,ind_a)));
[val3,ind_c] = min(RRcba(:,ind_b,ind_a));

sa = a_list(ind_a);
sb = b_list(ind_b);
sc = c_list(ind_c);
kg = Kcba(ind_c,ind_b,ind_a);

kc = kg/lambda;

myControllerN = kc*conv([1 p1],[1 sb]);
myControllerD = conv([1 sc],[1 sa]);
```

```
myController = tf(myControllerN,myControllerD);

zpk(myController)

GN = conv(conv([1 -z],[1 z]),[1 sb]);
GD = conv(conv(conv([1 -p1],[1 sa]),conv([1 -p2],[1 sc
   ])),[1 p2]);
G = kg*tf(GN,GD);

mar = allmargin(G);

figure
rlocus(G)

pole(feedback(G,1))
```

A.2 Optimization of Phase Margin

A.2.1 Phase Margin Optimization for Controller Type 0

```
%% CT 0 - PM Optimization
clc; clear all;
%%
% Plant from XinXin, 2013

lambda = -1.3545;
z = 1.281;
p1 = 2.24;
p2 = 6.101;

%       kc(s+p1)(s+p2)                kp(s-z)(s+z)
```

```
% C = ----------------  P= --------------------------
%        (s+z)(s+a)           (s+p1)(s+p2)(s-p1)(s-p2)

%            kg(s-z)
% G = -------------------
%        (s+a)(s-p1)(s-p2)

a_list = linspace(8,13,50);

for ka = 1:50

    GN = conv([1 -z],[1]);
    GD = conv([1 a_list(ka)],conv([1 -p2],[1 -p1]));
    G = tf(GN,GD);

    [GM1,PM1,omgcg,omgcp] = margin(G);

    K = linspace(GM1,abs(p1*p2*a_list(ka)/z),200);

    for k_i = 1:200
        if allmargin(K(k_i)*G).Stable == 0
            temp_PM(k_i) = 0;
        elseif isempty(allmargin(K(k_i)*G).PhaseMargin)
            temp_PM(k_i) = 0;
       elseif length(allmargin(K(k_i)*G).PhaseMargin)
         ==1
            temp_PM(k_i) = -allmargin(K(k_i)*G).
               PhaseMargin(1);
        else
            temp_PM(k_i) = min(-allmargin(K(k_i)*G).
               PhaseMargin(1),allmargin(K(k_i)*G).
               PhaseMargin(2));
        end
```

```
    end

    [PMM(ka),index] = max(PM);
    K_opt(ka) = K(index);

end
    figure
    plot(a_list,PMM)
    xlabel('a values')
    ylabel('Phase Margin')
    title('Max Phase Margin for "a" Values')

    % Calculate Final transfer function

    [max_pm, a_ind_max] = max(PMM);
    sa = a_list(a_ind_max);
    kg = K_opt(a_ind_max);

    kc = kg/lambda;

    myControllerN = kc*conv([1 p1],[1 p2]);
    myControllerD = conv([1 z],[1 sa]);
    myController = tf(myControllerN,myControllerD);

    zpk(myController)
```

A.2.2 Phase Margin Optimization for Controller Type 1

```
%% CT 1 - PM Optimization
clc; clear all;
%%
% Plant from XinXin, 2013
```

```

lambda = -1.3545;
z = 1.281;
p1 = 2.24;
p2 = 6.101;

%       kc(s+p1)(s+b)               kp(s-z)(s+z)
% C = -----------------  P= ---------------------------
%         (s+z)(s+a)            (s+p1)(s+p2)(s-p1)(s-p2)

%             kg(s-z)(s+b)
% G = --------------------------
%         (s+a)(s-p1)(s+p2)(s-p2)

a_list = linspace(12,20,50);
b_list = linspace(6,10,50);
for kb = 1:50
    for ka = 1:50
        GN = conv([1 -z],[1 b_list(kb)]);
        GD = conv(conv([1 a_list(ka)], [1 p2]),conv([1 -p2],[1 -p1]));
        G = tf(GN,GD);

        [GM1,PM1,omgcg,omgcp] = margin(G);

        K = linspace(GM1,abs(p1*p2*p2*a_list(ka)/(z*b_list(kb))),200);

        for k_i = 1:200
            if allmargin(K(k_i)*G).Stable == 0
                temp_PM(k_i) = 0;
            elseif isempty(allmargin(K(k_i)*G).PhaseMargin)
```

```
                temp_PM(k_i) = 0;
            elseif length(allmargin(K(k_i)*G).PhaseMargin)==1
                temp_PM(k_i) = -allmargin(K(k_i)*G).PhaseMargin(1);
            else
                temp_PM(k_i) = min(-allmargin(K(k_i)*G).PhaseMargin(1),allmargin(K(k_i)*G).PhaseMargin(2));
            end
        end
        [PMM(kb,ka),index] = max(PM);
        K_opt(kb,ka) = K(index);
    end
end

figure
surf(a_list,b_list,PMM)
xlabel('a')
ylabel('b')
zlabel('Phase Margin')
title('Max Phase Margin for "a" & "b" Values')


% Calculate Final transfer function

    [max_pm, b_ind_max] = max(max(PMM.'));
    [max_pm_val, a_ind_max] = max(max(PMM));

    sa = a_list(a_ind_max);
    sb = b_list(b_ind_max);
    kg = K_opt(b_ind_max,a_ind_max);
    kc = kg/lambda;
```

```
    myControllerN = kc*conv([1 p1],[1 sb]);
    myControllerD = conv([1 z],[1 sa]);
    myController = tf(myControllerN,myControllerD);
    zpk(myController)
```

A.2.3 Phase Margin Optimization for Controller Type 2

```
%% CT 2 - PM Optimization
clc; clear all;
%%
% Plant from XinXin, 2013

lambda = -1.3545;
z = 1.281;
p1 = 2.24;
p2 = 6.101;

%       kc(s+p1)(s+p2)             kp(s-z)(s+z)
% C = -----------------  P= -------------------------
%        (s+c)(s+a)            (s+p1)(s+p2)(s-p1)(s-p2)

%       kg(s-z)(s+z)
% G = -----------------------
%      (s+c)(s+a)(s-p1)(s-p2)


c_list = linspace(0.8,1.19,50);
a_list = linspace(12,30,50);

for kc = 1:50
    for ka = 1:50
        GN = conv([1 -z],[1 z]);
```

```
        GD = conv(conv([1 -p1],[1 a_list(ka)]),conv([1
           -p2],[1 c_list(kc)]));
        G = tf(GN,GD);

        [GM1,PM1,omgcg,omgcp] = margin(G);
        K = linspace(GM1,abs(p1*p2*c_list(kc)*a_list(ka
           )/(z*z)),200);
        for k_i = 1:200
            if allmargin(K(k_i)*G).Stable == 0
                temp_PM(k_i) = 0;
            elseif isempty(allmargin(K(k_i)*G).
               PhaseMargin)
                temp_PM(k_i) = 0;
            elseif length(allmargin(K(k_i)*G).
               PhaseMargin)==1
                temp_PM(k_i) = -allmargin(K(k_i)*G).
                   PhaseMargin(1);
            else
                temp_PM(k_i) = min(-allmargin(K(k_i)*G)
                   .PhaseMargin(1),allmargin(K(k_i)*G).
                   PhaseMargin(2));
            end
        end
        [PMM(kc,ka),index] = max(PM);
        K_opt(kc,ka) = K(index);
    end
end


figure
surf(a_list,c_list,PMM)
xlabel('a')
ylabel('c')
```

```
zlabel('Phase Margin')
title('Max Phase Margin for "a" & "b" Values')

% Calculate Final transfer function

    [max_pm, c_ind_max] = max(max(PMM.'));
    [max_pm_val, a_ind_max] = max(max(PMM));

    sa = a_list(a_ind_max);
    sb = c_list(c_ind_max);
    kg = K_opt(b_ind_max,a_ind_max);
    kc = kg/lambda;
    myControllerN = kc*conv([1 p1],[1 p2]);
    myControllerD = conv([1 sc],[1 sa]);
    myController = tf(myControllerN,myControllerD);
    zpk(myController)
```

A.2.4 Phase Margin Optimization for Controller Type 3

```
%% CT 3 - PM Optimization
clc; clear all;
%%
% Plant from XinXin, 2013

lambda = -1.3545;
z = 1.281;
p1 = 2.24;
p2 = 6.101;

%       kc(s+p1)(s+b)              kp(s-z)(s+z)
% C = -----------------  P= -------------------------
%        (s+c)(s+a)            (s+p1)(s+p2)(s-p1)(s-p2)
```

```
%          kg(s-z)(s+z)(s+b)
% G = -----------------------------
%       (s+c)(s+a)(s-p1)(s+p2)(s-p2)


c_list = linspace(0.9,1.9,10);
b_list = linspace(6,11,10);
a_list = linspace(10,20,10);

for kc =1:10
    for kb = 1:10
        for ka = 1:10

            GN = conv(conv([1 -z],[1 z]),[1 b_list(kb)
               ]);
            GD = conv(conv(conv([1 -p1],[1 a_list(ka)])
               ,conv([1 -p2],[1 c_list(kc)])),[1 p2]);
            G = tf(GN,GD);

            [GM1,PM1,omgcg,omgcp] = margin(G);

            K = linspace(GM1,abs(p1*p2*p2*a_list(ka)*
               c_list(kc)/(z*z*b_list(kb))),100);
            for k_i = 1:100
                if allmargin(K(k_i)*G).Stable == 0
                    temp_PM(k_i) = 0;
                elseif isempty(allmargin(K(k_i)*G).
                   PhaseMargin)
                    temp_PM(k_i) = 0;
                elseif length(allmargin(K(k_i)*G).
                   PhaseMargin)==1
```

```
                    temp_PM(k_i) = -allmargin(K(k_i)*G).PhaseMargin(1);
                else
                    temp_PM(k_i) = min(-allmargin(K(k_i)*G).PhaseMargin(1),allmargin(K(k_i)*G).PhaseMargin(2));
                end
            end
            [PMM(kc,kb,ka),index] = max(PM);
            K_opt(kc,kb,ka) = K(index);
        end
    end
end

[val,ind] = max(max(max(PMM)));
opt_a = ind;
[val2,opt_b] = max(max(PMM(:,:,opt_a)));
[val3,opt_c] = max(PMM(:,opt_b,opt_a));

sa = a_list(opt_a);
sb = b_list(opt_b);
sc = c_list(opt_c);
kg = K_opt(opt_c,opt_b,opt_a);

kc = kg/lambda;

myControllerN = kc*conv([1 p1],[1 sb]);
myControllerD = conv([1 sc],[1 sa]);
myController = tf(myControllerN,myControllerD);

zpk(myController)
```

A.3 Optimization of Vector Margin

A.3.1 Vector Margin Optimization for Controller Type 0

```
%% CT 0 - VM Optimization
%%
% Plant from XinXin, 2013

lambda = -1.3545;
z = 1.281;
p1 = 2.24;
p2 = 6.101;

%       kc(s+p1)(s+p2)              kp(s-z)(s+z)
% C = -----------------  P= ----------------------------
%        (s+z)(s+a)             (s+p1)(s+p2)(s-p1)(s-p2)

%            kg(s-z)
% G = --------------------
%        (s+a)(s-p1)(s-p2)

a_list = linspace(8,20,50); % for general plot
for ka = 1:50

    GN = conv([1 -z],[1]);
    GD = conv([1 a_list(ka)],conv([1 -p2],[1 -p1]));
    G = tf(GN,GD);

    [GM1,PM1,omgcg,omgcp] = margin(G);

    K = linspace(GM1,abs(p1*p2*a_list(ka)/z),200); % range of stabilizing K values

```

```
    % Vector Margin
    % \beta^(-1) = max ||S(jw)||_{inf}
    % where S(jw) is sensitivity function

    for k_i = 1:200
        if allmargin(K(k_i)*G).Stable == 0
            temp_beta(k_i) = 0;
        else
            temp_beta(k_i) = 1/hinfnorm(1/(1+K(k_i)*G))
               ;
        end
     end
    beta = temp_beta;
    [beta_max(ka),ii] = max(beta);
    K_opt_b(ka) = K(ii);
end

    figure
    plot(a_list,beta_max)
    xlabel('a values')
    ylabel('\beta')
    title('Max \beta for "a" Values')

    %%
    % Calculate Final transfer function

    [max_beta, a_ind_max] = max(beta_max);
    sa = a_list(a_ind_max);
    kg = K_opt_b(a_ind_max);
    kc = kg/lambda;

    myControllerN = kc*conv([1 p1],[1 p2]);
    myControllerD = conv([1 z],[1 sa]);
```

```
    myController = tf(myControllerN,myControllerD);

    GN = conv([1 -z],[1]);
    GD = conv([1 sa],conv([1 -p2],[1 -p1]));
    G = kg*tf(GN,GD);

    mar = allmargin(G);
    mag2db(mar.GainMargin(1));
    figure
    nyquist(G)

    pole(feedback(G1,1))
```

A.3.2 Vector Margin Optimization for Controller Type 1

```
%% CT 1 - VM Optimization

%%
% Plant from XinXin, 2013

lambda = -1.3545;
z = 1.281;
p1 = 2.24;
p2 = 6.101;

%       kc(s+p1)(s+b)               kp(s-z)(s+z)
% C = -----------------  P= ---------------------------
%        (s+z)(s+a)             (s+p1)(s+p2)(s-p1)(s-p2)

%            kg(s-z)(s+b)
% G = -------------------------
%        (s+a)(s-p1)(s+p2)(s-p2)
```

```

a_list = linspace(8,20,50);
b_list = linspace(3,8,50);
for kb = 1:50
    kb
    tic
    for ka = 1:50
        GN = conv([1 -z],[1 b_list(kb)]);
        GD = conv(conv([1 a_list(ka)], [1 p2]),conv([1 -p2],[1 -p1]));
        G = tf(GN,GD);

        [GM1,PM1,omgcg,omgcp] = margin(G);

        K = linspace(GM1,abs(p1*p2*p2*a_list(ka)/(z*b_list(kb))),200);

        % Vector Margin
        % \beta^(-1) = max ||S(jw)||_{inf}

        parfor k_i = 1:200
            if allmargin(K(k_i)*G).Stable == 0
                temp_beta(k_i) = 0;
            else
                temp_beta(k_i) = 1/hinfnorm(1/(1+K(k_i)*G));
            end
        end
        beta_par = temp_beta;
        [beta(kb,ka),ii] = max(beta_par);
        K_opt_b(kb,ka) = K(ii);
    end
    toc
```

```
end

figure
surf(a_list,b_list,beta)
xlabel('a')
ylabel('b')
zlabel('\beta')
title('Max\beta for "a" & "b" Values')


% Calculate Final transfer function

    [max_vm, b_ind_max] = max(max(beta.'));
    [max_vm_val, a_ind_max] = max(max(beta));
    if max_vm_val == max_vm
        sa = a_list(a_ind_max);
        sb = b_list(b_ind_max);
        kg = K_opt_b(b_ind_max,a_ind_max);
        kc = kg/lambda;

        myControllerN = kc*conv([1 p1],[1 sb]);
        myControllerD = conv([1 z],[1 sa]);
        myController = tf(myControllerN,myControllerD);

        GN = kg*conv([1 -z],[1 sb]);
        GD = conv(conv([1 sa], [1 p2]),conv([1 -p2],[1
           -p1]));
        G = tf(GN,GD);
        mar = allmargin(G);

        mag2db(mar.GainMargin(1))

        figure
```

```
        nyquist(G)

        pole(feedback(G1,1))
    end
```

A.3.3 Vector Margin Optimization for Controller Type 2

```
%% CT 2 - VM Optimization
clc; clear all;
%%
% Plant from XinXin, 2013

lambda = -1.3545; %old lr =0.55
z = 1.281; %old lr =0.55
p1 = 2.24;
p2 = 6.101;

%       kc(s+p1)(s+p2)            kp(s-z)(s+z)
% C = -----------------  P= -------------------------
%        (s+c)(s+a)           (s+p1)(s+p2)(s-p1)(s-p2)

%       kg(s-z)(s+z)
% G = -----------------------
%      (s+c)(s+a)(s-p1)(s-p2)


c_list = linspace(1,1.6,30);
a_list = linspace(8,25,30);

for kc = 1:30
    kc
    tic
```

```
    for ka = 1:30
        GN = conv([1 -z],[1 z]);
        GD = conv(conv([1 -p1],[1 a_list(ka)]),conv([1
           -p2],[1 c_list(kc)]));
        G = tf(GN,GD);

        [GM1,PM1,omgcg,omgcp] = margin(G);

        K = linspace(GM1,abs(p1*p2*c_list(kc)*a_list(ka
           )/(z*z)),200);

        % Vector Margin
        % \beta^(-1) = max ||S(jw)||_{inf}
        % where S(jw) is sensitivity function

        parfor k_i = 1:200
            if allmargin(K(k_i)*G).Stable == 0
                temp_beta(k_i) = 0;
            else
                temp_beta(k_i) = 1/hinfnorm(1/(1+K(k_i)
                   *G));
            end
        end
        beta = temp_beta;
        [beta_max(kc,ka),ii] = max(beta);
        K_opt_b(kc,ka) = K(ii);
    end
    toc
end

figure
surf(a_list,c_list,beta_max)
xlabel('a')
```

```
ylabel('c')
zlabel('\beta')
title('Max \beta for "a" & "c" Values')

%%
% Calculate Final transfer function

    [max_pm, c_ind_max] = max(max(beta_max.'));
    [max_pm_val, a_ind_max] = max(max(beta_max));
    if max_pm_val == max_pm
        sa = a_list(a_ind_max);
        sc = c_list(c_ind_max);
        kg = K_opt_b(c_ind_max,a_ind_max);

        kc = kg/lambda;

        myControllerN = kc*conv([1 p1],[1 p2]);
        myControllerD = conv([1 sc],[1 sa]);
        myController = tf(myControllerN,myControllerD);

        GN1 = conv([1 -z],[1 z]);
        GD1 = conv(conv([1 sc],[1 sa]),conv([1 -p1],[1
           -p2]));
        G1 = kg*tf(GN1,GD1);

        figure
        nyquist(G1)

        mar = allmargin(G1);
        mag2db(mar.GainMargin(1))

        pole(feedback(G1,1))
    end
```

A.3.4 Vector Margin Optimization for Controller Type 3

```
%% CT 3 - VM Optimization

%%
% Plant from XinXin, 2013

lambda = -1.3545;
z = 1.281;
p1 = 2.24;
p2 = 6.101;

%       kc(s+p1)(s+b)              kp(s-z)(s+z)
% C = -----------------   P= -------------------------
%        (s+c)(s+a)             (s+p1)(s+p2)(s-p1)(s-p2)

%         kg(s-z)(s+z)(s+b)
% G = ----------------------------
%      (s+c)(s+a)(s-p1)(s+p2)(s-p2)


c_list = linspace(0.1,2,20);
b_list = linspace(4,10,20);
a_list = linspace(8,50,20);

for kc =1:20
    kc
    tic
    for kb = 1:20
        for ka = 1:20

            GN = conv(conv([1 -z],[1 z]),[1 b_list(kb)]);
```

```
            GD = conv(conv(conv([1 -p1],[1 a_list(ka)])
               ,conv([1 -p2],[1 c_list(kc)])),[1 p2]);
            G = tf(GN,GD);

            [GM1,PM1,omgcg,omgcp] = margin(G);
            if abs(GM1 - abs(p1*p2*p2*a_list(ka)*c_list
               (kc)/(z*z*b_list(kb)))) < eps(0.5)
                K = GM1;
            else
                K = linspace(GM1,abs(p1*p2*p2*a_list(ka
                   )*c_list(kc)/(z*z*b_list(kb))),100);
            end

            % Vector Margin
            % \beta^(-1) = max ||S(jw)||_{inf}
            % where S(jw) is sensitivity function

            parfor k_i = 1:length(K)
                if allmargin(K(k_i)*G).Stable == 0
                    temp_beta(k_i) = 0;
                else
                    temp_beta(k_i) = 1/hinfnorm(1/(1+K(
                       k_i)*G));
                end
            end
            beta = temp_beta;
            [beta_max(kc,kb,ka),ii] = max(beta);

            if length(K) == 1
                K_opt_b(kc,kb,ka) = K;
            else
                K_opt_b(kc,kb,ka) = K(ii);
            end
```

```
        end
    end
    toc
end
%%
[val,ind] = max(max(max(beta)));
for ka =1:20
    figure;
    surf(c_list,b_list,beta(:,:,ka))
    xlabel('c')
    ylabel('b')
    zlabel('\beta')
end

opt_a = ind;
[val2,opt_b] = max(max(beta(:,:,opt_a)));
[val3,opt_c] = max(beta(:,opt_b,opt_a));

sa = a_list(opt_a);
sb = b_list(opt_b);
sc = c_list(opt_c);
kg = K_opt_b(opt_c,opt_b,opt_a);

kc = kg/lambda;

myControllerN = kc*conv([1 p1],[1 sb]);
myControllerD = conv([1 sc],[1 sa]);
myController = tf(myControllerN,myControllerD);

GN = kg*conv(conv([1 -z],[1 z]),[1 sb]);
GD = conv(conv(conv([1 -p1],[1 sa]),conv([1 -p2],[1 sc
   ])),[1 p2]);
G = tf(GN,GD);
```

```
mar = allmargin(G);
mag2db(mar.GainMargin(1))

figure
nyquist(G)

pole(feedback(G,1))
```