ADDIS ABABA UNIVERSITY

ADDIS ABABA INSTITURE OF TECHNOLOGY

ELECTRICAL AND COMPUTER ENEGINEERING

Joint Position Control of a Walking Humanoid Robot with MPC Controller.

A thesis submitted to School of Graduate Studies, Addis Ababa Institute of Technology, Addis Ababa University in partial fulfillment of the requirement for the Degree of Master of Science in Electrical Engineering (Control Engineering).

By

Afomiya Megersa

Advisor

Dr. Dereje Shiferaw

June,2024

Addis Ababa, Ethiopia

Addis Ababa University

Addis Ababa Institute Of Technology

Electrical and Computer Engineering

Joint Position Control of a Walking Humanoid Robot with MPC Controller

By:Afomiya Megersa

APPROVED BY BOARED OF EXAMINERS

Name	Signature	Date
_____	_____	_____

(Dean, School of Graduate Committee)

Dr. Dereje Shiferaw	_____	_____

(Advisor)

Dr. Lebsework Negashe	_____	_____

(Internal Examiner)

Mr. Nebiyu Tenaye	_____	_____

(External Examiner)

Declaration

I, the undersigned, declared that this MSc thesis is my original work, has not been presented for fulfillment of a degree in this or any other University and all sources and materials used for the thesis have been duly acknowledged.

Name Signature

_____ _____

Place:
Addis Ababa Institute of Technology,AAiT
Addis Ababa University,AAU
Addis Ababa,
Ethiopia.

Submitted in:June 2024

This thesis has been submitted for examination with my approval as a university advisor.

Advisor **Signature**

_____ _____

Acknowledgment

Words cannot express my gratitude to my almighty God who let me see the final fruit of my work and the mother of God.

The assistance provided by my advisers Dr,Dereje and chair person of masters of control engineering for their invaluable patience and feedback. I also could not have undertaken this journey without my classmates, especially my close friends, for their editing help and moral support.

Lastly, I would be remiss in not mentioning my family, especially my sister. Their belief in me has kept my spirits and motivation high during this process.

Abstract

With much of the industrial processes in Ethiopia relying on human labor, people are obliged to perform hazardous and monotonous tasks, such as lifting heavy objects, working in contaminated environments, and engaging in repetitive activities. Unfortunately, this places human lives at risk and exposes their health to potential harm. Moreover, these conditions also negatively impact the industry itself, leading to decreased production quantity and quality. The main objective of this research is to model and simulate a humanoid robot with each leg having 4 degrees of freedom (DOF) and each arm having 4 DOF, enabling it to carry heavy loads and move to different locations. To control the joint position of the robot, an MPC has been implemented and a comparison with a LQR has been done to evaluate the performance. Particle swarm optimization technique has been utilized to tune parameters of the controller yielding better performance. The approach first started with a thorough understanding of problem followed by a 3D modeling of a humanoid robot model in Solidwork. Further with the exported model in Simulink, different sets of angles were assigned to the robot joints so as to imitate human motion. To provide transition from a departure to destination, a sets of way points have been defined in MATLAB and the controller's ability to transit the dynamic system in a stable manner have been tested. The pure pursuit controller has been implemented to track the path, allowing the humanoid robot to navigate through the defined set of points. The implemented MPC has been found to provide efficient trajectory tracking as compared to LQR. The comparison have been further analyzed by calculating the peak error value, where the MPC Controller provided better tracking performance with peak error values of 0.13 in shoulder joint and 0.18 in both hip and knee joint as compared to error values of 0.38 and 0.58 obtained while implementing LQR.

Keywords: Humanoid Robot, MPC, LQR, Pure pursuit, PSO

Contents

List of Figures

List of Tables

Nomenclature

AC Alternative Current

CAD Computer-aided Design

CAE Computer-aided Engineering

DC Direct Current

dDIP Direct Driven Inverted Pendulum

DH Denavit Heartenberg

DOF Degree of Freedom

DSP Digital Signal Processing

DSPACE Digital Signal Processing and Control Engineering

FPGA Field Programmable Gate Array

IPMLSM Ironless Permanent Magnet Linear Synchronous Motor

LQR Linear Quadratic Regulator

MIMO Multiple Input Multiple Output

MPC Model Predictive Controller

NN Neural Network

PID Proportional, Integrator and Derivative

SISO Single Input Single Output

SMC Sliding Mode Controller

URDF Unified robotics description format

Chapter 1

Introduction

This chapter reviews the general overview on the background, statement of the problem, objectivity, methodology, scope and thesis outline.

1.1 Background

Manufacturing industry is accountable for converting raw materials and components into finished goods for marketing. This is mostly done on a large-scale production line using machinery and skilled labor. In Ethiopia, the manufacturing industry plays a momentous role in the country's economy, with companies producing processed food, soaps, tobacco, beverages, textiles, leather, and other goods.

[1]In most of those industries variety of tasks is done by humans such as dangerous and mundane tasks. For example, lifting heavy (and slightly heavy) loads, working in contaminated, dusty environment and engaging in repetitive activities. Humans are endangering their lives due to this, as their health is at risk. Additionally, the quality and quantity of production will decrease. This is because the repetitive and tiring nature of the work will cause workers to become fatigued and lose focus. This study aims to model a bipedal robot to perform tasks in place of humans.

There are two types of robot according to their structure, legged robot or wheeled robot. A legged robot is more flexible and mobile in difficult terrain as compared to wheeled robot. Wheeled robot are convenient on flat surface or prepared surface, so wheeled robot will be faster for those specific areas. Also wheeled robot are simpler in terms of mechanical architecture and control algorithm. But when it come to carrying different amount of loads, a legged robot is more suitable as the leg can be used as an anchor [2]. Due to this, the robot used in this paper is a bipedal robot. A bipedal robot is a two legged robot, which can resemble human's walking movement and can be programmed to perform different tasks as required. The ability of a bipedal robot imitating human's body movements or actions will make them an essential candidate to be used in our daily life. Bipedal robot will make human's live easier especially for elders and handicapped.

A two legged robot faces two problems, stability control and motion control. The stability control is associated with robot balance and the motion control is associated with ability to walk or move.Robust posture correction algorithms and effective pace balancing are essential for a stable walking motion in a bipedal robot. Despite humongous research efforts, developing or achieving and implementing intelligent motion algorithms to control a bipedal robot remains an extremely challenging task, especially in terms of stability, robustness and adaptability[3].

Recently research on robotics and robotics development has evolved from industrial robot manipulators to encompass autonomous and animal like or human like robot. Particularly the field of humanoid robotics has been notable progress, increased by advancements in actuators, computers and other enabling technologies.

When dealing with bipedal robot, robot's locomotion, sensing, reasoning,localization, path planning and tracking should be addressed.

Locomotion is how the robot moves with in the environment.That is the robot is wheeled or legged. This paper deals with the legged robot.

Sensing is how the robot gets information about itself and current state of the environment. The bipedal robot has a sensor on its joint,which will be discussed later on.

Reasoning is how the robot utilizes the information obtained from the sensors and make a decision or an action. So for this paper the decision will be made by the controller. This paper tries to see two controller MPC and LQR and compare their results.

Localization is determining the position and orientation of the robot. Since attaching a sensor to the 3D model was difficult, the pose (position and orientation) of the robot is dependent on the given starting point and ending point. And the bipedal robot should be placed on the given starting point.

When dealing with localization, the robot's surrounding should be considered which is the map of the environment. Mapping is the process of generating the map that is used by localization algorithms. There are different ways where a map is used. Different maps can be imported from map series in MATLAB, the robot itself can visualize its surrounding and form a map or the engineer can form a map in MATLAB by providing different syntax and values. For this paper a map is imported from the MATLAB.

Path planning is generation of the time sequence of points with location for the robot such that it reaches the desired goal without colliding with obstacles. Waypoints are utilized to establish a route, and an algorithm is developed to link the specified points.

Path tracking is an algorithm used to track the path. Pure pursuit controller is utilized for tracking the given set of location points. It is a type of path tracking algorithm used in robotics and autonomous vehicle navigation. It calculates the curvature of the path ahead and steers the vehicle accordingly. The controller continuously updates the desired path based on the current position and next position, allowing for smooth and accurate tracking of the desired trajectory. This type of controller is commonly used in applications such as mobile robots, self-driving cars, and unmanned aerial vehicles.

The aim is to control the joints of the bipedal robot, so as controller, MPC is implemented [4]. Model Predictive Control (MPC) is a popular advanced control strategy used in various industrial and engineering applications. Some of the key features of MPC include:

Predictive Control: MPC uses a dynamic model of the system to predict future behavior, allowing it to anticipate and account for future changes and disturbances.

Optimization-Based: MPC formulates the control problem as an optimization problem, enabling it to optimize control actions over a future time horizon while considering constraints on the system variables.

Multi-variable Control: MPC can handle multiple input and output variables simultaneously, making it suitable for complex multi-variable systems.

Handling Constraints: MPC can handle both input and output constraints, ensuring that the control actions remain within safe operating limits.

Robustness: MPC is inherently robust to disturbances and uncertainties, as it considers future predictions and constraints when calculating control actions.

Adaptability: MPC can be easily adapted to different system models and constraints, making it suitable for a wide range of applications.

Performance Trade-offs: MPC allows for the trade-off between conflicting control objectives, such as stability, set-point tracking, and minimizing control effort.

These features make MPC a powerful and versatile control strategy for a wide range of industrial processes, including chemical processes, power systems, automotive control, and robotics.

MPC is compared with LQR, an optimal controller that ensures good stability and guaranteed system stability margin.[5]The major distinction between MPC and LQR are that LQR optimizes over the entire time window where as MPC optimizes within a receding time window. Additionally, MPC computes a new solution each time, while LQR employs the same single or optimal solution throughout the whole horizon.

The controller controls the joint of the bipedal robot but first the joint has to be actuated so basically the controller controls the motor which in return rotates the joint of the bipedal robot.

An actuator is a device that triggers an action, such as a movement to a robot. This is accomplished by using motors, such as when an actuator is employed to rotate the joints of a robot or for a robot gripper to be opened or closed. In this study, a servo motor was utilized to rotate the bipedal robot's arm and leg.

Servo motors are electronic devices with rotary and linear actuators that rotate and push parts of a machine with precision. They are basically used to control linear and angular position, also specific velocity and acceleration[6].

Servo motors are widely used in various platforms. In unconventional vehicles, they regulate the speed by receiving electrical signals from the gas pedal and adjusting the engine speed through the car's computer. Commercial aircraft also use servos to operate different components within the plane. Moreover, servos are commonly used in industrial settings, such as robotics, pharmaceuticals, food services, and in-line manufacturing.

There are two types of servo motors. The first is the AC servo motor, mainly used in industrial fields and relying on encoders. These servo motors work through controllers

providing closed loop control and feedback. They are known to function with high accuracy and are easily controllable. The second type is the DC servo motor, which was used by Fuji Electric but is now rarely used, as AC servo motors are easier to use, more effective, advanced, and reliable.

Servo motors offer numerous benefits, but like everything else, they also have their drawbacks. One of the advantages of servos is their consistency and ability to operate at a constant speed. When the motor is under a heavy load, the driver increases the current to the motor coil to rotate it. This ensures that servo motors are always precise, allowing companies to operate them at high speeds. The downside of servo motors is their high maintenance and operation costs. Additionally, when a device using a servo is stopped, the motor continues to move back and forth by one pulse, which is not ideal for areas or devices that are not suitable for vibrations.

1.2 Statement of the problem

Industrial tasks that are dangerous, laborious, and repetitive for humans can be performed by bipedal robots, particularly humanoid robots, leading to improved working conditions and production levels in industries, as well as better health for workers.

Imagine a person working in factory where he or she has to lift heavy objects repeatedly throughout the day, putting strain on their bodies. Now, picture a humanoid robot capable of performing these tasks effortlessly and without any risk of injury. This robot can greatly relieve the workers of their physical burden and improve their overall well-being.

Bipedal robot control and path tracking accuracy are fundamental challenges in the field of robotics. Several previous methods have been attempted to address these problems, but they still pose significant difficulties. Some of these aspects are:

Walking Robot Control: Controlling a walking robot involves coordinating the movement of its limbs to achieve stable locomotion. Some of the basic problems in walking robot control include:

Gait generation: Designing an appropriate gait pattern for the robot to follow is crucial. This involves determining the sequence and timing of leg and arm movements to ensure stability and efficient locomotion.

Adaptive control: Dealing with uncertainties, such as uneven terrain or external disturbances, requires adaptive control techniques to dynamically adjust the robot's behavior.

Path Tracking Accuracy: Path tracking accuracy refers to the ability of a walking robot to accurately follow a desired trajectory or path.

Previous attempts to address these problems have utilized various control algorithms. Nevertheless, achieving reliable control over walking still poses a difficult challenge [7].

In summary, the basic problems in walking robot control and path tracking accuracy stem from the complexity of coordinating leg and arm movements and dealing with uncertainties. While various methods have been attempted to address these challenges, achieving robust and accurate walking control remains an active area of research in robotics. The key issue lies in developing a suitable controlling mechanism that can generate a control signal to minimize errors and enhance accuracy.

1.3 Objectives

1.3.1 General Objective

To design a controller for a 6-DOF bipedal robot located at the shoulder, hip, and knee to walk from one place to another while carrying a load.

1.3.2 Specific Objective

- 3D modeling of a bipedal robot and simulation.
- To design MPC and LQR controllers and implement to the walking robot.
- To design a path in the environment.
- To test the performance using simulation .

1.4 Methodology

Methodology	Task in detail
Literature review	Reading published research papers, books, articles related to this topic
Modeling of a Bipedal robot	Model the 3D modeling of the Bipedal robot.
Model verification	Verify the modeled Bipedal robot using Simulink
Forward and Reverse kinematics	By using DH parameter and elimination method.
Motion Planning	Imitating human's body movement by providing motion signal.
Based on the obtained model	Design MPC and LQR for reference stabilization.
Path planning	Generate a reference path by giving out way points.
Path tracking	Using a path tracking controller, bipedal robot follow the reference path.
Simulation	To examine the controller and whole system.

Table 1.1: Methodology

1.5 Scope

This thesis examines and develops a 3D model of a bipedal robot. Two different controllers are designed to govern the robot's motion tracking and minimize control effort. MPC and LQR are utilized as controllers, with PSO used to tune the parameters. Path planning involves using a set of points to lay out the path, and the robot's surroundings are addressed using a map imported from a MATLAB file. The path is tracked using a pure pursuit controller for navigation.

1.6 Thesis outline

Chapter one provides an introduction to the system and offers insight into the issues that will be discussed later. It essentially gives an overview of the entire system.

Chapter two discusses and reviews various papers that serve as a starting point. Each literature provides unique insights and perspectives, making it a valuable resource for understanding the bipedal robot and its control mechanism.

Chapter three explains the process of obtaining and verifying the bipedal robot model in Simulink. The model is modeled using Solidwork by first defining the length of the bipedal robot and number of joints needed, and DH parameters is utilized to calculate forward and inverse kinematics of the bipedal robot. Verification is done by simulating the joints at different angles and comparing the results to the expected outcomes, ensuring the accuracy of the model representation. And lastly since the joints has to be actuated, servo motor is modeled.

In chapter four, the controller design is introduced to enable the bipedal robot to track reference points using a walking pattern formulation. MPC and LQR are utilized as controlling mechanisms.

Chapter five provides the simulation result of the controller and compare the performance and chapter six concludes with recommendations for future work based on the findings.

References used in this thesis are presented in numerical order.

Chapter 2

Literature review

This chapter presents crucial examples for literature review. There has been significant work done in the field of bipedal robots and different controllers used for these kind of robots. The amount of work done is too extensive to list comprehensively, so only a few examples will be provided.

2.1 Literature Review on Humanoid Robot

The first thing to consider when approaching this project is the model of the bipedal robot.

In [8] a simulation of bipedal robot in Simscape Multibody is introduced. The bipedal robot includes 14 body types and six revolute joints, with one revolute joints represents one rotational degree of freedom, situated at ankle, knee and hip. In order to maintain consistent stiffness value regardless of the bipedal robot task a compression spring is placed between each link. The foot-ground contact behaviour is simulated using the simscape multibody contact forces Library, that offers force models for intermittent contact.

The other paper proposed a reactive controller (MPC) that allows the bipedal robot to walk blindly. SLIP (spring loaded inverted pendulum) modelling is used. This will helpful to conserve energy [9].

By using Denavit-Hetenberg dynamic method the bipedal robot kinematics is analyzed, only assuming the moving part of the leg. PD controller is used as controlling mechanism. The bipedal robot has two legs with each leg having six degree of freedom [10].

The other paper [] import a cad file in URDF format from SOLIDWORK and open it in Simulink. The cad file is a two legged robot with 6 DOF on each leg and it consistence of solid block, rigid block and joints. Solid block:-which tell us the geometry of body. Rigid block:- there is a conversion between the connected bodies. Joints:-depending on the degree of freedom. For the bipedal robot to move a 6 DOF revolute joint is attached to the torso of the robot. After this, contact force between the foot of the robot and the ground is established so that the bipedal robot will walk on the surface. After the release of 2019b MATLAB SIMSCAPE MULTIBODY has blocks called the spatial contact force. This spatial contact force develops a contact force that will make the robot walk. The contact force is developed as soon as the robot touched the ground.

When concerning with control of the walking of a stable bipedal robot, one must see its complexity and also at the same time not to overlook the firsthand need to ensure that the bipedal robot has to always maintain its balance while walking. Due to this thought process, the controlling of bipedal robot is an intriguing task.

For example in North-eastern University, they designed thruster-assisted walking bipedal robot, named harpy, it is equipped with the sum of six actuators, and two pairs of coaxial thrusters that is fixed to its torso. A pair of leg is equipped with three actuated joints, In order for the leg to move sideways actuators are used at the hip and parallelogram mechanism is implemented to realize actuation in the lower portion of the legs. The main objective of this research is to insure asymptotic stability and good performance with minimum cost[11].

Agility robotics designed a bipedal robot (Cassie) with light weight and compliant legs that uses spring mass principle to run and walk [12]. Its direct predecessors include Mabel [13] and ATRIAS [14]. In order to navigate obstacles, fast online tracking optimization formulation is proposed. Optimizing the trajectory is too expensive to be done online so to simplify this reduced order dynamic model and redefined contact is used.

This paper proposed a cascade control structure that connects the high-level learning based controller with the low level model based controller. The reference trajectory is computed by the high level controller, where as the low level controller helps with adjusting the reference trajectories based on instantaneous state feedback. This process compensates for uncertainty in the environment [15].

The robustness issue came because the environment outside the lab is not smooth and this makes it unsuitable for many applications. Most of the best controllers that enable a bipedal robot to walk over uneven terrains require predefined footstep locations or exact information about the terrain height changes [16], [17], [18].

Various automated structures were designed to imitate human movements in the early 18^th century. Leonardo da Vinci was among the first to develop such structure in the form of humanoid, as depicted in the figure 2.1. And the early 20^th century, the Westinghouse Society built the first electrical structure and coined as Elektro humanoid, as shown in figure2.2.

Figure 2.1: Leonardo robot

Figure 2.2: Elektro Robot

Most recent humanoid robots can display emotions and learn as they travel through autonomous interaction with the environment. Bipedal humanoid robotics presents an alternative research tool for studying and understanding human behaviour and at same time for cognitive science. Japan and Korea has the majority of research activity on this kind robots.

2.2 Literature Review on MPC

On literature review, optimal control is widely discussed. Model predictive control is a sophisticated process control method that can manage a process while satisfying specific constraints. It has been utilized in the chemical and oil industries since the 1980s and has more recently been employed in robotics systems.

Model predictive controllers depend on dynamic models of the process, mostly linear empirical models obtained through system identification. The key advantage of MPC is its ability to optimize the current time slot while also taking into account future time slots. This is achieved by optimizing a finite time horizon, implementing the current time slot, and then repeatedly optimizing again, differing from a linear quadratic regulator(LQR). Moreover, MPC can foresee future events and take control actions accordingly.

In contrast, PID controllers lack this predictive ability. MPC is nearly always implemented as a digital control.

The present paper[19] discuss mathematical modeling and non linear control of robotic manipulator. To derive the kinematics model, DH parameter are utilized, while the dynamics are based on the Euler Lagrange equation. Two modern control strategies, H∞ and model predictive control(MPC), are explored to design the control laws. For optimal performance the controllers have been amended through simulations conducted in the MATLAB Simulink environment. The designed control laws are subjected to different inputs and tested for effectiveness in transient parameters such as steady state error , overshoot, and settling time. The simulation results shows the effectiveness of the developed controllers in precisely tracking the reference motion trajectories.

An author suggests an optimization- based solution for controlling and estimating dynamics systems, specifically focusing on mobile robots. The solution employs model predictive control to control the mobile robot, utilizing both single and multiple shooting methods to convert the optimal control problem into a nonlinear programming problem and evaluate their effectiveness. The author highlights the drawback of single shooting as the propagation of non-linearity, resulting in highly nonlinear integrator functions for large prediction horizons. CasADi in MATLAB is utilized to design the MPC. The research centers on mobile robot with 2 inputs(linear and angular velocities) and 3 outputs (translatinal:X and Y , rotational: θ) [20].

[21] This paper proposes the model of six DOF robotic arm, as well as the derivation of two modern control laws: MPC and H∞. Simulation result detects that both control methods offer satisfactory tracking performance. When comparative analysis of the performance achieved by both controllers is done , it is revealed that MPC outperforms ∞, at the expense of higher overshoot for controlling the robotic arm. But when the value of prediction window is increased, the overshoot value can be reduced in MPC response.

This paper compare the performance of three different controllers: proportional derivative(PD) controller, sliding mode controller(SMC), and model predictive controller(MPC) when handling with uncertain load mass on a quad-copter. The authors [22] looked the uncertain load as bounded external disturbance on the quad-copter and for MPC design to calculate the linear discrete time space equation, small angle approximation is utilized. Finally the study summarizes that the crucial impact of load mass uncertainty lies in the quad-copter's stability rather than trajectory tracking error. In addition, it suggests that for strict requirements on the load transportation stability or fast response, a robust or fast controller design such as MPC is recommended.

[23]This paper presents a nonlinear, multi-input multi-output(MIMO), and coupled

robotic manipulator. It utilizes the nonlinear integral sliding mode controller in order to control the position of robotic manipulator. A nonlinear integral sliding mode controller, that is a robust, simple, and effectively increase the operation of the manipulator. Simulation is done by using MATLAB Simulink and the results demonstrate the robustness of this control method in the presence of uncertainties such as payload variation, external disturbance, and other inherent factors . The performance of the integral sliding mode controller is compared with another controller which is proportional-integral-derivative(PID) controller and it depicted that the proposed controller to be more effective and superior for the robotic manipulator.

The paper introduce a novel terminal sliding mode control for globally achieving finite time tracking of uncertain robot manipulators. A unique integral sliding surface is suggested to entirely eliminate singularity. Through the integration of the proposed sliding surface and finite-time stability theory, a straightforward non-singular terminal sliding mode control is formulated. The global finite time convergence of both the sliding surface and tracking errors is demonstrated. Simulation conducted on a two degree of freedom(DOF) robot are included to showcase the effectiveness and enhanced performance of the proposed approach[24].

2.3 Literature Review on LQR

[25],controller is mixture of two controllers which are PID algorithm, a conventional approach, and the linear quadratic regulator(LQR). the LQR controller uses a state space approach. One of the key advantage of LQR is that it generates a systematic way of computing the state feedback control gain matrix. LQR design minimizes both weighted squared state error and control effort. In this paper, we have compared the performance of both controllers in the presence of disturbance. This is implemented on the Quanser QNET 2.0 DC motor board for NI Elvis II, and the controllers are designed using Labview.

The present paper suggests a novel optimal PID control design, using the methodology of linear quadratic regulator to reach for the optimal parameters of the PID controller. The performance measure involves only output signals in the augmented state vector. Weighting function are determined through poles assignment, and the existence criteria of the optimal PID controller are derived. The new PID tuning algorithm is applied to the speed control of BLDC motors. Computer simulations and experimental results demonstrate that the performance of the optimal PID controller is superior to that of the traditional PID controller[26].

The effectiveness of controllers can be verified through the use of an inverted pendulum. The traditional inverted pendulum is driven by a rotational motor, and its control performance is impacted by friction and gap of the transmission mechanism. An Innovative Direct Driven Inverted Pendulum(dDIP) has been proposed, which consists of an inverted pendulum(IP) mounted on stage driven by an iron-less permanent magnet linear synchronous motor(UPMLSM). [27]this paper first analyzes the structure of the dDIP and builds its mathematical model. Subsequently, a LQR controller is designed, and numerical simulations in MATLAB are conducted. Finally, the dDIP is successfully controlled by dSPACE controller and TI28DSP-based control card. The results demonstrate that the dDIP can be stabilized to its upright position, and simultaneously, the cart's displacement can be regulated to zero after applying a step disturbance to the dDIP. Experiments indicate that the transition time is within4s, highlighting the excellent dynamic characteristics of the dDIP that cannot be achieved by traditional IP.

[28]This paper details the control design of buck converter-driven DC motor, centering on the application of Linear Quadratic Regulator (LQR) and proportional-Integral(PI) techniques in order to regulate the motor's speed. The investigation considers the dynamic system comprising the converter or the motor, examining it in state-space and transfer function forms. The study considers a comprehensive analysis of simulation results for both LQR and PI techniques in both frequency and time domains. Controller performance is evaluated based on angular velocity, duty cycle input energy, and armature current. Finally, the paper offers a comparative assessment of each controller's impact on system performance, discussing their respective effects.

2.4 Literature Review on Path Planning

In the field of robotics, path planning and trajectory planning are crucial, particularly in automation. The goal for automatic machines and robots is to operate at high speeds to achieve shorter production times. However, high operating speeds may hinder accuracy and the robot's repeatable motion, requiring extreme performance from the control system and actuator. Therefore, generating the path and selecting the actuator require particular care. Path planning and trajectory planning are increasingly significant in robotics as a result.

A path is sequence of points with starting and ending points. Path planning algorithms concerns with searching a geometric path from an initial value to a final one, where each configuration and state on the path is executable while taking time into account. The points are passed through per-defined through points, either in the joint space or in the operating space of the robot. Path planning also enables the robot to find the shortest

obstacle-free path from the initial point to the final point .

A trajectory is succession of states taken by robot, defined by time and possibly velocity. Trajectory planning involves time period planning for a robot to walk from one to the next state, while adhering to he robot's kinematic limits, which are determined by its dynamics and constraints. Additionally, trajectory planning algorithms incorporate time information into geometric path .

Navigation refers to driving a mobile robot or other device from place to place.The main aspects of navigation is:-

- Perception can be explained as the system that endows the bipedal robot with the ability to comprehend, perceive and reason about the surrounding environment.
- Localization is a process of locating a robot in its environment. Localization algorithms uses sensor and map data to estimate the position and orientation of mobile robot in a known environment in order to perform a given task. Localization can be classified in different aspects.

  Based on problem set up.

  - Position tracking
  - Global localization
  - Kidnapped problem

  Based on environment

  - Static environment (this paper uses a static environment)
  - Dynamic environment
  - Known environment
  - Partially known environment

  Based on number of robots

  - single localization
  - multi robot localization
- Cognition presents the behaviour of intelligent agents in the physical world or virtual world in case of simulated cognitive robotics (for this paper the bipedal robot does not embody any intellectual ability).

- Motion control is when the position or the velocity of the bipedal robot is controlled by using motion control device such as electric motor, linear actuator, servo motor or hydraulic pump . There are three types of moves where a bipedal robot uses to navigate around the physical world. They are linear, joint and circular movement. Even if the end goal is the same, the path the robot makes along the way is the determining factor. For this paper servo motor is used to control the joint of the bipedal robot.

To implement path planning, map of the environment is necessary. Mapping is the procedure of creating the map by using localization algorithms. Robot mapping includes recording environment data such as obstacles,features and objects to create a proper interaction between the mobile robot and its surroundings. Grids are typically used to present maps, where the grid position or location can be designated as either occupied or free space. This can be achieved by utilizing objects with arrays and parameters. The object will contain information such as:-

- The grid itself
- The map limits
- Origin information
- Resolution of the map

So that the map can be correlated to the physical word. Occupancy grid in MATLAB provides utilities for mapping and navigation workflow, by giving access to the information about the map like getting the occupancy, inflating the occupied space in the map to prevent the robot running in to the obstacle. Maps can be in different form. occupancy grid and dimensions. The occupancy grid can be in different form. Binary form or probabilistic form. And the dimension can be 2D or 3D. Also a map can be generated by defining walls and place objects wherever it is necessary or simply a map can be imported from MATLAB file.

For this paper, way points are used to generate a path for the robot to follow. In recent years there have been a great advancement in path planning for mobile robots.

This paper[29]presents a scalable path planner for humanoid robots, aiming to improve human-robot interaction. The method utilizes a time index to generate natural-feeling paths for humans and waypoint-based paths for robots, allowing accurate tracking even in complex dynamics. It can be easily expanded to cover wide areas. Experiments show that the robot successfully reached its goal location while iteratively replanning the path, even in unexpected exceptional situations.

2.5 Literature review on Path Tracking

The pure pursuit controller is path tracking controller that calculates the curvature to guide the mobile robot from its current position to its final destination. This involves computing the angular velocity command to move the mobile robot from its current location to predetermined look-ahead point. The linear velocity is considered constant, allowing for changes in the mobile robot's linear velocity at given point. The algorithm's primary concept is to select a goal position located at a certain distance ahead of the mobile robot along the path

The algorithm takes and moves the look-ahead point along the path based on the current position of the bipedal robot until it reaches the final point of the path. Essentially, this means that the mobile robot is continually pursuing a point ahead of it.

The controller can be tailored to a specific list of waypoints, along with desired linear and maximum angular velocities, based on the bipedal robot's specifications. Using the mobile robot's pose (position and orientation) as input, it calculates the linear and angular velocity commands. The execution of these commands by the mobile robot depends on the system, so it's important to consider how the robot can perform a motion based on these commands .

Pure pursuit controller uses a reference coordinate frame as inputs and outputs. The input way-points are $[x, y]$ coordinates, which are used to calculate the bipedal robot velocity commands.

The pose of the bipedal robot composes of an xy position and angle in the form [x, y ,θ]. The x is in positive right direction and and y is the up directions 2.3. The theta value is the angular orientation of the bipedal robot measured counterclockwise from the $x - axis$ (robot currently at 0 radians). The figure below shows the reference coordinate system

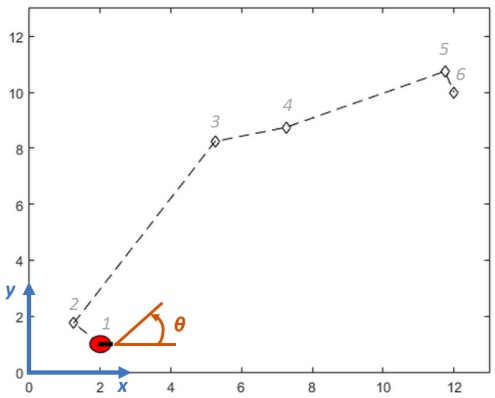

Figure 2.3: Reference coordinate system

The "LookAheadDistance" determines the placement of the look-ahead point and its very crucial in tuning the pure pursuit controller. This distance specifies how far ahead the bipedal robot should search along its path from its current location in order to calculate angular velocity commands. The figure below 2.4 shows the bipedal robot and the look-ahead point, and it is essential to note that the actual path may not directly align with the way-points.

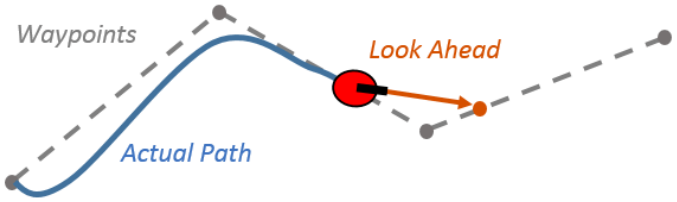

Figure 2.4: The look ahead distance and the mobile robot

Changing this parameter will change how the bipedal robot tracks the path. The major two goals are:

- Regaining the path
- Maintaining the path.

If the LookAheadDistance is small, the bipedal robot will move quickly towards the path, but it may oscillate and overshoot along the intended path. To avoid this, select a larger LookAheadDistance. Still, selecting a larger LookAheadDistance may result in greater curvature near the corners, so selecting the appropriate value is crucial. The figure below depicts this.

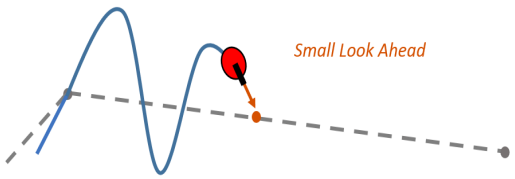

Figure 2.5: Choosing appropriate the LookAheadDistance

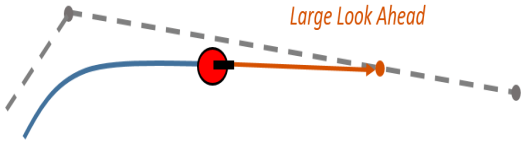

Figure 2.6: Choosing appropriate the LookAheadDistance

The LookAheadDistance should be selected according to the specification and should be considered for path controlling. The response will also be affect various angular and linear velocities.

Chapter 3

Model And Verification

This chapter provides modeling of bipedal robot in SOLIDWORK, DH parameter, forward and inverse kinematic of bipedal robot.

3.1 Modeling of a Humanoid Robot

When concering with robots, modeling and simulation permites for fast prototyping of algorithms and testing scenarios by imitating the behavior of real-world systems.

Modelling is about representation of things and allowing ideas to be analyzed; it is essential to all activities in the process for designing or creating an some form or other. In other words model is a way of symbolizing a particular view of an identifiable system of some kind. Models is used:

- To understand problems;
- Rather than applying it on larger scale or in the real word, a models are used to analyze so it will reduce the the risk

Robot models simulate the dynamic and kinematic properties of the robot manipulator and other rigid body systems. The models are rigidBodyTree objects containing rigidBody and rigidBodyJoint elements with joint transformations and inertial properties .

The bipedal robot has a quite large number of degree of freedom, so assigning the reference frame will be hard. So SOLIDWORK is a best fit for designing the bipedal robot.

3.1.1 SOLIDWORK

The fusion of robotics computer, electronics, and control systems has resulted in range of amazing products from smartphones to self deriving cars. To create these products, the first thing to do is digitally model and develop. SOLIDWORK is 3D modeling computer-aided design(CAD) and computer-aided engineering(CAE) software.

Jon Hirschtic was MIT graduate,who is responsible in developing SOLIDWORK and the software was bought by Dassault Systems in 1997. The software includes a various platform which can be used for both 2D and 3D design.

The intention is to use the bipedal robot in place of the workers and also for the bipedal robot to collaborate with the workers in the industries so that the workers will get some relief, and additionally benefit the industry.

The first thing to do is visualize what the robot should look like and choosing number of degree of freedom depending on the desired performance of the bipedal robot. Also check what has been done before[30] and use it as a starting point to model the bipedal robot in SOLIDWORK.

The desired performance is, the bipedal robot will walk from one point to another by carrying loads. The bipedal robot will have basic human being's structure; head, two arm, two leg and torso[31].

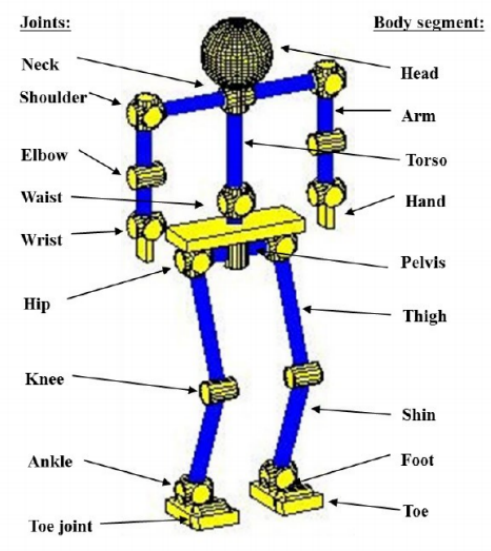

Figure 3.1: Name of body parts

To decided on the number of degree of freedom which the bipedal robot will have, basic human's movement should be considered. The selection approach consists of three main plane of motion, sagittal, frontal and transversal planes.

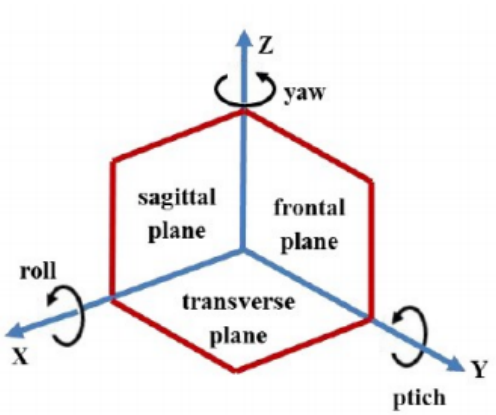

Figure 3.2: Main axis coordinate and plane

- Sagittal Plane: It includes forward and backward movements which is not sideways or rotating. For instance, squat where the upper boy is stabilized while the lower body is in motion and bicep curl for the upper body movement. It basically cuts the body into left and right halves.

- Frontal Plane: By cutting the body into front and back halves it creates side-to-side movements. Example for this kind of movement will be lateral leg rises, straight-arm lateral raises, side shuffle and side lunge.

- Transverse Plane: It is a twisting movement. The most common examples are spinal rotation and limb rotation. So transversal cuts the body into top and bottom halves.

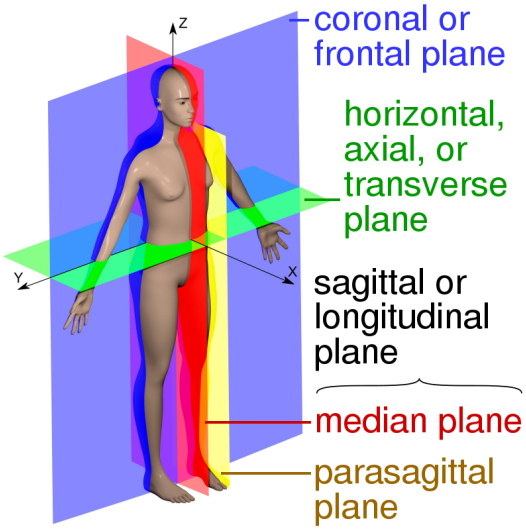

Figure 3.3: Plane of motion

To determine the plane of motion for specific movement, begin by analyzing how the movement would intersect with three imaginary lines or planes. To confirm if the movement is taking place within that plane of motion, it must align with the imaginary lines or planes.

When walking up the stairs, flexion(forward and upward movement) occur at the hip, knee, and ankle. This movement is considered as sagittal plane movement because it is parallel to the imaginary line that divides the body into right and left sides.

When somebody tries to grab something, they suddenly stops and reaches directly out to the side and this movement is happening in the frontal plane because the lateral reach would run parallel to a line dissecting the body into front and back sections.

When some one turn around to look behind, your torso rotation runs parallel to a line separating the body into a top section and a bottom section making the movement transversal plane. Mostly walking purpose sagittal plane of motion is needed.

So having all this in mind,the humanoid robot will have two DOF on the shoulder, one DOF on the elbow, one DOF on the wrist, two DOF the hip, one DOF on the knee and one DOF of the ankle. Which totally become 16 DOF.

For more realistic purpose and more resemblance, two DOF on the neck is added. The first one is to move the neck left and right direction and second one is to move the neck up and down. So the bipedal robot has 18 DOF in total.

After this, the length of the robot arm (from the shoulder to the elbow, from the elbow the wrist and the length of the gripper) , the length of the robot leg (from the hip to the knee, from the knee to the ankle), the length of the foot,the length of the torso, the length of the head and neck should be evaluated. The values are taken from the a research paper[30]. Lastly, before designing or building anything in SOLIDWORK coordinates frame must be selected. SOLIDWORK have three module;

- **Part Module** :Enables to model and design parts of the bipedal robot. One of the important thing when defining the parts of bipedal robot in part module is specifying the material its made up off. SimulationXpress needs to know the elastic properties of the material where parts are made off. The material can be assigned by picking a material from a material library.

  SOLIDWORKS material have two sets of properties, physical(mechanical) and visual. SimulationXpress uses the physical properties of the material specified in the SOLIDWORKS material library. Materials can be orthotropic, isotropic, or anisotropic. SimulationXpress currently handles isotropic materials..

  In the material library, there is aluminium, steel, copper, iron and others and also their alloys. The bipedal robot has to carry load so the element its made up of has to have good strength. Also choosing a lighter material is a good choice,since it helps with controlling the bipedal robot.

  Aluminum's low density , non-toxic nature, high thermal conductive, excellent corrosion resistance, and ease of casting, machining, and forming make it the ideal choice. Additionally, it is non-magnetic and non-sparking. Aluminum is the second most malleable metal and the sixth most ductile. As the 13th element in the periodic table, this silvery-white metal is surprisingly the most abundant metal on earth.

- **Assembly Module** :Enables to assemble the different parts of bipedal robot together and carry a motion study on the assembled system.

- **Drawing Module** : creates the drawing of the created parts in various formats. The last module is not used

When assembling the parts of bipedal robot, different mating techniques are available. Mates create geometric relationships between assembled components. By adding mates,

allowable directions of linear or rotational motion of the components are defined. Then the components can be moved within its degrees of freedom and be able to visualize the assembly's behavior. The most common or widely used mating are:-

- **Concentric mate** between cylindrical hinge surfaces of the two parts. This makes the degree of freedom to two. One rotational and one translational along the cylindrical axis.
- **Coincident mate** between two parts will remove the translational degree of freedom and left with the rotational degree of freedom.

N.B:-The applied mating type will also decide the number of degree of freedom. Applying the above techniques in SOLIDWORK, the humanoid robot looks like this:-

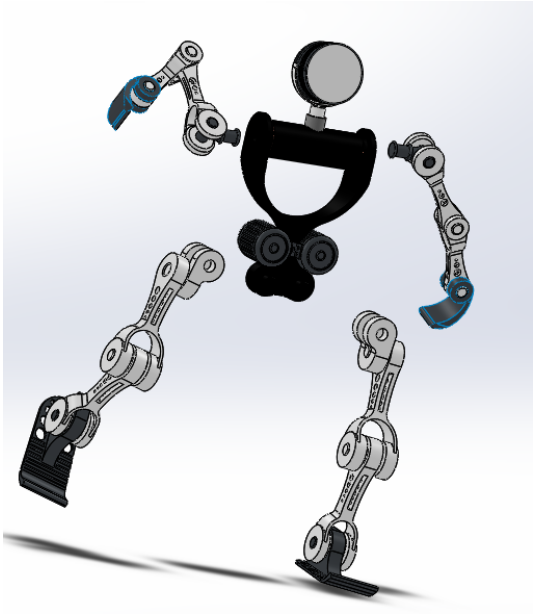

Figure 3.4: Assembled humanoid Robot

Values for each parts of Humanoid robot is:

Part of bipedal robot		Length	Mass
Arm	Upper arm	0.12m	0.16kg
	Lower arm	0.09m	0.15kg
Shoulder swive		-	0.03kg
Gripper		0.06m	0.08kg
Leg	lower leg	0.16m	0.2kg
	Upper leg	0.14m	0.22kg
Foot	Z-axis	0.03m	0.18kg
	Y-axis	0.1m	
Hip swive		-	0.04kg
Head		0.06m(diameter)	0.2kg
Neck		0.06m	0.12kg
Torso		0.18m	1.98kg
Shoulder		0.53m(width)	1.06kg
Total		0.63m	4.42kg

Table 3.1: Values of humanoid Robot

3.1.2 Denavit and Heartenberg

A systematic method for selecting coordinate frames for an articulated serial manipulator. DH parameters provide not only offers a systematic approach for selecting coordinate frames, but also established a framework for simplify forward kinematic analysis. DH parameters facilities the creation of homogeneous transformation matrices,which are crucial.

- Assign Z_i along the axis of joint i.
  - For a revolute joint, the joint axis is along the axis of rotation.
  - For a prismatic joint, the joint axis is along the axis of translation.

- Choose X_i to indicate along the shared perpendicular of Z_i and Z_{i+1} pointing towards the next joint.
  - if Z_i and Z_{i+1} intersect, then choose Xi to be normal to the plane of intersection.
- Choose Y_i to round out a right hand coordinate system.
  - The Y-axis is not used for Denavit Hartenberg so it is usually not drawn in the interest of less clutter.

DH parameters

- a_{i-1}: distance from z_{i-1} to z_i along x_{i-1}
- α_{i-1}:angle from z_{i-1} to z_i about x_i
- d_i:distance from x_{i-1} to x_i along z_{i-1}
- θ_i:angle from x_{i-1} to x_i about z_i

Assigning DH coordinate frame for bipedal robot:

- The $z - axis$ is to direction of rotation in revolute joint.
- The $x - axis$ must be perpendicular to both current $z - axis$ and previous $z - axis$.
- the $y - axis$ is determined form x and z axis in right hand rule.
- The $x - axis$ must intersect the previous $z - axis$ (does not apply the frame 0)

Joint	Dof
Head	2
Shoulder	2
Elbow	1
Wrist	1
Hip	2
Knee	1
Ankle	1
Total	18

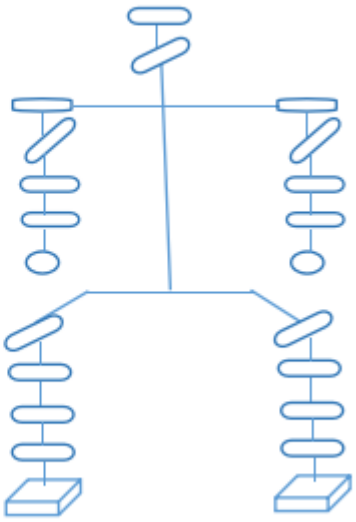

Figure 3.5: Overview of humanoid robot

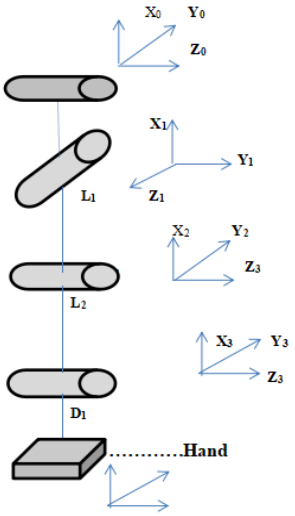

Figure 3.6: Arm for humanoid robot

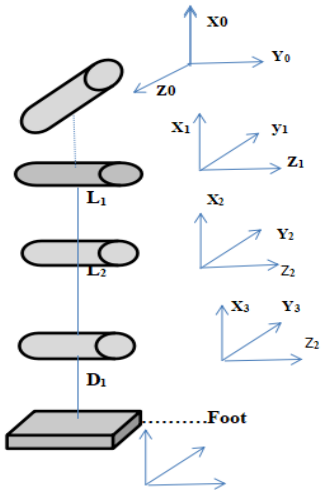

Figure 3.7: Foot for humanoid robot

Joint	a	α	d	θ
1	0	90	D_1	θ_1
2	L_1	90	0	θ_2
3	L_2	0	0	θ_3
4	A_1	0	0	θ_4

Table 3.2: DH parameter Of humanoid robot arm

Joint	a	α	d	θ
1	0	90	0	θ_5
2	L_1	0	0	θ_6
3	L_2	0	0	θ_7
4	A_1	0	0	θ_8

Table 3.3: DH parameter for humanoid robot leg

3.1.3 Forward And Inverse Kinematics

[32]Kinematics of a bipedal robot involves the geometric aspect of its motion in relation to a stationary reference coordinate system, disregarding the impact of the force.

A basic kinematic model is characterized by the length of its rigid segment, the degrees of freedom of its joints, their upper and lower joint limits and a hierarchical structure of the segments and joints. Each joints retains the current rotation applied to respective joint. Kinematic analysis can be done in two ways namely forward and inverse manner.

In forward kinematics, the end-effector position can be calculated for a specific set of joint parameters, which includes link lengths and joint angles. By adjusting the theta values, the bipedal robot can be placed at the desired location. Inverse kinematics is a technique used to determine joint angles for a given end-effector position and orientation, based on the link lengths.

The key difference between forward and inverse kinematics lies in the input parameters. While forward kinematics use theta value to determine bipedal robot movement, inverse kinematics use x, y, and z coordinates to determine the necessary theta values for reaching the intended position. Both forward and inverse kinematics rely on DH parameters, as mentioned above.

Since the bipedal robot has 4 DOF on the both the leg and the arm, the inverse kinematics will almost be the same. The difference between the arm and the leg is that, for the arm: on the shoulder the first movement is sagittal movement then comes the frontal movement but for the leg: the first movement is the frontal then later the sagittal movement this is because the mating sequence in SOLIDWORK.

Forward Kinematics

Forward kinematics asks where the end effector of the bipedal robotic arm will be following a sequence of rotations of the joints of the arm. To find the forward kinematics, DH parameter convention is utilized, In this convention, each homogeneous transformation T_n is represented as a product of four basic transformations[33].

$$R_(z,\theta_n): \begin{bmatrix} cos(\theta_n) & -sin(\theta_n) & 0 & 0 \\ sin(\theta_n) & cos(\theta_n) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} Trans_(z,d_n): \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$Trans_(x,a_n): \begin{bmatrix} 1 & 0 & 0 & a_n \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} R_(x,\alpha_n): \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\alpha_n) & -sin(\alpha_n) & 0 \\ 0 & sin(\alpha_n) & cos(\theta_n) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$T_n = R_(z,\theta_n)*Trans_(z,d_n)*Trans_(x,a_n)*R_(x,\alpha_n)$

The transformation matrix will be:

$$\begin{bmatrix} cos(\theta_n) & -cos(\alpha_n)*sin(\theta_n) & sin(\alpha_n)*sin(\theta_n) & \alpha_n cos(\theta_n) \\ sin(\theta_n) & cos(\alpha_n)*sin(\theta_n) & -sin(\alpha_n)*cos(\theta_n) & \alpha_n sin(\theta_n) \\ 0 & sin(\alpha_n) & cos(\alpha_n) & D_n \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Forward kinematics for the arm of humanoid robot:
θ_1, θ_2, θ_3 and θ_4 are the joint parameters for joint 1, joint 2, joint 3 and joint 4 respectively for the arm of humanoid robot and by using the rest three the DH parameter(link length,link twist and link offset) value which is provided in the above table3.2, the forward kinematics can be calculated

$$T_1^0 = \begin{bmatrix} cos(\theta_1) & 0 & sin(\theta_1) & 0 \\ sin(\theta_1) & 0 & -cos(\theta_1) & 0 \\ 0 & 1 & 0 & D_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_2^1 = \begin{bmatrix} cos(\theta_2) & 0 & -sin(\theta_2) & L_1 * cos(\theta_2) \\ sin(\theta_2) & 0 & cos(\theta_2) & L_1 * sin(\theta_2) \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_3^2 = \begin{bmatrix} cos(\theta_3) & -sin(\theta_3) & 0 & L_2 * cos(\theta_3) \\ sin(\theta_3) & cos(\theta_3) & 0 & L_2 * sin(\theta_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_4^3 = \begin{bmatrix} cos(\theta_4) & -sin(\theta_4) & 0 & A_1 * cos(\theta_4) \\ sin(\theta_4) & cos(\theta_4) & 0 & A_1 * sin(\theta_4) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Then the final homogeneous matrix will be:

$$T_4^0 = T_1^0 * T_2^1 * T_3^2 * T_4^3 \tag{3.1}$$

Forward kinematics for the leg of humanoid robot:
As θ_5, θ_6, θ_7 and θ_8 are the joint parameters for joint 5, joint 6, joint 7 and joint 8 respectively for the leg of humanoid robot. By referring table3.3 and inserting the values of DH parameter in to the transformation matrix, the transformation matrix for leg can be calculated.

$$T_5^4 = \begin{bmatrix} cos(\theta_5) & 0 & sin(\theta_5) & 0 \\ sin(\theta_5) & 0 & cos(\theta_5) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_6^5 = \begin{bmatrix} cos(\theta_6) & -sin(\theta_6) & 0 & L_1 * cos(\theta_6) \\ sin(\theta_6) & cos(\theta_6) & 0 & L_1 * sin(\theta_6) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_7^6 = \begin{bmatrix} cos(\theta_7) & -sin(\theta_7) & 0 & L_2 * cos(\theta_7) \\ sin(\theta_7) & cos(\theta_7) & 0 & L_2 * sin(\theta_7) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_8^7 = \begin{bmatrix} cos(\theta_8) & 0 & sin(\theta_8) & A_1 * cos(\theta_8) \\ sin(\theta_8) & 0 & -cos(\theta_8) & A_1 * sin(\theta_8) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Then the final homogeneous matrix will be:

$$T_8^4 = T_5^0 * T_6^5 * T_7^6 * T_8^7 \tag{3.2}$$

Inverse kinematics

Inverse kinematics asks what rotations of the joints will bring the end effector to a specified position. To calculate the inverse kinematics, the transformation matrix calculated in the above is used and by elimination method and equating the variable between matrix's the angle needed or required is found.

To calculate the inverse kinematics, the above equation 3.1 is used. First lets introduce T_e

Where T_e=T_4^0 is:

$$\begin{bmatrix} n_x & O_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Inverse kinematics for the arm of humanoid robot:

lets bring the T_1^0 to the left side and multiply it with T_4^0 in the equation of 3.1, after that equate the two matrix.

$$T_e * \frac{1}{(T_1^0)} = T_2^1 * T_3^2 * T_4^3 \tag{3.3}$$

$T_e * \frac{1}{(T_1^0)}$=

$$\begin{bmatrix} n_x cos(\theta_1) + n_y sin(\theta_1) & 0_x cos(\theta_1) + o_y sin(\theta_1) & a_x cos(\theta_1) + a_y sin(\theta_1) & p_x cos(\theta_1) + p_y sin(\theta_1) \\ -n_z & -o_z & -a_z & -p_z \\ n_y cos(\theta_1) - n_x sin(\theta_1) & o_y cos(\theta_1) - o_x sin(\theta_1) & a_y cos(\theta_1) - a_x sin(\theta_1) & p_y cos(\theta_1) - p_x sin(\theta_1) \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

and then $T_2^1 * T_3^2 * T_4^3$=

$$\begin{bmatrix} cos(\theta_4)cos(\theta_{23}) + sin(\theta_4)sin(\theta_{23}) & 0 & cos(\theta_4)cos(\theta_{23}) + a_y sin(\theta_1) & 0 \\ cos(\theta_4)sin(\theta_{23}) + cos(\theta_3)sin(\theta_2) + sin(\theta_4)cos(\theta_{23}) & 0 & sin(\theta_4)sin(\theta_{23}) - cos(\theta_4)cos(\theta_{23}) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Then equating the above two matrix:

$p_y cos(\theta_1) - p_x sin(\theta_1)$=0 .....$tan(\theta_1) = \frac{p_y}{p_x}$

$\theta_1 = \tan \text{inverse} \frac{p_y}{p_x}$

After that, lets bring T_2^1 to the left side of the equation3.3 and equate the two matrices.

$$T_4^0 * \frac{1}{T_1^0} * \frac{1}{T_2^1} = T_3^2 * T_4^3 \tag{3.4}$$

$T_4^0 * \frac{1}{T_1^0} * \frac{1}{T_2^1}$=

$$\begin{bmatrix} n_x cos(\theta_{12}) + n_y sin(\theta_{12}) & o_x cos(\theta_{12}) + o_y sin(\theta_{12}) & a_x cos(\theta_{12}) + a_y sin(\theta_{12}) & p_x cos(\theta_{12}) + p_y sin(\theta_{12}) - L_1 cos(\theta_1) \\ -n_z & -o_z & -a_z & -p_z \\ n_y cos(\theta_{12}) - n_x sin(\theta_{12}) & o_y cos(\theta_{12}) - o_x sin(\theta_{12}) & a_y cos(\theta_{12}) - a_x sin(\theta_{12}) & p_y cos(\theta_{12}) - p_x sin(\theta_{12}) + L_1 sin(\theta_1)) \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

And T_3^2*T_4^3=

$$\begin{bmatrix} cos(\theta_{34}) & 0 & sin(\theta_{34} & L_2cos(\theta_3) + L_3cos(\theta_3)cos(\theta_4) \\ sin(\theta_{34}) & 0 & -cos(\theta_{34}) & L_2sin(\theta_3) + L_3cos(\theta_4)sin(\theta_3) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

So by equating the matrices, we will get the value of the theta:

$O_xcos(\theta_{12}) + O_ysin(\theta_{12}) = 0$

Which gives: $O_xcos(\theta_{12}) = -(O_ysin(\theta_{12}))$

$-\frac{O_x}{O_y} = \tan(\theta_{12})$

$(\theta_{12}) = \tan\text{inverse}(\frac{-O_x}{O_Y})$

Then θ_2 will be:

$\theta_2 = \tan\text{inverse}(\frac{-O_x}{O_Y}) - (\theta_1)$

The last step will be T_3^2 to the left side of the equation3.4 and equate the two matrices.

T_4^0*$\frac{1}{T_1^0}$*$\frac{1}{T_2^1}$*$\frac{1}{T_3^2}$=T_4^3

And equate the two matrices:

$n_x((cos\theta_3)sin(\theta_{12})+sin(\theta_3)cos(\theta_{12}))+n_y(cos(\theta_3)cos(\theta_{12})-sin(\theta_3)sin(\theta_{12})) = 0$

$n_x(sin(\theta_{123})) + n_y(cos(\theta_{123})) = 0$

$\frac{n_y}{n_x}cos(\theta_{123}) = \frac{n_y}{n_x}sin(\theta_{123})$

$\tan\text{inverse}(\theta_{123}) = \frac{n_y}{n_x}$

θ_1+θ_2+θ_3=$\tan\text{inverse}\ \frac{n_y}{n_x}$

So θ_3 will be:

$\theta_3 = \tan\text{inverse}\ \frac{n_y}{n_x} - \theta_1 - \theta_2$ and θ_4 can be calculated from:

$sin(\theta_4) = a_xcos(\theta_3)cos(\theta_{12})-sin(\theta_3)cos(\theta_{12})+a_ycos(\theta_3)sin(\theta_{12})+sin(\theta_3)cos(\theta_{12})$

3.1.4 Dynamics of Humanoid Robot

Dynamics is the study of its motion and forces acting on it. It involves understanding how a robot's structure, actuators, and control inputs affect its movement and stability. The other advantage of using SOLIDWORK to design the bipedal robot is, the dynamics of the robot will also be available.

To determine the dynamics of the bipedal robot built in SOLIDWORK, utilize the SOLIDWORKs API (Application Programming Interface) to access and extract the necessary information such as joint velocities and joint torques. The API enables you to retrieve data and automate tasks from SOLIDWORK. For this paper the dynamics of the bipedal robot is not been included as the effect is small and can be neglected. Which is the torque value extracted from SOLIDWORK divided by gear value used is very small due to this the dynamics of the bipedal robot is not considered.

3.2 Model Verification

Before implementing any kind of controlling mechanism the exported model has to be verified first. So to check if joints or the revolt joints of the bipedal robot are accurately designed, different set of angles are given.

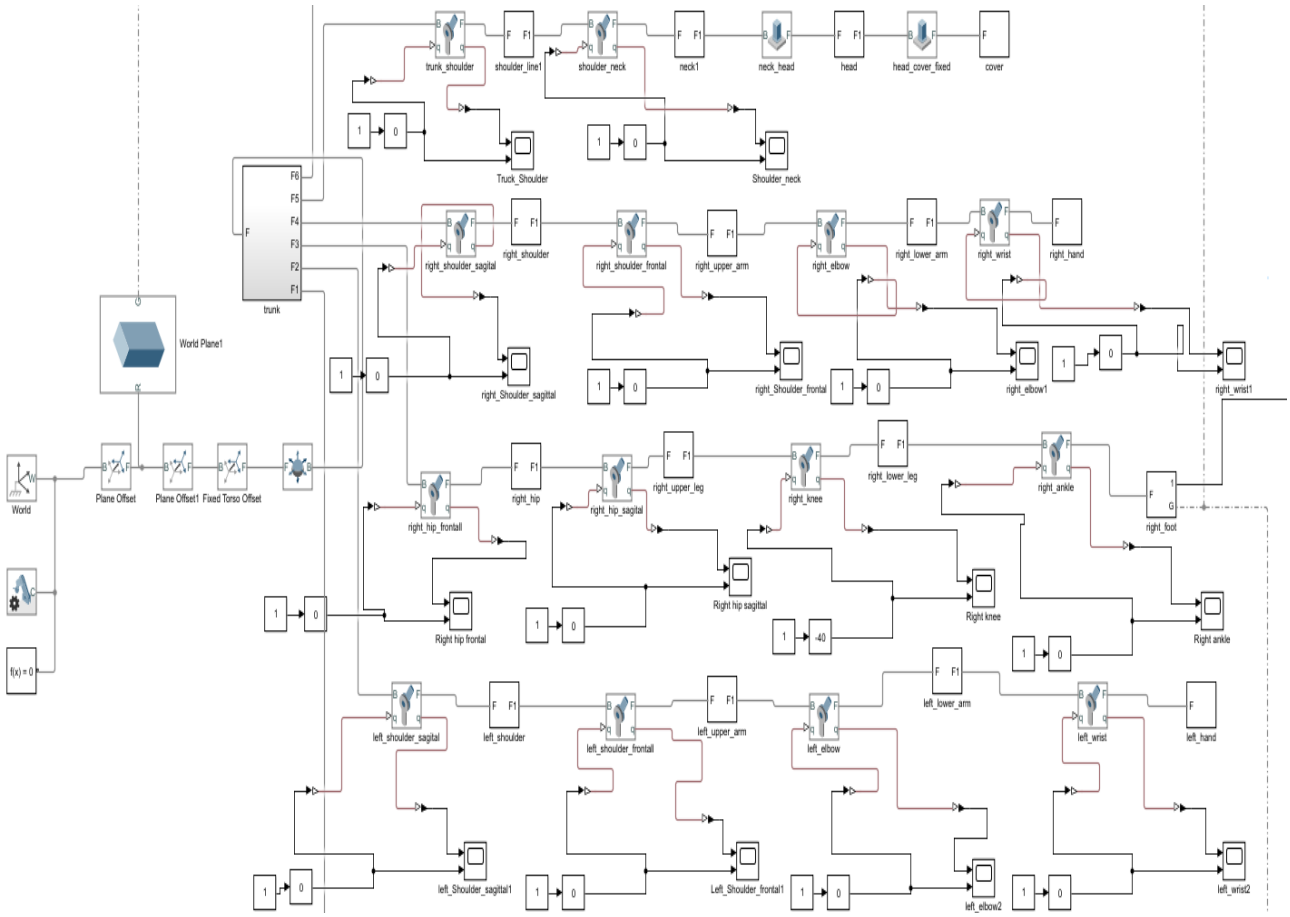

Figure 3.8: Model verification of humanoid robot

To export a 3D model of bipedal robot from SOLIDWORKS to MATLAB, Simscape Multibody Link needs to be added to MATLAB. This involves downloading the necessary files and installing them correctly. After installation, the Simscape Multibody Link option will be available in SOLIDWORKS, allowing for the export of assembly models.

The installation process involves running MATLAB as an administrator, adding the installation file to the MATLAB path, and registering MATLAB as an automation server. Once installed, the command "smlink_linksw" can be used to enable the connection between SOLIDWORKS and MATLAB.

After exporting the file from SOLIDWORKS to MATLAB, the file can be opened in MATLAB and different blocks will appear. The model block consists of three blocks: the solver configuration block, the mechanism configuration block, and the world frame. The solver configuration block is necessary for all models in Simscape, while the mechanism configuration block is used to assign the direction and magnitude of gravity.

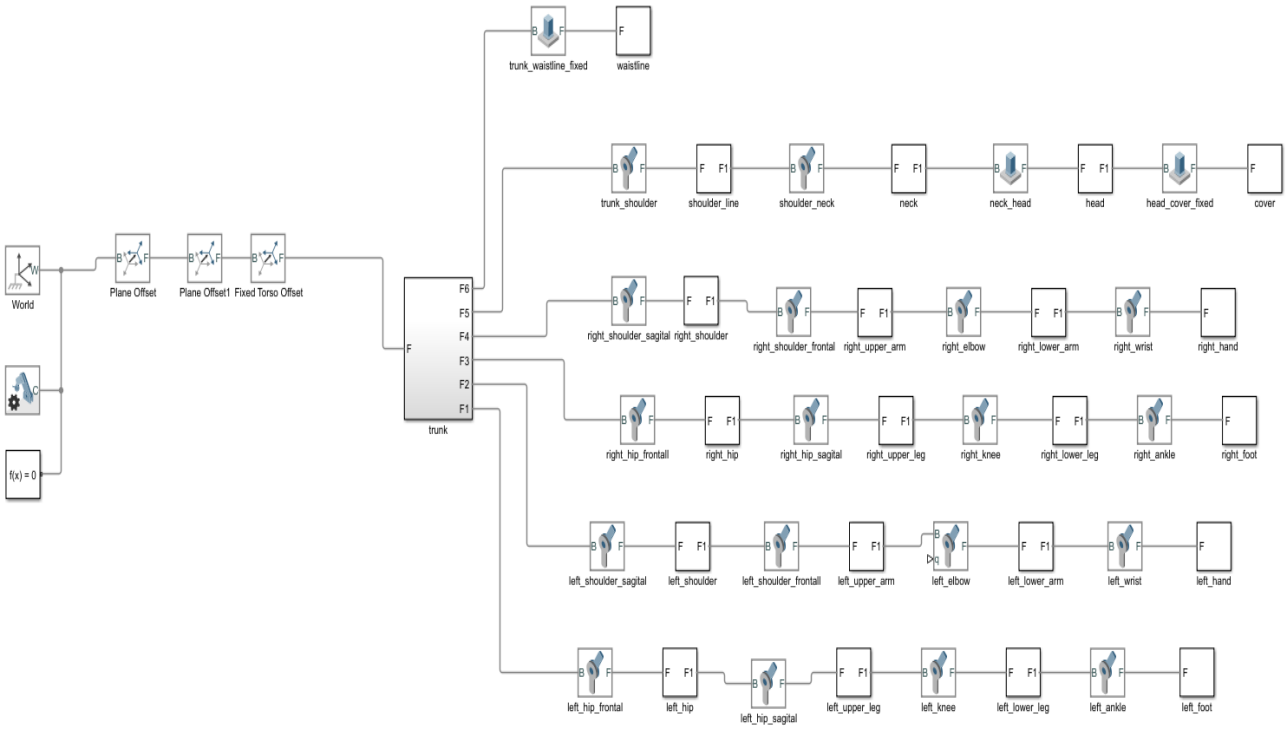

Figure 3.9: Assembly and part of Humanoid robot

After exporting the robot, when the model is simulated the movement of the robot is not organized so the first thing to do is to give no actuation value for all joints, so that bipedal robot will stand still. Then apply some theta value to all 18 joints step by step.

The next step is to check if the bipedal robot model complies with the given input. To convert the input signal into a format that the humanoid robot can understand, Simulink to physical converters are used. These converters ensure that the input signal is in the correct physical unit and can be properly interpreted by the robot. Additionally, a PS-Simulink converter is used to convert the physical output signal into a Simulink output signal. This allows for checking the output value of the model. To verify the correctness of the model, different inputs are given to the revolute joints of the bipedal robot. Actuation options for the revolute joints can be set to torque or motion.

A slider is used to give inputs to joints of the bipedal robot. As shown in the figure above3.8 ,not to leave the slider with no input and a constant block is added with the constant number 1 so that it wont have an effect.

Each graph has two line, the reference and the joint output line by seeing if the plotted output response matches the expected signal it indicates that the plant model is likely to be accurate. However, if there are significant differences, it suggests that the plant model may need to be refined or adjusted.The bellow graph suggest that the modeled bipedal robot is correct as the difference between the reference signal and output signal is minimal.

The below graph shows that the joint output follows the reference input :-

Two joints found in the neck(trunk shoulder and shoulder neck)

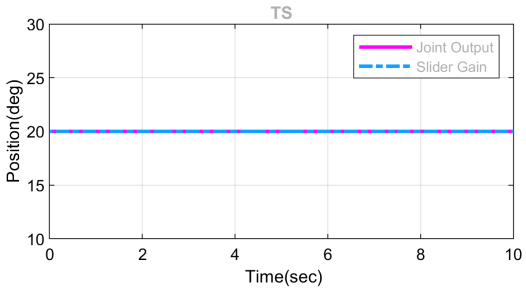

Figure 3.10: Trunk Shoulder

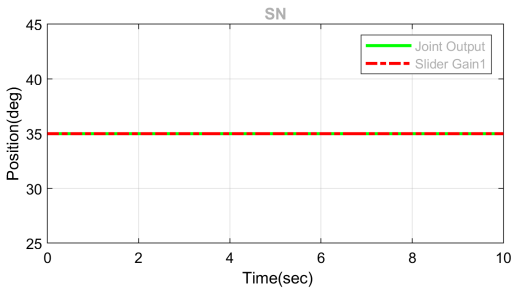

Figure 3.11: Shoulder Neck

The joints found in the right arm(right shoulder frontal, right shoulder sagittal, right elbow and right wrist):

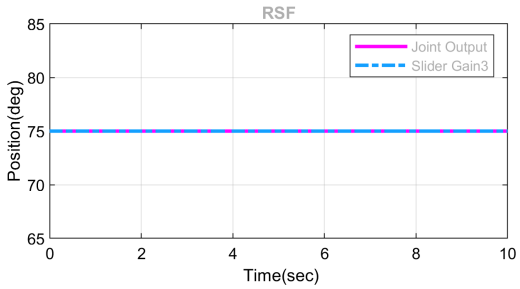

Figure 3.12: Right Shoulder Frontal

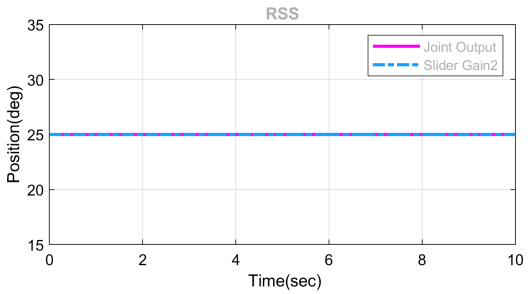

Figure 3.13: Right Shoulder Sagittal

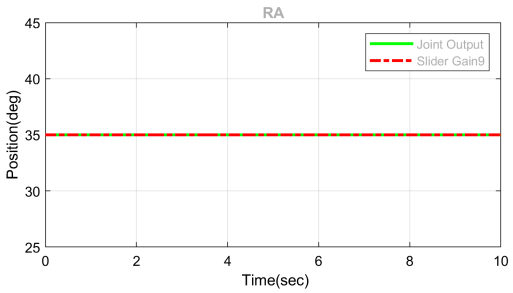

Figure 3.14: Right Elbow

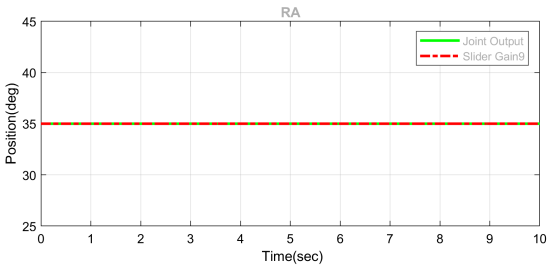

Figure 3.15: Right Wrist

The four joint found on left arm (left shoulder frontal, left shoulder sagittal, left elbow and left wrist):

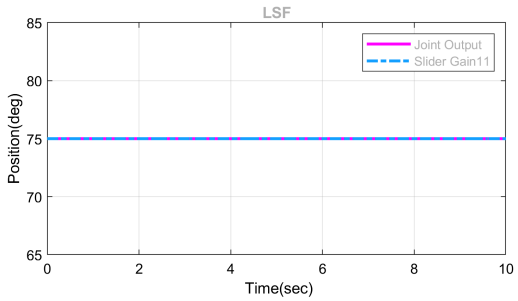

Figure 3.16: Left Shoulder Frontal

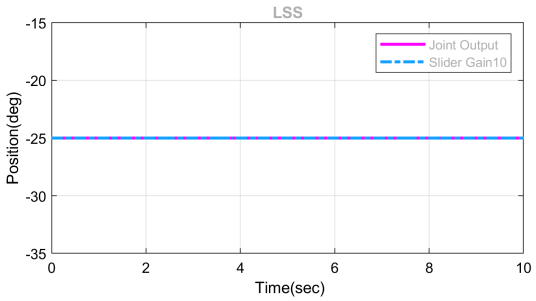

Figure 3.17: Left Shoulder Sagittal

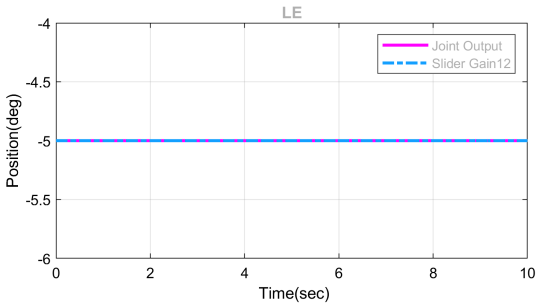

Figure 3.18: Left Elbow

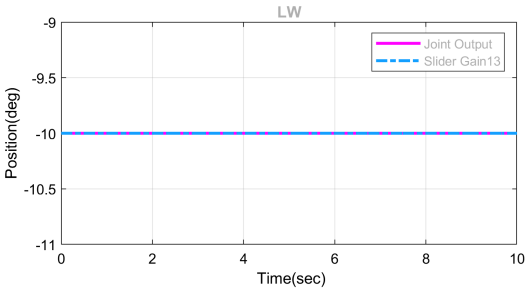

Figure 3.19: Left wrist

The joints found in the right leg(right hip frontal, right hip sagittal, right knee and right ankle):

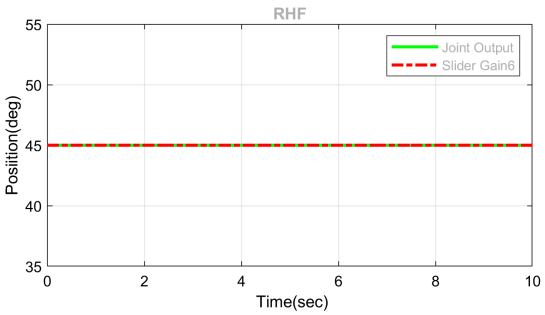

Figure 3.20: Right Hip Frontal

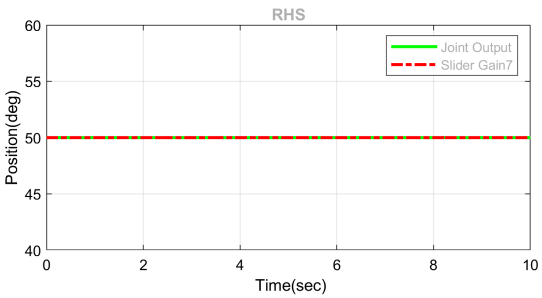

Figure 3.21: Right Hip Sagittal

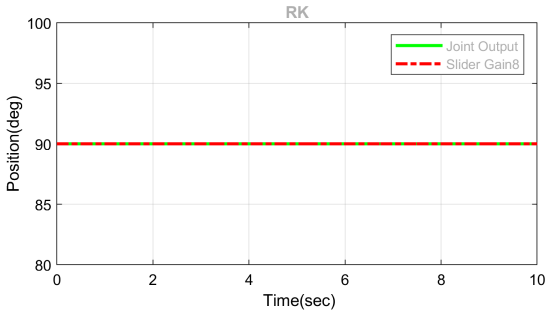

Figure 3.22: Right Knee

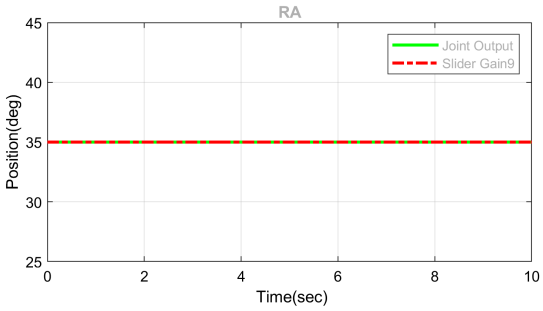

Figure 3.23: Right Ankle

The four joints found in the left leg(left hip frontal, left hip sagittal, left knee and left ankle):

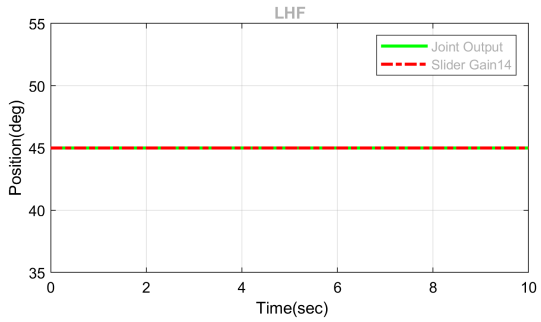

Figure 3.24: Left Hip Frontal

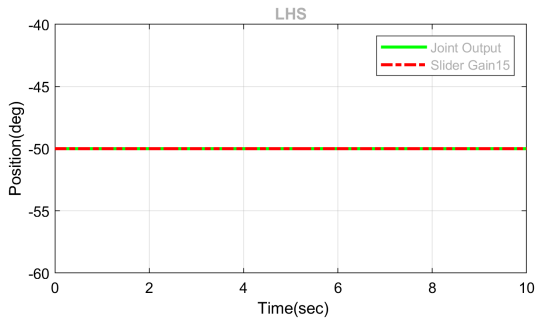

Figure 3.25: Left Hip Sagittal

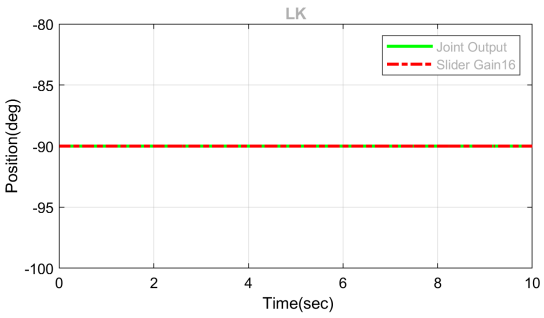

Figure 3.26: Left Knee

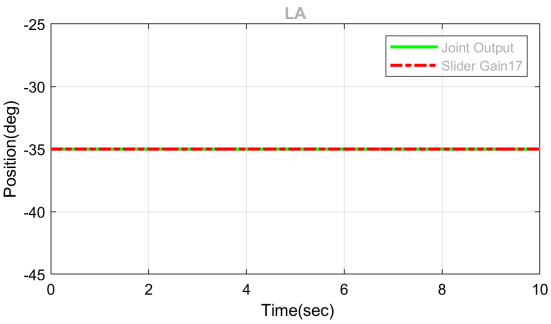

Figure 3.27: Left Ankle

And here are some examples to give some kind of visualization on how the actual bipedal robot looks like after different values of deg are given:-

Figure 3.28: 45 deg on both hip with frontal movement

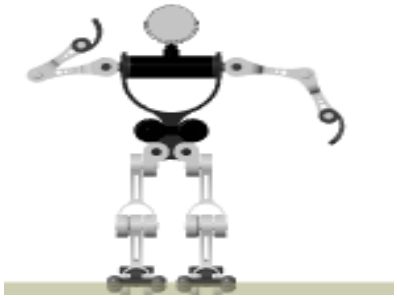

Figure 3.29: Different value of deg given to different arm joint

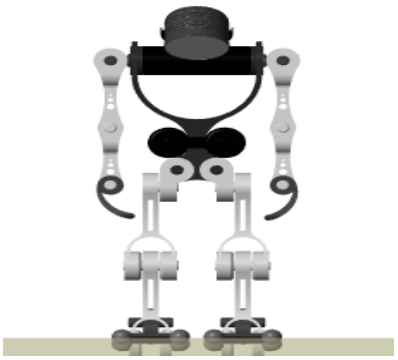

Figure 3.30: 45 deg on the trunk shoulder

Figure 3.31: 20 deg on shoulder neck

The above graph help us conclude that the exported model of bipedal robot from SOLIDWORK follows the input given. So the model is verified.

3.3 Motion Planning

This section discuss the general walking pattern of a bipedal robot. The bipedal robot should imitate the walking patter of human being's. Different set of inputs must be given to the revoulte joint .

The bipedal robot has to walk on surface, a given surface is imported from Simulink. To interact the plain(surface) with robot, Spatial contact force is utilized. This Spatial contact force is implemented on both foot of the bipedal robot.

The spatial contact force block is block found in MATLAB that is used to model the contact between two bodies. It utilize the built-in penalty method or custom force laws to

simulate a contact. This block supports a variety of bodies, including those created by a third-party CAD software or directly by multi-body blocks. However, some bodies, such as the point cloud and disk, are unable to make contact. The image below illustrates how the Spatial Contact Force block models the contact between a blue base geometry and a red follower geometry in spatial contact problem. .

In every contact, each geometry has contact frame. The two contact frames are always coincident and located at the contact point. The z-axis direction of the contact frame is an outward normal vector for the base geometry, but an inward normal vector for the follower geometry. During continuous contact, the contact frame moves around the geometry as the contact point moves.

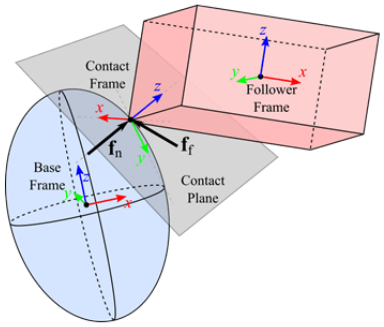

Figure 3.32: Spatial Contact Force

The spatial contact force block models the forces between the follower and base frames of solid bodies. The block properties are divided into three nodes: Normal force, Frictional force and sensing.

The normal force is the perpendicular force exerted by the surface when the object is affected by it, and it is based on penetration depth and velocity. Within normal force, stiffness, damping and transitional regional width has to be specified. Stiffness is defined as the force required to produce unit displacement in the direction of the applied force and damping is a force that resists the fast change in displacement and comes to rest.

$$sttifness = \frac{\text{the weight of the robot}}{\text{displacement}} \tag{3.5}$$

weight= mass * gravity, displacement= any acceptable overlap between the foot of the robot and the ground (a few mm) and damping = spring constant (mostly go 1 or 2 bellow spring constant)

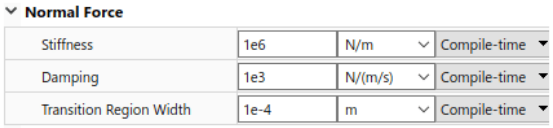

Figure 3.33: Normal Force On Spatial Contact Force

The other thing to worry about is the friction force, it is determined by the normal force and the value depends on the contact surface utilized. When determining frictional force, different parameters should be considered, such as the coefficient of static friction, coefficient of dynamic friction, and critical velocity. These parameters are set depending on the method used, if smooth stick-slip is enabled or not. When determining the values of the static and dynamic coefficients, it is important to first determine the type of contact and then identify the static and dynamic coefficient. The bipedal robot is constructed with aluminum material. The static coefficient of friction measures the amount of friction between two surfaces which are at rest . To initiate motion, the static coefficient friction must be overcome, while the dynamic coefficient determines the amount of friction on wet, level floors when walked upon. The results of the dynamic coefficient of friction are essential in assessing the potential risk of slipping and falling on tiled surface.

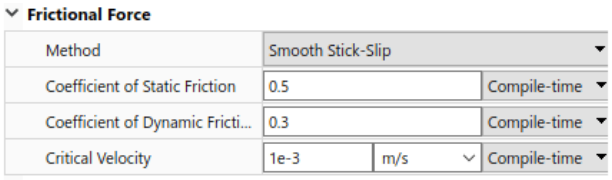

Figure 3.34: Frictional Force On Spatial Contact Force

Sensing in spatial contact force is used to sense.(Besides modeling the contact body)

- The value of separation distance between the two bodies.
- The value of the normal force exerted by each solid body on the other.

As discussed in the first chapter, the joint itself cannot be actuated, therefore an actuator is required. The servo motor is utilized as an actuator.

3.4 Modeling Of Servo Motor

Equivalent model of DC motor is shown[34].

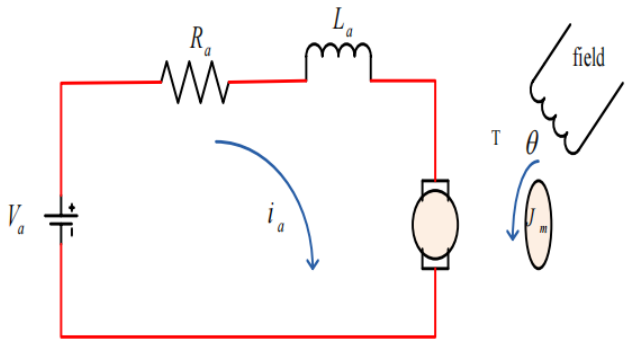

Figure 3.35: Equivalent model of DC servo motor

Where

Va = armature voltage(volt)

Ra = armature resistance(ohm)

La = armature inductance(H)

Ia = armature current(A)

T = motor torque(Nm)

Θ_r = reference angular position of rotor shaft(rad)

θ = angular position of rotor shaft(rad)

J_m = Moment of inertia of motor(kg/m2)

B = viscous friction constant of the motor

K_t = motor torque constant

K_e = electromotive force constant

Using Kirchhoff's voltage law, the armature voltage motor is given by:-

$$V_a = R_a i_a + L_a \frac{di_a}{d_t} + E_b \tag{3.6}$$

Motor torque related to the armature current is given by:

$$T = K_t i_a \tag{3.7}$$

The back emf of the motor is given by:

$$E_b = K_e \frac{d\theta}{d_t} \tag{3.8}$$

According to the newton's law and Kirchhoff's law to the motor system:

$$J_m \frac{d^2\theta}{dt^2} = T - B\frac{d\theta}{dt} \tag{3.9}$$

$$\frac{d\theta}{dt} = \omega \tag{3.10}$$

The three variables will be $x_1 = \theta - \theta_r$, $x_2 = \omega$ and $x_3 = i_a$. Then taking the derivatives the equation will be:-

By referring 3.10.....$\dot{x_1} = \omega = x_2$

By referring 3.9 .....$\dot{x_2} = \frac{K_m}{J} x_3$

By referring 3.6 .....$\dot{x_3} = \frac{V_a}{L_a} - \frac{E_b}{L_a} - \frac{R_a}{L_a x_3}$,

$$\frac{V_a}{L_a} - \frac{K_e}{L_a}\omega - \frac{R_a}{L_a}x_3$$

$$\frac{V_a}{L_a} - \frac{K_e}{L_a}x_2 - \frac{R_a}{L_a}x_3$$

Selecting the state variable:

$$x = [x_1, x_2, x_3]^T = [\theta, \omega, i_a]^T$$

Out put variable:

$$y = [x_1, 0, 0] = [x_1, 0, 0]$$

Control input matrix:

$$U = [V_a]$$

Then writing the standard state-space expression form as follows:

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

$$\dot{y} = Cx + Du$$

$$A_{3,3} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & \frac{K_t}{J_m} \\ 0 & -\frac{K_e}{L_a} & -\frac{R_a}{L_a} \end{bmatrix}$$

$$B_{3,1} = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L_a} \end{bmatrix}$$

$$C_{1,3} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

$$D_{1,1} = \begin{bmatrix} 0 \end{bmatrix}$$

To realize the optimal control of servo motor position tracking system, observability,controllability and stability of the established motor dynamics must be analyzed first.

The observability is judged by judging whether the observability matrix is full rank and observability discriminant matrix is obtained by the formula:

$$N_{3,1} = \begin{bmatrix} C \\ CA \\ CA^2 \end{bmatrix}$$

$$N_{3,3} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \frac{K_t}{J_m} \end{bmatrix}$$

Since the value of K_t is different from zero, observability discriminant matrix is full rank hence the system is observable.

The controllability is judged by judging whether the controllability matrix is full rank and observability discriminant matrix is obtained by the formula:

$$M_{1,3} = \begin{bmatrix} B & AB & A^2B \end{bmatrix}$$

$$\begin{vmatrix} 0 & 0 & \frac{K_t}{L_a J_m} \\ 0 & \frac{K_t}{L_a J_m} & -\frac{K_t R_a}{L_a^2 J_m} \\ \frac{1}{L_a} & -\frac{R_a}{L_a^2} & \frac{R_a^2}{L_a^3} - \frac{k_t K_e}{L_a^2 J_m} \end{vmatrix}$$

Since the value of K_t is different from zero, controllability discriminant matrix is full rank hence the system is controllable.

Stability is determined by whether the A matrix has negative eigenvalues. If any eigenvalue has a negative real part, the system will tend to return to a steady state (stable system).

The eigen values are

$$\lambda_1 = 0, \lambda_2 = -3.4, \lambda_3 = -0.6$$

.

If an eigenvalue has a positive real part, the system will tend to move away from the fixed point (unstable system). If an eigenvalue has a negative real part, the system will tend to move back to steady state (stable system). If an eigenvalue has an imaginary part, the system will oscillate around the steady state. If one eigenvalue is zero and the others are negative, then the system will be marginally stable.

Chapter 4

Controller Design for Servo Motor

This chapter provides general introduction to robot walking motion, model predictive controller, linear quadratic controller, mathematical formulation and overall design procedure.

4.1 Introduction To MPC Controller

In control system the controller's objective is to calculate the plant's input in order to guarantee that the plant's output adheres to a desired reference point. Model predictive control(MPC) employs a system model to predict the future behaviour of the system. MPC utilizes an online optimization algorithm to identify the optimal control action that aligns the predicted output with the reference point. MPC is able to handle systems with multiple inputs and outputs that may exhibit interactions, as well as managing input and output constraints. In addition, MPC own preview capability, letting it to incorporate future reference point information into the control problem to enhance controller performance. .

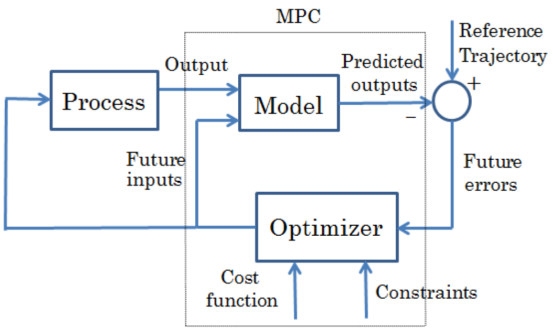

Figure 4.1: Over all MPC structure

4.1.1 Working Principle Of MPC

In 4.2, the control action u and x is presented as the previous evolved value which is before k. At given point, where the system is at k and the objective is to make a new action or a new decision so that the state x in time will reach a desired reference point(x_r). Hence MPC examines the next N time steps to determine the optimal or best series of control action.4.3 .

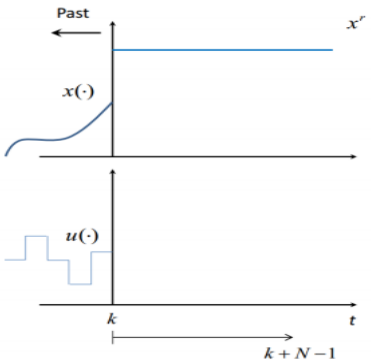

Figure 4.2: U and X in the past

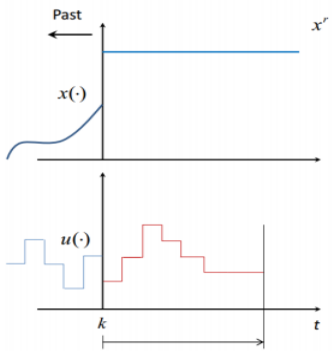

Figure 4.3: Predicted optimum Control action

Then it can be used in the system for the states to reach or approach the reference state value 4.4.

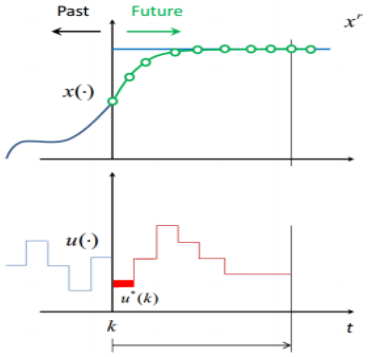

Figure 4.4: Next Step Prediction

As MPC's capability is to utilize online optimization, so at this point where it is after k the decision for optimization problem is solved which helps maintain optimal series of control actions to achieve the predicted state value to reach the desired state.

After the optimal series of control action and the predicted state values are obtained, MPC will first apply the first series of control action with out taking account the rest control action. 4.5. Since control input is applied to the plant, we can observe some changes

to the plant output. And unfortunately the plant out put might not be as intended. This incident might be due to different reasons such as uncertainties or unmeasured disturbance.

Thus, because of this factor , MPC will implement the second step subsequent to executing the initial control action 4.6. The MPC repeats this process to approach the reference, but not randomly;instead, it follows a systematic approach, where the optimizer plays a important role. The disturbance and model inaccuracies, both of which can cause the system output to diverge from the model's prediction.

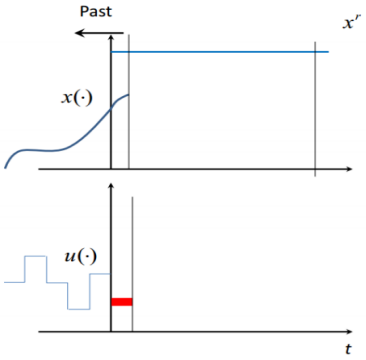

Figure 4.5: Actual state

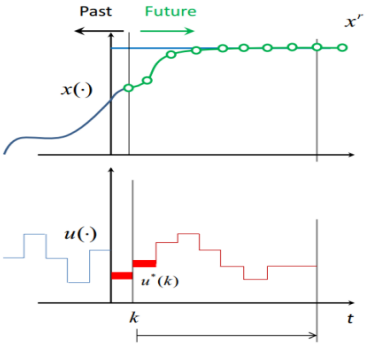

Figure 4.6: Next step prediction

In order to summarize:

- Prediction: able the state to evolve or acts:
- The online optimization helps decrease the difference between the reference/ desired sate and prediction by manipulating the control actions;
- Take the prediction horizon one step forward and repeat the process

4.1.2 MPC Parameter Design

The parameters utilized in MPC, affects both computational complexity and controller performance. In MPC algorithm the computation complexity involves with online optimization problem for each time step. Hence selecting appropriate parameters is crucial.

Prediction Horizon(N): MPC's predictive ability is calculated by the extent of its future time step converge, also known as the prediction horizon. This gives to the terms receding or moving horizon control. A shorter prediction horizon may result in failure to anticipate future events, while a longer one may miss disturbances. To ensure coverage of significant system dynamics, the prediction horizon must be carefully selected.

Control Horizon(M): The control action taken for each time step, denoted as M, is important. Each control moves within the control horizon is akin to free variable that requires computation. A smaller control horizon implies reduced computation. Opting for a control horizon of 1 may not yield the optimal result. By enhancing M, the controller becomes more assertive, demanding greater computational effort, but it also facilitates better predictions. A suitable approach is to set M at 10 to 20% of the prediction horizon(N), where $0.1\star N \leq M \leq 0.2\star N$.

Sample time(Ts): It is rate at which the time t executes the control algorithm. If the time is too big the controller wont react to the external disturbance fast enough and if the time is too small there will be computational load but the controller will be able to react to the external disturbance. The aim should be to find the right value that can balance the computation effort with the performance.

Weight(Q and R) The whole process is used so that the plant output follows the reference or desired values as close as possible but at the same time there should be smooth control moves.

- Q: this term or value measures convergence rate that includes settineg time and rise time:
- R: focus on computation load or penalizes assertive utilization of the input; If Q » R, when this happens the focus is given for the plant to track the desired or reference point or to decrease the deference between the plant output and reference than penalizing the input.

If R » Q, means it is more preferred to decrease the control effort as much as possible. The value of Q and R are used from previous work and try and error.

4.1.3 MPC Design For Servo Motor

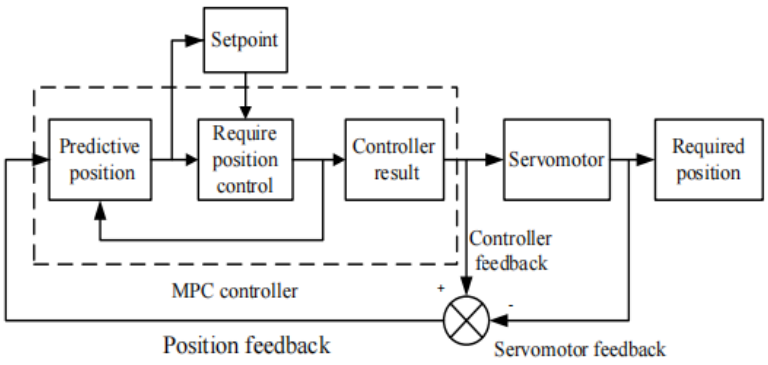

Figure 4.7: Block diagram of MPC servomotor controller

In here, the aim will be to control the position of the servo motor which consequently actuates the revolute joints. There are various techniques to control the position of a servo motor. This paper uses MPC controller toolbox in SIMULINK.

The MPC Toolbox is foundation of commands that is developed for the usage of analyzing and designing MPC systems. All MPC Toolbox has built-in display. The analysis and the simulation of MPC algorithm demands 1MB of memory relaying on the number of inputs and outputs as it is numerically intensive. Hence memory availability might limit the size of the system handled by the MPC Toolbox.

Model Predictive Controller (MPC) controls the position of servomotor depending on the applied voltage and parameters selected such as control horizon, prediction horizon and wights(Q and R). Before the controller is used, A stable servo system must be calculated which is already checked.

First servo motor specification are calculated, and this specification are changed into state space. The state space model is more suitable for muti-input multi-output systems. Since its is an analogue system it has to be changed into discrete system. Also the system has be checked for stability and to ensure stability trial and error method can be utilized.

To write the mathematical model for a discrete time system linear matrix difference equation can be utilized which can written in recursive formula:

$$x((k+1)) = Ax(k) + Bu(k)$$

$$y(k) = Cx(k) + Du(k)$$

Where x(k) an dimensional state vector at time t=kT, r-dimensional control vector u(K) and m- dimensional output vector y(K). Where T represents the constant sampling interval.

$$x(k) = [x_1(k), x_2(k), x_3(k).....x_n(k)]^T$$
$$u(k) = [u_1(k), u_2(k), u_3(k).....u_r(k)]^T$$
$$y(k) = [y_1(k), y_2(k), y_3(k).....y_m(k)]^T$$

MATLAB will change the continuous state space equation to discretize state space equation: sys = ss(A,B,C,D,Ts).....Ts sampling time(in seconds).

To linearize the plant or the servo motor, there is no need to do it by hand since the MPC toolbox will take care of it. After the simulation MPC toolbox with GUI, choosing prediction horizon, sample time , control horizon and weights(Q and R) will be the next

step.

The most important thing when designing a controller is assigning the right or appropriate value for the parameters of MPC. To assign for the value of N(prediction horizon , M(control horizon) and sample time, an open loop system of a servo motor is implemented and for the value of Q and R , PSO is used.

The Particle Swarm Optimization[35] (PSO) algorithm was proposed by Eberhart and Kennedy in 1995 and is was initiated by fish schooling and bird flocking. PSO is population based optimization method where the particles moves around in order to search a space with optimal solution and every particle is considered as potential solution. Each particle has its own velocity and position, the position presents the candidate solution and the velocity determines the speed of the movement and its direction .

The particles in the population move towards the best solution found so far, which is called the global best position. There is also a thing called a personal best position, which is the best solution found by an individual particle. The movement of particles is influenced by their personal best position, global best position, and their own velocity [36].

The algorithm starts by randomly initializing the population of particles randomly in the search space. Then, it iteratively updates the position and velocity of every particle based on the following equations:

Velocity update:

$$v_i(t+1) = w * v_i(t) + c1 * r1 * (p_i(t) - x_i(t)) + c2 * r2 * (g(t) - x_i(t)) \tag{4.1}$$

Position update:

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{4.2}$$

where v_i(t) is the velocity of particle i at time t, x_i(t) is the position of particle i at time t, p_i(t) is the personal best position of particle i at time t, g(t) is the global best position at time t, w is the inertia weight, c1 and c2 are acceleration coefficients, and r1 and r2 are random values between 0 and 1.

The algorithm continues for a predefined number of iterations or until a termination criterion is met, such as reaching a satisfactory solution or running out of computational resources.

After choosing the values for Q and R, to find the appropriate value for the prediction horizon and control horizon, first consider the following factors:

System dynamics: Examine the time constant and response time of the servo motor system. The prediction horizon should be long enough to capture these dynamics. To determine the dominant time constant of the system, analyze the step response of the system and identify the time it takes for the response to reach approximately 63% of its final value (also known as the time constant). This can be done using the "stepinfo" function in MATLAB.

Control objectives: Analyse the desired control objectives such as settling time, overshoot, or tracking accuracy. A longer prediction horizon can help achieve better performance for these objectives.

Computational constraints: Study the computational resources available for the MPC implementation. Longer prediction horizons require more computational effort.

The recommendation for choosing the sample time is to have 10-20 samples within the rise time of the open loop system response and for prediction horizon the recommendation is to have 20-30 samples covering the open loop transient system response and when it comes to control horizon, aligning it with the prediction horizon will only add complexity. This is because the initial move has a significant impact, while the subsequent ones have minimal effects. Therefore, it is advisable to limit the control horizon to 10-20% of the prediction horizon.

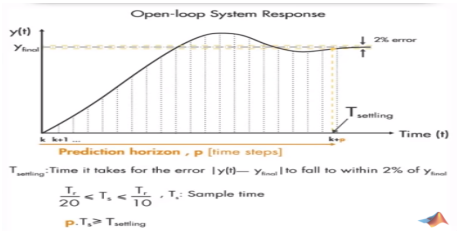

Figure 4.8: Selecting sample time

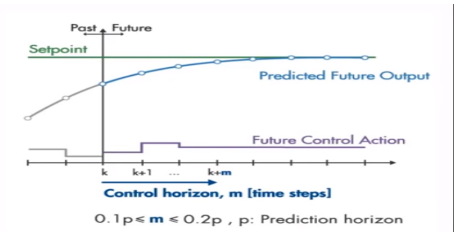

Figure 4.9: Selecting prediction horizon

Experimental tuning: Perform simulations or experiments with different prediction horizon values and evaluate the control performance for each case. Choose the value that provides the best trade-off between performance and computational complexity.

The MPC GUI model is used to set the controller parameters and shows whether the system is stable or not stable by visual perception of the output scenario under disturbance or under normal condition. First the MPC must be declare in the work space and import to the MPC GUI in Simulink toolbox, the same goes to the plant(servo motor).

The MPC Toolbox includes a built-in QP(Quadratic Programming) solver for solv-

ing optimization problem with quadratic objectives and linear constraints. It is frequently utilized to determine the optimal control inputs for a given system model. The exact implementation of QP solver in the MPC Toolbox may vary depending on the software or programming language being used.

QP Solvers

- Interior-point solver: This is mostly applicable for large scale optimization as it gives a better performance. It is the same with MPC system where MPC enforces constraints over extensive control and prediction horizon. By the help of Mehrotra predictor-corrector, it implements primal-dual algorithm. The solver has to set Optimizer.algorithm. Property into interior-point in order to utilize interior-point solver and for assembling purpose utilize the Optimizer InteriorPointOptions Property.

- The active solver: is a reliable and speedy option for tackling optimization problems of small to medium scale in both single and double precision. It implements the KWIK algorithm and set into active-set in Optimizer Algorithm property of your MPC controller for utilization purpose. To customize the algorithm setting, simply use the Optimizer.ActiveSetOptions property of your controller.

To define a custom solver for code generation or for simulation. In either instance, defining the custom solver involves using a custom function and configuring the controller function.

4.2 Introduction To LQR

Linear quadratic regulator(LQR) is familiar technique which provides optimal feedback gain for designing closed-loop stable and high performance systems. One of the advantage of LQR algorithm is that when optimizing the controller it reduces the workload for the control system engineer. However, the engineer still must specify the parameters for the cost function and compare the results with the design reference. After a while this process became iterative in which the engineer evaluates the optimal controllers produced through simulation and adjusts the parameters to better align with design goals.

The search for suitable state feedback is automated by using LQR, however some engineers choose other method which provide distinct relation between the parameter and controller behaviour such as pole placement. The effectiveness of LQR controller is restricted by the difficulty of calculating the suitable weighting factors .

For controlling the position of servo motor, LQR is recommended. For the particular

case, to calculate the controller parameter value specifically for the joints of bipedal robot a mathematical formula is utilized.Q and R, which are the weighting matrix is determined by the engineer and the algorithm minimizes a cost function typically expressed as a collection of deviation between their referenced value and crucial measurement.

4.2.1 LQR Design for Servo Motor

Consider for all the states to be measurable and the aim to find the state variable feedback control that gives a preferable closed loop performance. The initial state is x(0).

The control signal will be:

$$U = -kx \tag{4.3}$$

The equation **??** is control law that provides optimal value. Hence, if the value for the unknown matrix k is found that minimizes the performance index, then u= -kx is optimal for every initial state x(0). The block diagram below depicting the optimal configuration:

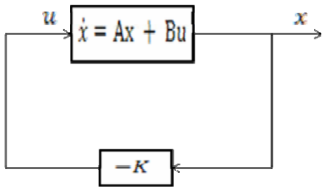

Figure 4.10: Control signal for LQR design

The cost function will be:-

$$J = lim_{t\to\infty} x^T Q x(t) + u^T R u(t) dt \tag{4.4}$$

The weight matrix Q(nxn) and R(mxm) are selected by the engineer. The values selected for these parameters will affect the closed loop output values. If the value for Q is large, the state x(t) must be kept smaller in order to make the cost function small and also if the value for R is large, the control value u(t) must be kept smaller as to make the cost function small. Both Q and R are positive definite Hermitian or real symmetric matrices.

When we try to see the implication of the value of Q, if the value is large, the poles of the closed loop $Ac = (A - BK)$ system will shift towards the s-plane and this will affect the state to decay more rapidly to zero. Comparatively when R is large, the control effort being utilized is less, there will slower poles and large state x(t) values. Q and R

matrix's imply the importance of the error and energy expenditure. It is very important to note that the second term on right side of the equation defines the energy expenditure of the control signal.

As the Q is positive semi definite and R to be positive definite, the scalar value for X^TQx is consistently non negative or zero each time for all function x(t) and at the same time the scalar value U^TRu is consistency non negative each time for all values f u(t). This makes the cost unction to be well defined.

When considering the eigen values of both Q and R matrix's, it must be positive and if both matrix's are diagonal, the entries of Q and R must be kept positive with possibility of some zeros on their diagonal.

$$\dot{x} = Ax + Bu \tag{4.5}$$

Lets minimize the cost function:

First lets introduce P: where p is p=p^T, then substitute it in equation4.4

J= $x^Tpx - x^Tpx + lim_{t\to\infty}x^TQx(t) + u^TRu(t)dt$

J= $x^Tpx + lim_{t\to\infty}[\frac{d}{dt}(x^Tpx) + x^TQx(t) + u^TRu(t)]dt$

$\frac{d}{dt}(x^Tpx) = \dot{x}^Tpx + x^Tp\dot{x}$

Then substitute $\dot{x}$ with equation4.5

$\frac{d}{dt}(x^Tpx) = (Ax + Bu)^Tpx + x^Tp(Ax + Bu)$

Substituting in the cost equation4.4

J= $x^Tpx + lim_{t\to\infty}[(Ax + Bu)^Tpx + x^Tp(Ax + Bu) + x^TQx(t) + u^TRu(t)]dt$

Group similar terms together:

J= $x^Tpx + lim_{t\to\infty}[x^T(A^Tp + pA + Q)x + u^TRu + x^TpBu + u^Tb^Tpx]dt$

Then lets take the element that is dependent on u:

$u^TRu + x^TpBu + u^TB^Tpx$=$(u + \frac{1}{R}B^Tpx)^TR(u + \frac{1}{R}B^Tpx) - x^T(pB\frac{1}{R}B^Tp)x$

Then substitute it in equation 4.4

J= $x^Tpx + lim_{t\to\infty}[x^T(A^Tp + pA + Q - pB\frac{1}{R}B^Tp)x + (u + \frac{1}{R}B^Tpx)^TR(u + \frac{1}{R}B^Tpx) - x^T(pB\frac{1}{R}B^Tp)x]$ dt

Setting the second part of the limit function to zero:

$(u + \frac{1}{R}B^Tpx)^TR(u + \frac{1}{R}B^Tpx) - x^T(pB\frac{1}{R}B^Tp)x$=0

Then the value of u became:

$u = -\frac{1}{R}B^T px$

$u = -kx$, which is full state feedback and where k is:

$k = \frac{1}{R}B^T p$

To find the value of p, we can examine the initial portion of the limit function.

$x^T(A^T p + pA + Q - pB\frac{1}{R}B^T p)x$, This equation is a Riccati equation. By setting the equation to zero, the matrix p can be determined. Once the value of p is determined, we can use it to find k and the control signal u.

To summarize the procedure taken:

- Selecting the weighting matrix Q and R
- Calculate the value of p by using the Riccati equation
- Determine the closed loop state feedback using $k = \frac{1}{R}B^T p$

The word regulator in LQR implies that the function of this feedback regulates the states to zero.

Chapter 5

Simulation Result

This chapter will provide the outcome of the two controller.

The below table indicates the parameter used in this paper.

Parameter(unit)	symbol	Value
Armature resistance(ohm)	R_a	2.Ω
Armature Inductance(H)	L_a	0.5H
Moment of inertia of motor(kg/m2)	J_m	$0.02kg.m^2$
Motor torque constant	K_t	$0.1N.m/A$
Electromotive force constant	K_e	$0.1V/(r.sec^-1)$

Table 5.1: Servo Motor Parameter

Parameter	symbol	Value
Maximum Iteration	Max_iter	10
Number of unknown population	npop	10
Inertia coefficient	w	0.7298
Damping ratio of inertia coefficient	wdamp	1
Personal acceleration coefficient	c1	1.4962
Social acceleration coefficient	c2	1.4962

Table 5.2: PSO Parameter

5.0.1 MPC Controller Result

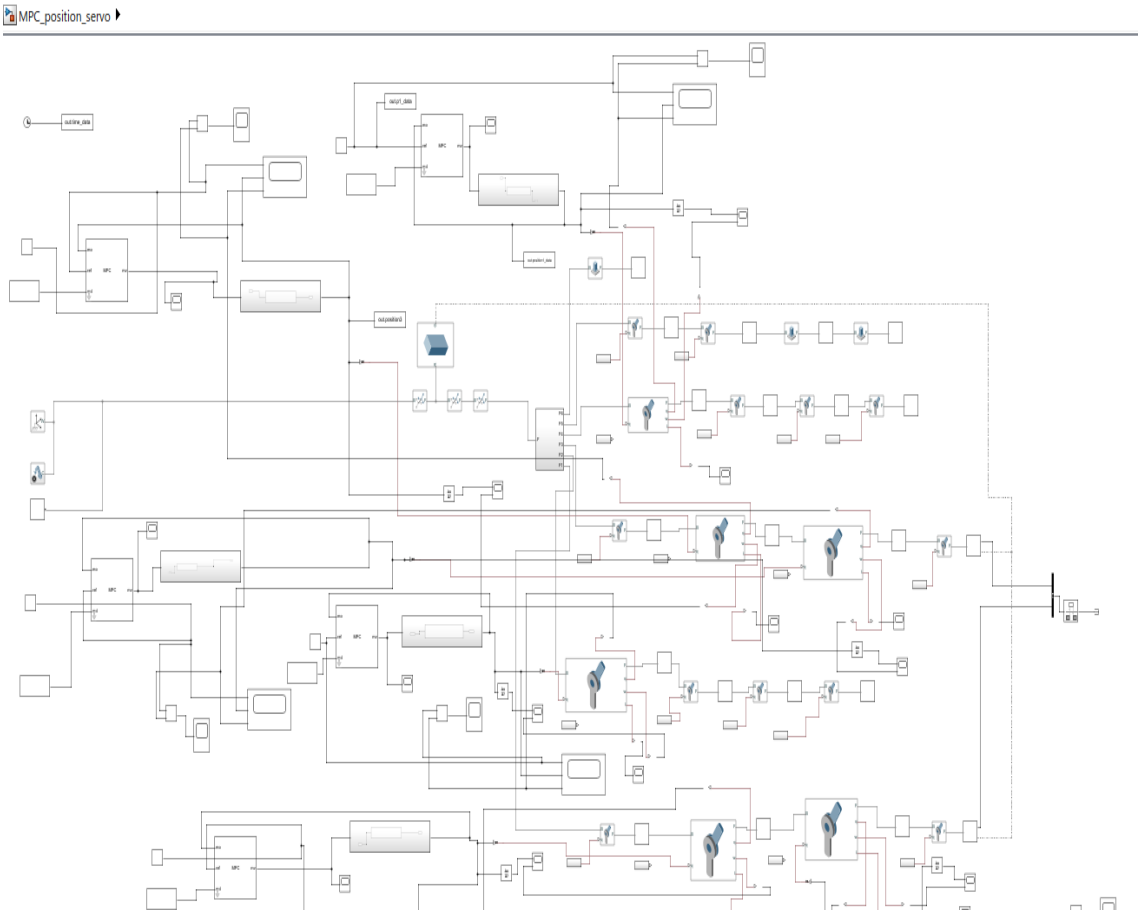

Figure 5.1: MPC with bipedal robot in Simulink

Parameter	Symbol	Value
Prediction Horizon	N	10
Control Horizon	M	2
Sampling Time	Ts	0.001

Q_11	Q_12	Q_13	Q_21	Q_22	Q_23	Q_31	Q_32	Q_33	R
9.93	0.05	0.04	0.075	9.989	0.077	0.057	0.014	9.979	9.99

The Left arm sagittal movement:

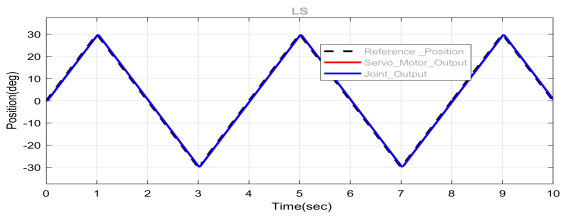

Figure 5.2: Left arm shoulder movement

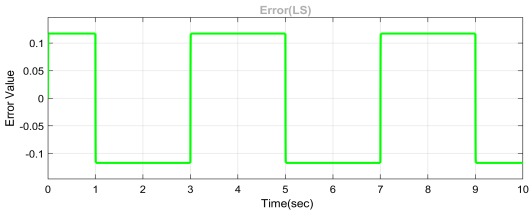

Figure 5.3: The difference between the reference signal and the output of the joint

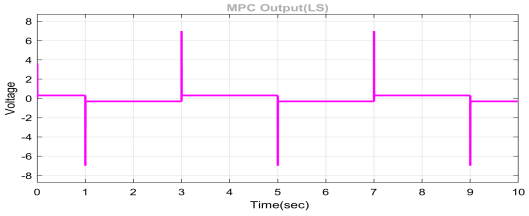

Figure 5.4: The control signal used in the left arm movement

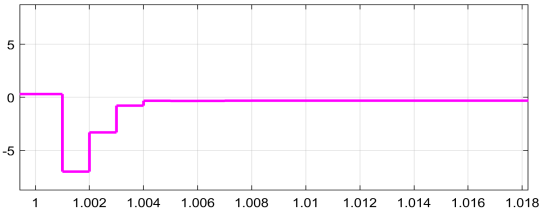

Figure 5.5: The control signal

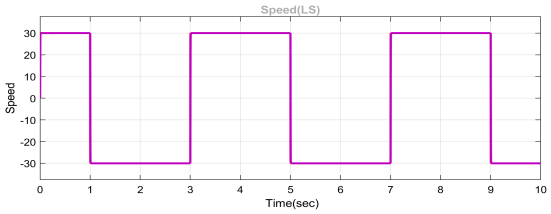

Figure 5.6: The speed value of the left arm movement

The Right arm sagittal movement:

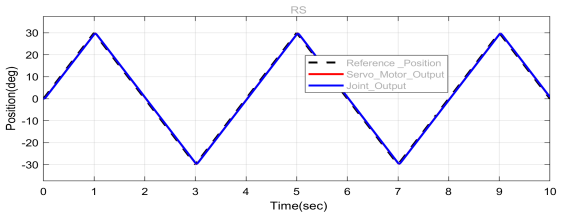

Figure 5.7: Right arm shoulder movement

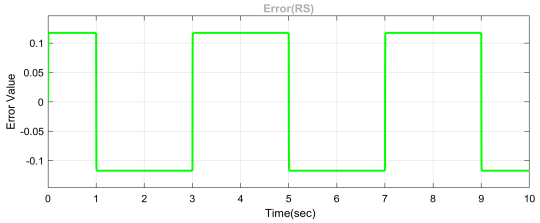

Figure 5.8: The difference between the reference signal and the output of the joint

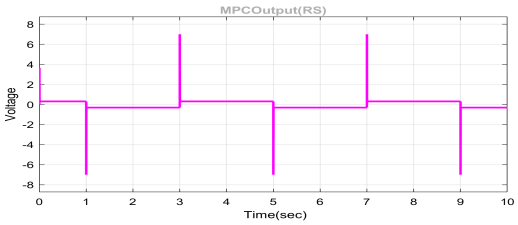

Figure 5.9: The control signal used in the right arm movement

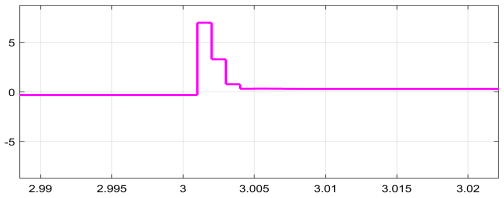

Figure 5.10: The control signal

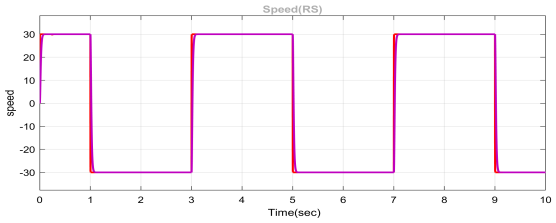

Figure 5.11: The speed value of the right arm movement

The MPC controller is utilized to follow the reference input, which is the motion signal causing the robot arm to move, given for each sagittal shoulder movement in both the right and left arm. In figure 5.2 and 5.7 , detects that the robot joint follow the reference input.

The controller proposed utilizes a predictive model to anticipate the future behavior of the system and determine the optimal control actions to minimize the difference between the desired input and the output motion of the robot arm. In figure 5.3 and 5.8,The error value is 0.13, that represents the peak value reached by the MPC and this error value is acceptable; it falls within the allowable range. This allows for precise and smooth movement of the arm, ensuring accurate and efficient performance in both right and left arm.

Figure 5.4 and 5.9 show the voltage values which is a controller output that affects

the result of servo motor. It Varies from 7v to -7v depending on the reference input and MPC parameter.

The graph 5.5 and 5.10 displays the discrete voltage values applied to the servo motor. This occurs due to the discrete nature of the MPC controller.

The speed of the motor has further been presented in Figure 5.6 and 5.11 showing the speed demanded by the motor the move each joint in the specified manner. The speed value remains the same in the next step as the error value.

The Left hip saggital movement:

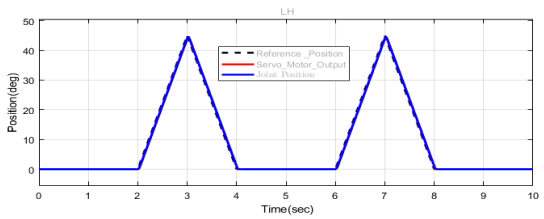

Figure 5.12: Left hip movement

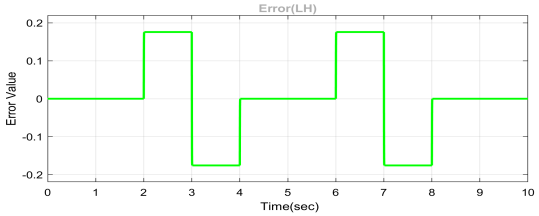

Figure 5.13: The difference between the reference signal and the output of the joint

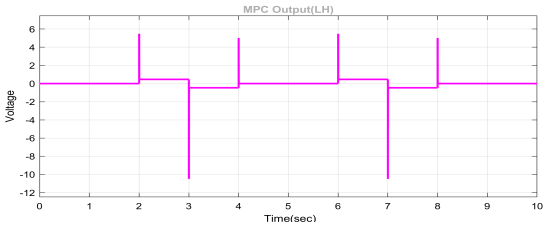

Figure 5.14: The control signal used in the left hip movement

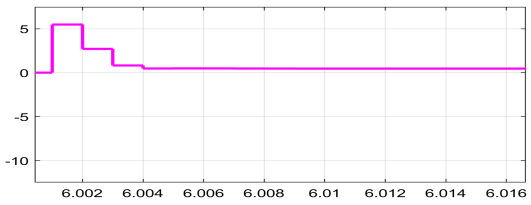

Figure 5.15: The control signal

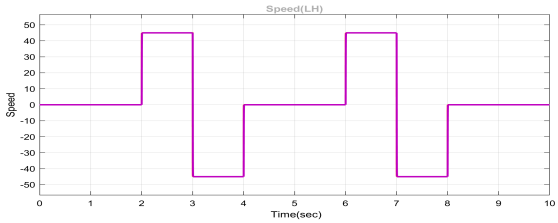

Figure 5.16: The speed value of the left hip movement

The Right hip saggital movement:

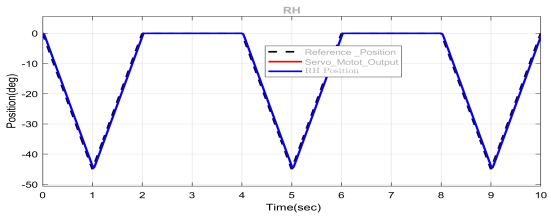

Figure 5.17: Right hip movement

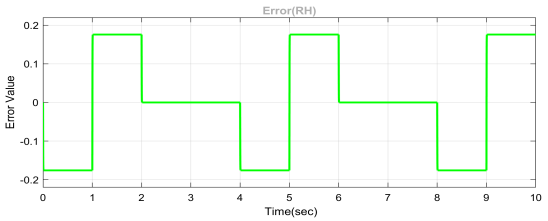

Figure 5.18: The difference between the reference signal and the output of the joint

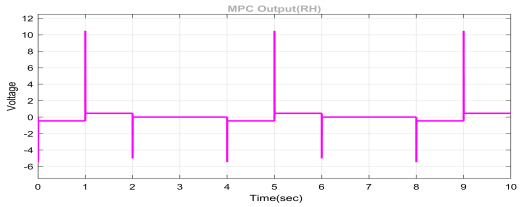

Figure 5.19: The control signal used in the right hip movement

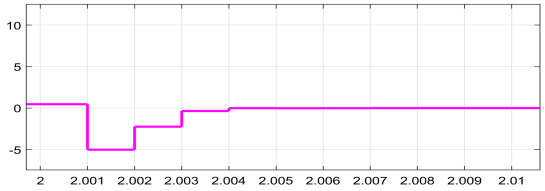

Figure 5.20: The control signal

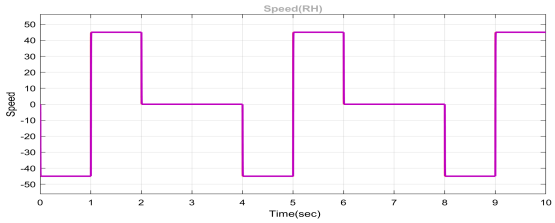

Figure 5.21: The speed value of the right hip movement

The controller is used to track the reference input, leading to the movement of the right hip joint and left hip joint.This movement is coordinated to ensure that the robot maintains its balance while walking. The controller receives feedback from sensors located throughout the robot's joint, allowing it to adjust the movement of the hip joints in real-time. In figure 5.12 and 5.17 , detects that the robot joint follow the reference input.

The controller calculate the optimal control actions to minimize the error between the reference input and the actual motion of the robot arm. In figure 5.13 and 5.18,The error value is 0.18, which represents the maximum value reached by the MPC. As the error value detects the performance of the robot, the controller plays a critical role in ensuring that the robot can walk smoothly and efficiently.

The graph5.14 and 5.19 show the voltage values which is the control signal provided by MPC. The voltage in left hip Varies from 5.8v to -10v and for right hip varies from

10v to -5.8v.

The graph 5.15 and 5.20 displays the discrete voltage values applied to the servo motor. This occurs due to the discrete nature of the MPC controller.

The speed of the motor has further been presented in figure 5.16 and 5.21 showing the speed demanded by the motor the move each joint in the specified manner.

The left knee saggital movement:

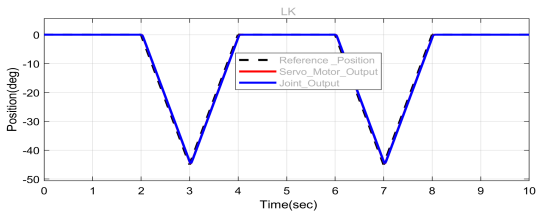

Figure 5.22: Left knee movement

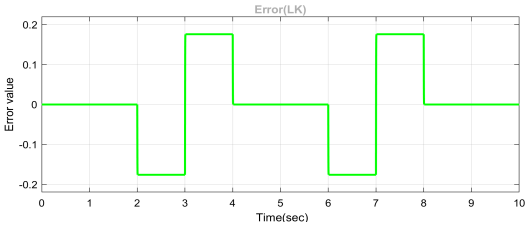

Figure 5.23: The difference between the reference signal and the output of the joint

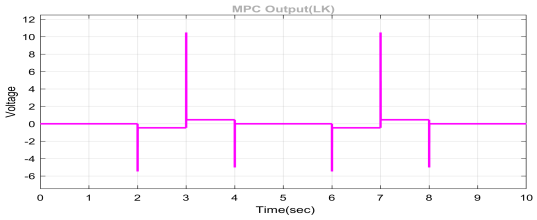

Figure 5.24: The control signal used in the left Knee movement

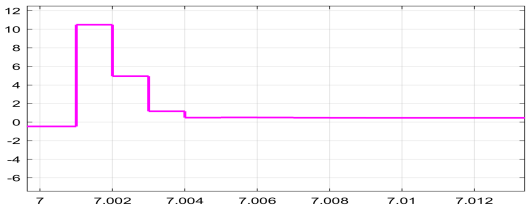

Figure 5.25: The control signal

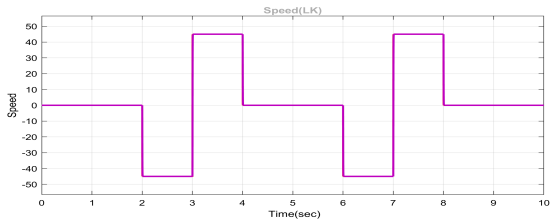

Figure 5.26: The speed value of the left knee movement

The Right knee saggital movement:

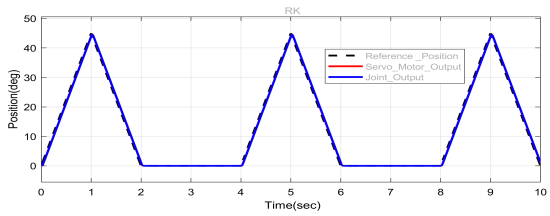

Figure 5.27: Right knee movement

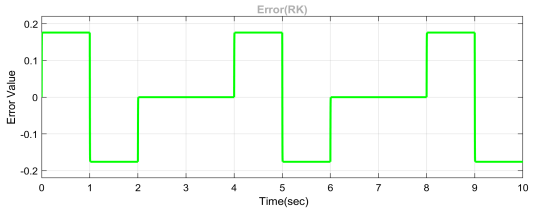

Figure 5.28: The difference between the reference signal and the output of the joint

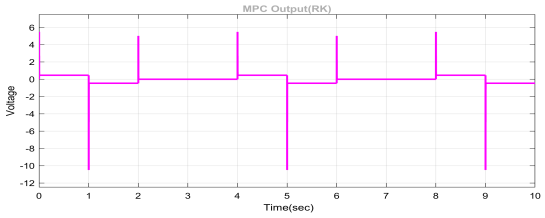

Figure 5.29: The control signal used in the right knee movement

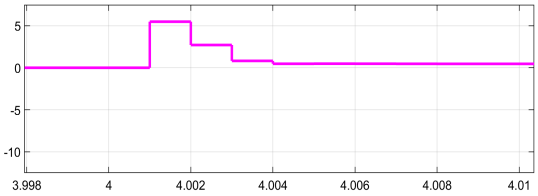

Figure 5.30: The control signa

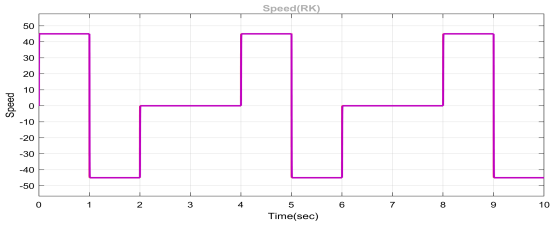

Figure 5.31: The speed value of the right knee movement

In figure 5.22 and 5.27 , detects that the robot joint follow the reference mtion signal.

The error between the reference input and the actual motion of the robot leg, 5.23 and 5.28 is 0.18, which represents the maximum value reached by the MPC.

The figure 5.24 and 5.29 show the voltage values which is the control signal provided by MPC. The voltage for the right knee Varies from 5.8v to -10v and the voltage for left knee varies from 10v to -5.8v.

The graph 5.25 and 5.30 displays the discrete voltage values applied to the servo motor. This occurs due to the discrete nature of the MPC controller.

The speed of the motor has further been presented in Fig 5.26 and 5.31 showing the speed demanded by the motor the move each joint in the specified manner.

5.0.2 LQR Controller Result

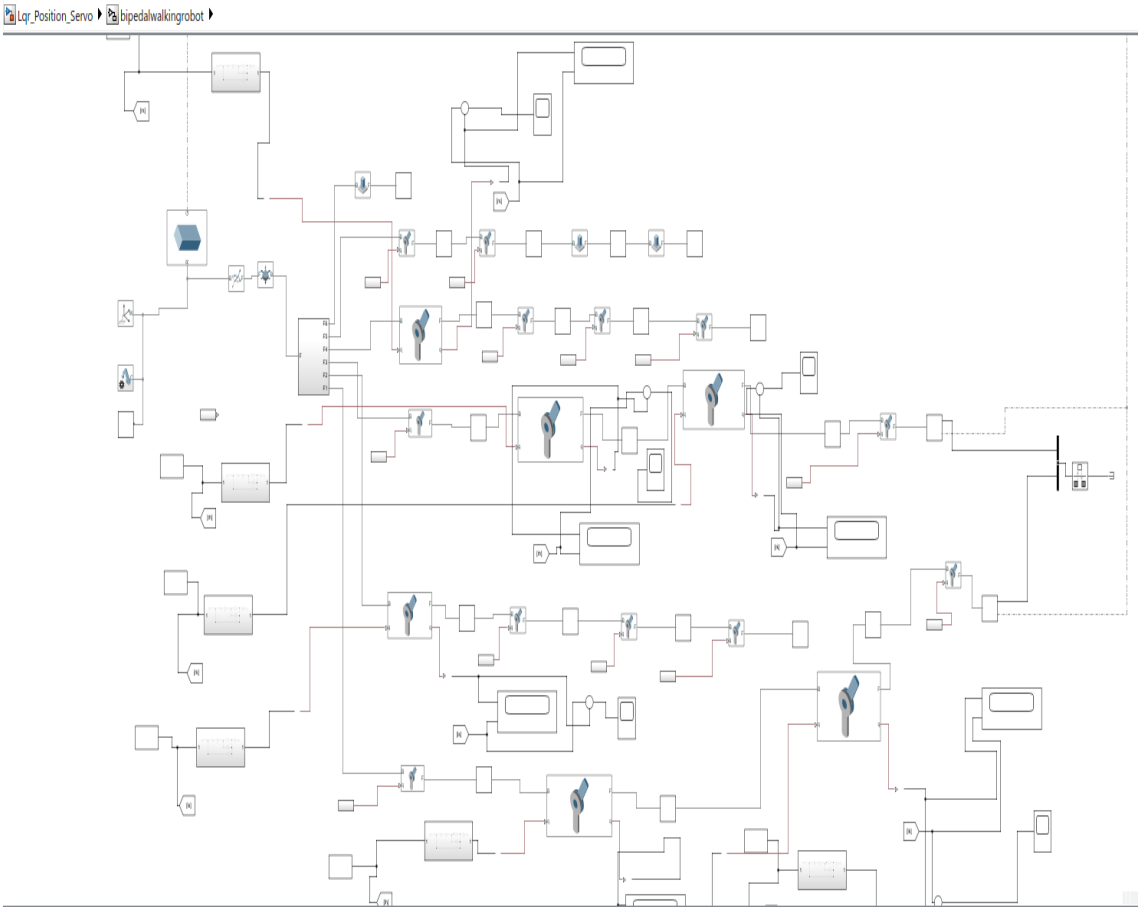

Figure 5.32: LQR with bipedal robot in simulink

Parameter(unit)	Symbol	Value
Constant	K	[1.0000 0.4615 0.3846]

The Left arm sagittal movement:

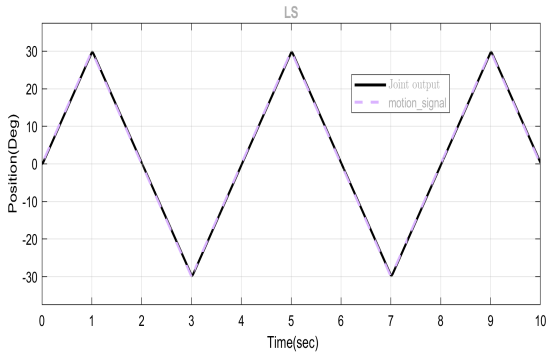

Figure 5.33: LQR left shoulder sagittal movement

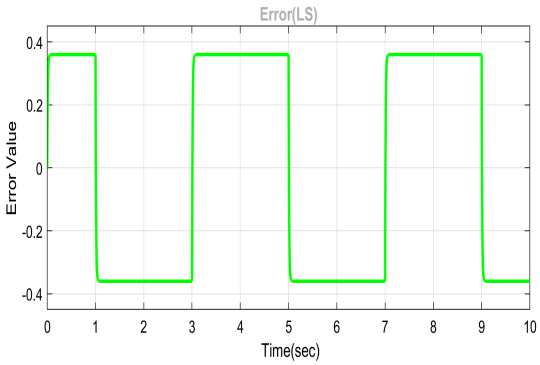

Figure 5.34: The error value in left arm

The Right arm saggital movement:

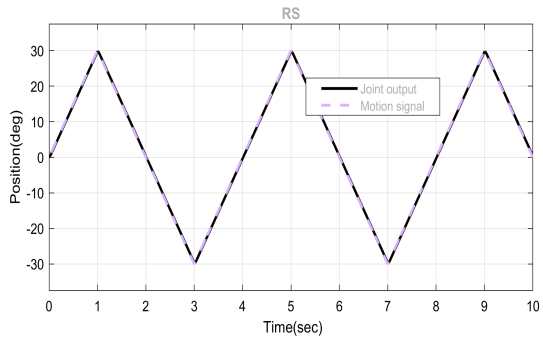

Figure 5.35: LQR right shoulder sagittal movement

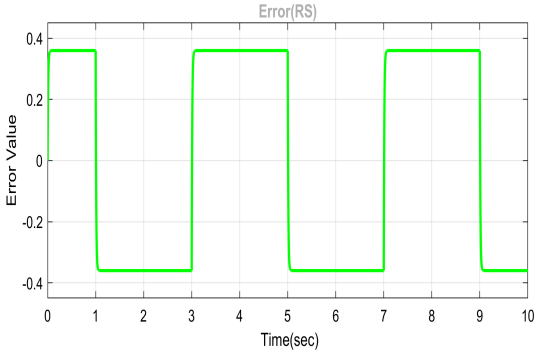

Figure 5.36: The error value in right arm

The LQR controller is used to ensure that the joints operate in compliance with the reference motion signal. The graph above5.33 and 5.35 shows that the position of both the left and right shoulders follows the reference motion signal.

The LQR controller decreases the difference between the actual joint position and the desired motion, resulting in smooth and accurate movement as seen in figure 5.34 and 5.36. This is crucial for tasks that require precise and coordinated motion. By continuously adjusting the control inputs based on feedback from the joint positions, the LQR controller ensures that the robot follows the desired trajectory closely, as demonstrated in the graph. The error between the desired motion signal and the output of joint position is 0.38(which is the peak value).

The Left hip saggital movement:

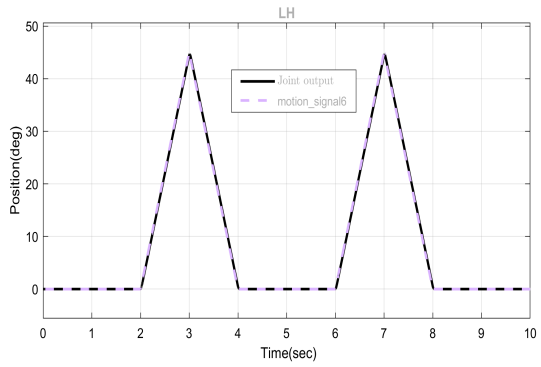

Figure 5.37: LQR left hip sagittal movement

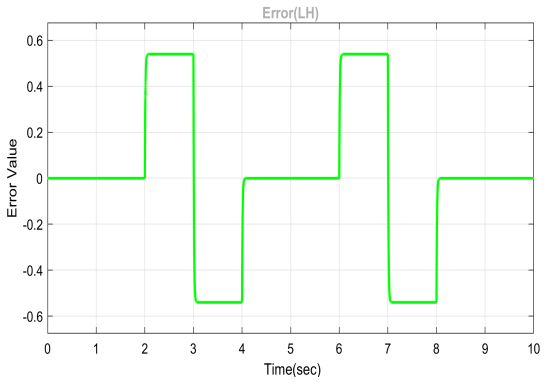

Figure 5.38: The error value in left hip

The Right hip saggital movement:

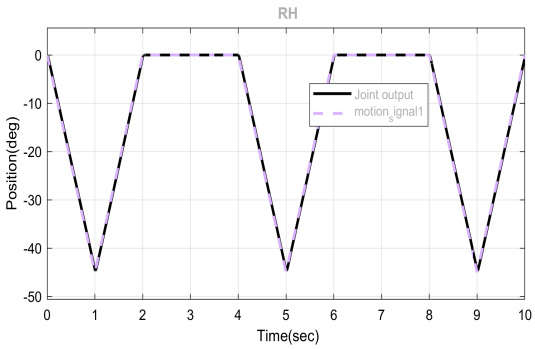

Figure 5.39: LQR right hip sagittal movement

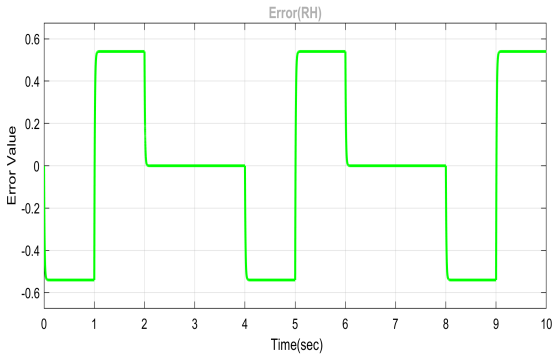

Figure 5.40: Error value in right hip

The controller is utilized to ensure that the joints to follow the reference motion signal. In the graph above, the reference signal is showed in broken line and the joint output in solid line. The graph above5.37 and 5.39 shows that the position of both the left and right shoulders follows the reference motion signal.

The controller minimizes the difference between the actual joint signal and the reference motion, in figure 5.38 and 5.40. The error between the desired motion signal and the output of joint position is 0.58.

The left knee saggital movement:

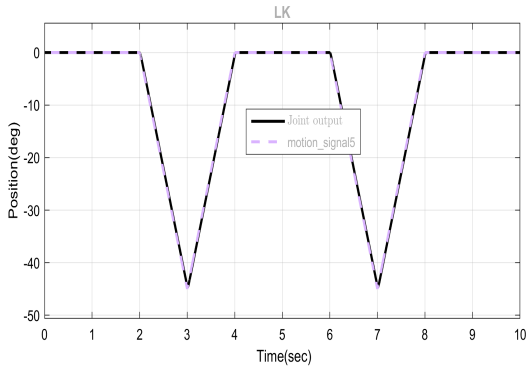

Figure 5.41: LQR left knee sagittal movement

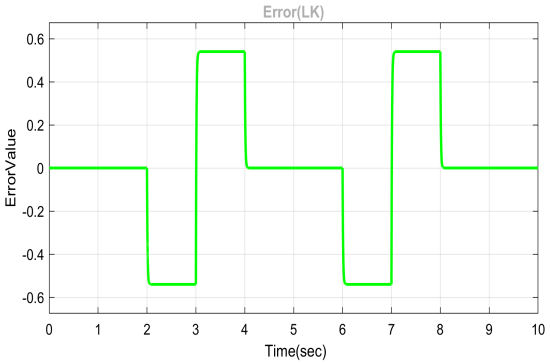

Figure 5.42: Error value in left knee

The Right knee saggital movement:

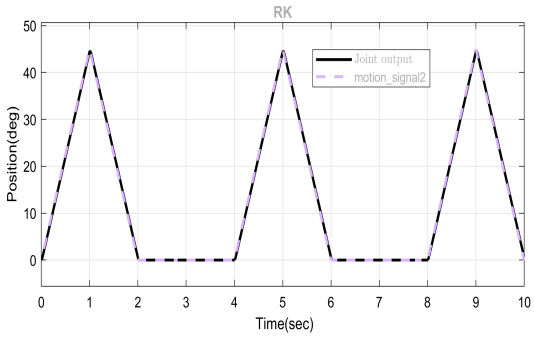

Figure 5.43: LQR right knee sagittal movement

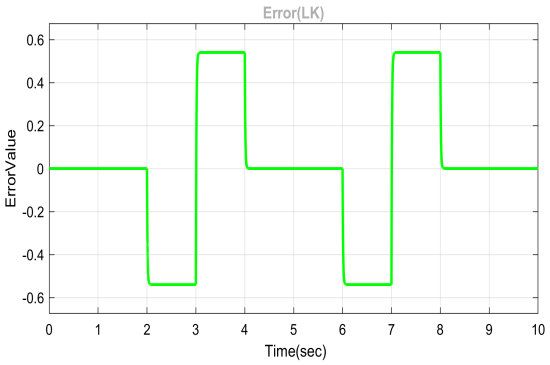

Figure 5.44: Error value in right knee

The controller ensures that the joints follow the reference motion signal. The graph above5.41 and 5.43 indicates that the position of both the left and right shoulders mirrors the reference motion signal..

The controller minimizes the difference between the actual joint signal and the reference motion in figure 5.42 and 5.44. The error between the desired motion value and the output of joint position is 0.58.

For better understanding, let's compare the performance of MPC and LQR in a single figure.

The left arm saggital movement:

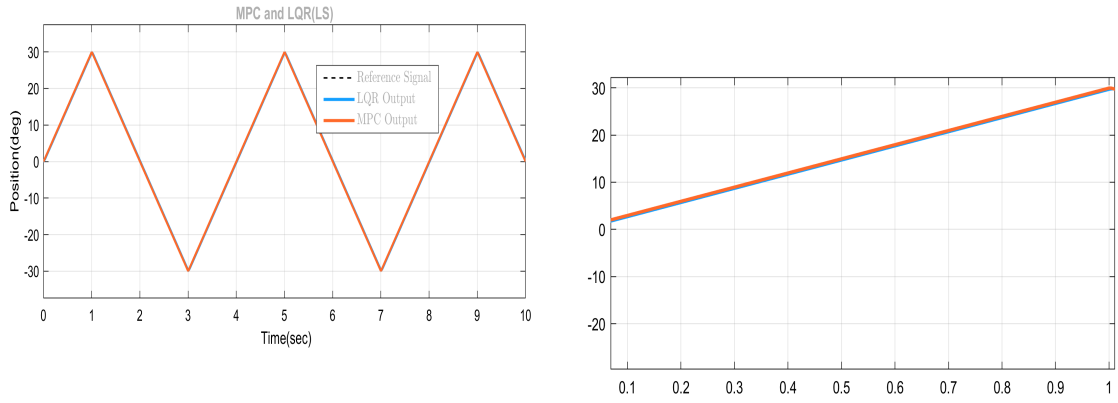

Figure 5.45: MPC and LQR left arm sagittal movement

The left hip saggital movement:

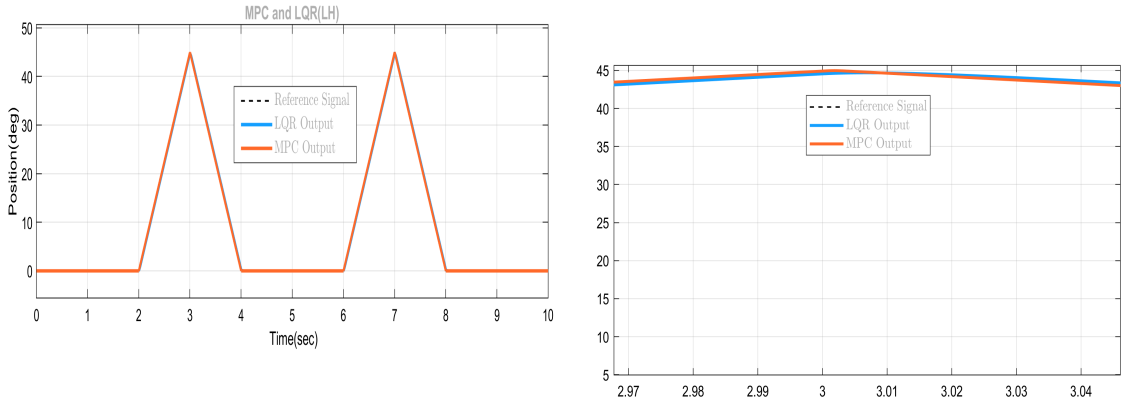

Figure 5.46: MPC and LQR left hip sagittal movement

The left Knee saggital movement:

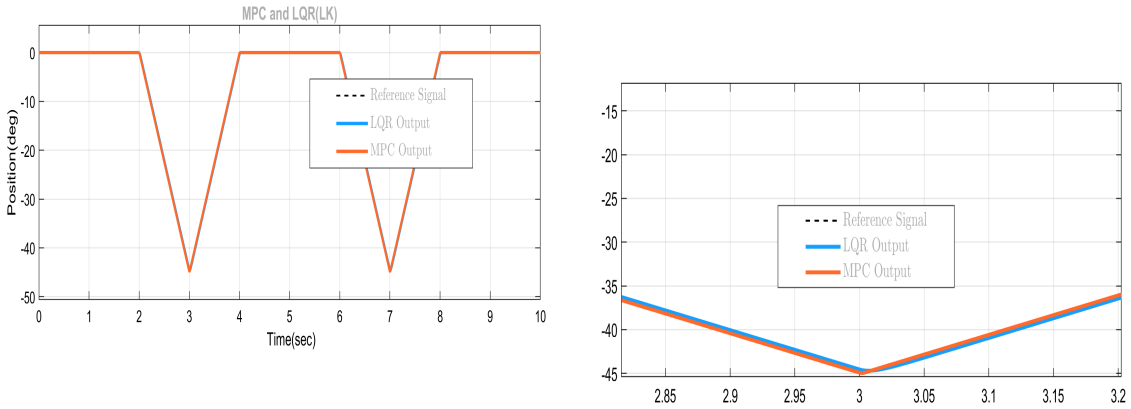

Figure 5.47: MPC and LQR left knee sagittal movement

The key variation between MPC and LQR is that MPC optimizes in receding time window where as LQR optimizes across the entire time window. In addition, MPC computes new solution frequently but LQR utilizes the same signal solution for entire time window. For this reason MPC has a better performance in tracking the reference motion signal, In the above graph 5.45, 5.46 and 5.47, indicates that MPC accurately tracks the reference signal.The MPC's ability to accurately track the reference signal is a testament to its advanced control algorithms and robustness. This ensures that the system responds quickly and accurately to changes in the reference signal, resulting in improved performance and stability. The ability to track the reference signal is particularly important in applications, where precise control is essential for achieving the desired outcomes. Overall, the MPC's ability to track the reference signal is a key feature that sets it apart from other control strategies and makes it a valuable tool for a wide range of applications.

5.1 Bipedal Robot with Load

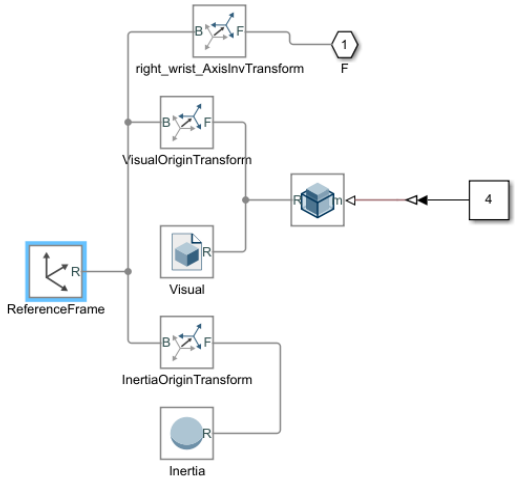

Figure 5.48: Connection between the brick and solid block of the arm

The bipedal robot is designed to carry external loads, which can cause disturbances. To address this, a variable solid block in MATLAB Simulink adjusts the mass value. Since the robot weighs 4.42 kg,It can hold up to 5 kg before falling. The disturbance signal, represented by the solid block, creates wanted inputs that affect the controller output and increase the system error. Therefore, the control system must be designed to eliminate the disturbance's effect on the output and system error.The control system design includes a feedback loop that measures the output and adjusts the input to minimize the error caused by the disturbance. The disturbance rejection capability of the control system is tested by applying different types of external loads and observing the robot's response. The simulation results show that the control system can effectively reject disturbances up to 5 kg, which is the maximum load the robot can carry. This makes the bipedal robot suitable for various applications.

5.1.1 MPC with Brick

The Left arm sagittal movement with brick:

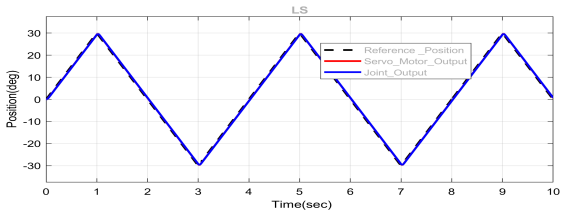

Figure 5.49: Left arm shoulder movement with brick

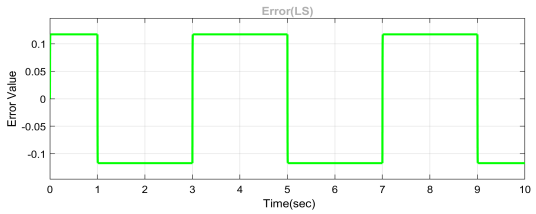

Figure 5.50: The difference between reference and actual

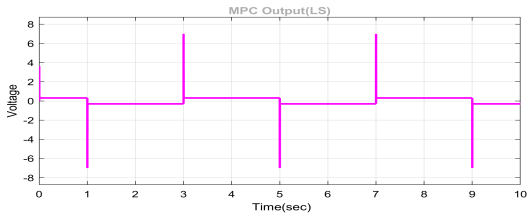

Figure 5.51: The control signal used in the left arm movement with brick

The Right arm sagittal movement with brick:

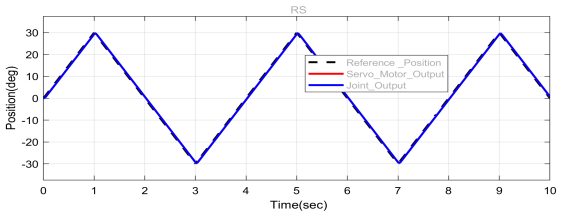

Figure 5.52: Right arm shoulder movement with brick

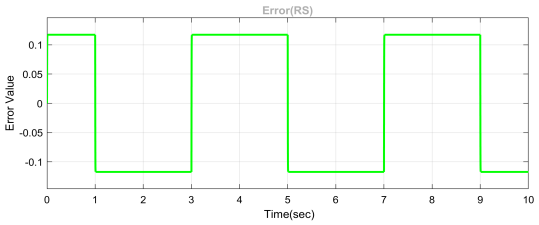

Figure 5.53: The difference between the reference and actual

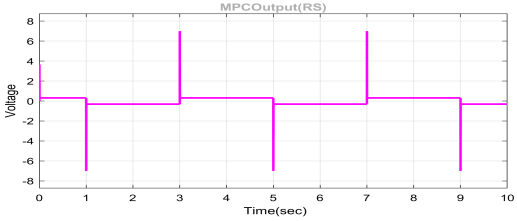

Figure 5.54: The control signal used in the right arm movement with brick

Figure 5.49 and 5.52 , detects that the robot joint follow the reference input. The MPC controller should eliminate the disturbance impact on the joint output and in the system error.The controller achieves this by continuously updating the control inputs based on the current state and predicted future behavior of the system. By considering the disturbance effects in the optimization problem, the MPC controller is able to proactively adjust the control actions to minimize the impact of disturbances on the system output and error. So a result in figure 5.50 and 5.53 the peak error value is 0.13(acceptable error value). This results robustness of the control system.

In order for the MPC to eliminate the effect of disturbances, the control signal does not necessarily have to be huge. The key is to design the MPC controller to account for disturbances and adjust the control signal accordingly. In figure 5.51 and 5.54, the control signal is some how similar with the first control signal with no brick.

The Left hip saggital movement with brick:

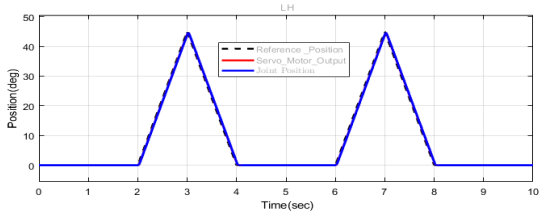

Figure 5.55: Left hip movement with brick

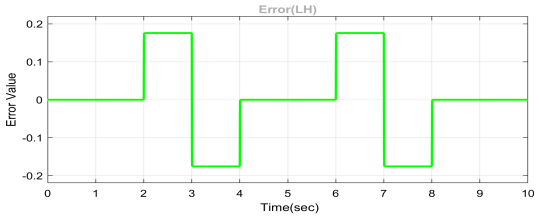

Figure 5.56: The difference between the reference and actual

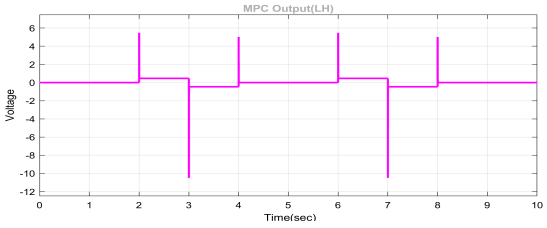

Figure 5.57: The control signal used in the left hip movement with brick

The Right hip saggital movement with brick:

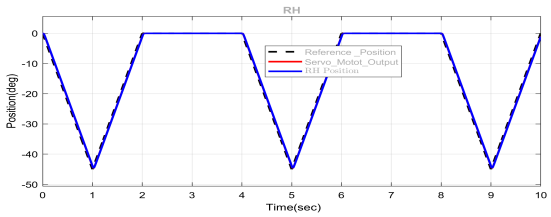

Figure 5.58: Right hip movement with brick

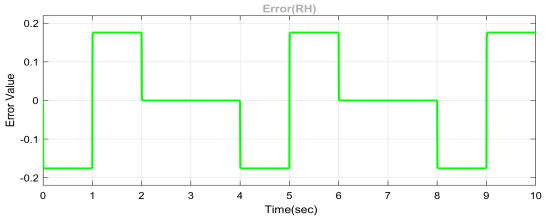

Figure 5.59: The difference between the reference and actual

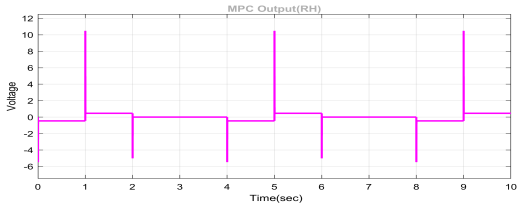

Figure 5.60: The control signal used in the right hip movement with brick

Figure 5.55 and 5.58 , Even with added disturbance, the right and left shoulder joints still follow the reference motion signal.This demonstrates the robustness and reliability of the hip joint control system, even in the presence of external influences.

As the controller eliminate the disturbance effect on the output and in the system error, resulting the the error value to become 0.18 seen in figure 5.56 and 5.59. The system error is effectively minimized, ensuring stable and accurate output.

In figure 5.57 and 5.60, the control signal used in the shoulder of the bipedal robot.

The left knee saggital movement with brick:

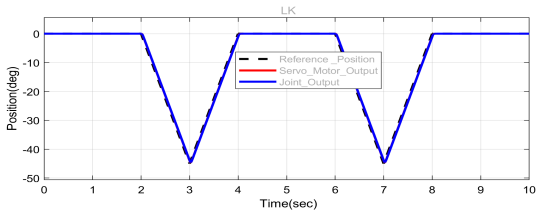

Figure 5.61: Left knee movement with brick

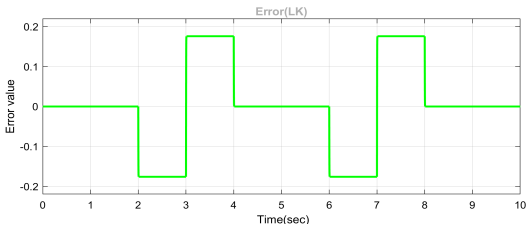

Figure 5.62: The difference between the reference and actual

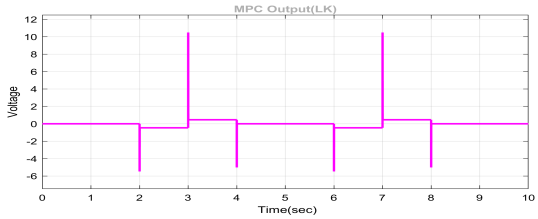

Figure 5.63: The control signal used in the left Knee movement with brick

The Right knee saggital movement with brick:

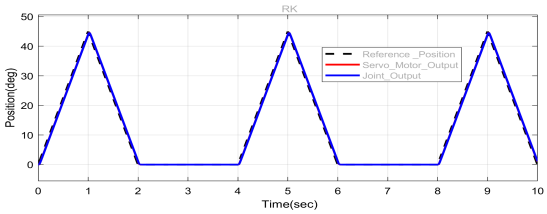

Figure 5.64: Right knee movement with brick

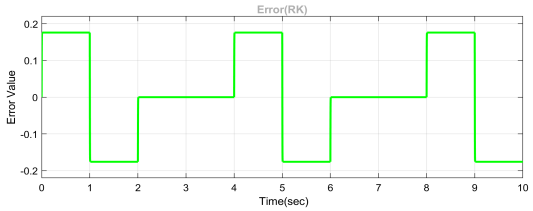

Figure 5.65: The difference between the reference and actual

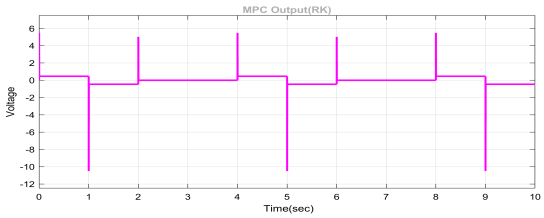

Figure 5.66: The control signal used in the right knee movement with brick

Figure 5.61 and 5.64 , shows the right and left knee movement follow the reference signal. The error value is 0.18 in figure 5.62 and 5.65.

Figure 5.63 and 5.66, shows the control signal value in right and left knee joint with brick.

5.1.2 LQR Controller Result with brick

The Left arm sagittal movement with brick:

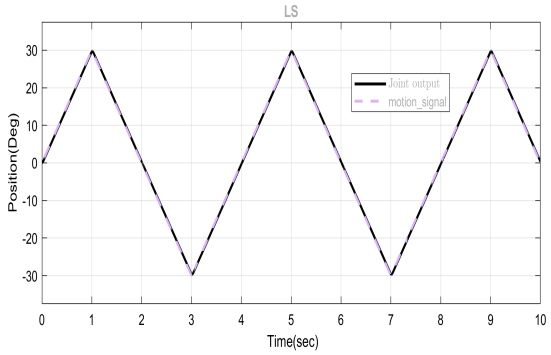

Figure 5.67: LQR left shoulder sagittal movement with brick

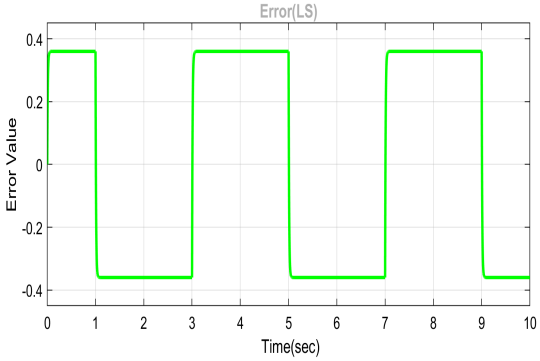

Figure 5.68: The error value with brick

The Right arm saggital movement:

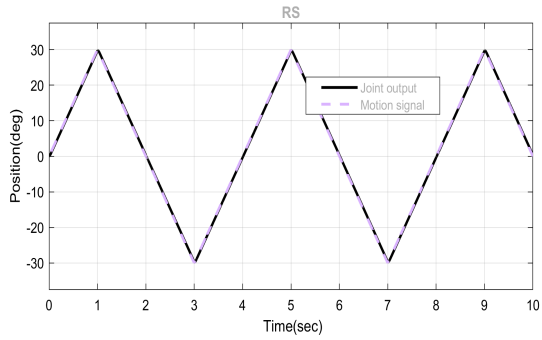

Figure 5.69: LQR right shoulder sagittal movement with brick

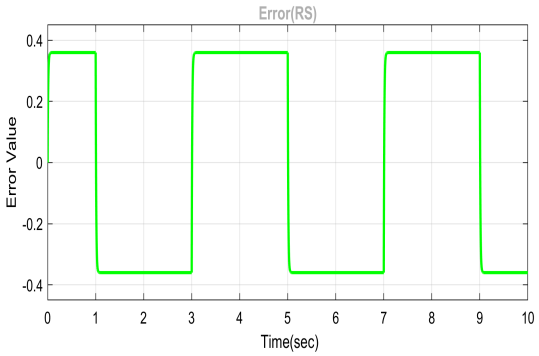

Figure 5.70: The error value with brick

In a system controlled with LQR, the addition of disturbance will affect the error value and path trajectory tracking. The LQR controller is designed to minimize the error between the desired trajectory and the actual trajectory of the system. The graph above5.67 and 5.69 shows that the position of both the left and right shoulders follows the reference motion signal. The peak error value is 0.38 in figure 5.68 and 5.70.

The Left hip saggital movement with brick:

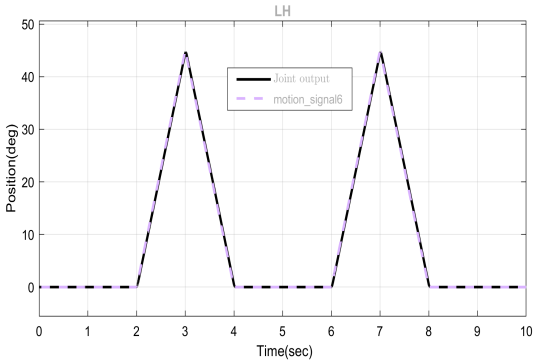

Figure 5.71: LQR left hip sagittal movement with brick

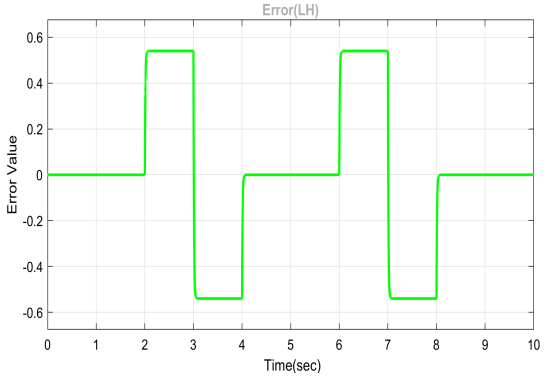

Figure 5.72: The error value in left hip with brick

The Right hip saggital movement:

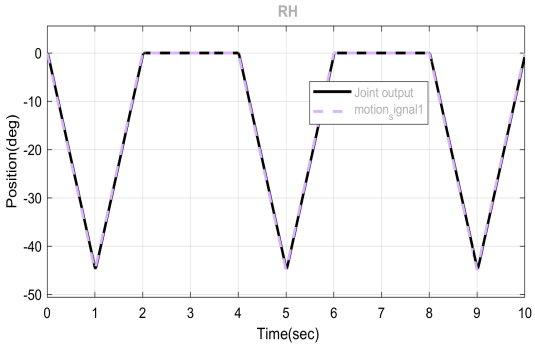

Figure 5.73: LQR right hip sagittal movement with brick

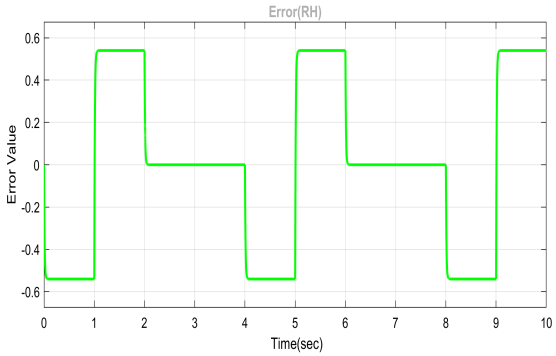

Figure 5.74: Error value in right hip with brick

The graph above5.71 and 5.73 shows that the position of both the left and right hip joint follows the reference motion signal and the error value in the graph 5.72 and 5.74 is 0.58.

The left knee saggital movement with brick:

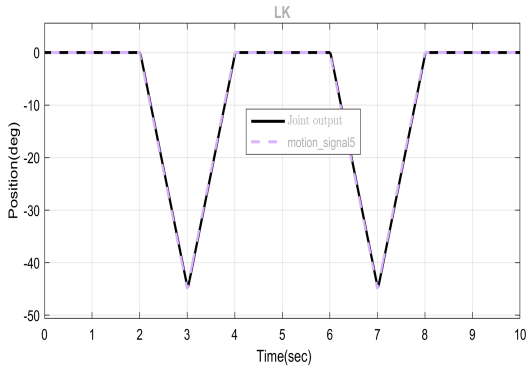

Figure 5.75: LQR left knee sagittal movement with brick

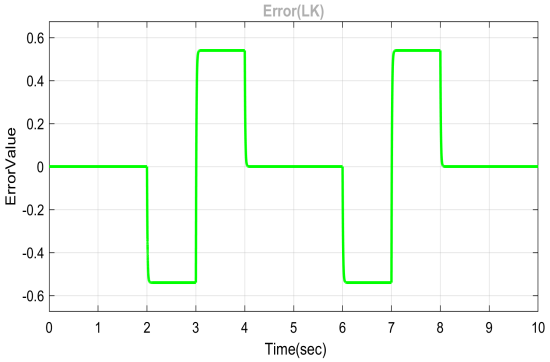

Figure 5.76: Error value in left knee with brick

The Right knee saggital movement:

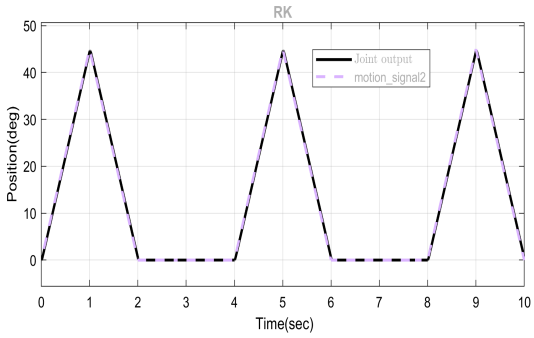

Figure 5.77: LQR right knee sagittal movement with brick

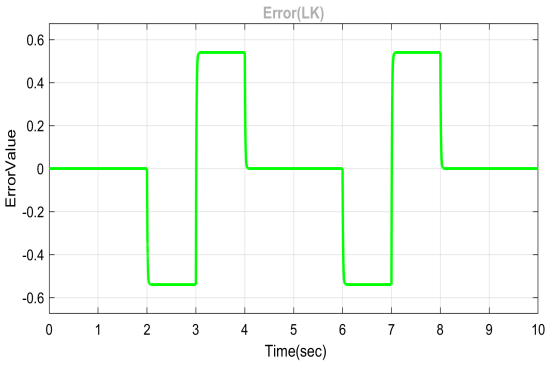

Figure 5.78: Error value in right knee

The figure in 5.75 and 5.77 detects that both the left and right knee joint follows the reference motion signal. The figure 5.76 and 5.78 shows the peak error value.

Linear Quadratic Regulator (LQR), the error between the reference signal and the output signal is not necessarily constant when both signals are rising or decreasing. The error depends on the specific dynamics of the system and the control law implemented.

However, in some cases, it is possible to design an LQR controller such that the error remains constant when both the reference and output signals are rising or decreasing. This can be achieved by carefully selecting the weighting matrices in the LQR cost function [37], [38] and [39].

Both MPC and LQR are designed properly so that the output of the control system is as desired.

5.1.3 Pure pursuit controller

The first step is to choose waypoints along the environment and create an algorithm to connect those waypoints, which will provide a path. To follow the generated path, a pure pursuit controller is utilized. This method allows for smooth and accurate navigation through the environment, ensuring that the bipedal robot reaches its destination while avoiding obstacles. Additionally, the pure pursuit controller can be fine-tuned to account for varying robot dynamics and environmental conditions, making it a versatile and effective navigation solution.

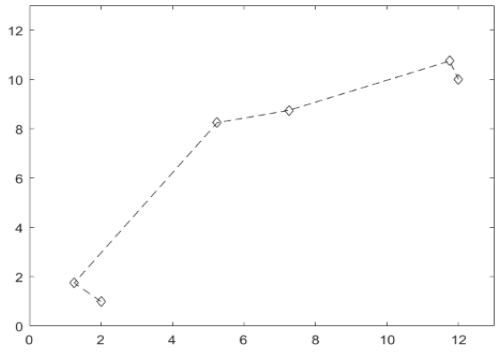

Figure 5.79: Way points given

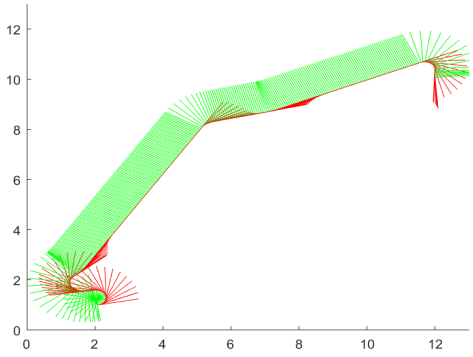

Figure 5.80: Result to way point given

Figure 5.79, is the laid out path where the bipedal robot has to carry, this laid out path are done by first specifying the points and by writing an algorithm to connect those specified points.

And figure 5.80 shows after the bipedal robot follows the provided path by using pure pursuit controller.

Chapter 6

Conclusion and Recommendation

This will be the last chapter, where the result will be discussed and future suggestion is given.

6.1 Conclusion

In this thesis a model predictive controller and linear quadratic controller is designed to control a bipedal robot that have the basic feature of human being. The bipedal robot is used to move around while carrying loads. First the presentation of bipedal robot is laid out and modeling was done using SOLIDWORK.

Once the model is exported to MATLAB and verified using Simulink, Motion signal is applied to different joints so that the bipedal robot will move like a human's, then ground is exported to MATLAB and aligned to the bipedal robot, where the robot will walk on. There should be connection between the ground and the foot of the robot and this is implemented by using spatial contact force block. The spatial contact force block plays the important role in how the two identities behave. The block applies equal and opposite force along the common contact plane and these forces conform to Newton's Third Law.The normal force is applied depending on penetration depth and penetration velocity, and the frictional force is applied depending on the normal force and relative velocities at the point of contact. The controller was designed.

After completing all the above tasks, two different controllers were designed. The main goal is to actuate the joints, but the joint itself cannot actuate, so a motor is needed. The selected motor is a servo motor, as it is recommended for position control.

This paper discusses the modeling and investigation of a DC servo motor, focusing on two control methodologies for tracking the position of the servo motor. The paper compares MPC and LQR control techniques.

The MPC controller used in this thesis is the toolbox found in MATLAB Simulink. The important thing in designing a controller is finding or assigning the appropriate values of the parameters, that is used by the controller. So for the value of control horizon, prediction horizon is found through open loop system of a servo motor and for the value of weights (Q and R) PSO is used. As MPC is compared with another controller called LQR. LQR controller is designed, by first formulating the state space equation of servo motor, then take feeds back the full state vector and multiplies it with the value of K and subtracts it from the reference. Firstly the system has to be controllable and observable. In LQR, there is no need to choose a pole location rather find the optimal value of K. The K value is chosen from the closed loop characters tics specifically, how well the system performs, and how much effort does it take to get that performance.

Analyzing the bellow table 6.1 comparison of control techniques suggests that MPC offers better control response compared to LQR. MPC exhibits faster rise times than LQR under normal conditions and in the presence of disturbances, while also demonstrating less undershoot and overshoot, ultimately leading to a lower error value. This makes MPC a more reliable and robust control technique, particularly in complex and dynamic systems. Overall, the comparison highlights the advantages of MPC over LQR in terms of control response and performance.

Controller technique		Overshoot	Undershoot	Rise time
LQR	left shoulder	0.505%	1.971%	16.585s
	Right shoulder	0.505%	1.970%	16.625s
	Left hip	0.505%	1.919%	792ms
	Right hip	0.505%	1.932%	792ms
	Left knee	0.505%	1.919%	792ms
	Right Knee	0.505%	1.932%	792ms
With disturbance	left shoulder	0.602%	1.996%	16.625s
	Right shoulder	0.602%	1.999%	16.84s
	Left hip	0.602%	1.993%	802ms
	Right hip	0.602%	1.9999%	802ms
	Left knee	0.602%	1.993%	802ms
	Right Knee	0.602%	1.999%	802ms
MPC	left shoulder	0.504%	1.755%	1.582s
	Right shoulder	0.504%	1.755%	1.582s
	Left hip	0.504%	1.852%	786.414ms
	Right hip	0.504%	1.852%	786.414ms
	Left knee	0.504%	1.931%	791.046ms
	Right Knee	0.504%	1.931%	791.046ms
With disturbance	left shoulder	0.515%	1.958%	1.612s
	Right shoulder	0.515%	1.958%	1.612s
	Left hip	0.505%	1.932%	791.064ms
	Right hip	0.505%	1.913%	791.064ms
	Left knee	0.514%	1.919%	792ms
	Right Knee	0.514%	1.932%	792ms

Table 6.1: Controller Result

For the path tracking, the bipedal robot is changed into dot with diameter value the same with the length between the right hand and left hand. Then the path will be laid out by using way point and tracked with pure pursuit controller.

6.2 Recommendation

In this implementation, a static environment has been considered. However, extending this work to a dynamic robot environment with obstacles moving in arbitrary directions and varying speeds could be a potential improvement. Additionally, to enhance accuracy, it is suggested to incorporate additional joints when controlling the bipedal robot. By introducing more joints, the robot can gain increased flexibility and a wider range of motion, resulting in more precise and efficient movements. The inclusion of extra joints can significantly enhance the robot's ability to navigate complex environments and perform intricate tasks. For example, by introducing additional joints in the robot's legs, it can adjust its stride length, step width, and foot orientation, thereby improving stability and adaptability on different terrains. Instead of using a motion signal extracted from a published paper, it is recommended to utilize the real motion signal of a human being and implement a single controller for controlling different joints, which would lead to greater accuracy. Lastly, to establish a connection between the bipedal robot and its environment, the implementation of AI would be an advancement. AI implementation enables the bipedal robot to effectively perceive and interact with its surroundings. By incorporating advanced sensors and cameras, the robot can gather real-time data about its environment, including information about the terrain, obstacles, and objects present in its vicinity.

Bibliography

[1] Mary Edwards. Robots in industry: An overview. *Applied ergonomics*, 15(1):45–53, 1984.

[2] Moritz Geilinger, Roi Poranne, Ruta Desai, Bernhard Thomaszewski, and Stelian Coros. Skaterbots: Optimization-based design and motion synthesis for robotic creatures with legs and wheels. *ACM Transactions on Graphics (TOG)*, 37(4):1–12, 2018.

[3] Teck Chew Wee. Design and control of a humanoid robot. 2014.

[4] Marko Bacic, Mark Cannon, Young Il Lee, and Basil Kouvaritakis. General interpolation in mpc and its advantages. *IEEE Transactions on Automatic Control*, 48(6):1092–1096, 2003.

[5] Roland Büchi. *State Space Control, LQR and Observer: step by step introduction, with Matlab examples*. Books on Demand, 2010.

[6] Heui Jae Pahk, Dong Sung Lee, and Jong Ho Park. Ultra precision positioning system for servo motor–piezo actuator using the dual servo loop and digital filter implementation. *International Journal of Machine Tools and Manufacture*, 41(1):51–63, 2001.

[7] Yong Zhang, Kangting Liu, Feng Gao, and Fengkui Zhao. Research on path planning and path tracking control of autonomous vehicles based on improved apf and smc. *Sensors*, 23(18), 2023.

[8] Andrea Maiorino and Giovanni Gerardo Muscolo. Biped robots with compliant joints for walking and running performance growing. *Frontiers in Mechanical Engineering*, 6:11, 2020.

[9] Ke Wang, Hengyi Fei, and Petar Kormushev. Fast online optimization for terrain-blind bipedal robot walking with a decoupled actuated slip model. *Frontiers in Robotics and AI*, 9:812258, 2022.

[10] Nguyen Xuan Hong. Controlling two-legged mobile robot. 2021.

[11] Pravin Dangol, Alireza Ramezani, and Nader Jalili. Performance satisfaction in harpy, a thruster-assisted bipedal robot. *arXiv preprint arXiv:2004.14337*, 2020.

[12] Taylor Apgar, Patrick Clary, Kevin Green, Alan Fern, and Jonathan W Hurst. Fast online trajectory optimization for the bipedal robot cassie. In *Robotics: Science and Systems*, volume 101, page 14. Pittsburgh, Pennsylvania, USA, 2018.

[13] Jessy W Grizzle, Jonathan Hurst, Benjamin Morris, Hae-Won Park, and Koushil Sreenath. Mabel, a new robotic bipedal walker and runner. In *2009 American Control Conference*, pages 2030–2036. IEEE, 2009.

[14] Alireza Ramezani, Jonathan W Hurst, Kaveh Akbari Hamed, and Jessy W Grizzle. Performance analysis and feedback control of atrias, a three-dimensional bipedal robot. *Journal of Dynamic Systems, Measurement, and Control*, 136(2):021012, 2014.

[15] Guillermo A Castillo, Bowen Weng, Wei Zhang, and Ayonga Hereid. Robust feedback motion policy design using reinforcement learning on a 3d digit bipedal robot. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5136–5143. IEEE, 2021.

[16] Igor Mordatch, Martin De Lasa, and Aaron Hertzmann. Robust physics-based locomotion using low-dimensional planning. In *ACM SIGGRAPH 2010 papers*, pages 1–8. 2010.

[17] Johannes Englsberger, Christian Ott, and Alin Albu-Schäffer. Three-dimensional bipedal walking control based on divergent component of motion. *Ieee transactions on robotics*, 31(2):355–368, 2015.

[18] Yiping Liu, Patrick M Wensing, David E Orin, and Yuan F Zheng. Trajectory generation for dynamic walking in a humanoid over uneven terrain using a 3d-actuated dual-slip model. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 374–380. IEEE, 2015.

[19] Jamshed Iqbal. Modern control laws for an articulated robotic arm. *Engineering, Technology & Applied Science Research*, 9(2):4057–4061, 2019.

[20] Mohamed W Mehrez Said. Optimization based solutions for control and state estimation in non-holonomic mobile robots: Stability, distributed control, and relative localization. *arXiv preprint arXiv:1803.06928*, 2018.

[21] Muhammad Imran Ullah, Syed Ali Ajwad, Muhammad Irfan, and Jamshed Iqbal. Mpc and h-infinity based feedback control of non-linear robotic manipulator. In *2016 International Conference on Frontiers of Information Technology (FIT)*, pages 136–141. IEEE, 2016.

[22] Xu Zhou, Xiaoli Zhang, Jiucai Zhang, and Rui Liu. Stabilization and trajectory control of a quadrotor with uncertain suspended load. *arXiv preprint arXiv:1612.04324*, 2016.

[23] Sudhir Nadda and A. Swarup. Integral sliding mode control for position control of robotic manipulator. In *2018 5th International Conference on Signal Processing and Integrated Networks (SPIN)*, pages 639–642, 2018.

[24] Yuxin Su and Chunhong Zheng. A new nonsingular integral terminal sliding mode control for robot manipulators. *International Journal of Systems Science*, 51(8):1418–1428, 2020.

[25] Janki Chotai and Krupa Narwekar. Modelling and position control of brushed dc motor. In *2017 International Conference on Advances in Computing, Communication and Control (ICAC3)*, pages 1–5. IEEE, 2017.

[26] Gwo-Ruey Yu and Rey-Chue Hwang. Optimal pid speed control of brush less dc motors using lqr approach. In *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583)*, volume 1, pages 473–478. IEEE, 2004.

[27] Xianmin Chen, Huixing Zhou, Ronghua Ma, Fuchang Zuo, Guofang Zhai, and Minli Gong. Linear motor driven inverted pendulum and lqr controller design. In *2007 IEEE International Conference on Automation and Logistics*, pages 1750–1754. IEEE, 2007.

[28] RMT Raja Ismail, MA Ahmad, and MS Ramli. Speed control of buck-converter driven dc motor using lqr and pi: A comparative assessment. In *2009 International Conference on Information Management and Engineering*, pages 651–655. IEEE, 2009.

[29] Soo-Hyun Ryu, Yeonsik Kang, Sin-Jung Kim, Keonyong Lee, Bum-Jae You, and Nakju Lett Doh. Humanoid path planning from hri perspective: A scalable approach via waypoints with a time index. *IEEE Transactions on Cybernetics*, 43(1):217–229, 2013.

[30] M Sanchez-Magos, M Ballesteros, David Cruz-Ortiz, Iván Salgado, and I Chairez.

Terminal sliding-mode control of virtual humanoid robot with joint restrictions walking on stepping objects. *Cybernetics and Systems*, 51(4):402–425, 2020.

[31] D. Cruz-Ortiz I. Salgado M. Sanchez-Magos, M. Ballesteros and I. Chairez. Terminal sliding-mode control of virtual humanoid robot with joint restrictions walking on stepping objects. *Cybernetics and Systems*, 51(4):402–425, 2020.

[32] Dayal R Parhi, BBVL Deepak, Devedutta Nayak, Anand Amrit, et al. Forward and inverse kinematic models for an articulated robotic manipulator. *International Journal of Artificial Intelligence and Computational Research*, 4(2):103–109, 2012.

[33] Christoph Klug, Dieter Schmalstieg, Thomas Gloor, and Clemens Arth. A complete workflow for automatic forward kinematics model extraction of robotic total stations using the denavit-hartenberg convention. *Journal of Intelligent & Robotic Systems*, 95:311–329, 2019.

[34] Zhengrong Xiang and Wenbin Wei. Design of dc motor position tracking system based on lqr. In *Journal of Physics: Conference Series*, volume 1887, page 012052. IOP Publishing, 2021.

[35] Arnisa Myrtellari, Petrika Marango, and Margarita Gjonaj. Analysis and performance of linear quadratic regulator and pso algorithm in optimal control of dc motor. *International Journal of Latest Research in Engineering and Technology*, 2(4), 2016.

[36] Federico Marini and Beata Walczak. Particle swarm optimization (pso). a tutorial. *Chemometrics and Intelligent Laboratory Systems*, 149:153–165, 2015.

[37] Robert F Stengel. *Optimal control and estimation*. Courier Corporation, 1994.

[38] Brian DO Anderson and John B Moore. Detectability and stabilizability of time-varying discrete-time linear systems. *SIAM Journal on Control and Optimization*, 19(1):20–32, 1981.

[39] Mark J Kaiser. Control systems engineering: by norman s. nise; ; benjamin/cummings; redwood city, ca, usa; 1995; xxiv+ 854 pp.; $49; isbn : 0 - 8053 - 5424 - 7, 1995$.