HUMAN INSPIRED ROBOT WALKING PATTERN PLANNING

A Thesis
Submitted to the Faculty of Graduate Studies and Research
In Partial Fulfillment of the Requirements
For the Degree of

Master of Applied Science
in
Industrial Systems Engineering
University of Regina

By
Mohammadreza Ranjbar

Regina, Saskatchewan
March, 2016

Copyright 2016: M. Ranjbar
Mohammadreza Ranjbar, candidate for the degree of Master of Applied Science in Industrial Systems Engineering, has presented a thesis titled, *Human Inspired Robot Walking Pattern Planning*, in an oral examination held on December 9, 2015. The following committee members have found the thesis acceptable in form and content, and that the candidate demonstrated satisfactory knowledge of the subject material.

External Examiner: Dr. Luigi Benedicenti, Software Systems Engineering

Supervisor: Dr. Rene Mayorga, Industrial Systems Engineering

Committee Member: Dr. Amr Henni, Industrial Systems Engineering

Committee Member: Dr. Mohamed Ismail, Industrial Systems Engineering

Chair of Defense: Dr. David Gerhard, Department of Computer Science
ABSTRACT

Development of full size humanoid robots can be simplified along with the bipedal walking formulation methods. These robots consist of rigid bodies connected with actuated joints supposed to mimic human like walking. Although many researchers have tackled the problem of bipedal dynamic walking, it is far from matching human skills. This particular problem is of the special interest and the main subject of this research.

Biped robot walking is a periodic path of an unstable phase, called Single Support Phase, following a stable phase, Double Support Phase. A periodic approach providing a scalable gait with characteristic parameters such as gait length, gait maximum height and gait time cycle is proposed. The methodology is divided in two parts, planning robot trajectory and dynamic stability examination. The lower body is responsible for the general bipedal walking trajectory where limited numbers of breakpoints in both stable and unstable phase are identified. Consequently, positions of ankle, hip, and knee joints are derived for a seven link biped robot. In order to generate a smooth walking trajectory, a search for fast and efficient computation algorithms resulted in exploring the field of Artificial Intelligence and Soft Computing with the purpose of finding a valid non-conventional approach. The represented approach for walking pattern planning based on Artificial Neural Networks using Radial Basis Function is intended to fit a curve on derived breakpoints.

Biped robot stability during walking cycles is investigated using the Zero Moment Point criterion. In the dynamic stability study part, ZMP for a stable condition in a determined polygon of support is calculated in every single gait step. Then, for trunk motion adjustment in order to compensate for lower limb movement, Linear Inverted
Pendulum model and ZMP criterion are employed to obtain upper body trajectory satisfying whole robot walking dynamic stability.

**Keywords:** Artificial Neural Networks, Radial Basis Functions, Bipedal Robot Walking, Trajectory Planning, Zero Moment Point, Dynamic Stability
ACKNOWLEDGEMENTS

I would like to thank Dr. Rene Mayorga for his supervision and guidance throughout my studying at the University of Regina.
DEDICATION

To The Ones I Love
Table of Contents

ABSTRACT .................................................................................................................................................. I

ACKNOWLEDGEMENTS .................................................................................................................. III

DEDICATION ........................................................................................................................................ IV

LIST OF TABLES ...................................................................................................................................... VIII

LIST OF FIGURES .................................................................................................................................. IX

NOMENCLATURE ................................................................................................................................ XI

ABBREVIATIONS ................................................................................................................................ XIII

1. INTRODUCTION ................................................................................................................................. 1

1.1. MOTIVATION .................................................................................................................................. 2

1.2. GOALS AND CONTRIBUTIONS ................................................................................................. 2

1.3. OUTLINE OF DISSERTATION .................................................................................................... 3

2. TERMINOLOGY ................................................................................................................................. 4

2.1. ROBOTS AND ROBOTICS ......................................................................................................... 4

2.2. SIMULATION .............................................................................................................................. 5

2.2.1. World Coordinates ............................................................................................................... 5

2.2.2. Motion Planes ....................................................................................................................... 6

2.3. TRAJECTORY GENERATION .................................................................................................... 7

2.4. KINEMATICS ............................................................................................................................... 7

2.4.1. Forward Kinematics ........................................................................................................... 8

2.4.1.1. Denavit-Hartenberg method ......................................................................................... 8

2.4.2. Inverse Kinematics ............................................................................................................ 9

2.4.2.1. Singularity .................................................................................................................... 10

2.5. DYNAMICS .................................................................................................................................. 10

2.6. STABILITY .................................................................................................................................... 10

2.6.1. Stability Margins .................................................................................................................... 11

2.6.2. Static Stability ....................................................................................................................... 11

2.6.2.1. Center of Mass ............................................................................................................. 12

2.6.3. Dynamic Stability ................................................................................................................. 12

2.6.3.1. Zero Moment Point ...................................................................................................... 12

2.6.3.1.1. Huang ZMP Equations ....................................................................................... 13

2.6.3.1.2. Cart-Table Model .............................................................................................. 13

2.7. DEGREE OF FREEDOM ............................................................................................................ 15

2.7.1. Redundancy ........................................................................................................................... 15

2.8. GAIT ............................................................................................................................................... 16

2.8.1. Gait Phases ........................................................................................................................... 16

2.8.1.1. Double Support Phase .............................................................................................. 16

2.8.1.2. Single Support Phase ............................................................................................... 16

2.8.2. Swing Leg and Stance Leg ................................................................................................... 17

V
List of Tables

TABLE 6.1 LINKS CONCENTRATED MASSES ................................................................. 73
TABLE 6.2 CONSTANT PARAMETERS ......................................................................... 74
TABLE 6.3 THE ROBOT’S GAIT SPECIFICATIONS ...................................................... 75
TABLE 6.4 RBFNs SPECIFICATIONS FOR FLAT SURFACE ....................................... 78
TABLE 6.5 THE ROBOT’S GAIT SPECIFICATIONS FOR NEGATIVELY SLOPED GROUND CONDITION .......................................................... 80
TABLE 6.6 RBFNs SPECIFICATIONS FOR DECLINED SURFACE ................................ 82
TABLE 6.7 THE ROBOT’S GAIT SPECIFICATIONS FOR PosITIVELY SLOPED GROUND CONDITION ........................................................... 84
TABLE 6.8 RBFNs SPECIFICATIONS FOR INCLINED SURFACE ................................ 85
TABLE 6.9 THE ROBOT’S GAIT SPECIFICATIONS FOR WALKING DOWN STAIRS CONDITION .............................................................. 87
TABLE 6.10 RBFNs SPECIFICATIONS FOR WALKING DOWN STAIRS .......................................................... 88
TABLE 6.11 THE ROBOT’S GAIT SPECIFICATIONS FOR WALKING UP STAIRS .............. 90
TABLE 6.12 RBFNs SPECIFICATIONS FOR WALKING UP STAIRS ........................................... 90
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 2.1</td>
<td>World Coordinates</td>
<td>6</td>
</tr>
<tr>
<td>Figure 2.2</td>
<td>Planes of Motions [6]</td>
<td>7</td>
</tr>
<tr>
<td>Figure 2.3</td>
<td>Forward Kinematics and Inverse Kinematics</td>
<td>8</td>
</tr>
<tr>
<td>Figure 2.4</td>
<td>Forward Dynamics vs. Inverse Dynamics</td>
<td>10</td>
</tr>
<tr>
<td>Figure 2.5</td>
<td>(a) Support Area in Double Support Phase (b) Support Area in Single Support Phase [10]</td>
<td>11</td>
</tr>
<tr>
<td>Figure 2.6</td>
<td>Cart-Table Model</td>
<td>14</td>
</tr>
<tr>
<td>Figure 2.7</td>
<td>Biped Robot Structure in Sagittal Plane</td>
<td>15</td>
</tr>
<tr>
<td>Figure 2.8</td>
<td>Single Support Phase</td>
<td>16</td>
</tr>
<tr>
<td>Figure 2.9</td>
<td>Double Support Phase Examples</td>
<td>17</td>
</tr>
<tr>
<td>Figure 4.1</td>
<td>A basic scheme of ANNs’ structure</td>
<td>26</td>
</tr>
<tr>
<td>Figure 4.2</td>
<td>A neuron with multiple inputs and single output</td>
<td>27</td>
</tr>
<tr>
<td>Figure 4.3</td>
<td>A multilayer feedforward network topology</td>
<td>30</td>
</tr>
<tr>
<td>Figure 4.4</td>
<td>A FeedForward Backpropagation Network. The activity from the inputs flows forward and error flows backward to adjust the weights</td>
<td>32</td>
</tr>
<tr>
<td>Figure 4.5</td>
<td>Radial Basis Function Network structure</td>
<td>36</td>
</tr>
<tr>
<td>Figure 5.1</td>
<td>Biped robot model in sagittal plane</td>
<td>42</td>
</tr>
<tr>
<td>Figure 5.2</td>
<td>DSP and SSP time duration</td>
<td>43</td>
</tr>
<tr>
<td>Figure 5.3</td>
<td>Five identified key points in a gait cycle</td>
<td>44</td>
</tr>
<tr>
<td>Figure 5.4</td>
<td>The robot’s foot specification</td>
<td>46</td>
</tr>
<tr>
<td>Figure 5.5</td>
<td>The swing foot leaving and landing angles</td>
<td>47</td>
</tr>
<tr>
<td>Figure 5.6</td>
<td>Position of swing ankle joint when it is at the highest point in XZ-plane</td>
<td>47</td>
</tr>
<tr>
<td>Figure 5.7</td>
<td>XED and XSD illustration</td>
<td>49</td>
</tr>
<tr>
<td>Figure 5.8</td>
<td>The swing knee joint trajectory by intersecting circles CIII and CAl</td>
<td>50</td>
</tr>
<tr>
<td>Figure 5.9</td>
<td>Characteristics of walking on stairs</td>
<td>56</td>
</tr>
<tr>
<td>Figure 5.10</td>
<td>Three characteristic points of foot sole</td>
<td>58</td>
</tr>
<tr>
<td>Figure 5.11</td>
<td>An Artificial Neural Network with input/output sets</td>
<td>61</td>
</tr>
<tr>
<td>Figure 5.12</td>
<td>The ANN initial structure for swing ankle joint interpolation</td>
<td>62</td>
</tr>
<tr>
<td>Figure 5.13</td>
<td>The multi-phase ANN architecture for the swing ankle joint motion planning</td>
<td>63</td>
</tr>
<tr>
<td>Figure 5.14</td>
<td>Two-phases ANN structure for real-time applications</td>
<td>65</td>
</tr>
<tr>
<td>Figure 5.15</td>
<td>The multi-phase ANN architecture for the hip joint motion planning</td>
<td>66</td>
</tr>
<tr>
<td>Figure 5.16</td>
<td>Modeling a biped robot with an inverted pendulum</td>
<td>68</td>
</tr>
<tr>
<td>Figure 5.17</td>
<td>The trunk trajectory planning algorithm</td>
<td>71</td>
</tr>
<tr>
<td>Figure 6.1</td>
<td>The biped robot walking trajectory on flat ground condition</td>
<td>75</td>
</tr>
<tr>
<td>Figure 6.2</td>
<td>Represents a closer view of robot walking during a single gait with applied dynamical considerations.</td>
<td>75</td>
</tr>
<tr>
<td>Figure 6.3</td>
<td>Ankle joint motion pattern</td>
<td>76</td>
</tr>
<tr>
<td>Figure 6.4</td>
<td>Sole of the swing foot angle</td>
<td>76</td>
</tr>
<tr>
<td>Figure 6.5</td>
<td>Hip joint trajectory in XZ-plane in one gait cycle</td>
<td>76</td>
</tr>
<tr>
<td>Figure 6.6</td>
<td>The swing leg knee joint in sagittal plane during a single gait period</td>
<td>77</td>
</tr>
<tr>
<td>Figure 6.7</td>
<td>The stance leg knee joint in sagittal plane during a single gait period</td>
<td>77</td>
</tr>
<tr>
<td>Figure 6.8</td>
<td>Support area in first gait cycle</td>
<td>77</td>
</tr>
<tr>
<td>Figure 6.9</td>
<td>The biped robot walking trajectory on flat ground condition obtained by spline interpolation</td>
<td>79</td>
</tr>
</tbody>
</table>
FIGURE 6.10 SOLE OF THE SWING FOOT ANGLE BY RBFN AND SPLINE ................................................................. 79
FIGURE 6.11 THE BIPED ROBOT WALKING TRAJECTORY ON DOWN HILL GROUND CONDITION ..................... 80
FIGURE 6.12 ANKLE JOINT MOTION PATTERN ON DECLINED GROUND CONDITION ........................................... 81
FIGURE 6.13 SOLE OF THE SWING FOOT ANGLE ...................................................................................................... 81
FIGURE 6.14 THE HIP JOINT IN SAGITTAL PLANE DURING A SINGLE GAIT PERIOD ............................................ 81
FIGURE 6.15 THE STANCE LEG KNEE JOINT PATTERN IN SAGITTAL PLANE DURING A SINGLE GAIT PERIOD .... 82
FIGURE 6.16 STICK DIAGRAM OF BIPED ROBOT WALKING TRAJECTORY ON DOWNHILL GROUND CONDITION
          OBTAINED BY SPLINE INTERPOLATION ............................................................................................................. 84
FIGURE 6.17 THE BIPED ROBOT WALKING TRAJECTORY ON UP HILL GROUND CONDITION ..................................... 84
FIGURE 6.18 STICK DIAGRAM OF WALKING UP HILL .................................................................................................. 84
FIGURE 6.19 THE BIPED ROBOT WALKING DOWN STAIRS TRAJECTORY ................................................................. 86
FIGURE 6.20 THE BIPED ROBOT WALKING DOWN STAIRS TRAJECTORY WITH A SMALLER STEP SIZE .......... 84
FIGURE 6.21 ANKLE MOTION PATTERN IN ONE GAIT CYCLE ..................................................................................... 87
FIGURE 6.22 HIP TRAJECTORY DURING A GAIT CYCLE ............................................................................................... 87
FIGURE 6.23 THE SOLE OF SWING FOOT MOTION PATTERN ................................................................................... 88
FIGURE 6.24 THE STICK DIAGRAM OF WALKING DOWN STAIRS ........................................................................... 89
FIGURE 6.25 THE BIPED ROBOT’S WALKING UP STAIRS TRAJECTORY ................................................................. 90
FIGURE 6.26 RBFN PERFORMANCE DIAGRAM ............................................................................................................ 91
### Nomenclature

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_{\text{COM}}$</td>
<td>Center of Mass position along X-axis</td>
</tr>
<tr>
<td>$m_i$</td>
<td>Mass of $i^{th}$ link</td>
</tr>
<tr>
<td>$x_i$</td>
<td>Center of Mass position of $i^{th}$ link</td>
</tr>
<tr>
<td>$\tau$</td>
<td>Torque</td>
</tr>
<tr>
<td>$\dot{\theta}_i$</td>
<td>Absolute angular velocity of $i^{th}$ component</td>
</tr>
<tr>
<td>$g$</td>
<td>Gravitational acceleration</td>
</tr>
<tr>
<td>$p$</td>
<td>ZMP position along the X-axis</td>
</tr>
<tr>
<td>$W_i$</td>
<td>Weight associated with $i^{th}$ input</td>
</tr>
<tr>
<td>$\eta$</td>
<td>Learning ratio</td>
</tr>
<tr>
<td>$d_k$</td>
<td>Error function</td>
</tr>
<tr>
<td>$x_h$</td>
<td>Activation signal of the $h^{th}$ hidden neuron</td>
</tr>
<tr>
<td>$w_{hk}$</td>
<td>Weight factor between the $h^{th}$ hidden neuron and the output node $k$</td>
</tr>
<tr>
<td>$\varphi$</td>
<td>Gaussian Radial Basis Function</td>
</tr>
<tr>
<td>$D_S$</td>
<td>Gait length</td>
</tr>
<tr>
<td>$T_c$</td>
<td>Gait cycle time or a gait period</td>
</tr>
<tr>
<td>$T_d$</td>
<td>Double support phase duration</td>
</tr>
<tr>
<td>$Xa_{SW}$</td>
<td>Swing leg ankle joint displacements along X-axis</td>
</tr>
<tr>
<td>$Za_{SW}$</td>
<td>Swing leg ankle joint displacements along Z-axis</td>
</tr>
<tr>
<td>$q_b$</td>
<td>Swing foot angle as it leaves the ground</td>
</tr>
<tr>
<td>$q_f$</td>
<td>Swing foot angle as it touches down the ground</td>
</tr>
<tr>
<td>Symbol</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>$x_{ed}$</td>
<td>Distance from hip joint to stance ankle joint at the initial points of the double support phase</td>
</tr>
<tr>
<td>$x_{sd}$</td>
<td>Distance from hip joint to stance ankle joint at the final points of the double support phase</td>
</tr>
<tr>
<td>$H_{max}$</td>
<td>Hip joint maximum height along Z-axis</td>
</tr>
<tr>
<td>$H_{min}$</td>
<td>Hip joint minimum height along Z-axis</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>Surface slope</td>
</tr>
<tr>
<td>$H_{Str}$</td>
<td>Stair height</td>
</tr>
</tbody>
</table>
### Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANN</td>
<td>Artificial Neural Network</td>
</tr>
<tr>
<td>RBFN</td>
<td>Radial Basis Function Network</td>
</tr>
<tr>
<td>COM</td>
<td>Center Of Mass</td>
</tr>
<tr>
<td>COG</td>
<td>Center Of Gravity</td>
</tr>
<tr>
<td>ZMP</td>
<td>Zero Moment Point</td>
</tr>
<tr>
<td>MSE</td>
<td>Mean Square Error</td>
</tr>
<tr>
<td>LMS</td>
<td>Least Mean Square</td>
</tr>
<tr>
<td>IK</td>
<td>Inverse Kinematic</td>
</tr>
<tr>
<td>FK</td>
<td>Forward Kinematic</td>
</tr>
<tr>
<td>ID</td>
<td>Inverse Dynamic</td>
</tr>
<tr>
<td>FD</td>
<td>Forward Dynamic</td>
</tr>
<tr>
<td>DOF</td>
<td>Degree Of Freedom</td>
</tr>
<tr>
<td>DSP</td>
<td>Double Support Phase</td>
</tr>
<tr>
<td>SSP</td>
<td>Single Support Phase</td>
</tr>
<tr>
<td>LIPM</td>
<td>Linear Inverted Pendulum Model</td>
</tr>
</tbody>
</table>
In recent years, dynamic bipedal walking and in general legged locomotion inspired growing interests. Partially, the reason for this interest is the demand for robots enabled to operate in human oriented environments. Biped robots perform in such environments more efficient than other types of robots yet devised. These robots are more mobile than the wheeled ones; for instance, where wheels require a continuous path of support, legged robots can traverse a discrete terrain. This flexibility in adopting configuration of the support base makes legged locomotion significantly adaptive. In addition, biped robot structure allows the body to travel along a different trajectory than the feet. In such a way, the whole body walking pattern might be smooth, despite the roughness of the environment.

Optimistically, bipedal robots eventually can be used to accomplish exhausting and dangerous tasks such as working in extreme environmental conditions, with hazardous chemicals or as an aide to humans in similar places of activity. Furthermore, the consequences of research in humanoid robots would be applied in enhancement of prosthetic devices.
1.1. Motivation

Locomotion is an essential task for any moving alive being. Among legs, wings, and fins, different methods of movement instances; human beings are adopting biped locomotion, an advantage of few species. Reasons to analyze and derive counterpart models of human being behavior in real life are persuasive, as the applications are widespread.

Legged robots coming to be progressively popular require an adaptive robust walking trajectory, as the first step of skillfully operation in various environments designed for human or its purposes. For example, NASA has developed many robots such as Robonaut [1], a humanoid robot, aimed to assist or substitute for astronauts in order to undertake difficult tasks such as equipment repairing in dangerous environments with radiation risks or doing routine procedures. Besides, there are a huge amount of potential applications in patients’ mobility, rehabilitation, and video games industries which could benefit from a reliable realistic humanoid robot walking method.

1.2. Goals and contributions

The starting point for biped robot locomotion is employment of an efficient method of trajectory planning. Ideally, it is featured with a simple implementation and fast calculation besides being compatible with real time applications. Moreover, the proposed trajectory implementation needs to accept user input parameters such as gait length, gait time period and maximum foot lift height to provide a method for manipulating the speed and position of the robot’s joints. In order to develop a non-conventional technique which is more efficient and computationally faster than the conventional methods, this research focuses on the field of Artificial Intelligence and Soft Computing. Numerical iterative
methods are usually applied locally to find a solution for functions; however, the amount of computation required in many cases is excessive for real-time implementations. Therefore, non-conventional techniques utilizing Artificial Neural Networks and/or Radial Basis Networks are implemented to shorten the computation time for online applications and ensure smoothness of the walking steps. Dynamic balance investigation of the robot is explored by the Zero Moment Point criterion. The stability methodology of the biped robot should derive trunk motion pattern such that it compensates for lower limb movement. Therefore, the physical issues of kinematic and dynamic balance in the articulated figure is treated under the proposed methodology.

1.3. Outline of dissertation

In chapter two, a comprehensive definition of the terminology used throughout the dissertation is given. In chapter three, a review of methodologies concerned with bipedal walking problem is presented. An introduction to Artificial Neural Network with an emphasis in its generalization ability is presented in chapter four. In chapter five an efficient approach utilizing RBFNs is proposed for robot bipedal walking pattern planning on various ground conditions. The performance of the suggested technique is demonstrated through multiple simulations in chapter six. Finally, a brief conclusion and several applications of this thesis are presented in chapter seven.
Chapter 2

TERMINOLOGY

As specialized knowledge emerges with increasing proficiency in the field of robotics, the technical terms and concepts are expanding. Besides, an appropriate terminology leads the communication to a right way. A list of terms relevant to this dissertation is described in this chapter. The definitions of these terms reflect their meaning in the field of robotics.

2.1. Robots and Robotics

The word robot comes from the Slavic word robota, which means labor. A robot is a mechanical constitution or an artificial agent enabled to sense its surrounding environment and employ acquired knowledge to follow specified instructions. A humanoid robot is an advanced mobile robot which resembles human beings.

The term robotics coined by Isaac Asimov refers to the study and use of robots [2]. Asimov also proposed "Laws of Robotics":

1. A robot may not injure humanity, or, through inaction, allow humanity to come to harm.

2. A robot may not injure a human being, or, through inaction, allow a human being to come to harm, unless this violates a higher order law.
3. A robot must obey orders given by human beings, except where such orders conflicts with a higher order law.

4. A robot must protect its own existence as long as such protection does not conflict with a higher order law.

2.2. Simulation

Simulation provides a powerful visualization, planning, and strategic tool in different areas of research and development. In robotics, simulation is utilized for the kinematics and dynamics analysis of robots, off-line programming, designing control algorithms, and designing mechanical structures [3]. Webots™ [4] and SimSpark [5] are examples of widely used robotic simulation software.

2.2.1. World Coordinates

In this dissertation, a three dimensional Cartesian axis fixed at the origin point defines the coordinate system. Measurements along the axes of the coordinate system locate the biped robot’s postures. The world coordinate system is imagined as depicted in Figure 2.1 with the forward direction of movement proceeding along the X-axis.

Besides, each axis is defined as a rotational. The x-axis is known as the roll axis, the y-axis as the pitch axis and the z-axis the yaw axis.
2.2.2. Motion Planes

Humanoid robot locomotion, similar to a human being’s, is analyzed in three different perpendicular planes. These specified planes demonstrate the surfaces along which an action is performed.

1. Sagittal plane
This plane is perpendicular to the ground dividing the body into left and right. The actions of bending and extension movements occur in this plane. In this thesis, the robot locomotion is mostly discussed in the frame of sagittal plane which is XZ-plane.

2. Frontal plane
Frontal plane which is parallel to the YZ-plane divides the robot’s body laterally into front and back halves. Adduction movements occur in this plane.

3. Transverse plane
This plane, also known as Horizontal plane is parallel to XY-plane. The transverse plane separates the superior body of the robot from the inferior. In other words, this plane divides the body into the head and the feet. Rotational movements are usually illustrated in this plane.
2.3. Trajectory Generation

In the field of mobile robotics, planning a trajectory satisfying several specified criteria for a prior defined pathway is a primary step toward robot’s locomotion. A trajectory includes velocities, accelerations, and jerks along the path. Typically, there are two distinguishable methods for trajectory planning: Offline and Online or Real-time trajectory planning. An offline calculated trajectory is not influenced during its execution, while an online trajectory planning method is able to recalculate and/or adapt the robot’s motion behavior during the movement [7]. Accuracy improvement, interaction with a new environment, or reaction to a sensor might point out the necessities of the recalculation and/or adaptation in the locomotion of a biped robot.

2.4. Kinematics

Kinematics, as the first step toward robot control, describes an analytical study of specified mechanism motion geometry, without regard to the forces causing the motion,
respecting to a reference coordinate system. In robotics, kinematic divides into Forward Kinematics (FK) and Inverse Kinematics (IK).

![Figure 2.3 Forward Kinematics and Inverse Kinematics](image)

### 2.4.1. Forward Kinematics

In Forward Kinematics, a procedure is intended to compute position of the end effector by utilization of a set of relations among the robot’s joints. For Forward Kinematics implementation in a systematic manner, an appropriate kinematics model should be employed. For the Forward Kinematics there is always a unique solution. Denavit-Hartenberg method describing the robot kinematics is a commonly used method.

#### 2.4.1.1. Denavit-Hartenberg method

The Denavit-Hartenberg method (D-H) computes the robot’s kinematics by handling of homogeneous transformation matrix. In D-H method, four parameters associated with a particular convention are utilized in order to attach reference frames to links of the robot. The Denavit-Hartenberg kinematics parameters are classified into two categories: Link parameters and Joint parameters. The link parameters which are invariant includes: Link Length $a_l$ and Link Twist $a_l$. On the other hand, the joint’s parameters which refer
to the possible movements and are variable parameters involves: Joint Distance \(d_i\) and Joint Angle \(\theta_i\).

The homogeneous transformation matrix is obtained through following four steps [8]:

1. A rotation \(\theta_{i+1}\) about the \(Z_n\)-axis to bring \(X_n\) parallel with \(X_{n+1}\)
2. A translation \(d_{n+i}\) along the \(Z_n\)-axis to make the \(X\)-axes collinear
3. A translation \(a_{n+i}\) along the \(X\)-axis to make the \(Z\)-axes coincide
4. A rotation \(\alpha_{n+i}\) about the \(X_n\)-axis to bring \(Z_n\) parallel with \(Z_{n+1}\)

Where \(n\) represents the number of linkages in robot’s structure.

\[
I_{n+1}^n = \begin{bmatrix}
\cos \theta & -\sin \theta & 0 & 0 \\
\sin \theta & \cos \theta & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & \left(d_{n+1}\right) \\
0 & 0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

### 2.4.2. Inverse Kinematics

In Inverse kinematics joints positions are given and joints rotation computation is the desired task. Inverse kinematics approach most often involves a set of nonlinear and complicated equations. The robot controller unit must be able to solve a set of nonlinear simultaneous algebraic equations. Although a solution for these nonlinear equations may be obtained, existence of a unique set of joint coordinates is not guaranteed. Therefore, IK deals with problems such as nonlinear equations, existence of multiple solutions or inexistence of a solution, and singularities.
2.4.2.1. Singularity

In robotics, singularity describes a situation when two joints of the robot align along a common axis. In configuration of singularity, the robot loses its smooth path tracking. In a humanoid robot, a kinematic singularity could happen when the robot’s leg is fully stretched and the knee angle is extended to 180 degrees. To avoid such kinematic singularities, the robot trajectory is designed such that the robot knees are always bent [9].

2.5. Dynamics

Dynamics of a robotic system describes the relation between the forces applied on a robot and the resulted accelerations. Similar with kinematics, robot dynamics can be dealt in Forward or Inverse Dynamics approaches. Figure 2.4 illustrates Forward Dynamics (FD) and Inverse Dynamics (ID) givens and the outputs.

![Forward Dynamics vs. Inverse Dynamics](image)

Figure 2.4 Forward Dynamics vs. Inverse Dynamics

2.6. Stability

Robust balance in biped robots defines as stability which is generally counted as an essential characteristic of mobile robots. Stability might be Static or Dynamic.
2.6.1. Stability Margins

To remain stable, a robot’s Center of Gravity (COG) must fall within its stability margin also known as polygon of support. The Support Polygon is shaped by the area between robot’s feet on the floor.

![Support Polygon Diagram](image)

Figure 2.5 (a) support area in double support phase (b) support area in single support phase [10]

As shown in above figure when both feet touch the ground, support polygon is the area between the two feet. The support polygon is the area under the contact foot when just one foot touches the ground.

2.6.2. Static Stability

The system is statically stable if the projection of the center of gravity or Center of Mass (COM) is kept within or on the edge of the feet support area. Actually, static stability obtained through the robot’s mechanical design means standing without falling over.
2.6.2.1. Center of Mass

Center of Mass, also known as Center of Gravity, represents a specified point in the robot’s body describing reaction of the system to external torques. Mathematically, center of mass is an average of the masses factored by their distances from the coordinate system origin point.

\[ X_{\text{COM}} = \sum_{i=1}^{n} \frac{m_i x_i}{m_i} \quad (2.1) \]

In Eq. 2.1 \( n \) represents the number of linkages in robot’s structure.

2.6.3. Dynamic Stability

In biped robots, the polygon of support is significantly smaller than the robot itself; therefore, static stability is not possible unless the feet are unnaturally big. Furthermore, static stability causes energy inefficient walking procedure.

Dynamic stability, as an alternative approach, prevents a robot to be tipped over while moving. Dynamic stability gives the robot greater speed; but, this stability approach requires active control.

2.6.3.1. Zero Moment Point

Zero Moment Point (ZMP) specifies a point on the ground, \( Z_{\text{ZMP}} = 0 \), where the net moment of the inertial and the gravity forces has no component along the horizontal axes. ZMP is a concept associated with dynamics and control of legged locomotion. The concept of ZMP assumes the contact area is a planar surface where high friction existence restraints sliding feet.
Applying the Zero Moment Point criterion simplifies stability analysis of biped robot walking. Moreover, bipedal locomotion dynamic stability can be controlled when this criterion is employed. By taking principles of the Zero Moment Point into consideration, a stable walking pattern for the robot is systematically determined.

2.6.3.1.1. Huang ZMP Equations

Equations of Huang et al. [11] for calculating the ZMP position is brought in following:

\[
X_{ZMP} = \frac{\sum_{i=1}^{n} m_i(\ddot{x}_i + g)x_i - \sum_{i=1}^{n} m_i\ddot{x}_i z_i - \sum_{i=1}^{n} l_{iy} \ddot{\theta}_{iy}}{\sum_{i=1}^{n} m_i(\ddot{x}_i + g)} \quad (2.2)
\]

\[
Y_{ZMP} = \frac{\sum_{i=1}^{n} m_i(\ddot{x}_i + g)y_i - \sum_{i=1}^{n} m_i\ddot{y}_i z_i - \sum_{i=1}^{n} l_{ix} \ddot{\theta}_{ix}}{\sum_{i=1}^{n} m_i(\ddot{x}_i + g)} \quad (2.3)
\]

\[
Z_{ZMP} = 0 \quad (2.4)
\]

Where \(m_i\) is mass of \(i^{th}\) linkage, \(l_{iy}\) and \(l_{ix}\) are the inertial components, \(\ddot{\theta}_{iy}\) and \(\ddot{\theta}_{ix}\) represent the absolute angular velocity components around X-axis and Y-axis at the Center of Mass of \(i^{th}\) linkage respectively. And \(g\) is the gravitational acceleration. \((x_i, y_i, z_i)\) is the coordinate of the COM of \(i^{th}\) linkage on Cartesian coordinate system.

2.6.3.1.2. Cart-Table Model

Kajita et al [12], [13] has proposed the idea of Cart-Table Model as a simplified model to obtain the ZMP position. Figure 2.6 illustrates a simplified model of a biped robot
including a mobile cart on a mass less table. The cart located at COM of biped robot has mass $m$.

![Figure 2.6 Cart-Table Model](image)

Therefore, the complicated controlling problem of the biped robot is transformed to control a linear inverted pendulum. The torque $\tau$ can be formulated as:

$$\tau = -mg(x_{COM} - p) + m\dot{x}_{COM}z_{COM} \quad (2.5)$$

Where $g$ is the gravitational acceleration, always toward the ground. In accordance with the ZMP concept: $\tau = 0$; therefore, $X_{ZMP} = p$, this results in

$$x_{ZMP} = p = x_{COM} - \frac{\ddot{x}_{COM}}{g}z_{COM} \quad (2.6)$$

The ZMP position along $y$-axis is obtained through similar approach.

$$y_{ZMP} = y_{COM} - \frac{\ddot{y}_{COM}}{g}z_{COM} \quad (2.7)$$
2.7. Degree of Freedom

A joint motion restricted to rotation in a specified plane or translation along an axis is referred to one Degree of Freedom (DOF). As illustrated in Figure 2.7, the under study robot drawn in sagittal plane has six degree of freedom.

![Figure 2.7 Biped robot structure in sagittal plane](image)

In Figure 2.7, the independent coordinates required to thoroughly specify the robot configuration in sagittal plane include two degree of freedom at hip joint, one DOF at each knee and one DOF at each ankle joint. Since there is an actuator for every DOF, all of these DOF are controlled.

2.7.1. Redundancy

In applications of robotics where the number of controllable DOF is greater than the total DOF, the robot is redundant. Actually, redundancy is caused by deliberate duplication of key elements of a mechanism in order to enhance the system reliability.
Biped robots sometimes reveal kinematic redundancy which refers to inexistence of a unique mapping from the ZMP reference path to the joint trajectories.

2.8. Gait

Walking is periodically established by moving just one foot at once. The backwards and running are not in the scope of this thesis. And the manner of walking is known as gait.

2.8.1. Gait Phases

2.8.1.1. Double Support Phase

Double Support Phase (DSP) refers to situations when the biped robot has two isolated contact with the floor. In other words, the robot is supported by both feet in DSP; however, just one sole of robot’s feet can be fully touched with the floor.

2.8.1.2. Single Support Phase

In spite of double support phase, the Single Support Phase (SSP) happens when only one robot’s sole of foot is completely attached to the ground. As an illustration, Figure 2.8 and Figure 2.9 depict single and double support phases of walking respectively.

Figure 2.8 Single Support Phase
2.8.2. Swing Leg and Stance Leg

The leg performing a gait or moving forward is denoted as Swing Leg. On the other hand, the Stance Leg is fully supported by the Stance Foot with the ground. The stance leg supports weight of the biped robot.
Chapter 3

LITERATURE REVIEW

Since 1970s when Ichiro Kato from Waseda University initiated his work on humanoid robots, significant amounts of effort have been focused on the bipedal walking development in both real and virtual environments. Current examples of humanoid robots are not completely matching human’s skills; however, improvements achieved in this field are considerable. In this chapter, an extensive review of relevant previous works is presented.

Generally, approaches to bipedal walking concern with two major issues: kinematic smoothness and dynamic balance. Kinematics describes motion of joints, linkages and whole body of the robot without consideration of the causes of the motion. However, Dynamics is involved in studying of forces and torques and their effect on motion. There are a lot of ways to generate biped walking patterns; kinematics techniques, trajectory optimization methods, and physics-based approaches are several commonly used examples of strategies to plan bipedal walk. According to the application purposes or the environment where the robot works in, one walking planning technique may hold a level of privilege.

Trajectory optimization method which uses models of a robot and forward dynamics simulations for motions synthesis is mostly employed in animation applications. This method automatically creates or alters trajectories that satisfy motion criteria. In this
method, an operator specifies starting and ending poses and a reasonable moving pattern is automatically created. The optimization method applied on the trajectories of joints is subjected to constraints like the operator inputs and physics laws. Wei et al presented an approach combining trajectory optimization method with computer vision techniques to rebuild human motions from captured motions [14]. Requiring operator’s parameters adjustment and optimization criteria make the trajectory optimization approaches improper for real-time applications especially when long motions are desired.

Motion data obtained from either motion capture or manual calculation is the basis of kinematics techniques. Respectively, driven points are interpolated in order to build motion trajectory. This method gives smooth and continuous walking pattern that might be different from the original input sequences. This method of locomotion which is a basis of this dissertation appropriately responds to input parameters justifications, follows desired paths through different environments and is capable of obstacles avoidance. In this thesis, Inverse Kinematics (IK) provides a computation tool for manual calculations of joints positions on a Cartesian plane to locate biped linkages as desired. In most robotic applications such as arm manipulation and bipedal walking, redundancy count as an associated problem with Inverse Kinematics. Thus, data driven approaches are recommended combined with IK as a way of solving redundancy complication [15].

Bipedal Motion is inherently unstable; hence, to control locomotion skills the first objective is balance maintenance. Balance is categorized as static or dynamic. In static balance, the projection of the Center of Mass on the ground is always kept within the support area of stance feet. In absence of active controllers, minor deviations vanish, as long as the COM projection is not near the support polygon margin. COM kept within
support area results in stable gaits but very low walking speed [16]. Static balance disadvantages led researches to dynamic walking methods [17], [18]. Zero Moment Point algorithm suggested by Vukobratović [19] is one of the most popular dynamic bipedal walking methods. The algorithm has evolved since it was first proposed and been utilized in numerous full-sized humanoid robots for walking purposes. In Dynamic balance, Zero Momentum Point is retained within the margin of support polygon. In this approach, the projected center of mass might fall outside of the support area. This approach makes faster gaits than static balance; but, if the inertial forces generated from the acceleration of the robot body are not suitably controlled, a biped robot easily falls down. Moreover, during dynamic walking, disturbances may easily tip over a biped robot. Petman [20] is an example of humanoid robot controlled by dynamic balance.

Stable walking pattern synthesis received many researchers’ attentions. Huang et al efficiently identified key points and foot motion parameters constraints in single and double support phases of gait cycles. Consequently, constraints formulation led to an adaptive foot trajectory generation by third spline periodic interpolation. They then applied an iterative computation to derive the hip trajectory by a method for formulating the problem of the smooth hip motion with the largest stability margin. In this approach, the trunk of body is considered in parallel with Z axis in gait cycles [11]. A similar approach has been utilized by Mousavi et al for various surfaces [21] and combined paths [22]. Fattah et al presented a simplified comparable method by keeping hip joint at a constant height. They also developed an optimization method using Genetic Algorithm for maximum stability and minimum energy consumption [23]. Kim et al developed an approach for dynamic bipedal walking on uneven floors including an off-line walking
trajectory planning and six on-line controllers for Upright pose, landing angular momentum, landing timing, landing position, and vibration reduction and landing shock absorber [24]. A number of researchers have presented methods employing forward kinematic approach; for example, Zhang et al parameterized Denavit-Hartenberg formulation for an off-line kinematics model with on-line adjustment ability where homogeneous transformation matrices deduce the kinematics equations [25]. And Deng et al developed a virtual environment for biped robot simulation utilized DH for biped locomotion and dynamic balance [26].

Kajita et al used a simple linear dynamic analysis, the Three Dimensional Linear Inverted Pendulum Mode, for real time control of a biped robot [12]. They made use of preview control theory to compensate for the ZMP error caused by the differences between a simple model and the precise multi-body model. This method let a simulated biped robot to walk on spiral stairs [13]. In a similar way, Suleiman et al proposed an adapted identification algorithm to identify a quadratic system. This algorithm is based on using multiple walking patterns in order to identify an accurate model. The limitation of the proposed model is that when the walking trajectories are curved the ZMP behavior of the robot cannot be captured accurately [27].

Humanoid robots need high degree of autonomous adaptive learning. Conventional approaches to bipedal walking did not really succeed in complicated environments particularly for real-time applications. The main reason for the limited success is that these methods use a set of kinematic equations to demonstrate the physical movements of the robot. Traditional approaches are also computationally time consuming methods that are not appropriate to real-time applications; soft computing and artificial intelligence
offer an alternative approach. Soft computing methods address problem solution in a complementary task and include techniques such as Artificial Neural Networks, Fuzzy logic, evolutionary computation like Genetic Algorithms and probabilistic reasoning [28].

Farzaneh et al implemented Takagi–Sugeno (T–S) fuzzy systems for finding Finite Fourier series constants in order to overcome time consumption problem of the off-line method of trajectory generation such that made the approach applicable to real time bipedal walking planning [29]. Luo et al used a periodic function to plan the biped moving trajectory in sagittal plane. To assure short response time in trajectory tracking, a Fuzzy Sliding Mode controller including two independent controllers for positive and negative compensation was implemented [30]. Park et al used Fuzzy Logic to reduce the swing motion of the trunk to minimize disturbances [31]. A posture control also using Fuzzy algorithm was proposed by Choi et al in order to improve walking stability [32].

Ferreira et al proposed an Adaptive Neural-Fuzzy walking control of a biped robot. The implemented system was trained with the driven expert knowledge data set of the biped motion control [33]. Fan et al developed a supervised learning Fuzzy Neural Network (FNN) for fast humanoid robot gait generation [34]. To overcome the limitation of processing time of an eight-link biped robot dynamic equations Ferreira et al proposed two alternative intelligent computing control techniques, support vector regression (SVR) and a first-order Takagi–Sugeno–Kang type Neuro-Fuzzy Network. Both methods aimed to return robot’s torso correction for balance in sagittal plane. The SVR and the TSK NF controllers exhibited similar stability; but, the SVR controller runs faster [35]. Inherent complexity and imprecision in the collected environmental data led Vundavilli et al to utilize soft computing methods to solve ascending and descending gait generation
problems for biped robots. They developed Genetic-Neural and Genetic-Fuzzy approaches to model biped walking trajectory where Genetic Algorithm optimized the weights and knowledge bases in Neural Network and Fuzzy Logic Controller respectively [36]. Cardenas-Maciel et al [37] presented a Takagi-Sugeno Fuzzy Logic Controller using a Neuro-Fuzzy learning algorithm for generation of walking motions. Sabourin et al increased the stability of dynamic gait of a biped robot in existence of external disturbances with two phases control strategy. After dynamic gait sequences generation, Neural Networks is trained on joint trajectories of the reference gait. In the second step, the Neural Networks generates trajectories as learned during the first step. This approach also makes walking on irregular grounds possible [38].

Locomotion through kinematic techniques highly depends on the amount of data. On the other hand, capturing the full range of human motions is giving infinite ways of trajectory planning. In this thesis, walking techniques capable of reproducing dynamic balance are desired. Moreover, computational performance, ease of implementation and naturalness of motion are considered as main factors for the locomotion synthesis. However, these attributes may sometimes contradict each other.
4.1. Introduction

An Artificial Neural Network (ANN) is a data driven structure inspired by the way a biological brain performs data processing. ANNs are useful and efficient explicitly in describing physical equations of systems with complicated characteristics of processes.

The concept of ANN is an integrative approach derived from Computer Science, Neuroscience, Engineering, Mathematics, Psychology and Physics. Precisely, ANNs are assortment of mathematical models designed to emulate some of the aspects observed in their biological counterpart. The structure of the data processing system presented in ANNs is composed of numerous highly interconnected processor units that are analogous to neurons. These units resemble the synaptic process in a biological neural system by weighted connections. ANNs are capable of learning through samples, a key characteristic like adaptive learning in biological systems. In general, the learning or training process in an ANN takes place when the network is exposed to an appropriate set of input/output data and an algorithm justifies the weights on the connections surrounding the neurons. After each input/output pair, the assigned weights point out the knowledge acquired by the ANN on that specific problem.

ANNs demonstrated their potential capabilities of solving classification and function approximation problems, systems control, pattern recognition, and optimization. In the
following sections, architecture of ANNs and their applications by focusing on Multilayer FeedForward Networks are introduced.

4.2. Why Artificial Neural Networks?

ANNs facilitate an efficient alternative approach to computation. Despite ANNs using an adaptive learning algorithm for finding a given set of examples pattern, conventional methods utilize a certain formulated approach previously arranged instructions to solve a problem. This characteristic enables ANNs to deal with mathematically complicated implementations and also problems whose solution has not been explicitly formulated. The case of fast and efficient approximation of human like walking pattern is the main concern of this research. In this regard, ANN is represented here as a proven effective universal approximation method for various continuous systems. Furthermore, ANNs ability in implementation of nonlinear transformations is a particular feature when only partial knowledge of the system under study is obtainable.

4.3. Fundamental Concepts of Artificial Neural Networks

Before a discussion on types of ANNs, several basic concepts regarding ANNs structure in general are explored. Figure 4.1 illustrates a primary scheme of ANNs topology. The ANN shown in Figure 4.1 is constituted by highly interconnected neurons, circles illustrate neurons, arranged in the Hidden and Output layers. Inputs fed to the network are not counted as a layer.
Generally, the hidden layer is including more than one layer. The activity of neurons in hidden layer is determined by the weights of connections between neurons in hidden layer and input units. Similarly, the behavior of processing units in the output layer is dependent on the activities of the neurons in hidden layer and the neural weights between the hidden and outputs layers.

Although a neuron can have more than one input as shown in Figure 4.1, it manipulates all inputs as one single value called global input. The combination of the input vectors $I_i$ to obtain the global input $I_G$ is obtained through a mathematical operation called input function. This function is expressed as:

$$I_G = (I_1 \times W_{1i}) \odot (I_2 \times W_{2i}) \odot \cdots \odot (I_n \times W_{ni})$$

(4.1)

Where: $\odot$ represents an appropriate operator, i.e. summation, multiplication, maximization, etc. $n$ represents the total number of inputs of a certain neuron, and $W_i$ stands for the weight associated with every single input. The input values are multiplied by the weights previously assigned to the neuron; in fact, the weights modify the inputs effects.
Figure 4.2 illustrates the stimulus-response process in a neuron. Once the input vector is transformed into the global input through an input function, an activation function defines the state of the neuron. The activation function calculates the state of activity of a neuron by transforming the subtraction of the threshold ($\theta_i$) from global input into a value representing the activity level of neuron. Activation functions are monotonically increasing functions bounded between either $(0, 1)$ or $(-1, 1)$. Below, three commonly used activation functions are presented.

1. **Linear function**

   \[
   f(x) = \begin{cases} 
   -1 & x < \frac{-1}{a} \\
   ax & \frac{-1}{a} \leq x \leq \frac{1}{a} \\
   1 & x > \frac{1}{a}
   \end{cases}
   \]  

   Where $x = I_{ci} - \theta_i$ and $a > 0$

2. **Sigmoid function**

   \[
   f(x) = \frac{1}{1 + e^{-\theta x}}
   \]  

   (4.3)
Where $x = I_{Gl} - \theta_i$ and $f(x) \in (0, 1)$. The slope of the activation function is modified by the value of $g$.

3. Hyperbolic tangent function

$$f(x) = \frac{e^{g\times } - e^{-g\times }}{e^{g\times } + e^{-g\times }} \quad (4.4)$$

Where $x = I_{Gl} - \theta_i$ and $f(x) \in (-1, 1)$. The slope of the activation function is modified by the value of $g$.

The output function is the final unit in the neuron producing an output. The output function determines the value transferring to the next connected neurons. No output is carried to the proceeding neurons if the activation function is below the specified threshold.

4.3.1. Artificial Neural Networks Key Features

ANNs quick learning ability in performing complex perception tasks provides many advantages over conventional methods. A Neural Network-based perception system robustly handles a wider variety of situations than hand-programmed systems due to the ability of adaption to various situations. In systems such as autonomous mobile robots, moving purposefully in different environments is highly demanded.

Furthermore, ANNs are able to drive the essential characteristics of the learned inputs allowing them to correctly process incomplete or noisy data. Besides, artificial neurons are aimed to work in parallel similar to their biological counterparts. In the case of using one single processing unit, achieving real parallel processing is impossible. ANNs architecture and operating mode make them suitable for parallel processing implemented in multiprocessor hardware systems [39].
4.3.2. The Learning/Training phase

ANNs applications include two phases: training or learning and validation phases. As previously stated, ANNs are capable of learning by assigning new weights to the interconnections after each iteration. These weights are updated on the basis of acquired knowledge of patterns found on every new training set. In general, the optimal values of weights are obtained through optimization of an energy function. For instance, a common criterion applied with supervised learning is the minimization of the least square error between the ANN output and the desired one. And in validation phase, the trained network outputs are compared with the desired output of the system.

There are basically two methods for training an Artificial Neural Network:

A. Supervised Learning/Training

B. Unsupervised or Adaptive Learning/Training

In supervised learning, inputs and outputs are provided. After inputs processing, the network compares the corresponding outputs against the desired outputs. Then, errors are fed back to the system in order to adjust the weights controlling the network. This process continues till the weights are properly improved. Actually, in this learning method, the process is controlled by an external agent, a supervisor, that determines the response expected to be generated by the ANN for a particular input.

Differently, a network with an unsupervised learning does not require any external guidance for weight adjustments. In unsupervised learning method, the network is provided with input but no desired output is available. Then, system decides what characteristics are utilized to categorize the input data. During the adaptive training, there is no knowledge of the environment indicating if the generated response is correct or not.
Actually, the driven output from networks employing unsupervised learning may be interpreted in different ways. For instance, output could describe level of resemblance between the current input and previous ones. Furthermore, in clustering applications, the output of the network designates to which category a particular input belongs.

### 4.4. Multilayer FeedForward Artificial Neural Networks

Architecture of the multilayer network involves inputs, hidden layers, and an output layer. Multilayer ANNs possess multi-perceptron hierarchical structure. A multilayer network is characterized by a highly connected topology since every single input is connected to all the neurons in the hidden layer, and all the processing units in the hidden layers is connected to all neurons in the front layer. A typical multilayer ANN topology is presented in Figure 4.3

![Figure 4.3 A multilayer feedforward network topology](image)

The network in Figure 4.3 includes three inputs, one hidden layer including four neurons, and two neurons in the output layer representing two outputs of a system. The input signals are propagated through the network form a layer to the next layer in a forward direction. This is why this particular architecture is called Multilayer FeedForward Neural Network.
A multilayer architecture is used to overcome some of the defects of single layer networks. The principal advantage of the Multilayer FeedForward Networks is their capability to approximate nonlinear multiple dimensions functions.

4.4.1. The Backpropagation Training Algorithm

A question arising with the Multilayer FeedForward Network is how to train the neurons in hidden layer if the desired outputs are not available. The Backpropagation algorithm has been developed to overcome this problem. This weight modifier method is a commonly used ANNs model which is based on Multilayer FeedForward architecture with supervised learning. First, the inputs set is presented to the network; then, the output of each layer and the final output of the network are obtained in a forward direction. Since the desired final outputs are known, the weights adjustment is performed similar with weights modification of single layer network. The weight factors of neurons in the hidden layers are adjusted by backward propagation of output error toward the hidden layers. This process is repeated for every single sample in the training set. The whole cycle through a training set is called an epoch. The number of epochs required for completion of training process predominantly depends on the error estimated in the output layer; however, many other factors could potentially affect it.

Delta rule, also known as the Least Mean Square (LMS) algorithm is the basis of training procedure. In this method, weight adjustment is accomplished by minimization of the mean square of the error. Conventionally, the LMS method was employed in the perceptron or single layer feedforward network for the weights modification. A generalized delta algorithm was further developed for the multilayer networks structure.
The generalized delta algorithm belongs to a category of methods performing gradient descent. Justifications made by the learning algorithm to the weights enhance the accuracy of the network through a descent along gradient of an error surface. The error surface involves all possible weights and input vectors; and, it is not varying with time. Generally, this highly complicated error surface contains numerous extremums. The backpropagation algorithm is intended to search the error surface for a minimum via gradient descent. A global minimum is obtained by the surface search; however, this is not guaranteed.

Figure 4.4 illustrates a network consisting of four inputs, single hidden layer including three neurons and three outputs in the last layer. As depicted in the figure, a backpropagation network is trained in two steps: the activity from the inputs toward the output layer and backward error propagation for the weights adjustment. The error function for the above figure can be presented as following equation.

\[ E = \frac{1}{2} \sum_k (d_k - y_k)^2 \]  (4.5)
Calculation of error derivative respecting to variation of weights fed to an output neuron is an essential part in order to determine the method of weights correction such that the descent of the error function along the gradient is maximized.

\[ \frac{dE}{dw_{hk}} = \frac{dE}{dy_k} \times \frac{dy_k}{du_k} \times \frac{du_k}{dw_{hk}} \]  \hspace{1cm} (4.6)

The left hand side of Eq. 4.6 stands for the derivative of the error function with respect to the \( h^{th} \) weight feeding to the \( k^{th} \) output neuron, \( \frac{dE}{dy_k} \) represents the error function derivative respecting to the \( k^{th} \) output neuron, \( \frac{dy_k}{du_k} \) is the derivative of the \( k^{th} \) output neuron signal with respect to the neuron activation function \( du_k \), and \( \frac{du_k}{dw_{hk}} \) stands for the derivative of the \( k^{th} \) output neuron activation with respect to its \( h^{th} \) weight. By calculation of the entire derivative on the right side of Eq. (4.6) the below equation is obtained.

\[ \frac{dE}{dw_{hk}} = (y_k - z_k)[y_k(1 - y_k)]x_h \]  \hspace{1cm} (4.7)

The weight adaptation algorithm in a backpropagation network is same as the delta rule. Thus, a generalized delta rule for backpropagation is obtained by:

\[ \Delta w_{hk} = \eta (z_k - y_k) [y_k(1 - y_k)]x_h \]  \hspace{1cm} (4.8)

Where \( \eta \) is a positive constant value representing the learning ratio. The above equation represents the weight correction of the \( k^{th} \) output neuron.

Eq. (4.8) is not applicable to the hidden layers. The hidden layers modifier function is formulated as following equation.

\[ \Delta w_{hk} = \eta d_kx_h \]  \hspace{1cm} (4.9)
Where, $d_k$ and $x_h$ respectively denote the error function and the activation signal of the $h^{th}$ hidden neuron connected to the $k^{th}$ output node. The error function $d_k$ is described by:

$$d_k = \eta \cdot y_k (1 - y_k) (y_k - z_k) \quad (4.10)$$

Weights correction in the hidden layers is initiated from the hidden layer next to the output layer; where, a neuron in this specified layer is modified regarding to the rate of its effect or contribution to the error of an output node. Therefore, the error function for a neuron in hidden layer is:

$$d_h = H(1 - H) \sum_k d_k w_{hk} \quad (4.11)$$

In Eq. (4.11) $w_{hk}$ signifies the weight factor between the $h^{th}$ neuron of hidden layer and the output node $k$, $H$ symbolizes the neuron of hidden layer output and $d_k$ is the output error of $k^{th}$ neuron. Once all the weight factors in the last hidden layer are adjusted, same procedure are applied to the next hidden layers. Since the error is propagated from the output layer back to hidden layers, this algorithm is known as backpropagation.

### 4.4.2. Feedforward Artificial Neural Network Implementation

Before proceeding to ANN training, characteristic elements of the network are required to be well defined. However, the key factors of a network are highly dependent on the type of the problem, for designing a network several guidelines are beneficial to be taken in account.
4.4.2.1. Number of Hidden Layers

When sufficient number of layers and neurons in each layer are employed, Multilayer FeedForward Neural Networks are efficiently capable of complicated approximation and classification. Although there is no routine parameterized algorithm returning the optimum number of layers and neurons for a certain task, there exist some hints of specifying the layers quantity in the network. According to the ANNs structure, networks require to adjust more weight factors as neurons are increased when the number of layers is increased. Therefore, systems are delayed longer in real time applications of Neural Networks. In addition, ANNs are usually trained empirically by trial and error algorithms. In other words, number of layers is varied until an output with a desired error tolerance is achieved.

4.4.2.2. Number of Neurons

There is no explicit method to determine the number of neurons in the hidden layers. Actually, quantifying the number of neurons is heuristic; however, this number and the size and complexity of the problem are directly related. As a rule of thumb, the more complex system is, the more neurons are required to meet the desired accuracy. Moreover, the more neurons are utilized in the hidden layers, the more computational time and effort is required. Literally, excessive neurons in the hidden layers lead to an over fitted network for the given data set and consequently a deficient generalization of the problem.

Furthermore, training data set samples should be greater than the number of connections in the network. Commonly, the trial and error approach with an initial
moderate number of neurons is applied and gradually increasing this number to achieve the desired error. The number of neurons in the output layer equals to the number of output of the system.

4.4.2.3. Initial Weights

Initial weights epitomize knowledge of the network at the beginning point. Whereas It is desired to get as close as possible to the global optima, the learning procedure is probably stuck in local minima in lack of appropriately modified weights. Empirically, different initial weights are fed into the network to improve the performance of it.

4.5. Radial Basis Function Networks

After the FeedForward Networks, the Radial Basis Function Network (RBFN) is of the mostly used Artificial Neural Network models. This specific network architecture embedded into a two-layer feedforward neural network including one hidden layer and an output layer. The idea of RBFN is based on locally tuned overlapping receptive fields found in the cerebral cortex. Therefore, approaches based on receptive fields are called Radial Basis Function Networks approximations.

![Figure 4.5 Radial Basis Function Network structure](image)
In Figure 4.5, the squares on arrows between hidden and output layers represent the weighted connection. The activation function of the \(i^{th}\) neuron in the hidden layer is:

\[
w_i = R_i(x) = (R_i \|x - u_i\|/\sigma_i)
\]

In Eq. 4.12, \(x\) is a multidimensional input vector and the vector \(u_i\) has same dimension as \(x\). \(R_i(x)\) denotes the \(i^{th}\) Radial Basis Function usually presented as a Gaussian distribution. Therefore:

\[
w_i = R_i(x) = R_i(\|x - u_i\|) = \exp\left[-\frac{(x-u_i)^2}{2\sigma_i^2}\right]
\]

The activation level of the RBF \(R_i(x)\) of the \(i^{th}\) node is at its maximum when the input vector \(x\) is at the center of \(u_i\). Therefore, the output of the Radial Basis Function Network can be treated as a weighted summation of the outputs.

\[
d(x) = \sum_{i=1}^{N} c_i w_i = \sum_{i=1}^{N} c_i R_i(x)
\]

Where \(c_i\) is the output value of the \(i^{th}\) radial basis neuron and \(N\) is the number of receptive field units.

Assigning a linear function to the Radial Basis units output function results in a linear mapping at Radial Basis Function Network output. The linear transformation of the outputs is obtained by the following expression:

\[
c_i = a_i^T x + b_i
\]

Where \(a_i\) is a vector parameter and \(b_i\) is a scalar parameter called bias.

As mentioned before, a nonlinear mapping from the input space into the hidden layer space is followed by a linear mapping from the hidden space to the output space in Radial Basis Function Network. Therefore, the learning phase consists of locating sufficient set of centers, and weights that better match the training data set.

37
4.5.1. Function Approximation by Radial Basis Function Networks

Assume \( x_i \), \( x_i \in \mathbb{R}^p \), a collection of \( N \) points and an equivalent set of target values \( x_i \in \mathbb{R}^1 \), \( i = 1,2,\ldots,N \) find a function \( \mathbb{R}^p \rightarrow \mathbb{R}^1 \) that satisfies this condition.

\[
f(x_i) = d_i \quad (4.16)
\]

The interpolating surface should pass through all the training points \( d_i \). Each training input \( x_i \) serves as a center for the Radial Basis Function. From Eq. (4.14) a function of interpolation is obtained as:

\[
f(x_i) = \sum_{i=1}^{N} c_i \varphi(||x - x_i||) \quad (4.17)
\]

where \( \varphi \) is the Gaussian Radial Basis Function.

From Eq. (4.16) and Eq. (4.17), a system of \( N \) linear equations for the \( N \) unknown weight coefficients \( c_i \) is achieved.

\[
\begin{bmatrix}
\varphi_{i1} & \cdots & \varphi_{iN} \\
\vdots & \ddots & \vdots \\
\varphi_{N1} & \cdots & \varphi_{NN}
\end{bmatrix}
\begin{bmatrix}
c_1 \\
\vdots \\
c_N
\end{bmatrix}
= 
\begin{bmatrix}
d_1 \\
\vdots \\
d_N
\end{bmatrix} \quad (4.18)
\]

Where \( \varphi_{ij} = \varphi(x_j - x_i) \), \( j, i = 1,2,\ldots,N \). The above equation can be simply shown as:

\[
\Phi c = d \quad (4.19)
\]

\( \Phi \) is positive definite; then, a solution is obtained through below equation as long as \( \Phi \) is nonsingular.

\[
c = \Phi^{-1} d \quad (4.20)
\]

Since Radial Basis neurons only respond to relatively small regions of the input space, an interpolation or approximation RBFN employs one neuron in the hidden layer for every single input [40]. Furthermore, RBFN learning is typically accomplished quite faster than training of a FeedForward Network with sigmoidal activation functions.
Depending on the problem, RBFN may result in more reliable performance; although, avoidance of over fitting during the training phase must be considered for an acceptable generalization.

4.6. Applications of ANNs

Artificial Neural Networks are widely used in a wide variety of applications. ANNs have been applied in a broad range of fields including Chemistry, Biomedical, Industrial, Physics, Robotics and Control Systems, and Financial. For example, applications of ANN methods such as the prediction of ionization potential, lipophilicity of chemicals and relative permittivity and oxygen diffusion of ceramic materials can be discovered in Computational Chemistry [41]. An example of ANNs application in manufacturing is the determination of a suitable control setting for a plant. In fact, complicated processes with a multitude of mathematical formulas can be heuristically modeled by an Artificial Neural Network.

Radial Basis Function Networks traditionally used for function approximation and classification are now used in a variety of applications, such as Face Tracking and Face Recognition, Robotic Control, Antenna Design, Channel Equalizations, Computer Vision and Graphics, and solving Partial Differential Equations with boundary conditions [42]. Further instances of ANNs application have also been presented in Chapter 2, Literature Review.
4.7. Summary

This chapter was mainly concerned with features of ANNs, where several critical points in question such as various ANNs architectures and learning/training methods were discussed. Adaptive learning capability was introduced as a key characteristic of ANNs providing a powerful solution for a variety of approximation and classification problems. Two commonly used types of ANNs: FeedForward Backpropagation Networks and RBFN were reviewed. It was also mentioned that the significant amount of interest in these two methods is particularly due to their capabilities in nonlinear continuous functions approximation. And finally, a review of multiple ANNs applications was reviewed.
Chapter 5

METHODOLOGY

5.1. Introduction

For stable bipedal walking on various surfaces, adaptation to the ground condition and stability maintenance are two requisite tasks of a humanoid robot. Adaptation here is defined as the robot capability to employ different appropriate pattern of motion as the conditions change. For instance, a biped robot utilizing an adaptive walking method is assumed to be capable of lifting its feet high enough to negotiate an obstacle or could take a wide enough gait length to pass over a pit such that its whole body posture is stable. In other words, a desirable walking pattern must fulfill the environmental conditions and balance constraints. The proposed methodology derives relationships of characteristic motion parameters which yields a motion scheme adaptive to variations in the interactive environment.

For the dynamic balance of the biped robot, an efficient method is suggested for upper body or trunk motion formulation so that stability of the biped robot during walking cycles is assured. Therefore, the ZMP concept is applied on a seven-link biped robot modeled as an inverted pendulum. The trunk is located such that ZMP position always falls within support area of robot’s feet. In following sections in this chapter, a non-conventional proposed method of dynamic trajectory planning which satisfies the surface conditions and dynamic stability constraints is discussed in detail.
5.2. The Biped Robot Model

An anthropomorphic biped robot as shown in Figure 5.1 is considered as the reference model throughout this thesis. In this model, each leg consists of a thigh, a shank, and a foot. The under study model has six degrees of freedom in sagittal plane including two DOF in the hip joint, one in each knee joint, and one in each ankle joint.

![Biped robot model in sagittal plane](image)

5.3. The Robot Kinematics

Bipedal walking is a periodic incident composed of two phases: Double Support Phase and Single Support Phase. During the DSP, both feet are in contact with the ground. This phase begins when the heel of the front foot touching the ground, and ends when the toe of the rear foot leaving the ground. In a gait, the double support phase is followed by a single support phase. During the SSP, one foot has a fixed position and the other foot swings forward. In human locomotion, the DSP time length is about 20% of a whole gait cycle [11]. In the below figure, double support phase and single support phase are
depicted. In Figure 5.2, the swing leg and stance leg are represented in red and blue colors respectively.

![Figure 5.2 DSP and SSP time duration](image)

Through the proposed methodology in this dissertation, both feet and hip joints trajectories are first derived; consequently, all the other components such as knee joint and sole of foot trajectories are mathematically determined by taking the biped robot kinematic constraints into consideration. Therefore, a unique walking pattern is formed by the foot and hip joints trajectories. In following sections, trajectory planning for a biped robot in sagittal plane on various ground condition such as flat, incline and decline surfaces and also walking up and down on stairs with varying height and depth are thoroughly discussed.

**5.3.1. Bipedal Walking on a Flat Surface**

During a gait cycle, a limited number of ankle and hip joints positions in both double support phase and single support phase can be identified with respect to the coordinate system origin. The coordinates of these several points are mathematically computed through relationships among specifications of the robot’s structure and a sort of key
assumptions. Once positions of all these featured points are calculated, the pattern of robot walking for the whole gait cycle can be interpolated. The interpolating techniques are discussed in section 5.4.

Although interpolation looks the most feasible method for mapping a desirable walking pattern to the derived points, there are other possible methodologies to obtain the trajectory. Developing an analytical set of equations or finding approximated points in between of each pair of available points are examples of other methods than interpolations. However, these mathematically complicated techniques lacking desired level of accuracy suffer from higher computation time.

A robot requires trajectories for both swing and stance legs to accomplish the procedures of one single gait. Although the stance foot is fixed during a gait, the knee joint position of stance leg is varying. The left and right legs in the seven-links biped model are conjunct in hip joint; therefore, a constant ankle joint position and the varying trajectory of hip joint result in a moving pattern of knee joint. By calculating all the joints positions through this inverse kinematic method, joints angles which are used in robot dynamic analysis and controlling system can be simply determined.

Figure 5.3 Five identified key points in a gait cycle
In Figure 5.3, robot’s postures at five distinguished time during a single gait are illustrated. In this illustration, $D_s$ represents gait length which refers to the distance between ankle joints of swing and stance legs at initial time ($t = 0$) or at the end of a gait cycle ($t = T_c$). $T_c$ stands for a gait cycle time or a gait period and $T_d$ is double support phase duration. The swing foot obtains its maximum height at $T_m$.

5.3.1.1. Trajectory of The Swing Leg

As the robot schematic in sagittal plane in Figure 5.3 indicates, the swing leg, in red color, trajectory identification requires computation of three joints including ankle, hip, and knee patterns of movement and also sole of the foot angle variation.

5.3.1.1.1. The Swing Leg Ankle Joint Trajectory

In order to facilitate adaptation to various surface mode for the biped robot, the robot’s foot trajectory must first be specified. During the walking cycles, the swing leg ankle joint displacements along X-axis and Z-axis are represented by $X_{a_{SW}}$ and $Z_{a_{SW}}$ respectively. Following equations describe the ankle joint positions in XZ-plane for the initial time, at the end of double support phase, and when the ankle joint reaches its maximum height. The coordinates of the ankle joint at the end of gait period and the end of DSP in the next cycle are also calculated through the formulations.
In Eq. 5.1, $L_{an}$ is the length of link connecting the ankle joint to the sole of foot, $L_{af}$ and $L_{ab}$ represent the rear and front parts of the foot as shown in Figure 5.4.

$q_b$ and $q_f$ are assigned angles to the swing foot as it leaves and touches down the ground respectively. Sole of the swing foot is assumed to be completely in contact with the walking surface at the end of gait period and DSP of next cycle. In comparison with methods which propose sole of the swing foot leveled with the ground, these particular assumptions here make the bipedal locomotion more human like.
In Figure 5.6, the swing ankle joint specifications at the time $T_m$ are illustrated; where, the $L_{ao}$ is the position of ankle joint on X-axis as the joint’s height reaches $H_{ao}$, its maximum point on Z-axis.

The projected swing ankle joint positions on Z-axis in a gait time interval are parameterized in Eq. 5.2.
\[ Z_{\text{SW}}(t) = \begin{cases} L_{an} & t = 0 \\ L_{af} \sin q_b + L_{an} \cos q_b & t = T_d \\ H_{ao} & t = T_m \\ L_{ab} \sin q_f + L_{an} \cos q_f & t = T_c \\ L_{an} & t = T_c + T_d \end{cases} \] (5.2)

And, the following equation represents sole of the swing foot angle variations.

\[ \theta_{\text{SW}}(t) = \begin{cases} 0 & t = 0 \\ q_b & t = T_d \\ -q_f & t = T_c \\ 0 & t = T_c + T_d \end{cases} \] (5.3)

Where \( q_b \) and \( q_f \) are the designated angles to the swing foot as it leaves and lands on the ground accordingly. As described in the Eq. 5.3 the swing foot sole is assumed to be completely in contact with the walking surface at the end of gait period (\( t = T_c \)) and DSP of next cycle (\( t = T_c + T_d \)).

5.3.1.1.2. The Hip Joint Trajectory

The hip joint motion pattern in sagittal plane can be particularly identified by calculation of \( X_h \) and \( Z_h \). \( X_h \) and \( Z_h \) indicate the hip joint displacements along X-axis and Z-axis respectively.

\[ X_h(t) = \begin{cases} \chi_{ed} & t = 0 \\ D_s - \chi_{sd} & t = T_d \\ D_s + \chi_{ed} & t = T_c \end{cases} \] (5.4)
Where \( x_{ed} \) and \( x_{sd} \) represent distances from hip joint to stance ankle joint at initial and final points of the double support phase respectively. \( x_{ed} \) and \( x_{sd} \) are illustrated in Figure 5.7.

\[ \begin{align*}
0 < x_{ed} < 0.5D_s \\
0 < x_{sd} < 0.5D_s
\end{align*} \tag{5.5} \]

\( x_{ed} \) and \( x_{sd} \) can be manually modified in appropriate intervals as given in Eq. 5.5 [11].

The projected hip joint positions on Z-axis for three key frames in a gait period are formulated in Eq. 5.6. In a gait duration, it is assumed that the hip joint obtains its maximum value with respect to the origin of the system coordinate along Z-axis, \( H_{max} \), at the middle of single support phase and the lowest position, \( H_{min} \), occurs at the middle of the double support phase.
5.3.1.1.3. The Knee Joint Trajectory

Due to the fact that ankle and hip joints are connected at the knee joint through thigh and shank links in a human leg structure, calculation of the ankle and hip joints trajectories can simply result in a unique knee trajectory. As illustrated in Figure 5.8, the point with greater value along X-axis resulted by intersecting circles $C_{hi}$ and $C_{ai}$ is the desired knee joint position in each step during a gait period. $C_{hi}$ is a circle centered at the hip joint with radius of thigh length, and $C_{ai}$ represents a circle with center at the swing ankle joint with radius of shank length.

\[
Z_h(t) = \begin{cases} 
H_{\text{min}} & t = 0.5T_d \\
H_{\text{max}} & t = 0.5(T_c - T_d) \\
H_{\text{min}} & t = T_c + 0.5T_d 
\end{cases} \quad (5.6)
\]

Since the leg hardly gets completely straight up which means tangent circles do not exist, there are always two intersection points. The point with greater value along X-axis
between two circles $C_{hi}$ and $C_{ai}$ intersection points is selected because human leg is only bent in one direction, forward direction.

### 5.3.1.2. Trajectory of The Stance/Support Leg

During a gait cycle, position of stance ankle joint is constant as it has no movement. The horizontal distance of support ankle joint ($X_a$) from the origin of coordinate system equals to the gait length ($D_S$). The joint height ($Z_a$) is the length of the link connecting ankle joint to the foot ($L_{an}$).

$$
\begin{align*}
X_a &= D_S & 0 \leq t \leq T_c \\
Z_a &= L_{an} & 0 \leq t \leq T_c
\end{align*}
$$

Since the hip joint is moving forward due to swing leg motion, knee joint of stance leg adopts a varying motion pattern. The stance knee joint trajectory is identified by the aforementioned technique of intersecting circles.

Once all the joints trajectories are recognized for a single gait period, in order to proceed walking in similar ground conditions, the joints trajectories can be just repeated for the next cycles. The task of replication for achieving successive gait cycles is accomplished through trajectories replacement in turn. For further clarification, if the left leg is assumed to be the swing leg in the first cycle, then it is the stance leg during the second cycle and the origin of coordinate system is horizontally shifted forward by a gait length ($D_S$). Therefore, left leg trajectory in previous cycle is the right leg trajectory during the next consecutive gait cycle.
Otherwise, if the robot encounters a different ground condition, the whole procedure of joints motion planning must be repeated such that updated trajectories adapting the new circumstance are obtained. Variation of several bipedal walk parameters such as gait length \((D_s)\), gait period \((T_c)\), maximum height of swing ankle joint \((H_{ao})\), \(T_m\), \(T_d\), \(x_{ed}\), \(x_{sd}\), \(q_b\), and \(q_f\) can desirably modify trajectory of the whole robot body locomotion. Due to controllability of gait length and gait time period, the average velocity of the robot displacement is highly adjustable.

\[
V_{avr} = \frac{D_s}{T_c} \quad (5.8)
\]

### 5.3.2. Biped Robot Walking Pattern on Sloped Surfaces

As the biped robots are intended to perform interactions with human being environments, the capability of walking on various surfaces is an essential task. The sloped surfaces, either inclined or declined surface, are of those circumstances that humans often encounter with. Bipedal trajectory planning techniques for walking on both inclined and declined surfaces are investigated in proceeding subsections.

#### 5.3.2.1. Walking on Inclined Surfaces

Similar with trajectory planning for bipedal walking on a flat surface, in a gait cycle of the robot’s walking on an inclined surface, with slope of \(\lambda\), several specified ankle and hip joints positions and consequently knee joint positions in both DSP and SSP can be identified regarding to origin of the coordinate system. Through a number of equations based on the robot’s structure characteristics and a group of assumptions, these points
coordinates are calculated. Subsequently, these computed points are interpolated so that the walking pattern for the whole gait cycle can be described by curves. Due to the fact that these curves consist of numerous ankle, knee, or hip joint positions at various time step, a smooth walking manner is facilitated. Following formulations result in positions of swing ankle \((X_{aSW,inc}, Z_{aSW,inc}, \theta_{aSW,inc})\) and hip \((X_{h,inc}, Z_{h,inc})\) joints along the X-axis and Z-axis respectively.

\[
X_{aSW,inc} = \\
\begin{cases}
  L_{ab} \cos \lambda - L_{an} \sin \lambda & t = 0 \\
  L_{ab} \cos(\lambda - q_b) - L_{an} \sin(\lambda - q_b) + \left( L_{af} + L_{ab} - (L_{af} + L_{ab}) \cos q_b \right) \cos \lambda & t = T_d \\
  L_{ab} \cos(\lambda - q_b) - L_{an} \sin(\lambda - q_b) + \left( L_{af} + L_{ab} - (L_{af} + L_{ab}) \cos q_b \right) \cos \lambda & t = T_m \\
  2D_s \cos \lambda - L_{an} \sin(\lambda + q_f) - L_{ab} (1 - \cos q_f) \cos \lambda & t = T_c \\
  2D_s \cos \lambda - L_{an} \sin \lambda & t = T_c + T_d 
\end{cases}
\]  

\[(5.9)\]

\[
Z_{aSW,inc} = \\
\begin{cases}
  L_{an} \cos \lambda + L_{ab} \sin \lambda & t = 0 \\
  X_{aSW,inc}(T_d) \tan \lambda + L_{an} \cos(\lambda - q_b) + \left( L_{af} \cos(\lambda - q_b) - L_{an} \sin(\lambda - q_b) \right) \tan \lambda - L_{af} \sin(\lambda - q_b) & t = T_d \\
  X_{aSW,inc}(T_m) \tan \lambda + L_{an} \cos(\lambda - q_b) + \left( L_{af} \cos(\lambda - q_b) - L_{an} \sin(\lambda - q_b) \right) \tan \lambda - L_{af} \sin(\lambda - q_b) & t = T_m \\
  L_{an} \sin(\lambda + q_f) + L_{ab} \sin(\lambda + q_f) + (2D_s - L_{ab}) \sin \lambda & t = T_c \\
  L_{an} \cos \lambda + 2D_s \sin \lambda & t = T_c + T_d 
\end{cases}
\]  

\[(5.10)\]

\[
\theta_{aSW}(t) = \\
\begin{cases}
  -\lambda & t = 0 \\
  -\lambda + q_b & t = T_d \\
  -\lambda - q_f & t = T_c \\
  -\lambda & t = T_c + T_d 
\end{cases}
\]  

\[(5.11)\]
\[ X_{h,\text{Inc}}(t) = \begin{cases} x_{ed} & t = 0 \\ D_S \cos \lambda - x_{sd} & t = T_d \\ D_S \cos \lambda + x_{ed} & t = T_C \end{cases} \] (5.12)

\[ Z_{h,\text{Inc}}(t) = \begin{cases} H_{\text{min}} + D_S \sin \lambda & t = 0.5T_d \\ H_{\text{max}} + D_S \sin \lambda & t = 0.5(T_C - T_d) \\ H_{\text{min}} + D_S \sin \lambda & t = T_C + 0.5T_d \end{cases} \] (5.13)

In above equations, \( \lambda \) represents the surface slope which is treated as an input in the proposed technique. The swing leg’s knee joint trajectory is obtained by intersecting circles technique as described before. For a biped robot to execute the procedure of a gait, both swing and support leg trajectories are required. Despite fixed position of the stance foot during a gait cycle, the knee joint position of stance leg is varying due to the hip joint movements.

5.3.2.2. Walking on Declined Surfaces

An analogous approach is employed to obtain a smooth bipedal walking pattern in the case of declined ground condition with slope of \( \lambda \). Following equations return swing leg’s ankle \( (X_{SW,\text{Dec}}, Z_{SW,\text{Dec}}, \theta_{SW,\text{Dec}}) \) and hip \( (X_{h,\text{Dec}}, Z_{h,\text{Dec}}) \) joints positions along the X-axis and Z-axis respectively.

\[ X_{SW,\text{Dec}} = \begin{cases} L_{ab} \cos \lambda + L_{an} \sin \lambda \\ L_{ab} \cos(\lambda + q_b) - L_{an} \sin(\lambda + q_b) + \left( L_{af} + L_{ab} - (L_{af} + L_{ab}) \cos q_b \right) \cos \lambda \\ 2D_S \cos \lambda - L_{an} \sin (q_f - \lambda) - L_{ab} \left( 1 - \cos q_f \right) \cos \lambda \\ 2D_S \cos \lambda - L_{an} \sin \lambda \end{cases} \]

\[ t = 0 \quad t = T_d \quad t = T_m \quad t = T_C \quad t = T_C + T_d \] (5.14)
In pattern planning for the robot walking on stairs, either going up or down stairs, the gait length \(D_s\) is set to stair depth. Bipedal trajectory planning during walking in such circumstances are proposed in following subsections.
5.3.3.1. Walking Up Stairs

Following equations describe swing leg’s ankle \((X_{SW,Ustr}, Z_{SW,Ustr}, \theta_{SW,Ustr})\) and hip \((X_{h,Ustr}, Z_{h,Ustr})\) joints positions at key time frames during a gait period along the X-axis and the Z-axis respectively when the biped robot is going up stairs.

\[
X_{SW,Ustr}(t) = \begin{cases} 
0 & t = 0 \\
L_{an} \sin q_b + L_{af} (1 - \cos q_b) / L_{ao} & t = T_d \\
2D_S - L_{an} \sin q_f - L_{ab} (1 - \cos q_f) / 2D_S & t = T_m \\
& t = T_c + T_d 
\end{cases} \tag{5.19}
\]

\[
Z_{SW,Ustr}(t) = \begin{cases} 
L_{an} & t = 0 \\
L_{af} \sin q_b + L_{an} \cos q_b / H_o + H_{Str} & t = T_d \\
H_{Str} + L_{ab} \sin q_f + L_{an} \cos q_f / H_{Str} + L_{an} & t = T_m \\
& t = T_c \\
& t = T_c + T_d 
\end{cases} \tag{5.20}
\]

\[
\theta_{SW,Ustr}(t) = \begin{cases} 
0 & t = 0 \\
q_b & t = T_d \\
-q_f & t = T_c \\
0 & t = T_c + T_d 
\end{cases} \tag{5.21}
\]
\[ X_{h,Ustr}(t) = \begin{cases} 
  x_{ed} & t = 0 \\
  D_s - x_{sd} & t = T_d \\
  D_s + x_{ed} & t = T_c 
\end{cases} \quad (5.22) \]

\[ Z_{h,Ustr}(t) = \begin{cases} 
  H_{\text{min}} & t = 0.5T_d \\
  H_{\text{max}} & t = 0.5(T_c - T_d) \\
  H_{\text{min}} & t = T_c + 0.5T_d 
\end{cases} \quad (5.23) \]

### 5.3.3.2. Walking Down Stairs

Following equations describe swing leg’s ankle \(X_{a_{SW,Dstr}}, Z_{a_{SW,Dstr}}, \theta a_{SW,Dstr}\) and hip \(X_{h,Dstr}, Z_{h,Dstr}\) joints positions at specified time steps in a gait cycle along the X-axis and the Z-axis respectively when the biped robot is going down stairs.

\[ X_{a_{SW,Dstr}}(t) = \begin{cases} 
  0 & t = 0 \\
  L_{an} \sin q_b + L_{af} (1 - \cos q_b) & t = T_d \\
  L_{ao} & t = T_m \\
  2D_s + L_{an} \sin q_f + L_{af} (1 - \cos q_f) & t = T_c \\
  2D_s & t = T_c + T_d 
\end{cases} \quad (5.24) \]

\[ Z_{a_{SW,Dstr}}(t) = \begin{cases} 
  L_{an} + H_{Str} & t = 0 \\
  L_{af} \sin q_b + L_{an} \cos q_b + H_{Str} & t = T_d \\
  H_{Str} & t = T_m \\
  L_{af} \sin q_f + L_{an} \cos q_f - H_{Str} & t = T_c \\
  L_{an} - H_{Str} & t = T_c + T_d 
\end{cases} \quad (5.25) \]

\[ \theta a_{SW,Dstr}(t) = \begin{cases} 
  0 & t = 0 \\
  q_b & t = T_d \\
  q_f & t = T_c \\
  0 & t = T_c + T_d 
\end{cases} \quad (5.26) \]
5.3.4. Sole of Foot Trajectory

For the biped robot simulation purposes, trajectory of foot sole is identified by motion patterns of three points. These points as illustrated in Figure 5.10 involve heel, toe, and the projected point of the ankle joint on foot which splits it into $L_{ab}$ and $L_{af}$. The formulation of both swing and stance feet motions are described in below.

\[
X_{h,Dstr}(t) = \begin{cases} 
  x_{ed} & t = 0 \\
  D_s - x_{sd} & t = T_d \\
  D_s + x_{ed} & t = T_c 
\end{cases} \quad (5.27)
\]

\[
Z_{h,Dstr}(t) = \begin{cases} 
  H_{min} & t = 0.5T_d \\
  H_{max} & t = 0.5(T_c - T_d) \\
  H_{min} - H_{str} & t = T_c + 0.5T_d 
\end{cases} \quad (5.28)
\]

![Figure 5.10 Three characteristic points of foot sole](image)

5.3.4.1. Sole of the Swing Foot Trajectory

Following sets of equations describe sole of the swing foot trajectory during walking on various surfaces.
\[
\begin{align*}
\begin{cases}
X_{2SW}(t) &= X_{SW}(t) - L_{an} \sin(\theta_{SW}(t)) \\
Z_{2SW}(t) &= Z_{SW}(t) - L_{an} \cos(\theta_{SW}(t)) \\
X_{1SW}(t) &= X_{2SW}(t) - L_{ab} \cos(\theta_{SW}(t)) \\
Z_{1SW}(t) &= Z_{2SW}(t) + L_{ab} \sin(\theta_{SW}(t)) \\
X_{3SW}(t) &= X_{2SW}(t) + L_{af} \cos(\theta_{SW}(t)) \\
Z_{3SW}(t) &= Z_{2SW}(t) - L_{af} \sin(\theta_{SW}(t))
\end{cases}
\end{align*}
\] (5.29)

5.3.4.2. Sole of the Stance Foot Trajectory

Following sets of equations describe sole of the support foot trajectory during walking on various surfaces.

5.3.4.2.1. Sole of the Stance Foot Trajectory for Flat Surface and Stairs

\[
\begin{align*}
\begin{cases}
X_1 &= X_a - L_{ab} \\
Z_1 &= Z_a - L_{an} \\
X_2 &= X_a \\
Z_2 &= Z_1 \\
X_3 &= X_a + L_{af} \\
Z_3 &= Z_1
\end{cases}
\end{align*}
\] (5.30)

5.3.4.2.2. Sole of the Stance Foot Trajectory for Inclined and Declined Surfaces

In the case of inclined surface \( \lambda \) has a negative value; in contrary, \( \lambda \) posses a positive value in declined ground condition. However, similar sets of equations are obtained for the stance foot coordination in both cases.
There is a growing interest in the applications of Artificial Neural Networks in different scopes of robotics and control systems. ANNs are able to model the non-linear relationships between several sets of model inputs and their equivalent outputs. ANNs are considered as data processing units developed based on construction of the biological brain neurons. ANNs are organized in several layers of interconnected processing nodes working together to transform the inputs to the outputs of the model. ANNs are appropriate for modeling complex systems whose parameters relationships are highly complicated to be identified.

A considerable research works have been dedicated to foot trajectories generated by polynomial and cubic spline interpolation methods. In existence of constraints such as walking surfaces conditions and different states of foot, the amount of computations is too high, particularly for online applications.

Two alternative approaches using Artificial Neural Networks based on Radial Basis Functions are proposed in this thesis as powerful methods to overcome the disadvantages of conventional interpolations techniques. According to the kinematics section

\[
\begin{align*}
X_2 &= X_a + L_{an} \sin \lambda \\
Z_2 &= Z_a - L_{an} \cos \lambda \\
X_1 &= X_2 - L_{ab} \cos \lambda \\
Z_1 &= Z_2 - L_{ab} \sin \lambda \\
X_3 &= X_2 - L_{af} \cos \lambda \\
Z_3 &= Z_2 - L_{af} \sin \lambda
\end{align*}
\]  

5.4. A Non-Conventional Methodology for Interpolation (or Generalization) Using ANN/RBFN
formulations, time (t) is assumed as the input variable for the swing ankle joint
\([X_{aSW}(t), Z_{aSW}(t), \theta_{aSW}(t)]\) and the hip joint \([X_{h}(t), Z_{h}(t)]\) characteristic parameters. When the number of outputs is always more than one, the number of inputs, ANNs return infinite mapping paths. Besides, an efficient error propagation cannot take place where the output layer neurons are more than the number of inputs. Due to these facts, desirable walking trajectory is not obtained through this ANN architecture.

In order to improve the generalization power of ANNs in such cases, new methods are proposed in this dissertation. Simulation of the implemented ANNs structure performance proves abilities of this tool in humanoid walking trajectories estimation. A Radial Basis Function Network has been utilized in this method which performs more fittingly on limited number of input/output data sets comparing with the Feed Forward Back Propagation Network. Following, an Artificial Neural Network is illustrated in terms of input/output sets.

![An Artificial Neural Network with input/output sets](image)

The output variables are functions of inputs of the system. The system equations are represented below:
\[ \begin{align*}
   y_1 &= f_1(x_1, x_2, x_3, \ldots, x_n) \\
   \vdots \\
   y_2 &= f_2(x_1, x_2, x_3, \ldots, x_n)
\end{align*} \]

\[ x \in \mathbb{R}^n, \quad y \in \mathbb{R}^m \quad (5.32) \]

When \( m > n \) there are infinite solutions for the system. In other words, the ANN is not able to properly find a unique solution for the system. If the number of outputs equals to the number of inputs \( (m = n) \) there is just one solution for the system. If \( m < n \), there may exist one approximate solution for the system. The proposed methodology given in following is aimed to find a desired unique solution by ANN/RBFN when there are more inputs in existence than outputs.

### 5.4.1. The Swing Ankle Joint Interpolated Trajectory

The ankle joint position in sagittal plane is identified by \([X_{SW}(t), Z_{SW}(t), \theta_{SW}(t)]\) where \(X_{SW}(t)\) and \(Z_{SW}(t)\) represent its horizontal and vertical positions respectively and \(\theta_{SW}(t)\) stands for sole of the swing foot angle at each time step. First step for setting up an ANN is to identify the training data set which includes specified times \( t \) as the input \(X_{SW}(t), Z_{SW}(t), \text{and } \theta_{SW}(t)\) as outputs.

![Figure 5.12 The ANN initial structure for swing ankle joint interpolation](image-url)
Therefore, to supply the ANN requirements for situation where number of inputs is greater than or equal to the number of outputs \( n \geq m \), the proposed methods are to either add adequate auxiliary input variables or implement a multi phase ANN structure.

In the first suggested method, additional inputs which could be several functions of outputs are intended to set the number of inputs to three. Thus, ANN can work efficiently as there exists a solution for the system \( m = n \). In implementation of this modified ANN, utilization of “dummy functions” may affect the approximation accuracy.

The novel Multi-Phase ANN Method is the focus of this thesis for the walking pattern interpolation. In this proposed technique, as illustrated in Figure 5.13, \( Xa_{SW}(t) \), \( Za_{SW}(t) \), and \( \theta a_{SW}(t) \) behaviors in time domain are obtained through three phases.

![Figure 5.13 The multi-phase ANN architecture for the swing ankle joint motion planning](image)

Each phase encompasses an Artificial Neural Network with \( m = n \) or \( n > m \). In the first phase, specified time steps set is the input which returns horizontal ankle joint positions pattern in the output. Second phase which are fed by the first ANN’s input and output sets interpolates vertical ankle joint behavior. And finally, the last phase with two inputs \( t \) and \( Za_{SW}(t) \) and single output \( \theta a_{SW}(t) \) is responsible for the sole of the swing foot angle calculation. All the ANNs utilize Radial Basis Function Networks in their
structure with limited available number of data sets. In the Radial Basis Function Network architecture including inputs, radial basis layer and output layer, number of neurons in the output layer equals to the number of the output variables which is one for all three phases. Since the number of neurons in the radial basis layer is determined automatically during the training process, it is not required to initially be specified. The radial basis function as the transfer function of the single hidden layer is also pre-specified for the Radial Basis Function Networks. Besides, the spread of radial basis function should be wide enough to let the radial basis neurons sufficiently overlap. The Levenberg-Marquardt algorithm is set as the learning method. Moreover, the network goal, the mean squared error performance function, can be adjusted according to the desired accuracy (i.e. $MSE = 10^{-5}$).

Once the network architecture design is accomplished, the training sets are fed into the Artificial Neural Networks. The numbers of the training sets for first, second and the last ANN are five, five, and four respectively. After the network is trained, the ankle joint position at any arbitrary time in the gait cycle is available.

So far, the ankle joint motion pattern is available through training a function based on limited number of sets of data where it can be assessed for smoothness, shape and being human-like motion. If the Radial Basis Function Network does not return a proper and satisfactory graph, all the steps would be retaken with new network characteristics so that a desirable pattern is obtained. Comparing with conventional interpolation methods where a non-adjustable result is returned, this can be counted as a valuable advantage of ANNs. It should also be pointed out that RBFN is more effectively dealing with generalization of multidimensional scattered data.
On the other hand, it might be questioned how the network is trained only on several data sets, four or five sets, and why a small number of data sets is sufficient to train the RBFN properly. The unique characteristics of this specific type of Artificial Neural Network along with utilizing a radial basis function, in this research Gaussian function, in the hidden layer neurons as the activation function ease the training in circumstances with small number of input/output data sets and answer the question.

5.4.1.1. Multi-Phases Modification for Real-Time Applications

Although the three-phase ANN structure generalizes the bipedal walking pattern similar to human being’s, it encompasses three Artificial Neural Networks. Whereas the number of training sets are quite small, the training process of networks is time consuming in on-line applications. Particularly in conditions of various combined surfaces where walking trajectory required to be updated frequently, three-network implementation might perceptibly delay the robot performance. In the proposed method which is more suitable for real-time applications, as illustrated in Figure 5.14, a decrement in the number of ANNs is observed. This suggested technique might reduce the naturalness of the walking pattern; but, it is significantly more time effective.

![Figure 5.14 Two-phases ANN structure for real-time applications](image)
In the Figure 5.14 topology, since the number of available training set for $\theta a_{sw}(t)$ is the least and it is intended to accommodate two outputs by the second network, the $\theta a_{sw}(t)$ pattern is first generalized.

5.4.2. The Hip Joint Trajectory Planning

For the hip joint generalization, since there are two outputs, a two-phase ANN structure will return desirable hip trajectory. As indicated in Figure 5.14, in the first ANN, specified set of time steps is the input set and the horizontal ankle joint positions ($X_h$) stands for the output set. Second phase which are fed by the first ANN’s input and output is aimed to generalize vertical ankle joint behavior ($Z_h$).

![Figure 5.15 The multi-phase ANN architecture for the hip joint motion planning](image)

5.5. The Biped Robot Dynamic Investigation

The biped robot dynamic is defined as the relationships between applied forces and the resulted accelerations. In this section, a balance maintenance method applicable to real-time biped robot’s tasks is introduced. Despite complicated Zero Moment Point formulations such as Eq. 2.2 and Eq. 2.3, a simplified model (LIPM) based on human body’s physics and structure for controlling of ZMP is proposed. The technique for the
robot stability locates robot’s trunk such that the ZMP is kept within the support area during both single and double support phases.

The concept of inverted pendulum with a massless rod modeling is based on the fact that 60% to 70% of a human body mass is concentrated at its trunk. Therefore, the whole robot structure is replaced with a single rigid body located at the COM with a mass of the robot’s. In addition, the desired ZMP is pre-specified for the most stable situation placed at center of the polygon of support where it has the maximum distances from the support area margins. Therefore, the position of the robot’s trunk is obtained for each single gait step through several equations as discussed below. Prior proceeding to the next gait step, ZMP position is investigated with respect to the new polygon of support. If ZMP is still within the support area, the previous trunk position is kept; otherwise, all the procedures to obtain new trunk location is repeated under new constraints.

5.5.1. The Linear Inverted Pendulum Model

For real-time walking dynamic analysis of a biped robot, an inverted pendulum motions on sagittal plane are considered. Instead of employing precise dynamic information of the robot such as mass, COM, and inertia of every single linkage, the proposed method utilizes limited amount of information of the robot’s dynamic specifications such as COM of the whole body and total angular momentum.
In above illustration, \( m_i \) stands for mass of \( i^{th} \) linkage and \( M \) represents the total center of mass. As indicated in Figure 5.16, the yellow point where the support foot is connected to the ground is the end point of the massless rod on the walking surface. In addition, the blue crossed circles on linkages represent each link’s center of mass. The seven links’ COM position along X-axis is calculated through below equations.

\[
\text{Trunk: } \frac{x_T(t) + x_h(t)}{2} \quad (5.33)
\]

\[
\text{Thigh: } \frac{x_h(t) + x_{kSW}(t)}{2} \\
\text{Swing leg: } \frac{x_{asw}(t) + x_{kSW}(t)}{2} \quad (5.34)
\]

\[
\text{Shank: } \frac{x_{asw}(t) + x_{kSW}(t)}{2} \\
\text{Foot: } x_{2,SW}(t)
\]
Due to the symmetry of robot’s body, following constraints are identified:

\[
\begin{align*}
    m_2 &= m_3 \\
    m_4 &= m_5 \\
    m_6 &= m_7
\end{align*}
\]  

(5.36)

Therefore, following formulation for the total center of mass is obtained from Eq. 5.5 and Eq. 2.1 in term of trunk position along the X-axis \( (X_T) \). In other words, the only unknown variable is \( X_T \).

\[
x_{\text{COM}}(t) = \frac{m_1}{m_1 + 2(m_2 + m_4 + m_6)} (X_T(t) + X_h(t)) + \frac{m_2}{m_1 + 2(m_2 + m_4 + m_6)} (X_h(t) + X_k(t) + X_k(t) + X_k(t)) + \frac{m_4}{m_1 + 2(m_2 + m_4 + m_6)} (X_{SW}(t) + X_{SW}(t) + X_{SW}(t) + X_{SW}(t) + X_{SW}(t) + X_{SW}(t))
\]

(5.37)

In this dissertation, the ZMP principles are taken into consideration and stable walking patterns for the biped robot are systematically determined. Zero Moment Point specifies a point on the ground \( (Z_{\text{ZMP}} = 0) \) where the net moment of the inertial and the gravity forces has no component along the X-axes. The dynamics and the concept of ZMP are applied on the inverted pendulum through Eq. 5.38 and Eq. 5.39 respectively [13].
\[ \tau = -Mg(x_{\text{COM}} - p_x) + M\ddot{x}_{\text{COM}}z_{\text{COM}} \quad (5.38) \]

\[ \tau = 0 \quad (5.39) \]

Where \( M \) represents the mass of the pendulum which equals to the robot’s mass, \( g \) is gravity acceleration and \( \tau \) stands for the torque. \( p_x \) is the ZMP along X-axis. By substituting Eq. 5.39 into the Eq. 5.38 we obtain:

\[ p_x = x_{\text{COM}} - \frac{x_{\text{COM}}}{g}z_{\text{COM}} \quad (5.40) \]

### 5.5.2. Stability Margins and Desirable ZMP

As described earlier, stability margins along X-axis are defined as the area between the two feet for the double support phase and the space under the contact foot when just one foot touches the ground.

\[
\begin{cases}
X_{\text{POS}}^{\text{Low}} = X_{1,\text{SW}}(t) \\
X_{\text{POS}}^{\text{Up}} = X_3 \\
X_{\text{POS}}^{\text{Low}} = X_{3,\text{SW}}(t) \\
X_{\text{POS}}^{\text{Up}} = X_3 \\
X_{\text{POS}}^{\text{Low}} = X_1 \\
X_{\text{POS}}^{\text{Up}} = X_3 \\
X_{\text{POS}}^{\text{Low}} = X_1 \\
X_{\text{POS}}^{\text{Up}} = X_{1,\text{SW}} \\
\end{cases}
\quad \begin{aligned}
t & = 0 \\
0 < t \leq T_d \\
T_d < t < T_c \\
t = T_c
\end{aligned}
\]

(5.41)
In all identified intervals, desired ZMP is obtained through average of the lower and upper boundary points of each interval.

5.5.3. The Robot’s Trunk Trajectory

By replacing the total center of mass calculated in Eq. 5.37 with its equivalent term in Eq. 5.40, ZMP is expressed in the term of trunk position.

$$X_{ZMP} = x_{COM}(X_T) - \frac{x_{COM}}{g} Z_{COM}$$  \hspace{1cm} (5.42)

At the first gait step, the ZMP along the X-axis is set as the middle point of the specified stability interval. For the next gait steps, if the pre-specified ZMP is still within the new support area, the previous trunk position is kept; otherwise, computation of new trunk location is repeated under new constraints. The following flowchart illustrates the steps of achieving the robot’s upper body pattern of motion.

![Figure 5.17 The trunk trajectory planning algorithm](image-url)
5.6. Chapter Summary

In this chapter, an adaptive walking pattern planning for a biped robot was proposed. Through this methodology, robot’s body at key frames were synthesized in order to obtain various joints characteristics. Then by applying the derived data into Artificial Neural Networks utilizing Radial Basis Functions, the robot trajectory was generalized. Due to the fact that ANNs are not capable of appropriate generalization of networks with more number of outputs than the number of inputs, the process of interpolation was accomplished in several phases. Each phase encompassed an ANN/RBFN such that the number of outputs was not greater than the number of inputs. When the lower body trajectory with respect to the ground condition was planned, the trunk trajectory could be identified such that it guaranteed the dynamic stability of the robot during walking cycles.
Chapter 6
RESULTS AND DISCUSSION

6.1. Introduction

The humanoid robot performance simulation prior to the physical implementation is a critical part of its advancement. In this chapter, the correlation between the robot’s joints specifications and walking patterns is demonstrated through the simulation developed in MATLAB. The simulation models for bipedal walking based on the gait generation confirm the validity of the proposed methodology.

6.2. Specification of Constant Parameters

The percentages of links concentrated masses and their respective quantities utilized in simulations are indicated in Table 6.1 [43]. It is assumed that the robot total mass is 82 Kg including two hands and head.

Table 6.1 links concentrated masses

<table>
<thead>
<tr>
<th>Upper body</th>
<th>Trunk (m$_1$)</th>
<th>43.02</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lower body</td>
<td>Thigh (m$_2$,m$_3$)</td>
<td>14.47</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Shank (m$_4$,m$_5$)</td>
<td>4.57</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Foot (m$_6$,m$_7$)</td>
<td>1.33</td>
<td>2</td>
</tr>
</tbody>
</table>
Length of the seven linkages are shown in Table 6.2. The robot’s body physical characteristics expressed in the below table are obtained through measurements of a human body attributes.

Table 6.2 Constant parameters

<table>
<thead>
<tr>
<th></th>
<th>Length (cm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upper body</td>
<td></td>
</tr>
<tr>
<td>Trunk Length</td>
<td>55</td>
</tr>
<tr>
<td>Lower body</td>
<td></td>
</tr>
<tr>
<td>Thigh Length</td>
<td>47</td>
</tr>
<tr>
<td>Shank Length</td>
<td>42</td>
</tr>
<tr>
<td>Foot specifications</td>
<td></td>
</tr>
<tr>
<td>L_an</td>
<td>10</td>
</tr>
<tr>
<td>L_ab</td>
<td>7</td>
</tr>
<tr>
<td>L_af</td>
<td>15</td>
</tr>
</tbody>
</table>

6.3. Simulation of Walking Pattern for an Ideal Ground Condition

Following figure depicts the robot’s body postures during two gait cycles with a fixed gait step D_s = 70 cm on a flat surface. The biped robot’s gait specifications in Figure 6.1 illustration are listed in table 6.3.
Figure 6.1 The biped robot walking trajectory on flat ground condition

<table>
<thead>
<tr>
<th>$D_S$</th>
<th>$q_b$</th>
<th>$q_f$</th>
<th>$L_{ao}$</th>
<th>$H_{ao}$</th>
<th>$x_{sd}$</th>
<th>$x_{ed}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>70 cm</td>
<td>$\frac{\pi}{12}$</td>
<td>$\frac{\pi}{12}$</td>
<td>30 cm</td>
<td>30 cm</td>
<td>35 cm</td>
<td>32 cm</td>
</tr>
</tbody>
</table>

Figure 6.2 represents a closer view of robot walking during a single gait with applied dynamical considerations.
Following figures illustrate motion pattern of joints at a gait cycle.

Figure 6.3 Ankle joint motion pattern

Figure 6.4 Sole of the swing foot angle

Figure 6.5 Hip joint trajectory in XZ-plane in one gait cycle
Figure 6.6 The swing leg knee joint in sagittal plane during a single gait period

Figure 6.7 The stance leg knee joint in sagittal plane during a single gait period

Figure 6.8 illustrates the support polygon in one gait cycle where in Single Support Phase has a constant area due to the fact that in this phase only the support foot is in complete contact with the surface.

Figure 6.8 Support area in first gait cycle
Below table gives the RBFNs specifications and the outputs’ error of each network.

<table>
<thead>
<tr>
<th>Neurons</th>
<th>MSE</th>
<th>Neurons</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ankle pattern along X-axis</td>
<td>0</td>
<td>0.401446</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.00587192</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.00287179</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>7.16926e-29</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>1.92453e-29</td>
<td>5</td>
</tr>
<tr>
<td>Hip pattern along X-axis</td>
<td>0</td>
<td>0.104422</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2.97877e-32</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1.33531e-32</td>
<td>3</td>
</tr>
<tr>
<td>Sole of the swing foot ankle</td>
<td>0</td>
<td>0.0342695</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.0212528</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1.8643e-31</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>7.78076e-31</td>
<td></td>
</tr>
</tbody>
</table>
The walking pattern simulation has also been developed with spline interpolation method. Following figure depicts stick diagram of a biped robot utilizing a conventional interpolation method.

![Figure 6.9 The biped robot walking trajectory on flat ground condition obtained by spline interpolation](image)

Above illustration represents the foot’s angle behavior obtained through RBFN during a single gait cycle, left hand side graph, and its counterpart with similar applied specifications resulted from spline interpolation methodology, right hand side graph. At first glimpse, it is revealed that RBFN returns a smoother graph which is more desirable for mimicking human walking; however, RBFN flexibility in pattern modifications on same data sets is a more significant advantage.
6.4. **Simulation of Walking Pattern on Sloped Ground Conditions**

Ground conditions with positive or negative slope are of very often situations that a robot performing in a human-acting environment faces with.

6.4.1. **Walking Down Hill**

Following figure depicts the robot’s motion trajectory in three gait cycles with a fixed gait step $D_S = 40 \text{ cm}$ when it is walking down hill. The biped robot’s gait specifications in Figure 6.9 illustration are listed in table 6.4.

![Figure 6.11 The biped robot walking trajectory on downhill ground condition](image)

Table 6.5 The robot’s gait specifications for negatively sloped ground condition

<table>
<thead>
<tr>
<th>$D_S$</th>
<th>$q_b$</th>
<th>$\lambda$</th>
<th>$q_f$</th>
<th>$L_{ao}$</th>
<th>$H_{ao}$</th>
<th>$x_{sd}$</th>
<th>$x_{ed}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>40 cm</td>
<td>$\frac{\pi}{12}$</td>
<td>$\frac{\pi}{10}$</td>
<td>$\frac{\pi}{12}$</td>
<td>30 cm</td>
<td>30 cm</td>
<td>15 cm</td>
<td>15 cm</td>
</tr>
</tbody>
</table>
Following figures illustrate motion pattern of joints in a gait cycle interval.

![Figure 6.12 Ankle joint motion pattern on declined ground condition](image1)

![Figure 6.13 Sole of the swing foot angle](image2)

![Figure 6.14 The hip joint in sagittal plane during a single gait period](image3)
Figure 6.15 The stance leg knee joint pattern in sagittal plane during a single gait period

Below table gives the RBFNs specifications and the outputs’ error of each network.

<table>
<thead>
<tr>
<th></th>
<th>neurons</th>
<th>MSE</th>
<th></th>
<th>neurons</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ankle pattern along X-axis</td>
<td>0</td>
<td>0.0848248</td>
<td>0</td>
<td>0.0306934</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.00188982</td>
<td>2</td>
<td>0.0016419</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1.51372e-06</td>
<td>3</td>
<td>0.00141331</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>4.37128e-30</td>
<td>4</td>
<td>1.09182e-28</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>8.27368e-30</td>
<td>5</td>
<td>1.52979e-29</td>
<td></td>
</tr>
<tr>
<td>Hip pattern along X-axis</td>
<td>0</td>
<td>0.0267988</td>
<td>0</td>
<td>0.00287066</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2.56791e-34</td>
<td>2</td>
<td>5.99863e-31</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2.56791e-33</td>
<td>3</td>
<td>8.48847e-30</td>
<td></td>
</tr>
<tr>
<td>Sole of the swing foot ankle</td>
<td>neurons</td>
<td>MSE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------</td>
<td>--------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0.0342695</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.0178725</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1.01766e-30</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2.68036e-29</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.16 Stick diagram of biped robot walking trajectory on downhill ground condition obtained by spline interpolation

6.4.2. Walking Up Hill

Following figure depicts the robot’s motion trajectory in three gait cycles with a fixed gait step $D_S = 50$ cm when it is walking up hill. The biped robot’s gait specifications in Figure 6.14 illustration are listed in table 6.7.
Figure 6.17 The biped robot walking trajectory on uphill ground condition

Table 6.7 The robot’s gait specifications for positively sloped ground condition

<table>
<thead>
<tr>
<th>$D_s$</th>
<th>$q_p$</th>
<th>$\lambda$</th>
<th>$q_f$</th>
<th>$L_{ao}$</th>
<th>$H_{ao}$</th>
<th>$x_{sd}$</th>
<th>$x_{ed}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 cm</td>
<td>$\frac{\pi}{12}$</td>
<td>$\frac{\pi}{10}$</td>
<td>$\frac{\pi}{12}$</td>
<td>30 cm</td>
<td>30 cm</td>
<td>25 cm</td>
<td>22 cm</td>
</tr>
</tbody>
</table>

Following figure illustrates stick diagram of motion pattern over more cycles.

Figure 6.18 Stick diagram of walking up hill
Below table gives the RBFNs specifications and the outputs’ error of each network.

Table 6.8 RBFNs specifications for inclined surface

<table>
<thead>
<tr>
<th></th>
<th>neurons</th>
<th>MSE</th>
<th></th>
<th>neurons</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ankle pattern</td>
<td>0</td>
<td>0.151986</td>
<td>Ankle pattern</td>
<td>0</td>
<td>0.0164403</td>
</tr>
<tr>
<td>along X-axis</td>
<td>2</td>
<td>0.00160429</td>
<td>along Z-axis</td>
<td>2</td>
<td>0.00166412</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.00122171</td>
<td></td>
<td>3</td>
<td>0.00108518</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5.96387e-29</td>
<td></td>
<td>4</td>
<td>4.52316e-30</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>7.44568e-30</td>
<td></td>
<td>5</td>
<td>1.10641e-30</td>
</tr>
<tr>
<td>Hip pattern</td>
<td>0</td>
<td>0.0496731</td>
<td>Hip pattern</td>
<td>0</td>
<td>0.00055555</td>
</tr>
<tr>
<td>along X-axis</td>
<td>2</td>
<td>1.28395e-33</td>
<td>along Z-axis</td>
<td>2</td>
<td>2.39945e-30</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3.82618e-32</td>
<td></td>
<td>3</td>
<td>6.60671e-30</td>
</tr>
<tr>
<td>Sole of the</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>swing foot</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ankle</td>
<td>0</td>
<td>0.0342695</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.0202608</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5.73927e-31</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>7.47646e-30</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
6.5. Simulation of Bipedal Walking Pattern on Stairs

Walking up and down stairs are of often situations that a robot performing in a human oriented environment faces with.

6.5.1. Walking Down Stairs

Following figure depicts the robot’s motion trajectory in five gait cycles with a fixed gait step $D_S = 70$ cm when it is walking down stairs. The biped robot’s gait specifications in Figure 6.16 illustration are listed in table 6.9.

![Figure 6.19 The biped robot walking down stairs trajectory](image1)

Following figure depicts the robot’s motion trajectory in four gait cycles with a smaller gait step $D_S = 50$ cm when it is walking down stairs.

![Figure 6.20 The biped robot walking down stairs trajectory with a smaller step size](image2)
Robot’s motion pattern when it is walking down stairs is a bit different from walking on other ground conditions. As it is represented in above figure, robot’s toe touches the surface before its heel at the end of a gait cycle.

Table 6.9 The robot’s gait specifications for walking down stairs condition

<table>
<thead>
<tr>
<th>$D_s$</th>
<th>$q_p$</th>
<th>$q_f$</th>
<th>$L_{ao}$</th>
<th>$H_{ao}$</th>
<th>$x_{sd}$</th>
<th>$x_{ed}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 cm</td>
<td>$\frac{\pi}{12}$</td>
<td>$\frac{\pi}{12}$</td>
<td>30 cm</td>
<td>30 cm</td>
<td>25 cm</td>
<td>22 cm</td>
</tr>
</tbody>
</table>

Following figures illustrate motion pattern of joints in a gait cycle interval.

Figure 6.21 Ankle motion pattern in one gait cycle

Figure 6.22 Hip joint trajectory during a gait cycle
Below table gives the RBFNs specifications and the outputs’ error of each network.

<table>
<thead>
<tr>
<th>neurons</th>
<th>MSE</th>
<th>neurons</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.408118</td>
<td>0</td>
<td>0.0424351</td>
</tr>
<tr>
<td>2</td>
<td>0.00785892</td>
<td>2</td>
<td>0.00023497</td>
</tr>
<tr>
<td>3</td>
<td>3.23159e-05</td>
<td>3</td>
<td>4.71206e-09</td>
</tr>
<tr>
<td>4</td>
<td>1.18026e-29</td>
<td>4</td>
<td>1.92593e-34</td>
</tr>
<tr>
<td>5</td>
<td>7.9159e-29</td>
<td>5</td>
<td>1.19022e-32</td>
</tr>
<tr>
<td>0</td>
<td>0.0838889</td>
<td>0</td>
<td>0.0116667</td>
</tr>
<tr>
<td>2</td>
<td>1.74618e-32</td>
<td>2</td>
<td>1.23342e-28</td>
</tr>
<tr>
<td>3</td>
<td>2.56791e-32</td>
<td>3</td>
<td>2.05383e-28</td>
</tr>
<tr>
<td>Sole of the swing foot ankle</td>
<td>neurons</td>
<td>MSE</td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0.0171347</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.00685892</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>8.42017e-30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1.64089e-30</td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.24 The stick diagram of walking down stairs

6.4.2. Walking Up Stairs

Following figure depicts the robot’s motion trajectory in two gait cycles with a fixed gait step $D_S = 70$ cm when it is walking up stairs. The biped robot’s gait specifications in Figure 6.21 illustration are listed in table 6.11.
Figure 6.25 The biped robot’s walking up stairs trajectory

Table 6.11 The robot’s gait specifications for walking up stairs

<table>
<thead>
<tr>
<th>$D_s$</th>
<th>$q_b$</th>
<th>$q_f$</th>
<th>$L_{ao}$</th>
<th>$H_{st}$</th>
<th>$x_{sd}$</th>
<th>$x_{ed}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>70 cm</td>
<td>$\frac{\pi}{12}$</td>
<td>$\frac{\pi}{12}$</td>
<td>35 cm</td>
<td>10 cm</td>
<td>25 cm</td>
<td>20 cm</td>
</tr>
</tbody>
</table>

Below table gives the RBFNs specifications and the outputs’ error of each network.

Table 6.12 RBFNs specifications for walking up stairs

<table>
<thead>
<tr>
<th>Neurons</th>
<th>MSE</th>
<th>Neurons</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ankle pattern along X-axis</td>
<td></td>
<td>Hip pattern along X-axis</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0.395435</td>
<td>0</td>
<td>0.0085644</td>
</tr>
<tr>
<td>2</td>
<td>0.00543438</td>
<td>2</td>
<td>0.00121301</td>
</tr>
<tr>
<td>3</td>
<td>0.00360959</td>
<td>3</td>
<td>0.00013269</td>
</tr>
<tr>
<td>4</td>
<td>7.09302e-29</td>
<td>4</td>
<td>1.92593e-32</td>
</tr>
<tr>
<td>5</td>
<td>1.29708e-28</td>
<td>5</td>
<td>3.34187e-31</td>
</tr>
<tr>
<td>Ankle pattern along Z-axis</td>
<td></td>
<td>Hip pattern along Z-axis</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0.0838889</td>
<td>0</td>
<td>0.00055555</td>
</tr>
<tr>
<td>2</td>
<td>1.74618e-32</td>
<td>2</td>
<td>8.18443e-30</td>
</tr>
<tr>
<td>3</td>
<td>2.56791e-32</td>
<td>3</td>
<td>3.97717e-30</td>
</tr>
</tbody>
</table>
The below figure represents the network training performance of the hip motion along Z-axis. As represented the desired error rate is zero; however, the network lowest overall Mean Square Error is $3.977 \times 10^{-30}$ with three epochs.

![Figure 6.26 RBFN performance diagram](image)
Chapter 7

CONCLUSION

7.1. Summary

The under study biped robot consists of rigid bodies connected with actuated joints supposed to mimic human walking pattern on various ground conditions. Bipedal walking is a periodic pattern of Single Support Phase followed by Double Support Phase. A periodic approach which facilitates a scalable gait with characteristic parameters is proposed. The proposed methodology is divided in two parts, planning robot trajectory and dynamic stability investigation. The bipedal walking trajectory which is adaptive to the ground condition is computed by limited numbers of breakpoints in both stable and unstable phases. Consequently, positions of ankle, hip, and knee joints are derived for a seven link biped robot. In order to satisfy the smoothness of walking pattern as well as efficiency of the computation algorithm, Artificial Neural Networks using Radial Basis Function to generalize a curve on derived key points is implemented.

Dynamic stability of the robot is simplified through the Zero Moment Point criterion. In the dynamic stability investigation part, ZMP for the most stable condition in a determined polygon of support is calculated. Then, to update the trunk motion such that compensate for lower limb movement, Linear Inverted Pendulum model and ZMP criterion are employed to attain upper body trajectory satisfying whole robot walking dynamic stability.
7.2. Contributions

The main contributions of this thesis are:

• A non-conventional methodology for efficient computation of robot walking trajectory applicable to on-line tasks.
• An adaptive human-like bipedal walking pattern planning to various ground conditions.
• A novel technique based on RBFNs to enhance performance of conventional numerical methods.
• A simplified and efficient dynamical walking stability investigations.

7.3. Future work

The proposed methodology is applicable to sagittal plane where the robot only walks in a specified direction. In order to develop an Omni-directional walking pattern, the key challenge is the transition state where the value of “Y” changes.
REFERENCES


APPENDICES

MATLAB CODE

A. Walking on Flat Surface

clear all;
clc

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%----------Biped Robot Walk on a flat surface----------%
%----------------Author: Mohammadreza Ranjbar----------------%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Tc = 1;     % Single gait period
Td = 0.2*Tc;        % Double-Support phase time interval
Tm = 0.4;        % At Tm ankle joint is at its highest point
Time_Stepsize = 0.1;
Ds = 0.7;       % Gait Length
Xa = [ ];        % x of Ankle joint
Lan = 0.1;      % Ankle joint height
qb = pi/12;     % Foot leaving angle
Lao = 0.3;      % Ankle position on X axis when t = iTc + Tm
Hao = 0.3;      % Ankle maximum height
qf = pi/12;      % Foot landing angle
Laf = 0.15;      % Ankle to toe length
Lab = 0.07;     % Heel to ankle length
Xsd = 0.35;
Xed = 0.32;       %***********************************************
Thigh = 0.47;       % Thigh length
Shank = 0.42;       % Shank length
Hh_max = Thigh + Shank + Lan/2;     % Maximum hip height
Hh_min = Hh_max - 0.05;       % Minimum hip height
g = 9.8;       % Gravity acceleration
weight = 82;        % RobotÆs total weight
m1 = 0.4302*weight;        % Trunk weight including head and hands
weight
m2 = 0.1447*weight;        % Thigh weight
m3 = 0.0457*weight;        % Shank weight
m4 = 0.0133*weight;        % Foot weight
M = m1 + 2*(m2 + m3 + m4);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Kinematic of the swing leg-%%%%%
%%----------------Ankle joint----------------%%%%
%First phase input: time, output: Xa

t(1) = 0;
t(2) = Td;
t(3) = Tm;
t(4) = Tc;
t(5) = Tc +Td;

Xa(1) = 0;
Xa(2) = Lan*sin(qb) + Laf*(1 - cos(qb));
Xa(3) = Lao;
Xa(4) = 2*Ds - Lan*sin(qf) - Lab*(1 - cos(qf));
Xa(5) = 2*Ds;

net = newrb(t,Xa);

t1 = 0:Time_Stepsize:Tc
Xa_sim = sim(net,t1);
%conventional interpolation method
%Spline(t,Xa);
%Second phase inputs: time, Xa, output: Za

Za(1) = Lan;
Za(2) = Laf*sin(qb) + Lan*cos(qb);
Za(3) = Hao;
Za(4) = Lab*sin(qf) + Lan*cos(qf);
Za(5) = Lan;

input(1,:) = t;
input(2,:) = Xa;

net1 = newrb(input,Za);

input1(1,:) = t1
input1(2,:) = Xa_sim;
Za_sim = sim(net1,input1);
%figure(1)
%plot(Xa_sim,Za_sim) %graph Za vs Xa

%conventional interpolation method
%Spline(t,Za);

%Third phase inputs: time, Xa, Za, output: Theta_a
Theta_a(1) = 0;
Theta_a(2) = qb;
Theta_a(3) = -qf;
Theta_a(4) = 0;

input2(1,:) = t;
input2(2,:) = Xa;
%input2(3,:) = Za;
input2(:,2) = [];

net2 = newrb(input2,Theta_a);

input3(1,:) = t1
input3(2,:) = Xa_sim;
%input3(3,:) = Za_sim;
Theta_a_sim = sim(net2,input3);
%figure(3)
%plot(input3(1,:),Theta_a_sim)
%conventional interpolation method
%Spline(t,Theta_a);

%------------------------Hip joint------------------------

%First phase input: time, output: Xh
input4 = t;
input4(3) = [];  
input4(4) = [];

Xh(1) = Xed;
Xh(2) = Ds - Xsd;
Xh(3) = Ds + Xed;

net3 = newrb(input4,Xh);

Xh_sim = sim(net3,t1);

%Second phase input: time, output: Zh
input4(1,:) = [0.5*Td 0.5*(Tc-Td) Tc-0.5*Td];
input4(2,:) = sim(net3,input4(1,:));
Zh = [Hh_min Hh_max Hh_min]

net4 = newrb(input4,Zh,0,10);

Zh_sim = sim(net4,t1);

%figure(3)
%plot(Xh_sim,Zh_sim)

%------------------------Knee joint------------------------

% intersection of two circles
cnst = Thigh^2-Shank^2
[size0,size_of_t1] = size(t1);

Xk_sim(1,:) = 0;
Zk_sim(1,:) = 0;
for i=1:size_of_t1
    p = (cnst-(Xh_sim(i))^2+(Xa_sim(i))^2-
    (Zh_sim(i))^2+(Za_sim(i))^2)/...
    (2*(Za_sim(i)-Zh_sim(i))));
    q = ( Xa_sim(i) - Xh_sim(i) )/(Za_sim(i) - Zh_sim(i));
    a = 1 + q^2;
    b = 2*(q*Zh_sim(i) - p*q - Xh_sim(i));
    c = (Xh_sim(i))^2 + (Zh_sim(i))^2 + p^2 - 2*(p*Zh_sim(i)) - Thigh^2;

    Root1 = (-b - sqrt(b^2 - 4*a*c))/ (2*a);
    Root2 = (-b + sqrt(b^2 - 4*a*c))/ (2*a);
    if Root1 > Root2
        Xk_sim(i) = Root1;
    else Xk_sim(i) = Root2;
end
Zk_sim(i) = p - q*Xk_sim(i);
end

%figure(4)
%plot(Xk_sim,Zk_sim)
%figure(1)

%%%-------------------Kinematic of the support leg-------------------

%%%%%------------------Ankle joint-------------------

Xa_suprt = Ds;
Za_suprt = Lan;

%%%%%------------------Hip joint-------------------
%there is just one hip joint
%Hip joint for both support and swing leg is same
Xh_suprt = Xh_sim;
Zh_suprt = Zh_sim;

%%%%%------------------Knee joint-------------------

% intersection of two circles
Xk_suprt(1,:) = 0;
Zk_suprt(1,:) = 0;
for i=1:size_of_t1

p = (cnst-(Xh_suprt(i))^2+(Xa_suprt)^2-(Zh_suprt(i))^2+...+(Za_suprt)^2)/(2*(Za_suprt-Zh_suprt(i)))
q = ( Xa_suprt - Xh_suprt(i) )/(Za_suprt - Zh_suprt(i));
a = 1 + q^2;
b = 2*(q*Zh_suprt(i) - p*q - Xh_suprt(i));
c = (Xh_suprt(i))^2 + (Zh_suprt(i))^2 + p^2 -...
   2*(p*Zh_suprt(i)) - Thigh^2;
Root1 = (-b - sqrt(b^2 - 4*a*c))/ (2*a);
Root2 = (-b + sqrt(b^2 - 4*a*c))/ (2*a);

if Root1 > Root2
    Xk_suprt(i) = Root1;
else Xk_suprt(i) = Root2;
end
Zk_suprt(i) = p - q*Xk_suprt(i);
end

%plot(Xk_suprt,Zk_suprt)

%%%%%------------------Sole foot-------------------

% Heel position
X1_suprt = Xa_suprt - Lab;
Z1_suprt = Za_suprt - Lan;
%Projected ankle on sole foot position
X2_suprt = Xa_suprt;
Z2_suprt = Z1_suprt;
%Toe position
X3_suprt = Xa_suprt + Laf;
Z3_suprt = Z1_suprt;
X13_suprt = [X1_suprt X3_suprt];
Z13_suprt = [Z1_suprt Z3_suprt];

% initilizing some parameters
figure(2)
Vi = 0;
for j = 0:0
    for i = 1:size_of_t1
        %clf
        time = (i-1)*Time_Stepsize;  % passed time in a cycle
        %Projected ankle on sole foot position for swing leg
        X2 = Xa_sim(i) - Lan*sin(Theta_a_sim(i)) + j*Ds;
        Z2 = Za_sim(i) - Lan*cos(Theta_a_sim(i));
        % Heel position for swing leg
        X1 = X2 - Lab*cos(Theta_a_sim(i));
        Z1 = Z2 + Lab*sin(Theta_a_sim(i));
        %Toe position for swing leg
        X3 = X2 + Laf*cos(Theta_a_sim(i));
        Z3 = Z2 - Laf*sin(Theta_a_sim(i));
        if time <= Td
            Xzmp_up(i) = X3_suprt + j*Ds;
            if time == 0
                Xzmp_low(i) = X1;
            else
                Xzmp_low(i) = X3;
            end
        end
        if time > Td && time < Tc
            Xzmp_up(i) = X3_suprt + j*Ds;
            Xzmp_low(i) = X1_suprt + j*Ds;
        end
        if time >= Tc
            Xzmp_up(i) = X1;
            Xzmp_low(i) = X1_suprt + j*Ds;
        end

        % Trunk position boundaries
        Xt_low(i) = Xh_sim(i) + j*Ds - 0.55*sin(pi/12);
        Xt_up(i) = Xh_sim(i) + j*Ds + 0.55*sin(pi/9);

        % CoM when the trunk angle is -15
        X1_CoM(i) = ((m1+2*m2)*(Xh_sim(i) + j*Ds) + (m2+m3)*(Xk_sim(i)
                    + j*Ds) + m3*(Xa_sim(i) + j*Ds)
                    + Xa_suprt + j*Ds) + 2*m4*(X2_suprt + j*Ds + X2) + ...
                    m1*Xt_low(i)/(2*M);

        % CoM when the trunk angle is 20
        X2_CoM(i) = ((m1+2*m2)*(Xh_sim(i) + j*Ds) + (m2+m3)*(Xk_sim(i)
                    + j*Ds) + m3*(Xa_sim(i) + j*Ds)
                    + Xa_suprt + j*Ds) + 2*m4*(X2_suprt + j*Ds + X2) + ...
\[
m_1 \times \frac{X_{\text{t up}}(i)}{(2 \times M)};
\]

% \textbf{Xf}: Final position along X axis in one gait step
% \textbf{Xi}: Initial position along X axis in one gait step
% \textbf{Vf}: Final velocity along X axis in one gait step
% \textbf{Vi}: Initial velocity along X axis in one gait step

\[
X_f = X_{h \_sim}(i) + j \times Ds;
\]
if \(i == 1\)
\[
Xi = X_f;
\]
else
\[
Xi = X_{h \_sim}(i-1) + j \times Ds;
\]
end
\[
V_f = 2 \times (X_f - Xi)/\text{Time\_Stepsize} - Vi;
\]
\% Acceleration at each gait step
\[
\text{acc} = (V_f - Vi)/\text{Time\_Stepsize};
\]
\[
X_{1\_ZMP}(i) = X_{1\_CoM}(i) - (Zh_{sim}(i)/g) \times \text{acc} + j \times Ds;
\]
\[
X_{2\_ZMP}(i) = X_{2\_CoM}(i) - (Zh_{sim}(i)/g) \times \text{acc} + j \times Ds;
\]
\% Calculating ZMP at each gait step
if \(X_{2\_ZMP}(i) \leq X_{zmp\_up}(i) \&\& X_{2\_ZMP}(i) > X_{zmp\_low}(i)\)
if \(X_{1\_ZMP}(i) > X_{zmp\_low}(i)\)
\[
X_{zmp}(i) = (X_{2\_ZMP}(i) + X_{1\_ZMP}(i))/2;
\]
else
\[
X_{zmp}(i) = (X_{2\_ZMP}(i) + X_{zmp\_low}(i))/2;
\]
end
end
if \(X_{2\_ZMP}(i) > X_{zmp\_up}(i)\)
if \(X_{1\_ZMP}(i) < X_{zmp\_up}(i)\)
\[
X_{zmp}(i) = (X_{1\_ZMP}(i) + X_{zmp\_up}(i))/2;
\]
else
\[
X_{zmp}(i) = X_{1\_ZMP}(i);
\]
end
end
if \(X_{2\_ZMP}(i) < X_{zmp\_low}(i)\)
\[
X_{zmp}(i) = X_{2\_ZMP}(i);
\]
end
\[
X_{CoM\_1}(i) = (m_1 + 2 \times m_2) \times X_{h\_sim}(i) + j \times Ds + (m_2 + m_3) \times (X_{k\_sim}(i) + \ldots + j \times Ds + X_{suprt}(i) + j \times Ds) + m_3 \times (X_{a\_sim}(i) + j \times Ds + X_{a\_suprt} + \ldots + j \times Ds) + 2 \times m_4 \times (X_{2\_suprt} + j \times Ds + X_2);
\]
\[
X_{t\_sim}(i) = ((2 \times M) \times (X_{zmp}(i) + (\text{acc} \times Zh_{sim}(i))/g) - X_{CoM\_1}(i))/m_1;
\]
\[
V_i = V_f;
\]
\% Line from Heel to Toe for swing leg
\[
X_{13} = [X_1 X_3];
\]
\[
Z_{13} = [Z_1 Z_3];
\]
\% Support leg
\[
X_{suprt} = [X_{2\_suprt} + j \times Ds X_{a\_suprt} + j \times Ds X_{k\_suprt}(i) + \ldots]
\]
j*Ds Xh_suprt(i)+j*Ds];
Z_suprt = [Z2_suprt Za_suprt Zk_suprt(i) Zh_suprt(i)];

% Swing leg
X = [X2 Xa_sim(i)+j*Ds Xk_sim(i)+j*Ds Xh_sim(i)+j*Ds Xt_sim(i)];
Z = [Z2 Za_sim(i) Zk_sim(i) Zh_sim(i) Zh_sim(i)+...
   sqrt((0.55)2-(abs(Xt_sim(i)-Xh_sim(i)-j*Ds))^2)];

%X = [X2 Xa_sim(i)+j*Ds Xk_sim(i)+j*Ds Xh_sim(i)+j*Ds
Xh_sim(i)+j*Ds];
%Z = [Z2 Za_sim(i) Zk_sim(i) Zh_sim(i) Zh_sim(i)+ 0.55];
% calculating ZMP lower and upper boundary
axis([-0.2 2 -0.1 2])
%axis square
hold on
line(X,Z)
line(X13,Z13)
X(1) = [];
Z(1) = [];
X(4) = [];
Z(4) = [];
%plot(X,Z,'r.','MarkerSize',20)
%set(findobj('Type','line'),'Color','k')
line(X_suprt , Z_suprt, 'Color', 'g')
line(X13_suprt+j*Ds , Z13_suprt, 'Color', 'g')
X_suprt(1) = [];
Z_suprt(1) = [];
X_suprt(3) = [];
Z_suprt(3) = [];
plot(X_suprt,Z_suprt,'r.','MarkerSize',20)
pause(0.3)
end

end

hold off
% axis([-0.5 2.5 -0.5 2.5])
% plot(t1,Xzmp_low)
% hold on
% plot(t1,Xzmp_up)

---

B. Walking Down Stairs

clear all;
clc

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%---------------------------
% Author: Mohammadreza Ranjbar
%---------------------------
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Tc = 1;        % Single gait period
Td = 0.2*Tc;        % Double-Support phase time interval
Tm = 0.4;        % At Tm ankle joint is at its highest point
Time_Stepsize = 0.01;


105
Ds = 0.7;       % Gait Length; in this case is tread depth
Xa = [];        % x of Ankle joint
Lan = 0.1;      % Ankle joint height
qb = pi/12;     % Foot leaving angle
Lao = 0.4;      % Ankle position on X axis when t = iTc + Tm
Hst = 0.37;     % Ankle maximum height from a stair
hst = 0.2;      % Each stair height, riser
qf = pi/12;     % Foot landing angle
Laf = 0.15;     % Ankle to toe length
Lab = 0.07;     % Heel to ankle length
Xsd = 0.25;
Xed = 0.20;
Thigh = 0.47;   % Thigh length
Shank = 0.42;   % Shank length
Hh_max = Thigh + Shank - 0.01;       % Maximum hip height
Hh_min = Hh_max - 0.05;       % Minimum hip height

%%%-----------Kinematic of the swing leg-------------%%%  
%%%----------------Ankle joint------------------------%%% 

%First phase input: time, output: Xa

% net = newrb(t,Xa);

t(1) = 0;
t(2) = Td;
t(3) = Tm;
t(4) = Tc;
t(5) = Tc + Td;

Xa(1) = 0;
Xa(2) = Lan*sin(qb) + Laf*(1 - cos(qb));
Xa(3) = Lao;
Xa(4) = 2*Ds + Lan*sin(qf) + Laf*(1 - cos(qf));
Xa(5) = 2*Ds;

net = newrb(t,Xa);

t1 = 0:Time_Stepsize:Tc
Xa_sim = sim(net,t1);

%Second phase inputs: time, Xa, output: Za

% Za(1) = Lan+hst;
Za(2) = Laf*sin(qb) + Lan*cos(qb)+hst;
Za(3) = Hst;
Za(4) = -hst + Laf*sin(qf) + Lan*cos(qf);
Za(5) = -hst + Lan;

input(1,:) = t;
input(2,:) = Xa;

net1 = newrb(input,Za);

input1(1,:) = t1
input1(2,:) = Xa_sim;
Za_sim = sim(net1,input1);
figure(1)
plot(Xa_sim,Za_sim)       %graph Za vs Xa

%Third phase inputs: time, Xa, Za, output: Theta_a
  Theta_a(1) = 0;
  Theta_a(2) = qb;
  Theta_a(3) = qf;
  Theta_a(4) = 0;

input2(1,:) = t;
input2(2,:) = Xa;
%input2(3,:) = Za;
input2(:,2) = [];
net2 = newrb(input2,Theta_a);

input3(1,:) = t1
input3(2,:) = Xa_sim;
%input3(3,:) = Za_sim;
Theta_a_sim = sim(net2,input3);
figure(2)
plot(input3(1,:),Theta_a_sim)

%%%%    ---------------    

---

Hip joint

%%%%

%First phase input: time, output: Xh
input4 = t;
input4(3) = [];
input4(4) = [];

  Xh(1) = Xed;
  Xh(2) = Ds - Xsd;
  Xh(3) = Ds + Xed;

net3 = newrb(input4,Xh);

Xh_sim = sim(net3,t1)

%Second phase input: time, output: Zh
input4(1,:) = [0.5*Td 0.5*(Tc-Td) Tc-0.5*Td];
%input4(2,:) = sim(net3,input4(1,:));
Zh = [Hh_min Hh_max Hh_min-hst];

net4 = newrb(input4,Zh,0,10);

Zh_sim = sim(net4,t1)
figure(3)
plot(Xh_sim,Zh_sim)

%%%%    ---------------    

Knee joint

%%%%

% intersection of two circles
cnst = Thigh^2-Shank^2;
[size0,size_of_t1] = size(t1);
Xk_sim(1,:) = 0;
Zk_sim(1,:) = 0;
for i=1:size_of_t1
    p = (cnst-(Xh_sim(i))^2+(Xa_sim(i))^2-
        (Zh_sim(i))^2+(Za_sim(i))^2)/...
        (2*(Za_sim(i)-Zh_sim(i)));
    q = ( Xa_sim(i) - Xh_sim(i) )/(Za_sim(i) - Zh_sim(i));
    a = 1 + q^2;
    b = 2*(q*Zh_sim(i) - p*q - Xh_sim(i));
    c = (Xh_sim(i))^2 + (Zh_sim(i))^2 + p^2 -
        2*(p*Zh_sim(i)) - Thigh^2;
    Root1 = (-b - sqrt(b^2 - 4*a*c))/ (2*a);
    Root2 = (-b + sqrt(b^2 - 4*a*c))/ (2*a);
    if Root1 > Root2
        Xk_sim(i) = Root1;
    else Xk_sim(i) = Root2;
    end
    Zk_sim(i) = p - q*Xk_sim(i);
end
figure(4)
plot(Xk_sim,Zk_sim)
%figure(1)

%%%%-----------Kinematic of the support leg----------%%%%
%%%%-------------Ankle joint------------------------%%%%

Xa_suprt = Ds;
Za_suprt = Lan;
%%%%-------------Hip joint------------------------%%%%
%there is just one hip joint
%Hip joint for both support and swing leg is same
Xh_suprt = Xh_sim;
Zh_suprt = Zh_sim;

%%%%-------------Knee joint------------------------%%%%

% intersection of two circles
Xk_suprt(1,:) = 0;
Zk_suprt(1,:) = 0;
for i=1:size_of_t1
    p = (cnst-(Xh_suprt(i))^2+(Xa_suprt)^2-(Zh_suprt(i))^2+...
        (Za_suprt)^2)/(2*(Za_suprt-Zh_suprt(i)));
    q = ( Xa_suprt - Xh_suprt(i) )/(Za_suprt - Zh_suprt(i));
    a = 1 + q^2;
    b = 2*(q*Zh_suprt(i) - p*q - Xh_suprt(i));
    c = (Xh_suprt(i))^2 + (Zh_suprt(i))^2 + p^2 -
        2*(p*Zh_suprt(i)) - Thigh^2;
Root1 = \((-b - \sqrt{b^2 - 4a*c})/ (2a)\);
Root2 = \((-b + \sqrt{b^2 - 4a*c})/ (2a)\);

if Root1 > Root2
    Xk_suprt(i) = Root1;
else Xk_suprt(i) = Root2;
end
Zk_suprt(i) = p - q*Xk_suprt(i);
end

%%%%-----------------------Sole foot-----------------------%%%%

%% Heel position
X1_suprt = Xa_suprt - Lab;
Z1_suprt = Za_suprt - Lan;
%Projected ankle on sole foot position
X2_suprt = Xa_suprt;
Z2_suprt = Z1_suprt;
%Toe position
X3_suprt = Xa_suprt + Laf;
Z3_suprt = Z1_suprt;
X13_suprt = [X1_suprt X3_suprt];
Z13_suprt = [Z1_suprt Z3_suprt];
figure(10)
for j = 2:2
    for i =1:size_of_t1
        %clf
        %ploting stairs
        k1 =0;
        for k = 1:2:15
            X_stair = [k*Ds/2 (k+2)*Ds/2 (k+2)*Ds/2];
            Z_stair = [k1*hst k1*hst (k1-1)*hst];
            line(X_stair , Z_stair)
            k1 = k1 - 1;
        end
        %Projected ankle on sole foot position for swing leg
        X2 = Xa_sim(i) - Lan*sin(Theta_a_sim(i)) + j*Ds;
        Z2 = Za_sim(i) - Lan*cos(Theta_a_sim(i)) - j*hst;
        % Heel position for swing leg
        X1 = X2 - Lab*cos(Theta_a_sim(i));
        Z1 = Z2 + Lab*sin(Theta_a_sim(i));
        %Toe position for swing leg
        X3 = X2 + Laf*cos(Theta_a_sim(i));
        Z3 = Z2 - Laf*sin(Theta_a_sim(i));
        %Line from Heel to Toe for swing leg
        X13 = [X1 X3];
        Z13 = [Z1 Z3];
        %Support leg
        X_suprt = [X2_suprt+j*Ds Xa_suprt+j*Ds Xk_suprt(i)+j*Ds...
Xh_suprt(i)+j*Ds];
Z_suprt = [Z2_suprt-j*hst Za_suprt-j*hst Zk_suprt(i)-j*hst...
Zh_suprt(i)-j*hst];

%Swing leg
X = [X2 Xa_sim(i)+j*Ds Xk_sim(i)+j*Ds Xh_sim(i)+j*Ds
Xh_sim(i)+j*Ds];
Z = [Z2 Za_sim(i)-j*hst Zk_sim(i)-j*hst Zh_sim(i)-j*hst...
Zh_sim(i)-j*hst+0.57];
axis([0.5 4 -2 2])
hold on
line(X,Z)
line(X13,Z13)
X(1) = []; Z(1) = [];
X(4) = []; Z(4) = [];
%plot(X,Z,'r.','MarkerSize',20)
%set(findobj('Type','line'),'Color','k')
line(X_suprt , Z_suprt , 'color' , 'g')
line(X13_suprt+j*Ds , Z13_suprt-j*hst,'color','g')
X_suprt(1) = []; Z_suprt(1) = [];
X_suprt(3) = []; Z_suprt(3) = [];
plot(X_suprt,Z_suprt,'r.','MarkerSize',20)
pause(0.01)
end
end
hold off

C. Walking Up Stairs

clear all;
clc

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-----------------------------------------------------------------------------------------------------------------
% Author: Mohammadreza Ranjbar---------------------------------------------------------------%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Tc = 1;     % Single gait period
Td = 0.2*Tc;        % Double-Support phase time interval
Tm = 0.4;        % At Tm ankle joint is at its highest point
Time_Stepsize = 0.1;
Ds = 0.7;       % Gait Length; in this case is tread depth
Xa = [];        % x of Ankle joint
Lan = 0.1;      % Ankle joint height
qb = pi/12;    % Foot leaving angle
Lao = 0.35;      % Ankle position on X axis when t = iTc + Tm
Hst = 0.1;      % Ankle maximum height from a stair
hst = 0.2;      % Each stair height, riser
qf = pi/12;    % Foot landing angle
Laf = 0.15; % Ankle to toe length
Lab = 0.07; % Heel to ankle length
Xsd = 0.25;
Xed = 0.20;
Thigh = 0.47; % Thigh length
Shank = 0.42; % Shank length
Hh_max = Thigh + Shank - 0.01; % Maximum hip height
Hh_min = Hh_max - 0.05; % Minimum hip height

%%%-------------------Kinematic of the swing leg------------------%%%
%%%------------------------Ankle joint------------------------%%%
%First phase input: time, output: Xa

t(1) = 0;
t(2) = Td;
t(3) = Tm;
t(4) = Tc;
t(5) = Tc + Td;

Xa(1) = 0;
Xa(2) = Lan*sin(qb) + Laf*(1 - cos(qb));
Xa(3) = Lao;
Xa(4) = 2*Ds - Lan*sin(qf) - Lab*(1 - cos(qf));
Xa(5) = 2*Ds;

net = newrb(t,Xa);

%Second phase inputs: time, Xa, output: Za

Za(1) = Lan;
Za(2) = Laf*sin(qb) + Lan*cos(qb);
Za(3) = hst + Hst;
Za(4) = hst + Lab*sin(qf) + Lan*cos(qf);
Za(5) = hst + Lan;

input(1,:) = t;
input(2,:) = Xa;
net1 = newrb(input,Za);

%Third phase inputs: time, Xa, Za, output: Theta_a

Theta_a(1) = 0;
Theta_a(2) = qb;
Theta_a(3) = -qf;
Theta_a(4) = 0;

input2(1,:) = t;
input2(2,:) = Xa;
%input2(3,:) = Za;
input2(:,2) = [];

net2 = newrb(input2,Theta_a);

input3(1,:) = t1
input3(2,:) = Xa_sim;
%input3(3,:) = Za_sim;
Theta_a_sim = sim(net2,input3);
figure(2)
%plot(input3(1,:),Theta_a_sim)

%%%% ---------------Hip joint-------------------%%%%
%First phase input: time, output: Xh
input4 = t;
input4(3) = [];
input4(4) = [];

Xh(1) = Xed;
Xh(2) = Ds - Xsd;
Xh(3) = Ds + Xed;

net3 = newrb(input4,Xh);

Xh_sim = sim(net3,t1)

%Second phase input: time, output: Zh
input4(1,:) = [0.5*Td 0.5*(Tc-Td) Tc-0.5*Td];
%input4(2,:) = sim(net3,input4(1,:));
Zh = [Hh_min Hh_max Hh_min];
net4 = newrb(input4,Zh,0,10);

Zh_sim = sim(net4,t1);
figure(3)
%plot(Xh_sim,Zh_sim)

%%%% -----------------------Knee joint-------------------%%%%
% intersection of two circles
cnst = Thigh^2-Shank^2;
[size0,size_of_t1] = size(t1);

Xk_sim(1,:) = 0;
Zk_sim(1,:) = 0;
for i=1:size_of_t1
    p = (cnst-(Xh_sim(i))^2+(Xa_sim(i))^2-(Zh_sim(i))^2+(Za_sim(i))^2) / ... 
        (2*(Za_sim(i)-Zh_sim(i)));

% code continues...
q = (Xa_sim(i) - Xh_sim(i))/(Za_sim(i) - Zh_sim(i));
\[ a = 1 + q^2; \]
\[ b = 2*(q*Zh_sim(i) - p*q - Xh_sim(i)); \]
\[ c = (Xh_sim(i))^2 + (Zh_sim(i))^2 + p^2 - 2*(p*Zh_sim(i)) - \text{Thigh}^2; \]
\[ \text{Root1} = (-b - \sqrt{b^2 - 4*a*c})/ (2*a); \]
\[ \text{Root2} = (-b + \sqrt{b^2 - 4*a*c})/ (2*a); \]
\[ \text{if Root1} > \text{Root2} \]
\[ Xk_sim(i) = \text{Root1}; \]
\[ \text{else Xk_sim(i) = Root2; } \]
\[ \text{end} \]
\[ Zk_sim(i) = p - q*Xk_sim(i); \]
\[ \text{end} \]
\%figure(4)
\%plot(Xk_sim,Zk_sim)
\%figure(1)

%%%----------------------Kinematic of the support leg---------------------%%%  
%%%----------------------Ankle joint----------------------%%%  
Xa_suprt = Ds;
Za_suprt = Lan;

%%%----------------------Hip joint----------------------%%%  
%there is just one hip joint  
%Hip joint for both support and swing leg is same  
Xh_suprt = Xh_sim;
Zh_suprt = Zh_sim;

%%%----------------------Knee joint----------------------%%%  
% intersection of two circles  
Xk_suprt(1,:) = 0;
Zk_suprt(1,:) = 0;
for \( i \) = 1:size_of_t1  
\[ p = (\text{cnst}-(Xh_suprt(i))^2+(Xa_suprt)^2-(Zh_suprt(i))^2+...  
\quad (Za_suprt)^2)/(2*(Za_suprt-Zh_suprt(i))); \]
\[ q = (Xa_suprt - Xh_suprt(i))/(Za_suprt - Zh_suprt(i)); \]
\[ a = 1 + q^2; \]
\[ b = 2*(q*Zh_suprt(i) - p*q - Xh_suprt(i)); \]
\[ c = (Xh_suprt(i))^2 + (Zh_suprt(i))^2 + p^2 - ...  
\quad 2*(p*Zh_suprt(i)) - \text{Thigh}^2; \]
\[ \text{Root1} = (-b - \sqrt{b^2 - 4*a*c})/ (2*a); \]
\[ \text{Root2} = (-b + \sqrt{b^2 - 4*a*c})/ (2*a); \]
\[ \text{if Root1} > \text{Root2} \]
\[ Xk_suprt(i) = \text{Root1}; \]
else Xk_suprt(i) = Root2;
end
Zk_suprt(i) = p - q*Xk_suprt(i);
end

%%%%---------------Sole foot-------------------%%%  

% Heel position
X1_suprt = Xa_suprt - Lab;
Z1_suprt = Za_suprt - Lan;
%Projected ankle on sole foot position
X2_suprt = Xa_suprt;
Z2_suprt = Z1_suprt;
%Toe position
X3_suprt = Xa_suprt + Laf;
Z3_suprt = Z1_suprt;
X13_suprt = [X1_suprt X3_suprt];
Z13_suprt = [Z1_suprt Z3_suprt];
figure(10)
for j = 0:4
for i =1:size_of_t1
clf
%ploting stairs
k1 =0;
for k = 1:2:15
X_stair = [k*Ds/2 (k+2)*Ds/2 (k+2)*Ds/2];
Z_stair = [k1*hst k1*hst (k1+1)*hst];
line(X_stair , Z_stair)
k1 = k1+1;
end

%Projected ankle on sole foot position for swing leg
X2 = Xa_sim(i) - Lan*sin(Theta_a_sim(i)) + j*Ds;
Z2 = Za_sim(i) - Lan*cos(Theta_a_sim(i)) + j*hst;
% Heel position for swing leg
X1 = X2 - Lab*cos(Theta_a_sim(i));
Z1 = Z2 + Lab*sin(Theta_a_sim(i));
%Toe position for swing leg
X3 = X2 + Laf*cos(Theta_a_sim(i));
Z3 = Z2 - Laf*sin(Theta_a_sim(i));
X13 = [X1 X3];
Z13 = [Z1 Z3];

%Support leg
X_suprt = [X2_suprt+j*Ds Xa_suprt+j*Ds Xk_suprt(i)+j*Ds...
Xh_suprt(i)+j*Ds];
Z_suprt = [Z2_suprt+j*hst Za_suprt+j*hst Zk_suprt(i)+j*hst...
Zh_suprt(i)+j*hst];

%Swing leg
X = [X2 Xa_sim(i)+j*Ds Xk_sim(i)+j*Ds Xh_sim(i)+j*Ds
Xh_sim(i)+j*Ds];
Z = [Z2 Za_sim(i)+j*hst Zk_sim(i)+j*hst Zh_sim(i)+j*hst...
Zh_sim(i)+j*hst+0.57];
D. Walking Up Hill

clear all;
clc

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%--------Biped Robat Walk on an inclined surface--------%
%------------------Author: Mohammadreza Ranjbar------------------%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Tc = 1; % Single gait period
Td = 0.2*Tc; % Double-Support phase time interval
Tm = 0.4; % At Tm ankle joint is at its highest point
Time_Stepsize = 0.01;
Ds = 0.5; % Gait Length
Xa = []; % x of Ankle joint
Lan = 0.1; % Ankle joint height
Lamda = pi/10; % Surface slope
qb = pi/12; % Foot leaving angle
Lao = 0.3; % Ankle position on X axis when t = iTc + Tm
Hao = 0.3; % Ankle maximum height
qf = pi/12; % Foot landing angle
Laf = 0.15; % Ankle to toe length
Lab = 0.07; % Heel to ankle length
Xsd = 0.25;
Xed = 0.22;
Thigh = 0.47; % Thigh length
Shank = 0.42; % Shank length
Hh_max = Thigh + Shank - 0.03; % Maximum hip height
Hh_min = Hh_max - 0.05; % Minimum hip height

%%%--------------Kinematic of the swing leg-----------------%%%  
%%%--------------Ankle joint-----------------------------%%%  
% First phase input: time, output: Xa

   t(1) = 0;  
   t(2) = Td;  
   t(3) = Tm;  
   t(4) = Tc;  
   t(5) = Tc +Td;  

   Xa(1) = Lab*cos(Lamda) - Lan*sin(Lamda);  
   Xa(2) = Lab*cos(Lamda-qb) - Lan*sin(Lamda-qb) + ...  
        ( Laf + Lab - (Laf+Lab)*cos(qb) )*cos(Lamda);  
   Xa(3) = Lao*cos(Lamda);  
   Xa(4) = 2*Ds*cos(Lamda) - Lan*sin(Lamda+qf) - ...  
        Lab*(1-cos(qf))*cos(Lamda);  
   Xa(5) = 2*Ds*cos(Lamda) - Lan*sin(Lamda);  

net = newrb(t,Xa);  

   t1 = 0:Time_Stepsize:Tc  
   Xa_sim = sim(net,t1);  

% Second phase inputs: time, Xa, output: Za

   Za(1) = Lan*cos(Lamda) + Lab*sin(Lamda);  
   Za(2) = Xa(2)*tan(Lamda) + Lan*cos(Lamda-qb) + ...  
        (Laf*cos(Lamda-qb)-Lan*sin(Lamda-qb))*tan(Lamda) -  
        Laf*sin(Lamda-qb);  
   Za(3) = Lao*sin(Lamda) + Hao;  
   Za(4) = Lan*sin(qf+Lamda) + Lab*sin(qf+Lamda) +  
        (2*Ds)*sin(Lamda);  
   Za(5) = Lan*cos(Lamda) + 2*Ds*sin(Lamda);  

input(1,:) = t;  
input(2,:) = Xa;  

net1 = newrb(input,Za);  

input1(1,:) = t1  
input1(2,:) = Xa_sim;  
Za_sim = sim(net1,input1);  

figure(1)  
plot(Xa_sim,Za_sim)     %garph Za vs Xa  

% Third phase inputs: time, Xa, Za, output: Theta_a  

   Theta_a(1) = -Lamda;  
   Theta_a(2) = -Lamda + qb;  
   Theta_a(3) = -Lamda - qf;  
   Theta_a(4) = -Lamda;
\[
\text{input2}(1,:) = \mathbf{t};
\text{input2}(2,:) = \mathbf{Xa};
%\text{input2}(3,:) = \mathbf{Za};
\text{input2}(:,2) = [];$
\]
\[
\text{net2} = \text{newrb}(\text{input2},\Theta_a);
\]
\[
\text{input3}(1,:) = \mathbf{t1}
\text{input3}(2,:) = \mathbf{Xa}\_sim;
%\text{input3}(3,:) = \mathbf{Za}\_sim;
\Theta_a\_sim = \text{sim}(\text{net2},\text{input3});
\]
\[
%\text{figure}(2)
%\text{plot}(\text{input3}(1,:),\Theta_a\_sim)
\]

%%%%
---------------
Hip joint
---------------
%%%%

%First phase input: time, output: Xh
\[
\text{input4} = \mathbf{t};
\text{input4}(3) = [];
\text{input4}(4) = [];
\]
\[
\mathbf{Xh}(1) = \mathbf{Xed};
\mathbf{Xh}(2) = \mathbf{Ds}\_\cos(\Lambda) - \mathbf{Xsd};
\mathbf{Xh}(3) = \mathbf{Ds}\_\cos(\Lambda) + \mathbf{Xed};
\]
\[
\text{net3} = \text{newrb}(\text{input4},\mathbf{Xh});
\]
\[
\mathbf{Xh}\_\text{sim} = \text{sim}(\text{net3},\mathbf{t1})
\]

%Second phase input: time, output: Zh
\[
\text{input4}(1,:) = [0.5*\mathbf{Td} \ 0.5*(\mathbf{Tc}-\mathbf{Td}) \ \mathbf{Tc}-0.5*\mathbf{Td}] ;
\text{input4}(2,:) = \text{sim}(\text{net3},\text{input4}(1,:));
\mathbf{Zh} = [\mathbf{Hh}\_\min+\mathbf{Ds}\_\sin(\Lambda) \ \mathbf{Hh}\_\max+\mathbf{Ds}\_\sin(\Lambda)
\mathbf{Hh}\_\min+\mathbf{Ds}\_\sin(\Lambda)];$
\]
\[
\text{net4} = \text{newrb}(\text{input4},\mathbf{Zh},0,10);
\]
\[
\mathbf{Zh}\_\text{sim} = \text{sim}(\text{net4},\mathbf{t1})$
\]
%figure(3)
%plot(Xh\_sim,Zh\_sim)

%%%%
---------------
Knee joint
---------------
%%%%

% intersection of two circles
cnst = \text{Thigh}^2-\text{Shank}^2;
[size0,size\_of\_t1] = \text{size}(\mathbf{t1});
\]
\[
\mathbf{Xk}\_\text{sim}(1,:) = 0;
\mathbf{Zk}\_\text{sim}(1,:) = 0;
\text{for } i=1:\text{size\_of\_t1}
\]
\[
p = (\text{cnst}-(\mathbf{Xh}\_\text{sim}(i))^2+(\mathbf{Xa}\_\text{sim}(i))^2-
(\mathbf{Zh}\_\text{sim}(i))^2+(\mathbf{Za}\_\text{sim}(i))^2)/...$
\[
(2*(\mathbf{Za}\_\text{sim}(i)-\mathbf{Zh}\_\text{sim}(i)));
q = (\mathbf{Xa}\_\text{sim}(i) - \mathbf{Xh}\_\text{sim}(i))/(\mathbf{Za}\_\text{sim}(i) - \mathbf{Zh}\_\text{sim}(i));$
\]
a = 1 + q^2;
b = 2*(q*Zh_sim(i) - p*q - Xh_sim(i));
c = (Xh_sim(i))^2 + (Zh_sim(i))^2 + p^2 - 2*(p*Zh_sim(i)) - Thigh^2;

Root1 = (-b - sqrt(b^2 - 4*a*c))/ (2*a);
Root2 = (-b + sqrt(b^2 - 4*a*c))/ (2*a);

if Root1 > Root2
    Xk_sim(i) = Root1;
else Xk_sim(i) = Root2;
end

Zk_sim(i) = p - q*Xk_sim(i);
end

%figure(4)
%plot(Xk_sim,Zk_sim)
%figure(1)

%%%-------------------Kinematic of the support leg-------------------%%%
%%%----------------Ankle joint---------------------%%%

Xa_suprt = Ds*cos(Lamda) - Lan*sin(Lamda);
Zh_suprt = Ds*sin(Lamda) + Lan*cos(Lamda);

%%%-------------------Hip joint----------------------%%%
%there is just one hip joint
%Hip joint for both support and swing leg is same
Xh_suprt = Xh_sim;
Zh_suprt = Zh_sim;

%%%-------------------Knee joint---------------------%%%

% intersection of two circles
Xk_suprt(1,:) = 0;
Zk_suprt(1,:) = 0;
for i=1:size_of_t1
    p = (cnst - (Xh_suprt(i))^2+(Xa_suprt)^2 - ...
        (Zh_suprt(i))^2+(Za_suprt)^2)/ (2*(Za_suprt-Zh_suprt(i)));
    q = (Xa_suprt - Xh_suprt(i))/(Za_suprt - Zh_suprt(i));
    a = 1 + q^2;
    b = 2*(q*Zh_suprt(i) - p*q - Xh_suprt(i));
    c = (Xh_suprt(i))^2 + (Zh_suprt(i))^2 + p^2 - ...
        2*(p*Zh_suprt(i)) - Thigh^2;
    Root1 = (-b - sqrt(b^2 - 4*a*c))/ (2*a);
    Root2 = (-b + sqrt(b^2 - 4*a*c))/ (2*a);

    if Root1 > Root2
        Xk_suprt(i) = Root1;
    else Xk_suprt(i) = Root2;
end
### Sole foot

---

%Projected ankle on sole foot position
\[
X2_{\text{suprt}} = Xa_{\text{suprt}} + Lan \cdot \sin(\Lambda); \\
Z2_{\text{suprt}} = Za_{\text{suprt}} - Lan \cdot \cos(\Lambda);
\]

% Heel position
\[
X1_{\text{suprt}} = X2_{\text{suprt}} - Lab \cdot \cos(\Lambda); \\
Z1_{\text{suprt}} = Z2_{\text{suprt}} - Lab \cdot \sin(\Lambda);
\]

%Toe position
\[
X3_{\text{suprt}} = X2_{\text{suprt}} + Laf \cdot \cos(\Lambda); \\
Z3_{\text{suprt}} = Z2_{\text{suprt}} + Laf \cdot \sin(\Lambda);
\]

\[
X13_{\text{suprt}} = [X1_{\text{suprt}} \ X3_{\text{suprt}}]; \\
Z13_{\text{suprt}} = [Z1_{\text{suprt}} \ Z3_{\text{suprt}}];
\]

\[
figure(10) \\
for \ j = 0:3 \\
    for \ i = 1:\text{size}_of\_t1 \\
        \%
    \end
\]

### Swing leg

\[
X2 = Xa_{\text{sim}(i)} - Lan \cdot \sin(\Theta_a_{\text{sim}(i)}) + j \cdot Ds \cdot \cos(\Lambda); \\
Z2 = Za_{\text{sim}(i)} - Lan \cdot \cos(\Theta_a_{\text{sim}(i)}) + j \cdot Ds \cdot \sin(\Lambda);
\]

% Heel position for swing leg
\[
X1 = X2 - Lab \cdot \cos(\Theta_a_{\text{sim}(i)}); \\
Z1 = Z2 + Lab \cdot \sin(\Theta_a_{\text{sim}(i)});
\]

%Toe position for swing leg
\[
X3 = X2 + Laf \cdot \cos(\Theta_a_{\text{sim}(i)}); \\
Z3 = Z2 - Laf \cdot \sin(\Theta_a_{\text{sim}(i)});
\]

% Line from Heel to Toe for swing leg
\[
X13 = [X1 \ X3]; \\
Z13 = [Z1 \ Z3];
\]

% Support leg
\[
X_{\text{suprt}} = \[X2_{\text{suprt}} + j \cdot Ds \cdot \cos(\Lambda) \ Xa_{\text{suprt}} + j \cdot Ds \cdot \cos(\Lambda) \ ... \ Xk_{\text{suprt}}(i) + j \cdot Ds \cdot \cos(\Lambda) \ Xh_{\text{suprt}}(i) + j \cdot Ds \cdot \cos(\Lambda)\]; \\
Z_{\text{suprt}} = \[Z2_{\text{suprt}} + j \cdot Ds \cdot \sin(\Lambda) \ Za_{\text{suprt}} + j \cdot Ds \cdot \sin(\Lambda) \ ... \ Zk_{\text{suprt}}(i) + j \cdot Ds \cdot \sin(\Lambda) \ Zh_{\text{suprt}}(i) + j \cdot Ds \cdot \sin(\Lambda)\];
\]

% Swing leg
\[
X = \[X2_{\text{sim}(i)} + j \cdot Ds \cdot \cos(\Lambda) \ Xa_{\text{sim}(i)} + j \cdot Ds \cdot \cos(\Lambda) \ ... \ Xh_{\text{sim}(i)} + j \cdot Ds \cdot \cos(\Lambda) \ Xh_{\text{sim}(i)} + j \cdot Ds \cdot \cos(\Lambda)\]; \\
Z = \[Z2_{\text{sim}(i)} + j \cdot Ds \cdot \sin(\Lambda) \ Za_{\text{sim}(i)} + j \cdot Ds \cdot \sin(\Lambda) \ ... \ Zh_{\text{sim}(i)} + j \cdot Ds \cdot \sin(\Lambda) \ Zh_{\text{sim}(i)} + 0.57 + j \cdot Ds \cdot \sin(\Lambda)\];
\]

axis([-0.2 2.8 -0.5 2.5])
\]

hold on
\]

figure(10)
\]

for \ j = 0:3 \\
    for \ i = 1:\text{size}_of\_t1 \\
        \%
    \end
\]
%plot(X,Z,'r.','MarkerSize',20)
%set(findobj('Type','line'),'Color','k')
line(X_suprt, Z_suprt, 'color', 'g')
line(X13_suprt+j*Ds*cos(Lamda),Z13_suprt+j*Ds*sin(Lamda),'color','g')
X_suprt(1) = []; Z_suprt(1) = [];
X_suprt(3) = []; Z_suprt(3) = [];
plot(X_suprt,Z_suprt,'r.','MarkerSize',20)
pause(0.1)
end
end
hold off

E. Walking Down Hill

clear all;
cclc

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%----------Biped Robat Walk on a declined surface---------%
%----------------Author: Mohammadreza Ranjbar-------------%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Tc = 1; % Single gait period
Td = 0.2*Tc; % Double-Support phase time interval
Tm = 0.4; % At Tm ankle joint is at its highest point
Time_Stepsize = 0.1;
Ds = 0.4; % Gait Length
Xa = []; % x of Ankle joint
Lan = 0.1; % Ankle joint height
Lamda = pi/10; % Surface slope
qb = pi/12; % Foot leaving angle
Lao = 0.3; % Ankle position on X axis when t = iTc + Tm
Hao = 0.3; % Ankle maximum height
qf = pi/12; % Foot landing angle
Laf = 0.15; % Ankle to toe length
Lab = 0.07; % Heel to ankle length
Xsd = 0.15;
Xed = 0.15;
Thigh = 0.47; % Thigh length
Shank = 0.42; % Shank length
Hh_max = Thigh + Shank; % Maximum hip height
Hh_min = Hh_max - 0.1; % Minimum hip height

%%%%%%%%%%%%%%%%------Kinematic of the swing leg-----------
%%%%%%%%%%%%%%%%------Ankle joint-------------------------
%First phase input: time, output: Xa

t(1) = 0;
t(2) = Td;
t(3) = Tm;
t(4) = Tc;
t(5) = Tc + Td;

Xa(1) = Lab*cos(Lamda) + Lan*sin(Lamda);
Xa(2) = Lab*cos(qb+Lamda) + Lan*sin(qb+Lamda) + ...
   ( Laf + Lab - (Laf+Lab)*cos(qb) )*cos(Lamda);
Xa(3) = Lao*cos(Lamda);
Xa(4) = 2*Ds*cos(Lamda) - Lan*sin(qf-Lamda) - ...
   Lab*(1-cos(qf))*cos(Lamda);
Xa(5) = 2*Ds*cos(Lamda) - Lan*sin(Lamda);

net = newrb(t,Xa);
t1 = 0:Time_Stepsize:Tc
Xa_sim = sim(net,t1);

%Second phase inputs: time, Xa, output: Za

Za(1) = Lan*cos(Lamda) + Lab*sin(Lamda);
Za(2) = Xa(2)*tan(Lamda) + Lan*cos(Lamda+qb) + ...
   (Laf*cos(Lamda-qb)-Lan*sin(Lamda-qb))*tan(Lamda) -
   Laf*sin(Lamda-qb);
Za(3) = Hao;
Za(4) = Lan*sin(qf+Lamda) + Lab*sin(qf+Lamda) - (2*Ds-
   Lab)*sin(Lamda);
Za(5) = Lan*cos(Lamda)-2*Ds*sin(Lamda);

input(1,:) = t;
input(2,:) = Xa;

net1 = newrb(input,Za);

input1(1,:) = t1
input1(2,:) = Xa_sim;
Za_sim = sim(net1,input1);
%figure(1)
%plot(Xa_sim,Za_sim)   %garph Za vs Xa

%Third phase inputs: time, Xa, Za, output: Theta_a

Theta_a(1) = Lamda;
Theta_a(2) = Lamda + qb;
Theta_a(3) = Lamda - qf;
Theta_a(4) = Lamda;

input2(1,:) = t;
input2(2,:) = Xa;
%input2(3,:) = Za;
input2(:,2) = [];

net2 = newrb(input2,Theta_a);

input3(1,:) = t1
input3(2,:) = Xa_sim;
%input3(3,:) = Za_sim;
Theta_a_sim = sim(net2,input3);
%figure(2)
%plot(input3(1,:),Theta_a_sim)

%%--------------Hip joint------------------%%
%First phase input: time, output: Xh
input4 = t;
input4(3) = [];
input4(4) = [];

Xh(1) = Xed;
Xh(2) = Ds*cos(Lamda) - Xsd;
Xh(3) = Ds*cos(Lamda) + Xed;

net3 = newrb(input4,Xh);
Xh_sim = sim(net3,t1)

%Second phase input: time, output: Zh
input4(1,:) = [0.5*Td 0.5*(Tc-Td) Tc-0.5*Td];
%input4(2,:) = sim(net3,input4(1,:));
Zh = [Hh_min Hh_max-Ds*sin(Lamda) Hh_min-Ds*sin(Lamda)];

net4 = newrb(input4,Zh,0,10);
Zh_sim = sim(net4,t1)
%figure(3)
%plot(Xh_sim,Zh_sim)

%%--------------Knee joint------------------%%
% intersection of two circles
cnst = Thigh^2-Shank^2;
[size0,size_of_t1] = size(t1);

Xk_sim(1,:) = 0;
Zk_sim(1,:) = 0;
for i=1:size_of_t1
    p = (cnst-(Xh_sim(i))^2+(Xa_sim(i))^2-(Zh_sim(i))^2+(Za_sim(i))^2)/...
        (2*(Za_sim(i)-Zh_sim(i)));
    q = ( Xa_sim(i) - Xh_sim(i) )/(Za_sim(i) - Zh_sim(i));
    a = 1 + q^2;
    b = 2*(q*Zh_sim(i) - p*q - Xh_sim(i));
    c = (Xh_sim(i))^2 + (Zh_sim(i))^2 + p^2 - 2*(p*Zh_sim(i)) - Thigh^2;
    Root1 = (-b - sqrt(b^2 - 4*a*c))/ (2*a);
    Root2 = (-b + sqrt(b^2 - 4*a*c))/ (2*a);
    if Root1 > Root2
        Xk_sim(i) = Root1;
else Xk_sim(i) = Root2;
end
Zk_sim(i) = p - q*Xk_sim(i);
end

figure(4)
plot(Xk_sim,Zk_sim)
figure(1)

%%%----------------------Kinematic of the support leg----------------------

%%%----------------------Ankle joint----------------------

Xa_suprt = Ds*cos(Lamda) + Lan*sin(Lamda);
Za_suprt = Lan*cos(Lamda) - Ds*sin(Lamda);

%%%----------------------Hip joint----------------------

there is just one hip joint
Hip joint for both support and swing leg is same
Xh_suprt = Xh_sim;
Zh_suprt = Zh_sim;

%%%----------------------Knee joint----------------------

% intersection of two circles
Xk_suprt(1,:) = 0;
Zk_suprt(1,:) = 0;
for i=1:size_of_t1

    p = (cnst-(Xh_suprt(i))^2+(Xa_suprt)^2-(Zh_suprt(i))^2+... 
        (Za_suprt)^2)/(2*(Za_suprt-Zh_suprt(i))); 
    q = (Xa_suprt - Xh_suprt(i))/(Za_suprt - Zh_suprt(i)); 
    a = 1 + q^2; 
    b = 2*(q*Zh_suprt(i) - p*q - Xh_suprt(i)); 
    c = (Xh_suprt(i))^2 + (Zh_suprt(i))^2 + p^2 -... 
        2*p*Zh_suprt(i) - Thigh^2; 

    Root1 = (-b - sqrt(b^2 - 4*a*c))/ (2*a); 
    Root2 = (-b + sqrt(b^2 - 4*a*c))/ (2*a); 
    if Root1 > Root2
        Xk_suprt(i) = Root1; 
    else Xk_suprt(i) = Root2; 
    end

    Zk_suprt(i) = p - q*Xk_suprt(i);
end

%%%----------------------Sole foot----------------------

%Projected ankle on sole foot position
X2_suprt = Xa_suprt - Lan*sin(Lamda);
Z2_suprt = Za_suprt - Lan*cos(Lamda); 
% Heel position
X1_suprt = X2_suprt - Lab*cos(Lamda);
\[ Z_{1\text{ suprt}} = Z_{2\text{ suprt}} + \text{Lab} \cdot \sin(\text{Lamda}); \]

% Toe position
\[ X_{3\text{ suprt}} = X_{2\text{ suprt}} + \text{Laf} \cdot \cos(\text{Lamda}); \]
\[ Z_{3\text{ suprt}} = Z_{2\text{ suprt}} - \text{Laf} \cdot \sin(\text{Lamda}); \]
\[ X_{13\text{ suprt}} = [X_{1\text{ suprt}} X_{3\text{ suprt}}]; \]
\[ Z_{13\text{ suprt}} = [Z_{1\text{ suprt}} Z_{3\text{ suprt}}]; \]

\text{figure}(10)
for \( j = 0:2 \)
\text{for} \( i = 1:\text{size of t1} \)
\%	ext{clf}
\[ X_{\text{inc}} = [0 \text{ 5}]; \]
\[ Z_{\text{inc}} = [0 \text{ -5} \cdot \tan(\text{Lamda})]; \]
\text{line}(X_{\text{inc}}, Z_{\text{inc}})
\%	ext{Projected ankle on sole foot position for swing leg}
\[ X_{2} = X_{\text{a sim}(i)} - \text{Lan} \cdot \sin(\text{Theta}_{\text{a sim}(i)}) + j \cdot \text{Ds} \cdot \cos(\text{Lamda}); \]
\[ Z_{2} = Z_{\text{a sim}(i)} - \text{Lan} \cdot \cos(\text{Theta}_{\text{a sim}(i)}) - j \cdot \text{Ds} \cdot \sin(\text{Lamda}); \]
\%	ext{Heel position for swing leg}
\[ X_{1} = X_{2} - \text{Lab} \cdot \cos(\text{Theta}_{\text{a sim}(i)}); \]
\[ Z_{1} = Z_{2} + \text{Lab} \cdot \sin(\text{Theta}_{\text{a sim}(i)}); \]
\%	ext{Toe position for swing leg}
\[ X_{3} = X_{2} + \text{Laf} \cdot \cos(\text{Theta}_{\text{a sim}(i)}); \]
\[ Z_{3} = Z_{2} - \text{Laf} \cdot \sin(\text{Theta}_{\text{a sim}(i)}); \]
\%	ext{Line from Heel to Toe for swing leg}
\[ X_{13} = [X_{1} X_{3}]; \]
\[ Z_{13} = [Z_{1} Z_{3}]; \]
\%	ext{Support leg}
\[ X_{\text{suprt}} = [X_{2\text{ suprt}} + j \cdot \text{Ds} \cdot \cos(\text{Lamda}) X_{\text{a suprt}} + j \cdot \text{Ds} \cdot \cos(\text{Lamda}) ... \]
\[ X_{k\text{ suprt}(i)} + j \cdot \text{Ds} \cdot \cos(\text{Lamda}) X_{h\text{ suprt}(i)} + j \cdot \text{Ds} \cdot \cos(\text{Lamda})]; \]
\[ Z_{\text{suprt}} = [Z_{2\text{ suprt}} - j \cdot \text{Ds} \cdot \sin(\text{Lamda}) Z_{\text{a suprt}} - j \cdot \text{Ds} \cdot \sin(\text{Lamda}) ... \]
\[ Z_{k\text{ suprt}(i)} - j \cdot \text{Ds} \cdot \sin(\text{Lamda}) Z_{h\text{ suprt}(i)} - j \cdot \text{Ds} \cdot \sin(\text{Lamda})]; \]
\%	ext{Swing leg}
\[ X = [X_{2} X_{\text{a sim}(i)} + j \cdot \text{Ds} \cdot \cos(\text{Lamda}) X_{k\text{ sim}(i)} + j \cdot \text{Ds} \cdot \cos(\text{Lamda}) ... \]
\[ X_{h\text{ sim}(i)} + j \cdot \text{Ds} \cdot \cos(\text{Lamda}) X_{h\text{ sim}(i)} + j \cdot \text{Ds} \cdot \cos(\text{Lamda})]; \]
\[ Z = [Z_{2} Z_{\text{a sim}(i)} - j \cdot \text{Ds} \cdot \sin(\text{Lamda}) Z_{k\text{ sim}(i)} - j \cdot \text{Ds} \cdot \sin(\text{Lamda}) ... \]
\[ Z_{h\text{ sim}(i)} - j \cdot \text{Ds} \cdot \sin(\text{Lamda}) Z_{h\text{ sim}(i)} + 0.57 - j \cdot \text{Ds} \cdot \sin(\text{Lamda})]; \]
\text{axis}([-0.1 2 -1 1.5])
\text{hold on}
\text{line}(X, Z)
\text{line}(X_{13}, Z_{13})
X(1) = [];
Z(1) = [];
X(4) = [];
Z(4) = [];
\%	ext{plot}(X, Z, 'r', 'MarkerSize', 20)
\%	ext{set}(	ext{findobj}('Type', 'line'), 'Color', 'k')
\text{line}(X_{\text{suprt}}, Z_{\text{suprt}}, 'color', 'g')
\text{line}(X_{13\text{ suprt}} + j \cdot \text{Ds} \cdot \cos(\text{Lamda}), Z_{13\text{ suprt}} - j \cdot \text{Ds} \cdot \sin(\text{Lamda}), 'color', 'g')
\text{X_{suprt}(1)} = [];
Z_{suprt}(1) = [];
\text{X_{suprt}(3)} = [];
Z_{suprt}(3) = [];
\text{plot}(X_{suprt}, Z_{suprt}, 'r', 'MarkerSize', 20)
\text{pause}(0.1)
end
end
\text{hold off}