# Optimization of Non-Linear Robust Controller for Complex Multi-link Robotic System

By

**BDEREDDIN ABDUL SAMAD**

**Thesis submitted in fulfilment of the requirement for the degree of**

**Doctor of Philosophy**

School of Engineering

Cardiff University

United Kingdom

**April 2023**

*In the name of Allah, the Most Gracious, the Most Merciful*
*Peace and blessings be upon our Prophet Muhammad*
*and upon his family and companions*

# ABSTRACT

This research focuses on integrating artificial intelligence and knowledge-based systems to improve the control of a complex multi-link mechanism. The Robogymnast is developed and analysed as a platform to study the intricacies and challenges of a three-link robot system. Through modelling, simulation, and advanced control techniques, the study aims to enhance the overall performance and manoeuvrability of underactuated mechanisms, contributing to advancements in robotics. The linearized mathematical model is employed to explore state space determination in the system. The motion of the robot is represented mathematically using Lagrange equations. However, controlling the movements of the robot gymnast poses challenges due to its nonlinear and multivariate characteristics.

The proposed approach for controlling the three-link Robogymnast robotic gymnast and evaluating its stability is examined and compared with existing methods. It compares the effectiveness of a conventionally configured linear quadratic regulator (LQR) with a hybrid approach that combines fuzzy logic and LQR (FLQR) for stabilizing the Robogymnast. The study investigates the application of LQR and FLQR controllers to the Robogymnast, analysing the system's behaviour in five scenarios, including the original value and distributions of ±25% and ±50%. It also explores factors affecting swing-up control in the underactuated three-link Robogymnast. Additionally, a system simulation using MATLAB Simulink is conducted to demonstrate the impact of factors such as under/overshoot, rise time, and settling time.

A linear quadratic regulator/fuzzy logic controller is employed to stabilize a three-link robotic mechanism. The controller system is optimized using two algorithms: Teaching-Learning-Based Optimization (TLBO) and Particle Swarm Optimization (PSO). The results demonstrate that the TLBO algorithm significantly enhances system stability compared to the conventional PSO algorithm. Specifically, the TLBO algorithm achieves a reduction in the overshoot metric to zero for the first link, 39% for the second link, and 23% for the third link. Moreover, the TLBO algorithm exhibits shorter rising and settling times. Notably, the Integral of Time multiplied by Absolute Error (ITAE) for the first joint is 1.688 with the TLBO algorithm, while it is 2.68 with the PSO algorithm. The ITAE values for the second and third links are approximately 0.3117 and 0.02145, respectively, for both algorithms.

Lastly, a new approach is developed to control the movement of the pendulum system through synchronization, and the performance of the system is investigated using the Robogymnast at Cardiff University. A simulation is created using MATLAB/Simulink to study the system's motion and swinging-up behaviour. The simulation of the Robogymnast and the implementation of the controllers are carried out using MATLAB® and STM32 microcontroller in the C++ program environment, respectively. The similarity of joints' motion in the real system and simulation exhibits error percentages of 30% or less, indicating reliable and accurate results for these joints. The research provides valuable insights into the optimization and design of robotic systems using advanced control techniques and optimization algorithms.

# DEDICATION

********

To

*My father, my role model*

*My blessed mother, the shining star in the dark*

*My beloved wife,*

*My lovely daughters, Ayla and Eline*

*The brightest part of my life*

*My brothers and sister,*

*Thank you for the great love and support*

*My late grandfather, Miloud*

*whose trust and prayers for me to succeed will always be remembered*

*May Allah Almighty have mercy on him*

*To my cherished country*

*‘**Libya**’*

*To all others who know me*

# ACKNOWLEDGEMENTS

**Alhamdullilah**, first, I am grateful to (Allah) the most-high, for enabling me to complete this work and for granting me the strength and opportunity to acquire knowledge.

I would like to offer my special thanks to my supervisor Dr. Fatih Anayi for his invaluable advice, continuous support, and patience during my PhD study, I am deeply thankful for the opportunity to work under his supervision and will always be appreciative of his contribution to my academic growth.

I am deeply thankful to my country, Libya, for the continuous support during my studies, including the Ministry of Higher Education and the Libyan embassy in London. I am forever grateful for the financial support provided throughout my PhD journey.

I would also like to extend my gratitude to all my colleagues at the Wolfson Centre for Magnetics for their support, encouragement, and productive discussions during meetings and seminars. A special thanks to Mr. Andrew Rankmore and the technical team at Cardiff University workshop for their contribution in building the practical aspect of the system.

Bdereddin Abdul Samad

Cardiff, 2023

# LIST OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ABBERVIATION

| | |
|---|---|
| ACE | Automation and Control Engineering |
| ACO | Ant Colony Algorithm |
| ACROBAT | Acrobatic Robot |
| AFLC | Adaptive Fuzzy Logic Control |
| ANN | Artificial Neural Network |
| BA | Bees Algorithm |
| CE | Change in Error |
| DLLIP | Double Link Linear Inverted Pendulum |
| DLRIP | Double Link Rotary Inverted pendulum |
| DOF | Degree of Freedom |
| ECM | Equivalent Centre of Mass |
| FCRs | Fuzzy Control Rules |
| FIS | Fuzzy Interference System |
| FL | Fuzzy Logic |
| FLC | Fuzzy Logic Control |
| FLM | Flexible Link Manipulator |
| FLQR | Fuzzy Linear Quadratic Regulator |
| FPD | Fuzzy Proportional Derivative |
| GSA | Gravitational Search Algorithm |
| $H\infty$ | H Infinity |
| IAE | Integral Absolute Error |
| ITAE | Integral Time Absolute Error |
| IGSA | Improved Gravitational Search Algorithm |
| IMU | Inertial Measurement Units |
| IP | Inverted Pendulum |
| IPS | Inverted Pendulum System |
| ISE | Integral Square Error |
| ITSE | Integral Time Square Error |

| | |
|---|---|
| IWO | Invasive Weed Optimization |
| L-E | Lagrange Equation |
| LMI | Linear Matrix Inequality |
| LQ | Linear Quadratic |
| LQG | Linear Quadratic Gaussian |
| LQR | Linear Quadratic Regulator |
| MCIWO | Modified Chaotic Invasive Weed Optimization |
| MOO | Multi Objective Optimisation |
| NHC | Nonlinear Hybrid Controller |
| NLCT | Non-Linear Control Theory |
| NNs | Neural Networks |
| PC | Personal Computer |
| PD | Proportional Derivative |
| PID | Proportional Integral Derivative |
| PSO | Particle Swarm Optimisation |
| PV | Process Variable |
| Robogymnast | Robot Gymnast |
| SFC | State Feedback Control |
| SLLIP | Single Link Linear Inverted Pendulum |
| SLRIP | Single Link Rotary Inverted Pendulum |
| SMC | Sliding Mode Control |
| SM | Stepper Motor |
| STLC | Small Time Local Controllable |
| TLBO | Teaching Learning Based Optimisation |
| TLLIP | Triple Link Rotary Inverted Pendulum |
| WDM | Wavelength Division Multiplexing |

# LIST OF Symbols

| **Symbol** | **Description** |
|---|---|
| $O_{sh}$ | Overshoot |
| $T_r$ | Rising time |
| $T_s$ | Settling time |
| $U_{sh}$ | Undershoot |
| $e$ | Error |
| $ce$ | Change in error |
| $\theta$ | (*Theta*) Angle |
| g | Gravity |
| iter | Iteration |
| L | Lagrange |
| T | Kinetic energy |
| t | Time |
| V | Potential energy |
| s | Seconds |
| *p.u.* | Per Unit |

# Chapter 1:

# Introduction

## 1.1 Introduction

Significant progress has recently been made in the study of multi-link robotic technology, which involves robots with multiple links or joints that provide increased flexibility and manoeuvrability. These developments have opened up new possibilities for the use of robots in many industries, from manufacturing and assembly to healthcare and transportation. Multi-link robots can perform complex tasks with precision and speed, which makes them valuable assets in many applications [1]. As the technology continues to grow, researchers and engineers are beginning to explore ways to further enhance the capabilities of multi-link robots, including improving their control systems [2], increasing their speed, and making them more intuitive and easier to use. In this era of robotics, multi-link robots hold great promise for transforming the way in which we work and live, offering innovative solutions to some of the world's most pressing challenges [3].

## 1.2 Motivation

Complex multi-link structures with under-actuation capabilities offer valuable opportunities for evaluating and comparing control methods. These structures are characterised by nonlinear behaviour and pose difficult modelling and control problems that are commonly encountered in real-life situations. Studying these systems will allow researchers to develop solutions to address motion problems faced by people with disabilities or limb impairments. By evaluating and optimising different control techniques on these structures, researchers can work towards developing more effective solutions for real-world problems [4].

The triple-link robotic system is also motivated by the desire to push the boundaries of robotics technology. By enabling robots to perform complex physical tasks, the Robogymnast has the potential to revolutionise the sports industry and create new forms of physical activity. This system can be used to train athletes more effectively and to develop new training methods that are not currently possible with traditional training equipment. The Robogymnast also has the potential to be used as an educational tool, teaching students about the principles of robotics, human anatomy, and biomechanics. Through hands-on experience with the system, students can develop a deeper understanding of these concepts and become better equipped to tackle future technological challenges.

The triple-link robotic system also has the potential to impact rehabilitation and physical therapy. By using the system to mimic human motion, physical therapists can create targeted exercises to help patients recover from injuries and improve their overall physical function. In conclusion, the motivation behind the triple-link robotic system (Robogymnast) is to advance the field of robotics, understand human movement and biomechanics, explore new possibilities in sports and physical activity, and impact fields such as rehabilitation and education.

## 1.3 Problem statement

Robotic systems have become increasingly important in many fields, including manufacturing, healthcare, and entertainment. One area of particular interest is the control of multi-link robotic systems, which can be highly complex due to the interactions between the different links and joints. This complexity problem covers area such as:

1. Dynamics: The dynamic behaviour of multi-link robotic systems can be complex, especially when considering the interactions between different links and joints. This can lead to issues in accurately predicting the motion of the system and in controlling its motion.
2. Real-time constraints: In many applications, the robotic system must respond quickly to changes in its environment or to commands from a human operator, which can require the control system to operate in real-time and to be able to handle fast-changing inputs and conditions.
3. System complexity: multi-link robotic systems can be highly complex, especially when considering the number of links and joints, the degrees of freedom of each link, and the interactions between the links. This can make it difficult to design and implement effective control algorithms for the system.
4. Scalability: The complexity of the control problem increases as the number of links in a multi-link robotic system increases. Therefore, it is important to design control algorithms that are scalable, and which can handle systems with a large number of links and degrees of freedom.

## 1.4 Research aim, objectives, and contributions

This research investigates the improvement and analysis of implementing intelligent and model-based control methods for the position control of a complex multi-link mechanism. This research focuses on integrating artificial intelligence and knowledge-based systems. The goal is to utilise modelling, simulation, and control of under-actuated mechanisms.

### 1.4.1 Research objectives

The aim described above is accomplished by achieving the following research objectives:

- To develop an innovative approach for initiating the swinging motion of a three-link Robogymnast system that is affixed to a high bar which rotates freely.
- To validate and evaluate the different proposed controllers experimentally by using them to swing and control the Robogymnast in the downward position during the application of an internal disturbance to each link.
- To use mathematical modelling and proposed controller techniques to simulate and analyse the control system for stabilising the Robogymnast in the inverted configuration.
- To utilize a new method of Teaching Learning Based Optimisation (TLBO) to tune a parameter that modifies the control action, which is simultaneously applied to both of the stepper motors that drive the robot.
- To implement the optimised parameters for swinging-up on the real-time robotic system.
- To analyse and validate the alternative simulation (Simscape) model and also compare the proposed controllers through simulation.

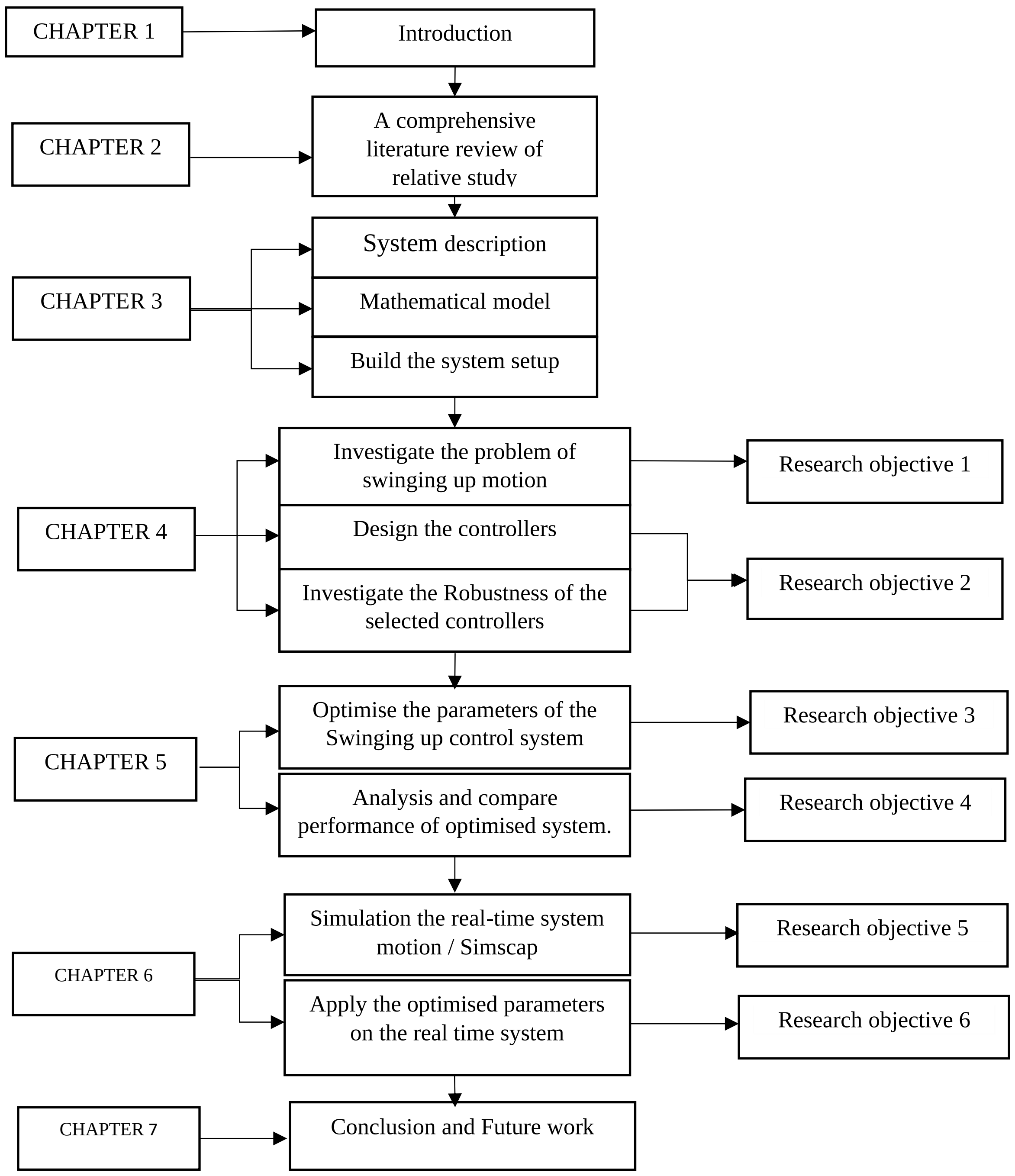

CHAPTER 1
Introduction
CHAPTER 2
A comprehensive literature review of relative study
CHAPTER 3
System description
Mathematical model
Build the system setup
CHAPTER 4
Investigate the problem of swinging up motion
Design the controllers
Investigate the Robustness of the selected controllers
Research objective 1
Research objective 2
CHAPTER 5
Optimise the parameters of the Swinging up control system
Analysis and compare performance of optimised system.
Research objective 3
Research objective 4
CHAPTER 6
Simulation the real-time system motion / Simscap
Apply the optimised parameters on the real time system
Research objective 5
Research objective 6
CHAPTER 7
Conclusion and Future work

Figure 1.1 Thesis Structure and research objectives

### 1.4.2 Contribution to knowledge

The Novelty of this work included:

- Developing a new hybrid linear quadratic regulator fuzzy controller and carry out a comprehensive analysis of the robustness and superiority of the proposed controller for the positioning control of the Robogymnast system.

- Implementing the Teaching Learning Based Optimisation (TLBO) algorithm for the first time in such robotic system, this algorithm is used to tune the parameters of the proposed controller.

- Achieving smooth motion by utilizing the stepper motors for the first time in this specific application through synchronizing and examining the movement.

## 1.5 Methodology

The following methodology will be used to achieve the proposed objectives:

- A comprehensive review of the literature will be carried out to identify the key requirements and challenges in controlling complex multi-link mechanisms. This review includes an extensive survey of the current state of the art and examines the various control methods that have been implemented and analysed. The objective is to gain a better understanding of the problems that have previously been encountered in controlling these mechanisms.
- The Euler-Lagrange method will be used to formulate a mathematical model and dynamic equations for the Robogymnast at the stable equilibrium point.
- The study will focus on maintaining swinging for the Robogymnast in an its position. A conventional LQR and hybrid FLQR controller will be developed, and its parameters are determined to validate the performance of the system, which will be evaluated through MATLAB® simulations to position the system to obtain the best possible response. The selected FLQR will be investigated in five scenarios to validate its stability.
- A modern triple-link robotic system (Robogymnast) will be redesigned and built using the most recent tools in the manufacturing world (e.g., aluminium and SLS materials).
- The motion of each link of the Robogymnast system will be synchronised for smooth motion using a geared stepper motor, which is a new method.
- The swing-up control simulation will be carried out using MATLAB® software and its accompanying toolboxes. The parameters will be optimised using the TLBO algorithm and the results will then apply to the real system through an STM32 microcontroller in a C++ programming environment.
- The Python application will be used to present and filter the accurate real-time data from the system via a header rotary encoder for the free first link, and two potentiometers for the middle and lower links.
- The outcomes of the simulation and real-time system will be finally investigated and compared.

## 1.6 Thesis outline

Table 1.1 The layout of this thesis

| **Chapter Number** | **Description** |
|---|---|
| **1** | This chapter provides a background of the topic, the aim, objectives, contributions, and published works. |
| **2** | This chapter presents a thorough and up-to-date comprehensive literature review on complex multi-link robotic systems, with a focus on stabilisation control problems and swing-up control strategies, as well as optimisation techniques related to the system. This chapter also provides an assessment of each section and gives a brief overview. |
| **3** | Chapter 3 introduces the system description and mathematical modelling of the Robogymnast. The overall system is discussed and illustrated using Figures and diagrams. The design of the system (Setup) is discussed, including the components and prototype of the system. |
| **4** | The fourth chapter explains the novel design and implementation of the new structures for a fuzzy logic controller combined with LQR, which are equipped as FLQR systems in the testbed systems. The robustness of these configurations has been examined and both controllers are compared against a wide range of parametric uncertainties of the investigated systems. |
| **5** | This chapter establishes the selected algorithms (i.e., teaching learning-based optimisation (TLBO) and particle swarm optimisation (PSO)) and provides an overview of the current state of the art for these algorithms. It also gives an understanding of the fundamental concept of both algorithms, the theoretical analysis carried out and its main applications. Finally, it compares a TLBO as a new application of the proposed system with a conventional PSO algorithm to find the best possible performance of the proposed controller. |
| **6** | This chapter presents the results of the simulation and real-time testing, and then performs a comparison to assess the performance of the system. The comparison is used to verify the initial and optimised findings for each link of the multi-link motion system. |
| **7** | The last chapter summarises the main points of this thesis and outlines potential avenues for future research. |

## 1.7 Publications

*Journal*

1. **B. A. Samad,** M. Mohamed and F. Anayi, "Enhanced the Control Strategy of a Triple Link Robotic System (Robogymnast)," in IEEE Access, vol. 11, pp. 31997-32005, 2023, doi: 10.1109/ACCESS.2023.3262190.
2. **B. Abdul Samad**, M. Mohamed, F. Anayi, and Y. Melikhov, "An Investigation of Various Controller Designs for Multi-Link Robotic System (Robogymnast)," *Knowledge*, vol. 2, no. 3, pp. 465–486, Sep. 2022, doi: 10.3390/knowledge2030028.
3. M. Mohamed, **B. A. Samad**, F. Anayi, and M. Packianather, "Analysing Various Control Technics for Manipulator Robotic System (Robogymnast)," *Computer, Materials & Continua*., 2023, doi.org/10.32604/cmc.2023.035312.

*Conferences*

1. **B. A. Samad**, F. Anayi, Y. Melikhov, M. Mohamed and E. Altayef, "Modelling of LQR and Fuzzy-LQR Controllers for Stabilisation of Multi-link Robotic System (Robogymnast)," *2022 8th International Conference on Automation, Robotics and Applications (ICARA)*, 2022, pp. 33-38, doi: 10.1109/ICARA55094.2022.9738577.
2. **B. A. Samad**, M. Mohamed, F. Anayi and Y. Melikhov, "A hybrid Fuzzy approach of different controllers to stabilize a 3-link swinging robotic (Robogymnast)," *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, 2022, pp. 2432-2437, doi: 10.1109/ICACITE53722.2022.9823768.
3. **Abdul Samad, B.;** Mohamed, M.; Anayi, F. Motion planning of triple links robotic system, *in Proceedings of the 3rd International Electronic Conference on Applied Sciences,* 1–15 December 2022, MDPI: Basel, Switzerland, doi:10.3390/ASEC2022-13774.
4. **B. A. Samad**, M. Mohamed, and F. Anayi "  Triple link robotic system (Robogymnast): A comprehensive overview," *2023 3rd International Conference on Advance*

*Computing and Innovative Technologies in Engineering (ICACITE)*, 2023, *in press.*

5. M. Mohamed, F. Anayi, M. Packianather, **B. A. Samad**, and K. Yahya, "Simulating LQR and PID controllers to stabilise a three-link robotic system," in *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, Apr. 2022, pp. 2033–2036. doi: 10.1109/ICACITE53722.2022.9823512.

# Chapter 2:

# Literature review

## 2.1 Preliminaries

The second chapter of this thesis investigates literature on control of the system's upswing, downwards synchronisation and balance, and draws comparisons with earlier studies. In addition, the high-complexity multiple-link mechanical system is described and explored and approaches to solving the challenge of inverted-pendulum based mechanisms are discussed. Further, the approaches to swing-up control used in the controller are examined, alongside optimisation algorithms for this challenge. The majority of studies discuss the problems listed above in relation to the methods they apply, and the objectives set.

As outlined in Chapter One, the literature review chapter will take the following structure: and in-depth background to the research reviewed is given in section 2.2; the section 2.3 describes the approach of stabilisation control problem with previous related works. and the subsequent section examines literature related to the study subject regarding swing-up in multi-link robots. Then section 2.5 provides the overview of the mechanism of complex link robotic systems and its components. section 2.6 presents important information regarding the various structural components which control robotic systems, with a particular focus on inverted pendulum-based mechanisms. section 2.7 considers various approaches to optimising control. Finally, section 2.8 summarises the chapter's main content.

## 2.2 Background

The Inverted Pendulum Systems (IPSs) are often used for education and research related to Non-Linear Control Theory (NLCT) in representations of unstable, underactuated mechanical systems [5]. The IPS has long been a focus in control systems engineering because it can be widely applied in different areas [6]. The pendulum system has been a commonly used experimental setup for education and research in the fields of robotics and control theories in recent years [7]. Inverted pendulums are directly applied within segways, while extended systems can be implemented in design and in forming models of complex types of system, such as walking on two legs, robot-based manipulation approaches, and guiding missiles, for example [8]. Continued advancement in the technology of controllers has led to developments in many areas of science and technology. Effective evaluation of a control system is achieved through developing a controller for application with a given system, and assessing how the system then performs. There is extensive research examining inverted pendulum dynamics [9], and experimental work on controls applying rail-cart structures to study this issue [10][11]. Recently however, pendulum systems which swing, including the Robogymnast and Acrobat, have attracted research attention because they relate to the problem of walking in robotic mechanisms [12][13].

The pendulum can be described as a unit in suspension from a set point which is thus capable of swinging in a free manner as subject to gravitational forces. Pendulums have frequently been applied in the regulation of movement [14], yet due to the nonlinearity which they provide, have continued to be useful models for study and to demonstrate many novel concepts being developed within nonlinear control research: e.g., pendulum swing up and catch. In addition, the pendulum is ideal for demonstrating hybrid or chaotic systems [15]. Various research has been performed looking at non-linear controls in which approaches are tested with 2- or 3- link pendulums.

The Robogymnast is a highly complex triple-link underactuated mechanism, and as such is suitable as a case study for evaluating and comparing control system methodologies [16]. Control of underactuated systems presents difficulties because they may lack full-state feedback linearizability about a set equilibrium point, and additionally may not represent a small-time local controllable (STLC) system [17]. Various researchers in control engineering and robotics have focused on solving this problem [18]. Inverted pendulums, suspended from a set location and which swing freely under the force of gravity, are applied to exemplify underactuated systems, Underactuated mechanisms are often demonstrated using the example of the inverted pendulum, in which a freely swinging object hangs from a fixed point, with a gravitational force acting on the object, and used for movement control and in hybrid/chaotic systems [19][20].

The problem of balancing triple-inverted pendulums is a major focus of robotics, particularly as it is analogous to human structural features and balancing systems. The unstable underactuation based acrobat robotic system mimics human acrobatic movements through its inverted pendulum structure, and thus is suitable for theory based and experimental studies of non-linear controls [18][21]. Through developing an intelligence-based control system which combined conventional fuzzy and adaptive-fuzzy controllers, this robot was able to achieve swing, catch, and balance in inversion [22]. Controllers were based on linear quadratic regulation, state variable feedback and proportional-integral-derivative (PID) methods.

Such non-linear systems contain challenges when attempting to model or control them. This literature chapter will address different types of systems movements, namely swing up, balancing and both of these together. It will also consider a range of approaches to optimising control parameters. This research specifically considers swing up in the non-linear, triple-link Robogymnast system [23] discussing design, construction, and control for the robot.

## 2.3 Stabilisation control approach

Controlling stabilization in a rotating inverted pendulum mechanism is a complex problem, as it requires the simultaneous stabilisation of the underactuated (non-stable) and each actuated state variable. A range of stability controller approaches have been put forward in previous work, as described in this section.

In 1995, Spong and Block. [24], conducted a study on a double-link pendulum attached to a fixed position, with an actuator powering only link one but not link two, thereby creating the Pendubot, a two-link underactuated planar revolute robotic system. This robot is widely utilized for educational and research purposes in ACE. A linear state feedback controller was developed to stabilize the system by linearizing the equations of motion around a point of equilibrium. Partial feedback linearization was utilized to achieve the swing-up of the Pendubot.

Wang’s 1996 work Li-Xin Wang [25] proposed a stable adaptive fuzzy controller in order to implement tracking in SLLIP. The approach is designed to allow the pendulum to maintain its upwards unstable equilibrium position, and fuzzy rules were used to build the controller. Online adjustments are made to fuzzy parameters based on laws of adaptation, in order to control the plant in tracking a specific path. The findings of simulations show the effective performance of the adaptive fuzzy controller in tracking.

Another 1996 study, by Cheng et al. [26], proposed a highly accurate FLC for stabilising DLLIP while upright, in which optimal and fuzzy control theories are combined to achieve the composition coefficient, to develop a fuzzy controller capable of working at high resolutions. The controller was successfully tested in practical experiments with the DLLIP.

In 1998, Eltohamy and Kuo [27] proposed a TLLIP feedback controller using one input and applying a method of non-linear optimisation: a necessary approach due to the inability of the conventional linear approach to designing controllers to integrate the system's physical constraints and non-linearity. The controller was successful in stabilising the TLLIP around its vertically upright stance. This was achieved through developing knowledge of the variables which affect control in this instance. The findings reported suggest that linear controllers are not sufficiently effective in stabilising the robot.

In a 2002 study by Aracil, Gordillo and Acosta [28], a method for producing stabilised, robust oscillation about the (SLRIP) Furuta pendulum's vertical upwards position is put forward. A control law from among the energy shaping approaches pushes the robot into a stable limit cycle. The findings are confirmed experimentally as well as in simulated form.

Nasir et al.'s 2010 [29] research involved the development of traditional PID and novel sliding mode control (SMC) controllers for application with an SLLIP. The two approaches were each effective controls for stabilising the robot. When comparing the approaches in terms of time specifications, the SMC outperformed the PID controller, judged on the basis of the findings from simulation. The SMC controller was therefore found to perform more effectively than the conventional approach.

In 2010, Kizir and Bingül [30] addressed issues of swing up and stability in an SLLIP through experiment, testing a range of controllers in this setting. A fuzzy logic controller was applied for pendulum up-swing, while the PID controller stabilised it at its point of unstable equilibrium. The researchers were successful in applying fuzzy logic and full-state feedback for control of the link in its upwards position. The controllers were confirmed to be robust in the experiments and also in a simulated environment.

Zhang and Zhang's 2012 investigation [31] involved development of a self-adjusting LQR controller for stabilisation of a planar double inverted pendulum-based robot. The controller proposed uses an optimisation factor to enhance the way the system is controlled. The study's finding show that rapid responses, robust performance, and stabilisation were achieved by the controller across various operational conditions tested.

In 2013, Li's [32] master's thesis examined control of stabilising action in a DLRIP, implementing a controller based on LQR, and analysing stabilisation using the Lyapunov technique. The author designed direct adaptive fuzzy control for enhancement of controller performance. Based on findings from simulating two algorithmic controls, it was found through comparative analysis of the simulations that the Adaptive Fuzzy Logic Controller (AFLC) enhanced the effectiveness of the LQR controller and made the DLRIP more robust.

Glück et al. [33] in 2013 explored challenges in up-swing and in stabilising a TLLIP, using practical experiment to test non-linear feedforward and optimal feedback controllers to address up-swing. Stabilisation along a given path was approached by developing a time variant Riccati controller, applying an Extended Kalman Filter (EKF) for estimation of states which could not be measured.

Researchers in [34] report the resolution of the problem of controlling swing-up and reaching stability in a robot based on a rotary inverted pendulum through the use of novel types of controls. The up-swing approach in these methods is developed based on planned trajectories and inertia effects in order to enable the swing of the pendulum to reach the point at which stabilisation controls are triggered. The stabilisation controller uses a non-linear adaptive neural network approach, as well as linear matrix inequation.

Susanto et al.'s (2020) [35] study implements self-erected inverted pendulum control through 2 techniques of switched controls: first, rules-based fuzzy controls to manage the upswing of the pendulum component from a downwards to an upwards vertical position; and second, optimised state feedback control to stabilise the pendulum in its upwards position near to the point of vertical equilibrium. The researchers were seeking solutions to the significant challenges of upswing and stabilisation in the upwards vertical balanced position for the self-erected inverted pendular mechanism. They demonstrated through experiment and simulated findings that these control approaches achieved both up-swing and stabilisation at only small impulses and levels of pulse disturbance.

Researchers in [36] put forward work which synthesised and implemented an automated self–tuning regulator to control an actual inverted pendulum. Their control approach was primarily based on strategies for controlling swing–up, utilising the inverted pendulum's energy, and regulating stabilisation through an LQR. Due to the inability to precisely determine each value for variables of the inverted pendulum, the authors put forward an automatic self–

tuning system for the controls developed, developing a process for determining parameters. The system as a whole allows up-swing to be achieved and stabilises the pendulum in its upwards position. Validation of the effectiveness of the controls designed was performed by simulating it application through MATLAB /Simulink using a model of an inverted pendulum, and also in experiments with a cart-mounted inverted pendulum structure.

The part describes several stability controller approaches proposed in previous studies to control the stabilization of various types of pendulum-based robots. Different methods have been developed, including linear state feedback controller, adaptive fuzzy controller, fuzzy logic controller, sliding mode control, LQR controller, direct adaptive fuzzy control, time variant Riccati controller, and non-linear adaptive neural network approach. The findings of these studies showed that the developed controllers are successful in stabilizing the pendulum-based robots in both simulation and practical experiments. In general, the approaches used in the studies are effective, and the controllers developed have shown significant improvements in the stabilization of the pendulum-based robots.

## 2.4 Swing-up control problem

Controlling swing-up in pendulum-based systems has been widely studied as a frequent subject of interest. The challenge lies in identifying and tracking a possible trajectory for this action which is in line with boundary constraints while minimising the actuator's work on the base [37]. Based on prior studies, various methods have been proposed to control swing-up, such as feedback or feedforward control, control based on predictions from nonlinear modelling, optimal trajectory, and energy shaping. This section highlights significant existing work on swing-up control in underactuated inverted pendulum systems.

Furuta et al.'s 1992 study [38] involved a novel algorithm for a bang-bang feedback controller to achieve pendulum swing-up to the upwards stance. This control system utilises a conventional LQ approach for maintaining position at the pendulum's point of unstable equilibrium. Findings from experiment demonstrate the robustness of the approach where there are uncertain systems parameters for computation in feedforward control.

An approach to moving a double pendulum to unstable equilibrium from stable equilibrium was put forward in 1993 by Yamakita, Nonaka and Furuta [39], who implemented the method experimentally with DLRIP. This approach showed its effectiveness when addressing problems of controlling swing-up. The authors applied a learning control approach for making modifications to feedforward control. Their methods in the study combined feedback and feedforward control. Robustness is not achieved however when applying feedforward control, and it is necessary to repeat the learning process if any systems parameter is altered.

Notably, Spong (1995) put forward a method using partial feedback linearisation, while the issue of controlling swing-up concerns moving the Acrobat, starting in a stable downwards vertical, to reach unstable inversion before balancing around this vertical position. This issue presents significant challenges based on the extent of the range of movement [19].

Yamakita, Iwashiro, Sugahara and Furuta's 1995 research [40] put forward effective approaches to swing-up in a double-link pendulum between different states of equilibrium. The study used two approaches to controlling swing-up. The first was an approach which relies upon energy function; while the second utilises a control to create in the system. the approaches put forward showed effectiveness when implemented in experiment using DLRIP.

Work in 1997 by Yasunobu and Mori [41] developed and applied fuzzy control methods utilising a human control approach and implementing this with an SLLIP. The developed controller was used with a system where the characteristics of the inverted pendulum were not all known, and modelling of swing-up and stability controls used FLC. This human control-based fuzzy control approach was found to be effective both in simulated and experimental contexts.

Aström and Furuta's [15] 2000 have investigated swing-up approaches used energy control with an SLRIP, and the authors provided explanations of findings across a range of scenarios. A study two years later by Rubi, Rubio and Avello [37] considered the issue of swing-up in a DLLIP. The researchers put forward a method for designing controlled trajectories in non-linear systems with under actuation. They determined a reference trajectory through optimising the initial trajectory, set by spline interpolation. Tracking of the reference was done by a gain scheduling linear-quadratic optimal controller which had been developed for this sole purpose.

Gordillo et al. [42], in 2003 reported solving the challenge of swing-up for a Furuta's pendulum through the application of Fradkov's speed-gradient (SG) technique with a 4-

dimensional system model, and then compared the novel law against the established Åström-Furuta method, which uses a 2-dimensional model. Comparative analyses are reported based on experimental work as well as simulation, demonstrating the robustness and benefits of the new law applied to pendulum swing-up

Graichen et al. [43] 2007 study considers swing-up and stabilisation in a DLLIP, using non-linear feedforward to address swing-up. Stabilisation is controlled through a controller based on linear feedback. Each controller was tested through implementing them in practice with a DLLIP.

A 2007 investigation by Xin and Kaneda [44] examined control of swing-up in a triple link planar robotic acrobat system in the vertical plane, having an initial passively operating joint and further active joints. The authors viewed the second and third links as almost forming one combined link, with the researchers proposing coordinating transformation for the active joints' angles. A new Lyapunov function was presented as linked to this transformation, and swing-up control based on energy was developed. Moreover, the study included overall motion analysis for the robotic system when using the controller, as well as identifying various conditions for control parameters in order to control swing-up effectively.

Doung et al. [45] in 2009, focuses on the use of a neural network (NN) and a genetic algorithm (GA) for controlling a three-DOF planar underactuated manipulator, commonly known as the three-link gymnastic robot. The article examines how constraints applied to the joint angles can make the problem more analogous to human gymnastics. The proposed controller is evaluated using various swing-up timings to investigate its performance, and control simulations are conducted. The results of numerical simulations demonstrate that the neurocontroller can effectively control the system within the given timings and constraints. On the other hand, the most significant disadvantage of this method is the strong reliance upon a precisely constructed model and movement controller to achieve later motion which is authentic. Control was analysed for a robotic gymnast with multiple links.

Authors in [46] analysed swing-up in a triple link acrobatics-based robot by applying a sensory-motor scheme, with control processes split into several phases. Another study in 2013 applied a Bang bang method while other cases applied a type of Proportional Derivative (PD) controller, and others combined these approaches. The simulations were run and produced findings in MATLAB, and swing-up to the vertical position was effective. Swing-up was also achieved for a cart-mounted 3-link inverted pendulum: a critical problem in control of this motion is to develop and appropriate feedforward controller [33].

Jaiwat and Ohtsuka's 2014 study [47] concerned non-linear model predictive control-based approaches to achieve swing-up for DLLIP, designing and validating a controller through simulated and experiment-based implementation. The research aimed to control the swing-up and stabilization of a double inverted pendulum using nonlinear model predictive control (NMPC). A fast computation algorithm called C/GMRES is utilized to solve a nonlinear two-point boundary value problem in real time over a receding horizon. The objective is to move the two pendulums from a downward position to an upright position. To simplify the tuning process of the performance index, the terminal cost in the performance index is obtained by solving the algebraic Riccati equation.

A 2017 study reported in [48] considered swing-up in a triple-link robotic manipulation mechanism, focusing on the problem of achieving movement from the system's downwards to its upwards positions. More precisely, the work considered methods for achieving a vertical upwards position for link one, based on the correct angle. This involved synchronising the motor input signals. Implementation of this approach requires consideration of the robot's time responses within the constraints on the actuator energy supply.

More recently, authors in [49] have put forward a novel neuro control approach for swing-up in a 3-DOF manipulator. Implementation of the developed controller utilised a genetic algorithm and neural net. The study discussed swing-up reproduction and large swinging motions in underactuated robotic systems imitating gymnastics movements on a horizontal bar. It focused specifically on equivalent centre of mass (ECM) for gymnasts and underactuated robotic mechanisms.

A 2021 study [6] examined effectiveness for a controller based on optimised fuzzy logic controlling real-time swing-up and stabilisation in a twin-arm inverted pendulum-based robot with rigid couplings. The design of this controller implements Lyapunov criteria for a stable system. Moreover, researchers in [50] put forward a novel type of non-linear hybrid controller (NHC) for swing-up and stabilisation in a robot based on a rotary inverted pendulum with underactuation. Design of the swing-up controller utilised both energy control and feedback linearisation. For stabilisation, a super-twisting sliding mode controller was developed using a novel sliding surface, and stabilising state variables for the active rotary arm as well as the passive pendulum component.

Nishiki's 2022 study [51] presented a novel control law for energy stabilisation in swing-up/giant swing movements for a double link robotic system (the Acrobat), achieving the required values through making regular modifications to link two. In the same year, Pierallini et al, suggested a solution to problems of swing-up, by first deriving a condition which sought to maintain flexibility in the robotic system being controlled, and second, designing a control law utilising iteration and which combined feedback/feedforward terms, and considering stabilisation around the point of vertical equilibrium. The authors then combined these elements to put forward selective gains for systems stabilisation and elasticity, and to confirm a convergence condition for the iteration-based control law.

The section discusses the various approaches for controlling swing-up in pendulum-based systems, which have been extensively studied. These approaches include feedback or feedforward control, control based on predictions from nonlinear modeming, optimal trajectory, and energy shaping. Several studies have been conducted to address the problem of controlling swing-up such as [52]–[54]. The authors used different control approaches, including learning control, PD, fuzzy control, and speed-gradient (SG) technique, to address the challenges of controlling swing-up. The findings from these studies show that the proposed methods are effective when implemented in experiments using different types of pendulum-based systems. Overall, the literature suggests that the development of robust swing-up control strategies remains an active research area.

## 2.5 Complex multi-link mechanism

Mechanisms are systems of interconnected links which allow mechanical movements to be transmitted and transformed [55]. The high-complexity multi-link mechanism contains multiple links, but fewer than the degrees of freedom (DOF) [56]. Such mechanisms can be referred to as underactuated and offer a range of benefits related to the use of special, energetic, and material resources for a range of applications [57]. Within the academic sphere, underactuation in systems is valuable for use in work to evaluate and compare various control approaches [23]. Among the more widely used underactuated systems in this regard are those based on inverted pendulums. Pendulums are defined as bodies in suspension from a fixed location, and which are freely swinging as gravity works upon them and are frequently applied in motion regulation [2]. Despite this, due to the non-linear characteristics of pendulums, they continue to present a useful model, with current uses including presentation of a range of emergent concepts within nonlinear controls, including swing-up and stabilisation of pendulums. In addition, the pendulum is effective for the demonstration of the hybrid system and in controlling the chaotic system [15]. Various research investigates non-linear controls as tested using pendulums with two or three links.  The multiple-link system takes a range of forms, as described in the subsections which follow.

### 2.5.1 Two (double) link pendulum

Two-link pendulums are frequently used in exploring non-linear dynamics, because of the varied, high-complexity dynamic phenomena this type of system produces. It provides a simple dynamic model showing complexity and chaotic aspects in its behaviours, Two-link pendulums comprise 2-point masses, each attached to a light rod, forming two simple pendulums, which when attached create on double pendulum which can oscillate freely within a plane. A double-link manipulator represents a robot with 2 degrees of freedom and is frequently applied due to its functional similarity to the human arm. Therefore, studying the movement of this type of mechanism is useful step towards developing knowledge about complex movements in the human arm  [58]. Figure 2.1 illustrates the double-link Acrobat system, where *l* represents the length, *q* represents the theta angle, and "*x*,*y*" represent the system's axes.

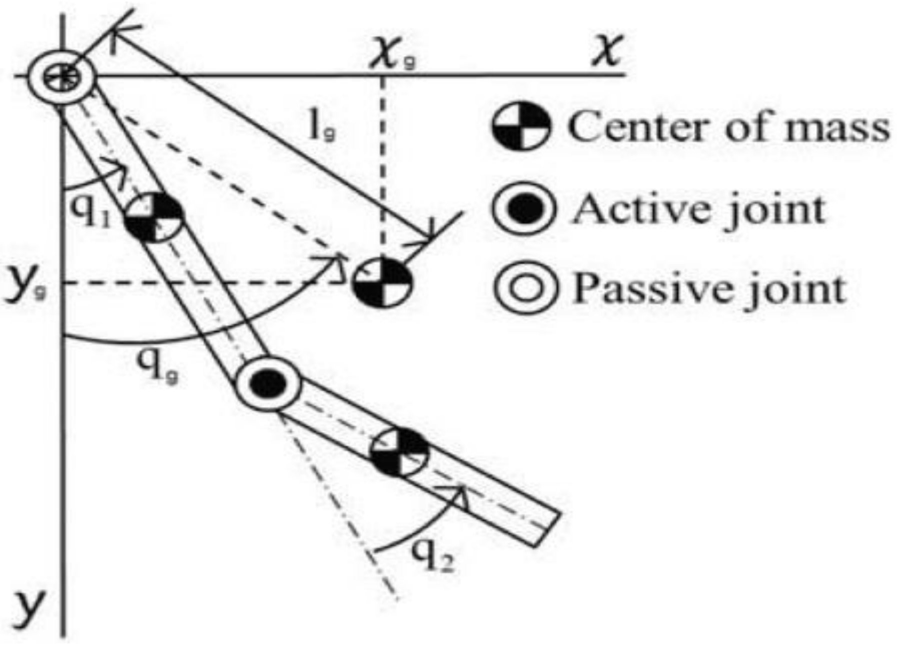


Figure 2.1 Model of two Links Acrobat [49]

Table 2.1 Definitions of parameters of Acrobat

| Parameters | Description |
|---|---|
| $m_i$ | Mass of ith link (kg) |
| $l_i$ | Length of ith link (m) |
| $l_{ci}$ | Length of ith link to the centre of mass (m) |
| $I_i$ | The inertia of the ith around the joint (kg.$m^2$) |

### 2.5.2 Triple link pendulum

Three-link pendulums form an interesting area of robotics due to their comparability to structures in the human body and similarities in how balance is achieved in these systems. The Acrobat robot, which takes its name and concept from acrobatics, provides an example of inverted pendulum systems, and features instability and underactuation, making it suited to use in theoretical and practical research on nonlinear control [18][21]. This system has been utilised with specially developed intelligent controls for balancing, in a study which combined traditional fuzzy and adaptive-fuzzy controls for swing-up and catching tasks as well as for balancing while inverted [9]. Research with the Acrobat has used LQR, state variable feedback and PID [59]. Figure 2.2 shows the triple-link robot, where the variables are defined as follows: *m* represents the mass, *l* denotes the length, *g* represents gravity, and *τ* signifies the torque for each link.

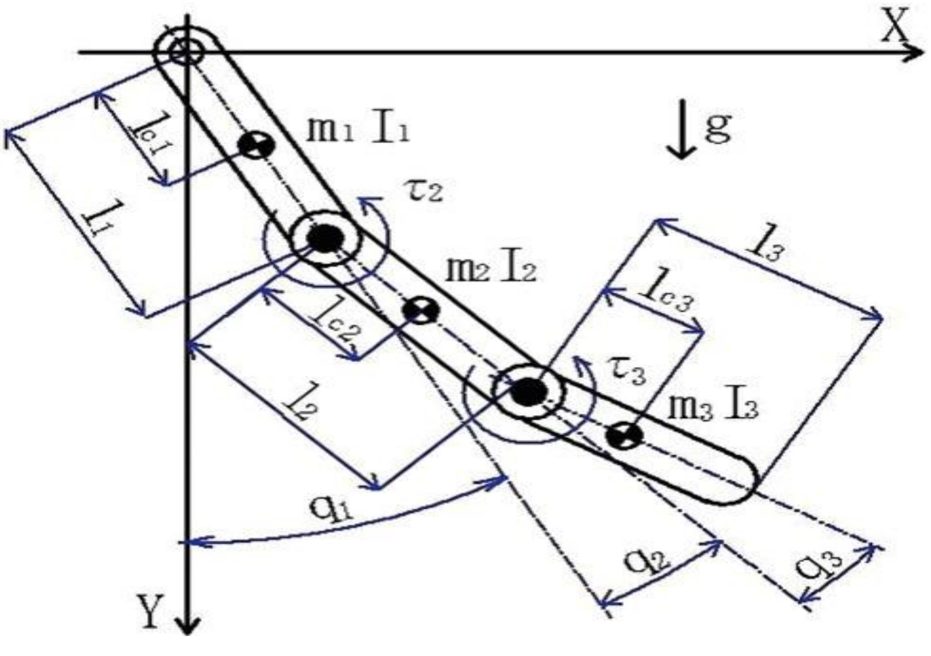


Figure 2.2 Three-Link Underactuated robot [49]

### 2.5.3 Multi-link pendulum

The N-link multiple-link pendulum is a mechanically based system in which multiple pendulums connect together. All the pendulums have individual swing, which in turn impacts upon how the others move. This type of dynamic system is not linear. N-link pendulums are deployed in complex systems behaviour research and can be applied within engineering and physics, including with controllers and in robots.

Figure 2.3 shows a planar multi-link pendulum. The $k^{th}$ link, ..., *N*, comprises a rigidly formed component which has a length of $l_k$, a punctual mass of $m_k$, and concentrates at a point $r_k$ in distance away from the link's base. Measurement of $\theta_k$ as the link angle takes place in relation to link $k_{-1}$'s axis, apart from $\theta_1$, for which measurement is in relation to the set Cartesian frame *xy*'s negative y-axis [60].

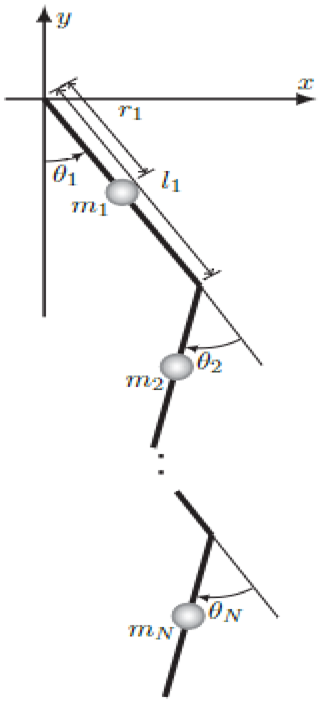


Figure 2.3 The N-link planar pendulum [60]

### 2.5.4 Robogymnast overview

The need for a testbed for research on control in high-complexity underactuated systems in general and inverted pendulums in particular led to the development of a 3-link pendulum-based robotic system, the Robogymnast. This robot is designed to function using the system's inherent dynamics, and emulates the motion used in the gymnastics field when the athlete swings on the high bar. Constructed with a non-powered initial joint, followed by two powered joints, the movement of the Robogymnast is challenging to control due to the passively operated initial joint. Previous relevant studies using the Robogymnast are discussed in this section.

E Eldukhari and Pham (2010)'s study [23] addresses control of swing-up in the Robogymnast. The researchers first applied sinusoidal torques with regular alterations to the motors powering the second and third links. With each increase in the angle of swing, sinusoidal torque amplitude was raised in each motor, and frequencies were lowered in proportion to the increased torque. The findings from this experiment demonstrated that this approach allowed the robot to swing successfully from its vertical downwards stability to upwards inversion.

Kamil et al. 2012 conducted a study investigating swing-up control in the Robogymnast by utilizing the Bees Algorithm. The primary objective was to facilitate a seamless transition of the robot from a stable downward position to an unstable inverted position by optimizing the parameters governing the amplitude and frequency of the sinusoidal signal for each motor. The findings of the study provided compelling evidence of the approach's efficacy, as the optimized parameter values successfully accomplished the desired swing-up motion in the Robogymnast. Consequently, this research significantly contributes to the current understanding of swing-up control optimization through the implementation of the Bees Algorithm, thereby advancing its application in the field of robotic systems [61].

Another study investigated by Kamil et al. in 2014 [62], the primary objective was to achieve stability and balance in the Robogymnast while in an upward inversion position. The researchers utilized optimal control theory and implemented a Discrete-Time Linear Quadratic Regulator (DLQR) as the controller. The study effectively demonstrated that the application of the DLQR approach, combined with the selection of an appropriate weighting matrix, successfully stabilized, and balanced the Robogymnast. Consequently, this research significantly contributes to the current understanding of control mechanisms in robotics, specifically in the context of achieving stability and balance in challenging positions.

Ismail et al. produced a study in the same year [16] which utilised Invasive Weed Optimisation (IWO) for an exploration of optimised control parameter values applied to

Robogymnast swing-up. One motor installed at the second joint, and one at the third are responsible for controlling the robot's motion. The first joint attaches rigidly to the high bar, which is ball-bearing mounted and capable of free rotation. The IWO algorithm optimises the swing-up process by identifying optimal parameter values for the management of sinusoidal voltage inputs for each motor. These values can be applied in both simulated and experimental work. The findings showed that this approach successfully swung the robot from a point of downwards stability to inversion.

In a study conducted by Ismail et al. in 2015 [63], the primary objective was to optimize the Q values in a Linear Quadratic Regulator (LQR) using Invasive Weed Optimization (IWO) to achieve balance in the inverted position of the Robogymnast. The study successfully demonstrated the effectiveness of IWO in attaining sustained stability of the Robogymnast in the upward position. This research significantly contributes to the advancement of control approaches in robotics, specifically in the realm of optimizing the LQR controller for inverted balancing of the Robogymnast.

Ismail et al.'s 2017 study, described in [64], The study involved the integration of two optimization approaches, namely the weighted criteria method IWO (WCMIWO) and the fuzzy logic IWO hybrid (FLIWOH) methods. These techniques were applied in determining optimal values in the LQR controller's Q matrix in balancing the Robogymnast when in an upward vertical position. The study firstly involved developing two types of LQR controller based on parameter values optimised by the respective optimisation techniques. This was done a second time with the robot subject to state disturbance, from which further LQR controller sets were developed. Testing of controller responses across various cases was simulated, with performance evaluation carried out. The findings demonstrate the capacity of each of the four controller types to achieve balance in the robotic system to differing extents. Moreover, training each controller based on disturbances led to reductions in settling time.

In summary, this section highlights the development of the Robogymnast, a robotic system specifically designed to study control in high-complexity underactuated systems, with a particular focus on inverted pendulums. Multiple studies have been conducted to investigate swing-up control, stability, and balance in the Robogymnast. These studies utilize various optimization techniques, including sinusoidal torques, the Bees Algorithm, optimal control theory, Invasive Weed Optimization (IWO), and Linear Quadratic Regulator (LQR) optimization. The findings from these studies significantly contribute to the understanding of control mechanisms in robotics and demonstrate the effectiveness of the proposed approaches in achieving desired motions and stability in the Robogymnast.

## 2.6 Control systems

Controllers for robots have been extensively studied in previous research: however, this field continues to pose significant challenge, as robotic systems behave with strong nonlinearity, have complex dynamic features, are underactuated, have redundant actuation, are kinematically limited, have singularities, and have limitations in real time, among other difficulties. The controller approaches put forward are demonstrated using various types of experimental platform, across four areas within robotics, namely: underwater, parallel, humanoid and underactuation robotics [65]. Mechanical linkages consist of components which are inter-connected with the aim of managing motion and forces. Geometry is used to study the motion of bodies/links, and links are assumed as rigid [66]. Modelling of the interconnections or joints which connect each link together assumes that ideal movement is attained (e.g., pure slides and rotational movements). This model links which show rigidity connected through ideal joints are termed kinematic chains [66]. Control schemes for swing-based robots with multiple links are applied to controlling motion and positioning in multi-joint/link robotic arms, as frequently used in industry on assembly lines and to weld or paint products. As multiple-link systems are complex, various control schemes are applicable.

Position control is a frequently used approach to controlling robotic systems and works by first identifying the required positions of the respective joints, before calculating the motions required to achieve those positions. Alternatively, velocity control can be used, involving setting the joint's required pace and trajectory as opposed to specifying its position. Controls may also be based on force control, where the force that the robot arm applies to its surroundings or an object for manipulation is monitored and controlled. Among various further control scheme types are: hybrid controllers, combining aspects of multiple approaches for a specified objective; and trajectory control, in which the route that the robot arm must take is specified.

The section discusses the various approaches for controlling swing-up in pendulum-based systems, which have been extensively studied. These approaches include feedback or feedforward control, control based on predictions from nonlinear modeming, optimal trajectory, and energy shaping. Several studies have been conducted to address the problem of controlling swing-up. The authors used different control approaches, including learning control, PD, fuzzy control, and speed-gradient (SG) technique, to address the challenges of controlling swing-up.

In summary, controllers are an essential consideration for multiple-link pendulum-based robots in order for them to move with accuracy and efficiency, with the control method selected depending upon what objectives are set and what the application requires. The findings from these studies show that the proposed methods are effective when implemented in experiments using different types of pendulum-based systems. Overall, the literature suggests that the development of robust swing-up control strategies remains an active research area.

### 2.6.1 Classical control

Classical/traditional control approaches for dynamic systems utilise mathematics-based modelling and analysis and encompass a wide range of techniques which have emerged across a span of many years. These approaches have wide applications within different engineering disciplines, engineering such as in aerospace, electrical and mechanical fields. Classical controls principally aim to develop a controller that is capable of stabilising the given system, regulating outputs, and tracking the aimed-for setpoints. These goals are reached through the application of mathematical systems modelling, e.g. by applying differential equations, in developing a controller capable of manipulating system input factors in order to create the needed outputs. The sub-sections which follow describe the two main classical control methods [67].

#### 2.6.1.1 PID

Employed-feedback Proportional Integral Derivative (PID) controllers have widespread popularity in managing industrial controls, as well as in other settings which require continuously modulated control. A PID controller functions by analysing and measuring error using differentials between the target Set Point (SP) and the measured Process Variable (PV), adjusting control in real time, in response to proportional, integral, and derivative parameters. Implementation of this approach leads to automated, accurate and response-based alterations to controls. This PID algorithm enhances system capabilities by bringing back measured outputs to desired inputs while minimising deferral error. The distinctive feature of the developed PID controller is its implementation of three control types, proportional, integral, and derivative, which each influence control output, to optimise control and make the controller more efficient. Control outputs are determined through proportional, integral, and derivative terms, and their measurement is also based on these terms [68]. With u(t) representing output, this PID controller can be written as:

$$u(t) = k_p e(t) + k_i \int_0^t e(\tau)d\tau + k_d \frac{de(t)}{dt} \qquad (2.1)$$

Where:

$k_p$ represents proportional gain, $k_i$ is Integral gain, and $k_d$ denotes Derivative gain.

$e(t)$ represents the error signal at time t, and $e(\tau)$ is error signal at $(\tau)$.

A simulation of movements analogous to human gymnastic movements was run, and the results of this compared to outcomes using another control type. The PID controller can be represented through the equation:

$$c = k\,(1+ \frac{1}{T_i s} + T_d s) \qquad (2.2)$$

Where:

*K* is overall gains, $T_i$ and $T_d$ are integral time and derivative time respectively. PID controllers require low-pass filters, as in a controller which solely used PID, high frequency gains would be infinite, which would not only be detrimental, but impossible [69].

### 2.6.1.2 LQR

LQR offers a dynamic approach to developing controllers for application in a complex system [70]. Moreover, state feedback control (SFC) exploits the position of each pole within a system, placing these and the state variables in gain matrix K. Using SFC, poles in closed-loop systems can be positioned wherever required, whereas output feedback controls allow the poles to be placed in fixed locations [71]. This approach applies the SFC controller for implementation of the state variables, which are used for feedback. Once each feedback component has been multiplied in the state feedback gain matrix, they can be compared with reference input values. State feedback control is generally used in gain matrix calculations, with widespread use if the LQR controller to achieve this. This type of controller would ideally be used with cost function (J) for state optimisation, x(t), and u(t) as system control signal among the parameters in the K matrix [72]. Other controllers applied for robot manipulators include PD [73] and LQG [74].

### 2.6.2 Intelligent control

This subsection discusses intelligent control, in which artificial intelligence approaches are used in controller design for application with a dynamic system. The rationale for this is to enhance the way the control system performs and render it more adaptable through enabling the system to learn, use knowledge and make decisions. Intelligent control aims to produce a controller which is able to respond to changing systemic or environmental conditions, to decide actions in relation to circumstances at that moment. The approach can utilise fuzzy logic, evolution-based algorithms, or neural network concepts, designing controllers capable of learning from the way the system behaves and using this information to inform decision-making. Intelligent controls are a recently emerging and quickly evolving area which brings together artificial intelligence and theories of control to produce high-performing, flexible controls. Intelligent control has widespread uses in robotics, aerospace engineering, and controlling processes in industry. Various intelligent control approaches will be discussed in the following sub-sections.

#### 2.6.2.1 Fuzzy logic

Fuzzy logic offers a stable and robust approach, and this has led to its broad use for research, in which it outperforms conventional control approaches. Fuzzy systems were first put forward by Zadeh in 1965 [75]. Fuzzy logic controllers are in the category of intelligent control, and rely upon fuzzy rules, which are systematically and carefully set [74]. The method begins from observation of the target system, before articulating an analogous system using fuzzy rules based on IF–THEN constructions. The fuzzy logic controller (FLC) is bounded by one/ zero and extent to which a component is describes by zero or one. The FLC brings systems nearer to the way human thought is expressed [76].

FLCs have three primary phases, which are: *fuzzification*, the fuzzy inference engine (logic of decisions); and *defuzzification*. These phases are illustrated via a block diagram in Figure 2.4.

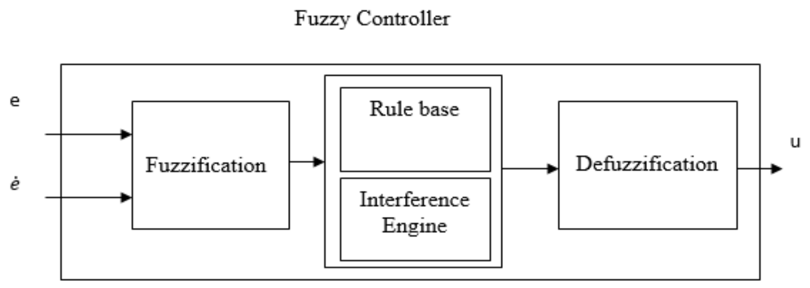


Figure 2.4 Fuzzy Logic controller architecture [77]

From the Figure, the initial stage of *fuzzification* involves the conversion of the input data components into degrees of membership through lookups within a single or multiple membership functions. Both rules and inference bases are able to simulate decisions taken by people, using fuzzy concepts as well as the capacity to infer fuzzy control acts using fuzzy implications and fuzzy-logic-based rules on inferences. Fuzzy set membership functions and rules of fuzzy control impact upon how controllers perform significantly. In the final phase, *defuzzification,* the fuzzy set developed undergoes defuzzification to form a precise control signal [77], in which *e* represents error in the input variable, *e** represents error derivation, and *U* represents the FLC's output variable. Increasing the rule numbers will increase accuracy but means that it will take longer to process the data.

### 2.6.2.2 Artificial Neural Networks (ANN)

Artificial neural networks (ANNs) form a computation-based model for the brain (Pham and Liu 1995) [2]. The ANN is made up from connected artificial neuron groups and performs information processing via the connectionist computational method. Generally, artificial neural networks are adaptive systems, and can modify themselves structurally in response to information from within or outside the system which moves through this network as part of the learning stage. The modern artificial neural network is applied to model non-linear statistical data, being designed as a generalisation of mathematical modelling for humans' cognitive processes and biological neural networks [78]. Frequently, artificial neural networks are categorised into single or multiple layer networks. When identifying layer numbers for a network, in general, input units are not considered to form a separate layer, as they do not have computational functions. The network's layers can be set based on counting the layers of weighted interconnecting links which connect neuron tranches, based on the significant information held by these weights.

### 2.6.3 Robust control

Robust control systems are discussed here as an approach to control in a dynamic system in order to ensure stability and adequate performance despite uncertainties and disturbance. This control method has frequent applications within systems being run in an evolving or unpredictable context: e.g., processes and systems in industry, aerospace, and transport. Robust control primarily aims to develop controllers able to manage changing system parameters and disturbance, including from sensor noise and disturbance from sources outside the system, and to accept uncertainty, while still performing effectively and maintaining stability. This works through methods including robust optimisation and robust stability analysis in controller design.

#### 2.6.3.1 Sliding mode control

The non-linear Sliding Mode Controller (SMC) has attracted significant research interest, and this has recently intensified. Sliding mode control was initially developed by Emelyanov and colleagues in the mid-twentieth century, and significant developments have occurred in this area since high-speed controls were invented [79] [80]. This controller has applications in many fields, when good performance is needed in addition achieving robust and stable control [81] [82]. Despite this widespread usage, SMCs have certain limitations. The first is the issue of chattering, which may lead to high-frequency oscillations in controller outputs. Second, SMCs have high noise sensitivity at an input signal of near zero. Third is the issue of non-linear equivalent dynamic formulation, as significant for an effectively performing control: but calculation of this variable is challenging as it relies upon a non-linear dynamic equation [83][84].

#### 2.6.3.2 H∞

*H*∞ controller offer robust control and are a source of widespread interest. The controller is highly effective in managing various problems of theory and practice and offers a range of benefits in comparison with classic control approaches. These include the strong provision of disturbance rejection, management of uncertainty, and strong capacity to maintain stability across various conditions of operation [85]. A *H*∞ controller developed using a state-dependent Riccati equation approach aimed to track position in a robust manner and suppress vibrations for a single FLM. The controller was shown to be robust in disturbance rejection at the cost of a small reduction in systems performance [86]. When loop-shaping-designed *H*∞ and linear quadratic Gaussian (LQG) controllers were compared in terms of position tracking for a single FLM it revealed a higher performance for the former controller [87]. Conversely, the LQG controller gave better performance compared with a *H*∞ controller based on linear matrix inequality (LMI) in control of a single FLM [88].

*H*∞ controllers were developed variously applying a mixed-*H* technique and loop shaping, for accurate tracking of the end effector in a single FLM. The findings show higher performance for the controller based on loop-shaping. Notwithstanding, designing an *H*∞ controller through this approach is highly costly in terms of time and effort [89]. This is due to the lack of a systematic formula for selection of appropriate weighting functions, which means that these must instead be trialled in experiment until target performance is reached [90]. *H*∞ control using T-S fuzzy modelling was put forward to reduce modelling error effects caused by stiffness variation in the flexible joint robot [91]. A controller integrating *H*∞ and conventional PID controls was put forward aimed at robustness in the presence of uncertainties in models. This controller was robust for the target position for the tip of a single FLM [74]. A controller using $\mu$-synthesis was put forward for robust modification of input trajectory, decreasing tip error for a single FLM [92].

### 2.6.4 Hybrid control

This section discusses hybrid controls as a technique for control of a dynamic system. This approach integrates aspects of various control methods, including classic, intelligent, and robust controls. Hybrid control principally aims to exploit the advantages from various techniques to perform certain tasks or manage various uncertainties.

The FPD controller integrates fuzzy logic and PD control. It is chosen in this work for stabilisation of the non-linearly unstable Robogymnast. FDP was selected for stabilisation to counter this issue [93]. The PD controller type uses feedback control, using control variables to form their outputs, generally derived from error ($e$) in comparison with a reference value specified by the user and a process variable measure. The error is used by the different parts of the controller to choose their action. First, in proportional control, error is multiplied by adjustable amplifier gain $K_p$. In a range of systems, stable processes are based upon $K_p$. and very low values of it cause PV to drift, with very high Kp values leading to oscillation. Second, derivative control involves multiplying error change rate by gain $K_d$. Across various different systems, $K_d$ controls system response, with PV oscillation occurring at overly high levels of

$K_d$ and overly low values causing slower PV response. Based on this, controller design must consider the potential amplification of noise in error signal based on derivative action. It is important to tune the PD controller through modifying $K_p$ and $K_d$ to optimise system responses [94].

Linear quadratic regulator - proportional-integral-derivative (LQR-PID) controllers combine PID and LQR controls and was applied in 2022 in order to stabilise the Acrobat, with results used for comparison with various controllers, in which each controller had distinctive advantages. The structures of the PID and LQR controller differ, meaning that they perform in a different manner in both tuning and modelling closed-loop systems. LQR provides lower rising and settling times compared with PID. Conversely, PID decreases error/overshoot, and implementation of this type of control is less complex. Each controller type produces different results because of their distinct approaches to feedback gain matrix calculations [95][16].

The Fuzzy-LQR controller offers control optimisation through LQR with fuzzy logic-based control (FLC) [96]. FLC is rules-based and utilises a Fuzzy Control Rule (FCR) set, with rules interrelated by fuzzy implication and the compositional rule of inference [97]. FLC design processes are described here. Fuzzy logic is used to create a supplementary controller to assist the primary controller when conditions change. It is widely applied across manufacturing processes, and in enhancing intelligence in technologies [98]. Mamdani's strategy for fuzzy model development is applied here to allow alterations to closed-loop controller feedback gains.

In summary, hybrid controllers are a recent innovation which is undergoing rapid development, based on combining advantages from various controls for the management of disturbance and uncertainty in various contexts and conditions, Hybrid control is applied within aerospace industries, robotics and for controlling processes in industry.

## 2.7 Optimisation techniques

Existing research includes numerous works which describe how evolutionary optimisation algorithms contribute to the development of control systems. Moreover, optimisation methods have broad applications across various fields of study, from mathematics and physics to business and decision-making. This section will discuss several algorithms which are used in the literature.

The Bees algorithm (BA) was put forward by Pham et al. (2005) in [99], Bees Algorithm: A Novel Approach to Function Optimisation, which makes a significant contribution to metaheuristic algorithm research. As discussed earlier, this algorithm was developed as inspired by the way in which bees forage in a natural environment. The algorithm starts in initialisation, in which a small number of scout bees (n) are placed at random within the search zone. After that comes the neighbourhood/local search phase, which involves the recruitment of bees to search in a localised manner in the optimal locations ($m$) in a specified range ($n_{gh}$) based on the rankings developed through the fitness function for the starting sample. The selection of elite bees (e) occurs from among the fittest m from the earlier selection. Recruitment of further bees occurs, and these search in the vicinity of the elite bees (e), with smaller numbers of bees assigned to searches in non-elite (m−e) locations. The rest of the scout bee (n−m) group are recruited for random searching within the search zone, as part of global searching [100].

Invasive weed optimisation was initially proposed by Mehrabian and Lucas, as a new numerical stochastic optimisation algorithm which takes its concept from the way in which weedy species invade and colonise an area [2]. Weeds are robust and seed prolifically, and these features are integrated into this simple, adaptable, and successful swarm-based optimisation approach. IWO differs from classic forms of numerical search algorithm in some of its features, such as spatial dispersal, competitive exclusion, and reproduction [101]. The study in [102] implements Modified Chaotic Invasive Weed Optimization (MCIWO) as an algorithm for gain optimisation of a torque based PID controller controlling motors in a bipedal robotic system to walk across an even surface. In order to design this controller, derivation of the robotic system's dynamics was performed following the Lagrange-Euler formulation. Following this, the PID controller applied in the algorithm was manually tuned to identify gain ranges. Further, it was applied in [16] in optimising control parameter values for swing-up in a three-link inverted pendulum robotic system, the Robogymnast, designed for research use into issues linked to this type of system.

The metaheuristic algorithm is an advanced optimization technique created to discover satisfactory or nearly optimal solutions for intricate optimization problems [103], an algorithm called ant colony optimisation (ACO) is applied for complex combinatorial optimisation [104]. ACO's foundational algorithm is derived from the way in which ant species leave pheromone trails in order to communicate. Analogous to this, ACO uses a simple 'ant' colony consisting of agents in indirect communication via 'pheromones' [105]. Another study combined visually based servo control alongside model-based image segmentation and ACO, developing an effective six-degrees-of-freedom (6-DOF) robotic manipulator in order to solve complex combined pick-and-place activities. The visual segmentation approach offered simplicity and efficiency in extracting information about objects through obtaining relevant features of the platform under control as the robotic system tracks patterns of manipulated images. The evolving ACO learning algorithm investigates close-to-optimal choices of paths, driving the 6-DOF robotic arm kinematic modelling to achieve picking and placing exercises in the fastest time [106].

Rashedi et al. (2009)'s gravitational search algorithm (GSA) is applied to the solution of optimisation problems. This heuristic algorithm draws on populations and uses laws regarding the interaction of gravity and mass. GSA involves searcher agent groups interacting via gravitational forces [107]. Each agent is viewed as an object with their mass indicating how well they have performed. Forces of gravity lead to global movements in which each object is drawn to objects of greater mass, which move more slowly [108]. A novel method of optimisation was developed for pathway trajectory in multi-robots with an improved GSA (IGSA) [109] within dynamic conditions. IGSA's enhanced features are based on memory, information, social and cognitive factors from particle swarm optimisation (PSO). Following this, the second generation's population is selected based on a greed-based approach. A system to plan pathways was designed via IGSA for optimal attainment of successive robot positions from current positions. Moreover, a comparison was made between findings from analysis and experiment with multi-robot pathway planning and findings from IGSA, GSA and PSO using comparable environments.

Multi-objective optimisation (MOO) models the preferences of decision-makers, and the MOO approach differs based on the articulation of such preferences by the decision-maker, falling into three main types [110]. The optimal value is identified via optimisation, and optimisations problems can be seeking a minimal/maximal value and based on single or multiple objectives. Any problem with multiple objectives falls under MOO. Such problems frequently arise in a range of fields, including engineering, mathematics, economics, sociology, agriculture, aerospace and car industries [111]. Authors in [112] put forward an approach to solving multi-robot dynamic task allocation problems. MOO was used for estimation and assignation offers, based on the aim of developing a solution with broad, domain-independent applications which can enhance efficient use of energy or time, for example. This algorithm offers a highly flexible approach suitable for implementation across a range of settings if the parameters modelled are in line with those developed. In addition to considering the distance involved, this addresses how efficient robots are at given tasks.

## 2.8 Summary

The second chapter of this thesis provides a comprehensive background review of various designs for underactuated multiple link systems. It discusses the existing literature on controlling swing-up and balance in these systems and reviews different control approaches that have been previously applied in controllers for a wide range of high-complexity robots, with a particular emphasis on studies related to controlling swing-up. Furthermore, the chapter also delves into published research on the application of optimization strategies to enhance controller performance.

This review of previous work has identified gaps in the literature regarding soft computational optimization methods, highlighting the need for further examination and development. Notably, no prior studies have been found that applied the Teaching Learning Based Optimization (TLBO) algorithm to develop a control system for the three-link Robogymnast. However, evidence showcasing the high performance and effectiveness of TLBO in optimization problems across various domains suggests its tremendous power and potential. Consequently, it is an ideal candidate for testing parameter optimization in hybrid FLQR control. The upcoming third chapter will provide a description of the Robogymnast, derive its mathematical model for the system.

# Chapter 3:

# Mathematical model, system description and design

## 3.1 Introduction

This chapter explains the system's features, design, and mathematical model. It also describes how the multi-link robotic system was linearized. The Robogymnast utilizes a complex, highly coupled mechanical framework with multiple joints. The multifaceted nature of this framework is not unusual for a mechanical structure of this complexity and a control system of this difficulty. The lack of full actuation brings challenges when setting up feasible trajectories and planning controllers. The Robogymnast design includes a total of three degrees of freedom, two of which are powered.

The main fundamental problems with Robogymnast are:

(i) the ability to move from one place to another

(ii) determining the complex mathematical equations of motion

(iii) the analysis and control of multiple kinds of movement [4].

The Lagrangian motion methodology has been used to determine the system dynamics of a serial-chain mechanical manipulator with a rigid link [10], [113]. The dynamic system model and equations of motion of the multilink system are determined mathematically using Lagrange‘s equation method to depict the system dynamics. This approach has been used in many previous studies [20], [27], [114].

The Robogymnast is a planar three-link system in the vertical plane, with two actuators (Stepper Motors) at the second joint (shoulder joint) and the third joint (hip joint), but no actuator at the first joint (hand joint). Lagrange's mathematical statements are used to determine the mathematical representation for the Robogymnast, in the same manner as used by [23].

The Robogymnast configuration was designed to be adaptable for future adaptations or modifications. These may include: changing the actuators for more powerful ones; increasing the degrees freedom by adding extra link(s); changing the length of the shaft or the free rotating bar; adding sensors to measure the angular velocity (e.g. tachometer) of the second and third links; attaching the angular position sensors (potentiometers) with more accurate ones;

disposing of the impact of the backlash in some if not all joints (e.g. hip joint) by utilising anti-backlash.

Section 3.2 presents a description of the system, including sketches and schematic diagrams of the Robogymnast. In Section 3.3, the derivation of the mathematical model of the system is presented and discussed, also, the step-by-step derivation from the Euler-Lagrange equation to the state space model of the system is demonstrated in this section. Lastly, the linearization of the model is determined.

## 3.2 System description

Robotic manipulators are nonlinear and highly complex systems in nature. These systems find wide applications in the industry; therefore, effective control of robotic manipulators becomes extremely important. These robotic manipulators have links, joints, and actuated end-effectors [115].

The basis of the design for the three-link robotic system used in this study comes from an established movement in gymnastics in which an individual hanging from a high bar uses free rotation to swing up and over the top. Figure 3.1 provides a block diagram representing the system. The first link is analogous to the arms of the gymnast, with no wrist or elbow jointing; Link 2 combines the gymnast’s trunk, neck, and head into one unified part, and Link 3 connects the legs and feet, with no jointing for the knees and ankles. The non actuated passive joint (Joint 1) in the system is analogous to the athlete’s hands, and the active second and third joints relate to the shoulder and then the hip joints in a human. The first link is consisted of 2 arms of 13mm from the shaft to the 1st joint, which include 90 mm diameter and 7 mm in thickness of SLS material connect with an aluminium part holder that hold the first stepper motor, then the second link is represent the body of the robot in length of 90 mm SLS tube as well connected by an aluminium holder contain the second stepper motor, lastly the last link which contains the low part of the system simulate the legs with 2 tubs of 20 cm SLS material tube which are cheap, rigid.

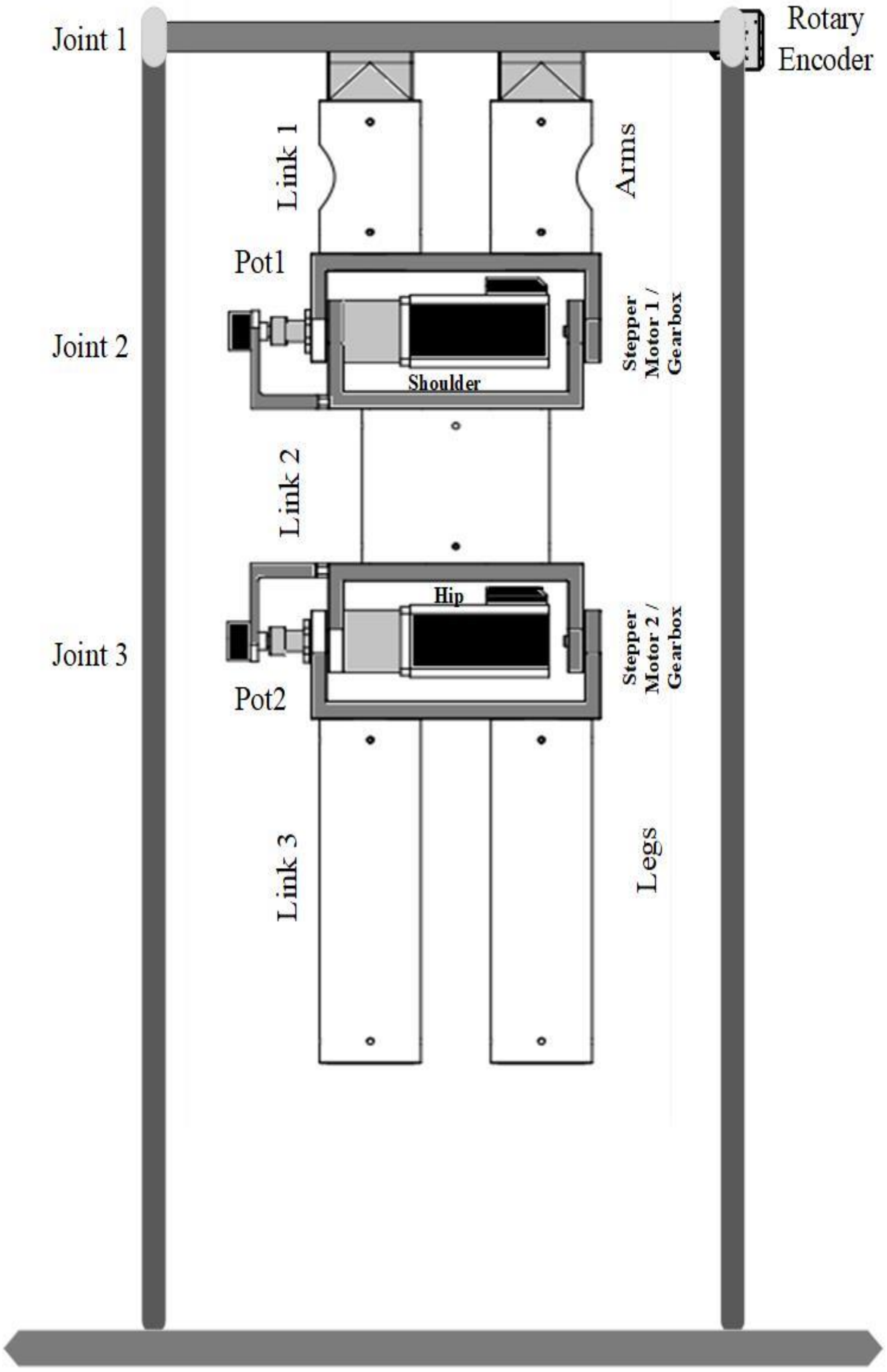


Figure 3.1 Robogymnast Diagram

The multi-link rotation robot (Robogymnast) is designed and though its similar to the human movement, the main idea is to rotate on freely high bar by reserve motion between shoulders and hip, as such acrobat motion.

The Robogymnast system is operated using two stepper motors connected to a stepper driver, which allows the system to move smoothly. In addition, the STM32F Microcontroller (ST, Italy) is used for programming of the control system, using the language C++ for converting commands passed from the PC-based control system and the robotic system. A sensor is also assigned to each of the links, with Link 1 being connected to a Single-turn rotary encoder, while higher precision potentiometers 2 and 3 were connected to Links 2 and 3, respectively. The sensors monitored the absolute angles at each position, Figure 3.2 shows the block diagram of Robogymnast operation system.

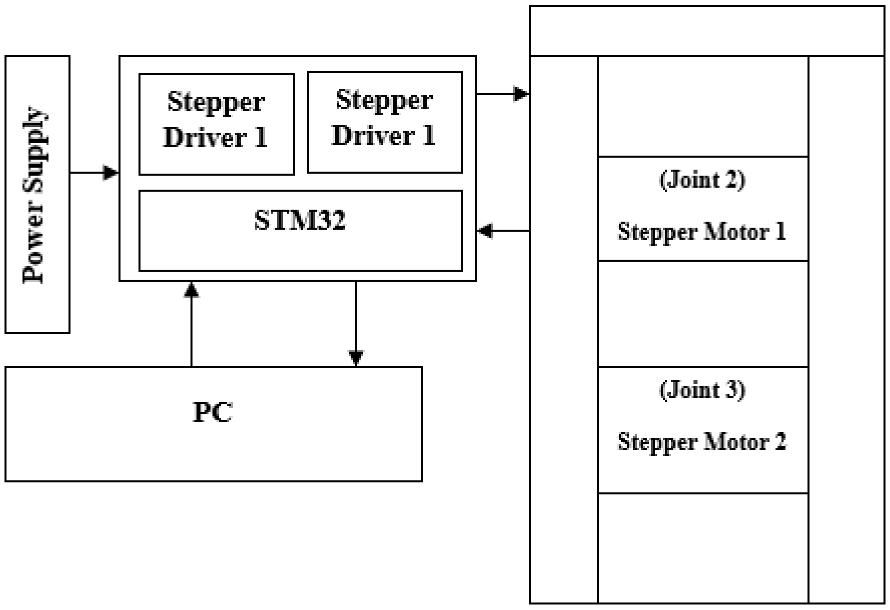
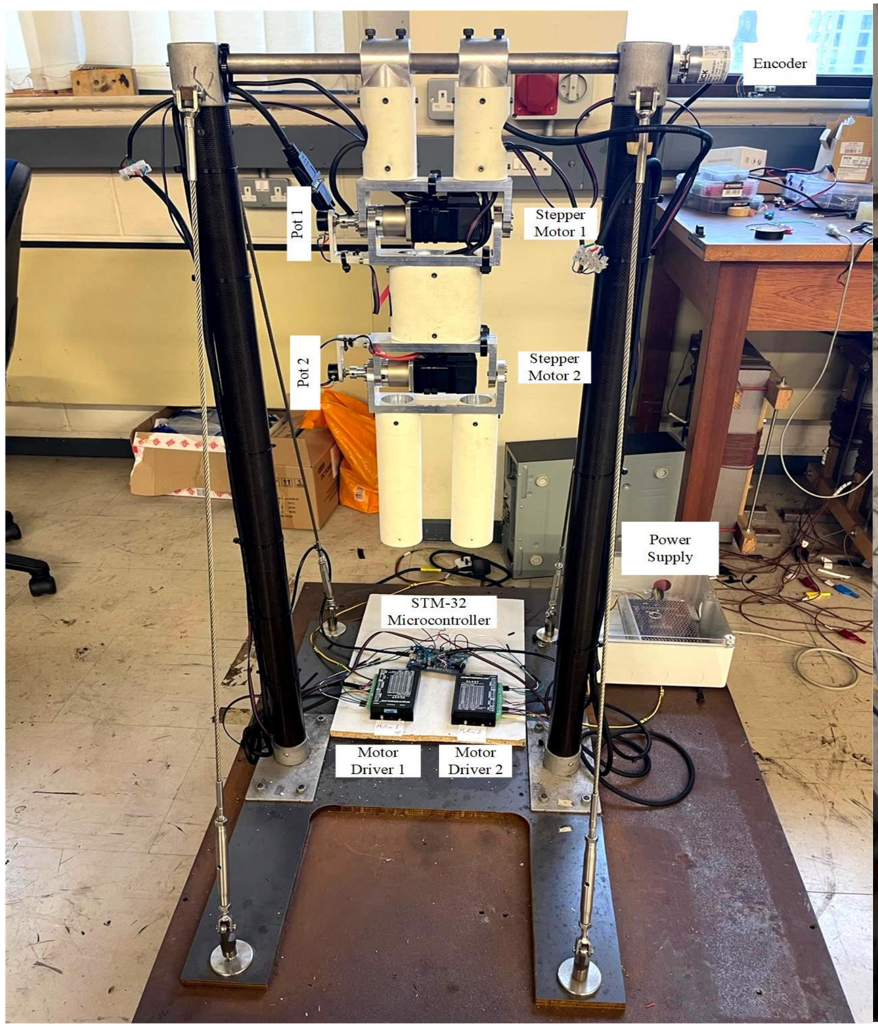


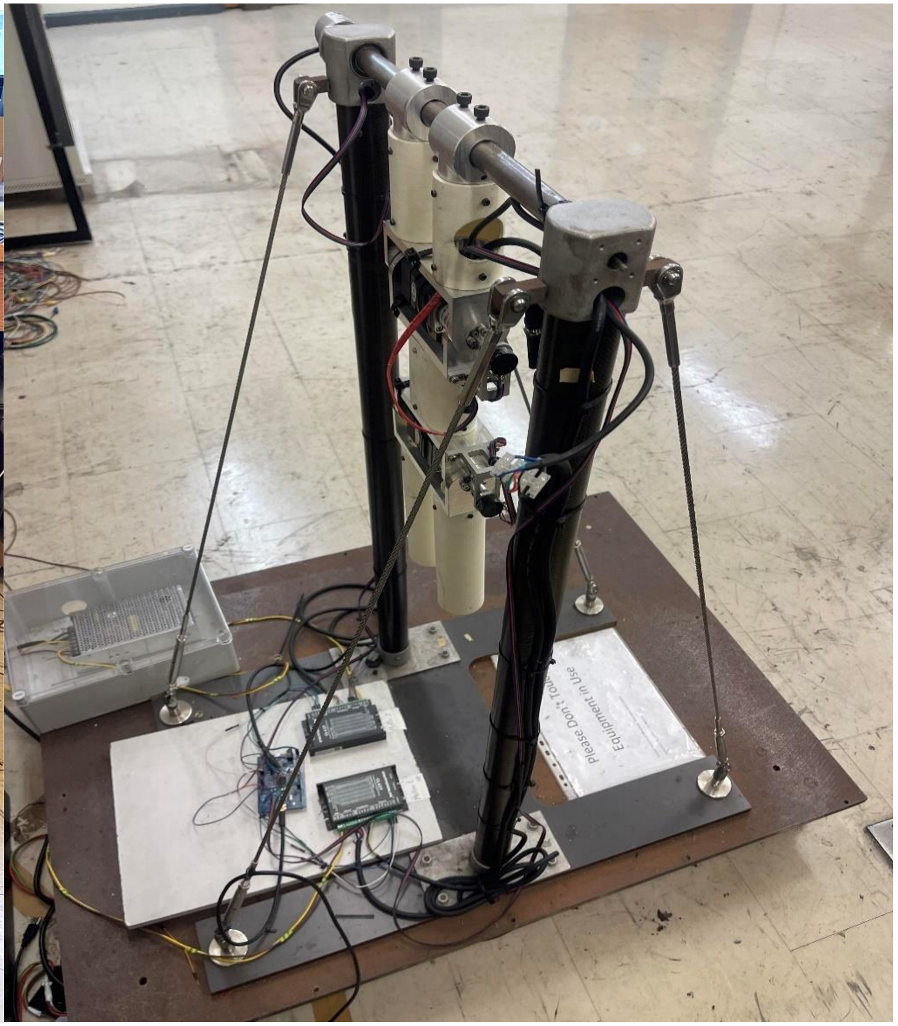
Figure 3.2 The operation system

The Robogymnast is controlled initially by a PC equipped with appropriate STM32 [116] as microcontroller. C++ programmes are used to transmit the input/output commands between the PC and Robogymnast test. C++ program environment is used to transmit the input/output commands between the Personal Computer (PC) and the Robogymnast powered by 5V DC. The program consists of a code to record experimental data and store the information on a hard disk. The computer controller programs contain: a state feedback controller, discrete integrator, and motion code.

The structure of Robogymnast is modelled on a human gymnast swinging on a freely rotating high bar with his/her hands firmly fixed to the bar. As mentioned in part 3.2 the main part of each link is manufactured from two rigid carbon SLS tubes, which are 90 mm in diameter. These are economical, simple to cut, and weigh just 0.085 kg/m. At both ends of each link, aluminium segments (7 mm in thickness) are appended to give structure on which to mount the sensors and actuators Joint 1 consists of a steel shaft mounted on ball bearings, at one end of the shaft a Rotary encoder is mounted to measure the angle of the first link. A description of the encoder is gives in 3.4.1.2 system design section. Joints 2 and 3 consist of two parts. The first part comprises a stepper motor/gearbox combination with its output shaft coupled to the respective link. Planetary gearboxes with a rated continuous torque capacity maximum of 3 Nm. The second part includes the potentiometer, which measures the relative angles between adjacent links. The potentiometer is attached to a short steel shaft mounted on both sides on ball bearings. One of the most important points in robot design is the selection of the actuators. Good selections give a robot more capability to achieve different types of movement. Conversely, the angular velocity and torque of the actuator are an important characteristic in the performance of the robot, and they play a key role in the selection of actuators to maintain the required movement. In this study, stepper actuators were selected to control the joints of the robot on the grounds that they are generally less expensive and lighter in weight than other available options. The movements of the Robogymnast are modelled mathematically as described in the following section.

Also, the output findings, and measurements are also collected by rotary encoder in first link and 2 potentiometers for links 2 and 3 are displayed on PC. Figure 3.3 (a,b) illustrates the front and side view of the actual environment.





(a) (b)

Figure 3.3 Robogymnast (a) Front view (b) Side view

## 3.3 Mathematical model

In this section, the mathematical model of the Robogymnast is described. Furthermore, for modelling purposes, the Robogymnast is regarded as a triple link pendulum in a stable equilibrium configuration [23]. In Figure 3.4 schematic representation of the Robogymnast is displayed, the system uses the Euler-Lagrange formula. This method involves only the derivatives of time, speed, and position. The most important part of the Lagrangian equation is obtaining the kinetic and potential energies of the entire system [117]. Also Figure 3.5 presents Multilink angel axis diagram.

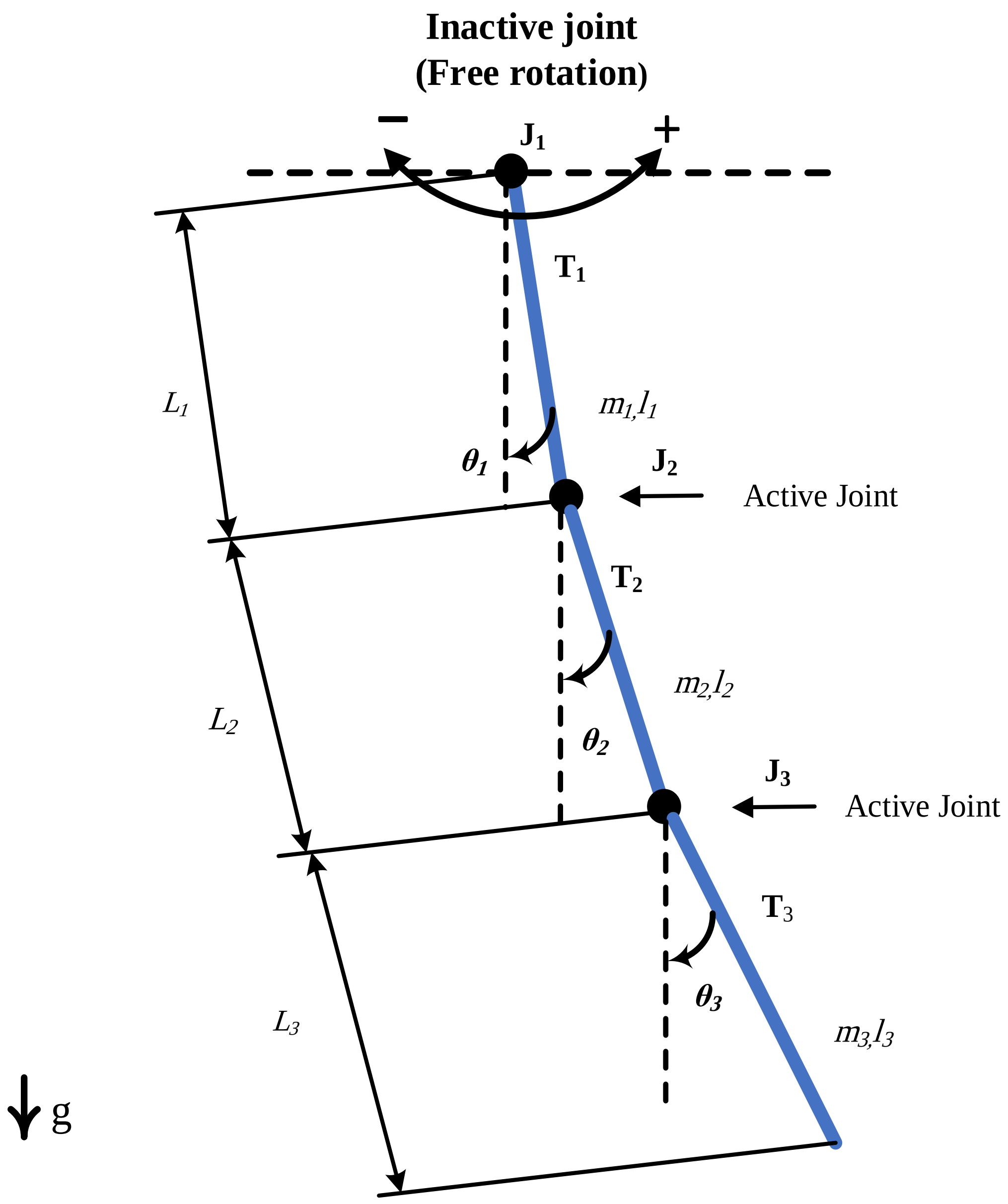


Figure 3.4 Schematic representation of Robogymnast in the downward position

Where:

| | | |
|---|---|---|
| $L_1$ = length of link 1 | $m_1$ = mass of link 1 | $j_1$ = joint 1 |
| $L_2$ = length of link 2 | $m_2$ = mass of link 2 | $j_2$ = joint 2 |
| $L_3$ = length of link 3 | $m_3$ = mass of link 3 | $j_3$ = joint 3 |

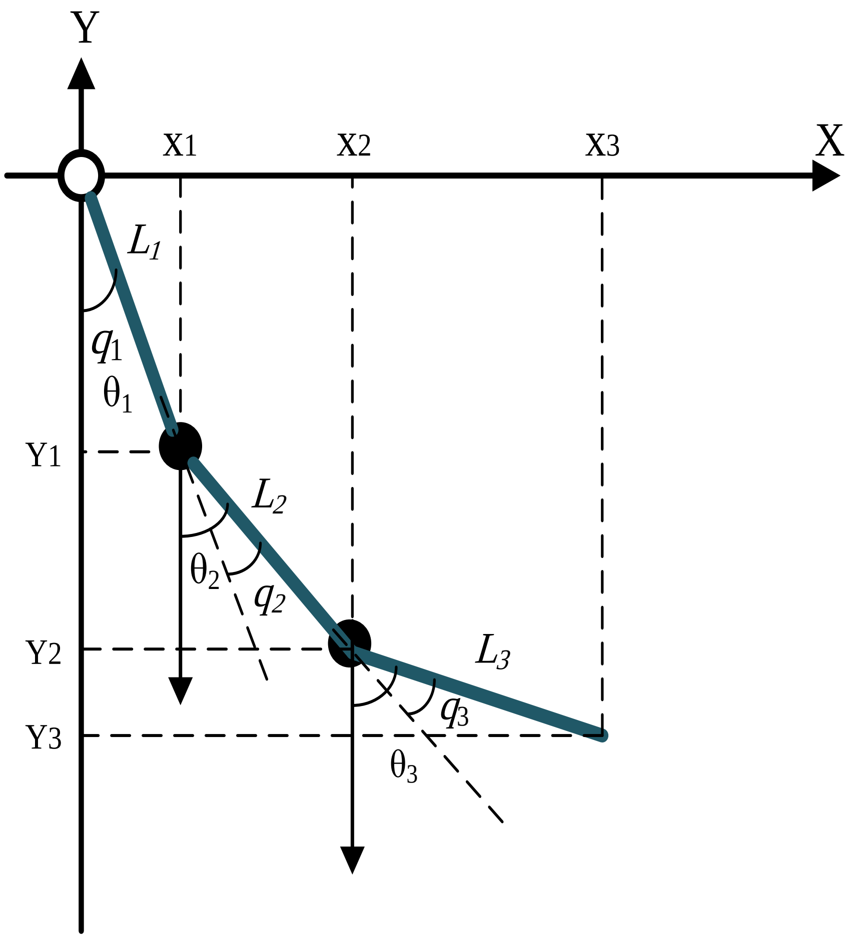


Figure 3.5 Multilink angles axis diagram

| **Angles** | **Velocity** |
|---|---|
| $\theta_1 = q_1$ | $\dot{\theta}_1 = \dot{q}_1$ |
| $\theta_2 = (q_1 + q_2)$ | $\dot{\theta}_2 = \dot{(q_1 + q_2)}$ |
| $\theta_3 = (q_1 + q_2 + q_3)$ | $\dot{\theta}_3 = \dot{(q_1 + q_2 + q_3)}$ |

The above symbols show the relationship between joint angles (θ) and joint positions (q) in a robot manipulator, where each joint's position q is added to the previous joint's position to obtain the corresponding joint angle θ.

$\theta_1$ equals the first joint position $q_1$. $\theta_2$ equals the sum of the first two joint positions $(q_1 + q_2)$. In the third point, $\theta_3$ equals the sum of all three joint positions $(q_1 + q_2 + q_3)$.

On the other hand, velocities represent the relationship between the time derivatives of the joint angles (θ) and the time derivatives of the joint positions ($\dot{q}$) in a robot manipulator, where each joint's velocity $\dot{q}$ is added to the previous joint's velocity to obtain the corresponding joint angle velocity $\dot{\theta}$.

These equations assume that the joints are serially connected, i.e., each joint is attached to the previous one in a chain-like manner. Therefore, the joint angle of a joint is the sum of all joint positions from the first joint up to that joint.

**Position:**

$$x_1 = L_1 \sin\theta_1 = L_1 \sin q_1 = L_1 S_1 \tag{3.1}$$

$$x_2 = L_1 S_1 + L_2 S_{1+2} \tag{3.2}$$

$$x_3 = L_1 S_1 + L_2 S_{1+2} + L_3 S_{1+2+3} \tag{3.3}$$

The equations represent the forward kinematics of a robot manipulator, specifically the Cartesian coordinates (x) of the end-effector in terms of the joint angles or positions ($\theta$ or $q$) and the length of each link in the manipulator ($L$).

In equation (3.1), $x_1$ represents the x-coordinate of the end-effector, which is equal to the length of the first link ($L_1$) multiplied by the sine of the first joint angle or position ($\theta_1$) or ($q_1$). This can also be written as $L_1$ times $S_1$. Where $S_1$ is the sine of ($q_1$).

In equation (3.2), $x_2$ represents the x-coordinate of the end-effector, which is equal to the sum of the x-coordinate of the first joint ($L_1$ times $S_1$) and the x-coordinate of the second joint ($L_2$ times $S_{1+2}$ ), where $S_{1+2}$ is the sine of $q_1$ and $q_2$.

In equation (3.3), $x_3$ represents the x-coordinate of the end-effector, which is equal to the sum of the x-coordinate of the first joint ($L_1$ times $S_1$), the x-coordinate of the second joint ($L_2$ times $S_{1+2}$), and the x-coordinate of the third joint ($L_3$times $S_{1+2+3}$), where $S_{1+2+3}$ is the sine of the sum of $q_1$, $q_2$ and $q_3$.

$$y_1 = -L_1 \cos\theta_1 = L_1 \cos q_1 = -L_1C_1 \quad (3.4)$$
$$y_2 = -L_1C_1 - L_2\,C_{1+2} = -(L_1C_1 + L_2\,C_{1+2}) \quad (3.5)$$
$$y_3 = -(L_1C_1 + L_2C_{1+2} + L_3C_{1+2+3}) \quad (3.6)$$

These equations represent the y-coordinate of the end-effector in terms of the joint angles or positions and the length of each link in the manipulator. In equation (3.4), $y_1$ represents the y-coordinate of the end-effector, which is equal to the negative of the length of the first link ($L_1$) multiplied by the cosine of the first joint angle or position ($\theta_1$) or ($q_1$). This can also be written as negative $L_1$ times $C_1$, where $C_1$is the cosine of $q_1$. Also, equation (3.5), $y_2$ represents the y-coordinate of the end-effector, which is equal to the sum of the y-coordinate of the first joint (negative $L_1$ times $C_1$) and the y-coordinate of the second joint (negative $L_2$ times $C_{1+2}$), where $C_{1+2}$ is the cosine of the sum of $q_1$ and $q_2$. Moreover, In equation (3.6), $y_3$ represents the y-coordinate of the end-effector, which is equal to the sum of the y-coordinate of the first joint (negative $L_1$ times $C_1$), the y-coordinate of the second joint (negative $L_2$ times $C_{1+2}$ and the y-coordinate of the third joint (negative $L_3$ times $C_{1+2+3}$). where $C_{1+2+3}$ is the cosine of the sum of $q_1$, $q_2$, and $q_3$.

**Velocity**:

$$\dot{x}_1 = L_1\,\dot{\theta}_1 \cos\theta_1 = L_1\,\dot{q_1} \cos q_1 = L_1\dot{q}_1C_1 \quad (3.7)$$
$$\dot{x}_2 = L_1\,\dot{q}_1C_1 + L_2\,(\dot{q}_1 + \dot{q}_2)\,C_{1+2} \quad (3.8)$$
$$\dot{x}_3 = L_1\,\dot{q}_1C_1 + L_2\,(\dot{q}_1 + \dot{q}_2)\,C_{1+2} + L_3\,(\dot{q}_1 + \dot{q}_2 + \dot{q}_3)\,C_{1+2+3} \quad (3.9)$$

These equations represent the velocity of the Robogymnast in three degrees of freedom, specifically $\dot{x}_1$, $\dot{x}_2$ and $\dot{x}_3$. The Robogymnast has three joints, each with a specific length, denoted as $L_1$, $L_2$ and $L_3$. And $C_{1+2+3}$ are the consine of the joint's angles.

$$\dot{y}_1 = L_1\,\dot{\theta}_1 \sin\theta_1 = L_1\,\dot{q_1} \sin q_1 = L_1\dot{q}_1s_1 \quad (3.10)$$
$$\dot{y}_2 = L_1\,\dot{q}_1s_1 + L_2\,(\dot{q}_1 + \dot{q}_2)\,s_{1+2} \quad (3.11)$$
$$\dot{y}_3 = L_1\,\dot{q}_1s_1 + L_2\,(\dot{q}_1 + \dot{q}_2)\,s_{1+2} + L_3\,(\dot{q}_1 + \dot{q}_2 + \dot{q}_3)\,s_{1+2+3} \quad (3.12)$$

These equations are similar to the ones mentioned earlier, except they represent the velocity of the Robogymnast in the y-axis direction. The values of $L_1$, $L_2$ and $L_3$ represent the lengths of the three joints of the robot. equation (3.10) defines the velocity of the first joint in the y-axis direction, $\dot{y}_1$ which depends on the joint's angular velocity $\dot{q}_1$ and the sine of the joint's angle, $s_1$. equation (3.11) defines the velocity of the second joint in the y-axis direction, $\dot{y}_2$, which depends on the angular velocities of the first and second joints, $\dot{q}_1$ and $\dot{q}_2$ and the sine of the sum of their respective angles, $s_{1+2}$. Finally, equation (3.12) defines the velocity of the third joint in the y-axis direction, $\dot{y}_3$, which depends on the velocities of all three joints $\dot{q}_1$ , $\dot{q}_2$ , and $\dot{q}_3$ , and the sine of the sum of their respective angles, $s_{1+2+3}$ These equations are critical to understanding the motion and behaviour of the Robogymnast in the y-axis direction, and they are important in designing and controlling the robot's movements.

### 3.3.1 Compute the Lagrange of the acrobat system:

$$L = T - V \tag{3.13}$$

Where:

| | |
|---|---|
| **L** | Lagrange |
| **T** | Kinetic Energy (K.E) |
| **V** | Potential energy (P.E) |

In (3.13) equation, L represents the Lagrangian, which is a function that describes the motion of a system. T represents the kinetic energy of the system, while V represents the potential energy.

The equation expresses the total energy of the system as the difference between its kinetic and potential energies. This formulation is often used to derive the equations of motion for a system using the principle of least action. The Lagrangian formulation is a powerful tool for analysing the dynamics of physical systems, and it is widely used in fields such as physics, engineering, and mathematics.

The total Kinetic energy (K, E) of the Acrobat system:

$$T = \frac{1}{2} m_1 v_1^2 + \frac{1}{2} m_2 v_2^2 + \frac{1}{2} m_3 v_3^2 \tag{3.14}$$

In above equation, T represents the total kinetic energy of the system, which is the energy associated with the motion of the particles. The subscripts 1, 2, and 3 refer to the three particles in the system, and $m_i$ and $v_i$ represent the mass and velocity of each particle, respectively.

The first term, $\frac{1}{2}m_1v_1^2$ represents the kinetic energy of the first particle, which is given by the product of its mass and the square of its velocity.

The second term, $\frac{1}{2}m_2v_2^2$ represents the kinetic energy of the second particle, which is also given by the product of its mass and the square of its velocity.

The third term, $\frac{1}{2}m_3v_3^2$ represents the kinetic energy of the third particle, which has a slightly different form. This is because the velocity of the third particle is raised to the power of 2, which indicates that its motion may be in a different direction than the first two particles.

The summation of kinetic energies of individual particles in equation (3.14) leads to the determination of the total kinetic energy of a given system. This equation serves as a valuable tool for computing the overall energy of a multi-particle system and comprehending the dynamics of the constituent particles.

$$v_1^2 = L_1^2\,\dot{q}_1^2 \tag{3.15}$$

Equation (3.15) is an expression of the kinetic energy of a particle in circular motion, in terms of its angular velocity and the length of the first link.

$$v_2^2 \;=\; L_1^2\,\dot{q}_1^2 + \; L_2^2\,(\,\dot{q}_1 + \dot{q}_2\,)^2 + 2\,L_1L_2\,\dot{q}_1\,(\,\dot{q}_1 \;+\; \dot{q}_2\,)\,C_1 \tag{3.16}$$

Equation (3.16) states of the kinetic energy of a second link of triple link system, where:

In this equation, $\dot{q}_1 + \dot{q}_2$ represent the angles that describe the positions of the first and second particles, respectively, $L_1L_2$ represent the lengths of the links 1 and 2, and $C_1$represents the cosine.

$$v_3^2 = L_1^2\,\dot{q}_1^2 + L_2^2\,(\dot{q}_1 \;+\; \dot{q}_2)^2 + L_3^2(\dot{q}_1 \;+\; \dot{q}_2 +\; \dot{q}_3)^2 + 2\,[(\,L_1\,L_2\,\dot{q}_1(\dot{q}_1 + \dot{q}_2)\,C_1 \tag{3.17}$$

$$+\,L_1\,L_3\,\dot{q}_1(\dot{q}_1 + \dot{q}_2 + \dot{q}_3)\,C_{2+3} + L_2\,L_2\,(\dot{q}_1 + \dot{q}_2)\,(\dot{q}_1 + \dot{q}_2 + \dot{q}_3)\,C_3]$$

Equation (3.17) represents a third link to calculate the kinetic energy of the Robogymnast system, where: $L_{1-3}$ denotes the links, $\dot{q}_{1-3}$ indicates the angels and $C_{1-3}$ signifies cosines.

The total Kinetic energy of the acrobat system: (K.E)

$$T = \frac{1}{2} m_1 v_1^2 + \frac{1}{2} m_2 v_2^2 + \frac{1}{2} m_3 v_3^2 \tag{3.18}$$

Equation (3.18) represents the total kinetic energy of the system, where T is the total kinetic energy, $m_1$, $m_2$ and $m_1$ are the masses of the three particles, and $v_3^1$, $v_2^2$, $v_3^2$ are the corresponding velocities.

$$T = \frac{1}{2} (m_1 + m_2 + m_3) L_1^2 \dot{q}_1^2 + \frac{1}{2} (m_2 + m_3) L_2^2 (\dot{q}_1 + \dot{q}_2)^2 + \frac{1}{2} m_3 L_3^2 (\dot{q}_1 + \dot{q}_2 + \dot{q}_3)^2 + (m_2 + m_3) L_1 L_2 \dot{q}_1 (\dot{q}_1 + \dot{q}_2) C_2 + m_3 L_1 L_3 \dot{q}_1 (\dot{q}_1 + \dot{q}_2 + \dot{q}_3) C_{2+3} + m_3 L_2 L_3 (\dot{q}_1 + \dot{q}_2) (\dot{q}_1 + \dot{q}_2 + \dot{q}_3) C_3 \tag{3.19}$$

Equation (3.19) is the total kinetic energy of the system, expressed in terms of generalized coordinates and velocities. It shows the contributions of each mass and its corresponding velocity term, as well as the interaction terms between them.

The total potential of the acrobat system:

$$V = m_1 \text{ g } y_1 + m_2 \text{ g } y_2 + m_3 \text{ g } y_3 \tag{3.20}$$

Equation (3.20) gives the potential energy of the system, which is the sum of the potential energies of each mass. It is given by:

$$V = - (m_1 + m_2 + m_3) L_1 \text{ g } C_1 - ( m_2 + m_3) L_2 \text{ g } C_{1+2} - m_3 L_3 \text{ g } C_{1+2+3} \tag{3.21}$$

Where g is the gravity force, equation (3.21) gives the potential energy of the system due to gravity. Here, $C_{1+2+3}$ denotes the cosine of all links, g is the gravity force. The negative sign indicates that the potential energy decreases as the height above the reference level increases.

**L = T – V**

Lagrange is given by L = T - V

$$L = \frac{1}{2} (m_1 + m_2 + m_3) L_1^2 \dot{q}_1^2 + \frac{1}{2} (m_2 + m_3) L_2^2 (\dot{q}_1 + \dot{q}_2)^2 + \frac{1}{2} m_3 L_3^2 (\dot{q}_1 + \dot{q}_2 + \dot{q}_3)^2 + (m_2 + m_3) L_1 L_2 \dot{q}_1 (\dot{q}_1 + \dot{q}_2) C_2 + m_3 L_1 L_3 \dot{q}_1 (\dot{q}_1 + \dot{q}_2 + \dot{q}_3) C_{2+3} + m_3 L_2 L_3 \dot{q}_1 (\dot{q}_1 + \dot{q}_2)(\dot{q}_1 + \dot{q}_2 + \dot{q}_3) C_3 + (m_1 + m_2 + m_3) L_1 \text{ g } C_1 + (m_2 + m_3) L_2 \text{g } C_{1+2} + m_3 L_3 \text{ g } C_{1+2+3} \tag{3.22}$$

Equation 3.19 where T is the kinetic energy given by equation (3.18) and V is the potential energy given by equation (3.20). Thus, equation (3.21) represents the total energy of the system, which is conserved if there is no external force acting on the system. The Lagrangian formulation of mechanics provides an alternative way to derive the equations of motion of a system compared to the Newtonian formulation.

By using (E – L)

The Euler-Lagrange equation to obtain the equation of motion:

$$\tau_i = \frac{d}{dt}\left(\frac{\vartheta L}{\vartheta \dot{q}_i}\right) - \frac{\vartheta L}{\vartheta q_1} = 0 \qquad i = 1, 2, 3 \qquad (3.23)$$

This is an equation representing the Lagrange equation of motion for the *i-th* generalized coordinate of a system. The equation states that the time derivative of the partial derivative of the Lagrangian L with respect to the *i-th* generalized velocity *q̇-i*, minus the partial derivative of L with respect to the *i-th* generalized velocity *q̇-i*, is equal to zero. This equation is used to derive the equations of motion for the system in terms of its generalized coordinates and velocities, where:

$\tau_i$: This symbol represents the generalized force associated with the $i_{th}$ degree of freedom in a mechanical system. It can be thought of as the rate of change of the system's momentum with respect to that degree of freedom. In this equation, $\tau_i$ represents the force acting on the system in the *i-th* degree of freedom.

$\frac{d}{dt}$: This is the notation for the derivative with respect to time. When applied to a function, it gives the rate of change of the function with respect to time.

$(\frac{\vartheta L}{\vartheta \dot{q}_i})$: This is the partial derivative of the Lagrangian function $L$ with respect to the i-th generalized velocity $\dot{q}_i$. It measures how much the Lagrangian changes as the velocity of the i-th degree of freedom changes.

$\frac{\vartheta L}{\vartheta q_1}$ This is the partial derivative of the Lagrangian function $L$ with respect to the i-th generalized coordinate, $q_1$. It measures how much the Lagrangian changes as the position of the i-th degree of freedom changes.

$i$ = 1, 2, 3: This equation is written for a system with three degrees of freedom. The index $i$ is used to distinguish between the three degrees of freedom. The equations for $i$=1, $i$=2, and $i$=3 represent the forces acting on the system in each of these degrees of freedom.

The given equations are just the Euler-Lagrange equation applied to the specific degrees of freedom *(i* = 1, 2, 3). The equation states that the time derivative of the partial derivative of the Lagrangian with respect to the generalized velocity minus the partial derivative of the Lagrangian with respect to the generalized coordinate equals zero (or the generalized force). This holds for each degree of freedom, giving us three equations in (3.24):

$$\tau_i = \frac{d}{dt}\left(\frac{\vartheta L}{\vartheta \dot{q}_2}\right) - \frac{\vartheta L}{\vartheta q_2} = 0 \tag{3.24}$$

Where, $\tau_i = \tau_1, \tau_2, \tau_i$

The numerical model of the Robogymnast is calculated by substituting the values of the parameters given in Table 3.1.

Table 3.1 Robogymnast Parameters

| **Parameters** | **Symbol** | **Mean Values** |
|---|---|---|
| Length of the first link | $L_1$ | 0.16 m |
| Length of the second link | $L_2$ | 0.18 m |
| Length of the third link | $L_3$ | 0.24 m |
| Weight of the first link | $m_1$ | 1.2 kg |
| Weight of the second link | $m_2$ | 1.2 kg |
| Weight of the third link | $m_3$ | 0.5 kg |
| Angles between pole 1,2,3 | $\theta$ | $\theta_1$, $\theta_2$, $\theta_3$ (rad) |
| Initial value of the angles | $q_1$, $q_2$, $q_3$ | 0 (rad) |
| Gravity | g | $9.81 m/s^2$ |

Table 3.2 Stepper Motor Parameter

| **Stepper Motor** | **Gear box** | **Encoder** |
|---|---|---|
| Step Angle without Gearbox: 1.8 degree | Gearbox Length: 35mm | Driving Voltage: 4.5V to 5.5V |
| Holding Torque without Gearbox:0.52Nm(73.65oz.in) | Gear Ratio: 13.73 | Output current: 20mA |
| Rated Current/phase: 1.68A | Efficiency: 81% | Resolution: 1000 Pulses per revolution (PPR) |
| Phase Resistance: 1.8ohms | Shaft Diameter: Φ8mm | Output Signal: 2 ch |
| Voltage: 2.80V | Shaft Length: 20mm | Bore Diameter: Φ5mm |
| Inductance: 3.20mH ± 20%(1KHz) | D-Cut Length: 15mm | Output Frequency: ≤60MHz |
| Number of Leads: 4 | Backlash at No-load: <=1deg | Running Speed: ≤3600rpm |
| Lead Length: 300mm | Torque: 3- 5 Nm(424.92oz.in) | IP Protection: IP20 |

The Robogymnast system in its vertical position was modelled in a linear, continuous-time, state-space manner via MATLAB and its various tools, along with further M-files produced by the author. For this reason, the system has to linearizing as in the following subsection:

### 3.3.2 Model Linearization of the system

The nonlinear of state-space form the Robogymnast system.

$$\dot{x} = f(x, u) \tag{3.25}$$

Where:

$\dot{x}$ is the state vector of Acrobat, $x^T = [q_1 \quad q_2 \quad q_3 \quad \dot{q_1} \quad \dot{q_2} \quad \dot{q_3}]$ . $u$ is the control input $u = \tau$ ( $\tau$ is the applied input vector), it is a scalar because there is only one actuator that provides torque input to the system. The expression for the angular acceleration, $\ddot{q} = [\ddot{q}_1 \ ; \ddot{q}_2 \ ; \ddot{q}_3]^T$ , was obtained by solving for $\ddot{\theta}$ in equation (3.26).

$$\ddot{q} = M^{-1}[\tau - C(q, \dot{q}) - G(q)] \tag{3.26}$$

Equation (3.26) is of the form $\dot{x} = f(x, u)$. The term $\dot{x}$ is a $2 \times 1$ matrix, which contains nonlinear elements. In these forms the first two elements of $\dot{x}$ are just the last two elements of x, also, G is the gravity vector. To linearize the last two elements about an operating point vector (OP), Taylor's expansion is used:

$$\delta\dot{x}(t) = \left(\frac{\partial f(x, u)}{\partial x}\right)_{x=OP, u=0} \delta x(t) + \left(\frac{\partial f(x, u)}{\partial u}\right)_{x=OP, u=0} \delta u(t) \tag{3.27}$$

$\dot{x}(t)$ is a small deviation of the states from the operating point. The coefficients of $\delta x(t)$ and $\delta u(t)$, termed A and B, respectively, are evaluated at the operating point. Thus, the linear state-space model of the system is given in the equation system becomes:

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \ + Du \end{aligned} \tag{3.28}$$

$\dot{x}$ is the state vector of the system, *y* is the output vector.

Table 3.1 provides nomenclature and values for the parameters. A, B, C, and D represent matrices for state-space modelling. Anti-swing controllers aim to give stability to pendulum links when aligned vertically downwards while minimizing vibrations. The point of stable equilibrium corresponding to the states of the links is $\theta_1 = \theta_2 = \theta_3 = 0$. The state space of the Robogymnast equations can be written as:

$$A = \begin{bmatrix} 0_3 & I_3 \\ A_{21} & A_{22} \end{bmatrix}$$

Where:

$$0_3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad I_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$A_{21} = \begin{bmatrix} 0 & 2.6825 & -0.0657 \\ 0 & 29.2751 & -15.8236 \\ 0 & -57.5286 & 247.5924 \end{bmatrix}$$

$$A_{22} = \begin{bmatrix} -0.0286 & -0.0083 & 0.0284 \\ -0.0391 & -0.1957 & 1.2358 \\ 0.0589 & 1.4085 & -18.0527 \end{bmatrix}$$

The values of above parameters obtained from matrix A by applying system parameters. The numerical model of the Robogymnast was calculated by MATLAB/toolbox to obtain A, also, State-space matrices for Robogymnast (A, B, C, and D) can be shown as below:

$$\mathrm{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 2.6825 & -0.0657 & -0.0286 & -0.0083 & 0.0284 \\ 0 & 29.2751 & -15.8236 & -0.0391 & -0.1957 & 1.2358 \\ 0 & -57.5286 & 247.5924 & 0.0589 & 1.4085 & -18.0527 \end{bmatrix}$$

$$\mathrm{B} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1.0314 \\ 1.6582 \\ -2.4837 \end{bmatrix} \qquad \mathrm{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad \mathrm{D} = [0]$$

The discrete-time model of Robogymnast is obtained by discretizing equations (3.28), then implementing these equations via MATLAB command window to apply mathematical model matrices into simulation part to find out the result.

### 3.3.3 Controllability and Observability of the sate-space

In control theory, the controllability and observability of a system are two important concepts that describe the ability of an external input and internal state of a system to be measured or controlled, respectively.

Controllability measures the ability of a particular actuator configuration to control all the states of the system; conversely, observability measures the ability of the sensor configuration to supply all the information necessary to estimate all the states of the system. Classically, control theory offers controllability and observability tests which are based on the rank deficiency of the controllability and observability matrices. Where this calculation was determined by MATLAB.

The system is controllable if the controllability and observability matrices are full rank. This answer is often not enough for practical engineering problems where it needs more quantitative information [118]. The rank of a matrix is the maximum number of linearly independent rows or columns in the matrix. For controllability, if the controllability matrix has full rank, it means that its columns are linearly independent. This implies the ability to apply control inputs that can reach within the system's state space.

Co = Ctrb(sys.A,sys.B);                                        where Co=Ctrb meaning controllable

rank (Co),   ans = 6

'*It is controllable'*

Observability, on the other hand, refers to the ability to determine the internal state of a system from its output measurements. A system is said to be observable if its internal states can be determined uniquely from its output measurements. Full rank in the observability matrix indicates linear independence of its rows, allowing accurate estimation or observation of the system's state variables based on input and output measurements. Similarly, full rank in controllability and observability matrices ensures complete controllability and observability, respectively, enabling effective control of the system's states and accurate estimation or observation using available measurements.

Ob = Obsv(sys.A, sys.C);                                   where Ob= Obsy meaning Observable

rank(Ob), ans = 6

*'It is observable'*

## 3.4 System design

In this section the setup of the Robogymnast system is discussed, all the equipment that has been used in this research to carry out the experimental tests. The test rig and the data acquisition have been also described. Furthermore, the data collection procedure has been explained. It also presents the multi-link motion signals of the Robogymnast to be used to detect results to optimise it by selected algorithms in the next chapters.

### 3.4.1 Physical body

The robot's body is similar to that of a human gymnast, with two arms, two legs, and a torso. This enables the system to perform a wide range of gymnastic movements, such as swing and manipulation.

The materials used to build the robot's body is SLS material, which is the stands for Selective Laser Sintering, where a 3D printing technology that uses a laser to fuse together small particles of material into a solid object. SLS is often used to create complex, high-quality parts with a high degree of accuracy and durability. In order to ensure stability during gymnastics movements, the robot's body needs to be able to maintain balance and adjust its movements in real-time. This could be achieved through the use of sensors that detect the robot's orientation and position, as well as advanced control algorithms that enable the robot to adjust its movements as needed. Also, the sensors are used to detect its surroundings and the position of the system, such as encoders and potentiometer. These sensors provide feedback to the multi-link control system, enabling it to adjust its movements as needed.

Overall, the design of the Robogymnast body would be a complex and challenging task, requiring expertise in materials science, robotics, biomechanics, and control theory. However, with the right combination of technologies and design features, it is possible to create a robot that can perform a wide range of gymnastics movements with precision and accuracy. Figure 3.6 shows the Robogymnast body’s itself, then the components of the system are illustrated and discussed in this chapter.

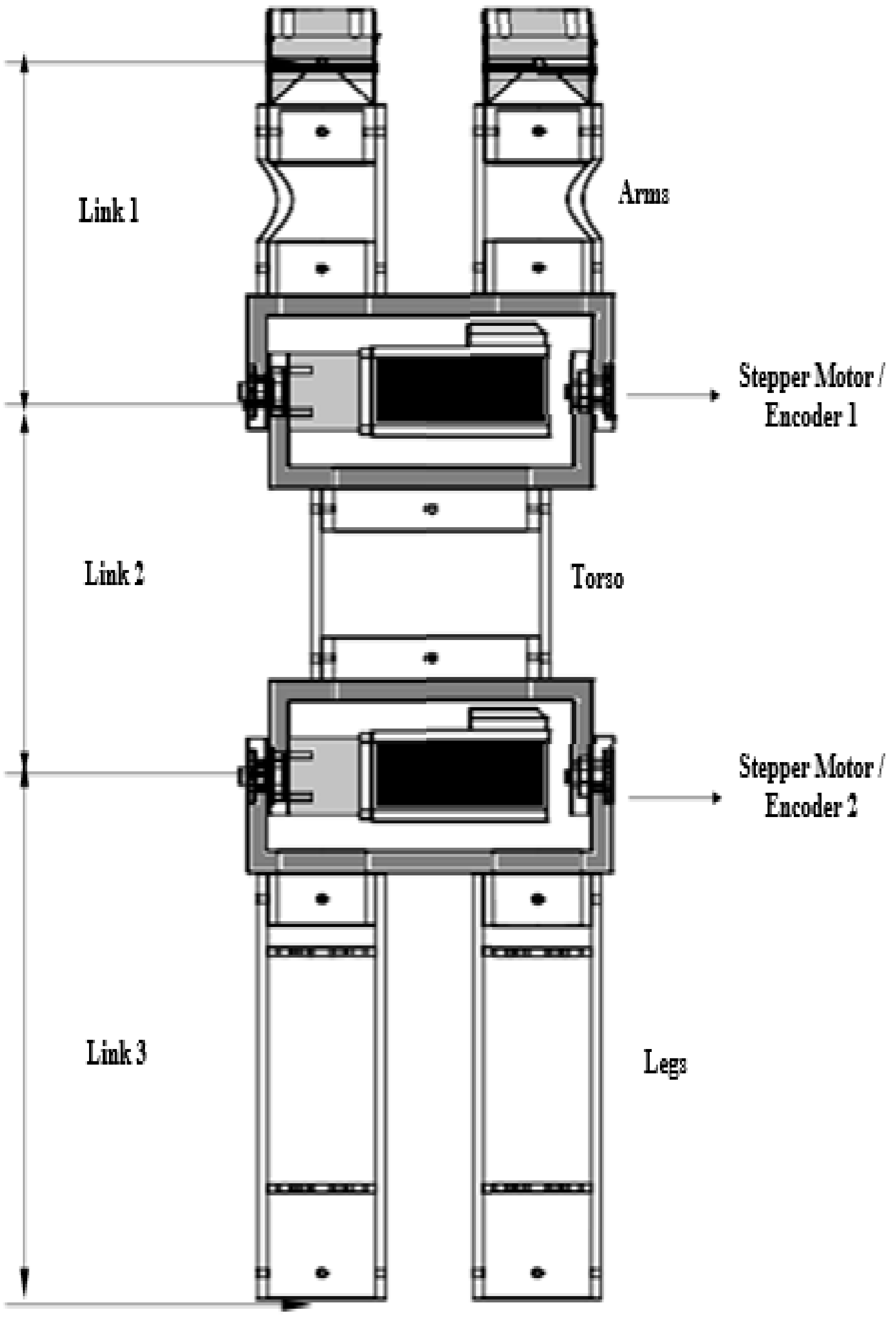


Figure 3.6 Robogymnast initial design

This subsection demonstrates the components used to build the robotic system such as the actuators and sensors.

### 3.4.1.1 Stepper motor

Stepper motors utilize a toothed rotor with a set of magnets that are attracted by multiple field windings in order to control the position of the motor in discrete steps. They have the benefit of not requiring closed loop feedback, but can lose steps under high shock load, which legged robot joints regularly experience. Although adding closed loop feedback could solve this issue, stepper motors area also typically extremely heavy and have very low transparency [119]. Figure 3.7 shows the stepper motor attached with encoder.

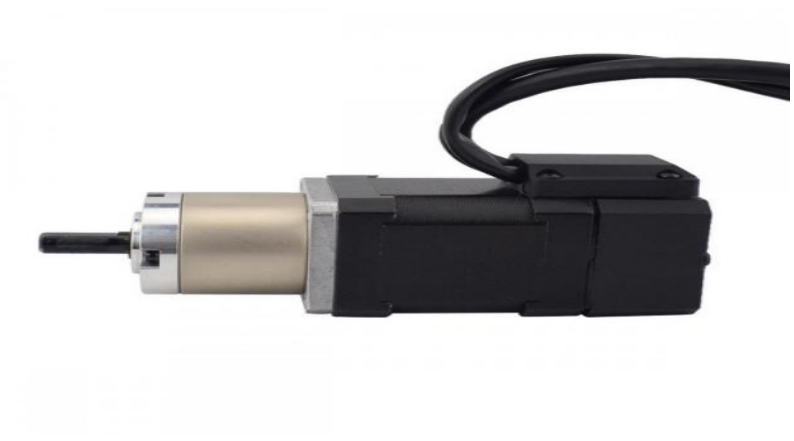


Figure 3.7 Stepper motor with Encoder [120]

### 3.4.1.2 Rotary Encoder (Header)

As shown in Figure 3.8 the SICK TTL DBS36 Incremental encoder provides a compact, robust, and easy-to-install solution for many speed and position applications. It provides 1024 PPR at its output, and the 1.5m long cable can be routed axially or radially to suit the installation conditions. The encoder has an 8mm hollow shaft, which can be reduced to 6mm using collet accessory. The encoder stator coupling has slots to enable flexible mounting on a Pitch Circle Diameter (PCD) ranging from 42-46 mm. A wide operating temperature range of -20 to +85 degrees Celsius and it has IP65 protection that ensures reliable operation even in harsh operating conditions.

Operating current is = 50 mA     Supply voltage 4.5 to 5.5V

Figure 3.8 Rotary Encoder [121]

#### 3.4.1.3 Rotary Encoder

The SP22E-10K is a Precision Single Turn Potentiometer with conductive plastic element, gold-plated terminals, high temperature thermoplastic housing and stainless-steel shaft, 320 ±5° Electrical angle, and 320 ±5° Mechanical angle. Figure 3.9 illustrates the Rotary potentiometer.

Figure 3.9 Rotary Potentiometer [122]

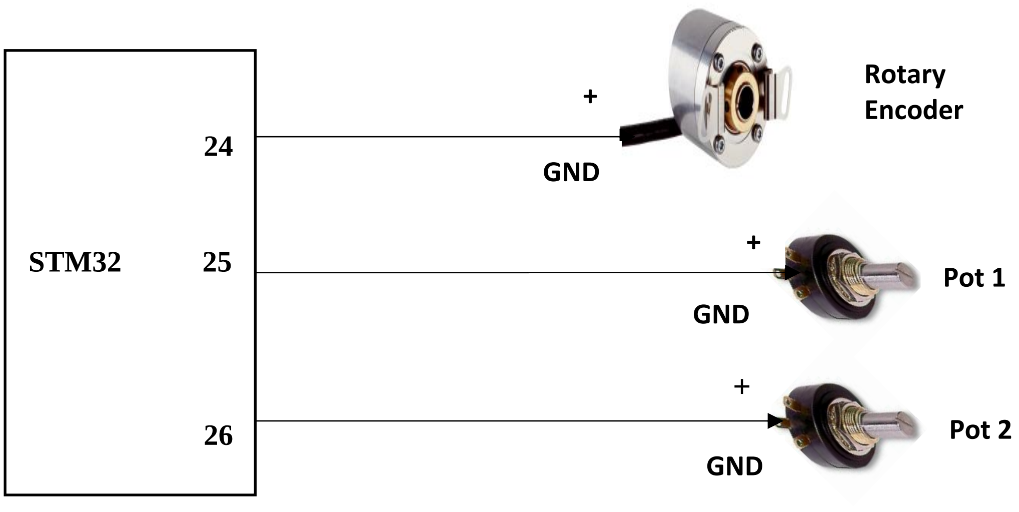


Figure 3.10 STM32 Encoders system

The encoder and potentiometers are utilized for precise data collection which is then sent to the microcontroller. The microcontroller displays this data as an output on a PC through the use of STM32, as illustrated in Figures 3.10 and 3.11. To achieve this, pin 24 is connected to the encoder header while pins 25 and 26 are connected to potentiometers 1 and 2 respectively.

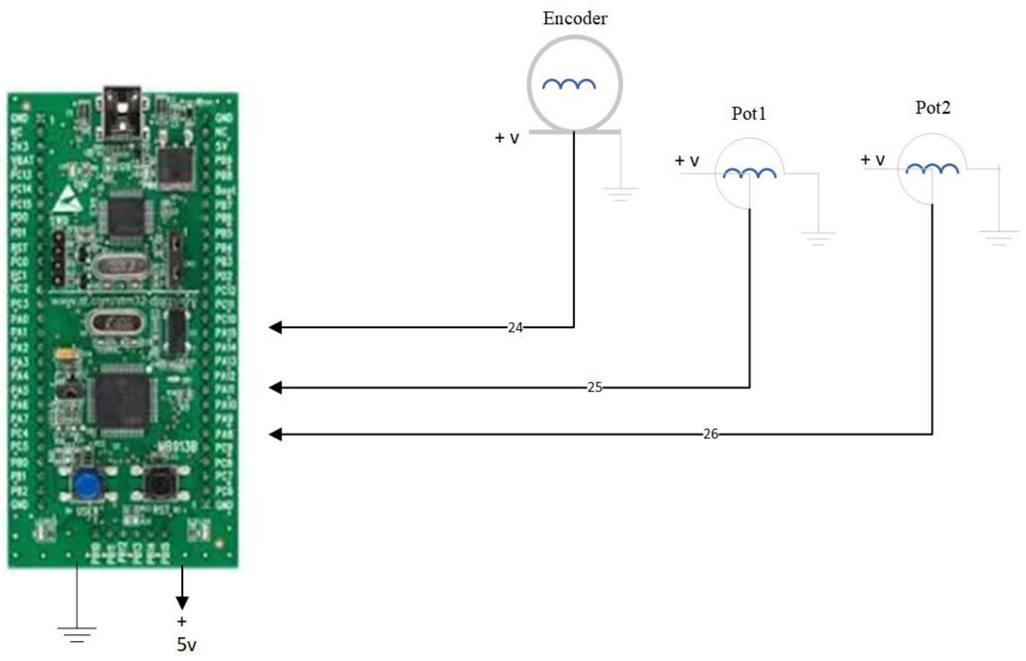


Figure 3.11 Sensors Circuit

### 3.4.2 Operation system

The operation of a Robogymnast system involves several different components working together to enable the robot to perform gymnastics movements with precision and control. Here are some additional details on the key components of a Robogymnast system, in this subsection, the components of the Robogymanst operation system which move and control the system. The main parts of this system are the motor driver which can control and send the commands to the stepper motor, as well as the microcontroller which using to program the whole system and connect all parts simultaneously. Figure 3.12 shows the microcontroller and the motor drivers.

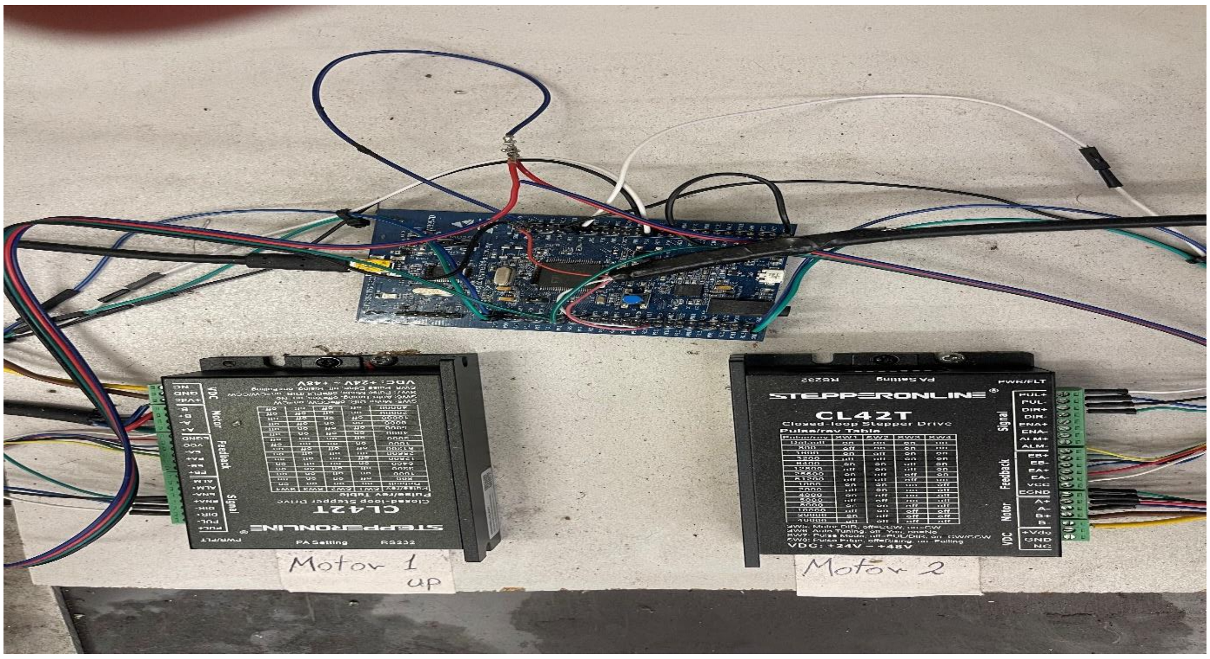


Figure 3.12 STM32 stepper driver

#### 3.4.2.1 Stepper motor driver

According to [117], as shown in Figure 3.13, the CL42T is a closed loop stepper driver designed to resolve the problem of lost steps in open loop stepper control systems. This increases system reliability with minimal cost increase. It utilizes advanced control algorithms based on decades of experience in stepper and servo controls. The CL42T is highly reliable, affordable, and performs exceptionally well in various industrial applications such as CNC, medical, electronics, and packaging. The CL42T can power 2-phase NEMA11, 14, and 17 stepper motors with incremental encoders, but the encoder resolution must be 1000-line. Compared to traditional open loop stepper systems, the CL42T's closed loop system can eliminate the potential for lost steps, perform real-time position error correction, and does not require torque reservation (100% torque implementation). Additionally, it can power the driven stepper motor with reduced heating, lower noise, and lower vibration.

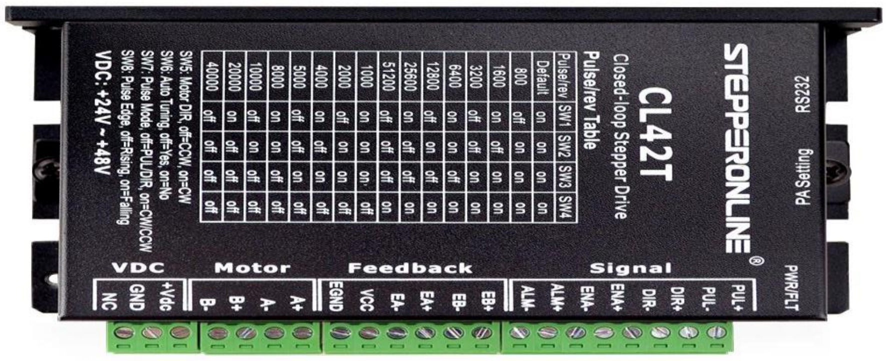


Figure 3.13 Stepper Driver [123]

Table 3.3 Stepper Driver Specifications

| **Key Features:** | **Electrical Specifications** | **Operating Environment** |
|---|---|---|
| No loss of step | Output Peak Current: 0~3 A | |
| No torque reservation | Input Voltage: +24~48VDC (Typical 24VDC) | Cooling: Natural Cooling or Forced cooling |
| No hunting or overshooting | Logic Signal Current: 7~16mA (Typical 10mA) | Environment: Avoid dust, oil fog and corrosive gases |
| No tuning for easy setup | Pulse Input Frequency: 0~200kHz | Ambient Temperature: 0℃ ー 65℃ |
| **2**4-48VDC supply voltage, max 3A output current | Pulse Width: 2.5μS | Operating Temperature: 0℃ ー 50℃ |
| Max 200 kHz input frequency | Isolation Resistance: 500MΩ | Vibration: 10-50Hz / 0.15mm |
| 15 micro step settings of 800-51,200 via DIP switches, or 200-51,200 via software (increase by 200) | | Storage Temperature: -20℃ ー 65℃ |
| Protections for over voltage, over current and position following error | | |

### 3.4.2.2 Microcontroller (STM-32)

A microcontroller as shown in Figure 3.14, is a small computer on a single integrated circuit that is designed to control specific devices or processes. It typically includes a central processing unit (CPU), memory, input/output, and other specialized hardware components to enable specific functions. The STM-32 microcontroller is a family of microcontrollers developed by STMicroelectronics. It is based on the ARM Cortex-M processor architecture and is designed for use in embedded systems and applications that require real-time control, low power consumption, and high performance [124].

The STM32F427xx and STM32F429xx devices are based on the high-performance Arm® Cortex®-M4 32-bit RISC core operating at a frequency of up to 180 MHz. The STM32F4 devices incorporate high-speed embedded memories (Flash memory up to 2 Mbyte, up to 256 Kbytes of SRAM, up to 4 Kbytes of backup SRAM, and an extensive range of enhanced I/0 and peripherals connected to two APB buses.

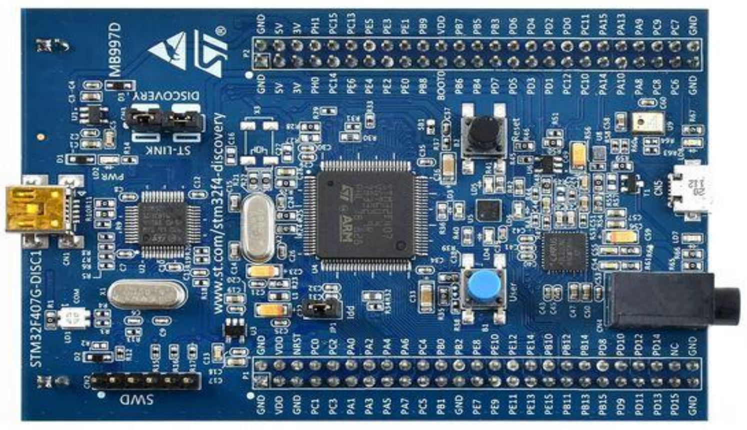


Figure 3.14 STM-32 Microcontroller [124]

These features make the STM32F427xx and STM32F429xx microcontrollers suitable for a wide range of applications:

- Motor drive and application control
- Home audio appliance
- Alarm systems, video intercom, and HVAC
- Medical equipment
- Printers, and scanners
- Industrial applications: PLC, inverters, circuit breakers

### 3.4.2.3 Power supply

As shown in Figure 3.15, the Professional 150W 48V 3.1A Switching CNC Power Supply is suitable for a wide range of applications in Industrial Automation and CNC Stepper/Servo Systems. It can be operated with either 115V or 230V power supply, selectable via a switch.

The power supply boasts powerful features such as PWM control, ensuring high efficiency and reliability. Its professional design further enhances its reliability and robustness. In addition, the low cost of the Professional 150W 48V 3.1A Switching CNC Power Supply makes it an attractive option for general use Table 3.4 shows some of the key features of the power supply.

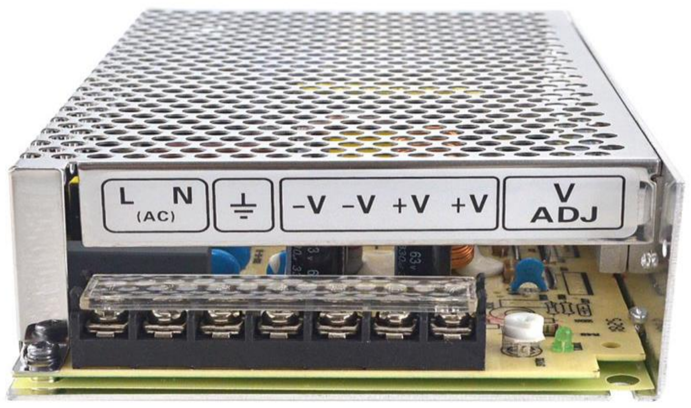


Figure 3.15 Power Supply [125]

Table 3.4 Power Supply Features

| | |
|---|---|
| 48V DC 3.1 A output | High efficiency low cost |
| AC input voltage range:92~132V/180~264VAC | Free air-cooling convection |
| 115V/230V AC selected by switch | Over current, over voltage, short circuit and overheat protections |

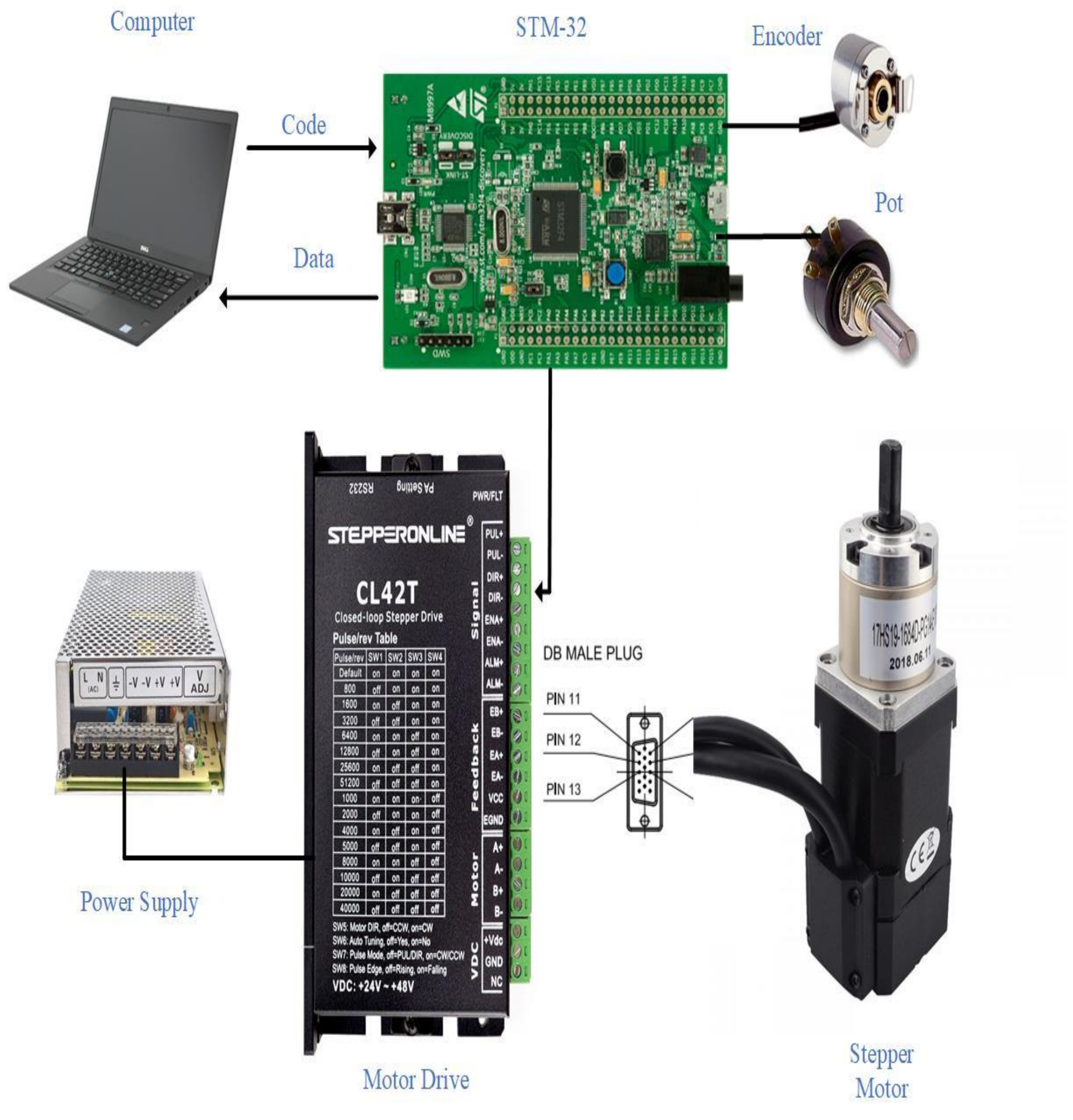
#### 3.4.2.4 PC

Figure 3.16 Dell PC [126]

As illustrated in Figure 3.16 Dell PC DESKTOP-INTNUGP Intel (R) Core (TM) i5-8265U CPU @ 1.60GHz   1.80 GHz is used to program the system, in addition to programming the system, the Dell PC is also used to write the STM-32 programming code, which is the code that will be loaded onto the STM32 microcontroller. The STM32 microcontroller is a type of embedded system that is commonly used in a variety of applications, from simple sensor readings to complex control systems. This processor runs at a base frequency of 1.60GHz but can reach up to 1.80GHz in turbo mode, allowing for fast and efficient programming.

Ultimately, the Dell PC is also responsible for providing 5V power to the STM32 microcontroller. This power supply is necessary for the proper operation of the microcontroller, as it requires a steady and reliable source of power to function correctly. Overall, the use of a powerful and capable computer like the Dell PC with an Intel Core i5-8265U processor is crucial for the successful programming and operation of the STM32 microcontroller system.

### 3.4.2.5 Operation system overview



| PIN | 1 | 2 | 3 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|
| Encoder | EA+ | VCC | GND | EB+ | EB- | EA- |

ENCODER RESOLUTION:1000PPR(4000CPR)

| 15-PIN DB MALE PLUG | 1 | 2 | 3 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|
| LEAD COLOR | BLK | RED | WHT | YEL | GRN | BLU |
| DESCRIPTION | EA+ | VCC | GND | EB+ | EB- | EA- |

Figure 3.17 Robogymnast Operation System

Here, is the final portrait of the multi-link robotics system (Robogymnast) designed and built at Cardiff University ready to motion.

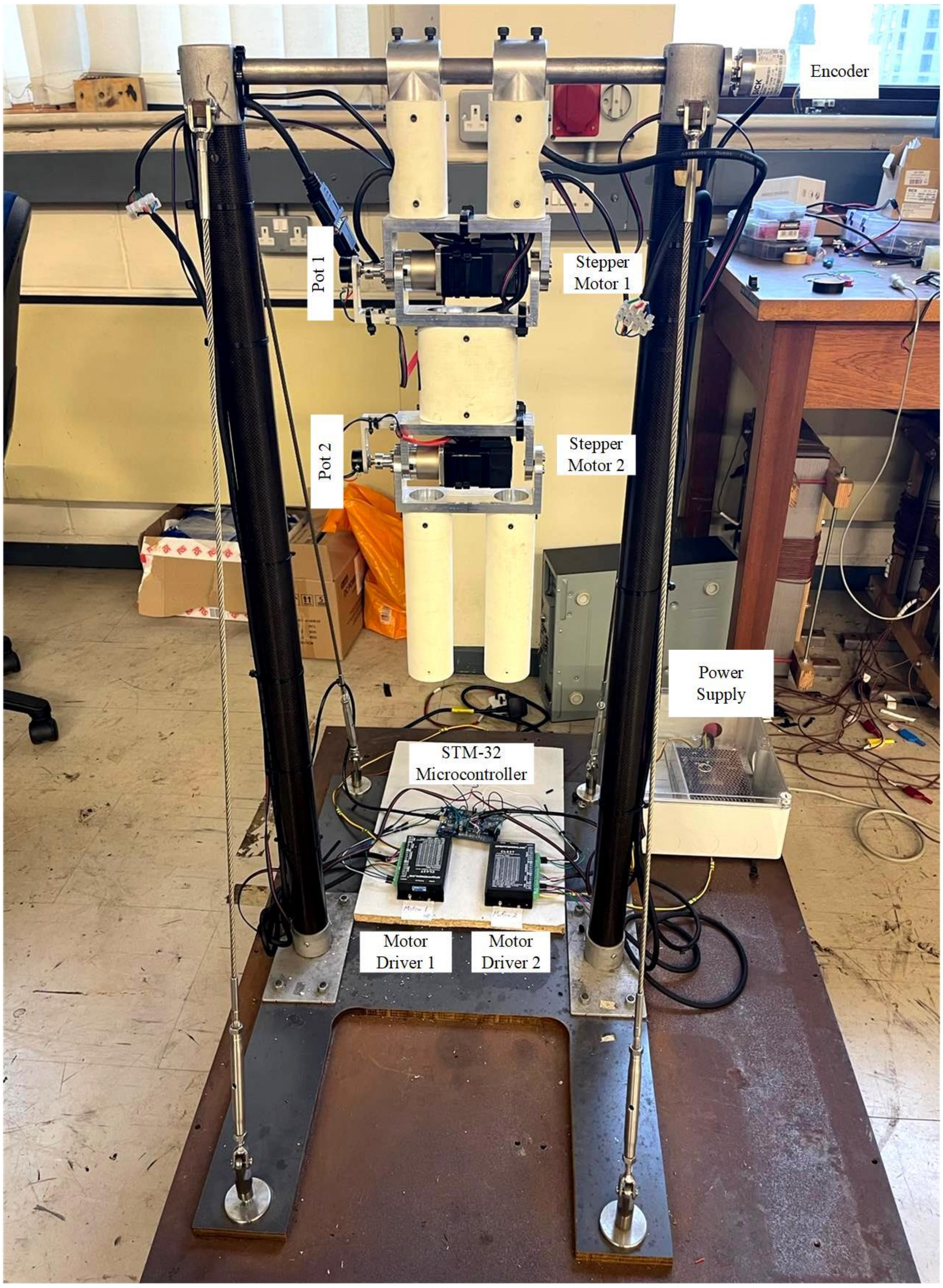


Figure 3.18 Final picture of the system

## 3.5 Summary

This chapter has provided a description of Robotgymnast's system and its mathematical modelling, as well as the setup of the system and the structure of the components is described. A detailed description of the entire Robogymnast system and its components was given. The system's process flow was described and illustrated. The objectives of the work presented in this chapter were to compute and design the simple mathematical model of the triple link robotic system (Robogymnast), a mathematical model of the Robogymnast has been derived based on the Euler-Lagrange approach describing the system dynamics. The linearised equations of motion and their state-space representation were then introduced. In Chapter 4, the swing-up control design of the Robogymnast is implemented and discussed.

# Chapter 4:

## Stability criteria and control strategies

### 4.1 Introduction

In this chapter, an approach to controlling the three-link Robogymnast robotic gymnast and assessing stability is proposed and examined. In the study, a conventionally configured linear quadratic regulator is applied and compared with a fuzzy logic linear quadratic regulator hybrid approach for stabilising the Robogymnast.

This section investigates the factors affecting the control of swing-up in the underactuated three-link Robogymnast. Moreover, a system simulation using MATLAB Simulink is conducted to demonstrate the impact of factors such as overshoot, rise time, and settling time. Additionally, the study includes the linearization of a mathematical model for the system, exploring the application of Lagrange's equation to determine the state space. The fuzzy logic linear quadratic regulator controller is employed to assess the extent to which the system responses are stabilised when implemented.

The Robogymnast is designed to replicate the movements of a human as they hang with both hands holding the high bar and then work to swing up into a handstand while still gripping the bar. The system has a securely attached link between the hand element and the shaft, which is mounted on ball bearings and can rotate freely.

The principal objective of the study lies in investigating how a linear quadratic regulator or fuzzy logic controller with a linear quadratic regulator (FLQR) can be applied to the Robogymnast, and to assess system behaviour under five scenarios, namely the original value, this value plus or minus ±25%, and plus or minus ±50%. In order to further assess the performance of the controllers used, a comparison is made between the outcomes found here and findings in the recent literature with fuzzy linear quadratic regulator controllers.

### 4.2 Stability control design

This section discusses the control methods applied within the study in detail. This begins with a discussion of the function of the LQR control system and how this was implemented for the parameters of the robotic system, with a similar discussion following for the FLQR control system. The stability control design is the process of designing a control system that ensures stability of a dynamic system [127]. A dynamic system is considered stable if its response is bounded and approaches state value over time, even in the presence of disturbances. Then a comprehensive investigation of both controllers is verified by comparing an integral time absolute error (ITAE) of the multi-link selected system.

This chapter discusses the control methods applied within the study in detail. This begins with a discussion of the function of the LQR control system and how this was implemented for the parameters of the robotic system, with a similar discussion following for the FLQR control system.

### 4.2.1 LQR control

The LQR approach is an active technique to acquire a controller for complicated systems performance [70]. Also, the State Feedback Control (SFC) utilizes the locations of the poles in the system, which are placed based on the gain matrix K along with state variables [128]. SFC allows for closed-loop system poles to be located at any desired point, but in output feedback control approaches, these poles can be set at a defined location [71]. The method uses a state feedback controller to implement each state variable, and these variables form feedback. The feedback components [129], after multiplication through the state feedback gain matrix, are used for comparison with reference inputs. SFC is primarily applied for gain matrix calculation [130].

LQR controllers are frequently utilized for this aim, and optimally for these controllers, K matrix parameters would include cost function (J) to optimize states, x(t), and system control signal u(t) [62]. In which Q represents constant symmetry positive, with matrix R being constant, optimal control can be given as:

The algebraic Riccati equation is used to calculate K and P values.

$$u(t) = -K\,x(t) \tag{4.1}$$

$$J = \frac{1}{2}\int_0^{\infty}(x^t\,Qx + u^t\,Ru)\,dt \tag{4.2}$$

$$U = R^{-1}\,B^T\,PX = -KX \tag{4.3}$$

The algebraic Riccati equation is used to calculate K and P values.

$A^T + PA - PBR^{-1}\, B^T P + Q = 0$ (4.4)

K = $R^{-1}B^T P = [K_1\ \ K_2\ \ K_3\ \ \ldots . K_6$ (4.5)

K = [0.2581 22.789 -507.886 0.940 -12.250 -19.480]

In this work, LQR is implemented using MATLAB/Simulink, as shown in Figure 4.1. The implementation involves using the LQR function in MATLAB to compute the optimal gain matrix, and then applying the gain to the system input and generating the controlled output using the LQR block in Simulink. This approach provides an efficient and flexible way to design and simulate LQR controllers for a wide range of control systems. Additionally, the value of K is obtained by applying the LQR parameters in MATLAB.

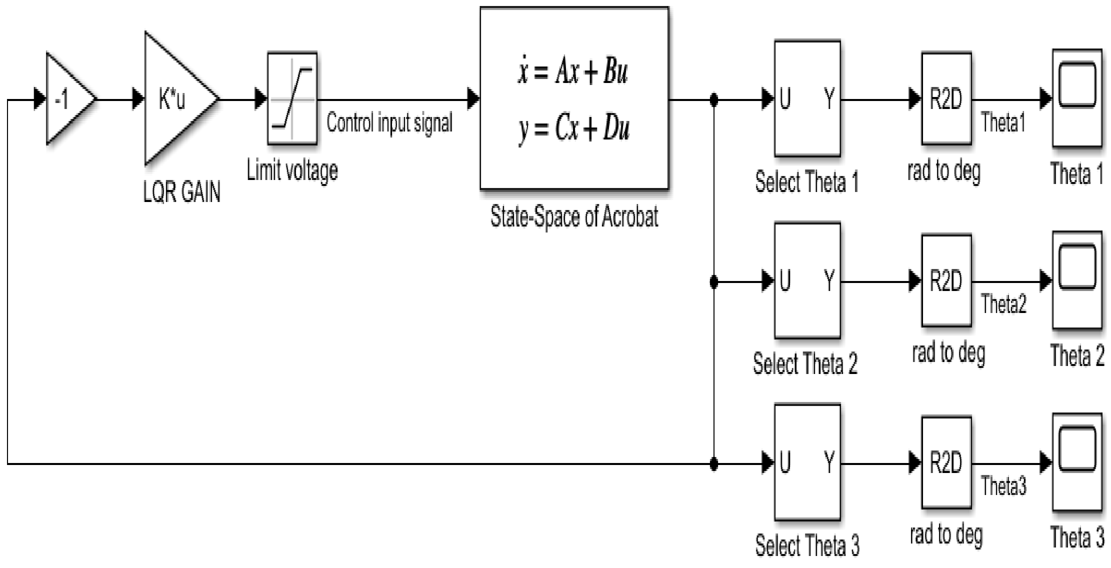


Figure 4.1 LQR Simulink

LQR Results:

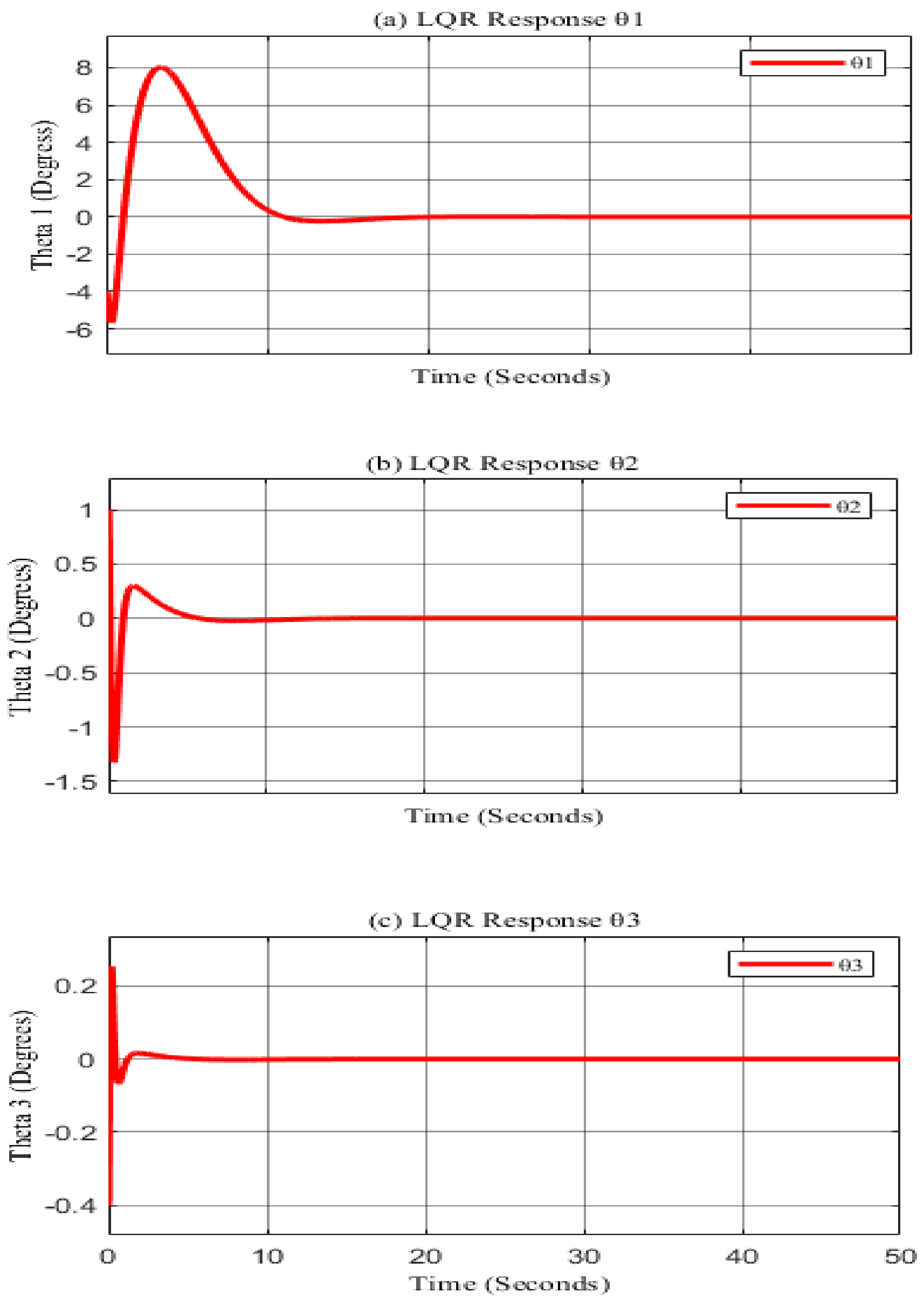


Figure 4.2 (a) The LQR system response for 1st link of Robogymnast; (b) The LQR system response for 2nd link of Robogymnast; (c) The LQR system response for 3rd link of Robogymnast

Table 4.1 LQR outcomes

| Symbol | Controller | $O_{sh}$ (p.u) | $U_{sh}$ (p.u) | $T_r$ (s) | $T_s$ (s) | ITAE |
|---|---|---|---|---|---|---|
| $\theta_1$ | LQR | 8.02 | -5.69 | 0.3557 | 15.5196 | 159.7 |
| $\theta_2$ | LQR | 1.03 | -1.32 | 0.0758 | 4.9142 | 0.322 |
| $\theta_3$ | LQR | 0.25 | -0.41 | 0.0509 | 3.3310 | 0.022 |

Where $O_{sh}$ is overshoot, $U_{sh}$ undershoot, $T_s$ is settling time and $T_r$ is the rising time, according to the results above which indicates the LQR response, the selected controller shows that the system is stable in all situations.

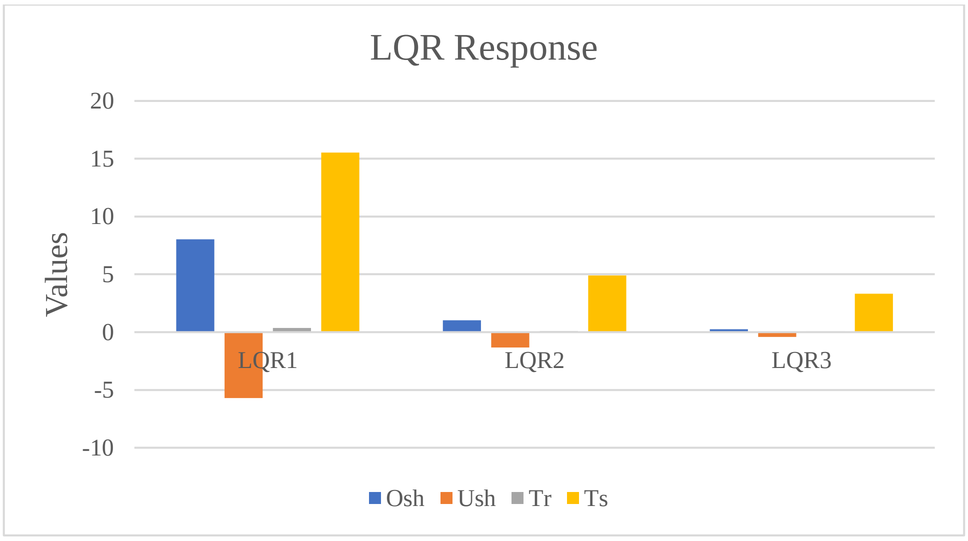


Figure 4.3 LQR Response

From the Table 4.1, the first link $\theta_1$ the overshoot ($O_{sh}$) is about 8.02 *p.u* On the other hand, the undershoot, the undershoot for the first link is -5.69 *p.u*, however, the time response of the upper link is 0.3557 second and 15.5196 second for rising and settling time respectively, lastly, the value of ITAE is 159.7. Secondly, the middle link $\theta 2$ outcomes is slightly different from $\theta 1$, where the overshoot ($O_{sh}$) is about 8.02 *pu,* and the overshoot ($O_{sh}$) is about -1.32 *p.u.* Furthermore, the time response is 0.0758 and 4.91 seconds respectively, then the error value of ITAE is 0.322. A third point is that the values of $\theta 3$ response, where link 3 overshoot ($O_{sh}$) is 0.25 *p.u* as well as undershoot ($U_{sh}$) of the lower link is -0.41 *p.u*, then the rising time ($T_r$) is 0.0509 second and the settling time ($T_s$) is 3.331 second, in addition, ITAE value is 0.022.

### 4.2.2 FLQR control

Development of a fuzzy model following the Mamdani method aimed to allow alteration to close-loop controller feedback gain [131]. E and EC are the error and error change, respectively, serving as input control signals, while U as an output variable were transformed to form language-based variables: NB=negative big; NM=negative medium; NS=negative small; Z=zero; PS=positive small; PM=positive medium; and PB= positive big. Graphic inference for inputs and outputs is carried out using triangular membership functions. In Figures 4.4 (a, b) error and change of error as input, while Figure 4.5 provides ranges for the output variable. Table 4.2 provides details of fuzzy rules used for the Robogymnast controller. Figure 4.7 gives an example of how Robogymnast is implemented in Simulink with an FLQR controller.

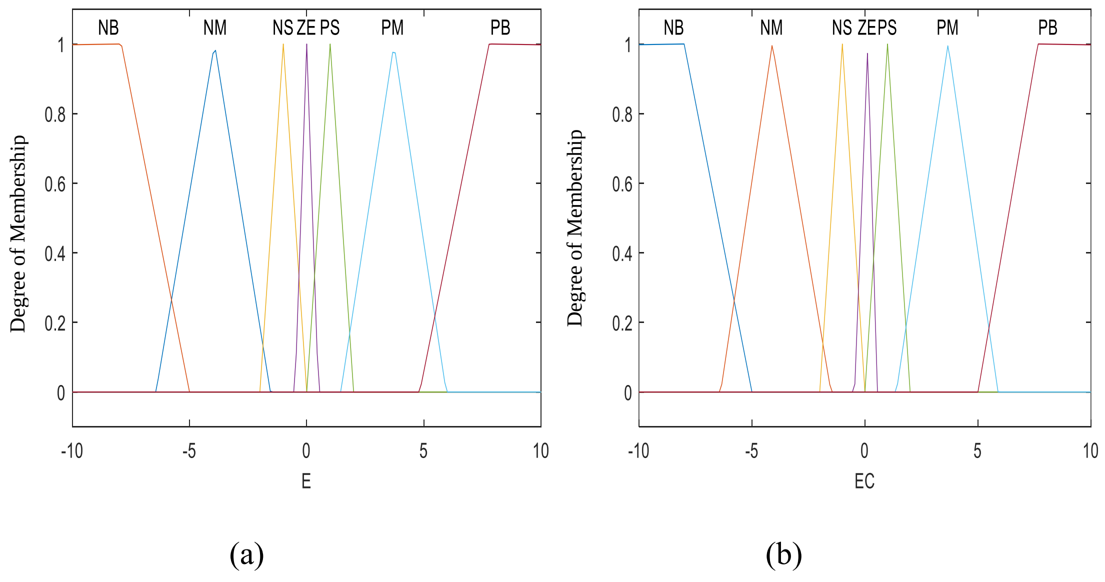


Figure 4.4 (a) Fuzzy input Membership error (E) and (b) error change (EC)

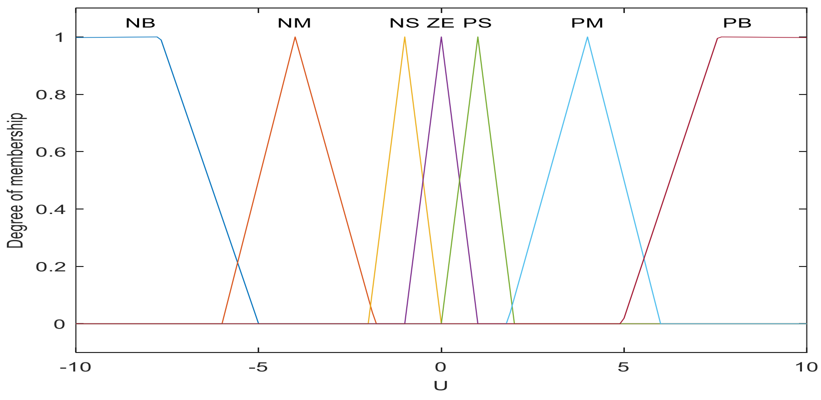


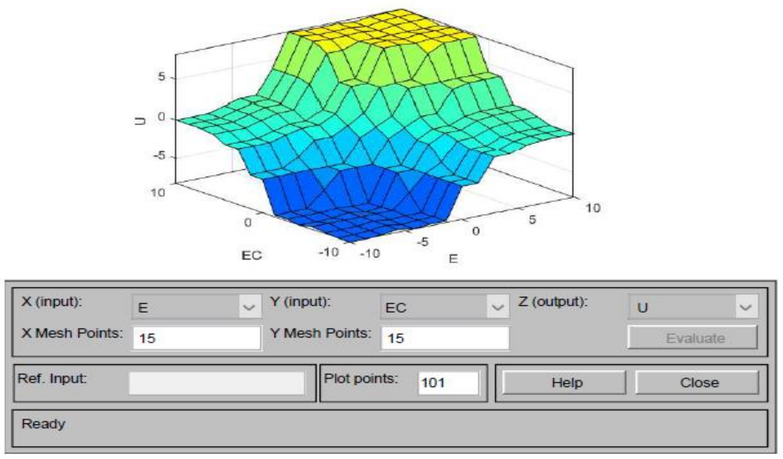
Figure 4.5 Membership function of Fuzzy output variable (U)

The fuzzy controller, which has a rule base comprising 49 fuzzy rules, establishes the connection between the input variables, and the output variable. The determination of these rules is guided by the consideration of practical limitations and the desired performance of the system.

Table 4.2 Rules of FL controller

| **Error** | **Change in Error** | | | | | | |
|---|---|---|---|---|---|---|---|
| | NB | NM | NS | Z | PS | PM | PB |
| NB | NB | NB | NB | NM | NM | NS | Z |
| NM | NB | NB | NB | NM | NS | Z | PS |
| NS | NB | NM | NM | NS | Z | PS | PM |
| Z | NM | NM | NS | Z | PS | PM | PB |
| PS | NM | NS | Z | PS | PM | PM | PB |
| PM | NS | Z | PS | PM | PM | PB | PB |
| PB | Z | PS | PM | PM | PB | PB | PB |

Fuzzy surface:



X (input): E
Y (input): EC
Z (output): U
X Mesh Points: 15
Y Mesh Points: 15
Ref. Input:
Plot points: 101
Ready

Figure 4.6 Fuzzy Surface

This controller is a combination of the optimal control approach of (LQR) and fuzzy control method [96]. Fuzzy logic controllers are rule-based systems [132]. The essential part of the FLC system is a set of Fuzzy Control Rules (FCRs) related by means of a fuzzy implication and the compositional rule of inference [97] . Figure 4.6 shows the surface of the fuzzy controller. In this section, the design procedures of the fuzzy logic controller are presented. The basic structure of a fuzzy logic controller (FLC) is shown in Figure 4.7, in which an FLC is used as a supplementary tool to enhance the existing control system in case of a change in conditions. Fuzzy logic is frequently used in manufacturing applications and their systems, also FLC is a method to enhance technology more intelligent [98].

The Simulink environment in MATLAB software was applied to find out the response of the system and then to design the proposed controllers. The step response of the closed-loop of the Robogymnast system with a controller has been carried out to evaluate the performance of the system in closed-loop mode. Modelling and simulation of the linear and non-linear LQR and FLQR controllers were conducted via MATLAB Simulink in order to stabilize the Robogymnast. The output response findings show that rise time, overshoot, and other factors were improved using FLQR, based on stepped system responses as illustrated in Figure 4.8 (a-c).

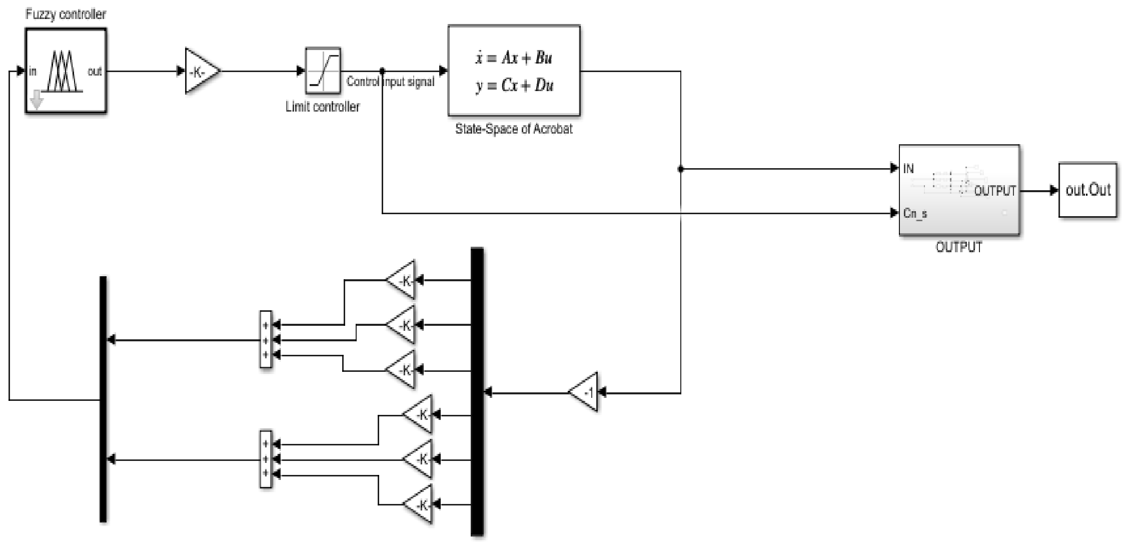


Figure 4.7 FLQR Simulink

MATLAB Simulink was used as a simulation environment for determining system responses, with the findings being used in designing the control systems put forward here. A performance evaluation of closed-loop systems operation was performed, related to the step response with the controller. Both the linear LQR and non-linear FLQR controller were modelled and their use in stabilising the robotic system was simulated in MATLAB Simulink [98]. Outcomes for output response demonstrated enhanced overshoot and rise time as well as other benefit when the FLQR controller was applied, considering the system’s step response (see Figure 4.8 a-c).

FLQR Results:

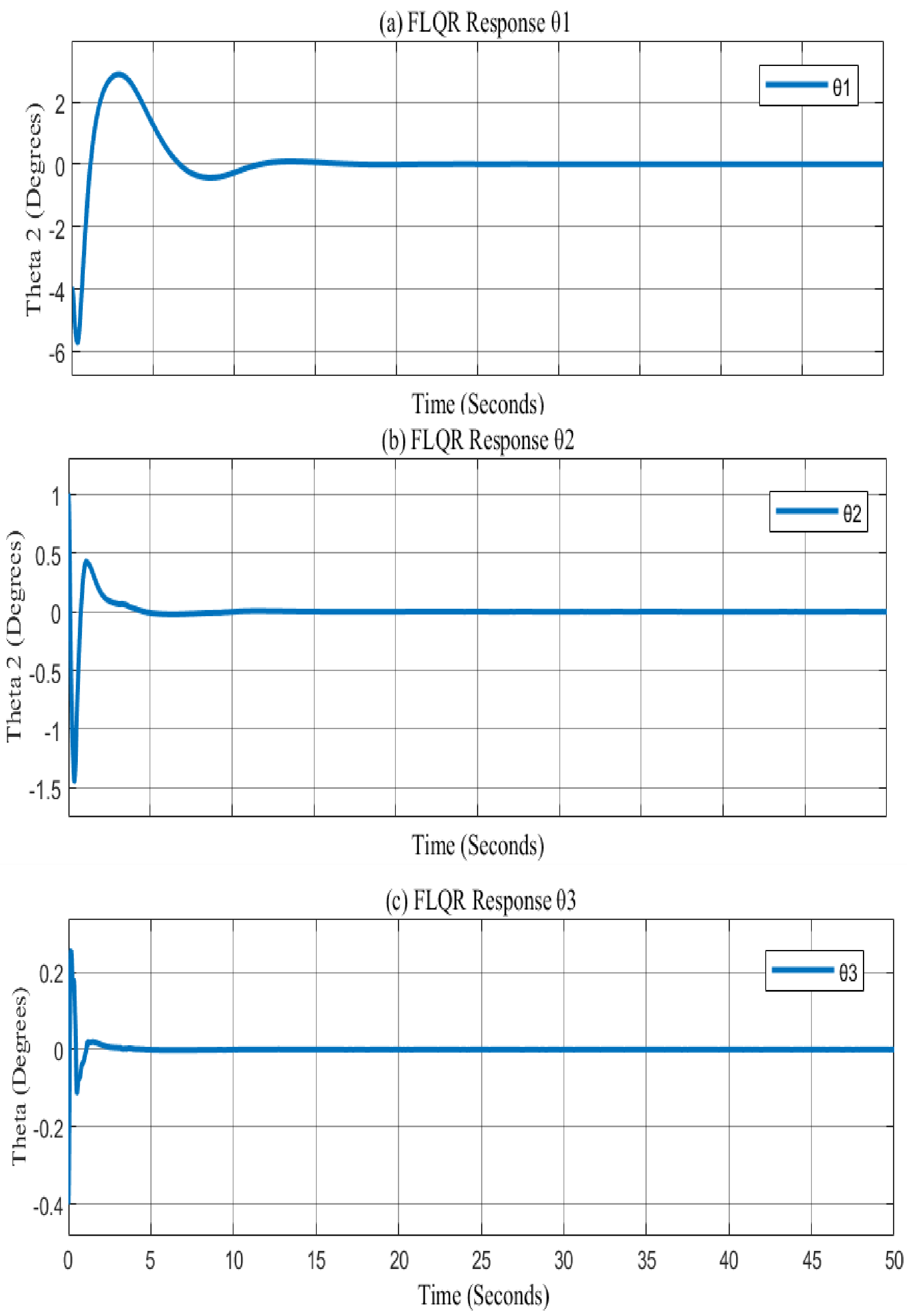


Figure 4.8 (a) FLQR system response for 1st link of Robogymnast; (b) The FLQR system response for 2nd link of Robogymnast; (c) The FLQR system response for 3rd link of Robogymnast

Table 4.3 FLQR outcome

| **Symbol** | **Controller** | $O_{sh}$ **(*p.u*)** | $U_{sh}$ **(*p.u*)** | $T_r$ **(s)** | $T_s$ **(s)** | **ITAE** |
|---|---|---|---|---|---|---|
| $\theta_1$ | FLQR | 2.88 | -5.71 | 0.4074 | 11.1823 | 21.29 |
| $\theta_2$ | FLQR | 0.42 | -1.44 | 0.0730 | 4.1694 | 0.313 |
| $\theta_3$ | FLQR | 0.25 | -0.40 | 0.0523 | 2.4428 | 0.021 |

This part has reported modelling and simulations for the application of FLQR controllers to stabilise the three-link Robogymnast robot carried out in MATLAB/SIMULINK. For the study, this type of controller was developed and then will be investigated and compared to evaluate their effectiveness in controlling the system. The IPS was first mathematically modelled, and then a model was developed for simulating a robotic control drive involving the fuzzy rules created for the system, fuzzification and defuzzification. The main parameters of the system were determined, with calculations of rising and settling times as well as overshoot, and an evaluation dynamic system performance was carried out.

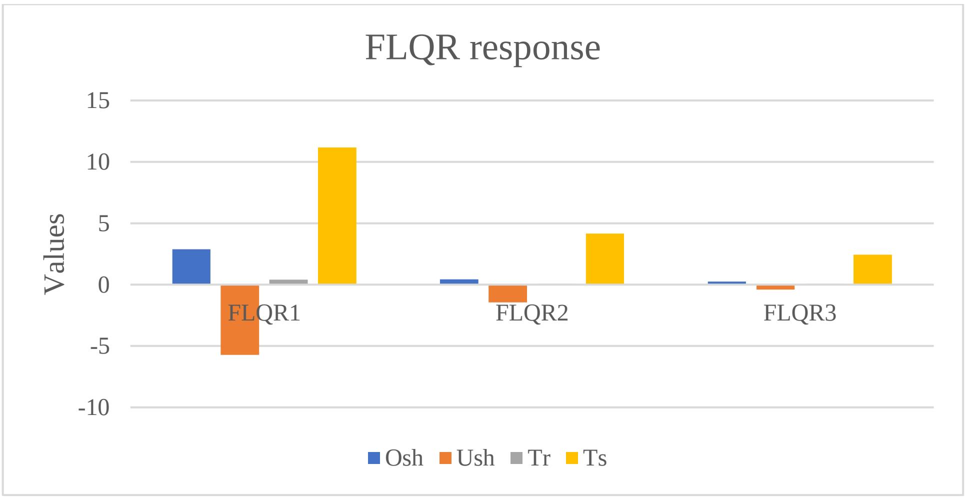


Figure 4.9 FLQR step response comparison

As shown in Figure 4.8 (a - c), and 4.9 the response of FLQR controller of triple link is, for the first link the overshoot is 2.88 *p.u*, the undershoot is -5.71 *p.u*. However, the time response is 0.4074 and 11.18 seconds. On the other hand, there is no huge different between the middle and lower link except in undershoot where theta 2 is -1.44 *p.u*, and $\theta_3$ is at -0.40 *p.u*.

## 4.2.3 Comparison LQR-FLQR controller

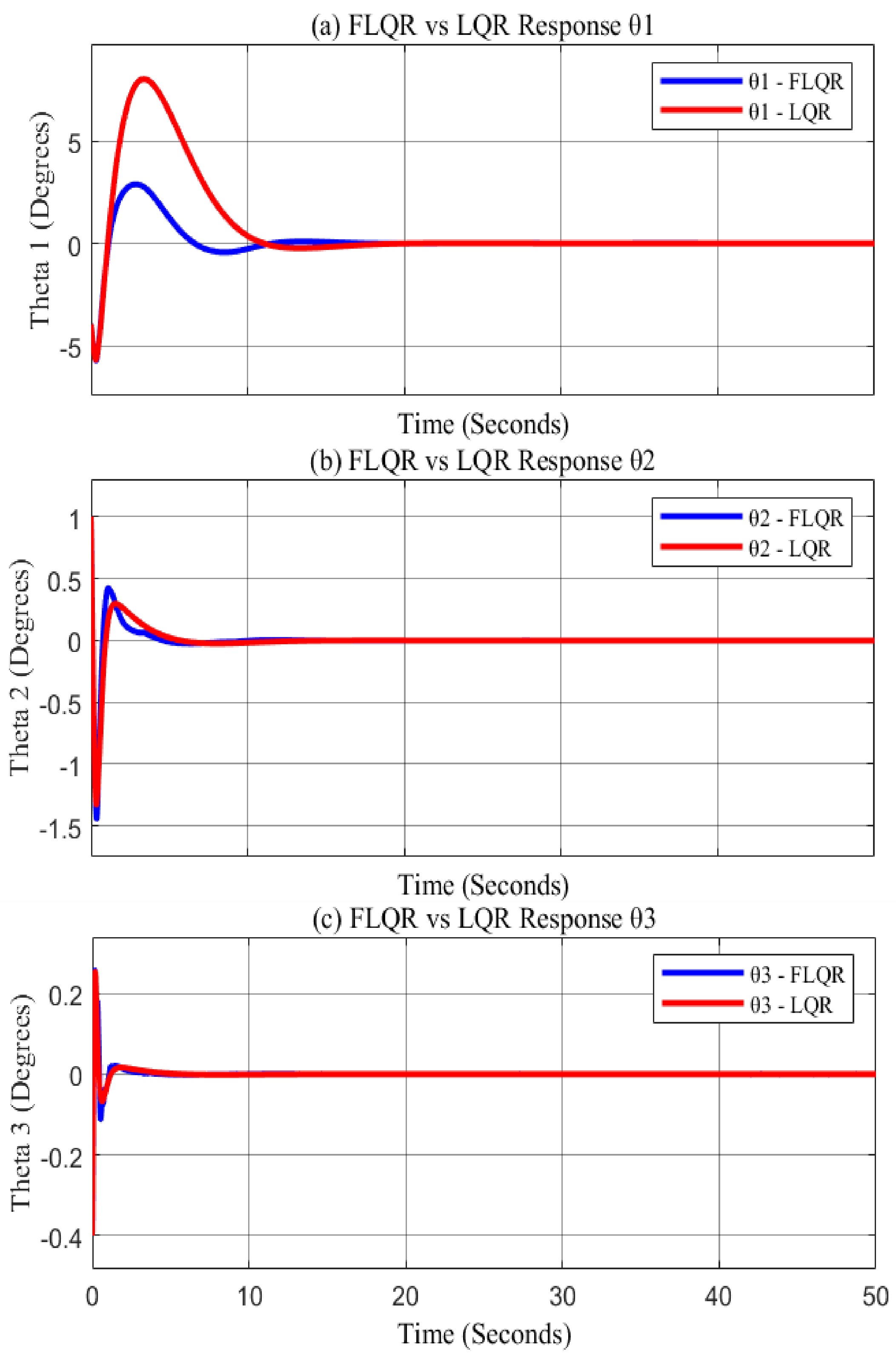


Figure 4.10 (a) The system response for 1st link of Robogymnast (T1); (b) The system response for 2nd link of Robogymnast (T2); (c) The system response for 3rd link of Robogymnast in (T3)

Table 4.4 LQR vs. FLQR performance

| Symbol | Controller | $O_{sh}$ (p.u) | $U_{sh}$ (p.u) | $T_r$ (s) | $T_s$ (s) |
|---|---|---|---|---|---|
| $\theta_1$ | LQR | 8.02 | -5.69 | 0.3557 | 15.5196 |
| | Fuzzy-LQR | 2.88 | -5.71 | 0.4074 | 11.1823 |
| $\theta_2$ | LQR | 1.03 | -1.32 | 0.0758 | 4.9142 |
| | Fuzzy-LQR | 0.42 | -1.44 | 0.30 | 4.1694 |
| $\theta_3$ | LQR | 0.25 | -0.41 | 0.0509 | 3.3310 |
| | Fuzzy-LQR | 0.25 | -0.40 | 0.0523 | 2.4428 |

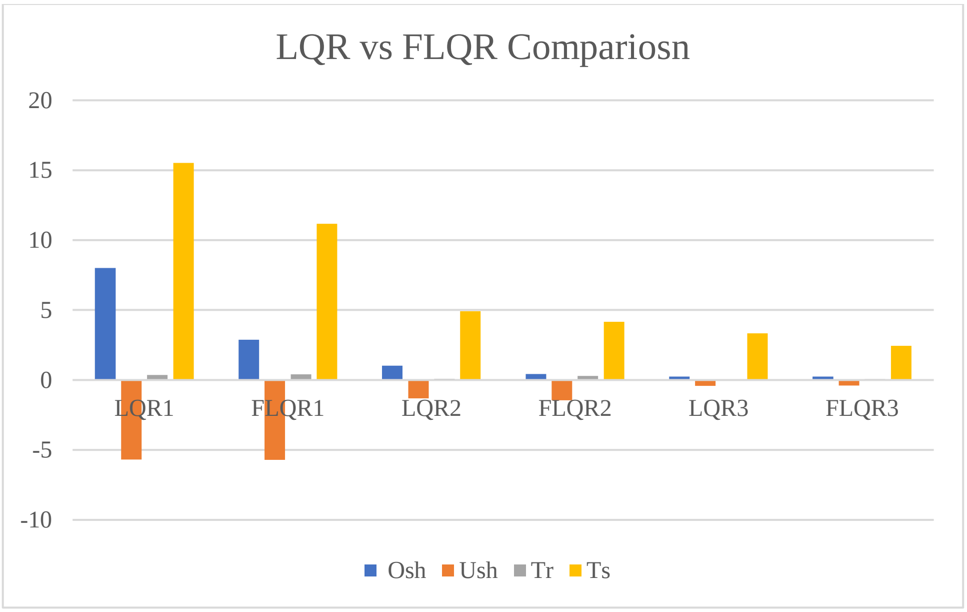


Figure 4.11 LQR and FLQR Comparison

Figure 4.10 (a-c) and Table 4.4 provides the comparison information about the performance of different LQR and FLQR controllers applied to three link robotic system, each link shows the performance separately in order to over and undershoot, rising and settling time. In $\theta_1$ it is clear that LQR is higher in overshoot and settling time compared to FLQR**.** For $\theta_2$ the FLQR controller has a smaller overshoot and similar settling time compared to the LQR controller. Lastly $\theta_3$ both controllers have similar performance, with very small overshoot, undershoot, rising time, and a settling time of around 1 second different. Also, Figure 4.11 shows the different between both controllers as mentioned above.

### 4.2.4 Robustness control analysis

In order to determine whether the proposed controller is robust, this subsection analyses parametric uncertainty within the three-link system and the impacts of this on stabilising the system, in a multi-scenario approach. Various possible parametric system conditions are explored, and these results are listed in next subsection.

Primarily, the testbed parameters were each altered in isolation, before altering multiple parameters at one time, increasing, and then decreasing them by 25% and 50% from the baseline. To evaluate both the LQR and the fuzzy LQR controller in terms of robust performance, variations were made to the testbed system as shown below, where the original values of both controllers were tested and compared as shown in Figure 4.10 a–c, and Table 4.4. also, Figure 4. 11 compared the controller in the original value where the Robogymnast parameters are constant.

Then plus 25% of the original parameters were implemented in Subsection 4.2.4.1 for both LQR and FLQR which are signified in Figure 4.12 a - c and the compared outcomes are displayed in Figure 4.13 and Table 4.5 to show the differences in the results obtained from the implemented parameters in comparison to the original parameters used in case 1.

A second scenario is by adding 50% more of the original parameters to verify the stability as seen in Figures 4.14 a - c and 4.15 and Table 4.6. On the other hand, -25% and -50% were implemented respectfully to the robotic system to verify the response of the system as seen in Subsections (4.2.4.3) and (4.2.4.4) in which case 3 is compared in Table 4.7. Moreover, Figures 4.16 a–c and 4.17 insulate the outcomes of −25% of the system. In the last scenario, −50% less of the original value was applied to the initial values as it can be seen in Table 4.8, and the difference is presented in Figures 4.18 a–c and 4.19.

Finally, Subsection (4.2.4.5) demonstrates the integral time of absolute error (ITAE) for each case 1−5, then Table 4.9. Figure 4.20 demonstrates the comparison between LQR and FLQR controllers in all scenarios respectively. As a result, parametric uncertainty conditions were formed that might frequently be encountered operationally within the testbed. Optimal gains were achieved in all cases; there are multiple variables within the testbed system that can change as operations are running and increases or decreases in any of this impact how stable the system is, where the parameters change mathematically.

#### 4.2.4.1 Case 1 (+ 25%).

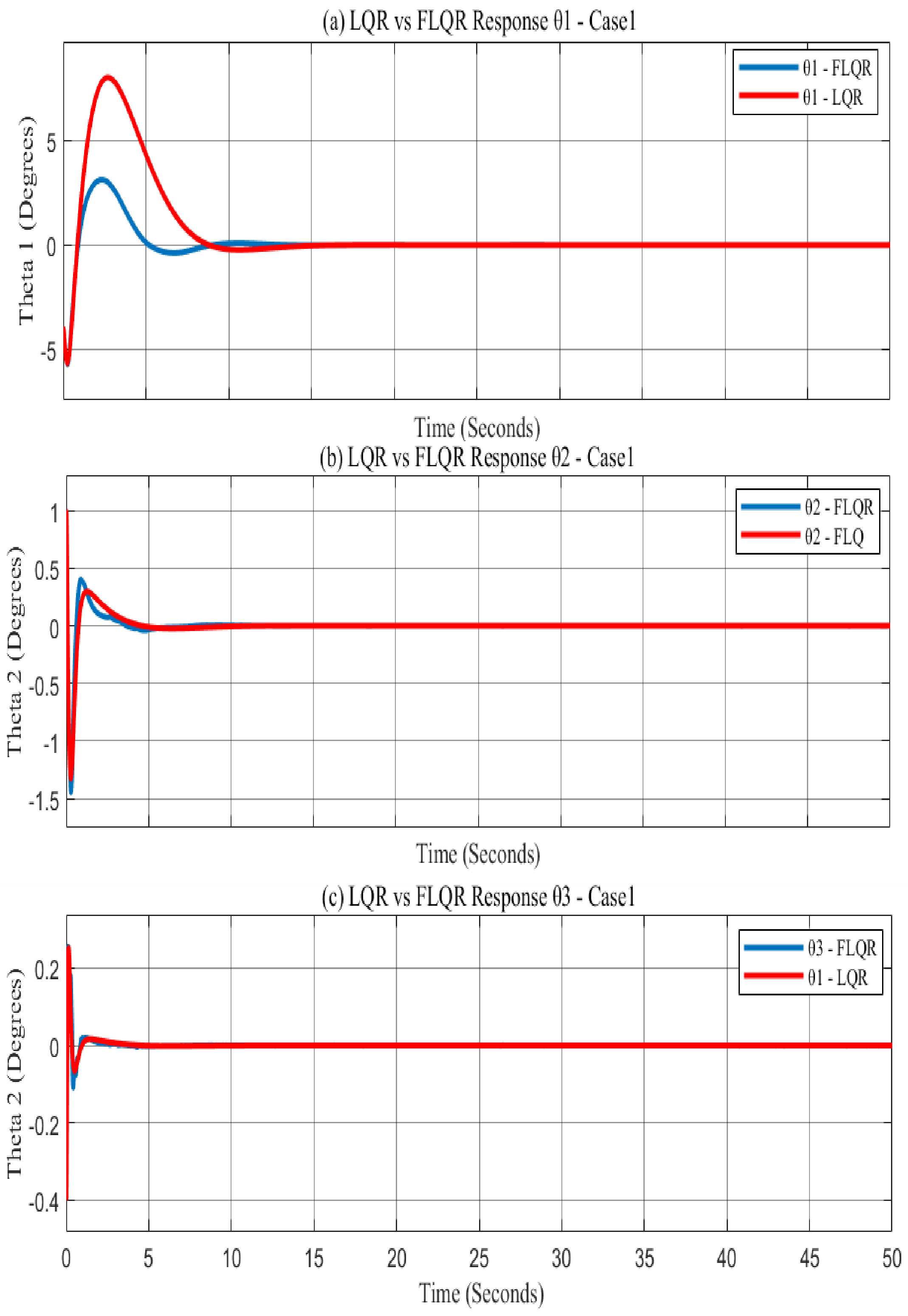


Figure 4.12 (a) The system response for 1st link of Robogymnast in Case1 (T1); (b) The system response for 2nd link of Robogymnast in Case 1. (T2); (c) The system response for 3rd link of Robogymnast in Case1 (T3)

Table 4.5 Comparison performance of LQR and FLQR controllers in Case 1

| **Symbol** | **Controller** | $O_{sh}$ **(p.u)** | $U_{sh}$ **(p.u)** | $T_r$ **(s)** | $T_s$ **(s)** |
|---|---|---|---|---|---|
| $\theta_1$ | LQR | 8.026 | -5.621 | 0.2844 | 12.4151 |
| | Fuzzy-LQR | 3.122 | -5.716 | 0.3249 | 8.9536 |
| $\theta_2$ | LQR | 1 | -1.326 | 0.0605 | 3.9305 |
| | Fuzzy-LQR | 1 | -1.445 | 0.84 | 3.3397 |
| $\theta_3$ | LQR | 0.2537 | -0.4 | 0.0406 | 2.6648 |
| | Fuzzy-LQR | 0.2562 | -0.4 | 0.0418 | 1.9637 |

In case 1. From Table 4.5 the performance of control system by adding (+25%) from the initial value, there is that big difference between this case and original one, where FLQR performs better than LQR in terms of overshoot, undershoot, and settling time. However, LQR has a shorter rising time compared to FLQR for all three input symbols. This suggests that the choice of the controller depends on the specific requirements of the control system and the desired trade-offs between different performance measures. Also, Figure 4.13 present the comparison between LQR and FLQR controllers in first scenario.

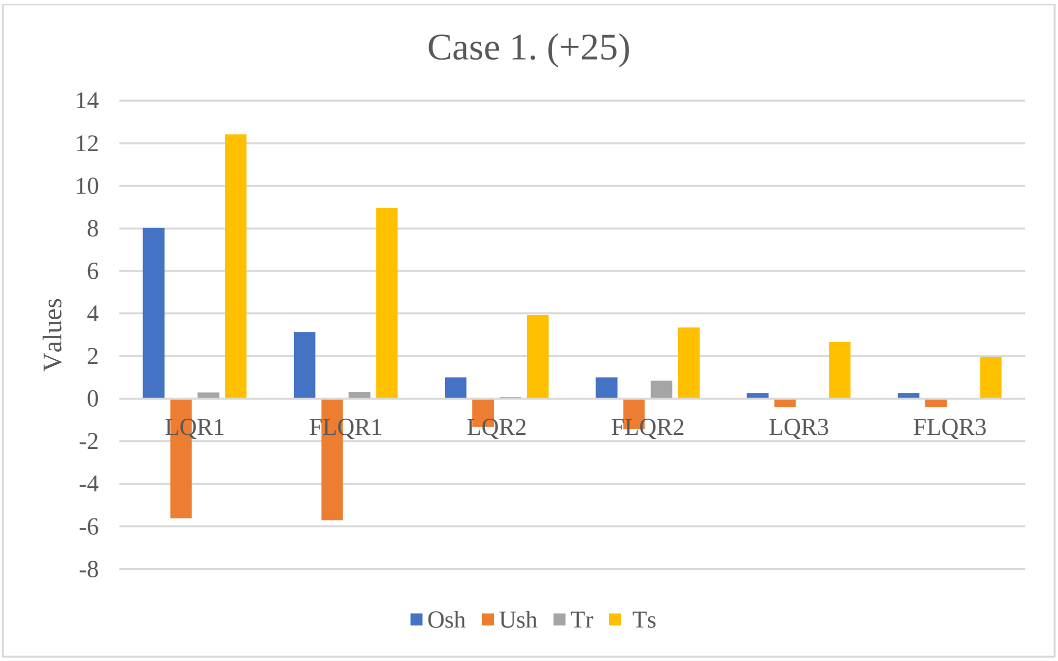


Figure 4.13 Case 1 comparison

### 4.2.4.2 Case 2 (+50%).

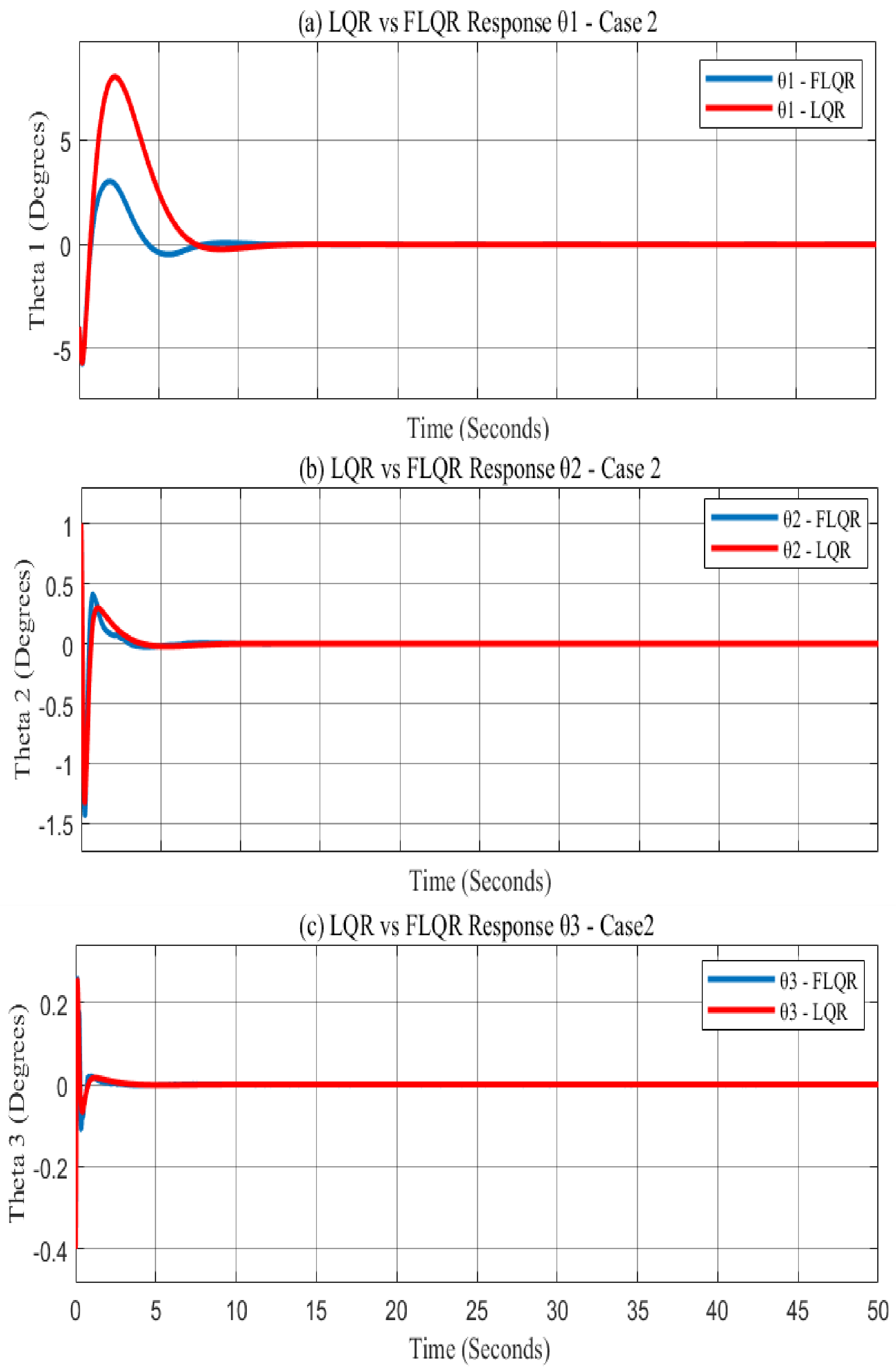


Figure 4.14 (a) The system response for 1st link of Robogymnast in Case2 (T1); (b) The system response for 2nd link of Robogymnast in Case2 (T2); (c) The system response for 3rd link of Robogymnast in Case2 (T3)

Table 4.6 Comparison performance of LQR and FLQR controllers in Case 2

| Symbol | Controller | $O_{sh}$ $(p.u)$ | $U_{sh}$ $(p.u)$ | $T_r$ (s) | $T_s$ (s) |
|---|---|---|---|---|---|
| $\theta_1$ | LQR | 8.025 | -5.679 | 0.2369 | 10.3460 |
| | Fuzzy-LQR | 3.013 | -5.711 | 0.2709 | 7.4641 |
| $\theta_2$ | LQR | 1 | -1.326 | 0.0504 | 3.2747 |
| | Fuzzy-LQR | 1 | -1.433 | 0.0486 | 2.7863 |
| $\theta_3$ | LQR | 0.2542 | -0.4 | 0.0337 | 2.2205 |
| | Fuzzy-LQR | 0.2582 | -0.4 | 0.048 | 1.6402 |

In the second scenario, by adding (+50%) Figure 4.15 and Table 4.16 presents the comparison of performance between LQR and FLQR again, which shows FLQR performs better than LQR in terms of overshoot, undershoot, and settling time. However, LQR has a shorter rising time compared to FLQR for all three input symbols, and no big different compared to original case.

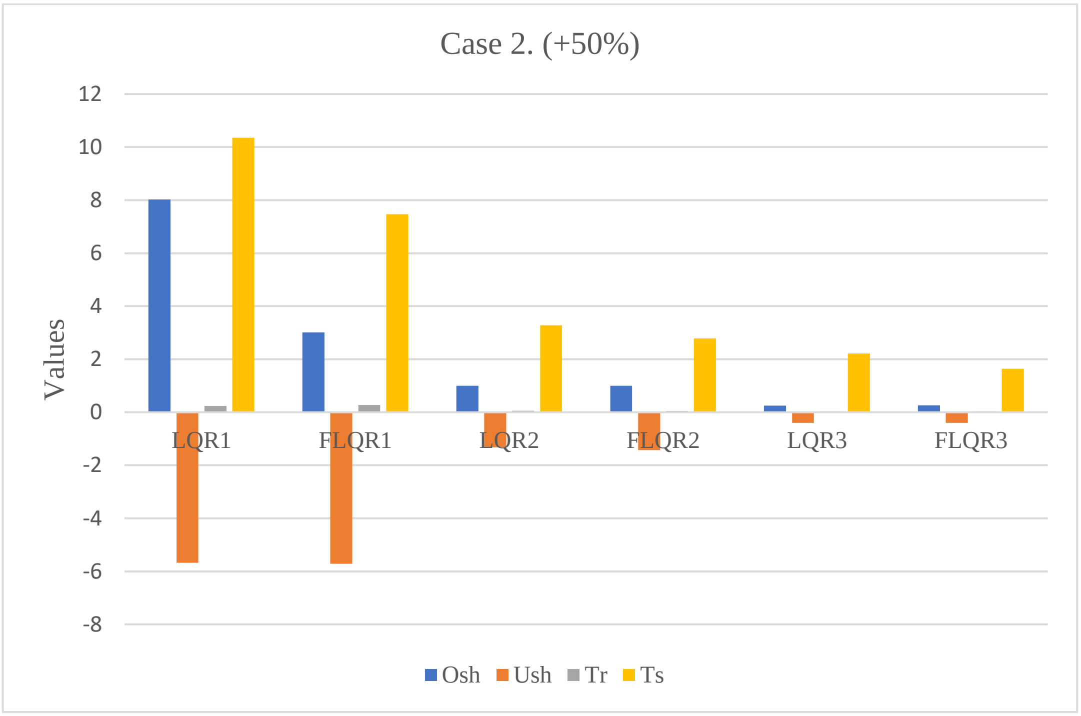


Figure 4.15 Case 2 Comparison

**4.2.4.3 Case 3 (- 25%).**

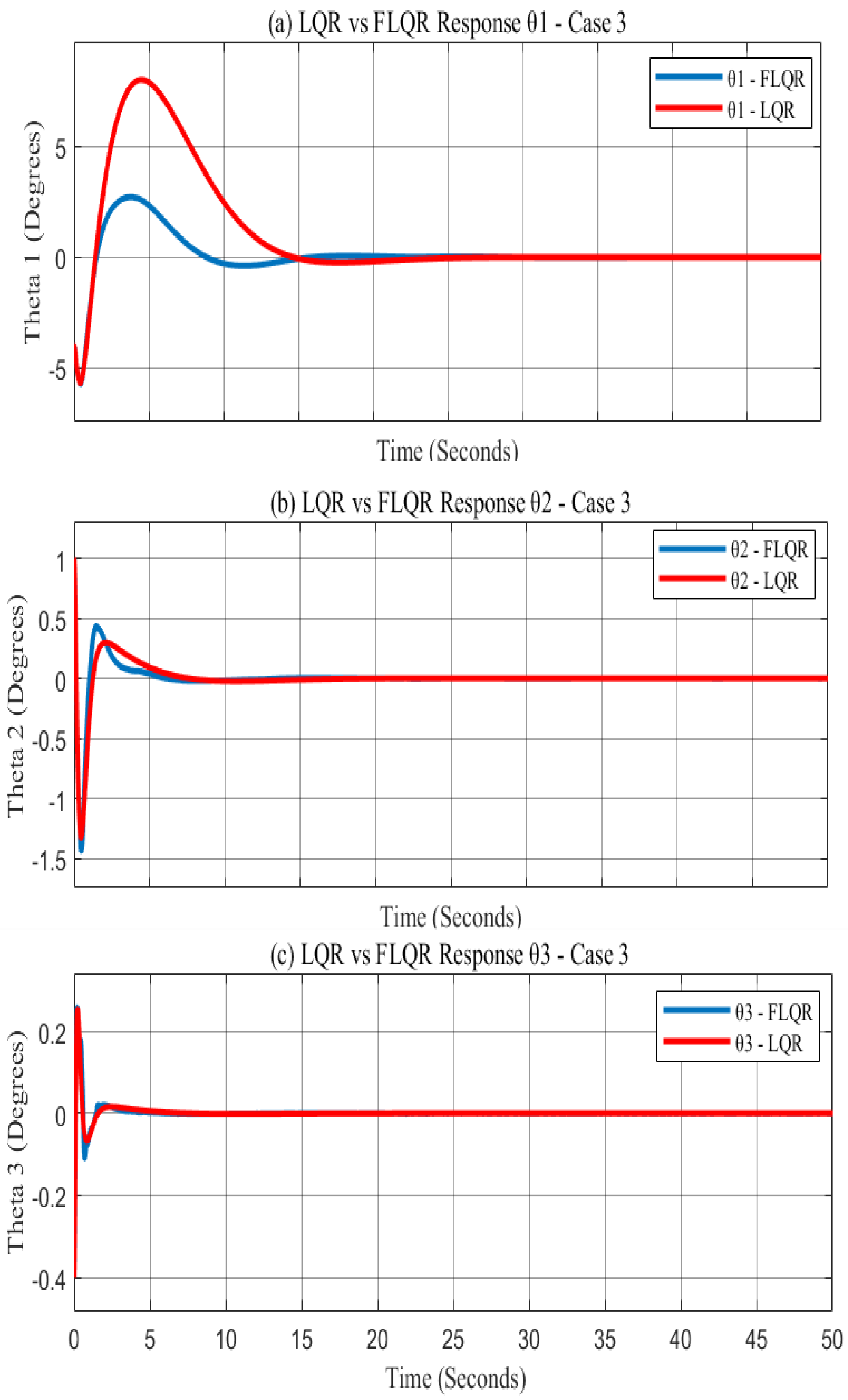


Figure 4.16 (a) The system response for 1st link of Robogymnast in Case3 (T1); (b) The system response for 2nd link of Robogymnast in Case3 (T2); (c) The system response for 3rd link of Robogymnast in Case3 (T3)

Table 4.7 Comparison performance of LQR and FLQR controllers in Case 3

| **Symbol** | **Controller** | $O_{sh}$ **(*p.u*)** | $U_{sh}$ **(*p.u*)** | $T_r$ **(s)** | $T_s$ **(s)** |
|---|---|---|---|---|---|
| $\theta_1$ | LQR | 8.025 | -5.679 | 0.4744 | 20.6922 |
| | Fuzzy-LQR | 2.742 | -5.712 | 0.5445 | 14.8923 |
| $\theta_2$ | LQR | 1 | -1.326 | 0.1012 | 6.5537 |
| | Fuzzy-LQR | 1 | -1.437 | 0.0975 | 5.5485 |
| $\theta_3$ | LQR | 0.2546 | -0.4 | 0.0681 | 4.4410 |
| | Fuzzy-LQR | 0.2584 | -0.4 | 0.0699 | 3.2399 |

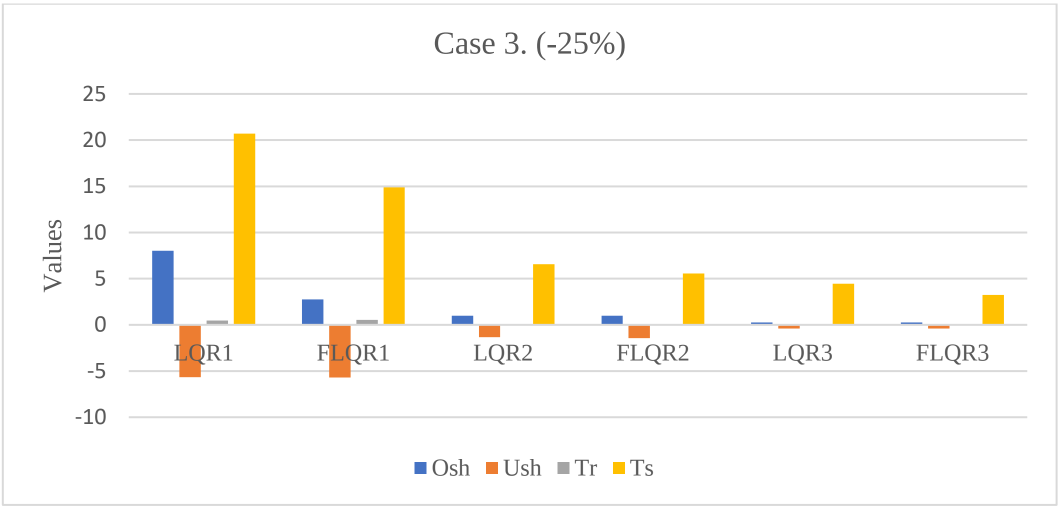


Figure 4.17 Case 3 Comparison

Case 3 shows a comparison between two different controllers (LQR and FLQR) by minimums (-25%) of the system with three different symbols ($\theta_1$, $\theta_2$ and $\theta_3$). For $\theta_1$, the LQR controller has an overshoot of 8.025 *p.u*, an undershoot of -5.679 *p.u*, a rising time of 0.4744 seconds, and a settling time of 20.6922 seconds. In comparison, the FLQR controller has a smaller overshoot of 2.742 *p.u*, a similar undershoot of -5.712 *p.u*, a slightly longer rising time of 0.5445 seconds, and a shorter settling time of 14.8923 seconds. For $\theta_2$, both controllers have an overshoot of 1 *p.u.* and an undershoot of around -1.32 *p.u*. The rising time for LQR is reported as 0.1012 seconds, while FLQR has a faster rising time of 0.0975 seconds. The settling time for LQR is 6.5537 seconds, which is longer than the settling time of 5.5485 seconds for FLQR. For $\theta_3$, both controllers have a small overshoot and a similar undershoot of around -0.4 *p.u*. The rising time for LQR is reported as 0.0681 seconds, while FLQR has a slightly longer rising time of 0.0699 seconds. The settling time for LQR is 4.4410 seconds, which is longer than the settling time of 3.2399 seconds for FLQR.

## 4.2.4.4 Case 4 (- 50%).

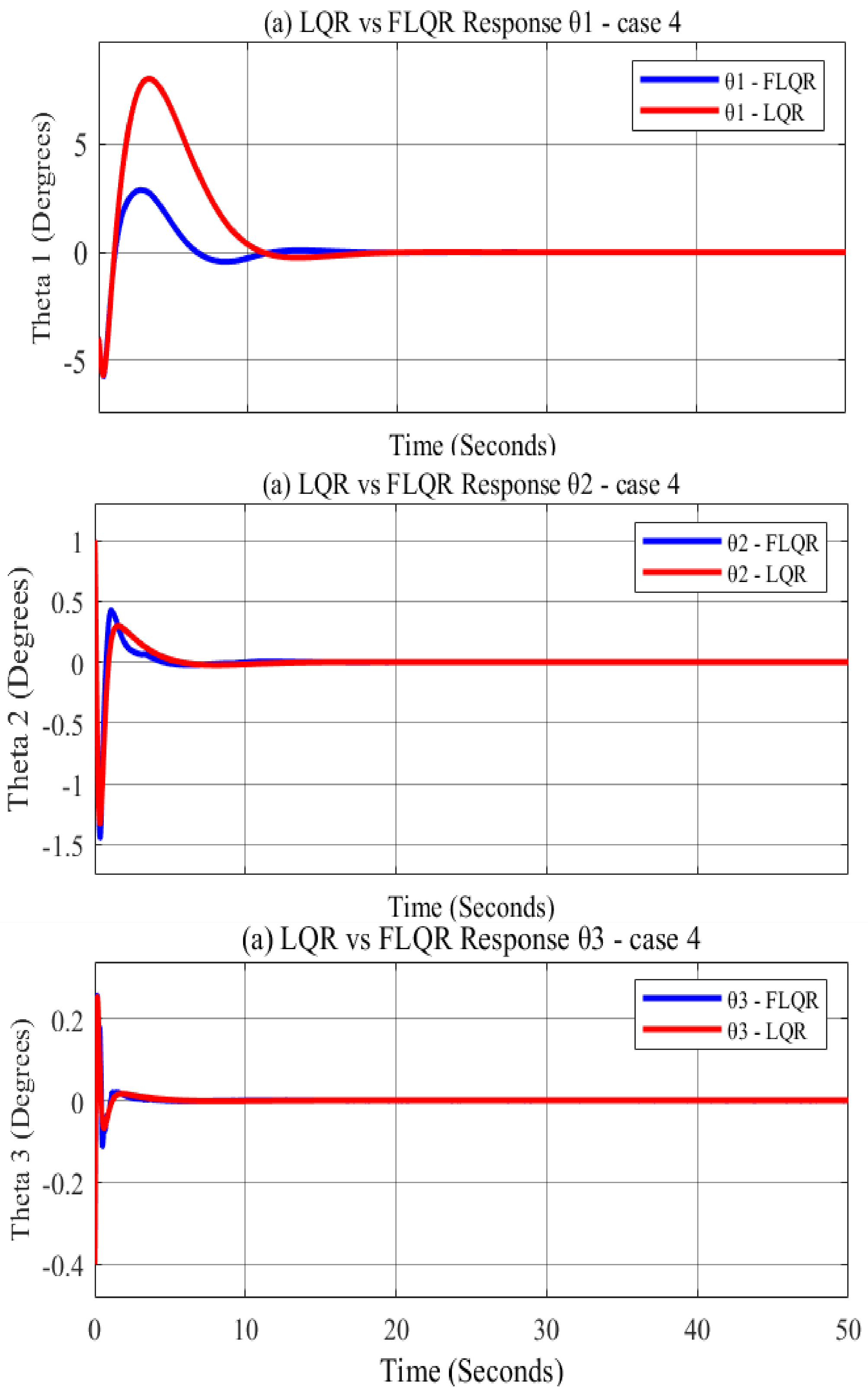


Figure 4.18 (a) The system response for 1st link of Robogymnast in Case 4 (T1); (b) The system response for 2nd link of Robogymnast in Case 4 (T2); (c) The system response for 3rd link of Robogymnast in Case 4 (T3)

Table 4.8 Comparison performance of LQR and FLQR controllers in Case 4.

| **Symbol** | **Controller** | $O_{sh}$ **(*p.u*)** | $U_{sh}$ **(*p.u*)** | $T_r$ **(s)** | $T_s$ **(s)** |
|---|---|---|---|---|---|
| $\theta_1$ | LQR | 8.025 | -5.679 | 0.711 | 31.0403 |
| | Fuzzy-LQR | 3.146 | -5.706 | 0.7952 | 21.3793 |
| $\theta_2$ | LQR | 1 | -1.3626 | 0.1520 | 9.8323 |
| | Fuzzy-LQR | 1 | -1.421 | 0.1483 | 7.8387 |
| $\theta_3$ | LQR | 0.2548 | -0.4 | 0.1024 | 6.6633 |
| | Fuzzy-LQR | 0.2604 | -0.4 | 0.1060 | 4.8405 |

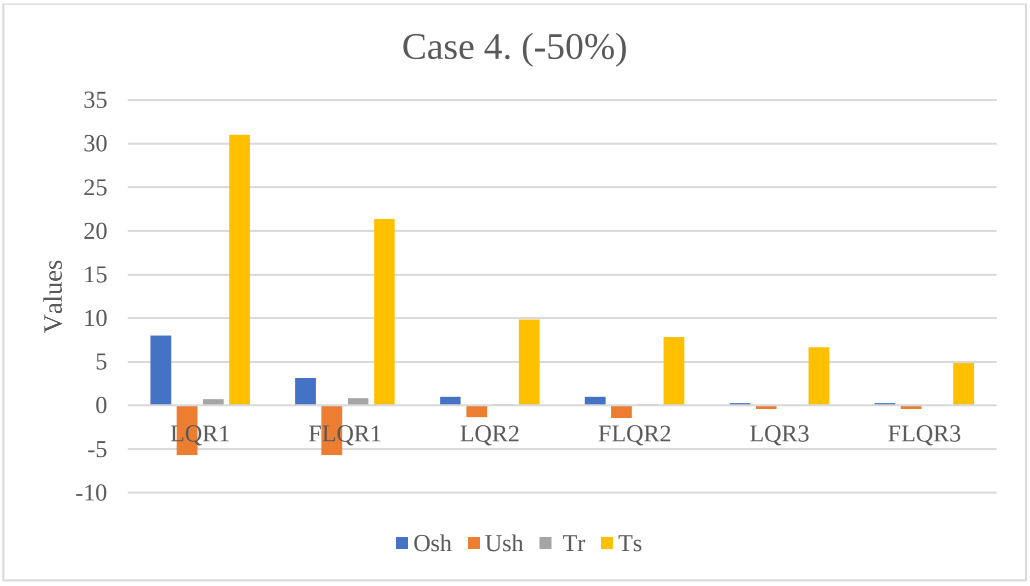


Figure 4.19 Case 4 comparison

The Figure 4.19 shows the performance of two different controllers, LQR and FLQR, for three different values of theta. The performance metrics measured are overshoot, undershoot, rising time, and settling time. Looking at the Table 4.8, it can see that for theta1, the FLQR controller has lower overshoot and rising time than the LQR controller. However, the LQR controller has a longer settling time than at the FLQR controller at 21.37 second. For $\theta_2$, both controllers have the same overshoot and settling time, however, the FLQR controller has a slightly longer rising time. For theta3, both controllers have similar performance, with the FLQR controller having a slightly higher overshoot and rising time, but a shorter settling time at 4.8405 second compared to 6.6633 second for LQR.

### 4.2.4.5 Comparison of ITAE:

ITAE refers to Integral Time Absolute Error as shown in Table 4.9 a comprehensive comparison between LQR and FLQR in all cases implemented. Where assuming (±25%) and (±50%) were added to the system to test their stability and robostance. As well as Figure 4.20 illustrates the comparison between the system outcomes for five different cases in terms of overshoot, undershoot, rising and settling time in both controllers examined.

Table 4.9 ITAE Values

| **Symbols** | **Controller** | **Original case 0** | **Case 1 (+25%)** | **Case 2 (+50%)** | **Case 3 (-25%)** | **Case 4 (-50%)** |
|---|---|---|---|---|---|---|
| $\boldsymbol{\theta_1}$ | LQR1 | 159.7 | 283.9 | 102.20 | 31.0403 | 70.98 |
| | FLQR1 | 21.29 | 38.16 | 26.72 | 87.19 | 8.971 |
| $\boldsymbol{\theta_2}$ | LQR2 | 0.322 | 0.5741 | 0.2136 | 1.29 | 0.148 |
| | FLQR2 | 0.313 | 0.5309 | 0.2066 | 1.257 | 0.143 |
| $\boldsymbol{\theta_3}$ | LQR3 | 0.022 | 0.0395 | 0.0138 | 0.088 | 0.009 |
| | FLQR3 | 0.021 | 0.0385 | 0.0142 | 0.086 | 0.009 |

Table 4.9 illustrates the performance of different controllers under different cases. The performance metric used is ITAE, which is a commonly used metric for evaluating the performance of control systems. It's also interesting to note that the performance of the controllers varies widely between the different cases. For example, the ITAE values for case 1 are much higher than for original case and case 3, suggesting that this case is more challenging for the controllers to handle. This highlights the importance of testing control systems under a variety of conditions in order to evaluate their overall performance.

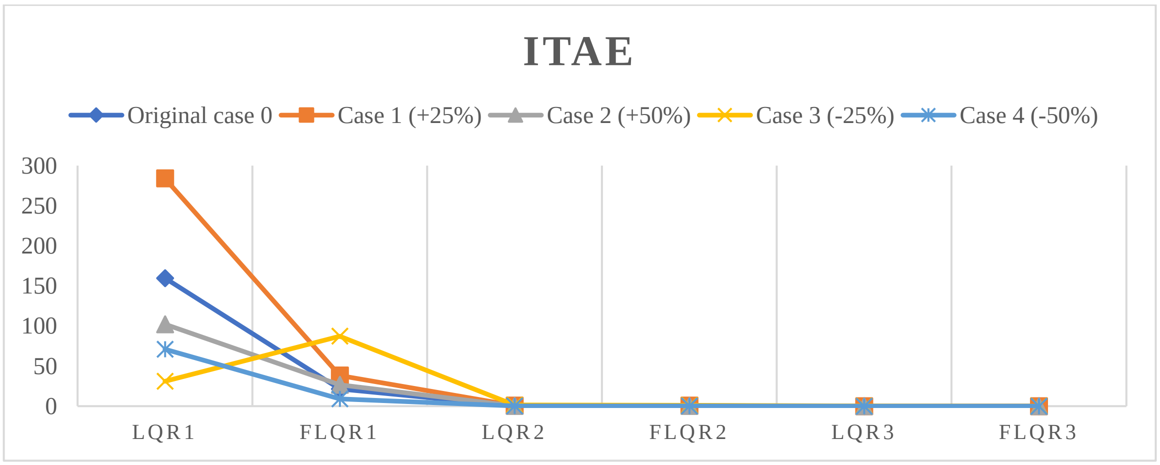


Figure 4.20 ITAE Comparison

## 4.2.5 Comparison Of LQR and FLQR

Table 4.10 Comparison of LQR and FLQR in all cases

| Case | Symbol | Controller | $O_{sh}$ $(p.u)$ | $U_{sh}$ $(p.u)$ | $T_r$ (s) | $T_s$ (s) |
|---|---|---|---|---|---|---|
| **Original value** | $\theta_1$ | LQR | 8.02 | −5.69 | 0.3557 | 15.5196 |
| | | Fuzzy LQR | 2.88 | −5.71 | 0.4074 | 11.1823 |
| | $\theta_2$ | LQR | 1.03 | −1.32 | 0.0758 | 4.9142 |
| | | Fuzzy LQR | 0.42 | −1.44 | 0.0730 | 4.1694 |
| | $\theta_3$ | LQR | 0.25 | −0.41 | 0.0509 | 3.3310 |
| | | Fuzzy LQR | 0.25 | −0.40 | 0.0523 | 2.4428 |
| **Case 1** | $\theta_1$ | LQR | 8.026 | −5.677 | 0.2844 | 12.4151 |
| | | Fuzzy LQR | 3.122 | −5.716 | 0.3249 | 8.9536 |
| | $\theta_2$ | LQR | 1 | −1.326 | 0.0605 | 3.9305 |
| | | Fuzzy LQR | 1 | −1.445 | 0.0584 | 3.3397 |
| | $\theta_3$ | LQR | 0.2537 | −0.4 | 0.0406 | 2.6648 |
| | | Fuzzy-LQR | 0.2562 | −0.4 | 0.0418 | 1.9637 |
| **Case 2** | $\theta_1$ | LQR | 8.025 | −5.679 | 0.2369 | 10.3460 |
| | | Fuzzy LQR | 3.013 | −5.711 | 0.2709 | 7.4641 |
| | $\theta_2$ | LQR | 1 | −1.326 | 0.0504 | 3.2747 |
| | | Fuzzy LQR | 1 | −1.433 | 0.0486 | 2.7863 |
| | $\theta_3$ | LQR | 0.2542 | −0.4 | 0.0337 | 2.2205 |
| | | Fuzzy LQR | 0.2582 | −0.4 | 0.0348 | 1.6402 |
| **Case 3** | $\theta_1$ | LQR | 8.025 | −5.679 | 0.4744 | 20.6922 |
| | | Fuzzy LQR | 2.742 | −5.712 | 0.5445 | 14.8923 |
| | $\theta_2$ | LQR | 1 | −1.326 | 0.1012 | 6.5537 |
| | | Fuzzy LQR | 1 | −1.437 | 0.0975 | 5.5485 |
| | $\theta_3$ | LQR | 0.2546 | −0.4 | 0.0681 | 4.4410 |
| | | Fuzzy LQR | 0.2584 | −0.4 | 0.0699 | 3.2399 |
| **Case 4** | $\theta_1$ | LQR | 8.025 | −5.679 | 0.711 | 31.0403 |
| | | Fuzzy LQR | 3.146 | −5.706 | 0.7952 | 21.3793 |
| | $\theta_2$ | LQR | 1 | −1.3626 | 0.1520 | 9.8323 |
| | | Fuzzy LQR | 1 | −1.421 | 0.1483 | 7.8387 |
| | $\theta_3$ | LQR | 0.2548 | −0.4 | 0.1024 | 6.6633 |
| | | Fuzzy LQR | 0.2604 | −0.4 | 0.1060 | 4.8405 |

Table 4.10 shows a full comparison between LQR and FLQR in all cases implemented where (±25%) and (±50%) were added to the system to test their stability and robostance.

On the other hand, Figure 4.21 illustrates the comparison between the system outcomes for 5 different cases in terms of Over and undershoot, rising and settling time in both controllers examined.

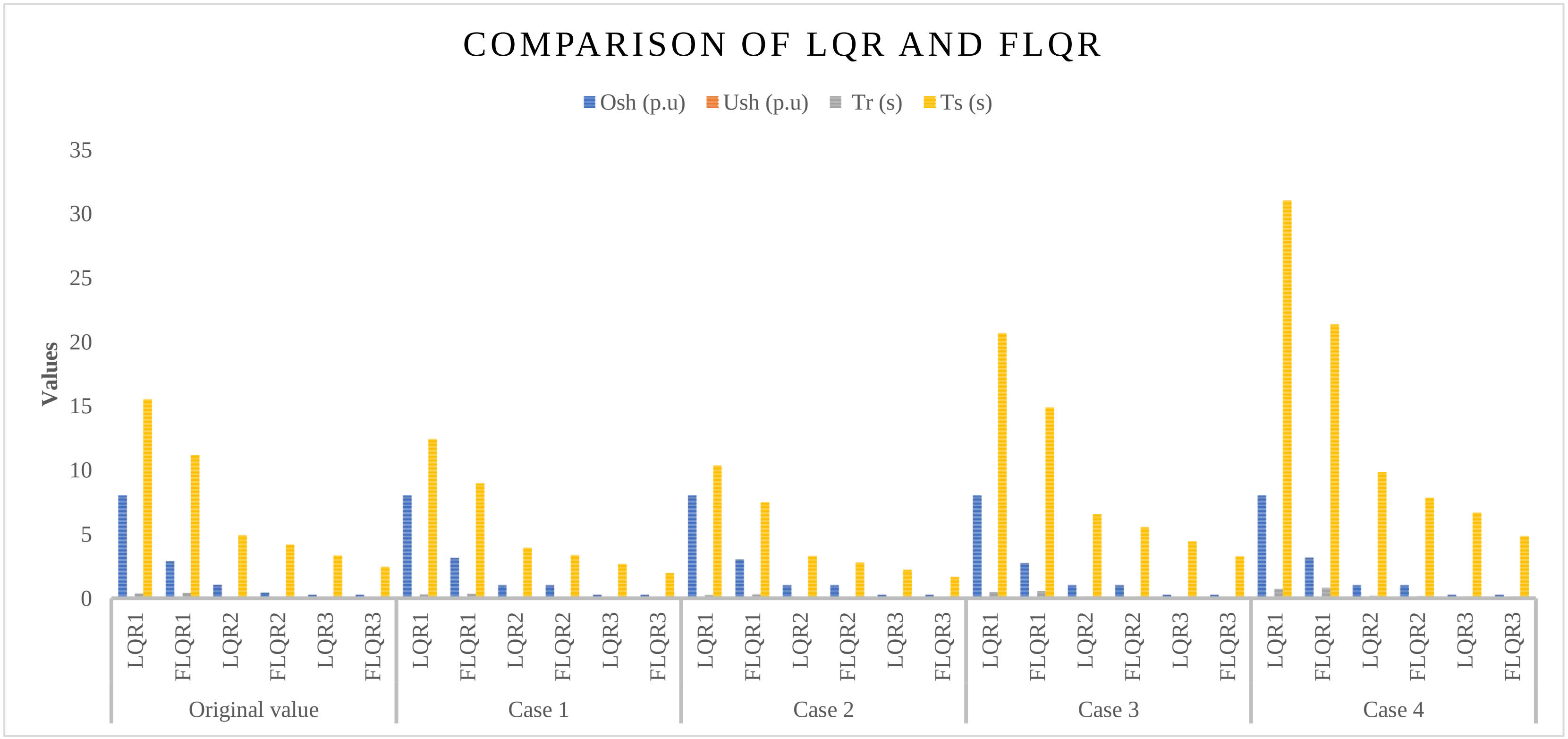


Figure 4.21 LQR and FLQR all cases comparison

## 4.3 Results and discussion

To conclude this work, it is important to declare that this work involved the investigation of the modulation of triple link robotics mechanism for the swing-up position. The proposed FLQR controller, as a combined LQR and fuzzy logic control method, performed well for the three-link Robogymnast robotics system, with the examination of the system's robustness showing that it outperformed the conventional controller across all variables, including time to settle and under- and over-shoot. A more detailed summary of the findings for robustness is provided here.

For the first scenario, the measured system response is shown in Figure 4.10 (a - c) and Table 4.4, giving baseline systems values with no alterations. Comparing the LQR with the FLQR controller, the latter shows decreased overshoot ($O_{sh}$) and undershoot ($U_{sh}$), with respective overshoot values of 8.02 *p.u* and 2.88 *p.u*. In addition, while the LQR controller's rise time was faster (c. 0.35 seconds), the FLQR showed a faster settling time (11.18 seconds).

For the case 1, in which the mathematical model values are increased by 25%, Table 4.5 shows the performance comparison between the two controllers examined for robustness. For the LQR controller, undershoot and overshoot did not differ significantly, but there was a difference of 0.07seconds in rise time, and the settling time dropped to 12.41 seconds from 15.51 seconds. In contrast, the findings for FLQR demonstrate an increase in overshoot of approximately 0.242 *p.u*, with no alteration in undershoot, and reductions in settling and rising times for the controllers. In addition, for LQR, there was a slight decrease in the second angle $\theta_2$ overshoot, while undershoot did not change significantly and the times were reduced for each parameter. For FLQR, decreases were found across each parameter. Finally, for $\theta_3$, no change was seen in under or overshoot for the LQR controller, although reductions occurred in the rising and settling times. For FLQR, time decreases were also found, while the values of the remaining parameters were unchanged from the initial scenario.

In the third scenario, values were increased by 50% over the system's baseline. While overshoot remained unchanged for the LQR, undershoot altered only slightly, from -5.69 *p.u* at baseline to -5.67 *p.u*, but both the rising and settling times were quicker, altering from 0.35 to 0.24 seconds for rising and a slight change of 5 seconds for the settling time. For link $\theta_2$ with the LQR controller, there was no change in under or overshoot, but a reduction in the two-time variables. Meanwhile, for the FLQR controller, times were also reduced, while no change was seen for under and overshoot. For the third link $\theta_3$, again, under and overshoot remained the same, while smaller values were found for the time variables.

For the fourth scenario case 3, system variables were reduced by 25%, as shown in Table 4.7. In the first link, for the LQR, a one-second increase in rising time was recorded for the LQR, and settling time was observed to change by roughly 5 seconds against the original case, from 15.51 seconds to 20.69 seconds. By contrast, the FLQR controller produced lower values for each time parameter, and only very small differences for the remaining variables. Moreover, for $\theta_2$, there was little change in under- or overshoot, with reductions in rising and settling times. Finally, for $\theta_3$, a small increase in rising time was seen compared with the original scenario, from 0.052 seconds to 0.0699 seconds. Settling time was altered by approximately 1s, with no changes in under or overshoot.

The final scenario case 4, reduced the system parameters by 50% and, again, stability and response were tested to indicate robustness. It is notable here that for the LQR controller, each of the links displayed small increases in rising and settling time differences, with major alterations in overshoot and undershoot. In contrast, FLQR performed markedly better here than the conventional controller. Settling times across each link were 21.37 seconds, 7.83 seconds, and 4.83 seconds, with higher rising times for each of the links. Under and overshoot were also greater across each link, all these outcomes presented in Figures 4.18(a - c) and 4.19, and the comparison in Table 4.8.

## 4.4 Summary

In summary, this chapter has described the implementation of the Robogymnast through a controller based on LQR fused with fuzzy logic and discussed systems improvements through the application of algorithms in MATLAB SIMULINK, comparing approaches for performance optimisation. This work sought to design and implement an FLQR controller, to understand how it performed in implementation with the Robogymnast. Simulation modelling was based on the designed controller and sought to analyse the major parameters of the system, identified as rising and settling times as well as overshoot. Also, ITAE used metric for evaluating the performance of control systems. An evaluation was also made of dynamic system performance based on the ITAE values.

To conclude, it is possible to declare that this work involved the investigation of the modulation of a triple-link robotics mechanism for the swing-up position, and the selected controller can be further extended to implement optimized algorithms for additional investigations.

# Chapter 5:

# Optimisation Techniques

## 5.1 Introduction

Optimisation is described as an attempt to solve problems which involve the maximisation or minimisation of either one objective or multiple-objective functions by identifying the best values from a domain with a range of satisfactory values, termed decision variables, while satisfying a set of restrictions [133]. The principle aim of optimisation is therefore to solve the target problem with the near best solution available. In order to optimise products or processes, it is necessary to first identify which conditions lead to the highest performance, using optimisation parameters given when the real model is formulated mathematically [134]. Optimisation has applications across a number of fields, including economics, design, engineering, and control. Optimisation however remains challenging, and in particular, certain engineering problems are highly complex [135].

Optimisation algorithms are a category of algorithm applied to identify the best solutions to problems through maximising or minimising objective functions, and frequently applied in areas including engineering, operations studies, and machine learning. For model parameter optimisation, and decision-making using data. Various classes of optimisation algorithm exist, such as genetic, gradient descent and simulated annealing algorithms, as well as particle swarm optimisation. The different algorithms have distinctive features and limitations and are selected based upon the particular problem approached and what outcomes are intended. Optimisation algorithms are critical for the solution of problems which present complexity, and for making systems more efficient. Such algorithms are often implemented for example: in machine learning, in order to train a model; to optimise engineering designs; and to optimise financial portfolios. For optimisation, an objective is used to represent the desired performance metric, and the algorithm updates solutions in an iterative manner in order to identify global minimums or maximums. In addition to the parameters and construction of the problem, computational issues including data volume and the needed convergence rate also influence the selection of an algorithm. Deterministic optimisation algorithms converge to an optimised solution predictably, and probabilistic algorithms apply randomisation in order to avoid localised optima. However, optimisations algorithms as a whole provide a vital function in the solution of problems with practical applications and in enhancing data-based decisions.

## 5.2 Optimisation method

In any optimisation problem, an objective which must be solved within certain boundaries referred to as constraints. Figure 5.1 illustrates the sequence for finding a solution to an optimisation problem. To formulate an optimisation algorithm, there is a need to identify the objective of the optimisation problem [136] [137]. The main purpose of the formulation procedure is to create the mathematical model of an optimal design problem, which is then solved using an optimization algorithm. The optimization algorithm requires the optimization problem to be presented in a specific format.

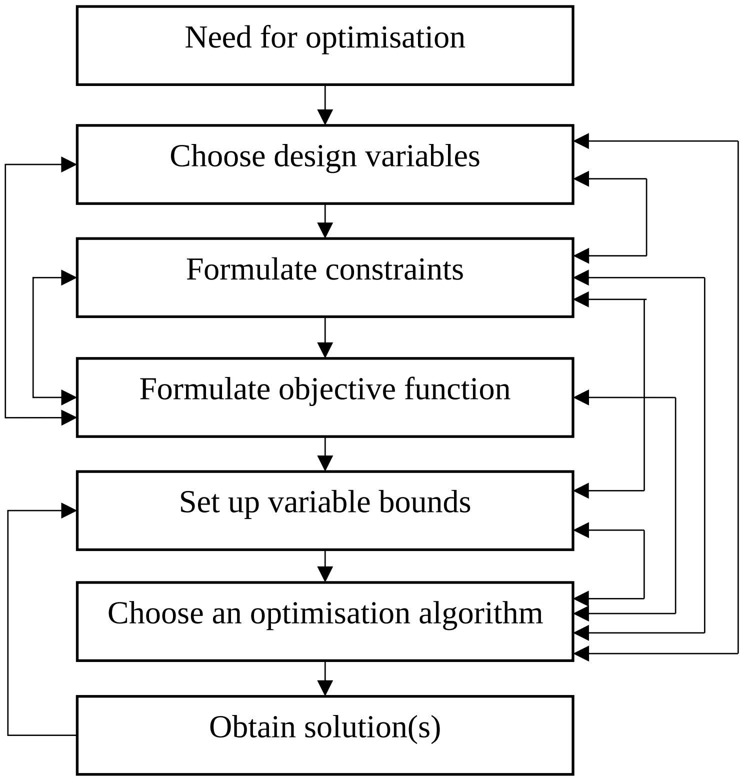


Figure 5.1 Optimal design procedure [138]

Decision variables: The decision variables sometimes referred to as design variables are the unknowns in the optimisation problem and will need to be determined by solving the problem. The speed and efficiency of the optimisation simulation depend on the number of decision variables to a large extent [139].

Constraints**:** With the design variables identified, the constraints or limitations to such a problem must be chosen. The constraints express the relationship between design variables and other parameters in order to meet the requirement of a physical phenomenon or limitation in resources [137]. Some examples of constraints are battery state of charge in EV and battery storage, voltage boundaries in distribution networks, thermal ratings of distribution network cables. The constraints may take the form of equality (=) or inequality (Less or equal to ≤, or greater than or equal to ≥). According to [138] most constraints in design problems are of the inequality type.

### 5.2.1 Objective function

There may be multiple objective functions in an optimisation problem which is referred to as multi-objective optimisation. The objective function may be minimised or maximised. With the aid of duality principal minimisation can be converted to maximum bounds on the decision variables. For Linear Quadratic Regulator (LQR) fuzzy logic controller, the objective function for minimizing the Integral of Time multiplied Absolute Error (ITAE) can be formulated in terms of the fuzzy sets and membership functions that are used to represent the input and output variables.

The objective function for a LQR fuzzy logic controller that minimizes the ITAE criterion can be expressed as follows:

$$\text{ITAE} = J = \int_0^{\text{Tsim}} |\Delta e\,| \times t \times dt \tag{5.1}$$

Where:

J is the objective function

$T_{sim}$ is the simulation period

Δe is the error

t is the time

The objective function represents the weighted sum of the absolute error between the system response and the desired response, weighted by the time error persists. The integral over time ensures that the objective function captures the error and its persistence over the entire time horizon of the system.

The membership function, which is a key component of the fuzzy logic controller, represents the degree to which a given input belongs to a particular fuzzy set. The membership function can be designed to incorporate knowledge about the system and the desired performance specifications and can be adjusted to optimize the performance of the controller.

To minimize the ITAE criterion using a fuzzy logic controller, the controller parameters, including the membership functions and the rule base, can be optimized using techniques such as genetic algorithms, particle swarm optimization, or gradient descent methods. The goal is to find the optimal set of parameters that minimizes the objective function and therefore achieves the desired control performance [140].

## 5.3 Teaching Learning Based Optimisation (TLBO) Algorithm

Rao [141] first put forward the TLBO algorithm in 2011, motivated by concepts of the classroom-based teaching learning process, and mimics the way in which teachers exert impacts on their students' outcomes. TLBO belongs to the family of swarm intelligence algorithms and is a population-based metaheuristic optimisation algorithm [135]. Various features of this algorithm have led to its widespread use across different problems and sectors of engineering, including the concept itself and the lack of a requirement for set parameters, as well as being simple and rapid to apply [142].

TLBO, as derived from the influence of teaching approaches on student outcomes, utilises two fundamental learning models: one, as mediated by a teacher (teacher stage); and two, by interacting with other students (learner phase). The population for the algorithm is based on a group of learners, with the various subjects which they study representing each design variable within the optimisation being targeted. Learners' outcomes represent "fitness" values for the optimisation problem. The teacher is defined as the optimal solution across the global population, while design variables comprise parameters related to the particular optimisation problem's objective function, with the optimal solution being that objective function's optimal value optimal value reference [1]. Figure 5.2 presents a flowchart for the TLBO algorithm.

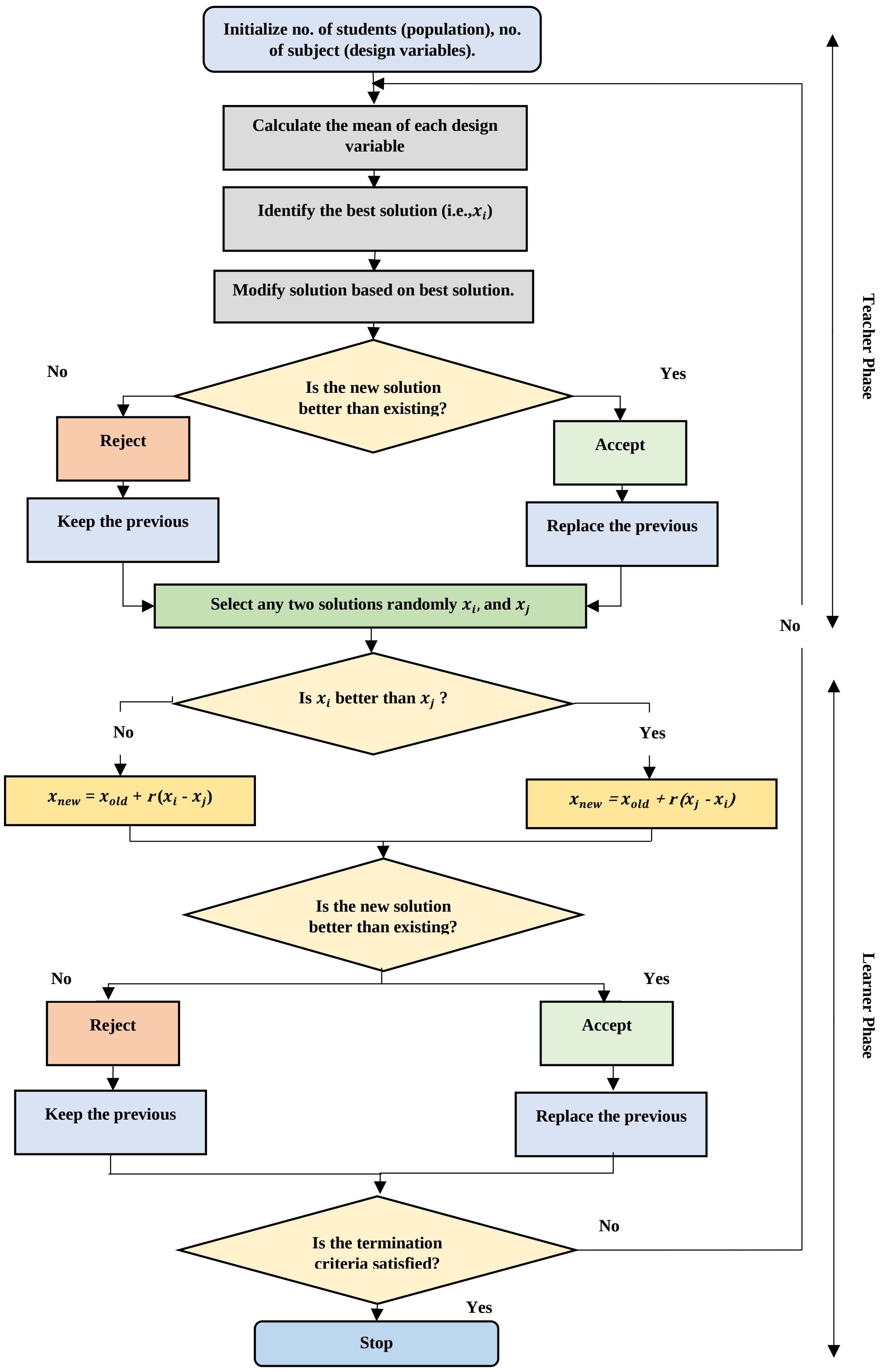

Initialize no. of students (population), no. of subject (design variables).
Calculate the mean of each design variable
Identify the best solution (i.e.,$x_i$)
Modify solution based on best solution.
Is the new solution better than existing?
No
Yes
Reject
Accept
Keep the previous
Replace the previous
Select any two solutions randomly $x_i$, and $x_j$
Is $x_i$ better than $x_j$ ?
No
Yes
$x_{new} = x_{old} + r(x_i - x_j)$
$x_{new} = x_{old} + r(x_j - x_i)$
Is the new solution better than existing?
No
Yes
Reject
Accept
Keep the previous
Replace the previous
Is the termination criteria satisfied?
No
Yes
Stop
No
Teacher Phase
Learner Phase

Figure 5.2  Flowchart of TLBO algorithm [135]

### 5.3.1 The mechanism of the TLBO

TLBO functions in two stages with different processes, termed the teacher and learner phases, as set out by Rao [141]. In the former, learning takes place via the teacher, while in phase two, learners interact for learning to take place (Rao, 2015) [143].

1. *Teacher phase*

In the teacher phase, each learner learns via the teacher, who in turn aims to raise the mean average results across the learners they teach, in relation to their capacity to do so.

For iteration *i*, subject numbers or design variable numbers are assumed to be '*m*', with '*n*' representing learner numbers or population numbers (*k=1,2,…,n*). Moreover, $M_{j,i}$ represents mean learner outcomes for a specific subject '$j$' ($j$=1,2,…,m). The highest results $X_{total-Kbest,i}$ for each subject combined across the global learner population may be defined as the results obtained by the highest-performing learner *kbest*. At the same time, because teachers are generally viewed as educated individuals providing training for the learner to improve their outcomes, the highest-performing learner is designated the teacher. The gap which separates the current mean outcome for a subject and the teacher's related outcome for the subject is found through [144].

$$Differences\_\ Mean_{j,k,i} = r_i(X_{j.kbest,i} - T_F M_{j,i}) \tag{5.2}$$

Where:

$X_{j.kbest,i}$ is the result of the best learner in subject $j$.

$T_F$ is the teaching factor which decides the value of mean to be changed, and

$r_i$ is the random number in the range [0, 1].

Value of $T_F$ can be either 1 or 2. The value of $T_F$ is decided randomly with equal probability as,

$$T_F = round\ [1 + rand\ (1,0)\ \{2\text{-}1\}] \tag{5.3}$$

Where $T_F$ is not one of the TLBO algorithm parameters and it is not given as input to the algorithm. However, its value is randomly decided by the algorithm by applying Eq. 5.3. Following various experiments across different benchmarked functions, the findings show that

TLBO's performance is best for $T_F$ values between 1 - 2. Thus, the algorithm can be simplified by having the teaching factor be 1 or 2, following the rounding-up criteria as set by Eq. (5.3).

Based on the $Differences\_\ Mean_{j,k,i}$, the existing solution is updated in the teacher phase according to the following expression.

$$X'_{j,k,i} = X_{j,k,i} + Differences\_\ Mean_{j,k,i} \tag{5.4}$$

Where:

$X'_{j,k,i}$ is the updated values of $X_{j,k,i}$ , $X'_{j,k,i}$ is accepted if it gives better function value. All the accepted function values at the end of the teacher phase are maintained and these values become the input to the learner phase. The learner phase depends upon the teacher phase.

2. Learner phase

Stage two of TLBO is the learner stage, in which each learner interacts with the others in a random manner to improve their knowledge. Learners learn something new when learner they interact with possesses greater knowledge. For 'n' as the population size, there is random selection of a pair of learners, 'P' and 'Q', so that

$$X''_{j,p,i} = X'_{j,k,i} + r_i(X'_{j,p,i} - X'_{j,Q,i}),\ if\ X'_{total-p,i} < X'_{total-Q,\ i} \tag{5.5}$$

$$X''_{j,p,i} = X'_{j,p,i} + r_i(X'_{j,Q,i} - X'_{j,p,i}),\ if\ X'_{total-Q,i} < X'_{total-P,i} \tag{5.6}$$

Where, $X''_{j,p,i}$ is accepted if it gives a better function value. The Eq. (5.5) and Eq. (5.6) are for minimization problems. In the case of maximization problems, the Eqs. (5.7 - 5.8) are used.

$$X''_{j,p,i} = X'_{j,k,i} + r_i(X'_{j,p,i} - X'_{j,Q,i}),\ if\ X'_{total-Q,i} < X'_{total-P,\ I} \tag{5.7}$$

$$X''_{j,p,i} = X'_{j,p,i} + r_i(X'_{j,Q,i} - X'_{j,p,i}),\ if\ X'_{total-P,i} < X'_{total-Q,\ I} \tag{5.8}$$

### 5.3.2 Application of the TLBO

TLBO has been applied across various scientific and engineering areas since it was first proposed in 2011. Among these, it is particularly used in manufacturing, thermal, structural, civil, and electrical engineering, mechanical design, as well as electronics and computer engineering, the physical and chemical sciences, economics biotechnology [144].

Moreover, new applications for both the original and adapted algorithm are increasingly being developed, demonstrating the strong potential attached to TLBO [144]. Rao et al. (2011, 2012) have demonstrated that a lower number of function evaluations are required by TLBO in comparison with different types of optimisation algorithm. While some experimental work was not performed by these authors in identical setting, fewer evaluation was selected, demonstrating that the algorithm performed better.

Notably, different research groups had applied various function evaluation numbers in assessing benchmark functions. Stop conditions for specific benchmarked functions as applied by Rao et al. (2011, 2012a) using a 30-minute running duration outperformed those seen in other work. Moreover, TLBO outperformed other algorithms while offering simpler computation at larger scales, as seen with high-dimension problems. Despite this, generally speaking, in order to compare optimisation algorithms consistently, each one should have identical numbers of function evaluations across each of the benchmarks compared. Moreover, notably, algorithms needing fewer function evaluations for an identical optimised solution can be said to outperform others. Where the algorithm provides a globally optimal solution at under a specific function evaluation number, utilising further evaluations could simply lead to repetition of these results. A study by Rao and Patel (2012) introduces an elitist idea within TLBO and assesses the impact of this for this algorithm’s performance in restricted optimisation problem types. Furthermore, controlling variables such as elite and population sizes and generation numbers have been studied in terms of their impacts on how TLBO performs, through applying these in various combinations [145].

### 5.3.3 TLBO Implementation on FLQR Robogymnast system

The implementation of TLBO optimization on the FLQR controller for a Robogymnast system involves the following steps:

1. Define the problem: The first step is to define the control problem, which involves identifying the plant model and the control objectives. In this case, the plant model is

the Robogymnast system, and the control objective is to minimize the error between the desired output and the actual output.

2. Design the FLQR controller: The next step is to design the FLQR controller. The FLQR is a type of fuzzy logic controller that uses a quadratic cost function to minimize the error between the desired output and the actual output.
3. Define the optimization problem: The optimization problem is defined as finding the optimal values of the FLQR controller parameters that minimize the performance measure. In this case, the performance measure could be the ITAE, overshoot, or settling time.
4. Implement the TLBO algorithm: The TLBO algorithm is used to optimize the FLQR controller parameters.
5. Evaluate the optimized controller: Once the optimization process is complete, the optimized controller is evaluated by simulating the Robogymnast system with the optimized controller. The simulation results can be used to evaluate the performance of the optimized controller in terms of the performance measure selected.

The implementation of TLBO optimization on the FLQR controller for a Robogymnast system can lead to improved control performance in terms of the selected performance measure. Lastly, as mentioned above, the selected algorithm is implemented on FLQR controller for triple link robotic system (Robogymnast). In Table 5.1, the optimized TLBO parameters are presented, specifically focusing on two implemented parameters: the number of iterations and the population size

Table 5.1 TLBO Parameters

| **Parameter** | **Description** |
|---|---|
| Maximum Number of Iterations | 100 |
| Population Size | 30 |
| completion time | 34 hours |

FLQR-TLBO response:

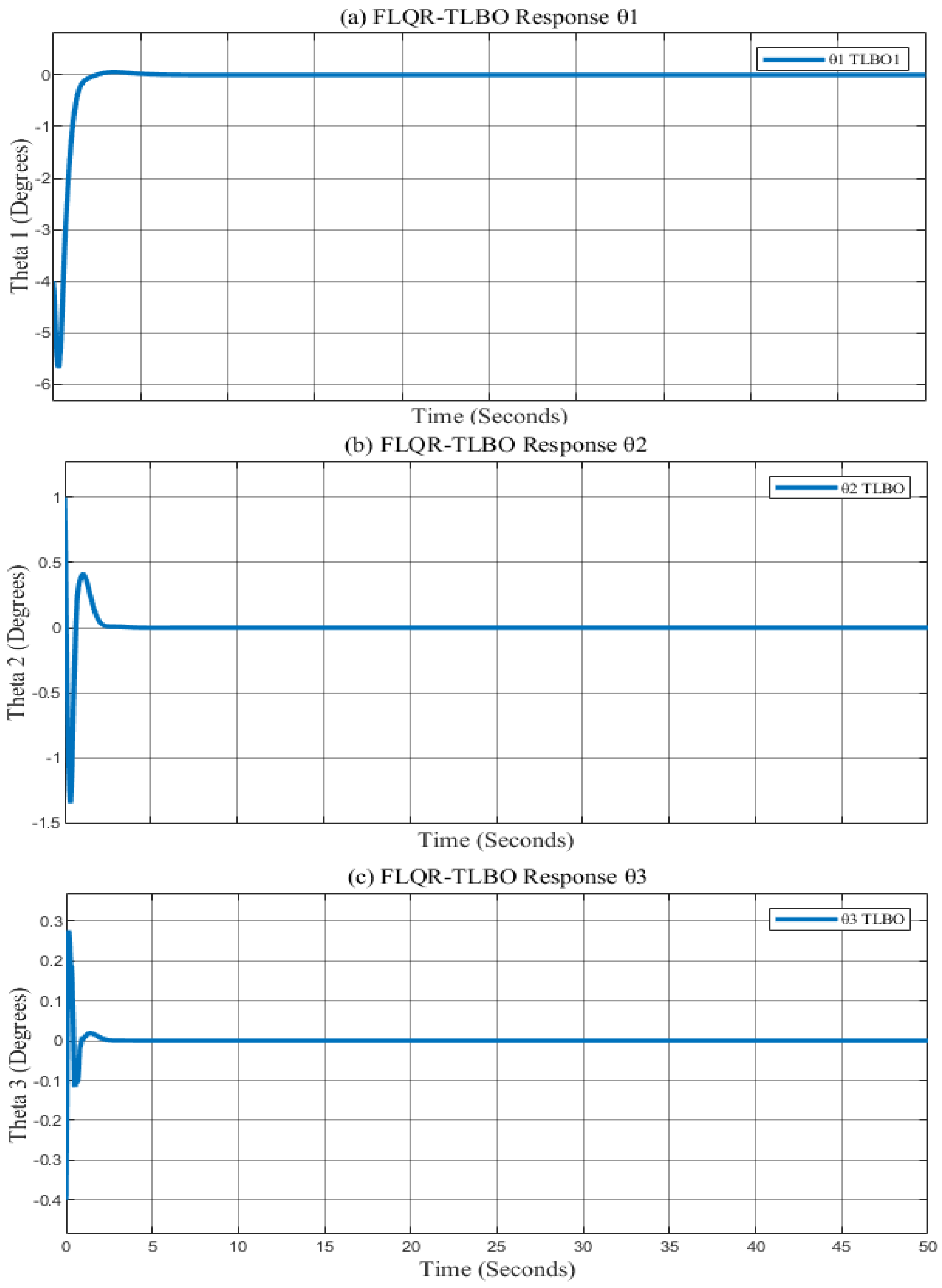


Figure 5.3 (a) Optimized FLQR-TLBO system response for 1st link of Robogymnast; (b) The FLQR-TLBO system response for 2nd link of Robogymnast; (c) The FLQR-TLBO system response for 3rd link of Robogymnast

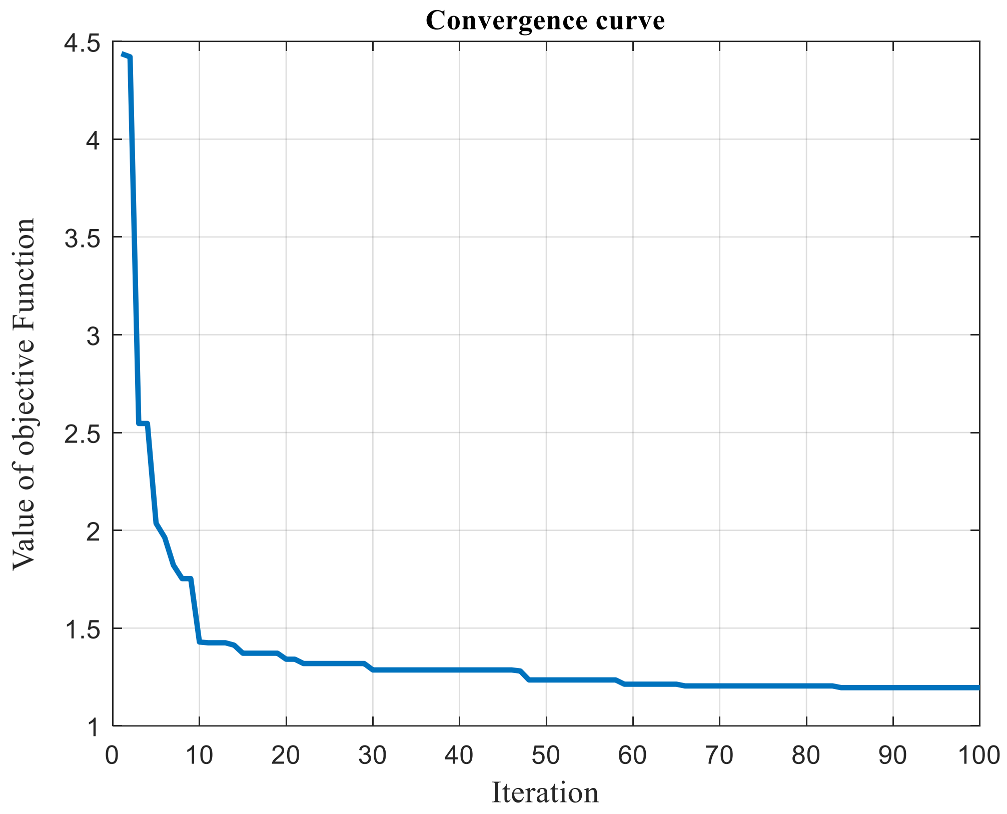


Figure 5.4 The convergence characteristic of TLBO Algorithm

Table 5.2 FLQR-TLBO Performance

| **Symbol** | **Controller** | $O_{sh}$ **(p.u)** | $U_{sh}$ **(p.u)** | $T_r$ **(s)** | $T_s$ **(s)** | **ITAE** |
|---|---|---|---|---|---|---|
| $\theta_1$ | FLQR- TLBO | 0 | -5.67 | 0.213 | 5.577 | 1.688 |
| $\theta_2$ | FLQR- TLBO | 0.39 | -1.32 | 0.0345 | 2.1668 | 0.3117 |
| $\theta_3$ | FLQR- TLBO | 0.23 | 0.4 | 0.0115 | 1.9675 | 0.02145 |

Table 5.2 displays the performance of different controllers for a system with three different values of the parameter angle theta. The performance measures used to evaluate the controllers include overshoot, undershoot, rising time, settling time, and ITAE (Integral of Time multiplied by Absolute Error). The optimized controller for all three values of thetas is TLBO-FLQR controller. This proposed controller has achieved zero overshoot and low undershoot compared to conventional FLQR for $\theta_1$, and low overshoot and undershoot for theta 2 and theta 3. The rising and settling times for these controllers are also relatively low, indicating that they can

quickly and effectively respond to changes in the system. The use of intelligent optimization algorithms in the design of the controller helps to automatically determine the best controller parameters to achieve the desired performance measures. This makes the design process more efficient and effective and can lead to better control performance.

## 5.4 PSO

The stochastic, population-derived particle swarm optimisation (PSO) algorithm was first proposed in the mid-90s by Eberhart and Kennedy and was inspired by swarm intelligence. The algorithm involves possible solutions known as particles flying in problem space as they follow the particles which are optimal at that moment. The approach comes from observed socially interactive animal behaviour: e.g., schools of fish and flocks of birds. PSO imitates food seeking behaviours which are both cooperative and competitive across a whole population. Swarms are groups of individuals 'particles' each representing a unique potential parameter set among those which are not known and for which optimisation is sought. Swarms begin by being populated with randomly generated solutions, and the swarm members then fly through the multiple-dimension solution space, changing location based on experiences drawn from itself and its adjacent swarm members. This is intended to lead to an efficient search of the space as particles swarm in the direction of optimal fit solutions from earlier in the iterative process, in order to find increasingly good solutions and ultimately converge upon one maximal/minimal solution. Particles' performances are evaluated individually based on a pre-set fitness function linked to the problem set. PSO is viewed as having potential in optimisation based on the fact that it is simple, does not present high computation costs, and that it performs well. Every individual particle is a potential solution for the optimisation issue, and it positions itself drawing on its experiences regarding the optimal location and this is termed its 'personal best position' (p-best). However, it also utilises the location belonging to the optimal particle across the whole population, and this is termed the 'global best position' (g-best). Each particle remembers the two types of best position, p-best and g-best, for every iterative stage. In addition, all of the particles have their own velocity [146].

PSO stops through a bounded condition attached to performance criteria. Frequently applied criteria for this include ITAE, IAE, ISE and ITSE, and different criteria offer differing features. In ISE for example, a larger error is penalised more harshly than a smaller error. Irrespective of the benefits of each approach however, the field lacks precision controls for overshoot, rising time and settling time. Therefore, the multi-purpose performance criterion is applied here to address the peak values for these factors [147].

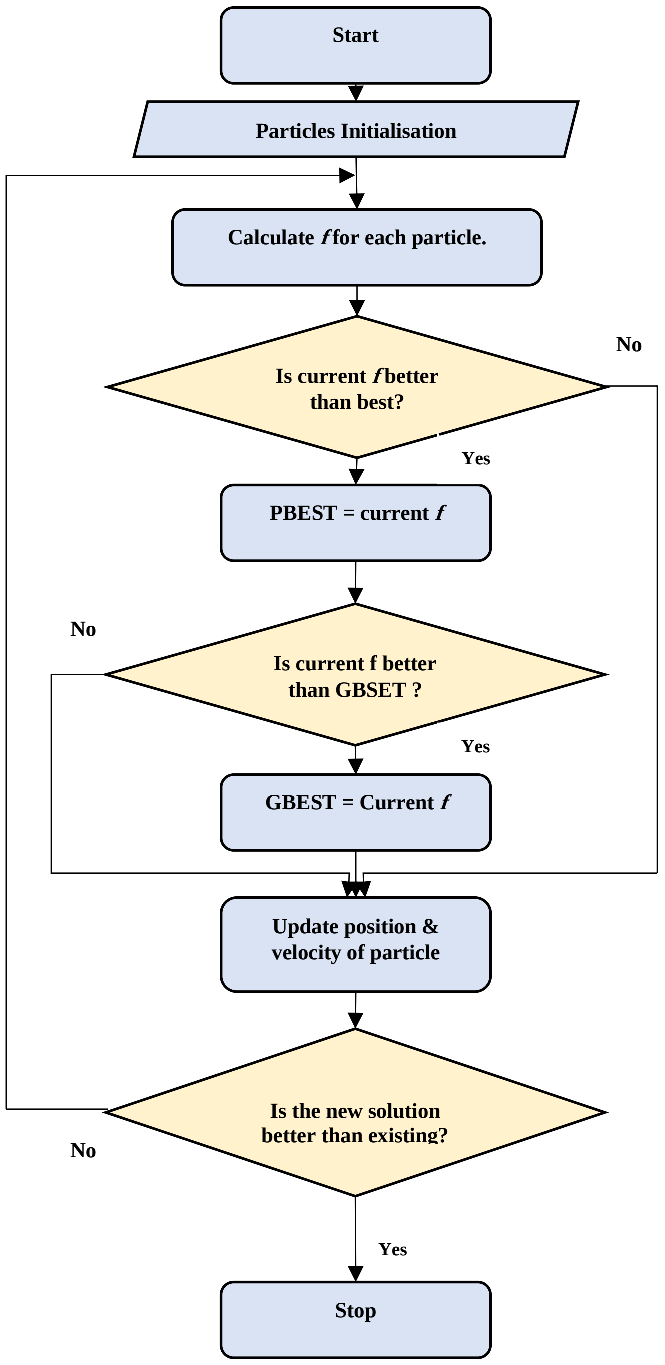

Start
Particles Initialisation
Calculate *f* for each particle.
Is current *f* better than best?
No
Yes
PBEST = current *f*
Is current f better than GBSET ?
No
Yes
GBEST = Current *f*
Update position & velocity of particle
Is the new solution better than existing?
No
Yes
Stop

Figure 5.5 Flowchart of PSO Algorithm

PSO is designed to calculate evolving problems and draws upon studies of the way in which birds behave when they are seeking a food source. Inspired by these behaviours, the algorithm seeks to iteratively find excellent solutions to the problem set, with potential solutions being imagined in PSO as dimension searches point spaces in the form of particles. Each member has a driven target function based on the position value of a decision, and every particle also decides flying velocity distance and direction. The algorithm follows the best particle and searches centrally within the solution space. Reynolds bases this on research detecting bird flight, in which a bird monitors its neighbours to a limited extent, while achieving global results in which it appears to be controlled through a central organising force, in which high complexity in the global phenomenon arises from far simpler rules interacting with each other [148].

### 5.4.1 Parameters of PSO

Various parameters are used in controlling PSO, including particle numbers, neighbourhood sizes, iteration numbers, acceleration coefficients, problem dimensions and inertia weight, in addition to randomly occurring values linked to the velocity update equation's socially and cognitively derived components [149]. As well as these, where constriction and velocity clamping are applied, the parameters of maximum velocity and constriction coefficient are regarded as highly significant, as reported by Engelbrecht in [150]. Such parameters strongly influence solutions gained in terms of convergence rate, quality and effectiveness, and are described in more detail in the list which follows:

1. *Population size*

The size of a swarm ($n_s$) refers to how many particles are present within that swarm globally. Greater numbers of particles can investigate a greater proportion of searching space for each iterative cycle: however, the larger the particle numbers, the more complex computation becomes for each iteration. Moreover, in some cases, a greater swarm size could result in fewer iterations being needed to optimise the solution to a problem. Based on heuristics in other published works, PSO has a larger chance of successfully finding a good solution where swarms of between 20 and 50 members are used, as reported by Eberhart and Shi in [151]. Despite this, there is evidence indicating that the optimal swarm size typically varies depending on the specific problem being addressed.

2. *Number of iterations*

As comparable to the size of swarms used, iteration numbers should also be closely associated with the problem being studied in order to achieve effective solutions through PSO. This is because, while using lower iteration numbers that could lead the searching stage to be ended too early, conversely, with higher iterative numbers, computation becomes increasingly more complex to achieve solutions of sufficient quality, where criteria for stopping the process rely on a given iteration number.

3. *Neighbourhood iteration*

Neighbourhood sizes correspond to how socially interactive the particles in a swarm are, with lower numbers of interactions within a small neighbourhood leading particle to converge slowly and reliably. Conversely, higher levels of interactivity within larger neighbourhoods increases convergence speeds. Thus, for example, where neighbourhood size is set at two, a particle (k) makes a fitness comparison with particle (k - 1) in addition to particle (k + 1) [149]. Based on reviewing heuristics in the literature, neighbourhood size generally comprises 15% of global population numbers for the majority of application types.

4. *Acceleration coefficients*

The cognitive/social element's influence on particle speeds is controlled via acceleration coefficients $c_1$and $c_{2,}$ as well as by random values $r_1$ and $r_2$ . The two acceleration coefficients must be adequately balanced, in which use of unsuitable values for $c_1$ and $c_1$ could lead to swarm divergence and cyclical behaviours.

5. *Velocity clamping*

Velocity clamping is a parameter within optimisation algorithms in which the balance of exploratory and exploitative aims is optimised. Application of the base version of PSO shows a rapid increase in velocity to far greater speeds, which leads particles to alter their positions quickly and makes the swarm diverge. This issue was addressed by the proposal of velocity clamping by Eberhart and Kennedy [151], in which speeds are clamped to remain within boundary limits. The greatest permissible speed to maintain the balance of overall exploratory and local exploitative activities is given by $V_{max}$. Where a particle travels at a higher speed than the value of $V_{max}$, its velocity is assigned instead as $V_{max}$. $V_{max}$ forms one of the significant parameters in the algorithm, due to its control of dramatic speed increases. A greater value of $V_{max}$ promotes exploratory activities globally, while introducing the potential for particles to miss good search areas.

6. *Inertia weight*

Inertia weight 'ω' as a coefficient for particles within the base algorithm for PSO was conceptualised initially by Shi and Eberhart [152], and sought to enhance convergence within the algorithm. Inertia weight acts to control exploratory activities both globally and locally, as well as to increase the swarm's exploitation capability through identifying how a particle's previously held speed affects its motion in the present time. Introducing inertia weight as a novel parameter is important as appropriate $\omega$ values lead to a reduction in mean iterations needed to identify a solution which is optimised to a satisfactory extent.

### 5.4.2 Application of the PSO

Studies show various ways in which PSO is applied, including in relation to energy, control, image processing optimising functions, and in robotics [153]. The algorithm is not conceptually complex, with only a small number of parameters available for modification. PSO can be used in various types of optimization problems, such as min-max, multi-objective, and constrained optimization. In addition, the algorithm has applications in structural and weight evolution in various types of neural networks (NNs). [154]. Work in [155] by Abe and Komuro involved tuning a neural network using PSO and applying it to gain energy savings in point-point movement in a flexible manipulator. The angle of joints as controlled through this NN served to suppress residual vibrations, thereby reducing motor torque to a minimum, as the objective function. These researchers studied the issue numerically as well as confirming their findings through experiment, to conclude the efficiency of PSO optimisation.

From the IEEE Xplore database, 7% of articles on applying PSO relate to control, and this forms one of the larger areas in the literature. Applications exist across automated generation control tuning, controller design, controlling traffic flows, adaptive inverse control, predictive control, PI/PID-type controllers [156], controlling strip flatness, ultrasonic motors, systems, and power plant control [157], as well as controlling chaotic systems, processes, adaptive compensation within a wavelength division multiplexing (WDM) network, fractional order controllers, controlling combustion, inertia system control, and controlling automatic landing [158]. Moreover, the field of robotics has demonstrated a number of studies which describe how PSO may be applied. Within robotics, this includes controlling a manipulator or arm, to plan and control movement, running in robots, collective robot searching, non-supervised robot learning, planning pathways, avoiding obstacles, robot swarms, navigating in unmanned vehicles, playing football, robotic vision, transportation, finding the source of odours, mapping of environments, and the voice-controlled robot [159].

Finally, based on its simple character and broad applications, PSO has attracted considerable research interest across a range of disciplines. Houssein et al.'s [160] comprehensive systematic review of particle swarm optimisation included stages of development, recently emerging directions, hybridised uses, parallelisation, and the spectrum of uses for this algorithm. With various research groups studying development of this algorithm, there is an increase in related studies, and some authors have pointed to issues they have detected in PSO as originally developed. These include too-early convergent tendencies, performance problems and others. However, much progress has been made towards minimising such issues and making PSO more efficient and effective.

### 5.4.3 PSO Implementation

The implementation of Particle Swarm Optimization (PSO) on the Fuzzy Logic-based Quadratic Regulator (FLQR) controller for a Robogymnast system involves the following steps:

1. The first step is to define the control problem, which involves identifying the plant model and the control objectives. In this case, the plant model is the Robogymnast system, and the control objective is to minimize the error between the desired output and the actual output.
2. The next step is to design the FLQR controller. The FLQR is a type of fuzzy logic controller that uses a quadratic cost function to minimize the error between the desired output and the actual output. The FLQR controller can be designed using standard fuzzy logic design techniques.
3. The optimization problem is defined as finding the optimal values of the FLQR controller parameters that minimize the performance measure. In this case, the performance measure could be the ITAE, overshoot, or settling time.
4. Implement the PSO algorithm: The PSO algorithm is used to optimize the FLQR controller parameters. The PSO algorithm is a type of swarm intelligence algorithm that mimics the behaviour of a swarm of particles searching for the optimal solution.
5. Once the optimization process is complete, the optimized controller is evaluated by simulating the Robogymnast system with the optimized controller. The simulation results can be used to evaluate the performance of the optimized controller in terms of the performance measure selected.

Lastly, Tables 5.3 and 5.4 shows the PSO parameters.

Table 5.3 PSO Parameters

| **Parameter** | **Description** |
|---|---|
| Maximum Number of Iterations | 100 |
| $\mathrm{pop}^{\mathrm{n}}$ size | 30 |
| Completion time | 36 hours |

Table 5.4 PSO Algorithm parameters

| **No. Particles** | $\mathbf{W_{min}}$ | $\mathbf{W_{max}}$ | **C1** | **C2** | $\mathbf{V_{max}}$ |
|---|---|---|---|---|---|
| 30 | 0.7 | 0.9 | 1.5 | 1.5 | 0.002 |

Where:

| | |
|---|---|
| $\mathrm{W_{min}}$ | Represents the minimum value of the inertia weight |
| $\mathrm{V_{max}}$ | Is the maximum value of the inertia weight |
| C1 | Represents the particle's self-confidence and determines the weight given to the particle's personal best solution |
| C2 | Represents the particle's social influence and determines the weight given to the swarm's global best solution |
| $\mathrm{V_{max}}$ | Denotes the maximum velocity limit of the particles in the search space |

FLQR-PSO response:

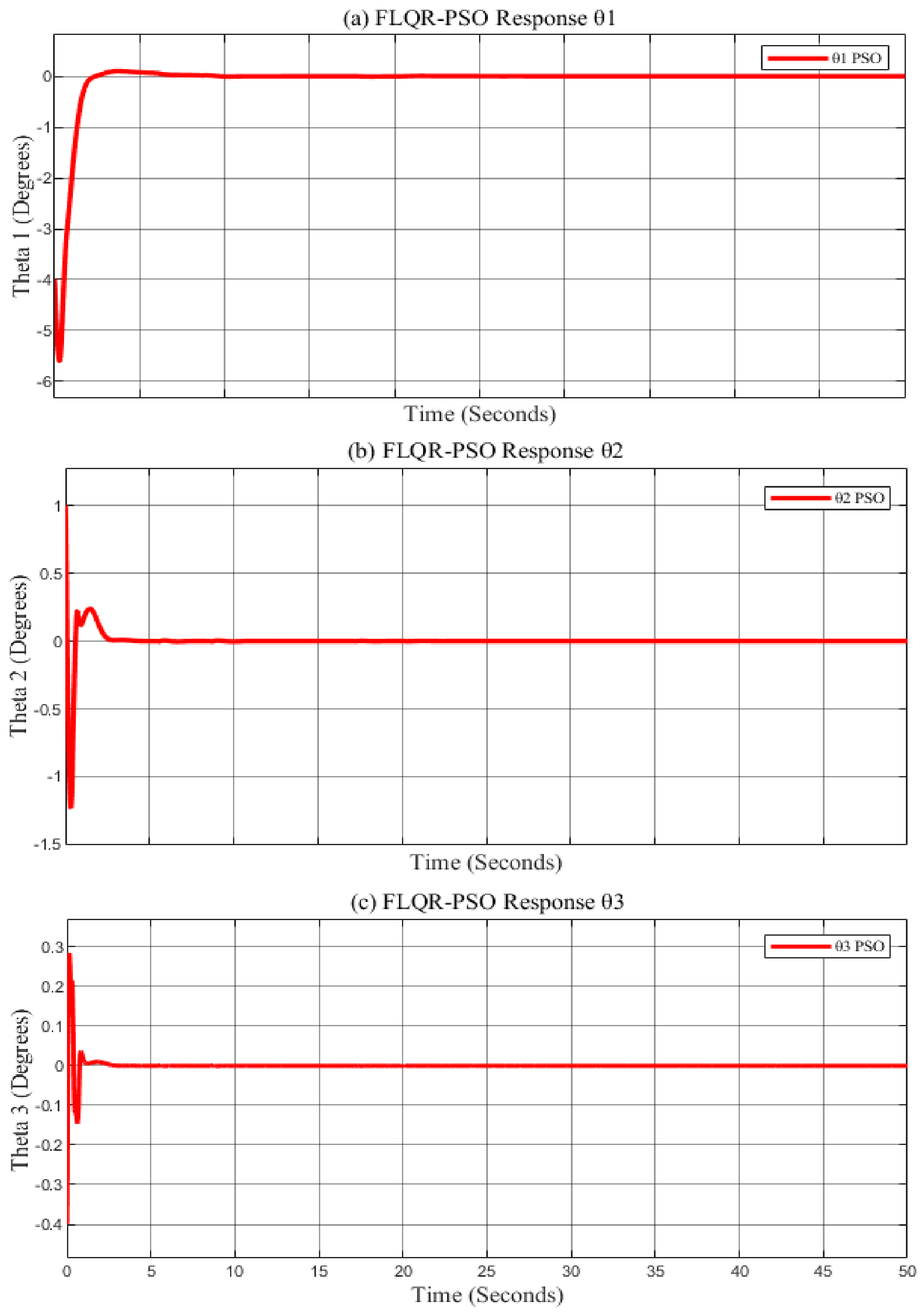


Figure 5.6 (a) Optimized FLQR-PSO system response for 1st link of Robogymnast; (b) The FLQR-PSO system response for 2nd link of Robogymnast; (c) The FLQR-PSO system response for 3rd link of Robogymnast

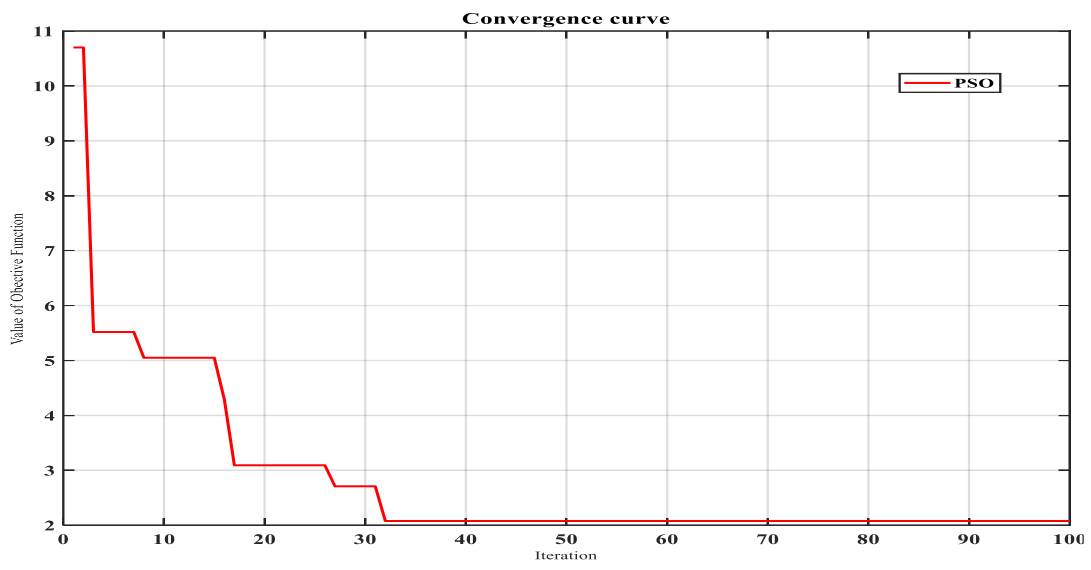


Figure 5.7 The convergence characteristic of PSO Algorithm

Table 5.5 FLQR-PSO Performance

| **Symbol** | **Controller** | $O_{sh}$ **(p.u)** | $U_{sh}$ **(p.u)** | $T_r$ **(s)** | $T_s$ **(s)** | **ITAE** |
|---|---|---|---|---|---|---|
| $\theta_1$ | FLQR- PSO | 0.028 | -5.599 | 0.2820 | 5.5862 | 2.688 |
| $\theta_2$ | FLQR- PSO | 0.20 | -1.222 | 0.0301 | 1.1731 | 0.3117 |
| $\theta_3$ | FLQR- PSO | 0.24 | -0.15 | 0.0400 | 2.3108 | 0.02145 |

Referring to Table 5.5 demonstrates the results obtained by implementing of the Particle Swarm Optimization (PSO) algorithm on the Fuzzy Logic-based Quadratic Regulator (FLQR) controller for a Robogymnast system, for three different values of the control parameter of each links. The performance of the PSO-FLQR controllers is evaluated using different performance measures, including overshoot, undershoot, rising time, settling time, and ITAE. The results show that the PSO-FLQR controller is able to reduce overshoot and settling time compared to the untuned FLQR controller for all three values of theta. Additionally, the PSO-FLQR controller has a longer rising time than the TLBO-FLQR controller for theta 1 and 2, while the rising time is similar for theta 3. Finally, the ITAE values are higher for the PSO-FLQR controller for all three values of theta. Overall, the results suggest that the PSO-FLQR controller performs well in terms of reducing overshoot and settling time, but at the cost of increased undershoot and longer rising time. The higher ITAE values suggest that there may be room for further optimization of the controller. Also, Figure 5.7 shows the convergence characteristic of the PSO Algorithm.

## 5.5 Optimisation comparison

In this subsection a comparison between TLBO and PSO algorithm are displayed in Figure 5.8 and 5.9, also, the finding is presented in Table 5.6.

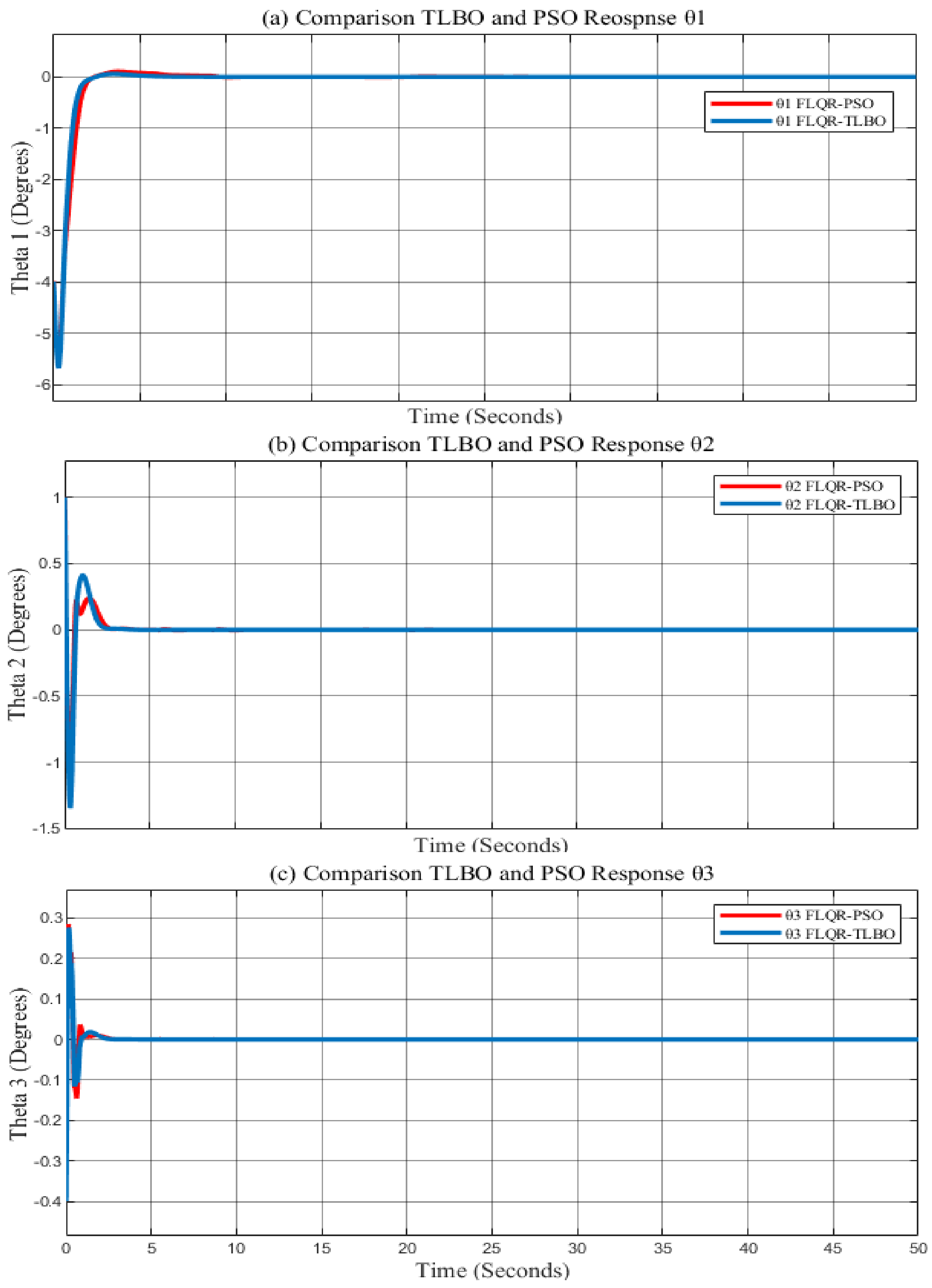


Figure 5.8 The response of TLBO and PSO algorithms

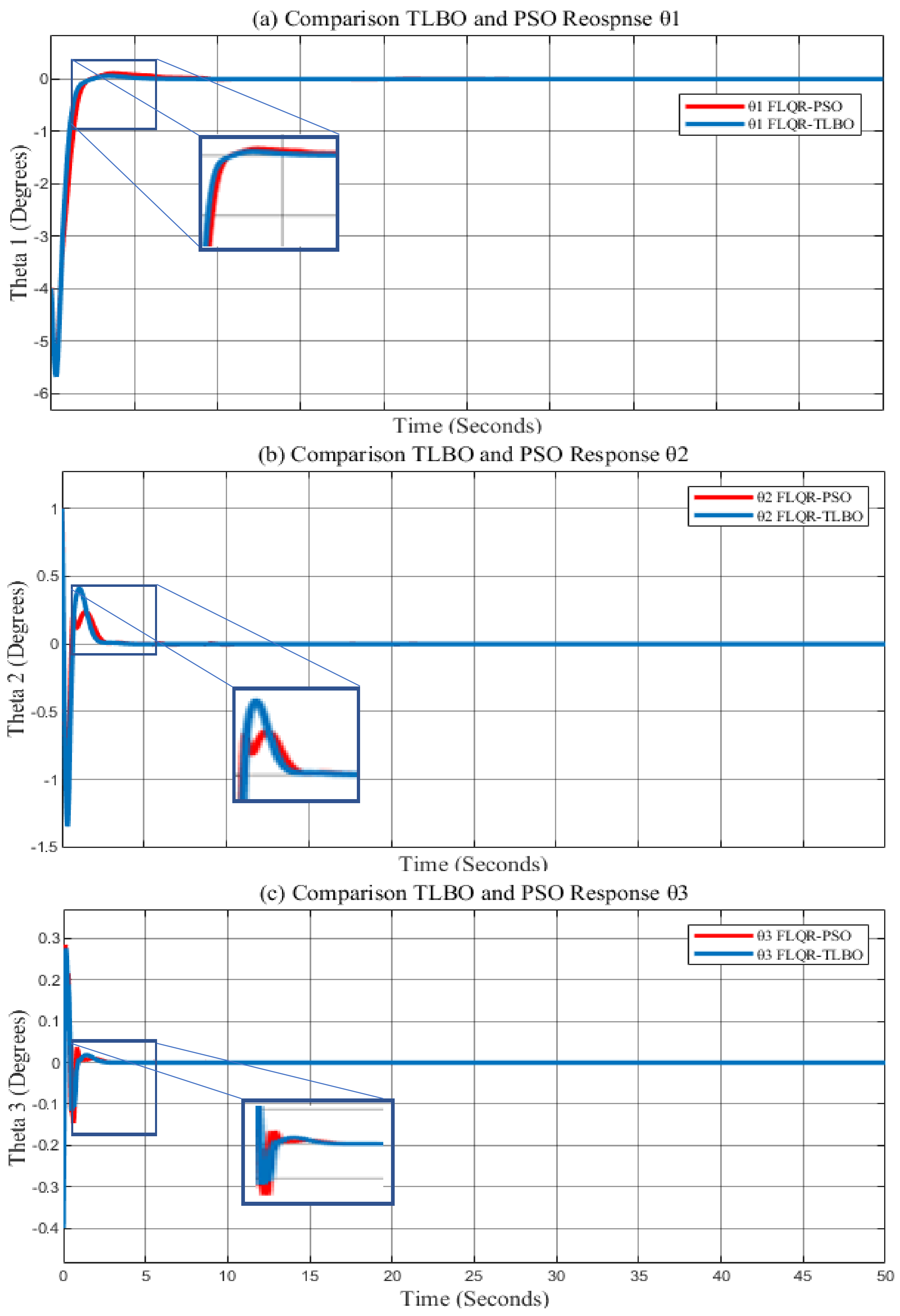


Figure 5.9 (a) Optimized FLQR for TLBO and PSO system response for 1st link of Robogymnast; (b) The TLBO and PSO system response for 2nd link of Robogymnast; (c) The TLBO and PSO system response for 3rd link of Robogymnast

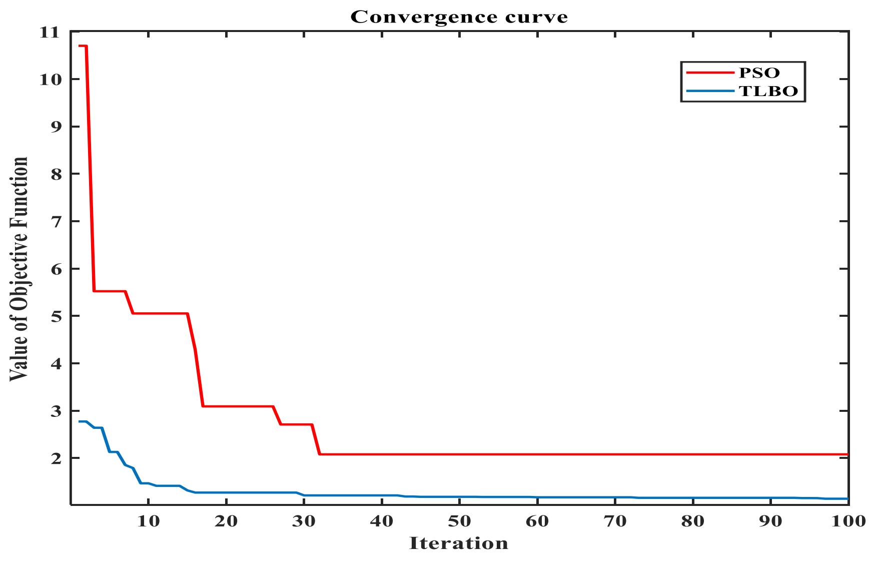


Figure 5.10 TLBO and PSO Convergence

TLBO algorithm has several convergence characteristics that make it an effective optimization algorithm. One important characteristic is that it has a strong global search capability, meaning that it is able to find the global optimum of a function in a satisfactory amount of time. Additionally, TLBO has been shown to have a fast convergence rate, meaning that it can quickly converge to a good solution in a relatively specific number of iterations. On the other hand, the convergence characteristic of the PSO algorithm is dependent on several factors, such as the swarm size, the velocity of the particles, the maximum number of iterations, the fitness function, and the inertia weight. The convergence of the PSO algorithm is determined by how fast the algorithm reaches the optimal solution and how accurately the solution is found.

Table 5.6 Comparison between TLBO and PSO response

| Symbol | Controller | $O_{sh}$ $(p.u)$ | $U_{sh}$ $(p.u)$ | $T_r$ (s) | $T_s$ (s) | ITAE |
|---|---|---|---|---|---|---|
| $\theta_1$ | FLQR- TLBO | 0 | -5.67 | 0.213 | 5.577 | 1.688 |
| | FLQR- PSO | 0.028 | -5.599 | 0.2820 | 5.5862 | 2.688 |
| $\theta_2$ | FLQR- TLBO | 0.39 | -1.32 | 0.0345 | 2.1668 | 0.3117 |
| | FLQR- PSO | 0.2 | -1.222 | 0.0301 | 1.1731 | 0.3117 |
| $\theta_3$ | FLQR- TLBO | 0.23 | -0.4 | 0.0115 | 1.9675 | 0.02145 |
| | FLQR- PSO | 0.24 | -0.15 | 0.0400 | 2.3108 | 0.02145 |

With reference to the preceding Figures, movements in the 3 links are illustrated sequentially, with a relating to link one, comparable to the gymnast's arms, comparing the Tables, it is evident that for the same plant and controller combination, the TLBO algorithm generally resulted in better performance than the PSO algorithm, as measured by metrics such as overshoot, undershoot, rising time, settling time, and ITAE.

For example, for theta1, both TLBO and PSO had zero overshoot, but TLBO had a larger undershoot of -5.67 *p.u* compared to -5.599 *p.u* for PSO. TLBO also had a smaller rising time and settling time, and a lower ITAE. Similarly, for $\theta_2$ and $\theta_3$. TLBO outperformed PSO in terms of overshoot, undershoot, rising time, settling time, and ITAE. Overall, the results suggest that TLBO may be a more effective optimization algorithm for tuning the FQLR controllers in this control problem.

It is worth mentioning that obtaining optimal values for FLQR structures requires a relatively lengthy computational time. For instance, using the TLBO method to enhance the FLRQ parameters takes more than 36 hours of computation. It is also approved that FLQR-TLBO technique is the best optimal method to implement for controlling the Robogymnast system.

## 5.6 Summary

This chapter covers the description of different algorithms that have been used to optimise the given problem in terms of overview, mechanism and application used for TLBO and PSO algorithms. Both algorithms have been implemented successfully in different application domains. The merits of these algorithms are favourable to the researchers to utilize them to solve a wide range of optimisation problems. Therefore, in this work, the TLBO is proposed to tune the parameters of the optimal FLQR controller suggested to optimize the triple-link robotic system, and the conventional PSO is also used to compare the response of the proposed system.

In conclusion, based on the presented findings, the TLBO-FLQR controller has been effective in proposing the system with different values of the parameter based on theta analysis. The combination of the TLBO optimization algorithm and the FLQR controller has helped to achieve good performance in terms of overshoot, undershoot, rising time, settling time, and ITAE. In summary, the FLQR controller and TLBO algorithm have demonstrated greater effectiveness compared to conventional controllers. The next chapter provides practical optimized results of the Robogymnast system compared with a simulation MATLAB/Simscape model in detail.

# Chapter 6:

# Motion planning of Robogymnast

## 6.1 Introduction

The Robogymnast was chosen to represent a complex, underactuated multiple-link mechanical system in order to evaluate and compare control systems based on a range of methods [16]. Designing a control system with under actuation presents challenges because full-state feedback linearization around a fixed point of equilibrium is often not possible for this type of mechanism, which is also frequently not small-time local controllable (STLC) [17]. This has led to considerable research interest in developing an underactuated system in the fields of control engineering and robotics. Inverted pendulums involve a component that swings freely from a fixed location and is suspended under the action of gravitational forces. Work involving regulating movement frequently uses this type of mechanism, and both hybrid and chaotic systems can be demonstrated using this approach [19]. An important robotics challenge is presented by the problem of balancing triple-inverted pendulum systems, and this is based on their similarity to the structural and balance factors of the human body. The acrobat is a robotic system that mimics acrobatic activity in humans, has an inverted pendulum form, and is designed to have instability and is underactuated. This makes the robot ideal for theory- and practice-based work on non-linear controls [21]. The acrobat was designed to balance using a specially developed, intelligent controller, which blended conventional control, fuzzy control, and adaptive fuzzy control in order to achieve swing, catch and balance in inversion [22]. The controllers tested were based on state variable feedback, as well as proportional-integral-derivative and linear-quadratic regulation approaches.

## 6.2 Robogymnast model design

The Robogymnast model is a complex robotic system that is designed to perform various gymnastic tasks autonomously. The design of the model involves the integration of various mechanical, electrical, and control components. The mechanical components include the body, limbs, joints, and actuators, while the electrical components include sensors, motors, and controllers. The control system is responsible for processing sensory data, generating control signals, and regulating the movement of the robot [23].

# Chapter 6: *Motion planning of Robogymnast*

The overall design of the Robogymnast model is based on the principles of biomechanics and robotics. The model is designed to mimic the movements and abilities of a human gymnast, while also incorporating advanced robotic technologies. The model is capable of performing a wide range of gymnastic manoeuvres, including swinging and motion planning, to ensure the effectiveness of the Robogymnast model, several design considerations must be considered. These include the weight and balance of the robot, the range of motion of the joints, the precision of the sensors, and the responsiveness of the control system. By optimizing these design factors, the Robogymnast model can perform complex gymnastic tasks with high accuracy and efficiency [63].

This section of the thesis consists of two main components. Firstly, it involves the development of a Simscape simulation model to accurately simulate the motion of the system. Secondly, it includes the presentation and analysis of the results obtained from the optimized hardware design, which validate the swinging motion of the system.

## 6.2.1 Simulation Design (Simscape Model)

The Simscape MATLAB model is a highly effective simulation tool that can accurately simulate a robotic system's dynamic behaviour. By enabling simulation under various loads and operating conditions, this model can evaluate the motion planning algorithm's effectiveness. To validate its performance and identify potential issues, the proposed motion planning algorithm utilizes the Simscape MATLAB model.

This subsection goes into detail regarding the modelling and control of a humanoid robot using MATLAB/Simscape/Multibody. The focus of the robot's design was to achieve human-like gymnastic behaviour. As such, the robot was designed with joints that replicate those found in the human body. The upper link corresponds to the arms, the middle link appears as the torso, and the lower link represents the legs.

In the physical world, the acrobat is subjected to gravitational acceleration force (g). The acrobat consists of three links and three-point masses. Figure 6.1 displays the acrobat's configuration, providing a visual representation of the robot's design. The use of the Simscape MATLAB model in the robot's design and control has proven to be highly effective in achieving human-like motion and behaviour, making it a valuable tool for future research and development in this field.

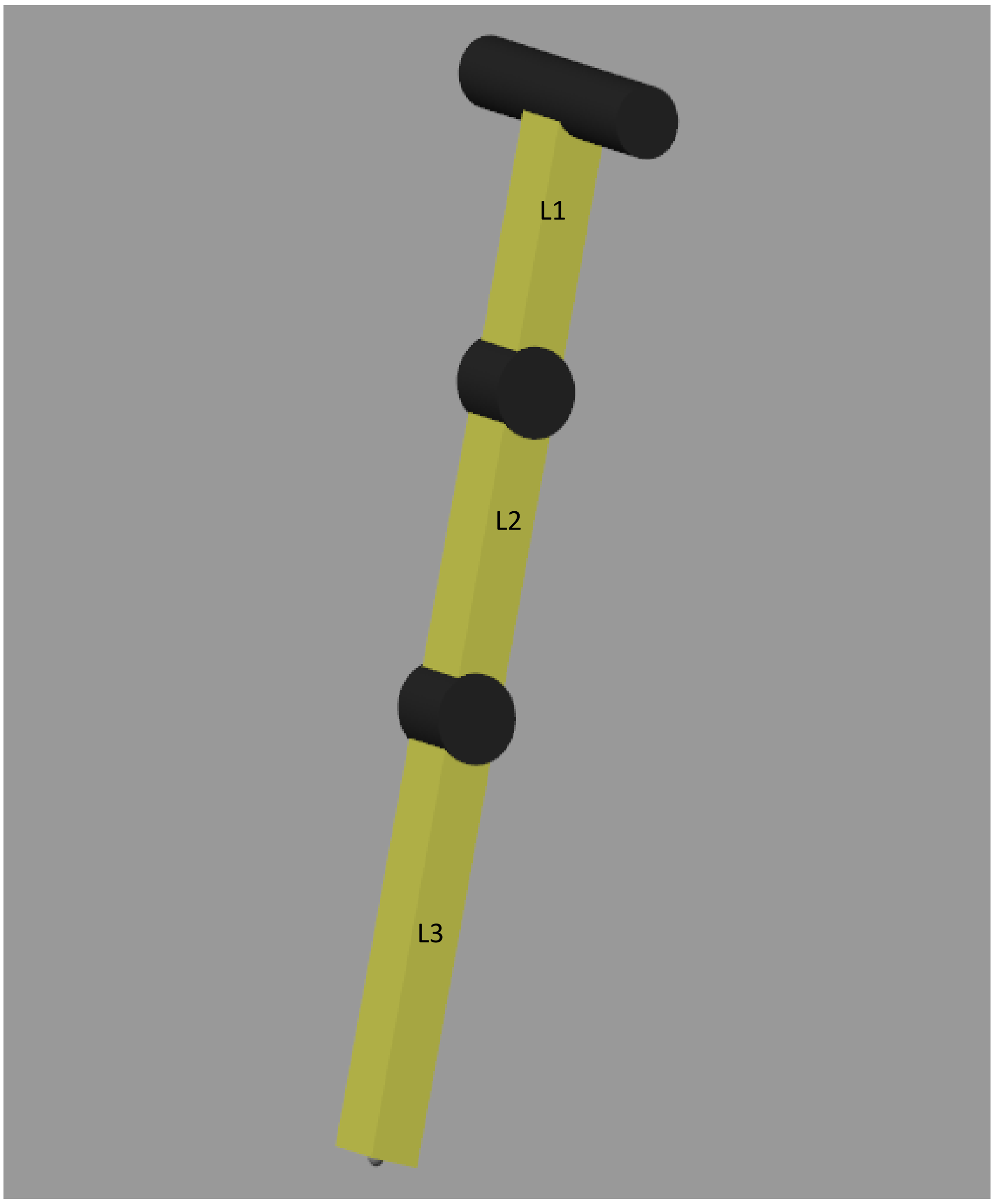


Figure 6.1 Triple-link robotic model

The first link's angle is labelled $q_1$, while the second and third angles are labelled $q_2$ and $q_3$, respectively. Because the equations of motion contain numerous trigonometric functions, a shorthand notation is employed for sine and cosine functions. Since there are plenty of trigonometric functions in the equations of motion, that use a shorthand for sinusoid functions $C_\theta$ represents cos (θ), and $S_\theta$ denotes sin (θ), also, $\dot{q}$ signifies the first-order time derivative of q, and $\ddot{q}$ represent the second time derivative of q. Torques, whether it arises from gravity or control input are denoted by τ.

Additionally, Table 6.1 represents the exact parameter of the real Robogymnast system that is implemented in the simulation model to validate the real system. The table includes critical information on the physical system, such as link lengths, masses, and etc. These parameters are essential for accurately simulating the dynamics of the Robogymnast system and validating the model's performance against real-world data.

Table 6.1 Acrobat Simscape parameters

| **Parameters** | **Link 1** | **Link 2** | **Link 3** |
|---|---|---|---|
| **Length** | $l_1$= 0.16 m | $l_2$= 0.180 m | $l_3$ = 0.245 m |
| **Mass** | $m_1$ = 1.2 kg | $m_2$ = 1.2 kg | $m_3$ = 0.5 kg |
| **Theta** | $\theta_1$ = 0 | $\theta_2$ = 0 | $\theta_3$ = 0 |

The parameters presented in Table 6.1 play a pivotal role in validating the performance of the simulation model against empirical data. By accurately modeling the dynamics of the Robogymnast system, designers can leverage the simulation model to forecast the behavior of the real system under varying conditions. They can subsequently contrast these prognostications with actual data obtained from the real system to ascertain the accuracy of the simulation model. This iterative procedure is critical in ensuring that the simulation model can effectively emulate the behavior of the physical system, thereby allowing designers to test and refine the system's performance.

Figure 6.2 shows the design of a triple-link robotic system to simulate the Robogymnast. This design was developed by using MATLAB (Simscape) tool. The design process utilized the MATLAB/SimMechanics toolbox, which enabled the creation of a highly accurate simulation model. The SimMechanics toolbox provided a range of modelling tools and features that enabled the creation of a realistic and dynamic simulation of the robotic system.

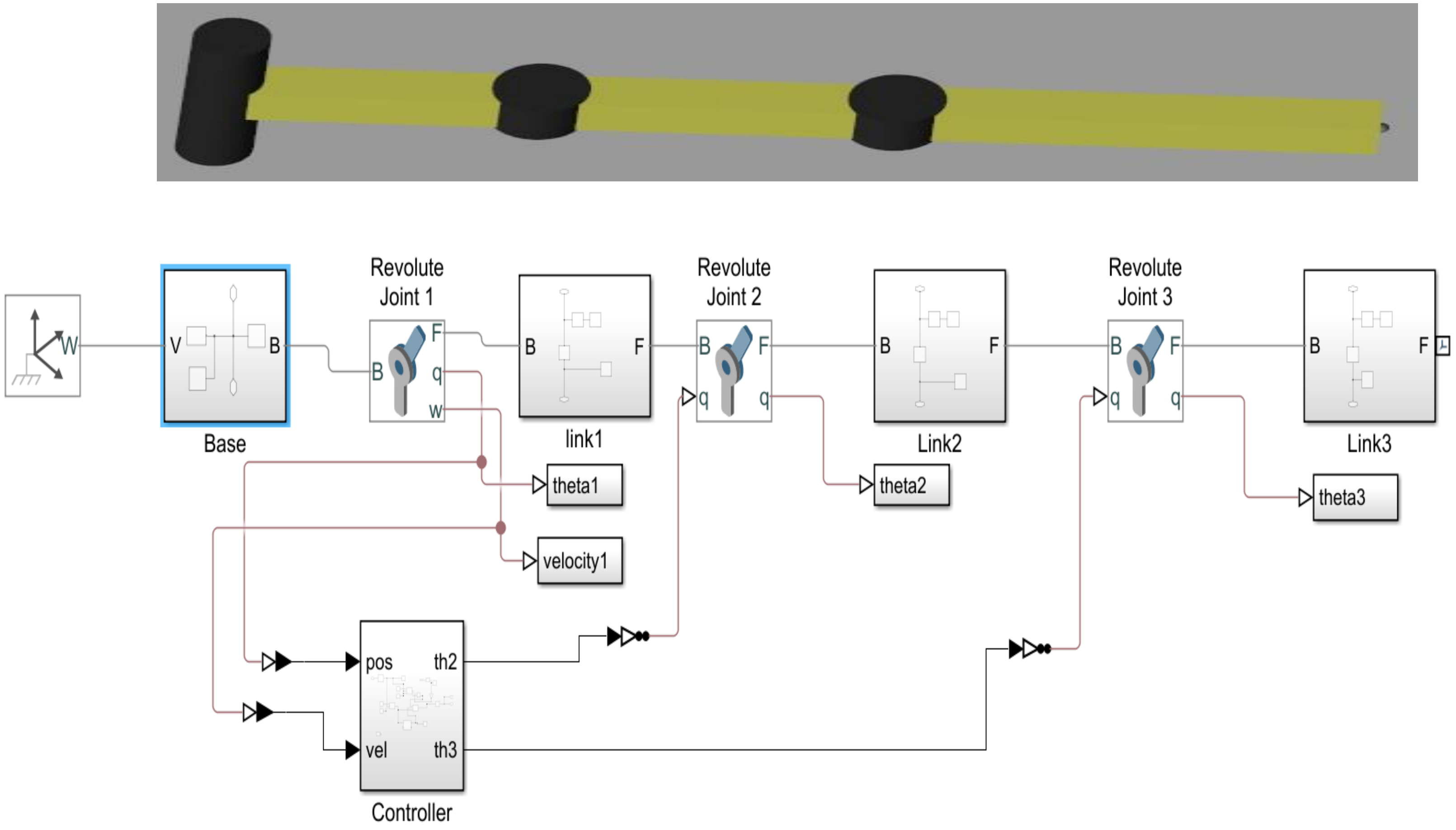


Figure 6.2 The designs of the triple link robotic system using Simscape/Multibody

The MATLAB/SimMechanics model of the ACROBAT as shown in Figure 6.2. Additionally, for this model, the initial conditions of ACROBAT joint positions are set to $\theta_1$, $\theta_3$, $\theta_3$ = 0°. The parameters used in MATLAB/SimMechanics are exactly the same as those of the physical system. The simulations are performed by the sampling time 1ms and 10s simulation time. A numerical method Bogacki-Shampine solver is selected with fixed-step. Lastly, in a Simulink/Simscape system, the control signal is an essential part of the overall control strategy. It is used to adjust the behaviour of the system in response to the input and output signals. The control signal is usually generated by a controller, which takes the system's current state and the desired state as input and produces the control signal as output.

In the following, Figure 6.3 shows different views from the virtual reality model of the triple-link robotic system in MATLAB Simulink is shown simultaneously.

**Simscap results:**

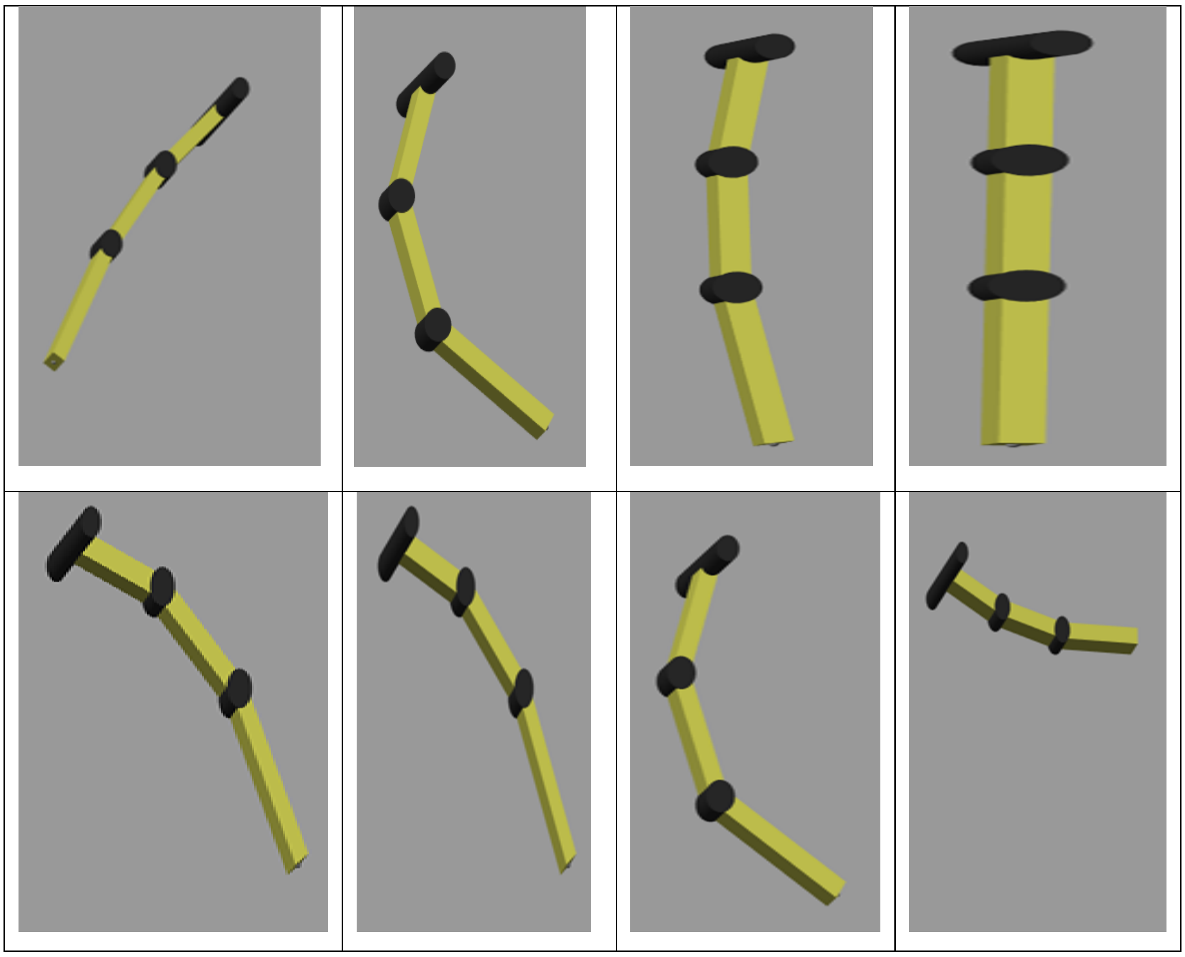


Figure 6.3 Side view of Simulink triple link motion

Figure 6.4 depicts the output of a sinusoidal response of a triple-link robotic system that has been modelled and simulated using MATLAB Simulink/Simscape. The system is controlled by an optimized TLBO-FLQR controller.

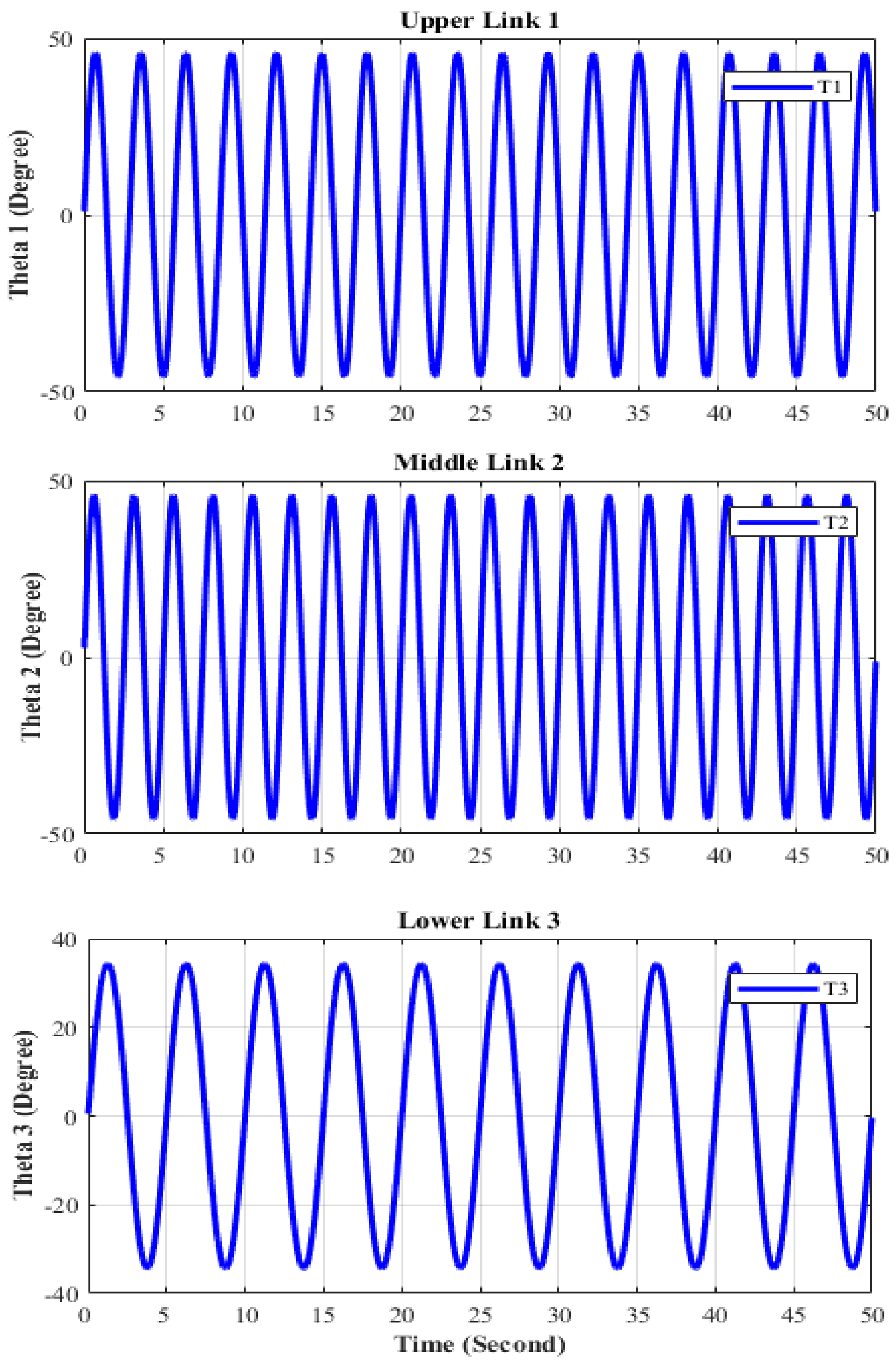


Figure 6.4 The sinusoidal output of triple link robotic system MATLAB Simscap controller

### 6.2.2 Hardware design

This section considers the setup of the system, where the system itself is attached by sensors connecting via joints 2 and 3 with dual potentiometers, and then the header connects with the encoder to detect the absolute value of motion. On the other hand, the operating system, as shown in Figure 6.5, contains 2 stepper drivers powered by 5V along with programming by an STM-32 microcontroller connecting to a PC to instruct the movement of the system and to read the signal returned from the sensors.

## 6.3 Results and Comparison

### 6.3.1 Practical outcomes

In this subsection, the initial results of the system are discussed for each link. The advancement of technology has paved the way for the development of the Robogymnast system, a gymnastics robot designed to perform and control gymnastics movements. This technology has significant practical outcomes in the field of robotics,

This work is applied to a triple-link Robogymnast mechanism, which imitates the gymnastic action of swinging-up to freely rotate over a high bar. The diagram given in Figure 6.5 illustrates the operating system components, which it is operated via two stepper motors, with each being subjected to the stepper driver control to achieve smooth movement. The control system programming is performed through the microcontroller STM32 using C++ language for translating commands between the robotic system, the control system, and the PC. Each link has its own sensor and link 1 connects to a rotary encoder, with the remaining links connecting to potentiometers 2 and 3, respectively, to allow for the detection of absolute angles across every position.

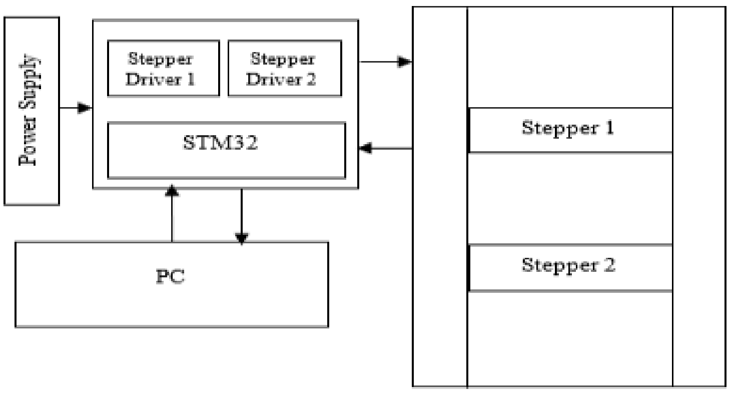


Figure 6.5 Robogymnast operation system

### 6.3.2 Initial motion planning

This part shows the initial motion without any improvement (optimisation), the motion results can be seen in Figures 6.6 (a-c), which demonstrate each link is in motion simultaneously, in which theta 1 represents the $\theta_2$ (stepper 1) and t3 (stepper 2) response, based on the opposing motion of the two motors. The second joint is operated by stepper motor 1, and if it moves in the positive direction, stepper 2 then moves in the opposite direction. In addition to this, the bottom link, which is link 3, is operated by stepper 2, and this begins to reverse at the point when the first motor moves forward, beginning the movements towards the up-swing of the robot in this system. Based on this, Table 6.2 provides the estimated values for the degree of motion in the multiple-link mechanism in both scenarios when, on average, the maximum points in both directions are positive and negative (forward and backwards).

#### 6.3.2.1 Initial results

The data from the initial motion of a triple-link robot gymnast was collected by a Python program code using appropriate sensors or motion capture devices. These sensors or devices can track the position and orientation of the robot's links in real time and provide data that can be processed using the Python program.

The collected data is processed using a Python program. The control system programming is performed through the microcontroller STM32 programming by C++ language. Each link has its own sensor, with link 1 connected to a rotary encoder and links 2 and 3 connected to potentiometers to detect absolute angles across every position. This part highlights the importance of accurate data collection and analysis for obtaining reliable results from the simulation or visualization, emphasizing the need for careful design of the Python program to ensure that it can collect high-quality data and process it accurately.

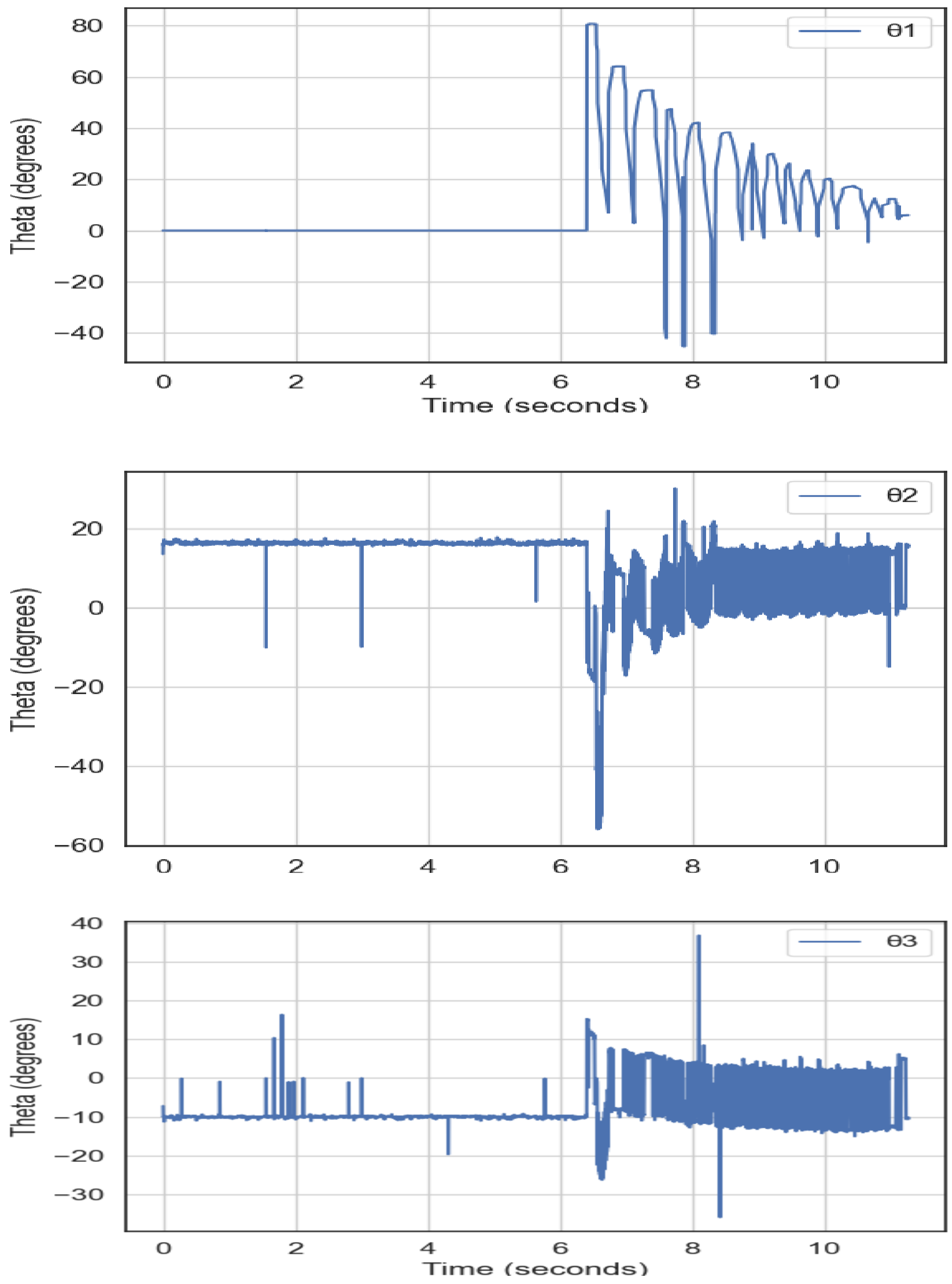


Figure 6.6 (a) Upper link free rotate; (b) middle link (stepper motor 1); and (c) lower link (stepper 2)

Table 6.2 Robogymnast Motion Results

| **Symbol** | **Parameters** | **Average (Degrees)** | **Max Point (Degrees)** |
|---|---|---|---|
| $\boldsymbol{\theta_1}$ | Link 1 (free rotate) upper | (−5) to 25 | (−45) to 80 |
| $\boldsymbol{\theta_2}$ | Link 2 (Motor 1) middle | (−5) to 15 | (−45) to 25 |
| $\boldsymbol{\theta_3}$ | Link 3 (Motor 2) lower | (−10) to 10 | (−35) to 35 |

Table 6.2 lists three symbols of the initial motion results of the system which represent each link average and maximum point of its movement. The parameters indicate the physical properties of the motion of each link in the system. Specifically, $\theta_1$, $\theta_2$ and $\theta_3$ represent the angles of link 1, link 2, and link 3, respectively. These parameters are essential in determining the overall movement in degrees.

This table provides specific numerical values for the average and maximum point of movement for each of the three links in a mechanical system. The symbol $\theta_1$ represents the upper link, which is free rotate, and has an average range of motion between -5 to 25 degrees and a maximum point range between -45 to 80 degrees. $\theta_2$ represents the middle link, which is connected to Motor 1 and has a narrower range of motion compared to $\theta_1$, with an average range of -5 to 15 degrees and a maximum point range between -45 to 25 degrees. Lastly, $\theta_3$ represents the lower link, which is connected to Motor 2 and has an average range of motion between -10 to 10 degrees and a maximum point range between -35 to 35 degrees.

Furthermore, by analysing the average and maximum point values for each link, it can be noticed that if there are any limitations that may affect the system's performance. For example, the narrower range of motion for $\theta_2$ compared to $\theta_1$ may suggest that the motor connected to this link is not as powerful or efficient, which could be addressed by optimizing or adjusting the system design.

Overall, this Table provides important information about the motion properties and expected behaviour of a mechanical system. Understanding these parameters and angle ranges is crucial in designing and analysing mechanical systems for improvement.

## 6.4 Enhanced results

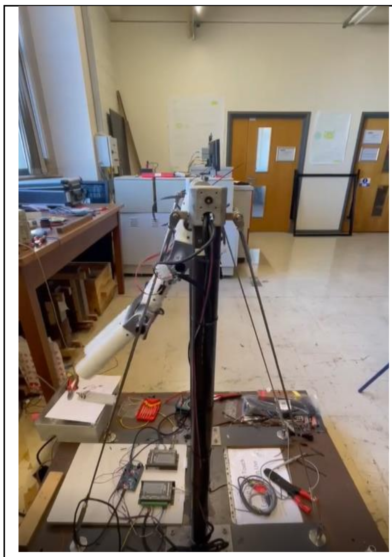
In this section, the focus is on the optimized outcomes that were achieved through the implementation of FLQR (Fuzzy Logic-based Quadratic Regulator) and tuning it using TLBO (Teaching-Learning-Based Optimization). The aim was to improve the motion control of a system consisting of multiple links, and the results clearly demonstrate the differences between the initial motion parameters and the optimized parameters.

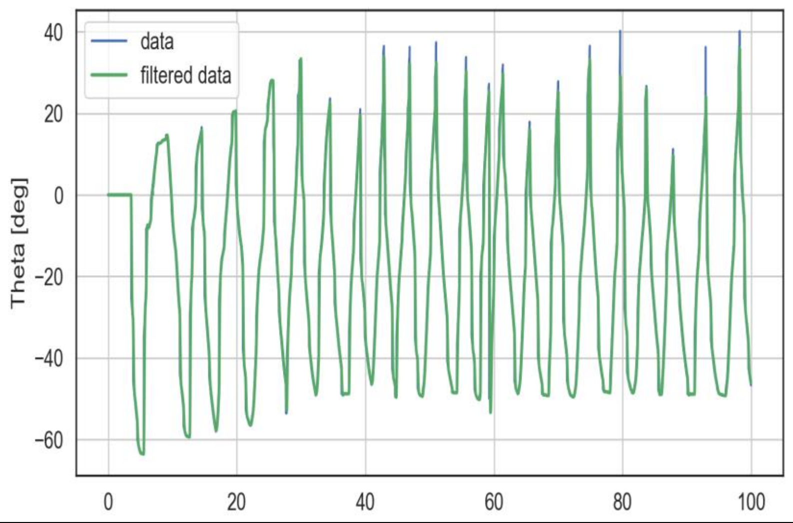
To evaluate the best movement parameters for each link, the enhanced parameters were applied to the system using a STM32 microcontroller. This involved programming the microcontroller with the optimized parameters and using it to control the motion of the system. The STM32 microcontroller is a powerful and versatile platform that is widely used in control systems and robotics applications.

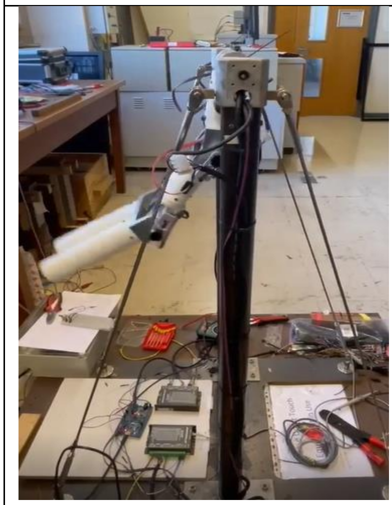
The combination of FLQR and TLBO allowed for the optimization of the control parameters of the system. The initial motion parameters were used as a starting point, and FLQR was used to design the optimal controller for the system. TLBO was then used to fine-tune the parameters of the controller to achieve the best possible performance.

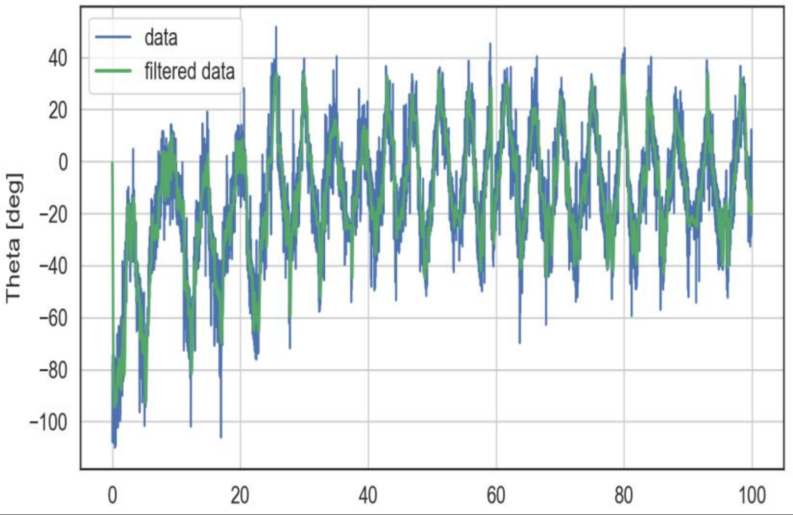
The optimized outcomes were then evaluated by applying the enhanced parameters to the system using the STM32 microcontroller. The results demonstrated significant improvements in the performance of the system, including improved accuracy, stability, and speed. The differences between the initial motion and optimized parameters were clearly demonstrated through this approach.

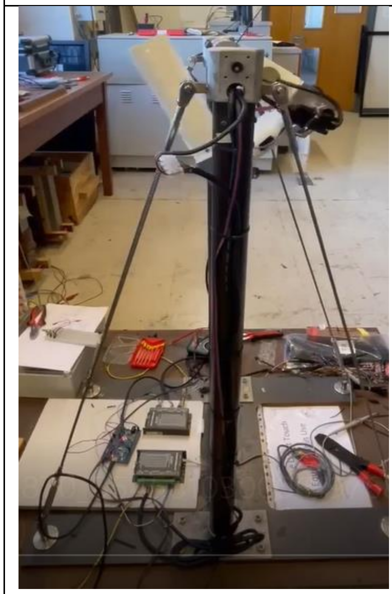
In conclusion, the implementation of FLQR and tuning it using TLBO provided a powerful approach to optimising the control parameters of a system consisting of multiple links. The STM32 microcontroller was used to apply the enhanced parameters to the system and evaluate the performance improvements. This approach can be applied to a wide range of control systems and robotics applications to improve their performance and achieve optimized outcomes.











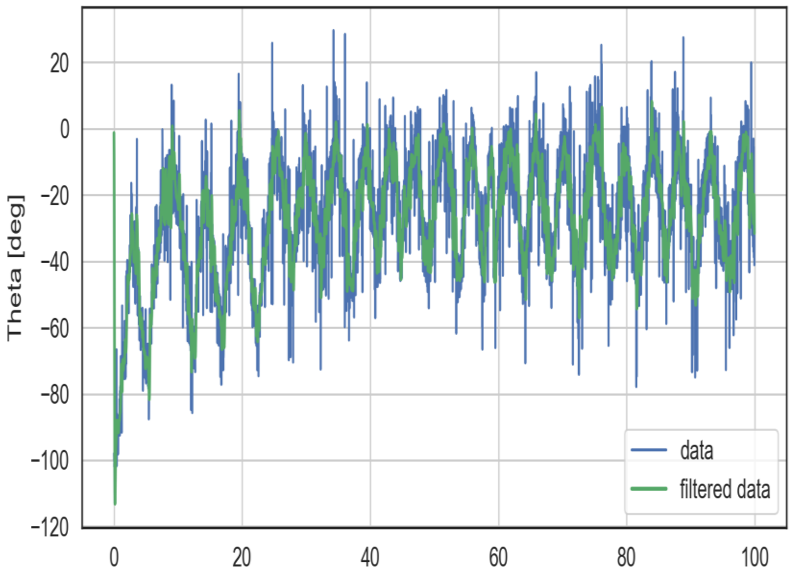


Figure 6.7 Enhanced motion of triple link system (Robogymnast)

Table 6.3 Optimized Robogymnast Motion Results

| **Symbol** | **Parameters** | **Average (Degrees)** | **Max Point (Degrees)** |
|---|---|---|---|
| $\boldsymbol{\theta 1}$ | Link 1 (free rotate) upper | (-45) to 20 | (−60) to 40 |
| $\boldsymbol{\theta 2}$ | Link 2 (Motor 1) middle | (−40) to 30 | (−110) to 45 |
| $\boldsymbol{\theta 3}$ | Link 3 (Motor 2) lower | (−50) to 20 | (−115) to 35 |

Table 6.3 outlines the physical motion of Robogymnast, a triple link robotic system presents the physical motion of triple-link robotic system. The table includes three symbols, namely $\theta_1$, $\theta_2$, and $\theta_3$, which indicate the angles of motion in degrees of the system. The parameters provided in the table offer insight into the physical properties of each link in the system, including the average motion and the maximum point that can be reached. These parameters play a crucial role in understanding the overall geometry and behaviour of the mechanical system.

Figure 6.7 and Table 6.3 provides information about the range of motion for each link in a three-link robotic system, along with details on the type of actuation used for each link. Specifically, the first link is a free-rotate joint, while the second and third links are controlled using motors. The average and maximum points for each link's range of motion are provided in degrees, which gives an idea of the flexibility and precision of each link's motion. For example, link 1 has an average range of motion from -45 to 20 degrees and a maximum range from -60 to 40 degrees. This suggests that link 1 has a fair amount of flexibility and can move through a fairly wide range of motion, with a power of 5V.

In contrast to the previous data, link 2 exhibits an average range of motion from -40 to 30 degrees and a maximum range from -110 to 45 degrees. These values indicate that link 2 possesses greater flexibility compared to the values presented in the previous table, with a larger maximum range. This observation implies that the motor governing link 2 may offer enhanced control precision in managing its motion as compared to the data presented in the previous table. Lastly, link 3 demonstrates an average range of motion from -50 to 20 degrees and a maximum range from -115 to 35 degrees, which is consistent with the values provided in the previous table

Overall, this table provides similar valuable information for understanding the capabilities and limitations of a three-link robotic system. Nevertheless, the augmented range of motion observed in link 2 implies that the system potentially possesses the ability to execute movements of greater complexity and precision. This information can be useful in designing and programming the system for specific applications, as well as in troubleshooting and diagnosing any issues that may arise during operation.

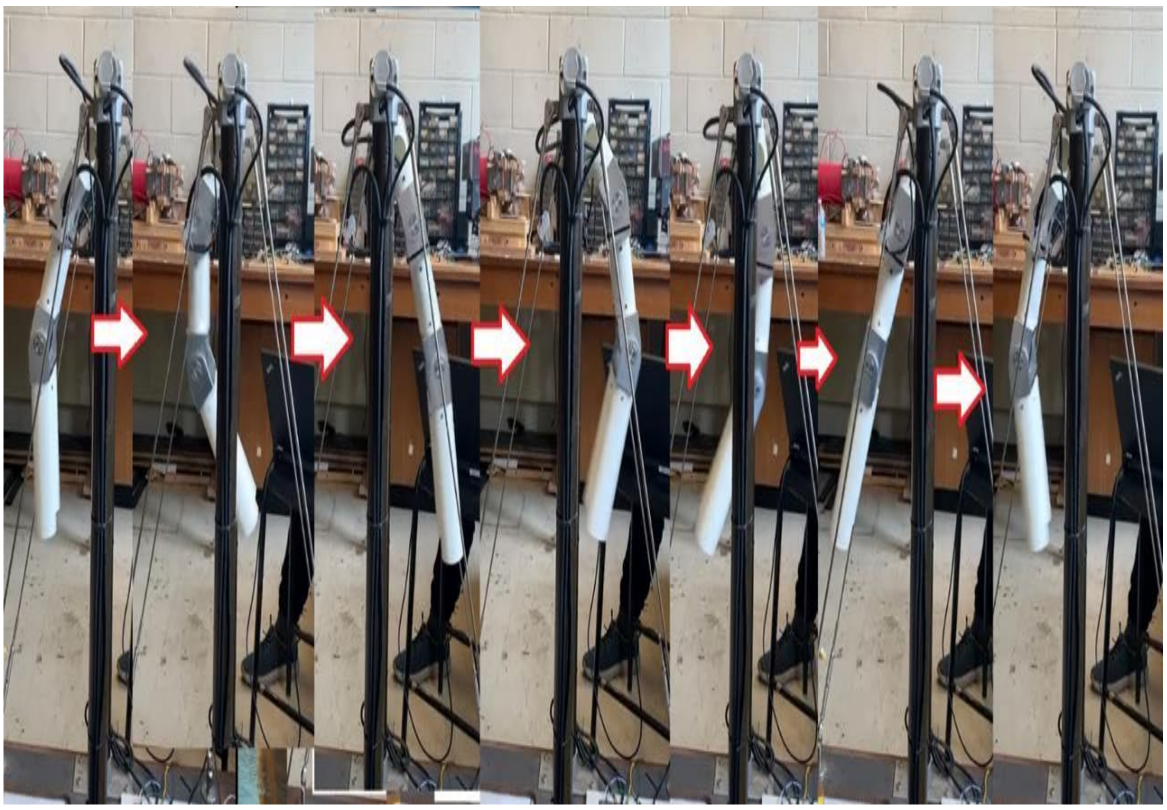


Figure 6.8 Robogymnast motion

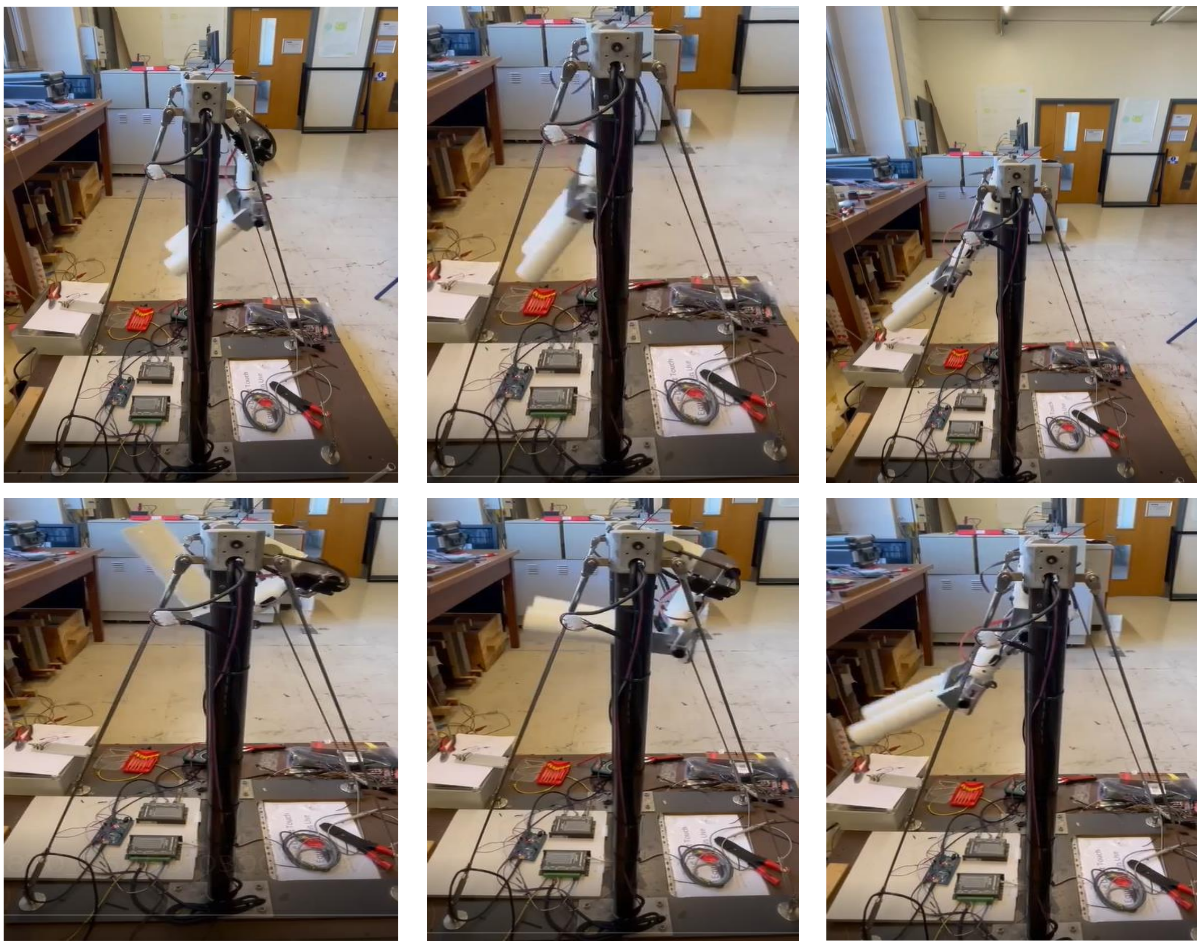


Figure 6.9 Side view of the Robogymnast motion

Figures 6.8 and 6.9 depict the synchronized motion of the Robogymnast system, which is an essential aspect of its performance. The figures show the movement of the robot's three links in a coordinated and synchronized manner. The synchronized motion of the Robogymnast system is crucial for its successful operation, as it enables the robot to perform complex movements and tasks accurately and efficiently. This synchronization is achieved through meticulous design and control of the robot's joints and actuators, underscoring its significance in enabling optimal system performance.

The synchronized motion of the Robogymnast system is a complex process that requires careful design and control. The system's joints and actuators must be precisely tuned and coordinated to ensure that the system moves smoothly and efficiently. In summary, Figures 6.8 and 6.9 serve as invaluable resources for examining and studying the synchronized motion of the Robogymnast system, providing essential tools for its analysis and comprehension.. The figures provide a visual representation of the robot's movement, allowing researchers and designers to gain insights into the performance of the system and identify areas for improvement. The synchronized motion of the Robogymnast system is a critical aspect of its operation and requires careful design and control to ensure accurate and efficient movement.

Table 6.4 provides valuable insight into the behaviour of a triple robotic system during movement. The data presented in the table gives information about the position and angles of the system at various points in time. This data can be used to analyse the movement of the system, identifying patterns or trends that can help optimize the design and performance of the robot. It is important to note that the data presented in the table is a random sample of a larger dataset. This means that the information contained in the table is representative of only a portion of the overall dataset. However, even with this limitation, the table provides valuable information about the behaviour of the robotic system during motion.

Table 6.4 Random samples of triple link motion in degrees

| **No.** | **Time (s)** | **Link1 ($\theta_1$)** | **Link2 ($\theta_2$)** | **Link3 ($\theta_3$)** |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 |
| 2 | 5 | -23.1 | -76.31 | -94.35 |
| 3 | 10 | -5.91 | -59.32 | -43.1 |
| 4 | 15 | -16.15 | -13.14 | -38.58 |
| 5 | 20 | -1.11 | 8.25 | -22.29 |
| 6 | 25 | 33.31 | 34.16 | 0.9 |
| 7 | 30 | -35.65 | -16.64 | 0.03 |
| 8 | 35 | -16.81 | -22.31 | -18.82 |
| 9 | 40 | -10.28 | -18.13 | -18.33 |
| 10 | 45 | 1.81 | 29.2 | -5.87 |
| 12 | 50 | 3.75 | 26.52 | -33.83 |
| 13 | 55 | -45.61 | -24.16 | -53.4 |
| 14 | 60 | -2.9 | -3.2 | -14.43 |
| 15 | 65 | 10.92 | 16.55 | -11.94 |
| 16 | 70 | 11.48 | -1.59 | -3.92 |
| 17 | 75 | -39.05 | 2.34 | -14.92 |
| 18 | 80 | 21.5 | 6.93 | 0.08 |
| 19 | 85 | -25.13 | 10.82 | -21.04 |
| 20 | 90 | 21.5 | 6.93 | 0.08 |

## 6.5 Comparison between simulation and real-time data of Robogymnast

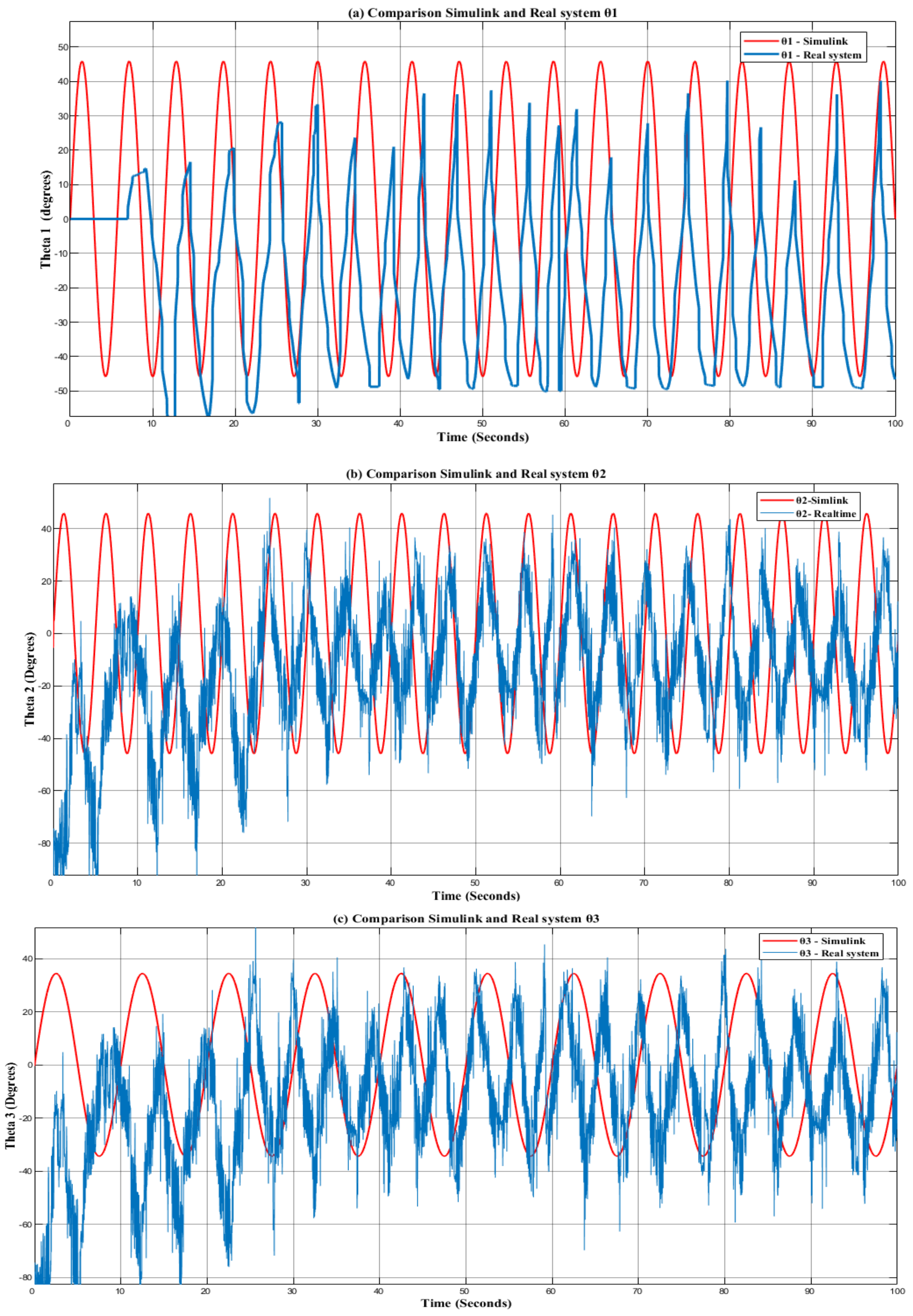


Figure 6.10 (a) Comparison of Simulink and real system of Robogymnast for ($\theta_1$); (b) comparison of Simulink and real system of Robogymnast ($\theta_2$); (c) comparison of Simulink and real system of Robogymnast ($\theta_3$)

Figure 6.10 presents a comparison between the Simulink model and the real system of Robogymnast for the three symbols $\theta_1$, $\theta_2$ and $\theta_3$. Each subplot shows the comparison for one of the symbols.

Subplot (a) presents the comparison of Simulink and real system for $\theta_1$. The red line represents the output from the Simulink model, while the blue line represents the output from the real system. The plot shows that there is a close match between the simulated and real-time data, with only minor deviations between the two. This indicates that the Simulink model accurately predicts the behaviour of the real system for $\theta_1$.

Subplot (b) presents the comparison of Simulink and the real system for $\theta_2$. The Figure shows a larger deviation between the simulated and real-time data compared to subplot (a). However, the overall trend of the two lines is still similar, with both lines showing a similar pattern of oscillation. This indicates that the Simulink model provides a reasonably accurate representation of the behaviour of the real system for $\theta_2$.

Subplot (c) presents the comparison of Simulink and the real system for $\theta_3$. The Illustration shows a larger deviation between the simulated and real-time data compared to subplots (a) and (b). However, the general pattern of the two lines is still similar, with both lines showing a similar trend of oscillation. This indicates that the Simulink model provides a reasonably accurate representation of the behaviour of the real system for $\theta_3$.

In summary, Figure 6.10 shows the Simulink model is an accurate representation of the behaviour of the real system for all three symbols $\theta_1$, $\theta_2$, and $\theta_3$. Although the accuracy varies between the symbols. This comparison study between the simulated and real-time data of Robogymnast indicates that combining simulation and real-world testing can provide a comprehensive understanding of the robot's performance.

According to this section, Simscap is a simulation tool in MATLAB Simulink that allows modelling and simulating physical systems, including mechanical, electrical, and robotic systems. Real-time data, on the other hand, refers to the actual data collected from potentiometers or measurements in a physical system. In the context of a triple-link robotic system, Simscap in MATLAB Simulink can be used to create a model of the robot's dynamics and kinematics, including the motors, gearboxes, and other mechanical components. Simulations can then be run to study the behaviour of the system under different conditions and inputs.

The simulation model was able to analyse the factors that affect the performance of Robogymnast, such as the speed of the motors and the position of the sensors. These factors were then tested in the real system of Robogymnast to determine the accuracy of the simulation model. The comparison of the simulated and real-time data demonstrated that by combining the use of Simulink and the real system of Robogymnast, it is possible to gain a better understanding of the behaviour of the robot and optimize its performance. The simulation model allows for testing and analysis in a controlled environment before conducting physical tests on the real robot, which can save time and resources.

To conclude, the study has demonstrated that Simulink serves as a proficient tool for predicting the behaviour of Robogymnast. Furthermore, integrating simulation with real-world testing offers a holistic comprehension of the robot's performance, thereby providing valuable insights into its capabilities and limitations.

The results can be used to optimize the design and performance of the robot for future applications such as healthcare or industrial robotics. Both Simscap and real-time data are valuable tools for studying and improving the performance of the robotic system. Simulations can optimize the design and control of the robot while real-time data can validate and improve the model's accuracy. Overall, the comparison study found a close match between the simulated and real-time data of Robogymnast, indicating the accuracy of the Simulink model for predicting the robot's performance in various scenarios.

### 6.5.1 Convergence between Simulation and Experimental of triple link system

Convergence percentage is a measure of how well a simulation in Simulink matches the behaviour of a real-time system. The percentage represents the degree of similarity between the simulation results and the actual experimental or real-time system results. A higher percentage indicates a closer match, while a lower percentage suggests a greater discrepancy between the simulation and real-time system results. Figure 6.11 illustrates Convergence percentage between Simulation and Experimental of triple link system.

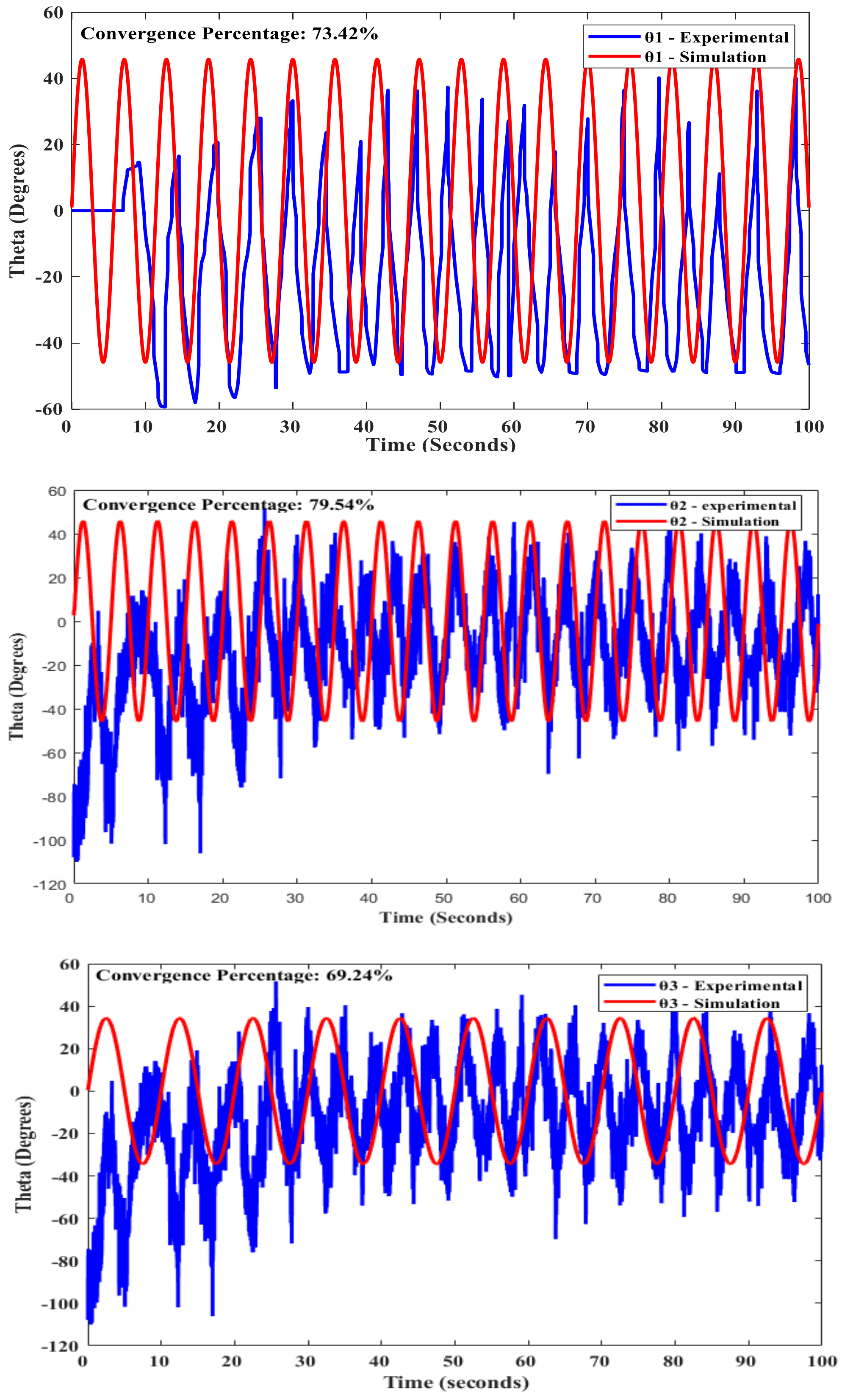


Figure 6.11 Convergence percentage between Simulation and Experimental of triple link system

Table 6.5 Convergence and error percentages for each joint angle

| **Symbol** | **convergence percentage** | **Error percentage** |
|---|---|---|
| $\theta_1$ | 73.42% | 26.58% |
| $\theta_2$ | 79.54% | 20.46% |
| $\theta_3$ | 69.24% | 30.76% |

Table 6.5 displays the convergence percentages for the joint angles of the triple-link robotic system, which can be used to evaluate the accuracy of the Simulink section. The convergence percentages for joints $\theta_1$ and $\theta_2$ surpass 70%, and the error percentage is lower than 30% on average. This implies that the Simulink simulation closely approximates thebehaviourr of the real-time system for these joints. Thus, these simulation results can be considered sufficiently reliable and accurate for the system.

However, the convergence percentage for joint $\theta_3$ is lower, at 69.24%, indicating that the Simulink simulation may be influenced by certain factors such as mass and actuator. These factors may cause deviations in the simulation from the actual system behaviour, leading to less accurate outcomes. Furthermore, it is important to note that convergence percentages may vary depending on the specific experimental setup and conditions.

In conclusion, convergence percentages provide a useful measure of simulation accuracy, but other factors such as the quality of the physical model and data input must also be considered. The convergence percentages for the triple link robotic system suggest that the Simulink simulation is relatively accurate for joints $\theta_1$ and $\theta_2$, but less accurate for joint $\theta_3$.

## 6.6 Summary

This chapter discusses the development of motion planning for a triple-link robotic system, comparing a Simscape MATLAB model with a real-world Robogymnast system. The proposed motion planning procedure integrates inverse kinematics and trajectory planning techniques to generate optimized motion trajectories for the system, and the algorithm parameters are evaluated using FLQR-TLBO on a practical triple-link robotic system. The study contributes to the field of robotics by presenting a comprehensive approach to motion planning for a triple-link robotic system and provides insights into the effectiveness of optimization techniques in improving the performance of motion planning algorithms. The study concludes that Simulink is an effective tool for predicting the behaviour of Robogymnast, and both Simscape and real-time data are valuable tools for studying and improving the performance of the robotic system. The convergence percentages for each joint angle suggest that the Simulink simulation is relatively accurate for joints $\theta_1$ and $\theta_2$, but less accurate for joint $\theta_3$. The findings of this study can inform the design and implementation of motion planning algorithms for robotic systems in industrial applications.

# Chapter 7:

# Conclusions, Limitations, and future work

This chapter provides a summary of the research and outlines some limitations and future work. The research has successfully addressed the objectives set out in the beginning, and the results have demonstrated the effectiveness and feasibility of the proposed approach.

## 7.1 Conclusions

In conclusion, all objectives of this work are met, where the main goal of the research was to study and investigate the swing-up control problem of the 3-DoF robot (Robogymnast). In order to obtain the best dynamic performance, several steps that were taken are given below:

- In terms of Controller:

The research presented modelling and simulation for the application of an LQR/fuzzy logic controller to stabilise a robotic gymnast system in MATLAB/Simulink. In this study, an FLQR was developed and then compared against an established LQR control approach for the Robogymnast. Mathematical modelling was performed for the starting values of variables within the pendulum-based system, and then a comprehensive model for the simulation of a robotic manipulation drive with the FLQR controller was developed. The primary variables were established, with calculations for overshoot as well as settle and rise times. An assessment was performed for the dynamic performance of the system. Calculations for stability and robustness were performed for each of the FLQR and LQR controllers, with comparisons across each scenario, in which variables were increased or decreased by several values. The results of the comparison show that the proposed FLQR controller performs better with the Robogymnast than the established LQR controller. To conclude, it is possible to declare that this work involved the investigation of the modulation of a triple-link robotics mechanism for the swing-up position, and the selected controller can be further extended to implement optimized algorithms for additional studies, this study examines control of swing-up in three-link robotic systems suggests that the proposed controller is effective.

- With regards to optimisation:

To summarize, this study implemented a Robogymnast system using an LQR/fuzzy logic controller (FLQR), and the system was improved using various algorithms in

MATLAB/SIMULINK to optimize its performance. The objective was to develop an FLQR controller and evaluate its effectiveness in the Robogymnast system. The simulation model included the FLQR controller, and primary system parameters were analysed, such as rise time, settling time, and overshoot. The proposed optimization method reduced parameters like ITAE to near-zero levels for link 2 and 3 and 1.688 p.u. for other parameters. Most importantly, it reduced the time and overshoot to zero in all cases. Additionally, the system's dynamic performance was evaluated. The results demonstrate that the TLBO algorithm with FLQR controller is more appropriate for positioning the robotic system than conventional controllers and other algorithms.

- In relation to the experiment of the proposed system:

In this work, a planned movement for swing-up in a 3-link inverted pendulum-based robot has been presented. The system's setup has been described, including the connection of each component of the system. Furthermore, this study provides a detailed investigation of motion planning development for a triple-link robotic system, with a focus on comparing a Simscape MATLAB model with a real-world Robogymnast system. The triple-link robotic system is widely renowned in industrial applications for its exceptional precision in executing intricate tasks. The motion planning approach proposed in this study combines inverse kinematics and trajectory planning techniques to generate optimized motion trajectories for the system. The algorithm's parameters are subsequently implemented using FLQR-TLBO on an operational triple-link robotic system to evaluate the approach's effectiveness. The thesis introduces a comprehensive methodology for motion planning in triple-link robotic systems, making it a significant contribution to the field of robotics. The findings have the potential to inform the design and execution of motion planning algorithms for robotic systems in industrial settings. Moreover, the research highlights the effectiveness of optimization techniques in improving the performance of motion planning algorithms. The study implements an FLQR controller fused with fuzzy logic to enhance the stability, time response, and efficiency of the Robogymnast system. The study compares the FLQR controller and TLBO algorithm with conventional controllers and other optimization algorithms, showing their superior performance in terms of performance metrics. Simulation modelling is used to analyse the system's major parameters and dynamic system performance, the convergence percentages for joints $\theta_1$ and $\theta_2$ exceed 70%, indicating that the Simulink simulation provides accurate results for these joints. However, the convergence percentage for joint $\theta_3$ is lower at 69.24%, which may suggest that the simulation is affected by certain factors such as mass and actuator. The research provides valuable insights into the design and optimization of robotic systems using advanced control techniques and optimization algorithms.

This work presents significant contributions to robotics, including the development of a robust hybrid linear quadratic regulator fuzzy controller for positioning control in the Robogymnast system. The implementation of the Teaching Learning Based Optimization (TLBO) algorithm optimizes controller parameters, improving system performance. Furthermore, the successful utilization of stepper motors enables smooth motion in the system. These findings advance robotics through innovative control strategies, optimization techniques, and motor selection for improved performance in the Robogymnast system.

## 7.2 Limitation

One of the limitations of triple-link robotic systems is their potential instability when carrying out complex tasks or operating at high speeds. For example, when a triple-link robotic system operates at speeds beyond a certain threshold, such as several meters per second or rotations per minute, it may experience issues with dynamic stability. The system's mechanical components, control algorithms, and feedback loops may struggle to respond accurately and quickly enough to maintain stability, leading to undesired vibrations, inaccuracies in motion, or even system instability. Additionally, as the number of links increases beyond three, their control algorithms can become increasingly complex. This complexity can make it more challenging to program and execute movements accurately. Finally, their mechanical design can make them more prone to wear and tear, which can affect their accuracy and performance over time.

Furthermore, stepper motors generally have lower power capabilities compared to other types of motors, such as AC induction motors or DC brushless motors. This lower power output may limit their ability to generate the necessary torque for effectively moving the pendulum. This is because their maximum rotational speed and torque are limited by the number of steps per revolution and the motor's design. Additionally, stepper motors can experience resonance and vibration at certain operating speeds, which can affect their accuracy and performance.

In addition to the mentioned limitations, another concern with stepper motors is the presence of backlash. Backlash refers to the play or clearance between the motor's rotor and the load it drives, resulting in a small amount of lost motion before the load starts to move. This can introduce inaccuracies and affect the precision of the system, particularly when changes in direction are involved.

Regarding assumptions, it's important to note that the limitations and concerns discussed are based on general observations and considerations for triple-link robotic systems. The specific performance and limitations of a particular robotic system depend on various factors such as the design, components used, control algorithms implemented, and the intended application. Therefore, it is essential to consider the specific details and specifications of the system in question when assessing its capabilities and limitations.

## 7.3 Future work

Following the analysis of the results described in this PhD thesis, this research may be further extended in future work based on the following points

1. One potential avenue to enhance the range of motion and stability of a triple-link pendulum system is to investigate the use of more powerful motors, such as servo motors or DC brushless motors, to drive the pendulum

2. As a future research direction, it is worth exploring the potential of incorporating sensors, such as inertial measurement units (IMUs) or force sensors, to enhance the accuracy and stability of the system.

3. Further advancement of the performance of triple-link pendulum systems could be achieved through the investigation of different advanced control techniques, such as adaptive control or optimal control.

4. The application of machine learning techniques, such as neural networks or reinforcement learning, could also potentially enhance the control and performance of 3-link pendulum systems.

5. Another possibility for improving the system is to incorporate machine learning algorithms to enable the robot to learn from its mistakes and enhance its performance over time. This could involve leveraging reinforcement learning or other techniques to optimize the robot's movements based on feedback from sensors and other source

## References

[1] A. Kivila, W. Book, and W. Singhose, “Modeling spatial multi-link flexible manipulator arms based on system modes,” *Int J Intell Robot Appl*, vol. 5, no. 3, pp. 300–312, Sep. 2021, doi: 10.1007/s41315-021-00201-3.

[2] H. A. Ismail, “Intelligent model-based control of complex multi-link mechanisms,” Cardiff University, 2016.

[3] C. Qingquan and N. Jing, “Design and Implementation of the Freedom Gymnastic Robot with the Ten DOF,” *Proceedings of the 31st Chinese Control and Decision Conference, CCDC 2019*, pp. 5194–5199, 2019, doi: 10.1109/CCDC.2019.8833029.

[4] H. G. Kamil, “Intelligent model-based control of complex three-link mechanisms,” Cardiff University, 2015.

[5] Z. Ben Hazem, M. J. Fotuhi, and Z. Bingül, “Development of a Fuzzy-LQR and Fuzzy-LQG stability control for a double link rotary inverted pendulum,” *J Franklin Inst*, vol. 357, no. 15, pp. 10529–10556, 2020, doi: 10.1016/j.jfranklin.2020.08.030.

[6] A. Jain, A. Sharma, V. Jately, B. Azzopardi, and S. Choudhury, “Real-Time Swing-Up Control of Non-Linear Inverted Pendulum Using Lyapunov Based Optimized Fuzzy Logic Control,” *IEEE Access*, vol. 9, pp. 50715–50726, 2021, doi: 10.1109/ACCESS.2021.3058645.

[7] Z. Ben Hazem and Z. Bingül, “Comprehensive Review of Different Pendulum Structures in Engineering Applications,” *IEEE Access*, pp. 1–1, 2023, doi: 10.1109/ACCESS.2023.3269580.

[8] T. Gurriet, M. Mote, A. Singletary, P. Nilsson, E. Feron, and A. D. Ames, “A Scalable Safety Critical Control Framework for Nonlinear Systems,” *IEEE Access*, vol. 8, pp. 187249–187275, 2020, doi: 10.1109/ACCESS.2020.3025248.

[9] C. H. Chiu, “The Design and Implementation of a Wheeled Inverted Pendulum Using an Adaptive Output Recurrent Cerebellar Model Articulation Controller,” *Neural Comput Appl*, vol. 19, no. 8, pp. 1153–1164, 2010, doi: 10.1007/s00521-009-0335-2.

[10] V. I. Arnold, *Mathematical Methods of Classical Mechanics*, 2nd ed. New york: Springer New York., 1989.

[11] X. Xiong and Z. Wan, “The simulation of double inverted pendulum control based on particle swarm optimization LQR algorithm,” in *Proceedings 2010 IEEE International Conference on Software Engineering and Service Sciences, ICSESS 2010*, 2010. doi: 10.1109/ICSESS.2010.5552427.

[12] D. LIU and H. YAMAURA, “Giant Swing Motion Control of 3-link Gymnastic Robot with Friction around an Underactuated Joint,” *Journal of System Design and Dynamics*, vol. 5, no. 5, pp. 925–936, 2011, doi: 10.1299/jsdd.5.925.

[13] M. W. Spong, “Swing up control of the acrobot,” in *Proceedings - IEEE International Conference on Robotics and Automation,* 1994. doi: 10.1109/robot.1994.350934.

[14] I. Fantoni, R. Lozano, and M. W. Spong, “Energy based control of the Pendubot,” *IEEE Trans Automat Contr,* 2000, doi: 10.1109/9.847110.

[15] K. J. Åström and K. Furuta, “Swinging up a pendulum by energy control,” *Automatica,* 2000, doi: 10.1016/S0005-1098(99)00140-5.

[16] H. A. Ismail, E. E. Eldhukhri, and M. S. Packianather, “Invasive weed optimization of swing-up control parameters for robot gymnast,” in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM,* 2014. doi: 10.1109/AIM.2014.6878052.

[17] X. Lai, A. Zhang, J. She, and M. Wu, “Motion control of underactuated three-link gymnast robot based on combination of energy and posture,” *IET Control Theory and Applications,* 2011, doi: 10.1049/iet-cta.2010.0210.

[18] N. Muškinja and B. Tovornik, “Swinging up and stabilization of a real inverted pendulum,” *IEEE Transactions on Industrial Electronics,* 2006, doi: 10.1109/TIE.2006.870667.

[19] M. W. Spong, “The Swing Up Control Problem For The Acrobot,” *IEEE Control Syst,* 1995, doi: 10.1109/37.341864.

[20] G. A. Medrano-Cerda, E. E. Eldukhri, and M. Cetin, “Balancing and Attitude Control of Double and Triple Inverted Pendulums,” *Transactions of the Institute of Measurement and Control,* 1995, doi: 10.1177/014233129501700306.

[21] K. Lakshmi, “Design of Robust Energy Control for Cart-Inverted Pendulum,” *International Journal of Engineering and Technology,* vol. 4, no. 1, pp. 66–76, 2007.

[22] S. C. Brown and K. M. Passino, “Intelligent Control for an Acrobot,” *Journal of Intelligent and Robotic Systems: Theory and Applications,* 1997, doi: 10.1023/A:1007953809856.

[23] E. E. Eldukhri and D. T. Pham, “Autonomous swing-up control of a three-link robot gymnast,” *Proceedings of the Institution of Mechanical Engineers. Part I: Journal of Systems and Control Engineering,* 2010, doi: 10.1243/09596518JSCE948.

[24] M. W. Spong and D. J. Block, “Pendubot: a mechatronic system for control research and education,” in *Proceedings of the IEEE Conference on Decision and Control,* 1995. doi: 10.1109/cdc.1995.478951.

[25] L. Wang, “Application to Inverted Pendulum Tracking :,” *October,* vol. 26, no. 5, 1996.

[26] F. Cheng, G. Zhong, Y. Li, and Z. Xu, “Fuzzy control of a double-inverted pendulum,” *Fuzzy Sets Syst,* vol. 79, no. 3, pp. 315–321, 1996, doi: 10.1016/0165-0114(95)00156-5.

[27] K. G. Eltohamy and C. Y. Kuo, “Nonlinear optimal control of a triple link inverted pendulum with single control input,” *Int J Control*, 1998, doi: 10.1080/002071798222811.

[28] J. Aracil, F. Gordillo, and J. A. Acosta, “Stabilization of oscillations in the inverted pendulum,” *IFAC Proceedings Volumes (IFAC-PapersOnline)*, vol. 35, no. 1, pp. 79–84, 2002, doi: 10.3182/20020721-6-es-1901.00263.

[29] A. N. K. Nasir, R. M. T. Raja Ismail, and M. A. Ahmad, “Performance comparison between sliding mode control (SMC) and PD-PID controllers for a nonlinear inverted pendulum system,” *World Acad Sci Eng Technol*, vol. 46, pp. 400–405, 2010.

[30] S. Kizir, Z. Bingul, and C. Oysu, “Fuzzy control of a real time inverted pendulum system,” *Journal of Intelligent and Fuzzy Systems*, vol. 21, no. 1–2, pp. 121–133, 2010, doi: 10.3233/IFS-2010-0441.

[31] J. Zhang and W. Zhang, “LQR self-adjusting based control for the planar double inverted pendulum,” *Phys Procedia*, vol. 24, pp. 1669–1676, 2012, doi: 10.1016/j.phpro.2012.02.246.

[32] B. Li, “Rotational double inverted pendulum,” pp. 1–57, 2013, [Online]. Available: https://pdfs.semanticscholar.org/4bea/c0f9b4abe3285bd3f6d29b8c9984904711ac.pdf

[33] T. Glück, A. Eder, and A. Kugi, “Swing-up control of a triple pendulum on a cart with experimental validation,” *Automatica*, 2013, doi: 10.1016/j.automatica.2012.12.006.

[34] X. Yang and X. Zheng, “Swing-Up and Stabilization Control Design for an Underactuated Rotary Inverted Pendulum System: Theory and Experiments,” *IEEE Transactions on Industrial Electronics*, vol. 65, no. 9, pp. 7229–7238, 2018, doi: 10.1109/TIE.2018.2793214.

[35] E. Susanto, A. Surya Wibowo, and E. Ghiffary Rachman, “Fuzzy Swing Up Control and Optimal State Feedback Stabilization for Self-Erecting Inverted Pendulum,” *IEEE Access*, vol. 8, pp. 6496–6504, 2020, doi: 10.1109/ACCESS.2019.2963399.

[36] M. Waszak and R. Langowski, “An Automatic Self-Tuning Control System Design for an Inverted Pendulum,” *IEEE Access*, vol. 8, pp. 26726–26738, 2020, doi: 10.1109/ACCESS.2020.2971788.

[37] J. Rubí, A. Rubio, and A. Avello, “Swing-up control problem for a self-erecting double inverted pendulum,” *IEE Proceedings: Control Theory and Applications*, 2002, doi: 10.1049/ip-cta:20020326.

[38] K. Furuta, M. Yamakita, and S. Kobayashi, “Swing-up Control of Inverted Pendulum Using Pseudo-State Feedback,” *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 206, no. 4, pp. 263–269, 1992, doi: 10.1243/pime_proc_1992_206_341_02.

[39] M. Yamakita, K. Monaka, and K. Furuta, “Swing Up Control od a Double Pendulum,” in *Proceedings of the American Control Conference*, 1993. doi: 10.1109/acc.1995.529255.

[40] M. Yamakita, M. Iwashiro, Y. Sugahara, and K. Furuta, “Robust swing up control of double pendulum,” *Proceedings of the American Control Conference*, vol. 1, pp. 290–295, 1995, doi: 10.1109/acc.1995.529255.

[41] S. Yasunobu and M. Mori, “Swing up fuzzy controller for inverted pendulum based on a human control strategy,” in *Proceedings of 6th International Fuzzy Systems Conference*, 1997, pp. 1621–1625.

[42] F. Gordillo, J. A. Acosta, and J. Aracil, “A new swing-up law for the Furuta pendulum,” *Int J Control*, 2003, doi: 10.1080/0020717031000116506.

[43] K. Graichen, M. Treuer, and M. Zeitz, “Swing-up of the double pendulum on a cart by feedforward and feedback control with experimental validation,” *Automatica*, 2007, doi: 10.1016/j.automatica.2006.07.023.

[44] X. Xin and M. Kaneda, “Swing-up control for a 3-DOF gymnastic robot with passive first joint: Design and analysis,” *IEEE Transactions on Robotics*, 2007, doi: 10.1109/TRO.2007.909805.

[45] S. C. Duong, H. Kinjo, E. Uezato, and T. Yamamoto, “Intelligent control of a three-DOF planar underactuated manipulator,” in *Proceedings of the 14th International Symposium on Artificial Life and Robotics, AROB 14th’09*, 2009. doi: 10.1007/s10015-009-0674-1.

[46] F. Xue, Z. Hou, Y. Hu, and C. Liu, “Human simulated intelligence motion control for a three-link acrobot,” in *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, 2011. doi: 10.1109/WCICA.2011.5970634.

[47] P. Jaiwat and T. Ohtsuka, “Real-Time Swing-up of Double Inverted Pendulum by Nonlinear Model Predictive Control,” *5th International Symposium on Advanced Control of Industrial Processes*, pp. 290–295, 2014.

[48] H. G. Kamil and A. A. Ahmed, “Tuning of Control Motion for a three link robot manipulator using Particle Swarm Optimization Technique,” vol. 15, no. 4, pp. 102–110, 2017.

[49] T. Henmi, M. Akiyama, and T. Yamamoto, “Motion control of underactuated linkage robot based on gymnastic skill,” *Electrical Engineering in Japan (English translation of Denki Gakkai Ronbunshi)*, vol. 206, no. 1, pp. 42–50, 2019, doi: 10.1002/eej.23142.

[50] N. P. Nguyen, H. Oh, Y. Kim, and J. Moon, “A nonlinear hybrid controller for swinging-up and stabilizing the rotary inverted pendulum,” *Nonlinear Dyn*, vol. 104, no. 2, pp. 1117–1137, 2021, doi: 10.1007/s11071-021-06317-2.

[51] Y. Nishiki, H. Kajiwara, and M. Aoyagi, “Control of swing-up and giant-swing motions of Acrobot based on periodic input,” *Nonlinear Dyn*, vol. 108, no. 3, pp. 2297–2308, 2022, doi: 10.1007/s11071-022-07312-x.

[52] S. Zeghlache, M. Z. Ghellab, A. Djerioui, B. Bouderah, and M. F. Benkhoris, “Adaptive fuzzy fast terminal sliding mode control for inverted pendulum-cart system with actuator faults,” *Math Comput Simul*, vol. 210, pp. 207–234, Aug. 2023, doi: 10.1016/j.matcom.2023.03.005.

[53] N. Bu and X. Wang, “Swing-up design of double inverted pendulum by using passive control method based on operator theory,” *International Journal of Advanced Mechatronic Systems*, vol. 10, no. 1, p. 1, 2023, doi: 10.1504/IJAMECHS.2023.128154.

[54] L. Fan, A. Zhang, G. Pan, Y. Du, and J. Qiu, “Swing-up and fixed-time stabilization control of underactuated cart-double pendulum system,” *IET Control Theory & Applications*, vol. 17, no. 6, pp. 662–671, Apr. 2023, doi: 10.1049/cth2.12386.

[55] M. Z. Kolovsky, A. N. Evgrafov, A. Semenov, Yu, and V. Slousch, A, *Advanced Theory of Mechanisms and Machin*. 195197 st. Petersburg, Russia: Springer Science & Business Media, 2012. doi: 10.1007/978-3-540-46516-4.

[56] J. J. Uicker, G. R. Pennock, J. E. Shigley, and J. M. McCarthy, “Theory of Machines and Mechanisms,” *Journal of Mechanical Design*, 2003, doi: 10.1115/1.1605769.

[57] H. Cheng, H. Chen, B. Gao, and X. Zhang, “General swing-up methodology for the vertical three-link underactuated manipulator,” in *2013 IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems, IEEE-CYBER 2013*, IEEE conferane, 2013. doi: 10.1109/CYBER.2013.6705475.

[58] B. Mahboub and D. Stephen, “A Two-Link Robot Manipulator: Simulation and Control Design,” *International Journal of Robotic Engineering*, vol. 5, no. 2, 2020, doi: 10.35840/2631-5106/4128.

[59] L. N. López de Lacalle, A. Lamikiz, J. A. Sánchez, and I. Fernández de Bustos, “Simultaneous measurement of forces and machine tool position for diagnostic of machining tests,” *IEEE Trans Instrum Meas*, vol. 54, no. 6, pp. 2329–2335, 2005, doi: 10.1109/TIM.2005.858535.

[60] A. M. Lopes and J. A. Tenreiro Machado, “Dynamics of the N-link pendulum: a fractional perspective,” *Int J Control*, vol. 90, no. 6, pp. 1192–1200, 2017, doi: 10.1080/00207179.2015.1126677.

[61] H. Kamil, E. Eldukhri, and M. and Packianather, “Optimisation of swing-up control parameters for a robot gymnast using the Bees Algorithm,” in *Proceedings of 8th International Symposium on Intelligent and Manufacturing Systems (IMS 2012)*, Antalya, Turkey, 2012, pp. 456–466.

[62] H. G. Kamil, E. E. Eldukhri, and M. S. Packianather, “Balancing Control of Robot Gymnast Based on Discrete-Time Linear Quadratic Regulator Technique,” in

*Proceedings - 2nd International Conference on Artificial Intelligence, Modelling, and Simulation, AIMS 2014*, 2014, pp. 137–142. doi: 10.1109/AIMS.2014.38.

[63] H. A. Ismail, M. S. Packianather, R. I. Grosvenor, and E. E. Eldhukri, "The application of IWO in LQR controller design for the Robogymnast," in *IntelliSys 2015 - Proceedings of 2015 SAI Intelligent Systems Conference*, 2015. doi: 10.1109/IntelliSys.2015.7361154.

[64] H. A. Ismail, M. S. Packianather, and R. I. Grosvenor, "Multi-objective invasive weed optimization of the LQR controller," *International Journal of Automation and Computing*, 2017, doi: 10.1007/s11633-017-1061-3.

[65] A. Chemori, "Control of complex robotic systems: Challenges, design and experiments," *2017 22nd International Conference on Methods and Models in Automation and Robotics, MMAR 2017*, pp. 622–631, 2017, doi: 10.1109/MMAR.2017.8046900.

[66] K. Manickavelan, B. Singh, and N. Sellappan, "Design, Fabrication and Analysis of Four Bar Walking Machine Based on Chebyshev's Parallel Motion Mechanism.," *European International Journal of Science and Technology*, vol. 3, no. 8, pp. 65–73, 2014.

[67] R. Herguedas, G. Lopez-Nicolas, R. Aragues, and C. Sagues, "Survey on multi-robot manipulation of deformable objects," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE, Sep. 2019, pp. 977–984. doi: 10.1109/ETFA.2019.8868987.

[68] C. Sun, H. Gao, W. He, and Y. Yu, "Fuzzy Neural Network Control of a Flexible Robotic Manipulator Using Assumed Mode Method," *IEEE Trans Industr Inform*, vol. 15, no. 2, pp. 5214–5227, 2018, doi: 10.1109/TII.2018.2818120.

[69] P. Nordfeldt and T. Hägglund, "Decoupler and PID controller design of TITO systems," *J Process Control*, vol. 16, no. 9, pp. 923–936, 2006, doi: 10.1016/j.jprocont.2006.06.002.

[70] E. Vinodh Kumar and J. Jerome, "Robust LQR controller design for stabilizing and trajectory tracking of inverted pendulum," *Procedia Eng*, vol. 64, pp. 169–178, 2013, doi: 10.1016/j.proeng.2013.09.088.

[71] Z. Ben Hazem, M. J. Fotuhi, and Z. Bingül, "Development of a Fuzzy-LQR and Fuzzy-LQG stability control for a double link rotary inverted pendulum," *J Franklin Inst*, vol. 357, no. 15, pp. 10529–10556, 2020, doi: 10.1016/j.jfranklin.2020.08.030.

[72] H. G. Kamil, E. E. Eldukhri, and M. S. Packianather, "Balancing Control of Robot Gymnast Based on Discrete-Time Linear Quadratic Regulator Technique," *Proceedings - 2nd International Conference on Artificial Intelligence, Modelling, and Simulation, AIMS 2014*, pp. 137–142, 2014, doi: 10.1109/AIMS.2014.38.

*References*

[73] R. Kelly, “PD control with desired gravity compensation of robotic manipulators: A review,” *International Journal of Robotics Research,* vol. 16, no. 5, pp. 660–672, 1997, doi: 10.1177/027836499701600505.

[74] E. A. Alandoli and T. S. Lee, “A Critical Review of Control Techniques for Flexible and Rigid Link Manipulators,” *Robotica*, vol. 38, no. 12, pp. 2239–2265, 2020, doi: 10.1017/S0263574720000223.

[75] M. R. H. Mohd Adnan, A. Sarkheyli, A. Mohd Zain, and H. Haron, “Fuzzy logic for modeling machining process: a review,” *Artif Intell Rev*, vol. 43, no. 3, pp. 345–379, 2015, doi: 10.1007/s10462-012-9381-8.

[76] R. Kumar and M. Kumar, “Improvement power system stability using Unified Power Flow Controller based on hybrid Fuzzy Logic-PID tuning in SMIB system,” *Proceedings of the 2015 International Conference on Green Computing and Internet of Things, ICGCIoT 2015*, pp. 815–819, 2016, doi: 10.1109/ICGCIoT.2015.7380575.

[77] Z. Bingül and O. Karahan, “A Fuzzy Logic Controller tuned with PSO for 2 DOF robot trajectory control,” *Expert Syst Appl*, vol. 38, no. 1, pp. 1017–1031, 2011, doi: 10.1016/j.eswa.2010.07.131.

[78] N. M. T. Luma and A. O. Yaseen, “Fast Training Algorithms for Feed Forward Neural Network,” *Ibn Al-Haitham Journal for Pure & Applied Science*, vol. 26, no. 1, pp. 275–280, 2013.

[79] F. Piltan and N. B. Sulaiman, “Review of sliding mode control of robotic manipulator,” *World Appl Sci J*, vol. 18, no. 12, pp. 1855–1869, 2012, doi: 10.5829/idosi.wasj.2012.18.12.208.

[80] K. K. D. Young, “Controller Design for a Manipulator Using Theory of Variable Structure Systems,” *IEEE Trans Syst Man Cybern*, vol. 8, no. 2, pp. 101–109, 1978, doi: 10.1109/TSMC.1978.4309907.

[81] L. Slotine, *APPLIED NONLINEAR CONTROL*. 1991.

[82] H. N. Iordanou and B. W. Surgenor, “Experimental Evaluation of the Robustness of Discrete Sliding Mode Control Versus Linear Quadratic Control,” *Brain Cogn*, vol. 32, no. 2, pp. 273–344, 1997, doi: 10.1006/brcg.1996.0066.

[83] M. Ertugrul and O. Kaynak, “Neural computation of the equivalent control in sliding mode for robot trajectory control applications,” *Proc IEEE Int Conf Robot Autom*, vol. 3, no. May, pp. 2042–2047, 1998, doi: 10.1109/ROBOT.1998.680617.

[84] E. Z. Taha and S. Kawaji, “Robust control of a constrained robot arm,” *1993 International Conference on Intelligent Robots and Systems*, pp. 91–96, 1993, doi: 10.1109/iros.1993.583085.

[85] A. Bansal and V. Sharma, “Design and Analysis of Robust H-infinity Controller,” *Control Theory and Informatics*, vol. 3, no. 2, pp. 7–14, 2013.

[86] A. M. Shawky, A. W. Ordys, L. Petropoulakis, and M. J. Grimble, “Position control of flexible manipulator using non-linear H ∞ with state-dependent Riccati equation,” *Proceedings of the Institution of Mechanical Engineers. Part I: Journal of Systems and Control Engineering*, vol. 221, no. 3, pp. 475–486, 2007, doi: 10.1243/09596518JSCE313.

[87] P. Jodouin, M. Saad, and R. Wamkeue, “Comparison of H∞ and loop transfer recover methods-application to an experimental arm,” in *17th IEEE Mediterranean Electrotechnical Conference, Beirut, Lebanon*, Beirut, Lebanon: IEEE, 2014, pp. 310–314.

[88] B. Altıner, A. Delibaşı, and B. Erol, “Modeling and control of flexible link manipulators for unmodeled dynamics effect,” *Proceedings of the Institution of Mechanical Engineers. Part I: Journal of Systems and Control Engineering*, vol. 233, no. 3, pp. 245–263, 2019, doi: 10.1177/0959651818791071.

[89] P. Axelsson, A. Helmersson, and M. Norrlöf, “H∞-controller design methods applied to one joint of a flexible industrial manipulator,” *IFAC Proceedings Volumes (IFAC-PapersOnline)*, vol. 19, pp. 210–216, 2014, doi: 10.3182/20140824-6-za-1003.00143.

[90] J. Hu, C. Bohn, and H. R. Wu, “Systematic H∞ weighting function selection and its application to the real-time control of a vertical take-off aircraft,” *Control Eng Pract*, vol. 8, no. 3, pp. 241–252, 2000, doi: 10.1016/s0967-0661(99)00157-4.

[91] X. Wang and D. Chen, “Output tracking control of a one-link flexible manipulator via causal inversion,” *IEEE Transactions on Control Systems Technology*, vol. 14, no. 1, pp. 141–148, 2006, doi: 10.1109/TCST.2005.859628.

[92] S. F. Atashzar, H. A. Talebi, M. J. Yazdanpanah, and F. Towhidkhah, “Tip position tracking of flexible-link manipulators based on online robust trajectory modification,” *IECON Proceedings (Industrial Electronics Conference)*, pp. 1651–1656, 2010, doi: 10.1109/IECON.2010.5675434.

[93] S.-E. Oltean, “Swing-up and Stabilization of the Rotational Inverted Pendulum Using PD and Fuzzy-PD Controllers,” *Procedia Technology*, vol. 12, pp. 57–64, 2014, doi: 10.1016/j.protcy.2013.12.456.

[94] M. Hasan, C. Saha, M. M. Rahman, M. R. I. Sarker, and S. K. Aditya, “Balancing of an Inverted Pendulum Using PD Controller,” *Dhaka University Journal of Science*, vol. 60, no. 1. pp. 115–120, 2012. doi: 10.3329/dujs.v60i1.10348.

[95] E. A. Alandoli, M. Z. A. Rashid, and M. Sulaiman, “A comparison of PID and LQR controllers for position tracking and vibration suppression of flexible link manipulator,” *J Theor Appl Inf Technol*, vol. 95, no. 13, pp. 2949–2955, 2017.

[96] L. Wang and Z. Sheng, “LQR-Fuzzy control for double inverted pendulum,” *Proceedings - 2010 International Conference on Digital Manufacturing and Automation, ICDMA 2010*, vol. 1, pp. 900–903, 2010, doi: 10.1109/ICDMA.2010.170.

[97] T. Yaren and S. Kizir, “Stabilization control of triple pendulum on a cart,” *2018 6th International Conference on Control Engineering and Information Technology, CEIT 2018*, no. October, pp. 25–27, 2018, doi: 10.1109/CEIT.2018.8751818.

[98] M. I. El-Hawwary, A. L. Elshafei, H. M. Emara, and H. A. A. Fattah, “Adaptive fuzzy control of the inverted pendulum problem,” *IEEE Transactions on Control Systems Technology*, vol. 14, no. 6, pp. 1135–1144, 2006, doi: 10.1109/TCST.2006.880217.

[99] D. T. Pham, M. Castellani, M. Sholedolu, and A. Ghanbarzadeh, “Bee Algorithm A Novel Approach to Function Optimisation,” no. September, pp. 1–37, 2005.

[100] D. O. F. Philosophy, “ENHANCING THE BEES ALGORITHM FOR GLOBAL OPTIMISATION USING SEARCH SPACE MANIPULATION by TURKI BIN BAKIR,” no. January, 2021.

[101] A. R. Mehrabian and C. Lucas, “A novel numerical optimization algorithm inspired from weed colonization,” *Ecol Inform*, 2006, doi: 10.1016/j.ecoinf.2006.07.003.

[102] R. K. Mandava and P. R. Vundavilli, “Implementation of modified chaotic invasive weed optimization algorithm for optimizing the PID controller of the biped robot,” *Sadhana - Academy Proceedings in Engineering Sciences*, vol. 43, no. 5, pp. 1–18, 2018, doi: 10.1007/s12046-018-0851-9.

[103] T. Dokeroglu, E. Sevinc, T. Kucukyilmaz, and A. Cosar, “A survey on new generation metaheuristic algorithms,” *Comput Ind Eng*, vol. 137, p. 106040, Nov. 2019, doi: 10.1016/j.cie.2019.106040.

[104] M. Dorigo and D. C. Gianni, “Ant Colony Optimization: A New Meta-Heuristic,” *In Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406)*. pp. 1470–1477, 1992.

[105] A. Akhtar, “Evolution of Ant Colony Optimization Algorithm -- A Brief Literature Review,” 2019.

[106] C. C. Wong, H. M. Feng, Y. C. Lai, and C. J. Yu, “Ant Colony Optimization and image model-based robot manipulator system for pick-and-place tasks,” *Journal of Intelligent and Fuzzy Systems*, vol. 36, no. 2, pp. 1083–1098, 2019, doi: 10.3233/JIFS-169883.

[107] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, “GSA: A Gravitational Search Algorithm,” *Inf Sci (N Y)*, vol. 179, no. 13, pp. 2232–2248, 2009, doi: 10.1016/j.ins.2009.03.004.

[108] N. M. Sabri, M. Puteh, and M. R. Mahmood, “A review of gravitational search algorithm,” *International Journal of Advances in Soft Computing and its Applications*, vol. 5, no. 3, 2013.

[109] P. K. Das, H. S. Behera, P. K. Jena, and B. K. Panigrahi, “Multi-robot path planning in a dynamic environment using improved gravitational search algorithm,” *Journal of Electrical Systems and Information Technology*, vol. 3, no. 2, pp. 295–313, 2016, doi: 10.1016/j.jesit.2015.12.003.

*References*

[110] R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and Multidisciplinary Optimization*. 2004. doi: 10.1007/s00158-003-0368-6.

[111] N. Gunantara, "A review of multi-objective optimization: Methods and its applications," *Cogent Eng*, vol. 5, no. 1, pp. 1–16, 2018, doi: 10.1080/23311916.2018.1502242.

[112] A. T. Tolmidis and L. Petrou, "Multi-objective optimization for dynamic task allocation in a multi-robot system," *Eng Appl Artif Intell*, vol. 26, no. 5–6, pp. 1458–1468, 2013, doi: 10.1016/j.engappai.2013.03.001.

[113] M. W. Spong, S. Hutchinson, and M. Vidyasagar, "Robot modeling and control," *IEEE Control Syst*, vol. 26, no. 6, pp. 113–115, 2006, doi: 10.1109/MCS.2006.252815.

[114] K. Furut, T. Ochiai, and N. Ono, "Attitude control of a triple inverted pendulum," *Int J Control*, 1984, doi: 10.1080/00207178408933251.

[115] C. T. Kiang, A. Spowage, and C. K. Yoong, "Review of Control and Sensor System of Flexible Manipulator," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 77, no. 1, pp. 187–213, 2015, doi: 10.1007/s10846-014-0071-4.

[116] ST Microelectronics, "DS6329, STM32F207xx," 2020. https://www.st.com/resource/en/datasheet/cd00237391.pdf

[117] A. Gmiterko and M. Grossman, "N-link inverted pendulum modeling," in *Recent Advances in Mechatronics 2008-2009*, 2009. doi: 10.2478/v10147-010-0043-z.

[118] A. Preumont, "Controllability and Observability," Springer-Verlag Berlin Heidelberg, 2011, pp. 275–297. doi: 10.1007/978-94-007-2033-6_12.

[119] C. N. Aithal, P. Ishwarya, S. Sneha, C. N. Yashvardhan, and K. V. Suresh, "Design of a Bipedal Robot," 2021. doi: 10.1007/978-981-16-0443-0_5.

[120] StepperOnline, "Closed-loop Geared Stepper L48mm Gear Raio 141-Encoder 1000CPR," 2020. https://www.omc-stepperonline.com/Nema-17-Closed-loop-Geared-Stepper-L48mm-Gear-Raio-141-Encoder-1000CPR.html (accessed Nov. 08, 2020).

[121] RS, "Sick Incremental Encoder 1024 ppr, Blind Hollow Shaft, DBS36E-BBAK01024," *Sick Incremental Encoder 1024 ppr, Blind Hollow Shaft, DBS36E-BBAK01024*, 2020.

[122] Faenell, "Rotary Potentiometer," *Rotary Potentiometer, SP22E-10K*, 2020. https://uk.farnell.com/eti-systems/sp22e-10k/potentiometer-1-turn-10k-ohm-1w/dp/1307135?ost=sp22e-10k&ICID=I-RS-STM7REC-0

[123] StepperOnline, "Closed Loop Stepper Driver 0-3.0A 24-48VDC for Nema 11, 14, 17 Stepper Motor," 2020. https://www.omc-stepperonline.com/closed-loop-stepper-driver-0-30a-24-48vdc-for-nema-11-14-17-stepper-motor (accessed Nov. 07, 2020).

[124] ST, "STM32F429 Discovery," *ST*, 2020. https://stm32f4-discovery.net/stm32f429-discovery/

[125] StepperOnline, “150W 48V 3.1A 115/230V Switching Power Supply Stepper Motor CNC Router Kits,” *StepperOnline,* 2020. https://www.omc-stepperonline.com/150w-48v-3-1a-115-230v-switching-power-supply-stepper-motor-cnc-router-kits.html

[126] DELL, “Latitude 5300 Laptop,” *Latitude 5300 Laptop,* 2020. https://www.dell.com/en-uk/work/shop/laptop-computers-for-businesses/latitude-5300-business-laptop/spd/latitude-13-5300-laptop

[127] A. Serrani, “Nonlinear control systems,” in *The Engineering Handbook, Second Edition*, 2004. doi: 10.1201/9781420037043.

[128] Z. Ben Hazem, M. J. Fotuhi, and Z. Bingül, “A Study of Anti-swing Fuzzy LQR Control of a Double Serial Link Rotary Pendulum,” *IETE J Res*, 2021, doi: 10.1080/03772063.2021.1911690.

[129] J. K. Adamu, M. F. Hamza, and A. I. Isa, “Performance Comparisons Of Hybrid Fuzzy-LQR And Hybrid PID-LQR Controllers On Stabilizing Double Rotary Inverted Pendulum,” *Journal of Applied Materials and Technology*, vol. 1, no. 2, pp. 71–80, 2020, doi: 10.31258/jamt.1.2.71-80.

[130] B. Bekkar and K. Ferkous, “Design of Online Fuzzy Tuning LQR Controller Applied to Rotary Single Inverted Pendulum: Experimental Validation,” *Arab J Sci Eng*, 2022, doi: 10.1007/s13369-022-06921-3.

[131] L. Wang and Z. Sheng, “LQR-Fuzzy control for double inverted pendulum,” *Proceedings - 2010 International Conference on Digital Manufacturing and Automation, ICDMA 2010*, vol. 1, pp. 900–903, 2010, doi: 10.1109/ICDMA.2010.170.

[132] G. S. Maraslidis, T. L. Kottas, M. G. Tsipouras, and G. F. Fragulis, “A Fuzzy Logic Controller for Double Inverted Pendulum on a Cart,” *6th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference, SEEDA-CECNSM 2021*, 2021, doi: 10.1109/SEEDA-CECNSM53056.2021.9566228.

[133] R. Venkata Rao, *No Teaching-Learning-Based Optimization Algorithm*. Springer, Cham, 2016.

[134] H. Malik, A. Iqbal, P. Joshi, S. Agrawal, and I. B. Farhad, *Metaheuristic and Evolutionary Computation : Algorithms and Applications*, vol. 916. 2021.

[135] K. Y. Gómez Díaz, S. E. De León Aldaco, J. Aguayo Alquicira, M. Ponce-Silva, and V. H. Olivares Peregrino, “Teaching–Learning-Based Optimization Algorithm Applied in Electronic Engineering: A Survey,” *Electronics (Basel)*, vol. 11, no. 21, p. 3451, 2022, doi: 10.3390/electronics11213451.

[136] J. Zhu, *Optimization of Power System Operation*, 2nd Editio. John Wiley & Sons, 2009.

[137] S. S. Rao, *Engineering Optimization: Theory and Practice,* 4th Editio. Wiley, 1996.

[138] M. Engineering, “Development of an Optimisation Algorithm for Auto- sizing Capacity of Renewable and Low Carbon,” 2011.

[139] J. Bisschop, *AIMMS - Optimisation Modeling*, 3 rd Editi. Lulu.com, 2006.

[140] M. Shouran, “Load Frequency Control for Multi-Area Interconnected Power System Using Artificial Intelligent Controllers,” Cardiff University, 2022.

[141] R. V. Rao, V. J. Savsani, and D. P. Vakharia, “Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems,” *CAD Computer Aided Design*, vol. 43, no. 3, pp. 303–315, 2011, doi: 10.1016/j.cad.2010.12.015.

[142] F. Zou, D. Chen, and Q. Xu, “A survey of teaching–learning-based optimization,” *Neurocomputing*, vol. 335, pp. 366–383, 2019, doi: 10.1016/j.neucom.2018.06.076.

[143] R. V. Rao, V. J. Savsani, and D. P. Vakharia, “Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems,” *CAD Computer Aided Design*, vol. 43, no. 3, pp. 303–315, 2011, doi: 10.1016/j.cad.2010.12.015.

[144] R. Venkata Rao, “Review of applications of tlbo algorithm and a tutorial for beginners to solve the unconstrained and constrained optimization problems,” *Decision Science Letters*, vol. 5, no. 1, pp. 1–30, 2016, doi: 10.5267/j.dsl.2015.9.003.

[145] R. V. Rao and V. Patel, “An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems,” *International Journal of Industrial Engineering Computations*, vol. 3, no. 4, pp. 535–560, 2012, doi: 10.5267/j.ijiec.2012.03.007.

[146] Veysel Gazi and Kevin M. Passino, *Swarm Stability and Optimization*. Springer, 2011. doi: 10.1007/978-3-642-18041-5.

[147] Ali Marzoughi, “Optimized proportional integral derivative (PID) controller for the exhaust temperature control of a gas turbine system using particle swarm optimization,” *International Journal of the Physical Sciences*, vol. 7, no. 5, pp. 720–729, 2012, doi: 10.5897/ijps11.1097.

[148] W. F. Zeng, Y. J. Zhang, and L. Yan, “Mechanism of particle swarm optimization and analysis on its convergence,” *Proceedings - 3rd International Symposium on Information Processing, ISIP 2010*, pp. 63–66, 2010, doi: 10.1109/ISIP.2010.46.

[149] M. Jain, V. Saihjpal, N. Singh, and S. B. Singh, “An Overview of Variants and Advancements of PSO Algorithm,” *Applied Sciences (Switzerland)*, vol. 12, no. 17, pp. 1–21, 2022, doi: 10.3390/app12178392.

[150] A. P. Engelbrecht, *Computational Intelligence: An Introduction: Second Edition*. 2007. doi: 10.1002/9780470512517.

[151] R. C. Eberhart and Y. Shi, “Particle swarm optimization: Developments, applications and resources,” *Proceedings of the IEEE Conference on Evolutionary Computation, ICEC*, vol. 1, pp. 81–86, 2001, doi: 10.1109/cec.2001.934374.

[152] Y. Shi and R. Eberhart, “A modified particle swarm optimizer,” in *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360),* IEEE, pp. 69–73. doi: 10.1109/ICEC.1998.699146.

[153] M. Imran, R. Hashim, and N. E. A. Khalid, “An overview of particle swarm optimization variants,” *Procedia Eng,* vol. 53, no. 1, pp. 491–496, 2013, doi: 10.1016/j.proeng.2013.02.063.

[154] R. Rajendra and D. K. Pratihar, “Particle Swarm Optimization Algorithm vs Genetic Algorithm to Develop Integrated Scheme for Obtaining Optimal Mechanical Structure and Adaptive Controller of a Robot,” *Intelligent Control and Automation*, vol. 02, no. 04, pp. 430–449, 2011, doi: 10.4236/ica.2011.24050.

[155] A. Abe and K. Komuro, “Trajectory Planning for Saving Energy of a Flexible Manipulator Using Soft Computing Methods,” in *International Conference on Control, Automation and Systems,* IEEE, 2010, pp. 1462–1467. doi: 10.1007/978-3-319-27836-0_6.

[156] Zwe-Lee Gaing, “A Particle Swarm Optimization approach for optimum design of PID controller for nonlinear systems,” *IEEE TRANSACTIONS ON ENERGY CONVERSION*, vol. 19, no. 2, pp. 384–391, 2004, doi: 10.1109/ICEESA.2013.6578478.

[157] J. S. Heo, K. Y. Lee, and R. Garduno-Ramirez, “Multiobjective control of power plants using particle swarm optimization techniques,” *IEEE Transactions on Energy Conversion*, vol. 21, no. 2, pp. 552–561, 2006, doi: 10.1109/TEC.2005.858078.

[158] R. Poli, “Analysis of the Publications on the Applications of Particle Swarm Optimisation,” *Journal of Artificial Evolution and Applications*, vol. 2008, pp. 1–10, Feb. 2008, doi: 10.1155/2008/685175.

[159] A. Chatterjee, K. Pulasinghe, K. Watanabe, and K. Izumi, “A particle-swarm-optimized fuzzy-neural network for voice-controlled robot systems,” *IEEE Transactions on Industrial Electronics*, vol. 52, no. 6, pp. 1478–1489, 2005, doi: 10.1109/TIE.2005.858737.

[160] E. H. Houssein, A. G. Gad, K. Hussain, and P. N. Suganthan, “Major Advances in Particle Swarm Optimization: Theory, Analysis, and Application,” *Swarm Evol Comput,* vol. 63, no. March 2020, p. 100868, 2021, doi: 10.1016/j.swevo.2021.100868.

# Appendix

## 1. Motion code

```
/* USER CODE BEGIN Header */
************************************
  * @file           : main.c
  * @brief          : Main program body
  * @attention
  * <h2><center>&copy; Copyright (c) 2021 STMicroelectronics.
  * All rights reserved.</center></h2>
  * This software component is licensed by ST under Ultimate Liberty license
  * SLA0044, the "License"; You may not use this file except in compliance with
/* USER CODE END Header */
/* Includes ------------------------------*/
#include "main.h"
#include "usb_host.h"
/* Private includes ---------------------*/
/* USER CODE BEGIN Includes */
#define MAX_TETHA1 3850
#define MIN_TETHA1 -3950
#define MAX_TETHA2 4000
#define MIN_TETHA2 -4000
#define DEG2PULS 50
#define m_speed1 30
#define m_speed2 35
#define pos_param 0.1
#define vel_param 3
#define mot1_dir_cw HAL_GPIO_WritePin(GPIOB,GPIO_PIN_5 , GPIO_PIN_RESET)
#define mot1_dir_ccw HAL_GPIO_WritePin(GPIOB,GPIO_PIN_5 , GPIO_PIN_SET)
#define mot2_dir_cw HAL_GPIO_WritePin(GPIOE,GPIO_PIN_6 , GPIO_PIN_RESET)
#define mot2_dir_ccw HAL_GPIO_WritePin(GPIOE,GPIO_PIN_6 , GPIO_PIN_SET)
#define ledg_on HAL_GPIO_WritePin(GPIOD,GPIO_PIN_12 , GPIO_PIN_SET)
#define ledg_of HAL_GPIO_WritePin(GPIOD,GPIO_PIN_12 , GPIO_PIN_RESET)
#define ledy_on HAL_GPIO_WritePin(GPIOD,GPIO_PIN_13 , GPIO_PIN_SET)
#define ledy_of HAL_GPIO_WritePin(GPIOD,GPIO_PIN_13 , GPIO_PIN_RESET)
#define ledr_on HAL_GPIO_WritePin(GPIOD,GPIO_PIN_14 , GPIO_PIN_SET)
#define ledr_of HAL_GPIO_WritePin(GPIOD,GPIO_PIN_14 , GPIO_PIN_RESET)
#define ledb_on HAL_GPIO_WritePin(GPIOD,GPIO_PIN_15 , GPIO_PIN_SET)
#define ledb_of HAL_GPIO_WritePin(GPIOD,GPIO_PIN_15 , GPIO_PIN_RESET)
#define  th2_cal_gin -0.0597
#define  th2_cal_angl_ofst 2000
#define  th2_cal_ofst 1.328
#define  th3_cal_gin  0.0542
#define  th3_cal_angl_ofst 2130
#define  th3_cal_ofst 1.1191
#define  delta_time 0.001
double theta_real=0,theta=0,theta_old1=0,theta_old2=0,theta_old3=0 , enc_real=0,msin=0 ;
double theta_polar=0,vel_sum=0,vel1=0,vel2=0,vel3=0,vel_old1=0,vel_old2=0,vel_old3=0, vel_polar=0, vel_p=0  ;
double theta1=0, theta2=0, theta3=0, timm=0, thetas[30],vels[30],vels_sum=0,velocity1=0,thetas_sum=0, theta1_old=0,delta_theta1=0;
int t3_sp=0,t9_sp=0,sp_st3=0,sp_st9=0;
int t3_st=0,t9_st=0,enc=0,enc_old=0,delta_enc=0,enc2=0;
int t3_sp_cn=0,t9_sp_cn=0,t3_sp_cn2=0,t9_sp_cn2=0;
int dir1=1,dir2=1,m_dir=1;
int cycle_time=0, cycle_time_old=0, ii=0;
 GPIO_PinState t3_state,t9_state; // Variable to store the state of the button
 uint16_t angl1, angl2,  angl3 ;
 int robo_data[20];
 unsigned int mot1_sp=m_speed1,mot2_sp=m_speed2;
 void acrobat_robot_int(){
        ledb_of;
 }
//************************************
 // motor(motor, direction , speed)
 // for select the motor : 1= motor 1, 2=motor2
 // for select the motor direction :
1=CW , -1=CCW
```

```
void motor(int mot,int dir,int pos) // Motor function
{
        if (mot==1)
        {
if (dir==1)//******* ***********
Direction        mot1_dir_cw;
else if (dir==-1)
        mot1_dir_ccw;
// motor1
 //boButton_state = HAL_GPIO_ReadPin(B1_GPIO_Port, B1_Pin);
   t3_st = HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_4);
if((t3_st==1)&&(sp_st3==0))
{ if(dir==1)
t3_sp_cn++;
else if (dir==-1)
t3_sp_cn--;
sp_st3=1;
ledg_on;
}
else if(t3_st==0)
{ sp_st3=0;
ledg_of;}
//*****************
if( t3_sp_cn>(pos))
t3_sp_cn--;
if( t3_sp_cn<(-pos))
t3_sp_cn++;
}
else if (mot==2)
{
if (dir==1)
//******************************** Direction
mot2_dir_cw;
else if (dir==-1)
mot2_dir_ccw;
//^^^^^^^^^^^^motor2
t9_st = HAL_GPIO_ReadPin(GPIOE, GPIO_PIN_5);
if((t9_st==1)&&(sp_st9==0))
 {
 if(dir==1)
t9_sp_cn++;
else if (dir==-1)
t9_sp_cn--;
sp_st9=1;
ledr_on;
}
else if(t9_st==0)
{
sp_st9=0;
ledr_of;}
//*****************
if( t9_sp_cn>(pos))
t9_sp_cn--;
if( t9_sp_cn<(-pos))
t9_sp_cn++;
}
}
//*******************************
void angulear_pos(){
//************************************Theta1
enc_real=enc;
theta_real=(enc_real/4096)*360;
theta=theta_real;
if ((theta_real>180)&&(theta_real<=360))
theta=theta_real-360;

theta_old3=theta_old2;
theta_old2=theta_old1;
theta_old1=theta;
theta_polar=(theta_old1+theta_old2+theta_old3)/3;
theta1=theta_polar;
vel1=(theta_old1-theta_old2)/delta_time;
vel2=(theta_old1-theta_old3)/(delta_time*2);
vel2=(theta_old2-theta_old3)/delta_time;
vel_sum=(vel1+vel2+vel3)/3;
vel_old3=vel_old2;
vel_old2=vel_old1;
vel_old1=vel_sum;
l_polar=(vel_old1+vel_old2+vel_old3)/3;
thetas_sum=0;
for(ii=0;ii<29;ii++){
thetas[ii]=thetas[ii+1];
thetas_sum+=thetas[ii];
}
thetas[29]=theta_polar;
thetas_sum+=thetas[29];
theta1=thetas_sum/30;
vels_sum=0;
for(ii=0;ii<29;ii++){
vels[ii]=vels[ii+1];
vels_sum+=vels[ii];
}
vels[29]=vel_polar;
vels_sum+=vels[29];
velocity1=vels_sum/30;
//**********Theta2 & Theta3
// Calibartion
theta2=th2_cal_gin*(angl2-th2_cal_angl_ofst) + th2_cal_ofst;
theta3=th3_cal_gin*(angl3-th3_cal_angl_ofst) + th3_cal_ofst;
```

```
}
void motor_dir()
{
 if (m_dir==1){
    motor(1,1,vel_p);// motor1
    motor(2,-1,vel_p);// motor2de
 }
 else if (m_dir==-1){
     motor(1,-1,vel_p);// motor1
     motor(2,1,vel_p);// motor2de
 }
}
//Algorithm TLBO
void acrobat()
{
   delta_enc=enc-enc_old;
   delta_theta1=theta1-theta1_old;
   theta1_old=theta1;
   if (delta_enc<0)
   {
        m_dir=1;
                ledy_on;
                        ledb_of;
                        HAL_Delay(1);
   }
   else if (delta_enc>0)
   {
         m_dir=-1;
                ledy_of;
                        ledb_on;
                        HAL_Delay(1);
   }
}
void conv_data()
{
            timm+=(delta_time*200);
            cycle_time_old=cycle_time;
            cycle_time=timm;
            if(((-360 <= theta_polar) || (theta_polar <= 360)) && ((-360 <= theta2) || (theta2<= 360)) && ((-360 <= theta3) || (theta3<= 360)) ){
            robo_data[0]=cycle_time;
            robo_data[1]=theta_polar*100;
            robo_data[2]=theta2*100;
            robo_data[3]=theta3*100;
printf("%d,%d,%d,%d\n",robo_data[0],robo_data[1],robo_data[2],robo_data[3]);
            }
}
void vel_select()
{
             if(cycle_time >2000){
                                  mot1_sp=35;
                                  mot2_sp=40;
             }
             /*
             msin =1+(sin(theta_polar*0.0174533))*pos_param;
vel_p=(velocity1*0.01)*vel_param;
if(vel_p<0) vel_p*=-1;
            mot1_sp=10+vel_p;
            mot2_sp=15+vel_p;
            */
}
/* USER CODE END Includes */
/* Private typedef ----------------------------*/
/* USER CODE BEGIN PTD */
/* USER CODE END PTD */
/* Private define -----------------------------*/
/* USER CODE BEGIN FLQR */
/* USER CODE END FLQR */
/* Private macro ------------------------------*/
int _write(int file, char *ptr, int len)
 /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
 HAL_Init();
 /* USER CODE BEGIN Init */
 /* USER CODE END Init */
 /* Configure the system clock */
 SystemClock_Config();
 /* USER CODE BEGIN SysInit */
 /* USER CODE END SysInit */
 /* Initialize all configured peripherals */
 /* USER CODE BEGIN WHILE */
 acrobat_robot_int();
 HAL_TIM_Encoder_Start(&htim1,
 while (1)
 {
 //*************Acrobat robot
            //  dir1=1;
            //  dir2=-1;
  //vel_select();
  acrobat();
  motor_dir();
  angulear_pos();
  conv_data();
            htim3.Instance->PSC=mot1_sp;
            htim9.Instance->PSC=mot2_sp;
            enc_old=enc;
            enc=TIM1->CNT;
            enc2=__HAL_TIM_GET_COUNTER(&htim1);
            // Get ADC value
            HAL_ADC_Start(&hadc1);
            HAL_ADC_PollForConversion(&hadc1, HAL_MAX_DELAY);
            angl1 = HAL_ADC_GetValue(&hadc1);
            HAL_ADC_Start(&hadc2);
            HAL_ADC_PollForConversion(&hadc2, HAL_MAX_DELAY);
            angl2 = HAL_ADC_GetValue(&hadc2);
            HAL_ADC_Start(&hadc3);
            HAL_ADC_PollForConversion(&hadc3, HAL_MAX_DELAY);
            angl3 = HAL_ADC_GetValue(&hadc3);
    /* USER CODE END WHILE */
    MX_USB_HOST_Process();
    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}

/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
  RCC_PeriphCLKInitTypeDef PeriphClkInitStruct = {0};
  /** Configure the main internal regulator output voltage
  */

    Error_Handler();
  }
 PeriphClkInitStruct.PeriphClockSelection = RCC_PERIPHCLK_I2S;
  PeriphClkInitStruct.PLLI2S.PLLI2SN = 192;
  PeriphClkInitStruct.PLLI2S.PLLI2SR = 2;
  if (HAL_RCCEx_PeriphCLKConfig(&PeriphClkInitStruct) != HAL_OK)
  {
    Error_Handler();
  }
}
/**
  * @brief ADC1 Initialization Function
  * @param None
  * @retval None
  */
static void MX_ADC1_Init(void)
{
  /* USER CODE BEGIN ADC1_Init 0 */
  /* USER CODE END ADC1_Init 0 */
  ADC_ChannelConfTypeDef sConfig = {0};
  /** Configure for the selected ADC regular channel its corresponding rank in the sequencer and its sample time.
  */
  sConfig.Channel = ADC_CHANNEL_2;
  sConfig.Rank = 1;
  sConfig.SamplingTime = ADC_SAMPLETIME_3CYCLES;
  if (HAL_ADC_ConfigChannel(&hadc2, &sConfig) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN ADC2_Init 2 */
  /* USER CODE END ADC2_Init 2 */
}/**
  * @brief ADC3 Initialization Function
  * @param None
    if (HAL_ADC_Init(&hadc3) != HAL_OK)
  /** Configure for the selected ADC regular channel its corresponding rank in the sequencer and its sample time.
  */
  sConfig.Channel = ADC_CHANNEL_3;
  sConfig.Rank = 1;
  sConfig.SamplingTime = ADC_SAMPLETIME_3CYCLES;
  if (HAL_ADC_ConfigChannel(&hadc3, &sConfig) != HAL_OK)
  {
    Error_Handler();
  /* USER CODE BEGIN I2S3_Init 0 */
  /* USER CODE END I2S3_Init 0 */
  /* USER CODE BEGIN I2S3_Init 1 */
  /* USER CODE END I2S3_Init 1 */
  hi2s3.Instance = SPI3;
  hi2s3.Init.Mode = I2S_MODE_MASTER_TX;
  hi2s3.Init.Standard = I2S_STANDARD_PHILIPS;
  hi2s3.Init.DataFormat = I2S_DATAFORMAT_16B;
  hi2s3.Init.MCLKOutput = I2S_MCLKOUTPUT_ENABLE;
  hi2s3.Init.AudioFreq = I2S_AUDIOFREQ_96K;
  hi2s3.Init.CPOL = I2S_CPOL_LOW;
  hi2s3.Init.ClockSource = I2S_CLOCK_PLL;
  hi2s3.Init.FullDuplexMode = I2S_FULLDUPLEXMODE_DISABLE;
  if (HAL_I2S_Init(&hi2s3) != HAL_OK)
  {
    Error_Handler();
  }
  if (HAL_TIM_Encoder_Init(&htim1, &sConfig) != HAL_OK)
  {
GPIO pin : PDM_OUT_Pin */
  GPIO_InitStruct.Pin = PDM_OUT_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  GPIO_InitStruct.Alternate = GPIO_AF5_SPI2;
  HAL_GPIO_Init(PDM_OUT_GPIO_Port, &GPIO_InitStruct);

  /*Configure GPIO pin : B1_Pin */
  GPIO_InitStruct.Pin = B1_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_EVT_RISING;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  HAL_GPIO_Init(B1_GPIO_Port, &GPIO_InitStruct);
  /*Configure GPIO pin : BOOT1_Pin */
  GPIO_InitStruct.Pin = BOOT1_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  HAL_GPIO_Init(BOOT1_GPIO_Port, &GPIO_InitStruct);
  /*Configure GPIO pin : CLK_IN_Pin */
  GPIO_InitStruct.Pin = CLK_IN_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  GPIO_InitStruct.Alternate = GPIO_AF5_SPI2;
  HAL_GPIO_Init(CLK_IN_GPIO_Port, &GPIO_InitStruct);
  /*Configure GPIO pins : LD4_Pin LD3_Pin LD5_Pin LD6_Pin
  Audio_RST_Pin */
  GPIO_InitStruct.Pin = LD4_Pin|LD3_Pin|LD5_Pin|LD6_Pin
HAL_GPIO_Init(OTG_FS_OverCurrent_GPIO_Port, &GPIO_InitStruct);
  /*Configure GPIO pin : PB5 */
  GPIO_InitStruct.Pin = GPIO_PIN_5;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);
/* USER CODE BEGIN 4 */
/* USER CODE END 4 */
/**
  * @brief  This function is executed in case of error occurrence.
  * @retval None
void Error_Handler(void)
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return state */
  __disable_irq();
  while (1)
  /* USER CODE END Error_Handler_Debug */
}
#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line number
  *         where the assert_param error has occurred.
  * @param  file: pointer to the source file name
  * @param  line: assert_param error line source number
  * @retval None
  */
void assert_failed(uint8_t *file, uint32_t line)
{
  /* USER CODE BEGIN 6 */
  /* User can add his own implementation to report the file name and line number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
  /* USER CODE END 6 */
}
#endif /* USE_FULL_ASSERT */
/************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

## 2. TLBO Code

```
%
st_param
% Cost Function
CostFunction = @(x) fitness_f(x);
nVar = 6;         % Number of Unknown
Variables
VarSize = [1 nVar]; % Unknown
Variables Matrix Size
VarMin = [0.3851-1 19.5174-15 -
492.3281-25 0.8957-1 -11.3107-5 -
18.2874-5 ];      % Unknown Variables
Lower Bound
VarMax = [0.3851+1 19.5174+5 -
492.3281+25  0.8957+3 -11.3107+5 -
18.2874+10 ];      % Unknown Variables
Upper Bound
%% TLBO Parameters
MaxIt = 100;       % Maximum Number of
Iterations
nPop = 30;         % Population Size
% Initialize Best Solution
BestSol.Cost = inf;
% Initialize Population Members
for i=1:nPop
   pop(i).Position =VarMin +
rand(1,nVar).*(VarMax - VarMin);
   pop(i).Cost =
CostFunction(pop(i).Position);
      if pop(i).Cost < BestSol.Cost
      BestSol = pop(i);
   end
end
% Initialize Best Cost Record
BestCosts = zeros(MaxIt,1);
%% TLBO Main Loop
for it=1:MaxIt
      % Calculate Population Mean
   Mean = 0;
   for i=1:nPop
      Mean = Mean + pop(i).Position;
   end
   Mean = Mean/nPop;
      % Teacher Phase
   for i=1:nPop
      % Create Empty Solution
      newsol = empty_individual;
            % Teaching Factor
      TF = randi([1 2]);
            % Teaching (moving towards
teacher)
newsol.Position = pop(i).Position +
(VarSize).*(Teacher.Position - TF*Mean);
            % Clipping
      newsol.Position =
max(newsol.Position, VarMin);
      newsol.Position =
min(newsol.Position, VarMax);

      % Evaluation
      newsol.Cost =
CostFunction(newsol.Position);
      % Comparision
      % Learner Phase
   for i=1:nPop
            A1 = 1:nPop;
      A1(i)=[];
      j = A1(randi(nPop-1));
            Step = pop(i).Position -
pop(j).Position;
      if pop(j).Cost < pop(i).Cost
         Step = -Step;
      end
            % Create Empty Solution
      newsol = empty_individual;
        Teaching (moving towards teacher)
      newsol.Position = pop(i).Position +
rand(VarSize).*Step;
            % Clipping
      newsol.Position =
max(newsol.Position, VarMin);
      newsol.Position =
min(newsol.Position, VarMax);
            % Evaluation
      newsol.Cost =
CostFunction(newsol.Position);
            % Comparision
      if newsol.Cost<pop(i).Cost
         pop(i) = newsol;
         if pop(i).Cost < BestSol.Cost
            BestSol = pop(i);
         end
     % Show Iteration Information
   disp(['Iteration ' num2str(it) ': Best Cost
= ' num2str(BestCosts(it))]);
   iter_pop(it)=BestSol;
end
x=BestSol.Position;
```

```
%% Results
figure;
plot(BestCosts, 'LineWidth', 2);
%ssemilogy(BestCosts, 'LineWidth', 2);
xlabel('Iteration');
ylabel('Best Cost');
title('Convergence curve');
grid on;
 disp('Best tunned K 1-6 are given below in sequence: ');
 disp([num2str(x(1)) ' '  num2str(x(2)) ' ' num2str(x(3)) ' ' num2str(x(4)) ' ' num2str(x(5)) ' ' num2str(x(6)) ]);
```

**3. PSO Code**

```
%%
clear;
clc;
close all;
st_param
Max_It=100;       % Maximum Number of Iterations
nP = 30;          % Population Size
%% Problem Definition
% Cost Function
fobj = @(x) fitness_f(x);
dim = 6;           % Number of Unknown Variables
lb = [0.3851-1 19.5174-15 -492.3281-25 0.8957-1 -11.3107-5 -18.2874-5 ];       % Unknown Variables Lower Bound
ub = [0.3851+1 19.5174+5 -492.3281+25 0.8957+3 -11.3107+5 -18.2874+10 ];
% Unknown Variables Upper Bound
warning('off')
c1 = 1.5;
c2 = 1.5;
Wmin = 0.7;
Wmax = 0.9;
Vmax = 0.002;
dim=6;
[Convergence_curve,Best_Cost,Best_X,iter_pop]=PSO(nP,Max_It,lb,ub,dim,fobj,Vmax,Wmax,Wmin,c1,c2);

%% Convergence graph
figure1=figure;
axes1 = axes('Parent',figure1);
hold(axes1,'on');
plot(Convergence_curve,'linewidth',1.5,'Color',[1 0 0])
xlabel('Iteration','FontWeight','bold','FontName','Times New Roman');
ylabel('Function value','FontWeight','bold','FontName','Times New Roman','FontWeight','bold');
box(axes1,'on');
% Set the remaining axes properties
set(axes1,'FontName','Times New Roman','FontSize',15,'FontWeight','bold','LineWidth',1.5);

x=iter_pop.Position;

%%
 disp('Best tunned K 1-6 are given below in sequence: ');
 disp([num2str(x(1)) ' '  num2str(x(2)) ' ' num2str(x(3)) ' ' num2str(x(4)) ' ' num2str(x(5)) ' ' num2str(x(6)) ]);
```

## 4. Controllability and Observability MATLAB code

```
A=[    0          0          0          1          0          0
       0          0          0          0          1          0
       0          0          0          0          0          1
       0      2.6835     -0.0657    -0.0286   -0.0083    0.0284
       0      29.2751    -15.8236   -0.0391   -0.1957    1.2358
       0     -57.5286    247.5924   0.0589     1.4085  -18.0527
];

B=[0 0 0 1.0314 1.6582 -2.4837];

C=[ 1 0 0 0 0 0;
    0 1 0 0 0 0;
    0 0 1 0 0 0;
    0 0 0 1 0 0;
    0 0 0 0 1 0;
    0 0 0 0 0 1];

D = zeros(6,1);

Co = ctrb(A,B);
% controbillty matrix

Ob = obsv (A,C);
% Observability matrix

if  rank (Co) == size (A,1)
   'It is controllable'
else
   'It is not Controllable'
end


if  rank (Ob) == size (A,1)
   'It is observable'
else
   'It is not observable'
end
```

## 5. Convergence percentage code

```
S = load('T001.fig', '-mat');
h  = findobj(gca, 'Type', 'line');
x1 = get(h(1), 'XData')
y1 = get(h(1), 'YData')
x2 = get(h(2), 'XData')
y2 = get(h(2), 'YData')

diff 1= x1 - y1;
diff 2= x2 – y2;

abs_diff = abs(diff1);
mean_diff = mean(abs_diff1);
convergence = mean(diff1);
 % Calculate the convergence percentage
convergence_percentage = (1 - sum(abs_diff1) / sum(abs(diff1))) * 100;
fprintf('Convergence Percentage: %0.2f%%\n', convergence_percentage);



% Display the convergence percentage on the figure
text(5, 8, sprintf('Convergence Percentage: %.2f%%', convergence_percentage))
```