Faculty of Mechanical Engineering

GIK Institute of Engineering Sciences & Technology

June 2020

Design, Fabrication and Control of a Bipedal Robot

Senior design project report

BY

Muhammad Idrees Khan	2016322
Muhammad Mustansar	2016337
Amina Waheed	2016072
Muhammad Owais	2016344

Supervised by

Dr. Abid Imran

submitted in partial fulfillment of the requirements for the degree of Bachelor of Science

Faculty of Mechanical Engineering

GIK INSTITUTE OF ENGINEERING SCIENCES & TECHNOLOGY

Senior Design Project Status/Completion Certificate

This is to certify that senior year design project has satisfied the following:

(i) The design part of the project is completed to a sufficient level. Specifically, the project has achieved.

a.	
b.	
c.	

(ii) The students have engaged in weekly meetings and demonstrated gradual progression of work, meeting the 6CH requirement.

(iii) The defined Scope / Objectives are;

Sr.#	**Objectives**	**KPI (min. 50%) achieved**	
		Advisor	**Ext. Examiner**
a.			
b.			
c.			
d.			
e.	Quality of Report (Technical content, breadth/depth)		

(iv) Based on the above score, the project stands______________. (complete/incomplete)

(v) I (Advisor) understand that the report may be subjected to external review.

________________ ____________________

Advisor **External Examiner**

(**To be filled by FYP Coordinator**)		**Good**	**Average**	**Poor**
(vi)	Overall structure of report is in-line with the provided FME guidelines.			
(vii)	Students followed the given deadlines by SDP committee			

____

FYP Coordinator

____

Dean FME

ABSTRACT

This report presents the design and fabrication of a bipedal robot which can walk straight without falling. Two main issues are focused. Firstly, a simple mechanical design named 'Bart-UH' is selected among few different design alternatives including: The Serial Parallel Hybrid Leg Mechanism, the Slider, and the Cassie. Secondly, an iterative solution is devised for the solution of the trajectory of the torso under the influence of swing leg motion using the ZMP Approach. The choice of the Bart-UH is based on its simplicity and robustness while the ZMP approach is selected as it is one of the common approaches used in robotics. ZMP is a point about which the sum of moments acting on the foot due to the ground reaction forces only have a vertical component while all other components are zero. By using the ZMP approach, the problem is simplified as the ground forces need not to be calculated.

After the ZMP approach is introduced, the next important problem is to solve the trajectory problem for the robot. For this purpose, the swing leg motion is adjusted arbitrarily, and the torso motion is calculated analytically to ensure stability. Three approaches are used to solve the trajectory problem. The Single IPM (Inverted Pendulum Model), TMIPM (Two mass Inverted Pendulum Model) and MMIPM (Multiple mass Inverted Pendulum Model). The TMIPM method is used. TMIPM method yields the joint trajectories which are used to proceed to the MMIPM method further. Without considering the TMIPM model, it is not possible to continue directly with the MMIPM method. Thus, the TMIPM poses as a pre-requisite for the MMIPM Model.

In the following section, the CAD Model of the robot is elaborated. All the components are shown in detail. Finally, the mathematical modelling is discussed in detail to show how the ZMP equation is solved and the joint trajectories are evaluated for the TMIPM and MMIPM methods. For the solution of the differential equations of the ZMP and generation of the trajectories, computational tools like MATLAB is used.

The next section takes into consideration the Controls, Instrumentation, and testing of the manipulator. The controls are explained in the step-by-step procedure that was followed; the selection of micro-controllers and the selection criterion, the required sensors, the selection of motor while considering the movement of our manipulator, the torque requirements, the shift from a closed-loop control to an open-loop control (using digital motors with built-in encoders), the development of an Android Mobile Application for user input, the Simulink connection with Arduino, the problems faced and the solutions to those problems.

Keywords: Zero Moment Point (ZMP), gait stability, inverted pendulum model (IPM), two mass inverted pendulum model (TMIPM), multiple inverted pendulum model (MMIPM), torso motion, bipedal robots, Bart-UH,

Nomenclature

$F_v(t)$:	Vertical force (N)
$F_H(t)$:	Horizontal Force (N)
IPM:	Inverted Pendulum Mode
m_{123}:	Concentrated mass residing at torso level
m_4, m_5, m_6:	Mass of the swinging leg
MMIPM:	Multiple Masses Inverted Pendulum Mode
SH:	Step Height
SW:	Step Width of torso
TMIPM:	Two Masses Inverted Pendulum Mode
y_H:	Torso Height
ZMP:	Zero Moment Point
s_1:	$\sin\theta_1$
c_1:	$\cos\theta_1$

List of Figures

Table of Contents

Chapter 1

Introduction

1.1 Background and Motivation

Humanoid Robots have always been of immense importance attracting researchers from all over the world. Attempts to build humanoid robots date back to the 1920s. There is a long list of inventions in the field of humanoid robots. With an increasing development in the field of robotics, humanoid walking robots have greatly captured researchers' and scientists' attention. Starting from **1937,** when Westinghouse created ELEKTRO; a human-like robot that could walk, talk and smoke and **1969,** when Ichiro Kato designed the first biped robot up till today when Oregon State University designed a new bipedal robot named Cassie under Agility Robotics.

Bipedal robots are used in a wide range of applications of which the most important are the search and rescue operations. Their ability to move in areas inaccessible to wheeled robots, makes them more beneficial to use. It is easier to interact with a human like robots rather than a non-human shape (Brooks, 1996). For example, the Chernobyl Nuclear Disaster that took place in 1986 had serious and adverse after-effects. The accident caused the largest uncontrolled radioactive release into the environment. About 200,000 people ('liquidators') from all over the Soviet Union were involved in the recovery and clean-up during 1986 and 1987. They received high doses of radiation. The highest doses were received by 1000 emergency workers and on-site personnel approximately, during the first day of the accident. Thus, bipedal robots that mimic human gait will be helpful in such disasters where human's health and safety is at stake. This would not only save humans from hazards and risks but would also speed up the process. Therefore, this project focuses on the, "Design, Fabrication and Control of a Bipedal Robot".

1.2 Problem Statement:

Design, Fabrication and Control of a Bipedal Robot with 10 DOF that can stabilize itself and then maintain its stable position. There will be 5 DOF in each leg; 3 in hip, 1 in knee and 1 in ankle. All the joints are expected to be revolute in nature.

1.3 Scope and Expected Outcomes

- This project consists of the derivation of a mathematical model for the stability problem and solution of the dynamics of a two-legged robot that will be able to stabilize itself during walking**.** This project is a minor extension to the work of [1] with the robot having 10 DOF. If the robot is successfully stabilized during the walking phase, machine learning will be applied to introduce obstacle avoiding into the algorithm. This, however, is considered as a future endeavour and is not included in the basic scope of the project!
- CAD design and fabrication of the robot.
- Installation of motors, electrical wiring and Control using a micro controller (Arduino or Raspberry Pi).

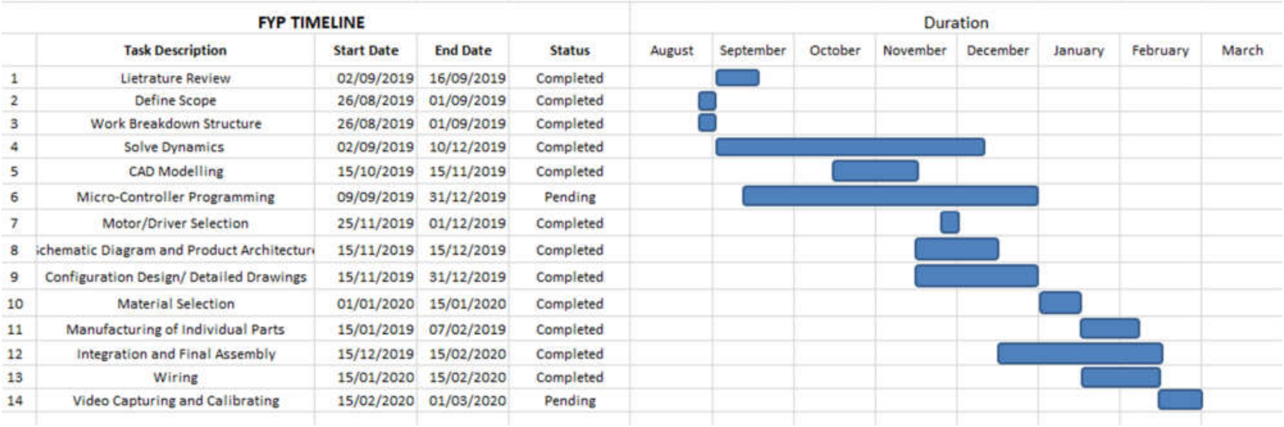
1.4 Report Outline

This report comprises of 7 chapters.

- **Chapter 1**, Introduction: This chapter provides a basic overview of the problem statement, the background, and the scope of the project. The schedule and the contribution of each member is also covered in this chapter.
- **Chapter 2**, Literature review: The major literature related to the project, the basic definitions, the different approaches used and the selected approach for the project is discussed in this chapter.
- **Chapter 3**, Design and Analysis: The design, mathematical modelling and codes are explained in detail in this chapter.
- **Chapter 4**, Physical Model Development, Controls and Testing: The procedures carried out for fabrication and testing are highlighted in this chapter along with detailed description of controls.
- **Chapter 5**, Results and Discussion: The results are demonstrated with the help of graphs and tables in this chapter. Their discussions are also included.
- **Chapter 6,** Impact and Economic Analysis: The social and environmental impact of the project, identified risks and hazards and sustainability analysis is briefed in this chapter.
- **Chapter 7**, Conclusion and Future Recommendation: The conclusions in relation to the problem statement discussed in chapter 1 and future improvements with the present work deficiencies in mind is included in the chapter.

The references are provided at the end of the report.

1.5 Project Schedule/ Timeline



FYP TIMELINE

	Task Description	Start Date	End Date	Status
1	Lietrature Review	02/09/2019	16/09/2019	Completed
2	Define Scope	26/08/2019	01/09/2019	Completed
3	Work Breakdown Structure	26/08/2019	01/09/2019	Completed
4	Solve Dynamics	02/09/2019	10/12/2019	Completed
5	CAD Modelling	15/10/2019	15/11/2019	Completed
6	Micro-Controller Programming	09/09/2019	31/12/2019	Pending
7	Motor/Driver Selection	25/11/2019	01/12/2019	Completed
8	;chematic Diagram and Product Architectur	15/11/2019	15/12/2019	Completed
9	Configuration Design/ Detailed Drawings	15/11/2019	31/12/2019	Completed
10	Material Selection	01/01/2020	15/01/2020	Completed
11	Manufacturing of Individual Parts	15/01/2019	07/02/2019	Completed
12	Integration and Final Assembly	15/12/2019	15/02/2020	Completed
13	Wiring	15/01/2020	15/02/2020	Completed
14	Video Capturing and Calibrating	15/02/2020	01/03/2020	Pending

Fig 1-1: Project Timeline

1.6 Individual and Team Contribution

The team consists of 4 members. The division of work amongst the members is done as follows:

- **Idrees (Dynamics and Mathematical Modelling):** Responsible for solving dynamics, mathematical modelling in MATLAB and for developing a Simscape Multibody model of the robot.
- **Mustansar (Design and Fabrication):** Responsible for the design and fabrication of the robot based on the design.
- **Amina (CAD model and documentation):** Responsible for the CAD model design and report writing.
- **Owais (Controls):** Responsible for the controls/Programming, Wiring, User Interface and Control, and Testing.

Chapter 2

Literature Review

2.1 Literature Review

There are several bipedal robot designs proposed in the past. Bipedal locomotion is one of the most important features of humanoid robots. Major work for bipedal robots focuses on two things:

- Agility
- Stability

Since, this project deals with the mechanism design of the leg structure, different design approaches are considered.

1) **Serial-Parallel Hybrid Leg Mechanism:**
   A bipedal robot that utilizes the Hybrid Leg mechanism. It is a leg mechanism that achieves 6 DOF with a combined structure of serial and parallel mechanism. It is designed to have a light structural inertia and large **workspace** for **agile** bipedal locomotion. [2]

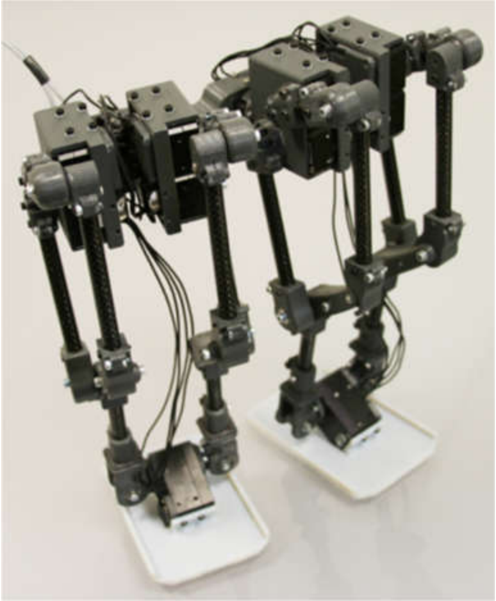


Fig 2-1: Serial-Parallel Hybrid Leg Mechanism *[3]*

2) **SLIDER: Knee-less Legs and Vertical Hip Sliding Motion:**
A bipedal walking robot with ultra-lightweight legs. Each leg has 4 DOF: hip pitch, hip roll, vertical hip slide and ankle pitch, resulting in 8 DOF in total. Compared with the conventional leg, SLIDER's leg doesn't have knees and has a vertical hip sliding motion. [4]

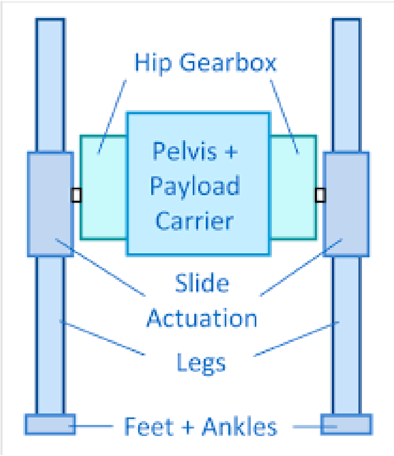


Fig 2-2: Slider: Knee-less Legs and Vertical Hip Sliding Motion *[5]*

3) **CASSIE:**
A bipedal robot designed for dynamic walking and running over all land terrains. Cassie has a 5-degrees-of-freedom in one leg with 3 DOF hip, allowing it to move its legs forward and backward, side to side, and rotate at the same time. And with powered ankles, it can stand in place without having to constantly move its feet. [6]

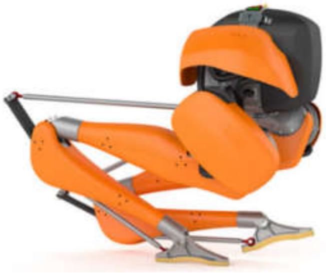


Fig 2-3: Cassie *[7]*

Reference Design: BARt-UH:

This reference design consists of 10 DOF with 5 DOF in each leg; 3 in hip, 1 in knee and 1 in ankle. BARt-UH, a bipedal autonomous robot [8] is selected as the reference design. The reason for selecting this design as the reference design is its comparatively easier mathematical modelling using the Multiple Mass Inverted Pendulum Mode (MMIPM) which assumes the ZMP (Zero Moment Point) at the pivot point of the supporting leg that acts as an inverted pendulum approach [1]. Thus, it eliminates the need for the calculation of the complex ZMP trajectory. The MMIPM method is discussed in detail in Chapter 3.

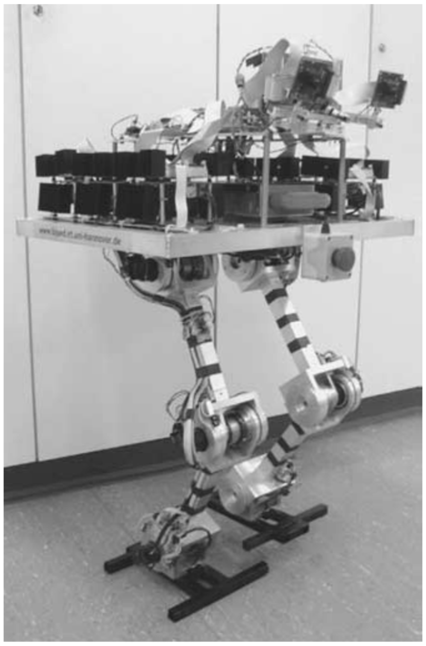


Fig 2-4: Bipedal robot BARt-UH *[1]*

ZMP is defined as a point located on the ground surface where the sum of the moments due to gravity forces and inertial forces in the plane of the foot are zero. Only vertical component is present. In other words, this is the point where the ground reaction forces and a ground reaction moment perpendicular to the plane of foot acts. It follows that if this point is within the stability area, the robot is stable and will not topple over during walking. This stability area depends on the walking phase and is bounded by the points where feet contact the ground. Generally, trunk motions are used to stabilize the locomotion of bipedal robots resulting in large leg trajectories. To reduce the motion range of the trunk a desired ZMP trajectory is generated. The trajectory is based upon the leg trajectories that are arbitrary selected. ZMP trajectory increases the stability of the locomotion. [9]

Another approach is the use of the genetic algorithm for obtaining an optimum result in control and motion synthesis of the biped robot. The genetic algorithms usually generate configurations of the robots to minimize the energy. Due to the long optimization process for the real-time application, genetic algorithms are hardly implemented.

The Inverted Pendulum method (IPM) is a simpler approach based on the Zero Moment Point technique. The IPM method is usually applied for robots without a trunk, in which the leg trajectories are solely responsible for overall motion as well as stability. The IPM approach works on the principle of taking concentrated mass at the torso while neglecting all other masses. The movement of the swinging leg is prescribed, and the trajectory of the torso is adjusted only with respect to the gravitational effect of the swinging leg. When the model is exact and perturbations are neglected, the torso motion is yielded in such a way that the desired motion of the ZMP is achieved [1]. The IPM method is discussed in Chapter 3.

2.2 Inferences drawn out of literature

The ease and practicality of the ZMP approach makes it the most popular criteria for path planning. Listed below are a few reasons to opt for this approach:

- The simplest method to generate trajectories is the inverted pendulum, but upon implication the stability of the biped robot is compromised by the masses that are neglected in generating the trajectory.
- The ZMP coincides with the Center of Pressure (the point on the ground surface where the resultant force of the ground reaction forces will act). This is because, the forces that act on the foot through the ground, are acting on the Zero Moment Point.
- This equivalence makes the concept of the ZMP very suitable for path planning because no external forces or torques act i.e. the process is made simpler and easier.
- By assuming the ZMP at the center of the foot, the mathematical model is simplified.

2.3 Summary

This chapter consists of a brief overview of the literature related to bipedal robots and the pervious work done in the robotics field. It provides basic definitions and terminologies related to bipedal robots. It also summarizes the inferences and conclusions drawn out of the literature and briefs the approach that we will be using for this project.

Chapter 3

Design and Analysis

3.1 Design Methodology and Mathematical Modelling

As discussed in the literature review, to solve the trajectory of the torso, we assume that the ZMP lies exactly at the centre of the foot. This assumption simplifies the calculations as it is impossible to solve both for the ZMP trajectory and the torso trajectory. The ZMP trajectory during walking can be calculated by employing the use of pressure sensors which is beyond the scope of this project. In order to solve for the trajectory, an insight into the IPM, TMIPM and MMIPM is necessary with a suitable algorithm. This is shown in Figure 3-1. At first, the foot trajectory is defined for a one step using a 3rd order polynomial and then the TMIPM method is applied to calculate the torso trajectory on the basis of the defined foot trajectory. Once the torso trajectory, $z_3(t)$, is evaluated, the inverse kinematics is applied to get the corresponding joint angles that will give the required torso trajectory (task space). After this, the forward kinematics is applied to generate the trajectories of the 3 masses attached to the swing leg. These are used to proceed to the MMIPM model which cannot be applied unless the trajectories of these 3 masses is known. This trajectory is now compared with the previous one which was obtained using the TMIPM method. If the error is large, the algorithm is repeated until a sufficient convergence is achieved. These steps are explained in detail in the following sections of the report.

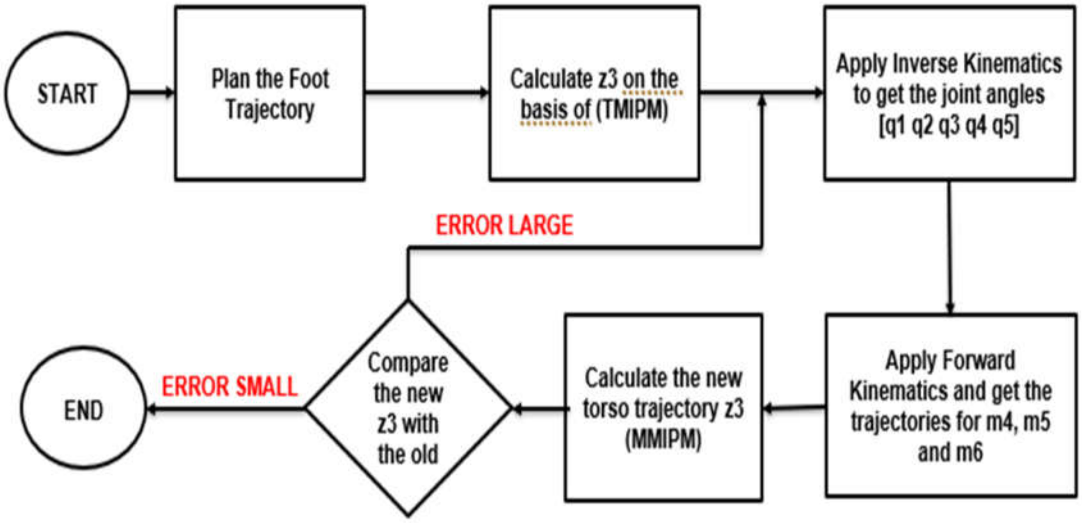

START
Plan the Foot Trajectory
Calculate z3 on the basis of (TMIPM)
Apply Inverse Kinematics to get the joint angles [q1 q2 q3 q4 q5]
ERROR LARGE
Apply Forward Kinematics and get the trajectories for m4, m5 and m6
Calculate the new torso trajectory z3 (MMIPM)
Compare the new z3 with the old
ERROR SMALL
END

Fig 3-1: Algorithm Flowchart

3.1.1 Defining the Foot Trajectory

The foot trajectories are defined using a 3rd order polynomial for the yz motion of the robot. These are shown below:

$$z_6(t) = \frac{SW}{2}\left(3\frac{t}{T_1} - \frac{t^3}{{T_1}^3}\right) \tag{3.1}$$

$$y_6(t) = SH\left(1 - \frac{t^2}{{T_1}^2}\right) \tag{3.2}$$

Where SH and SW are the step height and step width, respectively. At $t = -T1$, $z_6 = -SW, y_6 = 0$, at $t = 0$, $z_6 = 0$, $y_6 = SH$ and at $t = T1$, $z_6 = SW$, $y_6 = 0$. The step starts at $t = -$T1 when the swing leg foot is just leaving the ground and ends at $t = T1$ when the swing leg foot touches the ground.

3.1.2 Inverted Pendulum Model and Stability

In this approach the robot is modelled using only one leg: the supporting leg. A concentrated mass $M = m_{123}(total\ mass\ of\ the\ supporting\ leg)$ is assumed to represent the torso which is kept at a constant height of y_H throughout the motion. The location of the ZMP is assumed to lie at the centre of the supporting foot which should theoretically give the highest gait stability. The robot is stable if the ZMP lies within the foot area. The actual ZMP trajectory can be found by using pressure sensors which is beyond the scope of this project. It can also be calculated theoretically which will be discussed in chapter 5 of this report. Under the action of gravity, the system performs a natural motion of an inverted pendulum, with the horizontal force $F_H(t)$ and vertical force F_V(t) acting at the ZMP. The horizontal force $F_H(t)$ creates a counter moment to stabilize the pendulum. To generate the F_H(t) force, the torso needs to be accelerated and this acceleration introduces the push on the ground through the foot. The torso is accelerated by changing the configuration of the robot (using the joint actuators) with the aid of inverse kinematics which will be discussed later. Figure 3-2 shows the free body diagram of the IPM model. The torso traverses from $-\frac{SW}{2}$ to $\frac{SW}{2}$ while the vertical height (y_H) remains constant. By applying the moment balance at the ZMP we get:

$$\sum Fx = M\ddot{z}_3\,(t) => \ M\ddot{z}_3\,(t) = F_H(t) \tag{3.3}$$

$$\sum Fy = M\ddot{z}_3\,(t) => M\ddot{y}_3\,(t) = F_v(t) - Mg = 0 \tag{3.4}$$

$$\sum Mz = \ 0 => F_H(t)y_H = F_v(t)z_3(t) \tag{3.5}$$

Since $\ddot{y}_3(t) = 0$, as the torso is assumed to be at a constant height. For a Step Width of the torso equal to $\frac{SW}{2}$ and a time duration of $2T_1$ for one step, the differential equation for $z_3(t)$ is obtained from equation 3.3, 3.4 and 3.5. This is:

$$\ddot{z}_3\,(t) - \frac{g}{y_H}z_3(t) = 0 \tag{3.6}$$

$$\lambda = \sqrt{\frac{g}{y_H}}, \quad z_3(\pm T_1) = \pm\frac{SW}{2} \tag{3.7}$$

This equation can be easily solved either analytically or using a numerical method which employs a software like MATLAB. However, this method does not take into account the motion of the swing leg which has a considerable effect on the stability of the robot. Therefore, a better model, TMIPM (Two mass Inverted Pendulum Model) is considered instead.

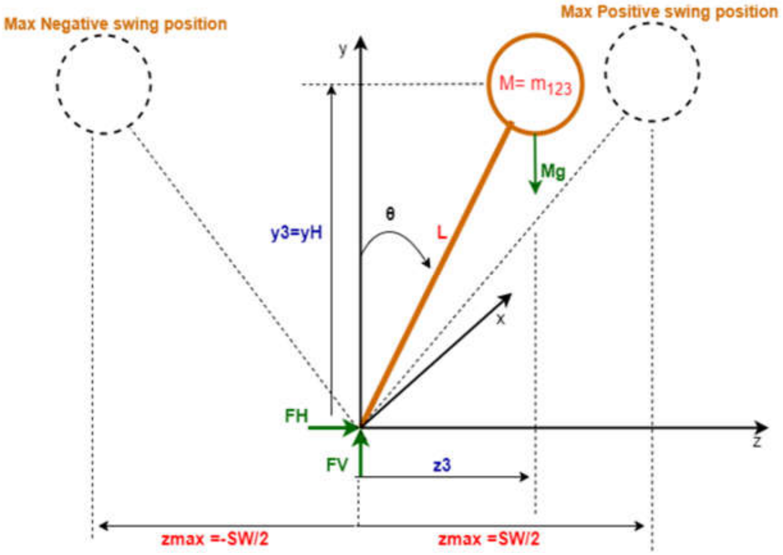


Fig 3-2: Free body Diagram for IPM

3.1.3 Two Masses Inverted Pendulum Mode (TMIPM)

To increase the accuracy of the results, it is desired to develop the motion of the supporting leg as the dynamics of the swing leg affect the motion of the torso. This is accomplished using a TMIPM model. The mass $\boldsymbol{m_6}$ (swing leg foot) is assumed to reside at the foot of the swing leg. The mass **M=**$\boldsymbol{m_{123}}$ represents the remaining masses of the robot and is assumed to reside at the torso. This is shown in Figure 3-3. The ODE for the torso trajectory is derived below:

For maintaining a constant torso height, we introduce the relations:

$$y_3(t) \equiv y_H \tag{3.8}$$

$$\ddot{y}_3(t) \equiv 0 \tag{3.9}$$

Summing the moments about the ZMP:

$$\sum M_{ZMP} = J_G\alpha + \sum \bar{r} \times m\bar{a} \tag{3.10}$$

As masses are assumed to be point masses, therefore, they do not exhibit any moment of inertia about their centre of gravities. Thus, by putting $J_G = 0$, we are left with the gravity and inertial terms only:

$$m_3 g(z_3) + m_6 g(z_6) = \quad m_3 \ddot{z}_3(y_3) - m_3 \ddot{y}_3(z_3) + \ m_6 \ddot{z}_6(y_6) - m_6 \ddot{y}_6(z_6) \tag{3.11}$$

Rearranging the equation, we get:

$$(m_3 y_3)\ddot{z}_3 - (m_3 g)z_3 - (m_3 \ddot{y}_3)z_3 = m_6 g z_6 - m_6 \ddot{z}_6 y_6 + m_6 \ddot{y}_6 z_6 \tag{3.12}$$

$$(m_3 y_3)\ddot{z}_3 + (-m_3 \ddot{y}_3 - m_3 g)z_3 = m_6(g z_6 + z_6 \ddot{y}_6 - y_6 \ddot{z}_6) \tag{3.13}$$

Dividing by $m_3 y_3$ we get,

$$\ddot{z}_3 + \frac{(-m_3 \ddot{y}_3 - m_3 g)z_3}{m_3 y_3} = \frac{m_6}{m_3 y_3}(g z_6 + z_6 \ddot{y}_6 - y_6 \ddot{z}_6) \tag{3.14}$$

Which is a Second order ODE in z_3, with excitation function, the motion of foot. This is:

$$f(t) = \frac{m_6}{m_3 y_3}(g z_6 + z_6 \ddot{y}_6 - y_6 \ddot{z}_6) \tag{3.15}$$

This excitation function now shows that the motion of torso is dependent on the motion of swing leg. Any change in the trajectory of foot motion will lead to a change in the trajectory of torso motion. Since, $y_3 = y_H(torso\ height)$ and $\ddot{y}_3 = 0$, we get a simplified version of the torso trajectory ODE is:

$$\boldsymbol{\ddot{z}_3(t) - \left(\frac{g}{y_H}\right) z_3(t) = \frac{m_6}{m_3 y_H}(g z_6 - y_6 \ddot{z}_6 + z_6 \ddot{y}_6)} \tag{3.16}$$

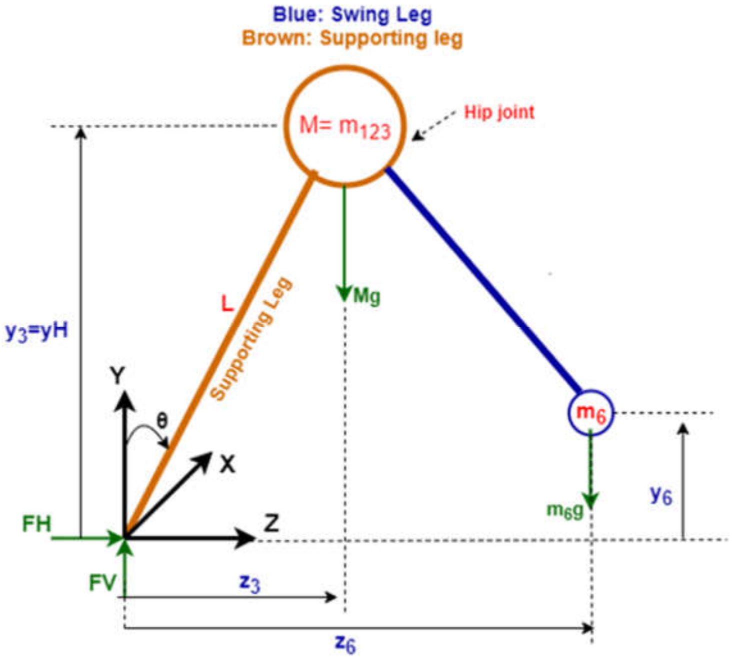


Fig 3-3: Free body Diagram for TMIP

3.1.4 Multiple Masses Inverted Pendulum Mode (MMIPM)

In this model, three masses are considered to model the swing leg. This approach increases the accuracy of the results by including the dynamics of the swing leg represented by 3 separate masses. The masses $m_4, m_5\ and\ m_6$ constitute the swing leg and represent the upper leg, lower leg and the foot respectively. This is shown in figure 3-4.

By taking moments about the ZMP point, we get:

$$\sum M_{ZMP} = J_G \propto + \sum \bar{r} \times m\bar{a} \tag{3.17}$$

As the masses are point masses, they do not exhibit any inertias about their own center of gravity ($J_G = 0$), the gravity and inertial terms are:

$$z_3(m_3 g) + z_4(m_4 g) + z_5(m_5 g) + z_6(m_6 g) = (m_3\ddot{z}_3)y_3 - (m_3\ddot{y}_3)z_3 + (m_3\ddot{z}_4)y_4 - (m_3\ddot{y}_4)z_4 + (m_3\ddot{z}_5)y_5 - (m_3\ddot{y}_5)z_5 + (m_3\ddot{z}_6)y_6 - (m_3\ddot{y}_6)z_6 \tag{3.18}$$

Taking the terms of (m_3) on one side we get,

$$\begin{aligned}(m_3\ddot{z}_3)y_3 + (-m_3\ddot{y}_3 - m_3 g)z_3 \\ &= (m_4 g)z_4 + (m_5 g)z_5 + (m_6 g)z_6 - (m_4 y_4)\ddot{z}_4 + (m_4 z_4)\ddot{y}_4 \\ &- (m_5 y_5)\ddot{z}_5 + (m_5 z_5)\ddot{y}_5 - (m_6 y_6)\ddot{z}_6 + (m_6 z_6)\ddot{y}_6\end{aligned} \tag{3.19}$$

Adjusting the terms,

$$\begin{aligned}(m_3 y_3)\ddot{z}_3 + (-m_3\ddot{y}_3 - m_3 g)z_3 \\ &= (m_4 g)z_4 + (m_4 z_4)\ddot{y}_4 - (m_4 y_4)\ddot{z}_4 + (m_5 g)z_5 + (m_5 z_5)\ddot{y}_5 \\ &- (m_5 y_5)\ddot{z}_5 + (m_6 g)z_6 + (m_6 z_6)\ddot{y}_6 - (m_6 y_6)\ddot{z}_6\end{aligned} \tag{3.20}$$

These gravity and inertial terms can be written in the form of a summation block as:

$$(m_3 y_3)\ddot{z}_3 + (-m_3\ddot{y}_3 - m_3 g)z_3 = \sum_{i=4}^{6} (g m_i z_i + (m_i z_i)\ddot{y}_i - (m_i y_i)\ddot{z}_i) \tag{3.21}$$

Dividing by $m_3 y_3$ and taking $y_3 = y_H, \ddot{y}_3 = 0, m_3 = M$, we get:

$$\boldsymbol{\ddot{z}_3(t) - \frac{g}{y_H} z_3(t) = \frac{1}{M y_H} \sum_{i=4}^{6} m_i(g z_i + z_i \ddot{y}_i - y_i \ddot{z}_i)} \tag{3.22}$$

Where the excitation function f(t) $= \frac{1}{M y_H} \sum_{i=4}^{6} m_i(g z_i + z_i\ddot{y}_i - y_i\ddot{z}_i)$ (3.23)

The excitation function is now not only dependent on the foot leg motion, but also the upper and lower leg motions. Thereby, more accurately modelling the dynamics of the robot. Any change in the trajectory of the upper and lower leg or the foot will lead to a change in the trajectory of the torso. It should be noted that this differential equation cannot be solved directly as it contains unknown terms due to the upper and lower leg motions as discussed previously in the algorithm. Therefore, to solve it, the upper and lower leg trajectories need to be known.

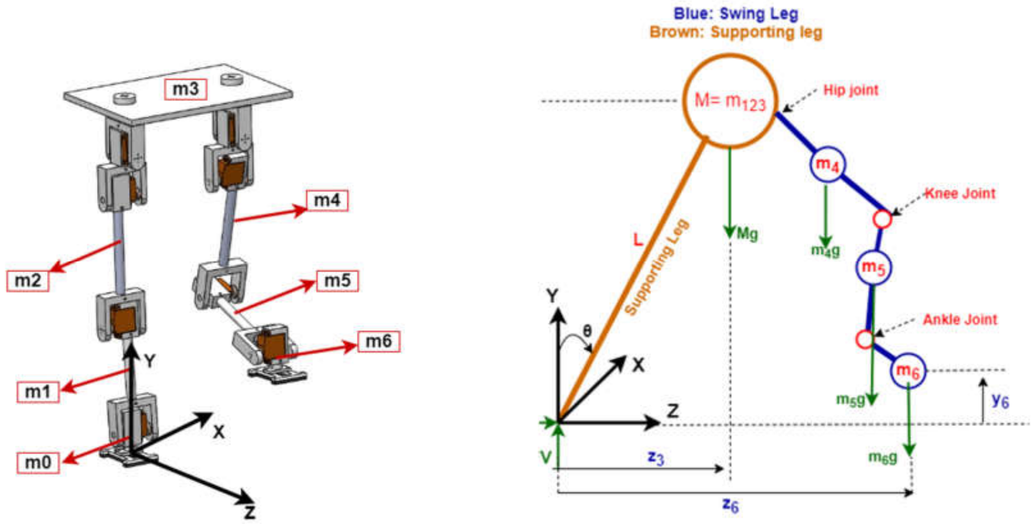


Fig 3-4: Free body Diagram for MMIPM

These trajectories are dependent on the motion of the foot so, in the first iteration the torso trajectory for the TMIPM method is obtained and then with the aid of inverse kinematics and forward kinematics, the cg trajectories of the upper and lower legs are found. Finally, the new torso trajectory is obtained from MMIPM model whose terms are now known. This torso trajectory is now compared with the previous one that was obtained from TMIPM and the error is computed. The algorithm is repeated until a sufficient convergence is achieved.

3.1.5 Inverse Kinematics

To calculate the joint angles, inverse kinematics is applied. Figure 3-5 shows the model for inverse kinematics with corresponding joint angles and coordinates.

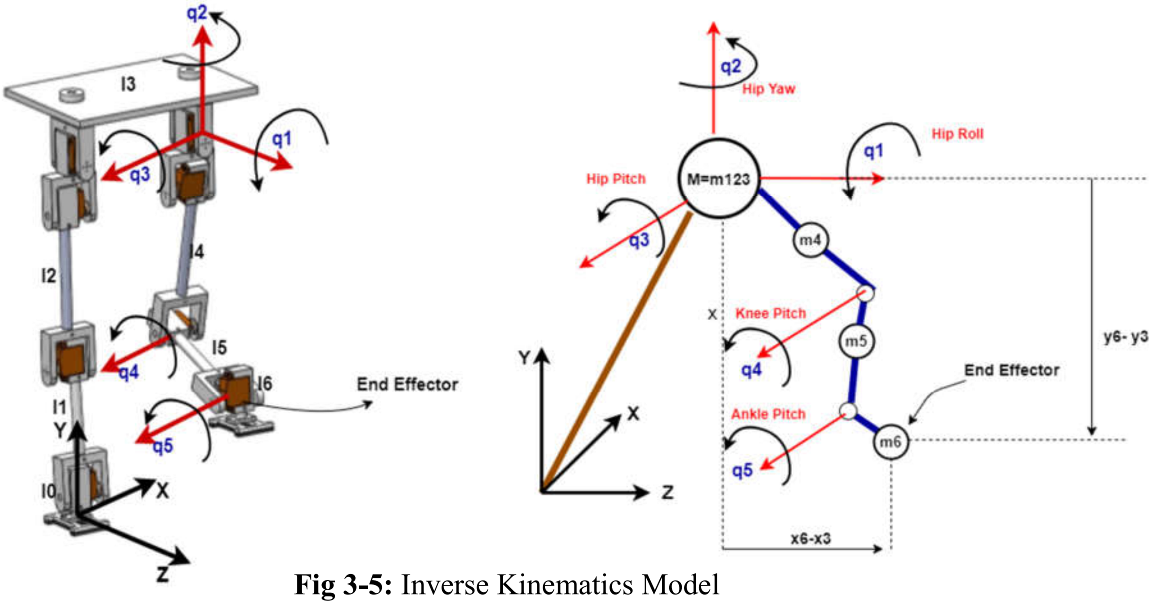


Fig 3-5: Inverse Kinematics Model

The input to the inverse kinematics solver is the foot yz trajectory of foot relative to the torso as shown in figure 3-5 and output is the 5 joint angles. The DH table is defined below:

I	α_{i-1}	a_{i-1}	d_i	q_i	
1	0	0	0	q_1	Roll (Hip)
2	$-\pi/2$	0	0	$q_2 - \pi/2$	Yaw (Hip)
3	$\pi/2$	0	0	q_3	Pitch (Hip)
4	0	l_4	0	q_4	Knee Pitch
5	0	l_5	0	q_5	Ankle Pitch
6	0	l_6	0	q_6	EE offset

DH Table

According to the DH table, the transformation matrices are

$$ {}^0_1T = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad {}^1_2T = \begin{bmatrix} s_2 & c_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ c_2 & -s_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad {}^2_3T = \begin{bmatrix} c_3 & -s_3 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ c_3 & c_3 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^3_4T = \begin{bmatrix} c_4 & -s_4 & 0 & l_4 \\ s_4 & c_4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad {}^4_5T = \begin{bmatrix} c_5 & -s_5 & 0 & l_5 \\ s_5 & c_5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad {}^5_6T = \begin{bmatrix} 1 & 0 & 0 & l_6 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Similarly, to find the transformation matrix, ${}^0_6T$

$$ {}^0_6T = {}^0_2T\,{}^2_6T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & px^0 \\ r_{21} & r_{22} & r_{23} & py^0 \\ r_{31} & r_{32} & r_{33} & pz^0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.24) $$

Where,

$$ {}^0_2T = \begin{bmatrix} c_1s_2 & c_1c_2 & -s_1 & 0 \\ s_1s_2 & c_2s_1 & c_1 & 0 \\ c_2 & -s_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ and } {}^2_6T = \begin{bmatrix} c_{345} & -s_{345} & 0 & l_4c_3 + l_5c_{34} + l_6c_{345} \\ 0 & 0 & -1 & 0 \\ s_{345} & c_{345} & 0 & l_4s_3 + l_5s_{34} + l_6s_{345} \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ ({}^0_2T)^{-1}\ {}^0_6T = {}^2_6T; \qquad {}^0_2T^{-1} = \begin{bmatrix} c_1s_2 & s_1s_2 & c_2 & 0 \\ c_1c_2 & c_2s_1 & -s_2 & 0 \\ -s_1 & c_1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.25) $$

With all the matrices found, the rows and columns are multiplied turn by turn to find the joint angles,

Firstly, Row 3 × Column 3 are compared to get:

$$[-s_1 \quad c_1 \quad 0 \quad 0]\begin{bmatrix} r_{13} \\ r_{23} \\ r_{33} \\ 0 \end{bmatrix} = 0\ ; \qquad (3.26)$$

$$-s_1 r_{13} + c_1 r_{23} = 0 \qquad (3.27)$$

From equation 3.21, θ_1 can be found using the Transcendental equations.

Similarly, by comparing Row 2 × Column 3

$$[c_1 c_2 \quad c_2 s_1 \quad -s_2 \quad 0]\begin{bmatrix} r_{13} \\ r_{23} \\ r_{33} \\ 0 \end{bmatrix} = -1\ ; \qquad (3.28)$$

$$(c_1 c_2) r_{13} + (c_2 s_1) r_{23} - s_2 (r_{33}) = -1 \qquad (3.29)$$

$$w_{34} = l_4 s_3 \quad + l_5 s_{34} + l_{56} w_{31} \qquad (3.30)$$

Finally, by comparing Row 1 × Column 3

$$[c_1 s_2 \quad s_1 s_2 \quad c_2 \quad 0]\begin{bmatrix} r_{13} \\ r_{23} \\ r_{33} \\ 0 \end{bmatrix} = 0\ ; \qquad (3.31)$$

$$(c_1 s_2) r_{13} + (s_1 s_2) r_{23} + c_2 (r_{33}) = 0$$

$$-s_1 r_{13} + c_1 r_{23} = 0 \qquad (3.32)$$

Using Transcendental Technique $a \cos\theta + b \sin\theta = c$ (3.33)

$$(c_1 r_{13} + s_1 r_{23}) c_2 - (r_{33}) s_2 = -1 \qquad (3.34)$$

$$(r_{33}) c_2 + (c_1 r_{13} + s_1 r_{23}) s_2 = 0 \qquad (3.35)$$

Here we apply Cramer's rule to find the (y, z) motion (3.36)

For fining the x trajectory (3D Motion), we compare Row 2 × Column 4 (3.37)

$$[c_1 c_2 \quad c_2 s_1 \quad -s_2 \quad 0]\begin{bmatrix} px^0 \\ py^0 \\ pz^0 \\ 1 \end{bmatrix} = 0\ ; \qquad (3.38)$$

$$(c_1 c_2) px^0 + c_2 s_1 py^0 - s_2 pz^0 = 0 \qquad (3.39)$$

$$(c_1px^0 + s_1py^0)c_2 - (pz^0)s_2 = 0 \tag{3.40}$$

Since θ_1 and θ_2 are known, we can calculate ${}^0_2\mathrm{T}^{-1}$,

$${}^0_2T^{-1}\,{}^0_6T = {}^2_6T \tag{3.41}$$

$$\begin{bmatrix} w_{11} & w_{12} & w_{13} & w_{14} \\ w_{21} & w_{22} & w_{23} & w_{24} \\ w_{31} & w_{32} & w_{33} & w_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} = {}^2_6T \tag{3.42}$$

The elements of this matrix are compared with the elements of ${}^2_6T$ which yields the following:

$$w_{14} = l_4c_3 \quad + l_5c_{34} \quad + l_6c_{345} \tag{3.43}$$

$$w_{34} = l_4s_3 + l_5s_{34} \quad + l_6s_{345} \tag{3.44}$$

$$w_{11} = c_{345} \tag{3.45}$$

$$w_{31} = s_{345} \tag{3.46}$$

$$w_{14} = l_4c_3 \quad + l_5c_{34} + l_6w_{11} \tag{3.47}$$

$$(w_{14} - l_6w_{11}) - l_4c_3 \quad + l_5c_{34} \tag{3.48}$$

$$(w_{34} - l_6w_{11}) - l_4s_3 \quad + l_5s_{34} \tag{3.49}$$

$$P_x = r_{13};\ P_y = r_{23};\ d_3 = 0 \tag{3.50}$$

Finally, all the joint trajectories are calculated using the following relations:

$\boldsymbol{q_1(Hip\ Roll)}$**:**

$$-s_1r_{13} + c_1r_{23} \quad = 0; \tag{3.51}$$

$$q_1 = a\tan 2(r_{23}, r_{13} - atan2(0, \pm\sqrt{{r_{13}}^2 - {r_{23}}^2})) \tag{3.52}$$

$$\boldsymbol{q_1 = a\tan 2(r_{23}\,, r_{13})} \tag{3.53}$$

$\boldsymbol{q_2(Hip\ Yaw)}$:

$$(c_{13}r_{13} + s_1r_{23})c_2 - (r_{33}) = -1 \tag{3.54}$$

$$(r_{33})c_2 + (c_1r_{13} + s_1r_{23})s_2 = 0 \tag{3.55}$$

$$A = \begin{bmatrix} c_{13}r_{13} + s_1r_{23} & -r_{33} \\ r_{33} & c_1r_{13} + s_1r_{23} \end{bmatrix} \tag{3.56}$$

$$\begin{bmatrix} c_{13}r_{13} + s_1 r_{23} & -r_{33} \\ r_{33} & c_1 r_{13} + s_1 r_{23} \end{bmatrix} \begin{bmatrix} c_2 \\ s_2 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \end{bmatrix} \tag{3.57}$$

$$c_2 = \begin{vmatrix} -1 & -r_{33} \\ 0 & c_1 r_{13} + s_1 r_{23} \end{vmatrix} \Big/ \det(A) \tag{3.58}$$

$$s_2 = \begin{vmatrix} c_{13}r_{13} + s_1 r_{23} & -1 \\ r_{33} & 0 \end{vmatrix} \Big/ \det(A) \tag{3.59}$$

$$(\boldsymbol{q_2}) = \boldsymbol{a\tan 2(s_2, c_2)} \tag{3.60}$$

$\boldsymbol{q_4(Knee\ Pitch)}$:

$$x' = w_{14} - l_6 w_{11} = l_4 c_3 + l_5 c_{34} \tag{3.61}$$

$$y' = w_{34} - l_6 w_{31} = l_4 s_3 + l_5 s_{34} \tag{3.62}$$

$$c_4 = \frac{({x'}^2 + {y'}^2 - {l_4}^2 - {l_5}^2)}{2 l_4 l_5} \tag{3.63}$$

$$s_4 = -\sqrt{1 - {c_4}^2} \tag{3.64}$$

$$(\boldsymbol{q_4}) = \boldsymbol{a\tan 2(s_4, c_4)} \tag{3.65}$$

$\boldsymbol{q_3(Hip\ Pitch)}$:

$$B1 = \begin{bmatrix} x' & -l_5 s_4 \\ y' & l_4 + l_5 c_4 \end{bmatrix} \tag{3.66}$$

$$B2 = \begin{bmatrix} l_4 + l_5 c_4 & x' \\ l_5 s_4 & y' \end{bmatrix} \tag{3.67}$$

$$B = \begin{bmatrix} l_4 + l_5 c_4 & -l_5 s_4 \\ l_5 s_4 & l_4 + l_5 c_4 \end{bmatrix} \tag{3.68}$$

$$c_3 = \frac{|B1|}{|B|} \tag{3.69}$$

$$s_3 = \frac{|B2|}{|B|} \tag{3.70}$$

$$\boldsymbol{q_3 = atan2(s_3, c_3)} \tag{3.71}$$

$\boldsymbol{q_5}$**: (Ankle Pitch)** (3.72)

$$c_{345} = w_{11} \tag{3.73}$$

$$s_{345} = w_{31}$$

$$(\boldsymbol{q_{345}}) = \boldsymbol{a}\,\mathbf{tan}\,\mathbf{2}(\boldsymbol{w_{31}}, \boldsymbol{w_{11}}) \tag{3.74}$$

$$(\boldsymbol{q_5}) = \boldsymbol{q_{345}} - (\boldsymbol{q_3} + \boldsymbol{q_4}) \tag{3.75}$$

The inverse kinematics is implemented using MATLAB code which is shown in Appendix A. The results of the inverse kinematics, the torso trajectories and the joint trajectories are discussed further in the results and discussion section of the report.

3.1.6 Dynamics and Joint Torques

After the inverse kinematics, the joint trajectories are obtained as a function of time. These trajectories are then converted into velocity and acceleration by taking first and second order derivatives of the joint trajectories respectively. Velocity is referred to first order kinematics and acceleration to second order kinematics. The position, velocity and the acceleration terms are required for calculating the joint torques. These torques can be calculated by using the following expression (Newton Law):

$$\tau = M(\theta)\ddot{\theta} + V\left(\theta, \dot{\theta}\right) + G(\theta)$$

Where $M(\theta)$ is the Inertia matrix and contains the inertial information like the mass and moment if inertia, $V\left(\theta, \dot{\theta}\right)$ contains the Coriolis and normal acceleration terms while $G(\theta)$ contains the gravity terms. τ represent the joint toques which are used for the actuation of the revolute joints. The derivation of toque equation in generalized form is tedious and prone to errors, therefore, built in functions in MATLAB and SIMULINK were used to solve the equations. The link masses and inertias were calculated using the CAD model. The results of the torques will be further discussed in chapter 5 of this report.

3.2 MATLAB Code

The equations derived in the mathematical modelling section were implemented on MATLAB and an iterative technique was used to find the stable joint trajectories. The MATLAB Code comprises of the following steps.

1. Selection of a suitable trajectory for the foot motion $(z_6(t), y_6(t), 0)$ and foot rotation $qEE(t)$ using standard polynomials of degree 3 and degree 4 respectively. Where $(z_6(t), y_6(t), 0)$ represents the location of the centre of gravity of the left foot and $qEE(t)$ represents the orientation of the left foot along the pitch axis in a fixed coordinate frame.
2. Solution of the ODE equation using ODE Solver in MATLAB to obtain the torso trajectory $(z_3(t),\ y_H,\ 0)$ for the case of TMIPM method. Where $(z_3(t),\ y_H,\ 0)$ is the location of the centre of gravity of torso in a fixed coordinate frame!

3. Definition of the 5 DOF manipulator using Robotics Toolbox by declaring the DH table. A robot object is then made using the $SerialLink()$ function of the MATLAB Robotics Tool Box.
4. Inverse Kinematics solution using ikine() function of the Robotics Tool box to find the joint variables $q_1(t)\ to\ q_5(t)$.
5. Generation of trajectories for $m_4\ and\ m_5$ $(z_4(t), y_4(t), 0)$ and $(z_5(t), y_5(t), 0)$ using Forward Kinematics. $m_4\ and\ m_5$ lie at the centre of gravity of the upper left and lower left leg.
6. Once these trajectories are obtained, the new excitation function is obtained on the basis of these trajectories. This excitation function is now dependent on the upper, lower and foot trajectories and will lead to more accurate model of the system as the dynamics of the swing leg have been included.
7. The ODE relating the motion of torso to the swing leg motion is now solved using the ODE Solver function and we get a new trajectory function for the torso. This is called $z_{3\ new}(t)$.
8. The Convergence of $z_3(t)$ and $z_{3\ new}(t)$ is checked by calling a function named $ConvergenceCheck()$. This function integrates the error between these two functions i.e. $z_3(t)$and $z_{3\ new}(t)$ and returns an error value.
9. If the error is large, the algorithm is repeated starting from step 4, the inverse kinematics is solved again, the upper and lower leg trajectories are obtained and finally, a new torso trajectory is obtained in order to minimize the error as much as possible. After three iterations, a sufficient convergence was achieved. The results of this code will be further discussed in chapter 5. The MATLAB code is attached in Appendix A.

3.3 SIMULINK Model

The Simulink model was developed to check the interaction of the robot with the physical world virtually. This was done in order to make sure that the robot is stable and does not topple over for the prescribed joint trajectories. The robot was checked virtually for various parameters to see how each and every parameter affects the performance and the torque requirements. Figure 3-6 shows the Simulink model. The model is a modular one which contains 3 main blocks. The Input Signal bock provides the joint trajectories to each leg over a period of step time. The Left Leg and Right Leg contains the structure of the left and right leg which is composed of solid body blocks, rigid transformation blocks and revolute blocks etc. The torso is defined between the two legs. The output of the Leg blocks is the torque which is an important dynamic variable of interest. The torque values will be discussed further in the results and discussion section of the report.

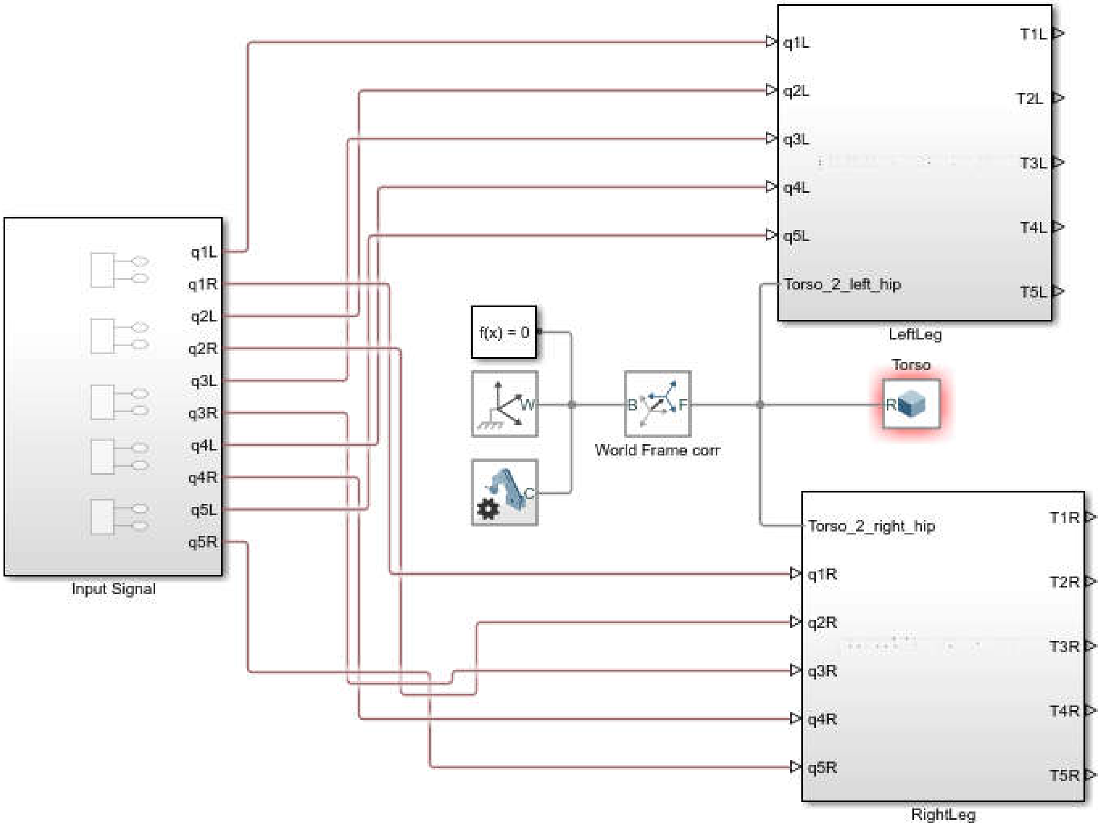


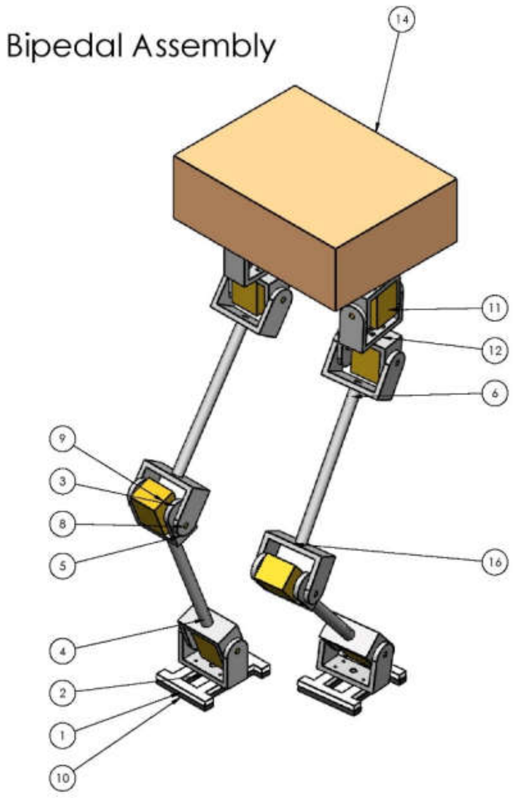
Fig 3-6: Simulink Model

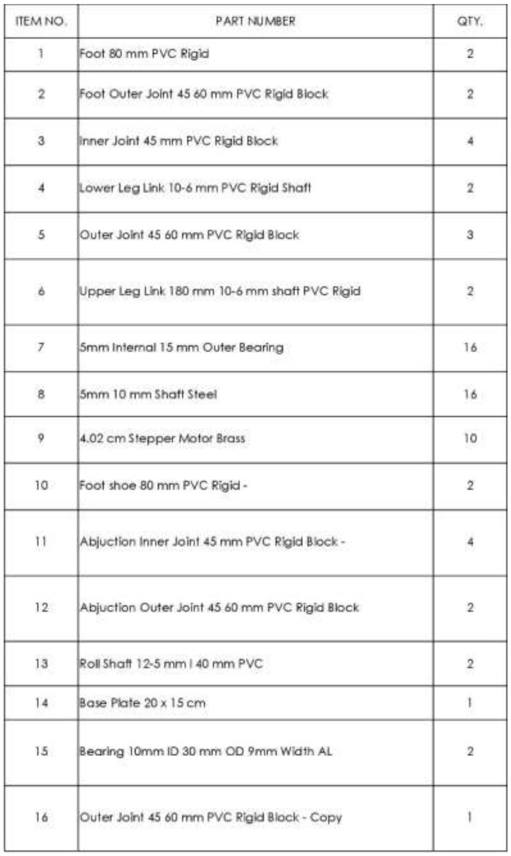
3.4 Design and Modelling

The bipedal was designed by taking various factors into mind and the most important was link masses and inertia. The links and joints are designed to reduce the torque as much as possible. To increase the stability and making the control calculations simple the major mass was confined to the torso. Since the motors and battery constitutes the maximum weight the links and joints are made up of PVC Rigid plastic, which can easily be 3D printed. The detailed design with proper simulations can be done after the present model satisfies the mathematical equations to give minimum torques.

3.4.1 Detailed Design

The geometric design of the bipedal assembly has been made using SolidWorks, major parts are made from the plastic PVC Rigid that can be easily 3D printed. Other than manufacturing point of view the plastic is much lighter than that of metals thus it gives high strength to mass ratio. Initial design for the robot has been made using 16 different components each one has its own function that gives proper rotation to the links. The Bill of material has been attached for the assembly with proper part number and description.





ITEM NO.	PART NUMBER	QTY.
1	Foot 80 mm PVC Rigid	2
2	Foot Outer Joint 45 60 mm PVC Rigid Block	2
3	Inner Joint 45 mm PVC Rigid Block	4
4	Lower Leg Link 10-6 mm PVC Rigid Shaft	2
5	Outer Joint 45 60 mm PVC Rigid Block	3
6	Upper Leg Link 180 mm 10-6 mm shaft PVC Rigid	2
7	5mm Internal 15 mm Outer Bearing	16
8	5mm 10 mm Shaft Steel	16
9	4.02 cm Stepper Motor Brass	10
10	Foot shoe 80 mm PVC Rigid -	2
11	Abjuction Inner Joint 45 mm PVC Rigid Block -	4
12	Abjuction Outer Joint 45 60 mm PVC Rigid Block	2
13	Roll Shaft 12-5 mm l 40 mm PVC	2
14	Base Plate 20 x 15 cm	1
15	Bearing 10mm ID 30 mm OD 9mm Width AL	2
16	Outer Joint 45 60 mm PVC Rigid Block - Copy	1

Fig 3-7: Geometric Model of Assembly with Bill of Materials

3.4.2 Part Description

Male – Female Joint Brackets

At the end of each link 60 mm wide brackets of PVC Rigid have been used, not only to support links but also to place motors and allow rotation at each end of the link. Two types of brackets male and female are designed to fix both ends together. Ball bearings of 5mm internal diameter are placed in the male brackets which holds the female brackets as well as the servos.

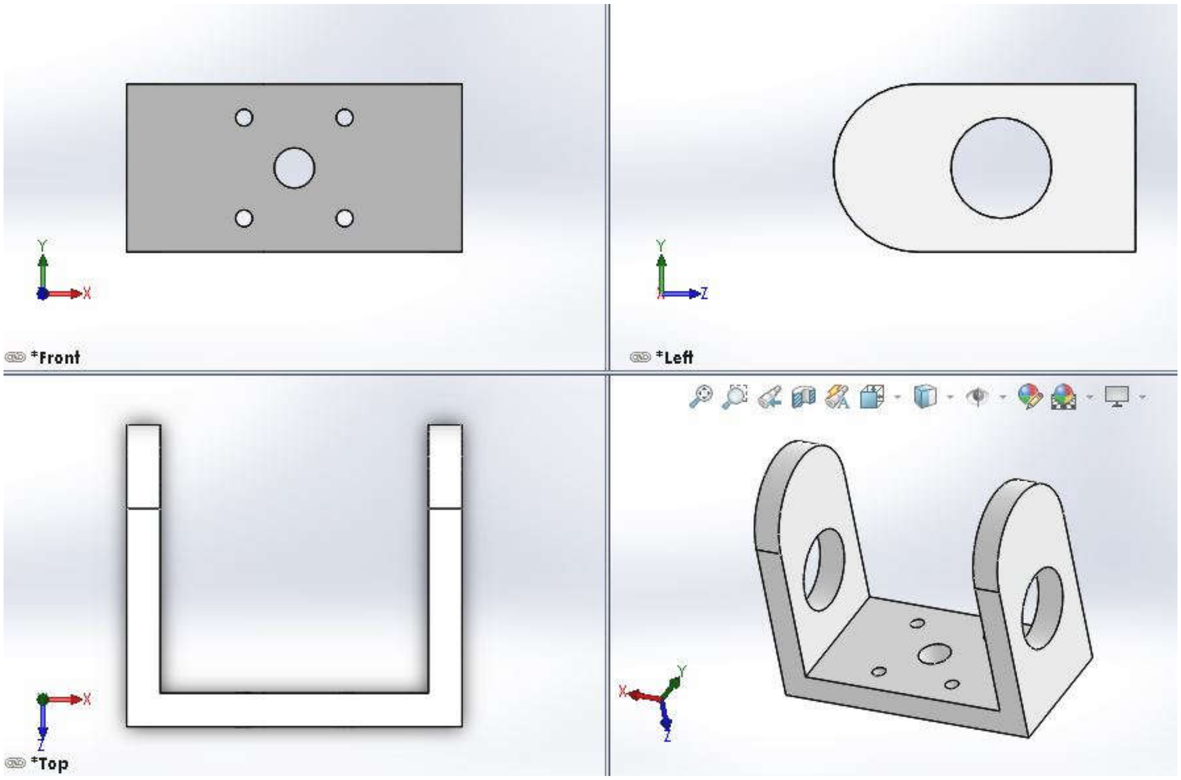


Fig 3-8: Motor Revolute Joint Male Coupler (PVC Rigid)

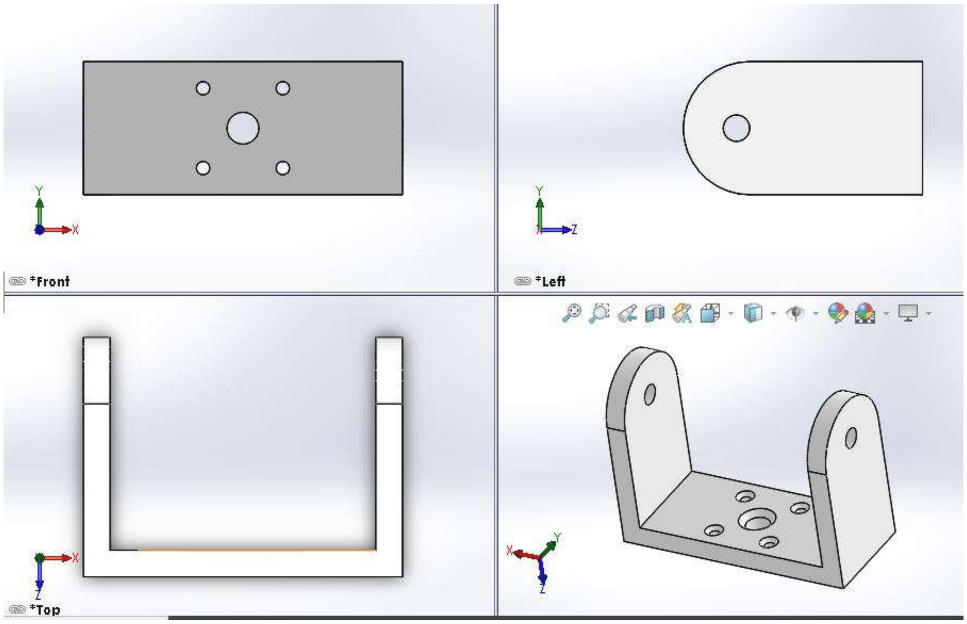


Fig 3-9: Motor Revolute Joint Female Coupler (PVC Rigid)

Shin – Tarsus Links

The main links are made using 10 mm outer with 8 mm inner diameter hollow PVC Rigid shaft. The main reason to use hollow shaft is to increase rigidity as well as to reduce the overall mass of the links. For the tarsus link length (l_1) is 140 mm while for the shin link length (l_2) is 180 mm. Fours such links are used to make both legs. These legs are attached to the brackets by bolts fixing into the internal threads made into the shafts.

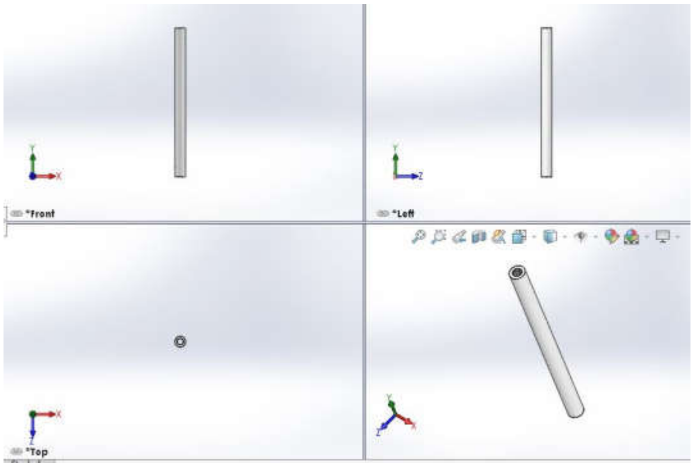


Fig 3-10: Leg Tarsus and Shin Link (PVC Rigid)

Foot Base

The foot base is made two composite layers. The upper layer which provide rigidity is made from PVC rigid which is also attached to the tarsus link using male female brackets. Where the lower layer is made from neoprene rubber just to provide it slide flexible surface so that it can hold properly.

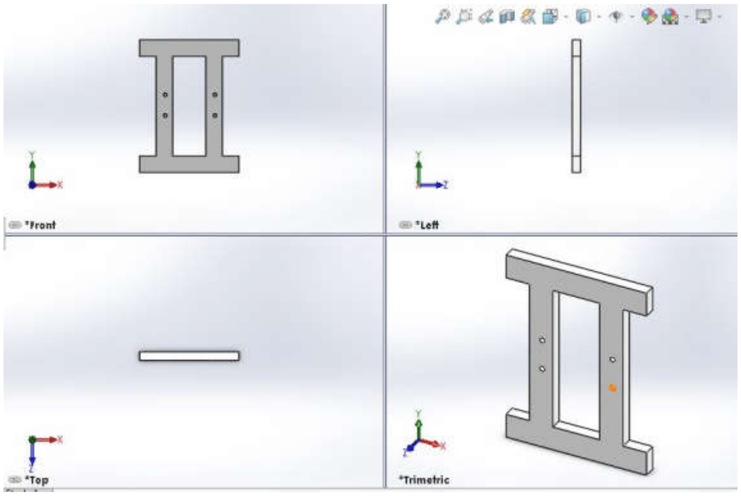


Fig 3-11: Foot Base Plate (PVC Rigid)

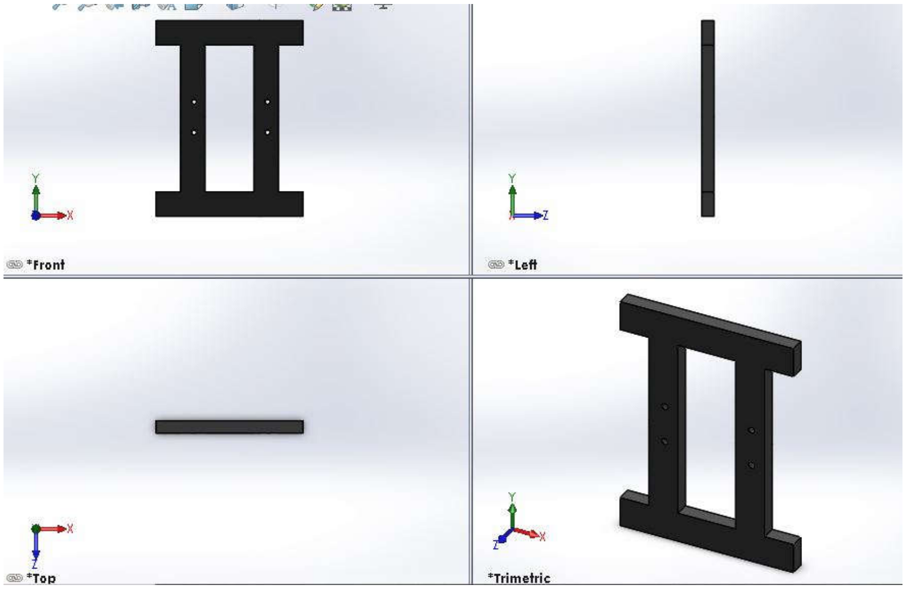


Fig 3-12: Foot Shoe (Neoprene Rubber)

Hip Joint

The main problem from the whole design is to provide 3 DOF at the hip joint, two revolving joints are made for the abduction and pitch. While for the leg revolve hip joint shaft is used which is fixed into the body of torso using thrust type ball bearing. The servo is fixed on the top of torso and the shaft is passed to the male female two adjacent joints thus it can not only revolve but also rotate its legs in all the three axes.

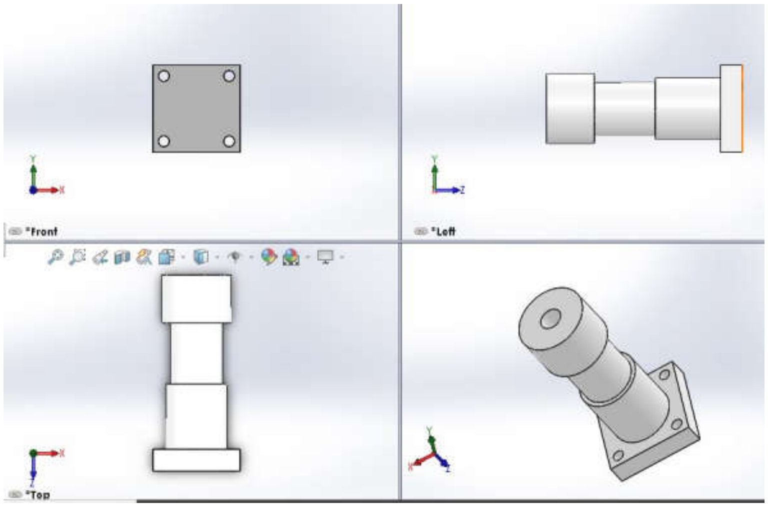


Fig 3-13: Revolving Motor Hip Joint Link (PVC Rigid)

3.5 Summary

This chapter summarizes the alternative design methodologies, mathematical modeling, the selected design approach and codes and analysis used for the project. It also includes the CAD models, the design process flow chart, the MATLAB code and the SIMULINK model.

Chapter 4

Physical Model Development, Controls and Testing

4.1 Development Processes

For the product development of the bipedal finalized design different conventional and advance fabrication processes are used. Selection of the fabrication process for each part depends on the part geometry and the required accuracy in the part for the final assembly. The final design for the manufacturing has been done and the required modification has been made on basis of available machining and fabrication resources. The fabrication processes involved in the physical model development is presented below in the process flow diagram.

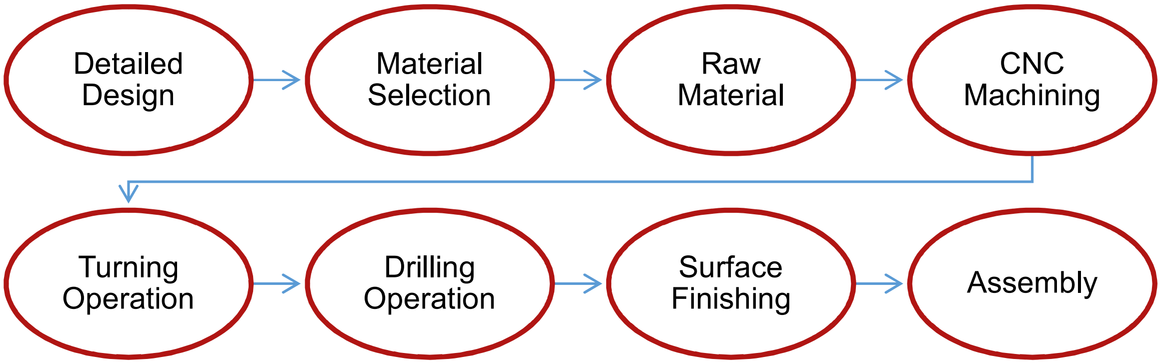


Fig 4-1: Fabrication Process Flow

4.1.1 Detailed Design

The detailed design for manufacturing includes all the minor details of the individual parts. This is required so that the fabrication schedule can be developed, and the relevant process can be selected. Before the prototyping of each part is analysed and the based on the part requirement and available machining resources the suggested modifications in the part has been done. Once, detailed design for individual part is ready, the fabrication can be planned.

4.1.2 Material Selection

After the completion of design for manufacturing the material for the physical model were selected. Based on the loading conditions and the machinability of the part, PTFE (Ploy Tetra Fluoro Ethylene) which is commonly known as TEFLON were selected. Teflon has high strength to mass ratio which means it is much lighter than aluminium or nylons which can also be used. It also presents good machinability and wear resistance.

4.1.3 CNC Machining

Starting off with the roughing, the TEFLON blocks were machined using CNC milling cut. All the parts which includes bipedal leg joints were machined on CNC mill. After roughing with 8 mm end mill cutter, the finish cut was made using 2 mm end mill cutter so that the fillets at the sharp edges can be made as small as possible which are required due to tight fitting of few parts. The machining operation for CNC milling is shown in Fig 4-2 presented below.

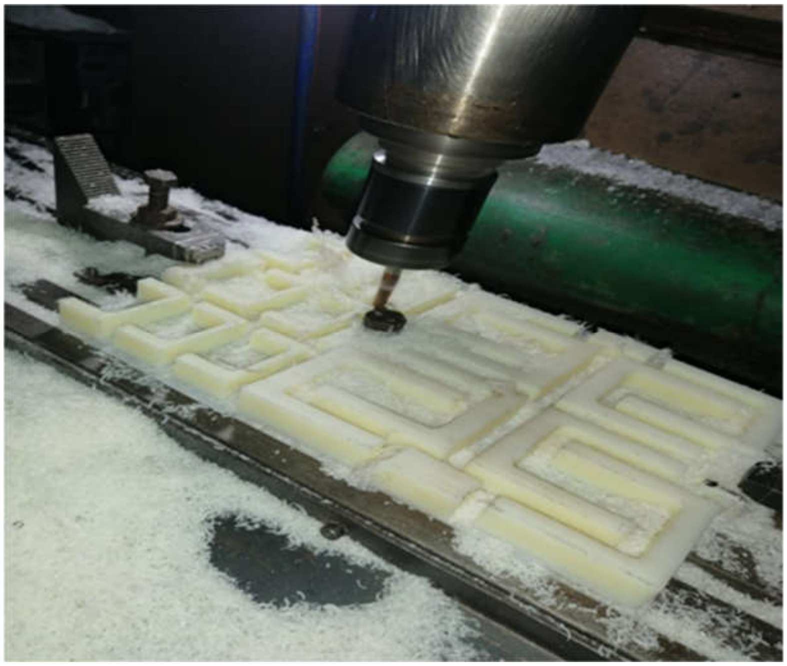


Fig 4-2: CNC milling end mill cutting operation

4.1.4 Turning Operation

For the round geometries which includes Teflon shafts for the leg links, turning operation has been done on conventional lathe machine. The Teflon shafts were made having 15 mm diameter. Link length were determined in the mathematical modelling phase; upper thigh link has larger length as compared to tarsus. All the round profiles were finalized in this section.

4.1.5 Drilling Operation

To assemble all the links and joints, drilling operation were done on the conventional drilling machine. The holes were made with tight fitting tolerances so that the use of screws and bolts can be reduced as much as possible to make light in weight assembly. After drilling, reamers were used to finish the hole and get rid of bur.

4.1.6 Surface Finishing

To remove all the bur, surface finishing operations were done using filling and surface grinder machines. All the parts were finished so that the tight fit tolerances can be utilized, and assembly can be done perfectly. The finished parts are shown in Fig 4-3 which are Teflon bipedal joints to be used as motor housing.

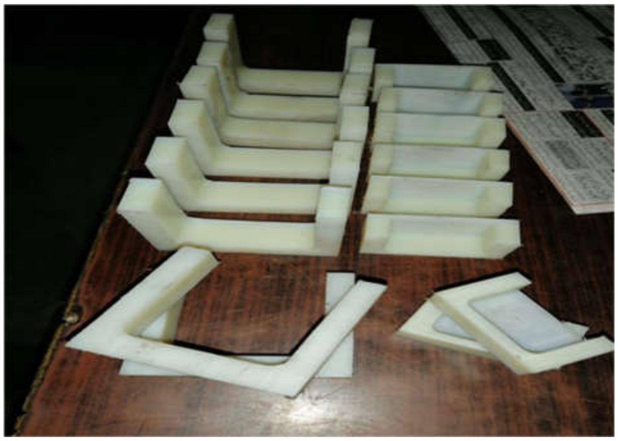


Fig 4-3: Finished motor housing joints

4.1.7 Assembly

After the fabrication of individual parts, the final assembly were done. The parts are joined using tight fit tolerances for the link-joint attachment so that jerks can be reduced, and ease of control can be achieved. For the attachment of motors, self-threading / wooden screws were used, as they can be assembled and disassembled easily. To transfer torque from motor to link 25T metal gear servo steering disks were attached directly to the U joints. Final bipedal assembly contains 2 legs with 5 DOF in each leg.

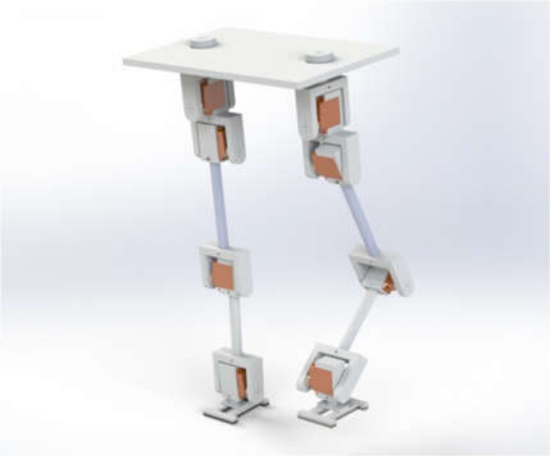


Fig 4-4: Final Assembly

4.2 Integration and Instrumentation

4.2.1 Definitions and Terminologies

- **Ultrasonic Distance Sensor (HC SR-04):** Determines the distance to an object by measuring the time taken by the sound to reflect from that object

- **Raspberry Pi 3B+:** Small single-board computers also known as micro-controller used for many projects and specific industrial applications.

- **Digital Servo Motor:** Rotary actuators that use a small microprocessor to receive and direct action at high frequency voltage pulses. The digital servo sends nearly six times the amount of pulses an analog signal does. These faster pulses provide consistent torque for quicker and smoother response times.

- **Gyroscope (MPU 6050):** The MPU 6050 is a 6 DOF (degrees of freedom) or a six-axis IMU sensor, which means that it gives six values as output: three values from the accelerometer and three from the gyroscope. The main purpose of the gyroscope is to get the orientation of the manipulator and use it to maintain a specific orientation. The MPU 6050 is a sensor based on MEMS (micro electro-mechanical systems) technology. Both the accelerometer and the gyroscope are embedded inside a single chip. This chip uses I2C (inter-integrated circuit) protocol for communication.

4.2.2 Controls

- Each leg is controlled by **3 motors**. One motor in ankle, one in knee and one in hip joint. Currently, we have restricted the motion of each leg to 2 dimensions for simplicity.
- **Servo motors** were selected to control the angular movement of legs due to their precise and accurate angle control.

- **20 kg-cm Digital Servo (DS 3218)** was selected after deciding on factor of safety **4.9**

Fig 4-5: 20kg-cm torque DS 3218 Digital Servo used for the manipulator

- For controlling the motor drivers, **Raspberry Pie 3 B+** (a high performance, micro-controller) was used at first. To program the Raspberry Pi, the recently introduced library known as GPIO Zero was used allowing much more convenience in programming. Using Python as the programming language, we were successful in receiving angle values from MPU 6050 (Gyro and Accelerator Module) using I2c communication protocol. Moreover, HC – SR04 was also successfully returning real-time values to the micro-controller which were then to be used in proximity control. Finally, simple tests were performed on the Digital Servo motors to change their angles. Unfortunately, the motors were **continuously jittering** after moving to a new position.

- Consequently, we checked if the same problem occurred if Arduino is used instead of Raspberry Pi. So, when we used Arduino to change the servo's angle, there was no jittering as opposed to Raspberry Pi. Hence, **Arduino was then selected** to program and move the manipulator.

- At first, our approach was to use **closed-loop feedback** control to make sure that the robot stands dynamically stable and it reaches the angle provided. The real-time actual values from an MPU-6050 (Accelerometer and Gyroscope module) sensor, installed at each joint, were to be used to calculate and then adjust the error using PID (Proportional, Integral, and Derivative Control).

- However, in our case, the Digital Servo Motors have built-in encoders for adjusting any real-time error that occurs thus eliminating the need of a closed-loop feedback control and making our system an **open-loop controlled system.** To verify that our motors have built-in encoders, Arduino's *servo.write()* function is used to analyze the response of the motors when given a simple value of 0, 90, or 180 degrees. If the motors have built-in encoders the motor will respond to the value '0' by moving to its '0' position otherwise it will start moving clockwise continuously.

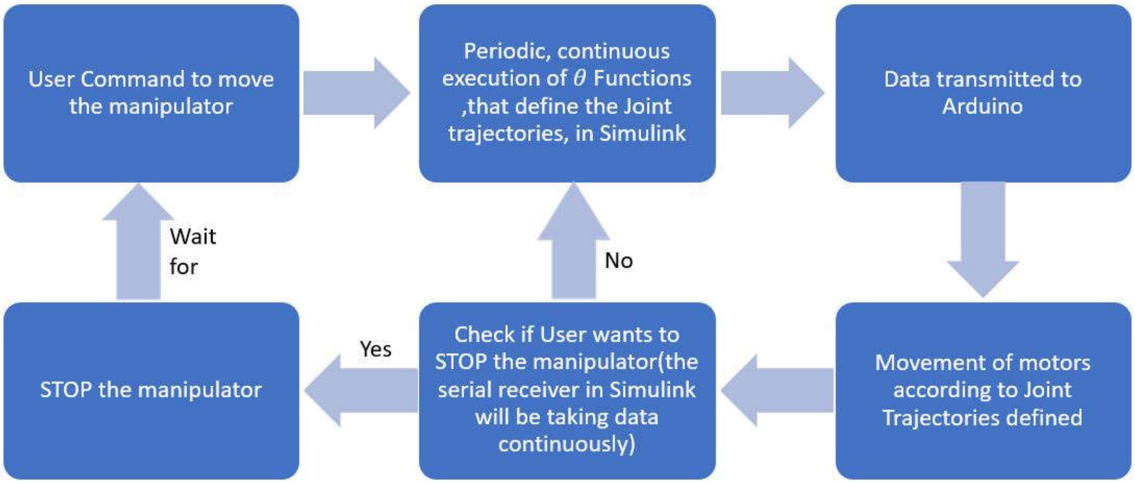

User Command to move the manipulator
Periodic, continuous execution of θ Functions ,that define the Joint trajectories, in Simulink
Data transmitted to Arduino
Movement of motors according to Joint Trajectories defined
Check if User wants to STOP the manipulator(the serial receiver in Simulink will be taking data continuously)
No
Yes
STOP the manipulator
Wait for

Fig 4-6: Open loop control

- Moreover, in future **HC – SR04 Ultrasonic Distance Sensors** can also be used to get data about position of the Robot and obstacle avoiding.
- Finally, to program this, Arduino software can either be interfaced with Simulink or be directly programmed. To directly program the Arduino, a simple time loop for the time period of 1 complete cycle of robot's leg can be run where the *millis()* function is used to generate and then feed time values, in seconds, to the theta functions (q1, q2, q3, q4, q5). These theta functions then return value of angles at any time t, which is then written as an argument in ***servo. Write()*** function. However, using **Simulink** is more convenient where it directly imports values of theta from MATLAB.
- User Input will be defined by using wireless control via Bluetooth.

4.2.3 User Interface

- An android mobile application is developed to receive the user's input and control the movement of the robot accordingly.
- The idea is to connect the application with the Arduino (having Bluetooth module) and send a value of '1' in order to move the robot and a value of '0' to stop the robot.
- As the Arduino will be connected to Simulink, it will transfer the input data to Simulink.

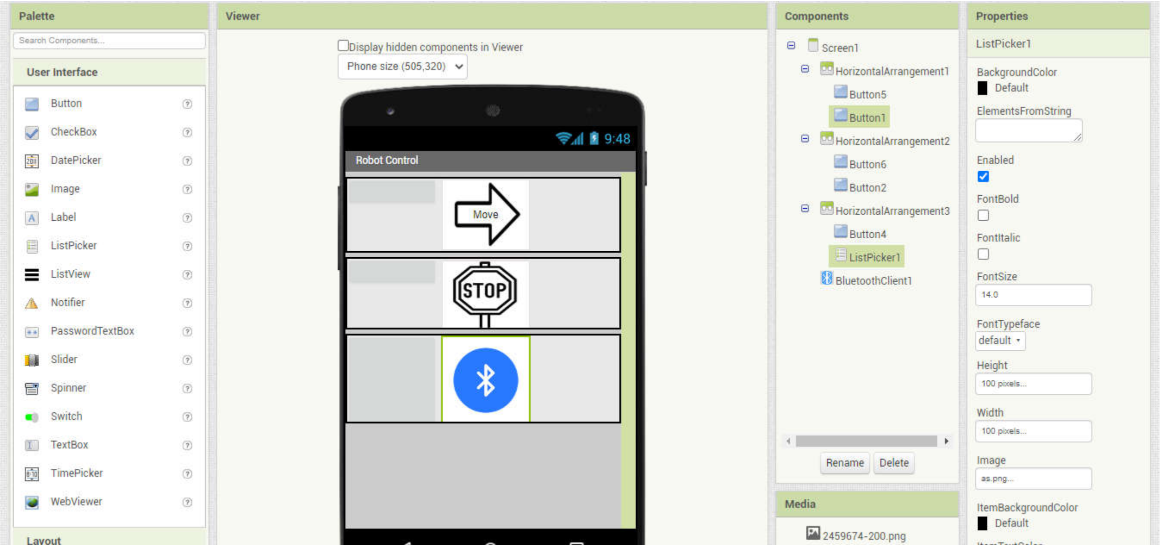
The first step to create an app was to design an interactive interface to control the robot's movement by the app.

- Initially we had to set the properties of the background screen of the application. This was done by the properties, options provided on the right side of the website.
- The background color of the app was selected to be light grey and the orientation of the application interface was selected to be portrait. Then the name of the app and the title of the background was set to be "**Robot Control**".
- Now, components were to be added to the application interface by the palette list on the left of the website.

1. Firstly, the layout was selected as horizontal arrangement and its height and width values were chosen from the properties. This was done to insert buttons to our application.
2. So, after choosing layout we added a button on the interface and adjusted its dimensions to match the ones of the layout.
3. Now the name of the button was entered in an appropriate font size and for user's ease an icon for the button was inserted which describes its functionality.

- Two buttons were added, one to move robot and the other one to stop it, step 1 to 3 were repeated for the other button as well.
- As the app will connect via Bluetooth to send the signal when a button is pressed from the application, a **Bluetooth client** was added from the **connectivity palette**.

- Afterwards a list picker was inserted from the **user interface palette** which will show the list of the discoverable Bluetooth devices.
- Its dimensions were adjusted, and a Bluetooth icon was added to it.



ListPicker1

BackgroundColor
■ Default

ElementsFromString

Enabled
☑

FontBold
□

FontItalic
□

FontSize
14.0

FontTypeface
default

Height
100 pixels...

Width
100 pixels...

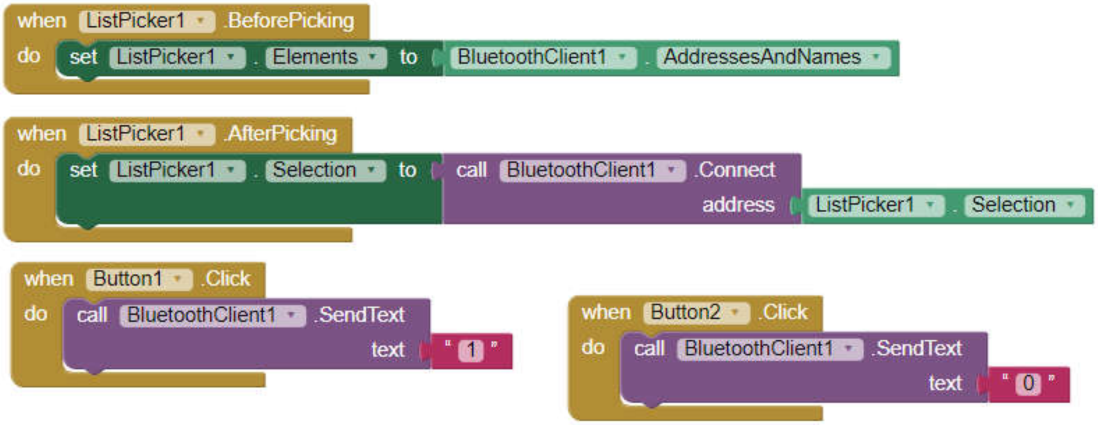
Image
as.png...

ItemBackgroundColor
■ Default

Fig 4-7: Mobile App developed in
MIT App Inventor

The second step was to create logics to enable function of the buttons on the interface. This was done by using Blocks in the MIT App Inventor.

- When the move button is clicked it will call the **send text** function of the Bluetooth client and a value is passed to the Arduino so the coding in there calls all the functions necessary for the movement of the robot.
- Another value is passed to Arduino when the stop button is clicked, this value triggers all the functions of the code which are used to stop the robot.
- Now when the Bluetooth button is clicked, it shows the list of all the nearby discoverable devices to choose from, as the blocks are connected to form the logic for calling the function of addresses and names of the **Bluetooth client**.
- Once a device id is chosen to connect to, the **connect address** function of the Bluetooth client is called.



```
when ListPicker1 .BeforePicking
do  set ListPicker1 . Elements to BluetoothClient1 . AddressesAndNames

when ListPicker1 .AfterPicking
do  set ListPicker1 . Selection to call BluetoothClient1 .Connect
                                                        address ListPicker1 . Selection

when Button1 .Click
do  call BluetoothClient1 .SendText
                          text " 1 "

when Button2 .Click
do  call BluetoothClient1 .SendText
                          text " 0 "
```

Fig 4-8: Final arrangement of the blocks

4.2.4 Simulink and Arduino

- In the first step the data in the form of 0 or 1 is sent from the Mobile Application depending on the user command to move or stop the robot to the Arduino via Bluetooth
- Then, the Arduino (which is linked to Simulink) communicates this data to the Simulink model.
- This data is received in Simulink with the **Serial Receiver** (as shown in the diagram below) and then interpreted in an **IF/ELSE Block.**
- Finally, depending on the user input, it will either move the robot (the data will bring the value of '1') or halt its motion (the data will bring the value of '0') if it was already moving.

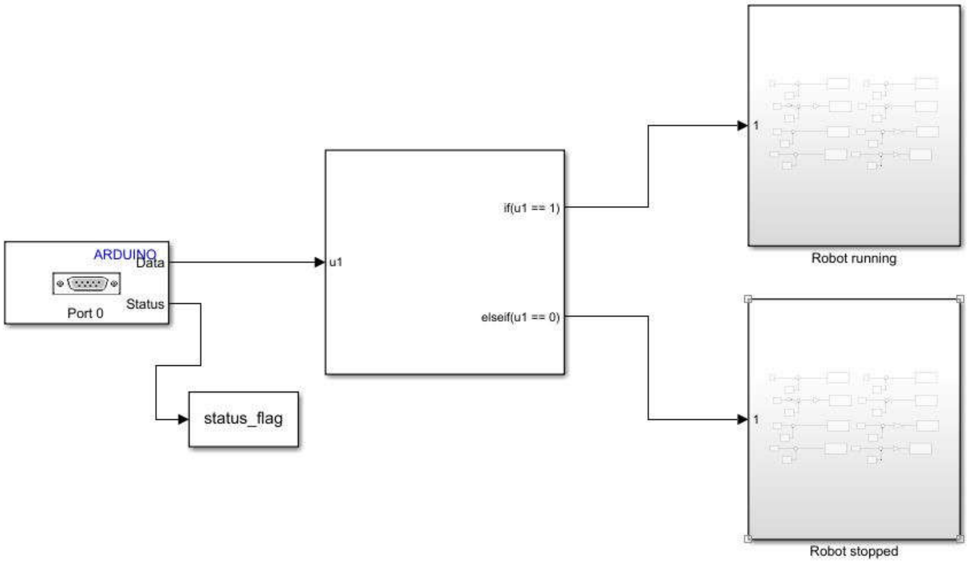


Fig 4-9: Simulink Model linked to the Arduino.

4.3 Testing/Experimental Procedures

This section should include details regarding testing and experimental procedure on the developed model.

Testing – Phase 1:

The first phase of testing included checking the response of a single motor to ensure that the motor is moving smoothly between the specified angles. At this the testing of the distance sensors that were to be installed also took place. The HC – SR04 Distance Sensor was working perfectly after wiring with the appropriate combination of resistors. Unfortunately, there was considerable jittering in the motion of servo motors. Hence, we tried performing the same movements using Arduino to see if the problem was specific to the microcontroller we were using. Fortunately, there was no jittering when the motors were run using Arduino.

Testing – Phase 2:

The second phase of tests included running of all the motors simultaneously to check if there are any problems with running all the motors simultaneously. This test was successful as the motors were providing the necessary torque, but the wires were heating up due to high amount of current being drawn. Hence, the wires were replaced with better quality wires.

Testing – Phase 3:

The third and final phase of testing includes the running of Theta functions (q3, q4, q5) which are the joint trajectories and see if the manipulator is walking as desired. For this purpose Simulink was used to run the program directly from MATLAB. During the testing of robot due to jittering of motors, lack of available power, and low hip rigidity the gait was not stable as we were expecting it to be; it required an external support to walk properly. Potential causes could be the small battery size and structural looseness at some points in torso. Unfortunately, due to the COVID-19 pandemic necessary measures could not be taken to resolve these issues

4.4 Summary

This chapter discusses in the detail the fabrication procedure, controls, and instrumentation of the robot. The testing phases for the robot are also included.

Chapter 5

Results and Discussions

5.1 Results

The mathematical model, inverse kinematics and the dynamics developed in the chapter 3 were solved for the following parameters of the robot:

Parameter	Symbol	Magnitude	Unit
Torso Mass	M (m_1+m_2+m_3)	3.8820	kg
Upper leg mass (Swing leg)	m_4	0.3020	kg
Lower leg mass (Swing leg)	m_5	0.5440	kg
Foot mass (Swing leg)	m_6	0.036	kg
Step Width	SW	5	cm
Step Height	SH	2	cm
Step Time	T_1	1	sec
Upper leg length (Swing leg)	L_4	22	cm
Lower leg length (Swing leg)	L_5	18	cm
Foot length (Swing leg)	L_6	4.5	cm
Torso Height	y_H	35	cm
Inertia matrix for Link 1	$I1$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	kgm^2
Inertia matrix for Link 2	$I2$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	kgm^2
Inertia matrix for Link 3	$I3$	$10^{-9}\begin{bmatrix} 358897 & -0.38 & 0.26 \\ -0.38 & 133672 & -162883 \\ 0.26 & -162883 & 273096 \end{bmatrix}$	kgm^2
Inertia matrix for Link 4	$I4$	$10^{-9}\begin{bmatrix} 271290 & 0.40 & 0.16 \\ 0.40 & 49662 & 89240 \\ 0.16 & 89240 & 248535 \end{bmatrix}$	kgm^2
Inertia matrix for Link 5	$I5$	$10^{-9}\begin{bmatrix} 19353 & 74 & 2.5 \\ 74 & 29106 & 84 \\ 2.5 & 84 & 22353 \end{bmatrix}$	kgm^2

5.1.1 Foot Trajectory

The foot trajectory defined in the mathematical modelling section and was implemented on MATLAB for the following parameters: $SW = 5\,cm, SH = 2\,cm, T1 = 1\,sec$. The trajectory is shown below in figure 5-1.

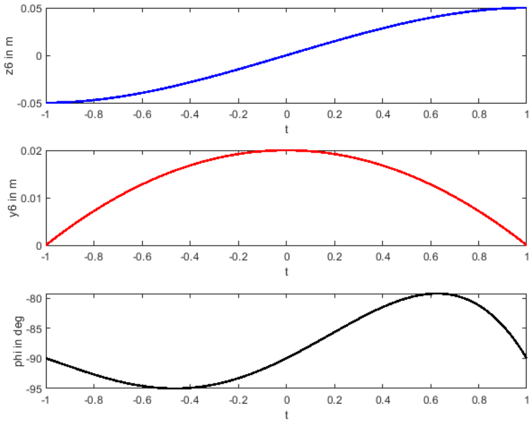


Fig 5-1: Foot trajectories

5.1.2 Torso Trajectory for TMIPM Model

The torso trajectory obtained in the mathematical modelling section was obtained using MATLAB for the following parameters: $SW = 5\,cm, SH = 2\,cm, T1 = 1\,sec$. The torso trajectory for the TMIPM is shown below in figure 5-2.

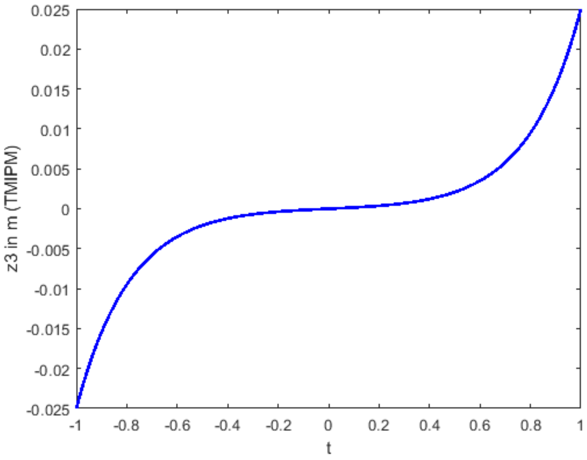


Fig 5-2: Torso trajectory for TMIPM

5.1.3 Inverse Kinematics

Once the stable torso trajectory is obtained using TMIPM, inverse kinematics is applied in order to solve for the final joint trajectory functions which are $q_1(t) to\ q_5(t)$. These joint trajectories are then plotted for $t = -T1\ to\ t = T1$, where T1 is half of the step time. Changing $T1$ will affect the average velocity of the robot as it moves. It can be seen in Figure

5-4 that $q_2(t)$ and $q_3(t)$ remain zero throughout the step period. This is because 3D motion is not yet introduced for the robot. As $q_2(t)$ and $q_3(t)$ represent the yaw and roll rotation of the hip joint, so any rotation along these axes will lead to a motion in x direction. The input to the inverse kinematics solver is shown in figure 5-3 and the joint trajectories in figure 5-4

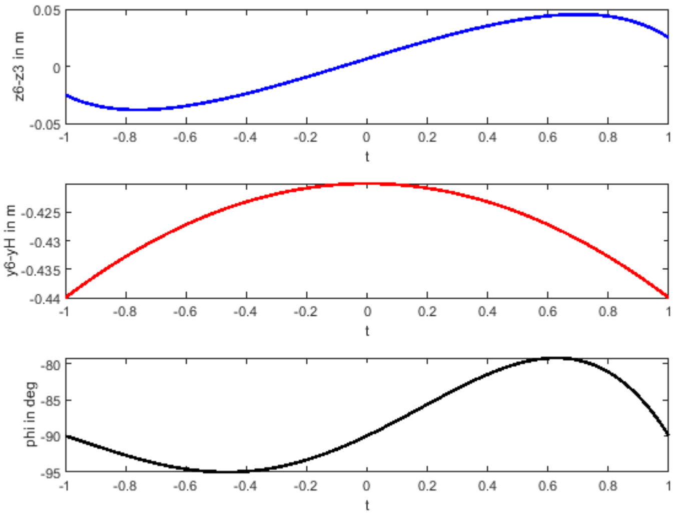


Fig 5-3: Input to the Inverse Kinematics Solver

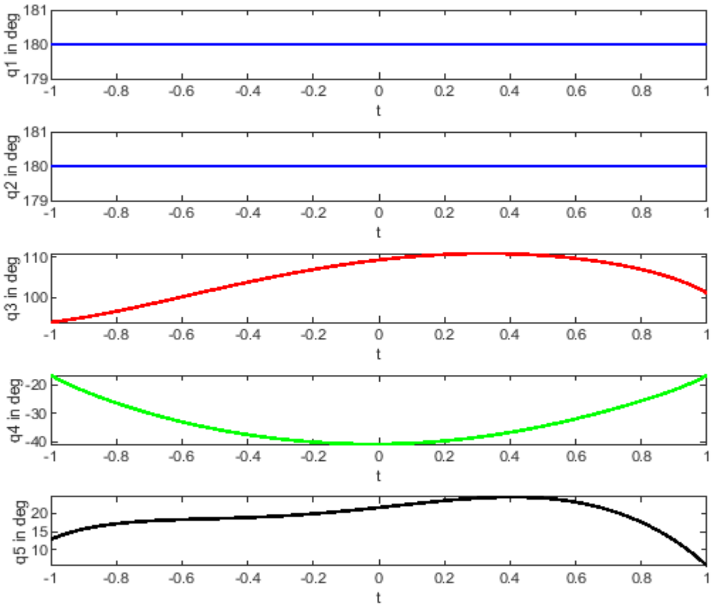


Fig 5-4: Output of the Inverse Kinematics Solver

5.1.4 Forward Kinematics and CG (centre of gravity) Trajectories for swing leg

After the inverse kinematics is solved, the position, velocity, and acceleration of any point on the robot can be obtained using forward kinematics. The forward kinematics is solved in MATLAB using $fkine()$ function. The solution yields cg trajectories including $(z_4(t), y_4(t), 0)$, $(z_5(t), y_5(t), 0)$ and $(z_6(t), y_6(t), 0)$. Where these represent the trajectories of cg of upper leg, lower leg and foot respectively. No anomaly was observed in these trajectories. All are smooth polynomials of degree 9 which means the velocity and acceleration functions are also smooth. Therefore, the torques required for the joint motors should also be smooth The CG trajectories are shown in figure 5-5.

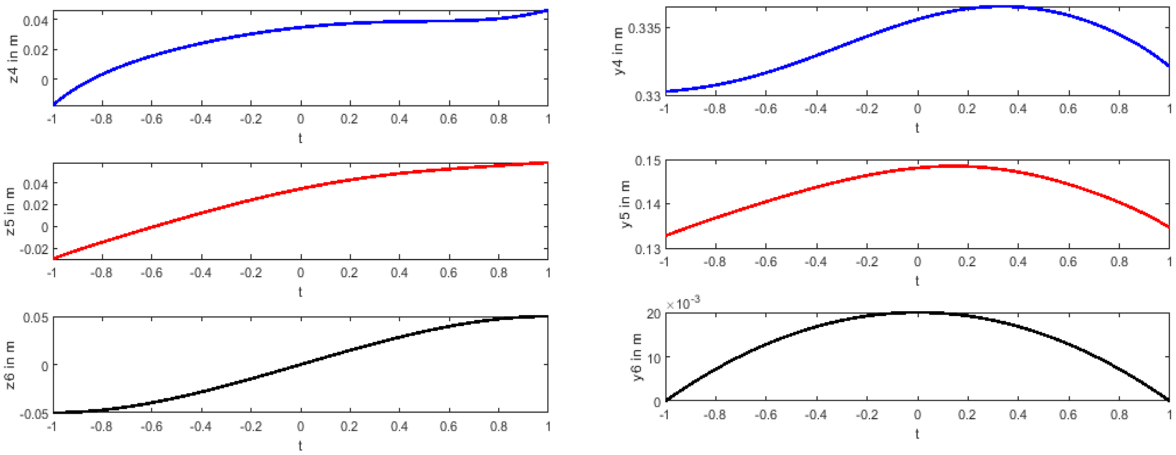


Fig 5-5: CG trajectories for swing leg

5.1.5 Torso Trajectory for MMIPM

After getting the cg trajectories for the masses attached to the swing leg, trajectory for the MMIPM model can be calculated using MATLAB by implementing the ODE given in the mathematical modelling section. The torso trajectory is shown in figure 5-6. The trajectory is tilted downwards for $t > 0$ which is due to the dynamic effect of the swing leg on the torso.

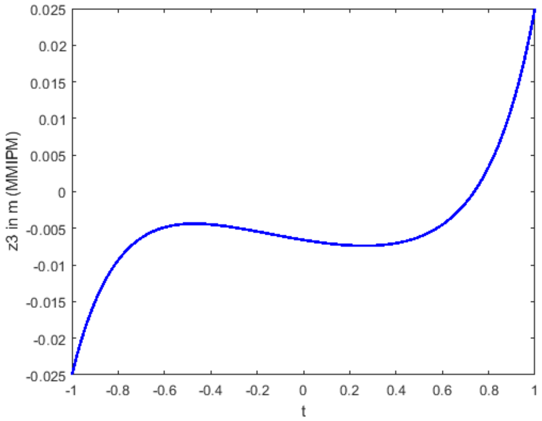


Fig 5-6: Torso Trajectory for MMIPM model

5.1.6 Joint Torques

From the Simulink simulation and MATLAB results, the following results were obtained for the parameters that are defined at the start of chapter 5.

$$\tau 1 = Hip\ Roll, \tau 2 = Hip\ yaw, \tau 3 = Hip\ pitch, \tau 4 = Knee\ Pitch\ , \tau 5 = \ Ankle\ Pitch$$

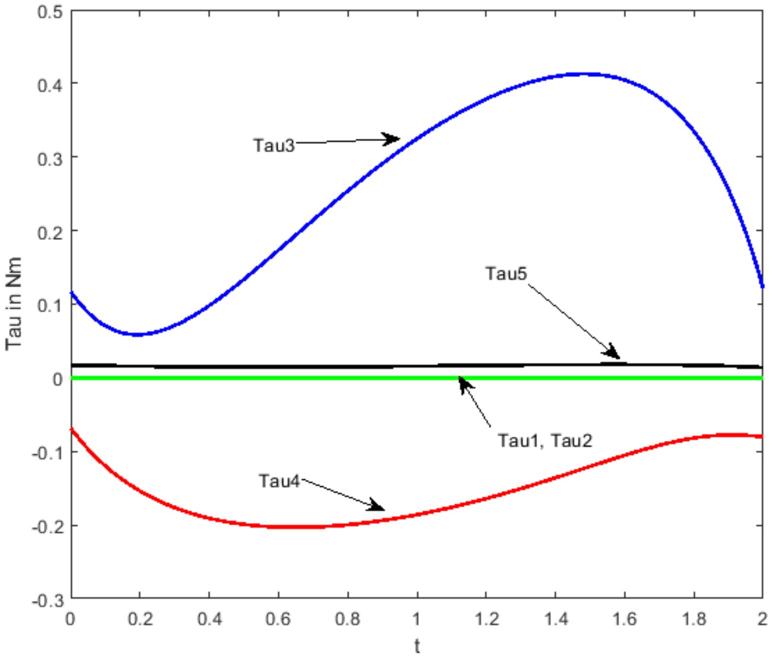


Fig 5-7: Joint Torques from Simulation Results

5.1.7 ZMP Trajectory

The ZMP trajectory was obtained by solving the equations given by [1]. The equations were modified according to the robot world frame. ZMPZ is represented by $p_z(t)$ and ZMPX by $p_x(t)$. These equations are:

$$p_z(t) = \frac{(\sum_i m_i z_i(\ddot{y}_i + g) - \sum_i m_i y_i \ddot{z}_i\ )}{\sum_i m_i z_i(\ddot{y}_i + g)} \tag{5.1}$$

$$p_x(t) = \frac{(\sum_i m_i x_i(\ddot{y}_i + g) - \sum_i m_i y_i \ddot{x}_i\ )}{\sum_i m_i z_i(\ddot{y}_i + g)} \tag{5.2}$$

Solving these for the parameters defined at the beginning of the results section, we get the following results (Figure 5-8):

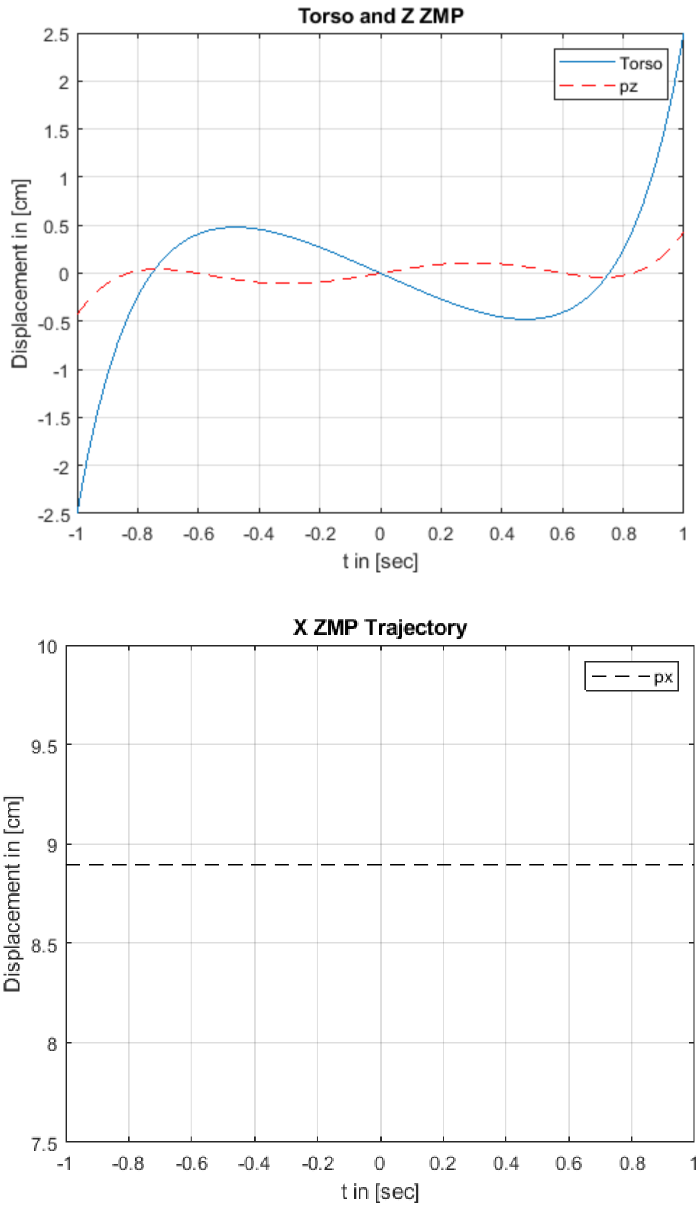


Fig 5-8: ZMP Trajectory

5.2 Discussions

5.2.1 Discussions on the Joint/cg Trajectories:

The trajectories obtained are smooth functions with only discontinuities at the end of the gait. These discontinuities may occur due to the gait change where both legs interchange after a short period of double support phase. It was observed that some of the parameters should not be exceeded from their maximum limits otherwise joint limits will be violated and the inverse kinematic solver will return an error. In general, the following maximum limits were obtained by changing one parameter and keeping other fixed.

Parameter	Maximum Limit	Fixed Parameters
Torso Height (y_H)	44 cm	SW= 5cm, SH= 2 cm
Step Width (SW)	11 cm	y_H= 44 cm, SH= 2 cm
Step Height (SH)	35 cm	y_H= 44 cm, SW= 5 cm

5.2.2 Discussions on the Joint Torques:

The robot was tested successfully in the Simulink and it was observed that the joint torques increase when the torso height is reduced or when the step height is increased. In general, the following trends are observed in the torques.

Change of Parameter	Effect on Joint Torques
Increase in the torso height	Decreases the joint torques
Increase in the step width	Increases the joint torques
Increase in the step height	Increases the joint toques
Increase in the average velocity of torso	Increases the joint torques

Therefore, the best results are obtained if the torso height is kept at maximum without violating the joint limits and by keeping the step width and step height as small as possible. However, decreasing the step width and height will also slow down the robot. Therefore, a trade-off should be established between the torso speed and the torque requirements. The motors installed in the Bart-UH robot have a maximum capacity of 20 kg.cm, thus any value above this will be undesirable. The maximum torque obtained out of these results is around 0.4 Nm (4 kg.cm), which is well below the maximum limit of the actuator.

5.2.3 Discussions on the ZMP Trajectory:

It was observed that the location of the ZMP changes if the parameters are changed. The results were limited to the ZMP_Z (Horizontal ZMP) only. As the maximum ZMP_X location is 9 cm, a foot width (in x direction) of more than 9 cm will ensure that the ZMP_X lies within the supporting area of the of foot. For the, ZMP_Z in general, the following trends were observed by changing various parameters. The ZMP_Z location increases with an increase in the step height and approaches a maximum limit of 1 cm for a step height of 35 cm, step width of 5 cm and torso height of 44 cm. This is shown in figure 5-9. Similarly, the trend of ZMP_Z location vs the step width is linear. A maximum of 1.5 cm occurs for a maximum allowed step width of 11 cm by keeping the step height of 2 cm and torso height of 44 cm. This is shown in figure 5-10. The effect of torso height on the ZMP_Z location is approximately inverse square one. Figure 5-11 shows that the best results are obtained by keeping the torso height maximum. A ZMP_Z location of about 0.93 cm is obtained for a torso height of 44 cm.

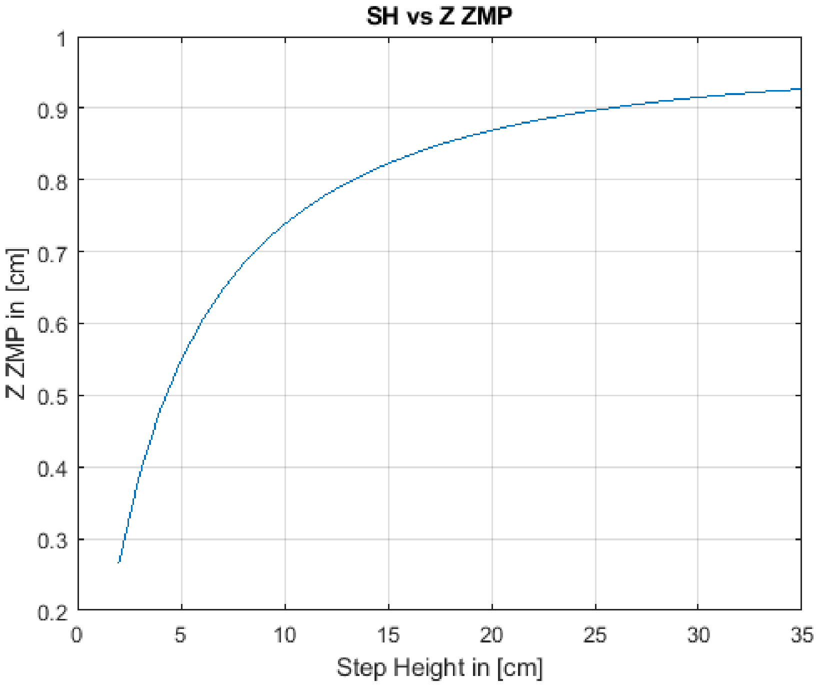


Fig 5-9: ZMP_Z location vs Step height

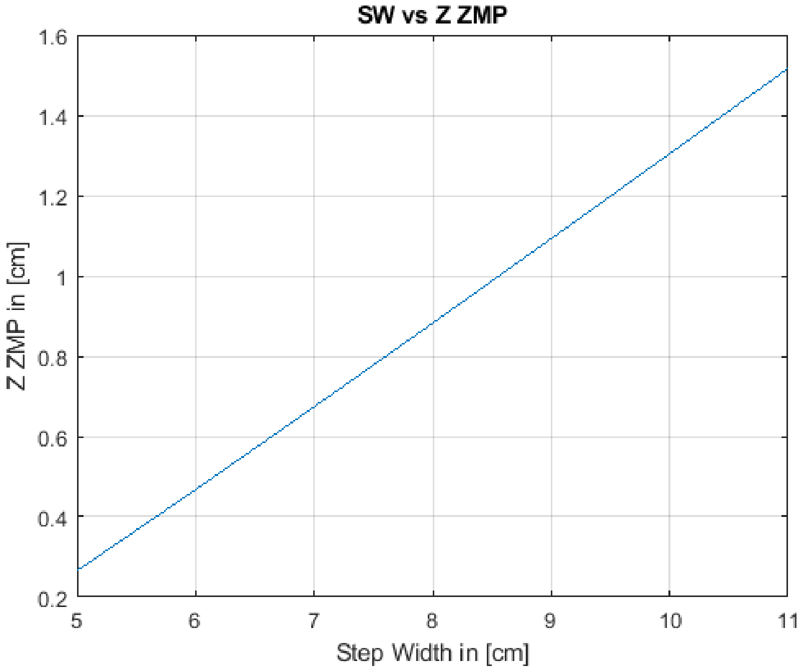


Fig 5-10: ZMP_Z location vs Step Width

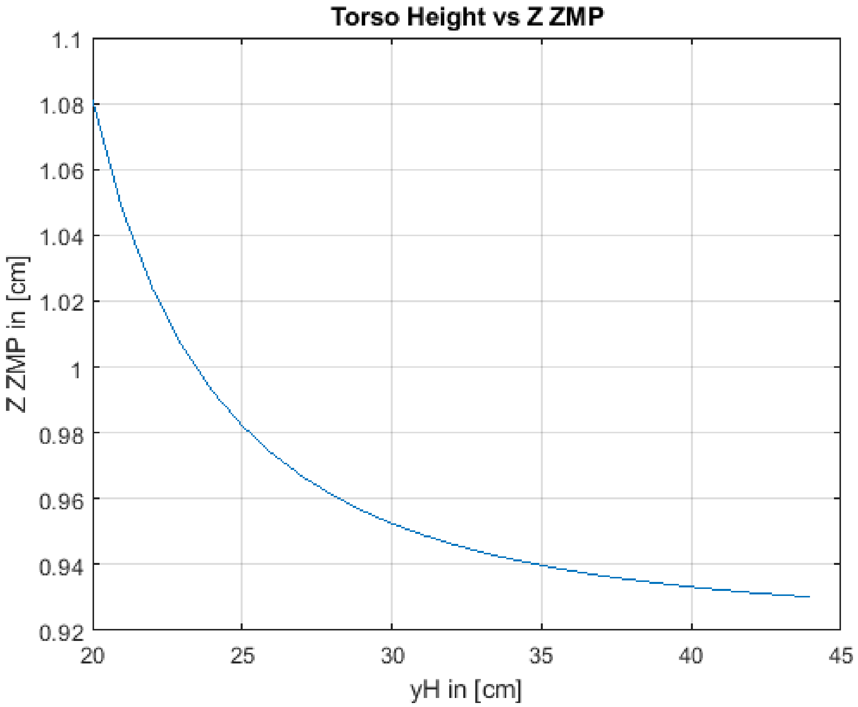


Fig 5-11: ZMP$_Z$ location vs Torso Height

The foot length in the z direction is 10 cm. Which means that the ZMP$_Z$ should be bounded by the range -5 cm to 5 cm. From the analysis, the ZMP$_Z$ is always bounded by 5 cm, therefore the robot should theoretically be stable as per the stability criteria. The maximum theoretical value of the ZMP$_Z$ is 1.5 cm as observed for a step width of 11 cm (Figure 5-10) which is well below the critical limits. It should however be noted that this ZMP$_Z$ analysis is theoretical and does not represent the real case scenario. For measuring the real time location of ZMP$_Z$, multiple sensors can be employed which include the pressure sensors, gyro/accelerometer and rotary encoders. The installation of pressure sensors was out of the scope of the project and the other two could not be implemented due to the lockdown situation that the world is facing. Therefore, the real time ZMP trajectory was not verified and remains a future endeavour for the project!

5.3 Summary

This chapter presents the complete results including graphs of different trajectories, inverse kinematics, forward kinematics and joint torques. The results are then discussed, individually.

Chapter 6

Impact and Economic Analysis

6.1 Sustainability Analysis

As the world of science progresses, there is a significant advancement in technological developments. Robotics being an emerging field, faces various social, technological and research challenges. Based on the specific attributes of humanoid bipedal robots, the sustainable adaptation of this new technology can be shaped by various kinds of perceived benefits mainly considering the economic factors of harsh environment performance and robot survivability. To address the sustainability factors, the evaluation strategy is constructed to specifically address the economic-related robot safety and re-usability factors in harsh and hazardous environments. In hazardous scenarios, the robot should be able to perform effectively while also minimising the energy consumption rate.

In a sustainable system, the battery consumption rate is a typical criterion, that is used to evaluate the cost of any mechanical or computational activity. An effective system should conserve as much energy as possible without comprising on the overall performance. In relation to this project, the amount of energy required to stabilize the robot and to enable its locomotion is evaluated. Furthermore, such robots when utilised in search and rescue operations, are to adapt to the unknown environment and follow a collision free path.

Adding to this, the probability of potential damage that may occur to the robot is a crucial factor, that is to be considered for developing a sustainable system. In harsh and hazardous environments and rugged terrains there are numerous unpredictable events and dangerous objects that can destroy the bipedal robot. For example, fire damage can be a potential damage source, because of the fire-related event frequency in real-world harsh environment scenarios. This factor can be addressed by using simple geometry—calculating the distance from the observation location to the detected hazardous object. From this distance, the radius of the hazardous obstacle effect zone will be subtracted to estimate the safe area zone. Furthermore, the probability of colliding with any unseen obstacle is also high in harsh environments. For this purpose, a constant tracking of the distance to the nearest unseen obstacle, is required.

The figure on the next page summarized the sustainability analysis and shows the relationship between the factors and effects. With all these factors contributing towards a sustainable robot, the future usage of this technology will only be effective when the end users are satisfied. Therefore, the sustainability analysis is incomplete without considering the robot’s social and environmental impacts

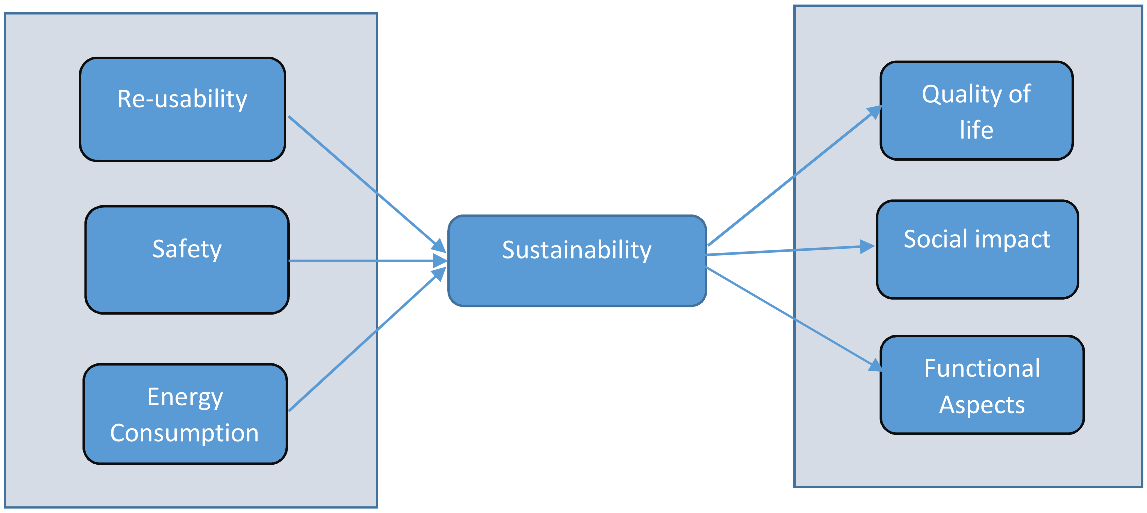


Fig 6-1: Sustainability Analysis

6.2 Social Impact

A humanoid robot is a robot that resembles human's physical attributes and has the capability to interact with humans and other robots, interpret information, and perform limited activities according to the user's input. Humanoid robots are equipped with sensors and actuators and are pre-programmed to perform specific activities. This project involves the control and programming of a bipedal robot that stabilizes itself and mimics human gait. Based on typical applications, such humanoid (bipedal) robots can be utilised in:

1. Service Robotics: Bipedal robots are better adapted to an environment that is destined for humans because their anthropomorphic aspect allows them to be accepted psychologically in a more spontaneous way. They can assist humans in daily tasks, for example, manoeuvring and distribution or in housework and hospital servicing work.
2. Dangerous terrains/ Hazardous areas: Bipedal robots have the mobile capacity to intervene in dangerous and hazardous environments, for example, in the case of catastrophes, such as earthquakes they can intervene to explore destroyed buildings and locate and save victims. Similarly, in the case of nuclear catastrophes, bipedal robots can be used to explore and transmit collected data from contaminated zones. This way, human exposure to dangerous radiations will be reduced.
3. Medical Prostheses: The use of bipedal robots for medical assistance is on the increase. Research in bipedal robots has resulted in the development of active or passive leg prostheses. Furthermore, bipedal robots can also be used to provide functional and re-educational assistance, to help strengthen weakened limbs or assist with training sessions.

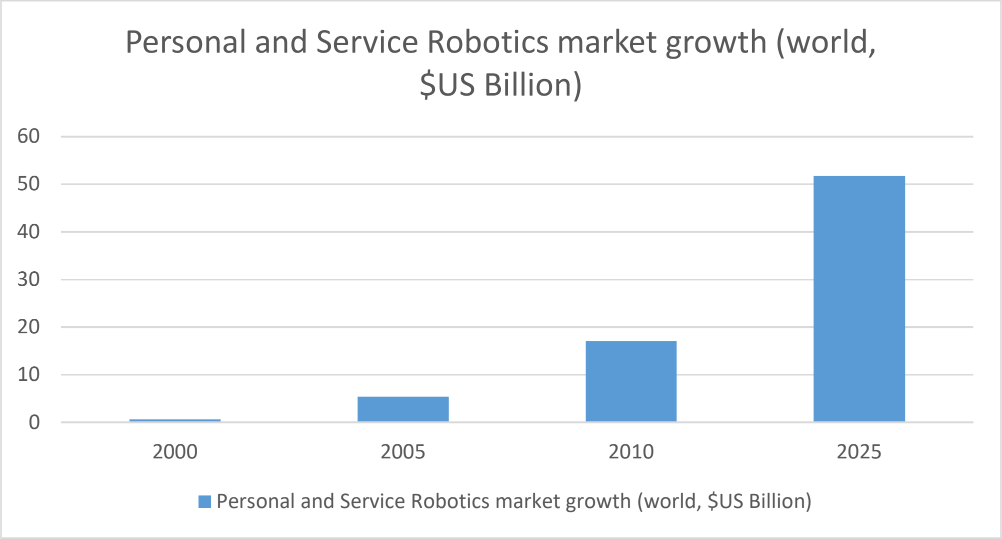


Fig 6-2: Economic Analysis

6.3 Environmental Impact

The material that was selected to manufacture this bipedal robot was Teflon (PTFE: Polytetrafluoroethylene). PTFE has numerous industrial applications due to its properties like low friction, durability, electrical and corrosion resistance. But, one of the main disadvantages of PTFE, that gives the material some limitation, is that, it overheats at temperatures above 570°F (300°C).

Teflon coatings start to breakdown and release toxic chemicals into the air. Inhaling these fumes can lead to polymer fume fever, commonly known as Teflon flu. Keeping this in view, in case of a hazard like a fire-related event, the robot might not be able to withstand the immense heat and will release dangerous chemicals in the environment.

Although, the primary focus of this project was not on fabrication but the methods that were used during the fabrication process, include CNC machining to manufacture the joints and links, finishing and polishing. These processes do not have any significant adverse or harmful effect on the environment and were carried out safely in the workshop.

6.4 Hazard Identification and Safety Measures

Bipedal Robots have better mobility than the conventional wheeled robots. Therefore, they are more practical to use in rugged terrains and harsh environments. But there are a few risks and hazards associated with the fabrication and electrical wiring of the bipedal robot, or any robot in general. These risks and hazards are classified as follows:

- Hazards related to electrical wiring and equipment such as motors and sensors: Faulty sensors may destroy the robot or harm the operator. Similarly, if the electrical wires are not properly insulated, an electric shock can occur.

- Hazards during fabrication processes: Any jagged edges or sharp pointed corners that stick out of the robot might cause a hazard. The material used for fabrication, TEFLON (PTFE) can breakdown and release toxic fumes if exposed to high temperatures.
- During the testing phase of the robot, any collision with an obstacle can seriously damage the robot depending on the proximity of the collision.
- Hazards caused by human error or foreseeable misuse.

To address these hazards, a safety strategy was devised based on the existing safety standards for industrial robotics. The strategy involves:

- Risk assessment of each hazard
- Identification of all the possible hazards
- Fulfilment of the safety measures
- Elimination of residual hazards (the probability of exposure to a potential hazard even after it has been identified and eliminated)
- Risk reduction
- User's guide to provide the user relevant information and safety measures

Based upon the safety strategy, the safety measures adopted were:

- Proper insulation of wires and use of new and efficient motors and sensors.
- Avoid exposure of the robot to high temperatures.
- Distance sensors to avoid obstacles
- Smooth and polished surface of the robot to avoid sharp edges

6.5 Summary

This chapter briefs the hazards and safety measures associated with the project and, also discusses the social impact, environmental impact and sustainability analysis.

Chapter 7

Conclusion and Future Recommendations

7.1 Conclusion

The mathematical model has been discussed in detail in chapter 3. Similarly, the virtual results obtained including the foot trajectory, torso trajectory based on the TMIPM model and MMIPM model, inverse kinematics, swing leg trajectory based on the forward kinematics, joint torques and ZMP trajectory are presented through graphs in Chapter 5. With these results obtained and with connection to the problem statement, the conclusion of this project is that the Zero Moment Point approach based on the TMIPM and MMIPM model is an adequate approach to solve the stability problem of a two legged robot.

The scope of this project included the mathematical modelling, CAD model design, fabrication, dynamics, SIMSCAPE virtual model, stability techniques and control of a bipedal robot. The project was destined to install the gyro/accelerometer or the rotary encoders to capture the ZMP data, but due to the unfortunate situation of Corona Pandemic, the team fell short of time and resources and therefore, the real time ZMP tracking could not be accomplished! Also, some other minor flaws were observed in the design of the robot after testing which lead to the robot not walking properly as expected. These included the jerky motion of the legs and overheating of motors. These issues can be rectified by using more rigid materials and high performance motors. Nevertheless, considering the overall scope, this project has almost achieved all the milestones within the timeline planned.

- Once the mathematical modelling was complete, it was concluded that the best design suited for this bipedal robot is BART-UH out of all the designs researched and discussed in the literature review.
- After the CAD model design, a research carried out on the material selection, concluded that TEFLON (PTFE) is the material with adequate strength, suitable for the fabrication of the robot.
- Finally, when the robot was completely manufactured, a research on controls and programming concluded that Arduino instead of Raspberry Pi is feasible to program and move the manipulator (details discussed in chapter 4)

Therefore, throughout this project, conclusions were derived based on comparison and with an aim to achieve maximum stability.

7.2 Future Recommendations

After analysing the results, and concluding the project, there were some deficiencies observed. Therefore, these are some of the future recommendations and endeavours:

- The principles of Kinematic redundancy can be applied to ensure a desired operational motion as well as the ZMP trajectory as recently done by [10].
- Installation of pressure sensors, gyro/accelerometer or the rotary encoders for real time ZMP tracking and validation of stability experimentally.
- Implementation of AI algorithms for path planning and avoiding object collision with the help of sophisticated sensors like the LIDAR.
- The vibrations of the robot can be minimized by improving the design and by using more rigid materials.
- High torque motors can be installed to increase the performance of the robot.

References

[1] A. AMOS and G. WILFRIED, “Analytic Path Planning Algorithms for Bipedal Robots without a Trunk,” *Journal of Intelligent and Robotic Systems,* vol. 36, p. 109–127, 2003.

[2] J. K. K. Y. Kevin G. Gim, “Design and Fabrication of a Bipedal Robot using Serial-Parallel Hybrid Leg Mechanism,” in *Intelligent Robots and Systems (IROS)*, 2018.

[3] K. G. Gim, J. Kim and K. Yamane, “Design and Fabrication of a bipedal robot using serial-parallel hybrid leg mechanism,” in *IEEE/RSJ International Conference of Intelligent Robots and Systems (IROS)*, Madrid,Spain, 2018.

[4] A. S. P. K. Ke Wang, “SLIDER: A Bipedal Robot with Knee-Less Legs and Vertical Hip Sliding Motion,” in *Robot Intelligence Lab*, London, 2018.

[5] K. Wang, A. Shah and P. Kormushev, “SLIDER:A Bipedal Robot with Knee-less legs and vertical hip sliding motion,” in *WPSC-Proceedings*, London,UK, 2018.

[6] A. R. a. O. S. University, “Cassie,” United States, 2016.

[7] S. Rogers, “This Bipedal Robot could be the next great delivery system,” interesting engineering, 10 February 2017. [Online]. Available: https://interestingengineering.com/bipedal-robot-cassie-next-great-delivery-system. [Accessed 06 June 2020].

[8] W. Schwanen, “Developing a kinematic and a dynamic model of a bipedal robot with twelve degrees of freedom,” Eindhoven University of Technology, 2002.

[9] Y. K. R. Jong Hyeon Park, “ZMP trajectory generation for reduced trunk motions of biped robots,” in *International Conference on Intelligent Robots and Systems*, Victoria, Can, 1998.

[10] S. Byung-Rok, R. HwanTaek and Y. Byung-Ju, “ZMP-based motion planning algorithm for kinematically redundant manipulator standing on the ground,” *Intel Serv Robotics,* vol. 8, p. 35–44, 2015.

[11] J. Ejdys and K. Halicka, “Sustainable Adaptation of New Technology_ The case of humanoids used for the care of older adults,” *MDPI,* 2018.

[12] C. Chevellereau, G. Bessonet and G. Abba, Bipedal Robots; Modeling, Designing and Building walking robots, London, UK; Hoboken, USA: ISTE Ltd. and John Wiley and Sons Inc., 2007.

[13] R. Bausys and F. Cavallro, “Application of Sustainability Principles for Harsh Environment Exploration by Autonomous Robot,” *MPDI,* 2019.

[14] M. Eleftheria and M. G. Spyridon, “Safety strategy for standardization of a personal AGV-Robot,” Democritus University of Thrace, Xanthi, Greece.

[15] S. Collins, A. Ruina and R. Tedrake, “Efficient Bipedal Robots Based on Passive Dynamic Walkers,” Science, 2005.

[16] W. Schwanen, “Developing a kinematic and a dynamic model of a bipedal robot with twelve degrees of freedom,” Eindhoven University of Technology , 2002.

[17] X. Bajrami, A. L. R. Dermaku and A. Kikaj, “Trajectory planning and inverse kinemetics solver for real bipad robot with 10 DOF,” in *International Federation of Automatic Control*, 2016.

[18] E. O. Jara, C. Cisneros, E. Alfaro and O. Jose, “Modelling and control of a bipad robot based on the capture of human movement patterns,” Peru, 2016.

[19] B.-R. So, H. Ryu and B.-J. Yi, “ZMP-based motion planning algorithm for kinematically redundant manipulator standing on the ground,” Springer-Verlag Berlin Heidelberg, Korea, 2014.

[20] A. Hereid, O. Harib, R. Hartley and Y. Gong, “Rapid Trajectory Optimization using C-FROST with illustration on a Cassi-Series dynamic walking bipad,” arXiv, USA, 2019.

[21] A. Senior and S. Tosunoglu, “Design of a bipad robot,” in *Florida Conference on Recent Advances in Robotics*, Miami, Florida, 2006.

[22] A. Nath, G. Das and A. Mallick, “Design implementation and stabilization of a bipedal robot,” in *AIP Conference Proceedings 1919*, Bangladesh, 2017.

Appendix A

MATLAB CODE

```
clc;
close all;
clear all;
format
short;

m0= 0.036;
m1= 0.544;
m2= 0.302;
m3= 3.0;   %Torso
m4= 0.302; %upper leg 0.302
m5= 0.544; %lower leg 0.544
m6= 0.036; %Foot      0.036
SW_z = 0.05; %5 cm
SH_y= 0.02;  %2cm
yH= 0.40;  %44cm max
T1 = 1;    %half step
time l0=0.045;
l1=0.180;
l2=0.220;
l4= 0.22; %upper leg 0.22
l5= 0.18; %lower leg 0.18
l6=0.045;  %Foot      0.045
g= 9.81;
lamda=
sqrt(g/yH);
res=50;
         %Resolut
ion y3=yH;

%define End effector
orientation t= linspace(-T1,
T1, 5);
qE= [degtorad(-90) degtorad(-95) degtorad(-90) degtorad(-80) degtorad(-
90)]; qE_c= polyfit(t, qE, 4);      %4th order fit

%Define Foot
Trajectory
t=linspace(-T1, T1,
res);
y6= SH_y*(1 - (t.^2)./(T1.^2));
z6= SW_z/2*((3.*t./T1) - (t.^3)./(T1.^3));


y6_c= polyfit(t, y6,
4); z6_c= polyfit(t,
z6, 4);

y6_dot_c =
polyder(y6_c);
z6_dot_c =
polyder(z6_c);

y6_2dot_c =
polyder(y6_dot_c);
z6_2dot_c =
polyder(z6_dot_c);

m3_t= m3+m2+m1;  %Rest of the mass of the robot
%Excitation function f(t) for TMIPM derived on the basis of ZMP Equation
m6_t= m4+m5+m6;
f_c= m6_t./(m3_t.*yH).*(g.*[0 0 0 0 z6_c] + conv(z6_c, [0 0 y6_2dot_c]) - conv(y6_c, [0
0]

z6_2dot_c]));

%solve the ODE using bvp4c to get torso trajectory

z3 t=linspace(-T1, T1, res);
solinit = bvpinit(t, @guess);
sol = bvp4c(@(t,y) [y(2), polyval(f_c,t) + lamda.^2.*y(1)], @(ya, yb) [ya(1) + SW_z/2,
yb
(1) - SW_z/2],
solinit); z3=
sol.y(1,:);

% Convert z3 into a
polynomial t=linspace(-T1,
T1, res); z3_c= polyfit(t,
z3, 5); figure;
plot(t, polyval(z3_c,t), 'color', 'b', 'LineWidth', 2);
xlabel('t');
ylabel('z3 in m (TMIPM)');


%% Inverse
Kinematics k=4;
while k>=1

j=1;
for t=linspace(-T1, T1, res)

%End effector location relative to Torso in World Frame
%px(j)= SW_x/2*((3.*t./T1) - (t.^3)./(T1.^3));
px(j)=0;
py(j)=polyval(y6_c,t)-yH;
pz(j)=polyval(z6_c,t)-
polyval(z3_c,t);
%pz(j)=0;
phi(j) = polyval(qE_c,t);

R= RY(-pi/2)*RZ(phi(j));
r11(j)= R(1,1); r12(j)= R(1,2); r13(j)= R(1,3);
r21(j)= R(2,1); r22(j)= R(2,2); r23(j)= R(2,3);
r31(j)= R(3,1); r32(j)= R(3,2); r33(j)= R(3,3);

%end Effector Tranf Matrix
```

```
T06=[r11(j) r12(j) r13(j)
px(j);
    r21(j) r22(j) r23(j) py(j);
    r31(j) r32(j) r33(j)
    pz(j); 0 0 0 1];

%Inverse Kinematics Solution
%q1(j) (Hip Roll)
q1(j)= atan2(r23(j),

r13(j)); if q1(j)>= pi



    q1(j)=0;
end

%q2(j) (Hip Yaw)
A1= [-1, -r33(j); 0, cos(q1(j)).*r13(j)+sin(q1(j)).*r23(j)];
A2= [cos(q1(j)).*r13(j)+sin(q1(j)).*r23(j), -1; r33(j), 0];
A= [cos(q1(j)).*r13(j)+sin(q1(j)).*r23(j), -r33(j); r33(j), cos(q1(j)).*r13(j)+sin(q1 ↙
(j)).*r23(j)];

%other solution for Yaw q2(j) (x motion)
%A1= [-1, -r32(j); 0, -pz(j)];
%A2= [cos(q1(j)).*r13(j)+sin(q1(j)).*r23(j), -1; cos(q1(j)).*px(j)+sin(q1(j).*py(j)),
0];
%A=[cos(q1(j)).*r13(j)+sin(q1(j)).*r23(j),-r32(j);                                      ⌙
cos(q1(j)).*px(j)+sin(q1(j)).*py(j),
-pz(j)];

c2= det(A1)./det(A);
s2= det(A2)./det(A);
q2(j)= atan2(s2,

c2); if q2(j)>=

pi

    q2(j)=0;
end

if q2(j)< 1*10^-
    4 q2(j)=0;
end

%calculate W
T02in= [cos(q1(j)).*sin(q2(j)), sin(q1(j)).*sin(q2(j)), cos(q2(j)), 0;
    cos(q1(j)).*cos(q2(j)), cos(q2(j)).*sin(q1(j)), -sin(q2(j)), 0;
    -sin(q1(j)), cos(q1(j)), 0, 0;
    0, 0, 0, 1];

W=
T02in*T06;
w11=
W(1,1);
w14= W(1,4);
w34= W(3,4);
```

```
w31= W(3,1);

%q4(j)  (Knee
Pitch) x_bar=
w14-l6.*w11;
y_bar= w34-
l6.*w31;
c4= (x_bar.*x_bar+y_bar.*y_bar-l4.*l4-l5*l5)./(2.*l4.*l5);
s4= -sqrt(1 - c4.*c4);   %2 solutions, (knee up and knee
down) q4(j)= atan2(s4, c4);   %This finds theta2

%q3(j) (Hip Pitch)
B1= [x_bar -l5.*s4; y_bar
l4+l5.*c4]; B2= [l4+l5.*c4 x_bar;
l5.*s4 y_bar];
B= [l4+l5.*c4 -l5.*s4; l5.*s4 l4+l5.*c4];

c3= det(B1)./det(B);
s3= det(B2)./det(B);
q3(j)= (atan2(s3, c3));   %theta 3

%q5(j) (Ankle Pitch)
q345(j)= atan2(w31,
w11);
q5(j)= (q345(j)- (q3(j)+q4(j)));    %theta 5

j=j+1;
end


%define New 5DOF DH Table
L(1)= Revolute('d',0,'a',0,'alpha',-pi/2,'offset',0);
L(2)= Revolute('d',0,'a',0,'alpha',pi/2,'offset',-pi/2);
L(3)=  Revolute('d',0,'a',l4,'alpha',0,'offset',0);
L(4)=  Revolute('d',0,'a',l5,'alpha',0,'offset',0);
L(5)= Revolute('d',0,'a',l6,'alpha',0,'offset',0);
bart = SerialLink(L);
%plot Bart
j=1;
for t=linspace(-T1, T1, res);

q= [q1(j) q2(j) q3(j) q4(j) q5(j)];
%bart.plot(q);
[T, ALL]= bart.fkine(q);  %Apply Forward Kinematics where ALL stores      ↙
individual transformation

for i=1:5
%Converting SE3 object to a 4x4 Matrix
temp1=  eye(4,3)*(tr2rt(ALL(i))*eye(3,4));
temp2=
eye(4,3)*(transpose(ALL(i).transl))*eye(1,4);
temp3= circshift(temp2, [0, 3]); %Right shift 3                                ↙
units
TT(:,:,i) =temp1+temp3+[0 0 0 0;0 0 0 0;0 0 0 0;0 0 0 1];   %Transforamtion Matrix
for Node 1 to N, 4x4 Order
end

%cg of m4 and m5 are assumed to be at the half lengths of links
```

```
r4prime(:,1,j)= transpose((transpose(TT(:,:,3)*[-l4/2;0;0;1]))*eye(4,3));
r5prime(:,1,j)=  transpose((transpose(TT(:,:,4)*[-l5/2;0;0;1]))*eye(4,3));     ↙
                                                                   %conta
ins x5prime, y5prime and z5prime 3x1 Mtx%contains x4prime, y4prime and z4prime
3x1 Mtx r6prime(:,1,j)= transpose((transpose(TT(:,:,5)*[0;0;0;1]))*eye(4,3));
j=j+1;
end



%Now defining everything in the global coordinate/World Frame
x4= r4prime(1, :);
y4= r4prime(2, :) +
yH; z4= r4prime(3,
:) + z3;



x5= r5prime(1, :);
y5= r5prime(2, :) +
yH; z5= r5prime(3,
:) + z3;

x6= r6prime(1, :);
y6= r6prime(2, :) +
yH; z6= r6prime(3,
:) + z3;


%Convert into
Polynomials
t=linspace(-T1, T1,
res); x4_c=
polyfit(t, x4, 5);
y4_c= polyfit(t, y4,
5); z4_c= polyfit(t,
z4, 5);

x5_c= polyfit(t, x5,
5); y5_c= polyfit(t,
y5,    5);     z5_c=
polyfit(t, z5, 5);

x6_c= polyfit(t, x6,
5); y6_c= polyfit(t,
y6,    5);     z6_c=
polyfit(t, z6, 5);

%Find the First oder
erivatives x4_dot_c=
polyder(x4_c); y4_dot_c=
polyder(y4_c); z4_dot_c=
polyder(z4_c);

x5_dot_c=
polyder(x5_c);
y5_dot_c=
polyder(y5_c);
```

```
z5_dot_c=
polyder(z5_c);

x6_dot_c=
polyder(x6_c);
y6_dot_c=
polyder(y6_c);
z6_dot_c=
polyder(z6_c);

%Find the 2nd oder
derivatives x4_2dot_c=
polyder(x4_dot_c);
y4_2dot_c=
polyder(y4_dot_c);
z4_2dot_c=
polyder(y4_dot_c);

x5_2dot_c=
polyder(x5_dot_c);
y5_2dot_c=
polyder(y5_dot_c);
z5_2dot_c=
polyder(y5_dot_c);

x6_2dot_c=
polyder(x6_dot_c);
y6_2dot_c=
polyder(y6_dot_c);
z6_2dot_c=
polyder(y6_dot_c);

%Now we need to apply the method of MMIPM and find the excitation function
z= [z4; z5; z6];
y= [y4; y5; y6];

z_dot= [polyval(z4_dot_c,t); polyval(z5_dot_c,t); polyval(z6_dot_c,t)];
y_dot= [polyval(y4_dot_c,t); polyval(y5_dot_c,t); polyval(y6_dot_c,t)];

z_2dot= [polyval(z4_2dot_c,t); polyval(z5_2dot_c,t);
polyval(z6_2dot_c,t)]; y_2dot= [polyval(y4_2dot_c,t);
polyval(y5_2dot_c,t); polyval(y6_2dot_c,t)];

m = [m4 m5 m6];

%%
%Calculating the Excitation funciton for MMIPM
fMM=0;
for i=1:3
   fMM= fMM+ (m(i)./(m3_t.*yH).*(g.*z(i,:) + z(i,:).*y_2dot(i,:) -
z_2dot(i,:).*y(i,:))); end

t=linspace(-T1, T1,
res); fMM_c=
polyfit(t, fMM, 5);

%Now, Solve the Diff Equ for the MMIPM Method using
bvp4c t=linspace(-T1, T1, res);
```

```
solinit = bvpinit(t, @guess);
sol = bvp4c(@(t,y) [y(2), polyval(fMM_c,t) + lamda.^2.*y(1)], @(ya, yb) [ya(1) +
SW_z/2, yb(1) - SW_z/2], solinit);
z3_new= sol.y(1,:);

%convert z3_new into a
polynomial t=linspace(-T1,
T1, res); z3_new_c=
polyfit(t, z3_new, 5);

%calculate error
error= max((z3_new-z3).^2)

%update the new value
of z3 z3=z3_new;
z3_c= z3_new_c;
z3_dot_c=
polyfit(t,z3,7);
z3_2dot_c=
polyder(z3_dot_c);

k=k-1;
end

index=j;

% figure;
% t=linspace(-T1, T1, res);
%

plot(t,x4)

; figure;



t=linspace(-T1, T1, res);
plot(t, polyval(z3_c, t),
'r'); xlabel('t');
ylabel('z3 in [m] MMIPM');

%%  Convert  q  into  a  polynomial  and  calculate
derivatives t=linspace(-T1, T1, res);
q1_c= polyfit(t, q1,
7); q2_c= polyfit(t,
q2,    7);     q3_c=
polyfit(t,   q3,  7);
q4_c= polyfit(t, q4,
7); q5_c= polyfit(t,
q5, 7);

v1_c=
polyder(q1_c);
v2_c=
polyder(q2_c);
v3_c=
polyder(q3_c);
v4_c=
polyder(q4_c);
v5_c=
polyder(q5_c);

a1_c=
polyder(v1_c);
a2_c=
polyder(v2_c);
a3_c=
polyder(v3_c);
a4_c=
polyder(v4_c);
a5_c=
polyder(v5_c);

%% find values of v and a now:
v1=  polyval(v1_c,
t);            v2=
polyval(v2_c,  t);
v3=  polyval(v3_c,
t);            v4=
polyval(v4_c,  t);
v5=  polyval(v5_c,
t);

a1=  polyval(a1_c,
t);            a2=
polyval(a2_c,  t);
a3=  polyval(a3_c,
t);            a4=
polyval(a4_c,  t);
a5=  polyval(a5_c,
t);


%% For complete robot walk. Left and right both
t=linspace(0, 4*T1, 2*res);
for j=1:index

q1(index-2+j)= q1(index-
1);       q2(index-2+j)=
q2(index-1);   q3(index-
2+j)=       q3(index-1);
q4(index-2+j)= q4(index-
1);       q5(index-2+j)=
q5(index-1);

v1(index-2+j)= v1(index-
1);       v2(index-2+j)=
v2(index-1);

v3(index-2+j)= v3(index-
1);       v4(index-2+j)=
v4(index-1);   v5(index-
2+j)=v5(index-1);

a1(index-2+j)= a1(index-
1);       a2(index-2+j)=
a2(index-1);   a3(index-
2+j)=       a3(index-1);
a4(index-2+j)= a4(index-
1);       a5(index-2+j)=
a5(index-1);

end

t=linspace(0, 4*T1,
2*res); for j=1:index-1
    q3R(j)=q3(2*(index-1));
    q4R(j)=q4(2*(index-1));
    q5R(j)=q5(2*(index-1));
end

for j=1:index-1
   q3R(index-1+j)=
   q3(j);   q4R(index-
   1+j)=        q4(j);
   q5R(index-1+j)=
   q5(j);

end


if
    ~exist('actuatorType','
    var') actuatorType=1

end

%% Foot
Dimension
foot_x=0.09;  %9
cm foot_y=0.05;
foot_z=0.01;

torso_x=0.
1;
torso_y=0.
1;
torso_z=0.1
5;


%% Inertia Matrices for each
link I1= zeros(3,3);
I2= zeros(3,3);
I3= 10^-9*[358897 -0.38 0.26;
    -0.38 133672 -162883;
    0.26 -162883 273096];
I4= 10^-9*[271290 0.40 0.16;
    0.40 49662 89240;
    0.16 89240 248535];
I5= 10^-9*[19353 74 2.5;
    74 29106 84;
    2.5 84 22353];

%cg poistions for each link 3x1
matrix r1=[0; 0; 0];
r2=[0; 0; 0];
r3=[-l4/2; 0; 0];
r4=[-l5/2; 0; 0];
```

```
r5=[0; 0; 0];

%gains and offsets for Servo
Control q2L_offset=0;
q3L_offset=
94;
q4L_offset=-
29;
q5L_offset=-
10;

q2R_offset
=0;
q3R_offset=
70;
q4R_offset=
-8;
q5R_offset
=1;

%gains
q4R_gain=
-1;


function g =
guess(t) g = [0
     0];
end

function output3=
RX(cc) output3= [1 0
0;
    0 cos(cc) -sin(cc);
    0 sin(cc) cos(cc)];
end

function output2= RY(bb)
output2= [cos(bb) 0
sin(bb);
    0 1 0;
    -sin(bb) 0 cos(bb)];
end

function output1= RZ(aa)
output1= [cos(aa) -
sin(aa) 0;
    sin(aa) cos(aa) 0;
    0 0 1];
end
```