University of Cincinnati

Date: 11/6/2017

I, Ameya S Chamnikar, hereby submit this original work as part of the requirements for the degree of Master of Science in Mechanical Engineering.

It is entitled:
Genetic Algorithm Based Trajectory Generation and Inverse Kinematics Calculation for Lower Limb Exoskeleton.

Student’s name: Ameya S Chamnikar

This work and its defense approved by:

Committee chair: Manish Kumar, Ph.D.
Committee member: Tamara Lorenz, Ph.D.
Committee member: David Thompson, Ph.D.
Genetic Algorithm Based Trajectory Generation and Inverse Kinematics Calculation for Lower Limb Exoskeleton

A thesis submitted to the Graduate School of the University of Cincinnati in partial fulfillment of the requirements for the degree of Master of Science in the Department of Mechanical and Materials Engineering.

Ameya Sanjay Chamnikar
B.Tech. Dr. Babasaheb Ambedkar Marathwada University, July 2014
Committee Chair: Manish Kumar, Ph.D.
Population of older people aged 60 and above is increasing at an unprecedented rate across all nations. Aging results in decrease joint strength. Joints are essential to perform Activities of Daily Living (ADL). One of the important ADL is Sit-to-Stand (STS). Ankle, Knee and Hip joints are in charge of moving the entire body from sitting posture to standing posture. With aging, joints lose capacity to provide sufficient torque to move the body and hence need support to complete the task. Future prediction for aging population made by United Nations in 2015 shows that there will be 1.4 billion people aged 60 or above and this number will reach 2.1 billion by 2050. With expected lack of personal care, there is dire need to develop devices which will support aging population. We propose the use of lower-limb exoskeleton which provides partial torques at Ankle, Knee and Hip joint and act as an assisting device. This thesis addresses two of the main challenges in the implementation of lower limb exoskeletons: i) to generate human like reference trajectory for Centre of Mass (COM), and ii) inverse kinematics calculations to calculate individual joint angles trajectory based on the COM information. The proposed solution uses Genetic Algorithm (GA) to calculate the reference trajectory for COM with fitness function that takes into account the the fact that, for stable STS transition, the COM of the entire body should lie within outermost boundaries of the footprint of the person. Whereas, fitness function for GA designed to carry out the the inverse kinematics calculations uses squared difference between joint angles for successive steps and modulus of difference between COM position calculated using the same angles. A simple controller was implemented to then track the obtained trajectory using 3 Degree of Freedom (DOF) exoskeleton. Using
Euler-Lagrange method, dynamics for a lower limb exoskeleton approximated as an inverse 3 DOF pendulum was derived. A highly non linear dynamics was then linearized using an input-output feedback control system. Then this linearized dynamics was controlled using a PD controller. The performance of the controller was validated using simulation results. Results shows that GA successfully generated the COM reference trajectory and calculated individual joint angle trajectories and controller was able to track the reference trajectories successfully.

Thesis Supervisor: Manish Kumar

Title: Associate Professor
To my loving parents and grandparents.
Acknowledgments

First, I would like to acknowledge and express my gratitude and respect to my advisor, Dr. Manish Kumar. I am thankful to him for his time and knowledge. He is definitely one of the most influential person in my life.

I would also like to thank my committee members Dr. Tamara Lorenz and Dr. David Thompson for taking the time from their busy schedules to give me the much needed advice and direction to complete this project. They have always motivated me towards my research and bring out the maximum outcome.

It was an awesome experience to be a part of Cooperative Distributed Systems Lab. I owe special thanks to my lab seniors Alireza Nemati, Mohammad Sarim, Mohammadreza Radmanesh and Gaurav Patil for their help and support during our collaborative research work. My lab mates Rumit Kumar, Suyash Kulkarni, Aditya Deshpande, Shraddha Barawkar, Siddharth Sridhar, Paul Guentert Hans, Alekhya Akkapeddi and Amanda Miller deserve a special thanks for creating such a friendly lab environment where we could work on our respective research and enjoy at the same time.

I am extremely thankful to my friends and family members for their support and encouragement.

Finally, I want to dedicate my graduate research thesis to my parents and grand parents. I thank them for their blessing and support.

Ameya Chamnikar
# Contents

**Abstract** iii

**Acknowledgments** vi

**Contents** vii

**List of Figures** ix

## 1 Introduction 1

1.1 Motivation ................................. 1
1.2 Exoskeleton ................................. 2
1.3 Sit-to-Stand ................................. 3
1.4 Problem Formulation ....................... 7
1.5 Objectives ................................. 7
1.6 Contributions ............................... 7
1.7 Organization ............................... 8

## 2 Lower Limb Exoskeleton: Literature Review 9

## 3 Human-Exoskeleton System Dynamics 16

3.1 Assumptions ............................... 16
3.2 Derivation ................................. 16
3.3 Centre of Mass ............................. 24

## 4 Control Strategy 26

4.1 Compensation of Non-linear Terms .......... 26
4.2 Application ............................... 26
6-6 Mutation ................................................................. 37

7-1 Performance of GA for CASE I .................................................. 43
7-2 Convergence of Fitness Function for CASE I ................................. 44
7-3 Comparison of Actual Joint Angle Trajectories Vs Joint Angle Trajectories Generated Using GA for CASE I .................... 45
7-4 Comparison of Actual COM Trajectory Vs COM Trajectory Generated Using GA for CASE I ............................ 46
7-5 Output of PD Controller for CASE I ........................................... 47
7-6 Comparison of Actual Joint Torque Trajectories Vs Joint Torque Trajectories Generated Using GA for CASE I .................... 47
7-7 Performance of GA for CASE II ............................................... 49
7-8 Convergence of Fitness Function for CASE II .............................. 49
7-9 Comparison of Actual Joint Angle Trajectories Vs Joint Angle Trajectories Generated Using GA for CASE II .................... 50
7-10 Comparison of Actual COM Trajectory Vs COM Trajectory Generated Using GA for CASE II ............................ 51
7-11 Output of PD Controller for CASE II ........................................... 51
7-12 Comparison of Actual Joint Torque Trajectories Vs Joint Torque Trajectories Generated Using GA for CASE II .................... 52
7-13 Comparison of Actual Joint Angle Trajectories Vs Joint Angle Trajectories Obtained by Inverse Kinematics for CASE I .......... 53
7-14 Comparison of Actual COM Trajectory vs COM Trajectory Obtained from Joint Angles Obtained by Inverse Kinematics for CASE I ................. 54
7-15 Difference Between Actual Joint Angle Trajectories and Joint Angle Trajectories Obtained by Inverse Kinematics for CASE I ................. 54
7-16 Comparison of Actual Joint Angle Trajectories Vs Joint Angle Trajectories Obtained by Inverse Kinematics for CASE II ................. 55
7-17 Comparison of Actual COM Trajectory vs COM Trajectory Obtained from Joint Angles Obtained by Inverse Kinematics for CASE II ................. 56
Difference Between Actual Joint Angle Trajectories and Joint Angle Trajectories Obtained by Inverse Kinematics for CASE II
Chapter 1

Introduction

Population of people aged 60 and above is increasing rapidly. In the world, total number of people aged 60 and above is projected to be 1.4 billion by the year 2030 which is approximately 56% increase from current 901 million and by the year 2050, the population will reach 2.1 billion, almost double of current size [1]. The number of home care providers, nursing homes etc. are not sufficient to take care of such a large population. One of the main activities of daily living (ADL) is sit-to-stand (STS) transition. Ankle, Knee and Hip joints are involved in STS and provide necessary torque during STS transition. With aging, joints lose capacity to provide necessary torque to complete the STS activity.

1.1 Motivation

Rise in older population age 60 and above and deficiency in number of personal care providers created necessity to develop new technology that can support ADL in particular STS. There is plenty of research on lower limb exoskeleton for walking but in case of STS the problem becomes more complex due to inherent unstable transition. Also, most of the algorithms available uses tracking cameras, Electromyography (EMG) or predefined trajectories for each joint as reference trajectory. Also, human tend to follow the trajectory that achieves stability early on during the transition. This makes it difficult to generate reference trajectories for each individual.
1.2 Exoskeleton

In the context of this research, an exoskeleton is an external robotic structure that provides necessary support torque during STS transition. Fig. 1-1 shows the overall link and motor assembly for the exoskeleton. In this study only half of the exoskeleton as shown in Fig. 1-1 is modeled.

Figure 1-1: Exoskeleton

Exoskeletons are used for rehabilitation and assistance for different gait patterns such as walking, STS etc. While aging the joints fall short of required torque for STS or even walking. Hence, older people need some support to do their daily chores involving ankle, knee and hip joints. Traditionally, people use a cane to support the weight of the body and provide torque just enough to move the body. This creates a hindrance if the activity involves ankle, knee and hip joint movement and also use of hand may be to hold something or to throw something. Exoskeletons which are capable of providing the required torque or part of required torque can allow an older human being to perform multiple activities simultaneously. This makes exoskeletons of a high research interest among researchers.
### 1.3 Sit-to-Stand

Sit-to-Stand (STS) is most important ADLs. STS activity involves coordination between torque applied by ankle, knee and hip joints. STS can be independent activity or augmented with other activities for example STS and grab something, STS and move to different location and then sit again etc. Fig. 1-2 and 1-3 shows sitting and standing position.

**Figure 1-2: Sitting Position**

Sitting position is defined as the resting position in which shank and trunk makes 90° with horizontal axis whereas thigh link makes 180° with horizontal axis. Standing position is when all three links makes 90° with horizontal axis as depicted in Fig. 1-3. As can be seen from following two figures, centre of mass of the body changes position and plays an important role in stability of the whole body. The body experiences unbalanced torques due to gravity if the centre of mass is outside the boundaries of feet. X axis shows horizontal width in meters where as Y axis shows the height of the person in meters during sitting and standing position. Three links represent shank (green), thigh (blue) and entire body above waist joint (red). The black dot represents approximate location COM of the body. The links shown have mass and moment of inertia.
Fig. 1-3 shows the joint angle trajectories for ankle knee and hip during STS. These angle trajectories were recorded using OptiTrack system. The subject performing STS was healthy human being. These angle trajectories are recorded using OptiTrack system and subject is
healthy human being. As seen from the Fig. 1-4, it is clear that each joint angle trajectory is unique during STS and serves a specific purpose that helps human to perform a smooth and stable transition.

Also, from Fig. 1-4, knee angle always keeps on decreasing till it reaches final value of \( \pi/2 \). On the other hand ankle and hip angle have tendency to decrease in the beginning and then increase during the second half of the STS. This happens due to human body trying to move COM inside the outermost boundary of the feet to make the body stable first and then move towards the standing position[2].

![Intermediate Positions During STS](image)

**Figure 1-5: Transition During STS**

Fig. 1-5 shows different intermediate postures during STS. It is very interesting to see how the body does not choose to go for least torque trajectory, but chooses the one that guarantees stability. In the Fig. 1-5 posture 1 and 5 are initial and final positions. Posture 2 shows the body starting to lean forward. Posture 3 corresponds to seat off which is defined as posture at which body looses contact with the chair or object and transfer of weight from chair to feet happens. Around seat off posture along with link 3, link 1 also tries to lean forward. As COM reaches within outermost boundary of the feet, then body goes towards standing position depicted by posture 4 in Fig. 1-5.
Fig. 1-6 represents existence of multiple solutions for same end effector position. One possible end effector position that can be defined with $\theta_1$, $\theta_2$, and $\theta_3$ or $\theta_1'$, $\theta_2'$, and $\theta_3'$. For 3 link mechanism, forward kinematics will always yield one unique end effector position for different joint angles except for singularities. For the same end effector position there exist multiple sets of joint angles. This creates ambiguity in terms of selection of a particular set. As can be seen from the following figure $\theta_1$, $\theta_2$, and $\theta_3$ are part of one solution set whereas $\theta_1'$, $\theta_2'$, and $\theta_3'$ are part of second solution set for the same end effector position. In some cases, there could be more than two solution sets making inverse kinematics problem hard to solve. Especially when the inverse kinematics is going to be used for a device which is going to interact with human, safety becomes a major concern because of existence of multiple solution. For instance, if a solution set contains two sets of solution. Set one contains value of $\theta_1$ and $\theta_3$ between $(0$ to $\pi/2)$ and value of $\theta_2$ between $(\pi/2$ to $\pi)$. Similarly, set two contains value of $\theta_1$ and $\theta_3$ outside of $(0$ to $\pi/2)$ and value of $\theta_2$ between $(\pi/2$ to $\pi)$. It is clear that, based on human joint constraints set one is valid whereas set two is invalid. Algorithm calculating inverse kinematics for the devices used with human interaction needs to be robust enough so that they do not select an
invalid solution which may harm user.

1.4 Problem Formulation

Existing process to generate reference trajectory is by attaching number of markers to a healthy human body and then recording the marker position using motion tracking system. It is an expensive and time consuming process. Also, online inverse kinematics solver remains a challenge in lower limb exoskeleton area because of the fact that there is no single solution set of joint angles for any end effector position. The existing methods do not provide portability because of large large motion tracking setup and inverse kinematics control using COM as point of control.

1.5 Objectives

For this thesis, the objectives are listed as follows:

- Deriving mathematical model of human - robot system approximated as inverse 3-DOF system.
- Generating reference COM trajectory using GA by fitting a 5th degree polynomial.
- Deriving individual joint angle trajectories by doing Inverse kinematics using GA.
- Developing a PD controller to track the reference trajectory generated using GA.

1.6 Contributions

The main contribution of this thesis is developing a fitness function for GA. GA is then used to find a coefficient of the 5th degree polynomial which closely resembles actual human joint angle trajectories. Once the polynomial is found for all three joints then a controller is designed to closely follow these reference trajectories. The output of the controller then can be manipulated based on the requirement. For example, controller can provide 100% of the torque
or partial torque of the total required torque. The entire process is divided in two phases. Phase I is to input all link lengths (length of human’s shank, thigh and entire body above waist), link weights and moment of inertia. Once all the parameters are inputted, GA generates reference trajectory for controller to follow.

Second part of this thesis deals with inverse kinematics problem. A good control for STS is COM of the body. Again, GA is used to find individual joint angle trajectories from COM location. Two cases with different search constraints are presented. Case I assumes the constraints on search space similar to respective human joints for each iteration. Whereas, case II assumes $\pm 0.2$ rad constraint on each joint for each iteration because the change in COM is very small for consecutive steps and it is safe to assume that for such a small change in COM is not going to cause abrupt change in joint angles. This allowed us to speed up the process of calculating inverse kinematics.

1.7 Organization

This thesis is organized as follows: Chap. 2 provides a brief overview of the literature available that formed the basis of this research. Chap. 3 introduces the exoskeleton, mathematical model of the exoskeleton and derivation of dynamics of the exoskeleton using Lagrange’s equation. Chap. 4 presents brief overview of Genetic Algorithm, development of fitness function used in this thesis. Chap. 5 describes working of control strategy that tracks the reference trajectory to complete STS. Chap. 6 explains the inverse kinematics concept and how GA is used to calculate acceptable individual joint angle trajectories using COM trajectory. Chap. 7 discusses the simulation results generated using Matlab. Chap. 8 talks about conclusion and future work.
Chapter 2

Lower Limb Exoskeleton: Literature Review

Considering the recent trends for aging population and future projection in home health care is gaining more attention. One of the main task for care providers is to lift the person from sitting position to standing position and vice versa. Depending on weight of the person the task can be painful. Hence there is huge potential in development of assistive devices which help care provider or human. The main goal is to ease the pain of care provider as well as provide required assistance to human for completing STS. Lower limb exoskeletons and lifting devices have potential to provide such assistance to aging population.

Oymagil et al. [3,4,5,6,7,8] talks about regenerative braking powered ankle foot orthosis. The setup shown in Fig. 2-1 consist of a motor, spring, lever and a structure that wraps around foot and shank. Motor manipulates spring in a controlled fashion.

Spring actuates lever which eventually moves foot and shank orthosis in the direction depending on whether the spring is compressing or expanding. The controller is preprogrammed with desired gait pattern to follow. This desired (reference) gait pattern is essentially trajectories tracked by a healthy human subjects. This reference trajectory is then divided in multiple segments to decide input to the motor. This research provides a novel idea of regenerative braking power which is aimed at reducing power used by exoskeleton. This reduction in power consumption makes exoskeleton portable. Also, based on time between two successive heel strikes algorithm predicts future trend for gait and adjusts input frequency.
During the previous research on exoskeleton, Varol et al. [9] talks about the input as a sum of active and passive torque components as shown in Fig. 2-2. Varol et al.’s research focuses on knee and ankle joint. Instead of using echo control the research explores the area where two different parts of torques are combined together to get the final input. Also, it uses mechanical sensors to get the feedback of force and torque applied by the user. This feedback is then used as an input to generate active part of the torque. The gait pattern is divided in segments to identify which segment uses passive torque and which one will store energy which then later can be extracted to provide passive part of the torque. However this research does not validates the algorithm with exoskeleton that provides support for hip joints. Moreover, expensive sensors are involved in feedback loop.

Shomin et al. [10] introduces an idea of self balancing robot to assist STS as oppose to strapping an exoskeleton to an user. The robot can tilt to a maximum of 15° of angle, can exert 120N of force and has its own set of hands for user to hold. It is a novel approach to
solve the problem as it does not involve strapping an exoskeleton to the user’s body. Fig. 2-3 shows the setup. Experimentally found force trajectories are given as an input to the robot’s controller which then based on mathematical formula determines required leaning angle. This leaning angle is actual input to the motors of the robot. As robot leans backwards when facing user, it generates pulling force which pulls the user holding end of the robot’s arm. Since each individual will have different speed for STS, a trajectory tracking control for force trajectory may not be feasible. Also, for identification of user’s posture at any given point during STS, shoulder joint is used as a cue rather than popular hip, knee or angle joint angles. A basic impedance based controller is implemented which basically takes shoulder position in xz plane in account while calculating output of the controller.

![Figure 2-2: Structure of the transformal prosthesis control system](image)

While the idea of using a self balancing robot is very unique and interesting, it limits the usage of robot to very specific type of surfaces. On rough surfaces the performance may deteriorate. If there is system failure then robot and user may fall on top of each other depending on where is the COM of the entire system at the time of failure. The STS for each individual varies greatly from person to person. Since the robot uses preprogrammed force, trajectories it may not be able take care of inherent variation in the STS. Fig. 2-4 shows intermediate position during STS. It can be seen from the figure that the trunk of the human being remains almost straight during entire STS which is completely opposite to actual STS. Human tends to lean the trunk forward first and then rotate knee and ankle joints to lift the body. Since the STS transition obtained by using robot is not natural it may not be comfortable after extensive use. Also, unnatural joint angle trajectories may not allow the use of this robot for rehabilitation.
Figure 2-3: Self Balancing Robot Pulling the User to Perform STS

Figure 2-4: Intermediate Postures During STS Using the Self Balancing Robot

On the similar subjects, Pasqui et al. [11] talks about fitting a curve for robot’s handle to follow. The setup consist of a 2-DOF system mounted on an active robotic platform as shown
in Fig. 2-5. It is a novel setup in terms of its simplicity for application purposes. It is important to note that the arrangement of system is such that the handle position is near to the thigh of the user in the beginning and user needs to apply vertical force on the handle. When the handle follows a particular trajectory the user performs STS with assistance from the system. Since the user needs to provide a vertical force on handle, the body above waist joint will remain upright during entire STS, which is opposite to what a healthy human tries to do. Since the robotic system is providing necessary support to balance the system in the transition phase there is no problem of user falling due to instability. However, such STS can be uncomfortable for elderly people.

![System Setup for Robot-Human Cooperation](image)

A trajectory is proposed for handle to follow to reduce the jerks and obtain a smooth trajectory for the handle of the system. The equation is as follows:

$$X(t) = X_i + (X_f - X_i) \times (10 \times (t/T)^3 + 15 \times (t/T)^4 + 6 \times (t/T)^6)$$  \hspace{1cm} (2.1)

It can be seen from the above equation that if you differentiate above equation twice with respect to time, the result will be a second-degree curve in time which will ensure a smooth transition between acceleration and deceleration and also no there will be no jerks since the curve is differentiable for all possible values of $X$. 

---

13
Fig. 2-6 shows the trajectory generated for the handle. As can be seen from the figure it is clear that the trajectory forces handle to go in positive x direction, which causes lift off and then change in Y coordinate of the trajectory in positive direction surpasses the change in X coordinate of the trajectory in positive direction making user stand up completely. Fig. 2-7 shows Fig. 2-6 superimposed on Fig. 2-5. Fig. 2-7 shows how handle is going to travel and thereby moving user along with it. Since the handle goes in positive x-direction first and then in positive y-direction, the user comes close to what healthy humans tries to do during STS. It forces the COM of human to move closer to outermost boundaries of the feet which makes sure that the human body achieves stable posture. The second half of the trajectory moves the user in positive y-direction which makes the user stand. This makes it a novel approach to solve the problem that allows user to follow a trajectory that comes close to actual healthy human trajectory for STS without any assist or support.
Even though this comes close to what healthy human tries to do during actual STS, it lacks the actual rotation of Hip joint which is very essential during STS. It forces the human to hold the handle and apply force in vertical direction and not in horizontal direction. On one hand the system ensures that user does not get dragged because of horizontal force application but it fails to allow user to lean forward which a healthy human tries to do. This makes it uncomfortable for the user to perform STS with unnatural trajectory. It is a good replacement for crutches in a sense that it allows lift off without human needing to provide any joint torque, since the robotic setup provides it. For rehabilitation and assistance purpose there needs to be substantial improvement for this setup to be acceptable. The changes can be in terms of mechanical changes to the setup so that it allows human to track natural trajectory or it could be in terms of trajectory for handle so that it forces human to track natural trajectory.
Chapter 3

Human-Exoskeleton System Dynamics

3.1 Assumptions

For successful completion of STS, human needs to actuate three joints: Hip, Ankle and Knee. The appropriate torque provided at each joint allows human to stand up. The joint torque requirement at each joint varies from joint to joint primarily because each joint contributes to the STS for a unique reason and carries different amount of load. Also, the load is not uniform since humans have flexible joints and more than one degrees of freedom at each joint. For example, in case of back, spinal cord is actually a linkage formed by small joints which allows the spinal cord to bend. This makes hard to model the body above waist joint since it is not stiff and each joint need to be considered as a separate entity during modelling. For the purpose of this study, human body is considered as a 3 rigid link system. One link is between ankle and knee, the second between knee and hip and the third link is entire body above waist. To make the dynamics simple, the links are considered rigid, that means there is no compression in link or links do not bend.

3.2 Derivation

To derive dynamics for the 3 link system, Euler-Lagrange’s equation is used. Torque calculation using Euler-Lagrange’s equation is done using following formula for torque:

\[ \tau = \frac{d}{dt} \left( \frac{\partial L}{\partial \dot{\theta}} \right) - \frac{\partial L}{\partial \theta} \]  

(3.1)
Where, $\mathcal{L}$ is algebraic difference between kinetic energy and potential energy.

$$\mathcal{L} = K.E. - P.E. \tag{3.2}$$

Where, $K.E.$ is Kinetic Energy and $P.E.$ is Potential Energy. Potential Energy ($P.E.$) of the system is calculated based on mass of each link and height of the centre of mass from datum.

$$P.E. = m_1gh_1 + m_2gh_2 + m_3gh_3 \tag{3.3}$$

Where, $h_1, h_2$ and $h_3$ distance of center of mass of link 1,2 and 3 respectively with respective to ground. From Fig. 3-1 we can establish following relations between $h_1, h_2, h_3$ and $l_1, l_2, l_3$.

$$h_1 = l_1 \sin(\theta_1)$$

$$h_2 = L_1 \sin(\theta_1) + l_2 \sin(\theta_2)$$

$$h_3 = L_1 \sin(\theta_1) + L_2 \sin(\theta_2) + l_3 \sin(\theta_3)$$
On substituting these values in Eqn. 3.3

\[ P.E. = m_1 g l_1 \sin(\theta_1) + m_2 g (L_1 \sin(\theta_1) + l_2 \sin(\theta_2)) + m_3 g (L_1 \sin(\theta_1) + L_2 \sin(\theta_2) + l_3 \sin(\theta_3)) \]  

(3.4)

After simplifying Eqn. 3.4 we get,

\[ P.E. = (m_1 l_1 + m_2 L_1 + m_3 L_1) g \sin(\theta_1) + (m_2 l_2 + m_3 L_2) g \sin(\theta_2) + m_3 l_3 g \sin(\theta_3) \]  

(3.5)

The K.E. of the system is sum of linear and rotational kinetic energy of each link. The equation for systems’s K.E. can be written as follows,

\[ K.E. = \frac{1}{2}(m_1 v_1^2 + m_2 v_2^2 + m_3 v_3^2 + I_1 \dot{\theta}_1^2 + I_2 \dot{\theta}_2^2 + I_3 \dot{\theta}_3^2) \]  

(3.6)

Where, \( v_1, v_2 \) and \( v_3 \) are linear velocities of center of mass of link 1,2 and 3 respectively. To find the equation of velocity for each link, we first need to find velocity of each link along x and y axis. Square root of sum of squares of velocities along x and y axis will give linear velocity of each link. The calculation is as follows:

\( x_1, x_2 \) and \( x_3 \) are displacements of each center of mass along x-axis. Similarly, \( y_1, y_2 \) and \( y_3 \) are displacement of each center of mass along y-axis. From Fig. 3-1 we can write following equations for \( x_1, x_2, x_3, y_1, y_2 \) and \( y_3 \).

\[
\begin{align*}
y_1 &= l_1 \sin(\theta_1), \\
y_2 &= L_1 \sin(\theta_1) + l_2 \sin(\theta_2), \\
y_3 &= L_1 \sin(\theta_1) + L_2 \sin(\theta_2) + l_3 \sin(\theta_3), \\
x_1 &= l_1 \cos(\theta_1) \\
x_2 &= L_1 \cos(\theta_1) + l_2 \cos(\theta_2) \\
x_3 &= L_1 \cos(\theta_1) + L_2 \cos(\theta_2) + l_3 \cos(\theta_3)
\end{align*}
\]

Differentiating the displacement with respect to time gives velocities. Differentiating all six displacements with respect to time gives 6 different linear velocity equations as follows:
\[
\frac{d}{dt}(y_1) = l_1 \frac{d}{dt}(\sin(\theta_1)) \quad (3.7)
\]

\[
y_1 = l_1 \cos(\theta_1)\dot{\theta}_1 \quad (3.8)
\]

\[
\frac{d}{dt}(y_2) = L_1 \frac{d}{dt}(\sin(\theta_1)) + l_2 \frac{d}{dt}(\sin(\theta_2)) \quad (3.9)
\]

\[
y_2 = L_1 \cos(\theta_1)\dot{\theta}_1 + l_2 \cos(\theta_2)\dot{\theta}_2 \quad (3.10)
\]

\[
\frac{d}{dt}(y_3) = L_1 \frac{d}{dt}(\sin(\theta_1)) + L_2 \frac{d}{dt}(\sin(\theta_2)) + l_3 \frac{d}{dt}(\sin(\theta_3)) \quad (3.11)
\]

\[
y_3 = L_1 \cos(\theta_1)\dot{\theta}_1 + L_2 \cos(\theta_2)\dot{\theta}_2 + l_3 \cos(\theta_3)\dot{\theta}_3 \quad (3.12)
\]

\[
\frac{d}{dt}(x_1) = l_1 \frac{d}{dt}(\cos(\theta_1)) \quad (3.13)
\]

\[
x_1 = -l_1 \sin(\theta_1)\dot{\theta}_1 \quad (3.14)
\]

\[
\frac{d}{dt}(x_2) = L_1 \frac{d}{dt}(\cos(\theta_1)) + l_2 \frac{d}{dt}(\cos(\theta_2)) \quad (3.15)
\]

\[
x_2 = -(L_1 \sin(\theta_1)\dot{\theta}_1 + l_2 \sin(\theta_2)\dot{\theta}_2) \quad (3.16)
\]

\[
\frac{d}{dt}(x_3) = L_1 \frac{d}{dt}(\cos(\theta_1)) + L_2 \frac{d}{dt}(\cos(\theta_2)) + l_3 \frac{d}{dt}(\cos(\theta_3)) \quad (3.17)
\]

\[
x_3 = -(L_1 \sin(\theta_1)\dot{\theta}_1 + L_2 \sin(\theta_2)\dot{\theta}_2 + l_3 \sin(\theta_3)\dot{\theta}_3) \quad (3.18)
\]
By squaring and adding linear velocities along x and y axis we get,

\[ v_1^2 = x_1^2 + y_1^2 \]  

(3.19)

By substituting the values of \( x_1 \) and \( y_1 \) we get,

\[ v_1^2 = l_1^2 \dot{\theta}_1^2 \]  

(3.20)

\[ v_2^2 = x_2^2 + y_2^2 \]  

(3.21)

By substituting the values of \( x_2 \) and \( y_2 \) we get,

\[ v_2^2 = L_1^2 \dot{\theta}_1^2 + L_2^2 \dot{\theta}_2^2 + 2L_1L_2 \dot{\theta}_1 \dot{\theta}_2 \cos(\theta_1 - \theta_2) \]  

(3.22)

\[ v_3^2 = x_3^2 + y_3^2 \]  

(3.23)

By substituting the values of \( x_3 \) and \( y_3 \) we get,

\[ v_3^2 = L_1^2 \dot{\theta}_1^2 + L_2^2 \dot{\theta}_2^2 + L_3^2 \dot{\theta}_3^2 + 2L_1L_2 \dot{\theta}_1 \dot{\theta}_2 \cos(\theta_1 - \theta_2) + 2L_1L_3 \dot{\theta}_1 \dot{\theta}_3 \cos(\theta_1 - \theta_3) + 2L_2L_3 \dot{\theta}_2 \dot{\theta}_3 \cos(\theta_2 - \theta_3) \]  

(3.24)

Substituting values of \( v_1^2, v_2^2 \) and \( v_3^2 \) in Eqn. 3.6 we get,

\[
K.E. = 1/2(m_1L_1^2 + m_2L_1^2 + m_3L_1^2 + I_1)\dot{\theta}_1^2 + 1/2(m_2L_2^2 + m_2L_2^2 + I_2)\dot{\theta}_2^2 + 1/2(m_3L_3^2 + I_3)\dot{\theta}_3^2 + L_1(m_2L_2 + m_3L_3)\dot{\theta}_1 \dot{\theta}_2 \cos(\theta_1 - \theta_2) + \\
+ L_2(m_3L_3 + I_3)\dot{\theta}_1 \dot{\theta}_3 \cos(\theta_1 - \theta_3) + m_3L_1L_3\dot{\theta}_1 \dot{\theta}_3 \cos(\theta_1 - \theta_3) + m_3L_2L_3\dot{\theta}_2 \dot{\theta}_3 \cos(\theta_2 - \theta_3) 
\]  

(3.25)
Substituting Eqn. [3.5] and Eqn. [3.25] in Eqn. [3.2], we get,

\[
L = 1/2(m_1l_1^2 + m_2L_1^2 + m_3L_1^2 + I_1)\dot{\theta}_1^2 + 1/2(m_2l_2^2 + m_3L_2^2 + I_2)\dot{\theta}_2^2 + 1/2(m_3l_3^2 + I_3)\dot{\theta}_3^2 + L_1(m_2l_2 + m_3L_2)\dot{\theta}_1\dot{\theta}_2 \cos(\theta_1 - \theta_2) + m_3L_1l_3\dot{\theta}_1\dot{\theta}_3 \cos(\theta_1 - \theta_3) + m_3L_2l_3\dot{\theta}_2\dot{\theta}_3 \cos(\theta_2 - \theta_3) - (m_1l_1 + m_2L_1 + m_3L_1)g \sin(\theta_1) - (m_2l_2 + m_3L_2)g \sin(\theta_2) - m_3l_3g \cos(\theta_3)
\]

Calculating all partial and total derivatives with respect to variables as required by Eqn. [3.1]

\[
\frac{\partial L}{\partial \theta_1} = -L_1(m_2l_2 + m_3L_2)\dot{\theta}_1\dot{\theta}_2 \sin(\theta_1 - \theta_2) - m_3L_1l_3\dot{\theta}_1\dot{\theta}_3 \sin(\theta_1 - \theta_3) - (m_1l_1 + m_2L_1 + m_3L_1)g \cos(\theta_1) 
\]

\[
3.27
\]

\[
\frac{\partial L}{\partial \theta_2} = -L_1(m_2l_2 + m_3L_2)\dot{\theta}_1\dot{\theta}_2 \cos(\theta_1 - \theta_2) - m_3L_2l_3\dot{\theta}_2\dot{\theta}_3 \sin(\theta_2 - \theta_3) - (m_2l_2 + m_3L_2)g \cos(\theta_2)
\]

\[
3.28
\]

\[
\frac{\partial L}{\partial \theta_3} = m_3L_1l_3\dot{\theta}_1\dot{\theta}_3 \sin(\theta_1 - \theta_3) + m_3L_2l_3\dot{\theta}_2\dot{\theta}_3 \cos(\theta_2 - \theta_3) - m_3l_3g \sin(\theta_3)
\]

\[
3.29
\]
\[
\frac{\partial L}{\partial \theta_3} = (m_3l_3^2 + I_3)\dot{\theta}_3 + m_3L_1l_3\dot{\theta}_1 \cos(\theta_1 - \theta_3) + m_3L_2l_3\dot{\theta}_2 \cos(\theta_2 - \theta_3) \tag{3.32}
\]

\[
\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_1}\right) = (m_1l_1^2 + m_2L_1^2 + m_3L_1^2 + I_1)\ddot{\theta}_1 + L_1(m_2l_2 + m_3L_2)\ddot{\theta}_2 \cos(\theta_1 - \theta_2) \\
-L_1(m_2l_2 + m_3L_2)\dot{\theta}_2 \sin(\theta_1 - \theta_2)(\dot{\theta}_1 - \dot{\theta}_2) + m_3L_1l_3\dot{\theta}_3 \cos(\theta_1 - \theta_3) \\
-m_3L_1l_3\dot{\theta}_3 \sin(\theta_1 - \theta_3)(\dot{\theta}_1 - \dot{\theta}_3) \tag{3.33}
\]

\[
\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_2}\right) = (m_2l_2^2 + m_2L_2^2 + I_2)\ddot{\theta}_2 + L_1(m_2l_2 + m_3L_2)\ddot{\theta}_1 \cos(\theta_1 - \theta_2) \\
+m_3L_2l_3\dot{\theta}_3 \cos(\theta_2 - \theta_3) - m_3L_2l_3\dot{\theta}_3 \sin(\theta_2 - \theta_3)(\dot{\theta}_2 - \dot{\theta}_3) \\
-L_1(m_2l_2 + m_3L_2)\dot{\theta}_1 \sin(\theta_1 - \theta_2)(\dot{\theta}_1 - \dot{\theta}_2) \tag{3.34}
\]

\[
\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_3}\right) = (m_3l_3^2 + I_3)\ddot{\theta}_3 + m_3L_1l_3\ddot{\theta}_1 \cos(\theta_1 - \theta_3) - m_3L_1l_3\dot{\theta}_1 \sin(\theta_1 - \theta_3)(\dot{\theta}_1 - \dot{\theta}_3) \\
+m_3L_2l_3\ddot{\theta}_2 \cos(\theta_2 - \theta_3) - m_3L_2l_3\ddot{\theta}_2 \sin(\theta_2 - \theta_3)(\dot{\theta}_2 - \dot{\theta}_3) \tag{3.35}
\]

Substituting the values in Eqn. (3.1) we get,

\[
\tau_1 = (m_1l_1^2 + m_2L_1^2 + m_3L_1^2 + I_1)\ddot{\theta}_1 + L_1(m_2l_2 + m_3L_2)\ddot{\theta}_2 \cos(\theta_1 - \theta_2) \\
+m_3L_1l_3\dot{\theta}_3 \cos(\theta_1 - \theta_3) + L_1(m_2l_2 + m_3L_2)\dot{\theta}_2^2 \sin(\theta_1 - \theta_2) \\
+m_3L_1l_3\dot{\theta}_3^2 \sin(\theta_1 - \theta_3) + (m_1l_1 + m_2L_1 + m_3L_1)g \cos(\theta_1) \tag{3.36}
\]

\[
\tau_2 = (m_2l_2^2 + m_2L_2^2 + I_2)\ddot{\theta}_2 + L_1(m_2l_2 + m_3L_2)\ddot{\theta}_1 \cos(\theta_1 - \theta_2) \\
+m_3L_2l_3\dot{\theta}_3 \cos(\theta_2 - \theta_3) - L_1(m_2l_2 + m_3L_2)\dot{\theta}_1^2 \sin(\theta_1 - \theta_2) \\
+m_3L_2l_3\dot{\theta}_3^2 \sin(\theta_2 - \theta_3) + (m_2l_2 + m_3L_2)g \cos(\theta_2) \tag{3.37}
\]
\[ \tau_3 = (m_3 l_3^2 + I_3) \ddot{\theta}_3 + m_3 L_1 l_3 \dot{\theta}_1 \cos(\theta_1 - \theta_3) + m_3 L_2 l_3 \dot{\theta}_2 \cos(\theta_2 - \theta_3) - m_3 L_1 l_3 \dot{\theta}_1^2 \sin(\theta_1 - \theta_3) - m_3 L_2 l_3 \dot{\theta}_2^2 \sin(\theta_2 - \theta_3) + m_3 l_3 g \cos(\theta_3) \] (3.38)

These torque equations can be rewritten in terms of matrix as follows:

\[ [\tau_i]_{3 \times 1} = B \dot{\theta}_i + [C(\theta, \dot{\theta})]_{3 \times 1} + [G_\theta]_{3 \times 1} \] (3.39)

Where,

\[ [\tau_i]_{3 \times 1} = [\tau_1; \tau_2; \tau_3] \] (3.40)

\[ [\dot{\theta}]_{3 \times 1} = [\dot{\theta}_1; \dot{\theta}_2; \dot{\theta}_3] \] (3.41)

\[ B_{\theta,1} = (m_1 l_1^2 + m_2 L_1^2 + m_3 l_3^2 + I_1) \]
\[ B_{\theta,2} = L_1 (m_2 l_2 + m_3 L_2) \cos(\theta_1 - \theta_2) \]
\[ B_{\theta,3} = m_3 L_1 l_3 \cos(\theta_1 - \theta_3) \]
\[ B_{\theta,2} = L_1 (m_2 l_2 + m_3 L_2) \cos(\theta_1 - \theta_2) \]
\[ B_{\theta,3} = (m_2 l_2^2 + m_2 L_2^2 + I_2) \] (3.42)

\[ B_{\theta,3} = m_3 L_2 l_3 \cos(\theta_2 - \theta_3) \]
\[ B_{\theta,1} = m_3 L_1 l_3 \cos(\theta_1 - \theta_3) \]
\[ B_{\theta,2} = m_3 L_2 l_3 \cos(\theta_2 - \theta_3) \]
\[ B_{\theta,3} = (m_3 l_3^2 + I_3) \]

\[ C(\theta, \dot{\theta})_{1 \times 1} = L_1 (m_2 l_2 + m_3 L_2) \dot{\theta}_2^2 \sin(\theta_1 - \theta_2) + m_3 L_1 l_3 \dot{\theta}_3^2 \sin(\theta_1 - \theta_3) \]
\[ C(\theta, \dot{\theta})_{2 \times 1} = -L_1 (m_2 l_2 + m_3 L_2) \dot{\theta}_1^2 \sin(\theta_1 - \theta_2) + m_3 L_2 l_3 \dot{\theta}_3^2 \sin(\theta_2 - \theta_3) \] (3.43)
\[ C(\theta, \dot{\theta})_{3 \times 1} = -m_3 L_1 l_3 \dot{\theta}_1^2 \sin(\theta_1 - \theta_3) - m_3 L_2 l_3 \dot{\theta}_2^2 \sin(\theta_2 - \theta_3) \]
\[ G_{\theta_1} = (m_1l_1 + m_2L_1 + m_3L_1)g \cos(\theta_1) \]
\[ G_{\theta_2} = (m_2l_2 + m_3L_2)g \cos(\theta_2) \]
\[ G_{\theta_3} = m_3l_3g \cos(\theta_3) \]

As can be seen from the Eqn. 3.39 that the dynamics of the system is highly non-linear. Before designing a controller for such system needs to be linearized. This will be discussed in detail in next chapter.

### 3.3 Centre of Mass

The equation for centre of mass for human body and exoskeleton together is calculated using following two equations.

\[
\begin{align*}
COM_x &= (\cos(\theta_0)l_0m_0 + \cos(\theta_1)l_1m_1) + (\cos(\theta_1)L_1 + \cos(\theta_2)L_2)m_2 + (\cos(\theta_1)L_1 + \cos(\theta_2)L_2 + \cos(\theta_3)L_3)m_3)/m \\
COM_y &= (\sin(\theta_0)m_0l_0 + \sin(\theta_1)l_1m_1) + (\sin(\theta_1)L_1 + \sin(\theta_2)L_2)m_2 + (\sin(\theta_1)L_1 + \sin(\theta_2)L_2 + \sin(\theta_3)L_3)m_3)/m
\end{align*}
\]

These equations are derived using the fact that the COM of the entire body is combination of COM of each individual segments. The x-coordinate of COM of each segment is multiplied with mass of that segment. Sum of all such multiplication is then divided by total mass of the body which gives x-coordinate for COM of the entire body. Similar calculations were done for y-coordinate of COM for the body and Eqn. 3.46 and Eqn. 3.47 were obtained. The segmental data used for calculating segment length and mass based on entire body’s height and weight is listed in following table[12][13]:
<table>
<thead>
<tr>
<th>Segment</th>
<th>Total Segment Count</th>
<th>% weight of total body weight</th>
<th>% length of total body height</th>
<th>% distance of COM from proximal end</th>
</tr>
</thead>
<tbody>
<tr>
<td>HAT</td>
<td>1</td>
<td>60.28</td>
<td>40.75</td>
<td>63</td>
</tr>
<tr>
<td>Thigh</td>
<td>2</td>
<td>14.16</td>
<td>23.2</td>
<td>43.4</td>
</tr>
<tr>
<td>Shank</td>
<td>2</td>
<td>4.33</td>
<td>24.7</td>
<td>43.3</td>
</tr>
<tr>
<td>Feet</td>
<td>2</td>
<td>1.37</td>
<td>4.25</td>
<td>50</td>
</tr>
</tbody>
</table>
Chapter 4

Control Strategy

There are various options available to control non-linear systems such as input-output feedback linearization, sliding mode, gain scheduling, backstepping etc. All of these methods have their own advantages and disadvantages. For simplicity in this research feedback linearization control strategy is used.

4.1 Compensation of Non-linear Terms

Compensation of non-linear terms is a method used to linearize the system. First, using the system feedback non-linear terms in the plant equation are calculated. Then, this value is added to any other input to the system and total input is calculated. This input cancels out the non-linearity in the plant and makes it linear.

4.2 Application

Eqn. 3.39 is plant dynamics. We used this equation to design control strategy. As explained in the previous section to linearize a non-linear plant, a synthetic input is designed which is capable of cancelling the non-linear effect of the plant and make it linear. After making plant dynamics linear, we applied PD controller to compensate the error in the system.
4.3 Block Diagram

Fig. 4-1 shows overview of system used during this research. The entire system consists of 4 main blocks. From left to right, first block is an input to the system. The joint angle trajectories generated using Genetic Algorithm and reference trajectory block provides the trajectory generated by Genetic Algorithm to the system for tracking purpose.

![Block Diagram](image)

Figure 4-1: Block Diagram

PD Controller block is a Proportional-Derivative controller block. The function of this block is to take input from reference trajectory block and feedback from plant to calculate error in the system. Once the error is calculated a proportional gain constant is used to multiply the error and a derivative gain constant is used to multiply the derivative of the error. The sum of these two products is output of the "PD Controller" block. This output actually controls any deviation in the system. The proportional and derivative gains were obtained by trial and error method.

\[
K_p = \begin{bmatrix} 14.7 \\ 543.9 \\ 543.9 \end{bmatrix}
\]

\[
K_d = \begin{bmatrix} 654.5 \\ 269.5 \\ 269.5 \end{bmatrix}
\]

Unit for \(K_p\) gains is \(Nm/rad\) and for \(K_d\) gains it is \(NmSec/rad\).
Compensation of Non-linear terms block compensates the non-linearity of the plant model. As discussed in chap. 3 the plant model is highly non linear.

$$\tau_{i3} = B_{\theta i3} [\ddot{\theta}]_{3+1} + [C_\theta(\theta, \dot{\theta})]_{3+1} + [G_\theta]_{3+1}$$

In the above equation $[C_\theta(\theta, \dot{\theta})]_{3+1} + [G_\theta]_{3+1}$ part creates non-linearity in the system. The compensation of non-linear terms block calculates the value of $[C_\theta(\theta, \dot{\theta})]_{3+1} + [G_\theta]_{3+1}$ term in the system dynamics equation. Output of this block is then added to output of PD Controller block to calculate total input to the system. The total input to the system was given by:

$$\tau = B_{\theta i3}(K_p e + K_d \dot{e}) + [C_\theta(\theta, \dot{\theta})]_{3+1} + [G_\theta]_{3+1}$$

Final and most important block is plant block which contains the dynamics of the system. It takes the input from PD Controller block and Compensation of Non-linear terms block and provides angular displacement and angular velocities as output. By equating input to the system to the plant dynamics we got,

$$B_{\theta i3}(K_p e + K_d \dot{e}) + [C_\theta(\theta, \dot{\theta})]_{3+1} + [G_\theta]_{3+1} = B_{\theta i3} [\ddot{\theta}]_{3+1} + [C_\theta(\theta, \dot{\theta})]_{3+1} + [G_\theta]_{3+1}$$

Non-linear terms were cancelled from both sides,

$$B_{\theta i3}(K_p e + K_d \dot{e}) = B_{\theta i3} [\ddot{\theta}]_{3+1}$$

Multiplying both sides by $B^{-1}_{\theta i3}$,

$$B^{-1}_{\theta i3} B_{\theta i3}(K_p e + K_d \dot{e}) = B^{-1}_{\theta i3} B_{\theta i3} [\ddot{\theta}]_{3+1}$$

$$[\ddot{\theta}]_{3+1} = K_p e + K_d \dot{e}$$

Eqn. 4.1 linearized version of the plant was obtained after performing basic mathematical
manipulation. The $\dot{\theta}$ was integrated twice to get angular displacement and angular velocities. Then these values were used to compare against reference values and errors were generated and cycle continues till the final position is reached.

4.4 PD Controller

PD controller stands for proportional-derivative controller. In this controller proportional part uses direct difference between actual state values and desired state values represented by "$e$" in Eqn. [4.1]. Proportional gain constant minimize this error by providing input in the opposite direction of the error equivalent to gain times the error. This may cause an oscillating effect in the system if the gain is not properly tuned.

To damp the possible overshooting and oscillatory behavior caused by proportional block, a derivative part is added. The derivative part of the controller is represented by "$\dot{e}$" which is time derivative of "$e$". So, derivative part can not make error zeros on it’s own because it tries to make time derivative of change in error zero. Which means system with only derivative part will inherently have some steady state error. This effect is nullified by a proportional part.

Hence proportional and derivative part compliment each other and are capable of controlling the system with minimum deviation from the reference values.
Chapter 5

Inverse Kinematics

Inverse kinematics is a method to find joint displacements for a specific point in workspace of the kinematic chain. Whereas inverse kinematics may have multiple solutions for same point in space. Inverse kinematics is very well studied concept and it used for industrial robots, filming etc. The key is to find a solution that lies within the inherent constraints of the mechanism.

The main issue while solving inverse kinematics problem is multiple solutions and choosing the best on out of it. The problem significantly becomes complex if the reference point to calculate inverse kinematics does not lie on the mechanism itself. In case of exoskeletons the
COM of human and exoskeleton does not lie on any of the link for the most part of STS as shown in Fig. 5-1. For first 4 postures the COM does not lie on the mechanism. Also, there is no correlation between COM and any of the end points of the links. This makes it even harder to solve the inverse kinematics problem. Fig. 5-2 shows the COM Space.

![COM Space](image)

Figure 5-2: COM Space

The space was generated using the constraints on the human joints which are $\pi/2$ to 0 for ankle and hip joints and $\pi$ to $\pi/2$ knee joint with increment of 0.1 radians.

Another approach to solve inverse kinematics problem is using heuristic methods. Which means to explore the search space and try to find the closest match for particular point with given set of constraints. Since we are using the result of inverse kinematics for exoskeleton, we used constraints for human joints as constraints for calculation. Human lower limb joints have very peculiar set of constraints. Also there are some combinations of joint angles which are also not possible. Generating space and then employing a search algorithm is costly.
Chapter 6

Genetic Algorithm

Genetic Algorithm (GA) is a well studied concept [14]. GA is a meta-heuristic search algorithm. It is inspired by natural process of selection. When two parents have their offspring, the offspring gets half of the qualities from each parent, which means that there is a good chance that an offspring might get good part of both parents and will evolve. That’s how evolution on earth took place. Offspring that became fit survived and one that lagged behind the evolution process became extinct. The same logic is used in search algorithm that uses GA. A random selection is made out of search space. A fitness function is defined which determines the fitness of each generation. For example, in case of minimization problem the fitness function could be the value of the function that needs to be minimized. Then using fitness function value entire population pool is ordered in ascending or descending order based on minimization or maximization problem.

Fig. 6-1 shows the flow of the logic while using GA. Evolution has two important components namely crossover, mutation attached to it. They play an important role in randomness of the process. Crossover is a process of selecting portion of a parents gene. This selection largely depends on the method used for crossover such as one point crossover, two point crossover, multi point crossover and bitwise crossover. These concepts will be discussed in more details in following sections. Similarly mutation is a process by which a gene or certain genes are altered based on mutation probability. Mutation is necessary to make sure that the algorithm does not get stuck in local minima. In absence of mutation process there is no way that GA can ensure that it achieves global minima for each trial.

In implementation of the Genetic Algorithm the genome of the parents or offsprings is
coded in binary numbers that is 1’s and 0’s. For example, if the search space is from 0 to 100 with two decimal precision then the length of genome will consist of 15 bits of 1’s and 0’s. While calculating fitness function cost this genome will be converted to a decimal number by multiplying $2^x$ where $x$ is the position of the particular bit in the genome. After converting calculation of fitness function becomes easy. After evaluating whether to do another iteration or not the offsprings are arranged in ascending or descending order and the lease fit offspring is omitted from the group that is going to produce next generation. This makes sure survival of the fittest. When the solution converges, all the offsprings will have a fitness function value very close to global minima. GA does not guarantee the same value for different trials but a close enough to the actual value.

![Flow Chart for GA](image)

Figure 6-1: Flow Chart for GA
6.1 Crossover

As discussed earlier, there are various ways of performing crossovers. It largely depends on what is the scope of objective function and complexity of it. Crossovers are nothing but methods to select the genes from each parent. Fig. 6-2 shows one point crossover method. In this method a predefined point divides the entire genome in two parts. This point is used to select one side of one parent and other side of the second parent. Remaining genes of the parents makes up another offspring. Fig. 6-2 shows how the division happens and how each offspring gets the part of each parent.

![1 Point Crossover](image)

**Figure 6-2: One Point Crossover**

Similar technique to that of one point crossover is two point crossovers. In this method everything is same to one point crossover except there are two points that divides the genome in three different sections. Then one section from one parent and two sections of another parent make for genome of one offspring and rest of the genes of parents make up for another offspring. Fig. 6-3 shows the division of parent genome and formation of offspring genome.
Uniform crossover is same as one point and two point crossover in a sense that it also divides the parent genome and produces offsprings using part of parent one and part of parent two. In uniform crossover the ratio for contribution towards each offspring for each parent is
0.5. What that means is that each parent parent will contribute 50% of the gene to produce next generation. Fig. 6-4 shows how uniform crossover is carried out. Basically a random number from 0 to 1 is generated using normal distribution. Each gene of parent is chosen based on the value of this random number. If the value is less than 0.51 then gene from parent 1 genome is selected and if the number is greater than 0.51 then gene from parent 2 genome is selected. This can be reversed also since the distribution is normal. It converges fast because if there are two fit parents producing offspring there is a high probability that one of the spring will have all the good genes from both of the parents.

![Figure 6-5: Crossover](image)

Fig. 6-5 shows how the genome is coded in binary digits. Each of the bit in the figure is multiplied by 2 to the power it's position in the genome from right end. For example, for parent 1, the decimal equivalent can be calculated as follows:

\[
Parent1 = 0 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0
\]

\[
Parent1 = 5966
\]
Same calculations are done for other parent and offsprings. The these decimal values are then used to calculate the fitness function value which determines the fitness of the offsprings and whether they survive to produce next generation.

### 6.2 Mutation

Mutation is a process that adds randomness to the evolution process. The interpretation of this in terms of search problems is that to avoid algorithm getting stuck in local minima the mutation (randomness) is introduced in the system.

![Mutation](image)

In the Fig. 6-6 mutation with mutation probability of 0.05 is shown. Mutation probability of 0.05 means that 5 out of every 100 genes will be altered. In figure the changed gene is circled. Entire genome of the offspring remains same except on in this case. These gene alteration have drastic effect on convergence and optimization capabilities of the algorithm. Higher the mutation probability, higher the randomness. This may lead to alteration of fit offsprings gene which is undesirable since it prolongs the time algorithm takes to converge. A right choice is very important. It is a compromise between convergence time and making sure the algorithm is robust enough so that it does stuck in local minima.
6.3 Application

6.3.1 Trajectory Generation

In this research GA is used to find coefficient of a fifth degree polynomial in time. Eqn. 6.1 shows the polynomial in consideration. The joint angle trajectories can be represented using these polynomials [15].

\[ \theta(t) = at^5 + bt^4 + ct^3 + dt^2 + et + f \] (6.1)

We know the initial and final angular positions, velocity and acceleration. These are tabulated in following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_i )</td>
<td>0</td>
</tr>
<tr>
<td>( \theta_{1i} )</td>
<td>( \pi/2.2 )</td>
</tr>
<tr>
<td>( \theta_{2i} )</td>
<td>( \pi )</td>
</tr>
<tr>
<td>( \theta_{3i} )</td>
<td>( \pi/2.1 )</td>
</tr>
<tr>
<td>( \theta_{1f} )</td>
<td>1.3265</td>
</tr>
<tr>
<td>( \theta_{2f} )</td>
<td>1.6630</td>
</tr>
<tr>
<td>( \theta_{3f} )</td>
<td>( \pi/2 )</td>
</tr>
<tr>
<td>( \dot{\theta}<em>{1i}, \dot{\theta}</em>{2i}, \dot{\theta}_{3i} )</td>
<td>0</td>
</tr>
<tr>
<td>( \dot{\theta}<em>{1f}, \dot{\theta}</em>{2f}, \dot{\theta}_{3f} )</td>
<td>0</td>
</tr>
<tr>
<td>( \ddot{\theta}<em>{1i}, \ddot{\theta}</em>{2i}, \ddot{\theta}_{3i} )</td>
<td>0</td>
</tr>
</tbody>
</table>

Initial time is 0 sec where the simulation starts. In ideal situation initial \( \theta \) values for ankle and hip joint angles are \( \pi/2 \) where as for knee it is \( \pi \). Also final joint angle values should be \( \pi/2 \). Since we recorded a STS performed by healthy human, the initial and final angles were not perfect. To validate we used the output of the tracking system in order to replicate similar results.

The initial and final values listed in above table are used to calculate \( d_k, e_k, \) and \( f_k \) \( k = \)
1, 2, 3) coefficients. Where as substituting the values of coefficients $d_k$, $e_k$, and $f_k$ ($k = 1, 2, 3$) equation for $b_k$ and $c_k$ in terms of $a$ are derived. Equations to calculate each coefficient are as follows:

\[
f_k = \theta_{ik} \quad (6.2)
\]
\[
e_k = \dot{\theta}_{ik} \quad (6.3)
\]
\[
d_k = \ddot{\theta}_{ik} \quad (6.4)
\]
\[
c_k = (4\theta_{f_k} - \dot{\theta}_{f_k}t_{f_k} - 4f_k - 2d_kt_f^2 - 3e_kt_f)/t_f^3 + a_kt_f^2 \quad (6.5)
\]
\[
b_k = \dot{\theta}_{f_k}/t_f^3 - (3/t_f^4)\theta_{f_k} + (3/t_f^4)f_k + d_k/t_f^2 + (2/t_f^3)e_k - 2a_kt_f \quad (6.6)
\]

GA generated values for $a_1$, $a_2$ and $a_3$. These values were substituted in respective coefficient $c_d$ and $d_k$ equation. This gave us three 5\textsuperscript{th} degree polynomial in time. This was then used as a reference trajectory for the controller as explained in earlier chapters.

Eqn. 6.7 shows the fitness function used for the GA. This fitness function takes into account the x-coordinate of the COM of the entire body. GA tried to minimize the cost incurred because of distance between x-coordinate of COM of the body and $x = 0$. This made COM track the trajectory which forced the COM to go in horizontal direction first in order to reduce fitness function cost and then travel in vertical direction. Which reduced the cost as COM travels close to $x = 0$.

\[
F = 2 \times 10^2 \times COM^2 + \text{penalty} \quad (6.7)
\]

Whenever trajectory generated using coefficients generated by GA showed behavior which was not realistic or could not be exhibited by human beings under normal circumstances a “penalty” was added to the fitness function cost. The constraints to decide whether to add the
penalty or not are as follows:

- **Constraint 1**: $-212 \leq \tau_1 \leq 212$
- **Constraint 2**: $-290 \leq \tau_2 \leq 290$
- **Constraint 3**: $-200 \leq \tau_3 \leq 200$
- **Constraint 4**: $0 \leq \theta_1 \leq \pi/2$
- **Constraint 5**: $\pi/2 \leq \theta_2 \leq \pi$
- **Constraint 6**: $0 \leq \theta_3 \leq \pi/2$
- **Constraint 7**: $COM_x \leq 0.05$
- **Constraint 8**: $t_f > 1$

To make sure that COM does not erratically tries to achieve certain position these constraints were imposed. Some of the constraints are based on real world hardware availability such as motors and others are based on human joint limitations.

Simulations were carried out with initial population size of 1001 candidates and over 100 generations. The results for different cases are discussed in detail in the following chapter.

### 6.3.2 Inverse Kinematics

GA is also used to calculate inverse kinematics. The goal is to use COM location and calculate ankle, knee and hip joint values. For each COM location a set of $\theta_1$, $\theta_2$ and $\theta_3$ value is generated. GA is restricted to find the joint angle values in the vicinity of the joint angle values for previous step. For instance, initial joint angles are $\theta_{1i}$, $\theta_{2i}$ and $\theta_{3i}$ that defines $COM_i$ location. Then for $COM_{i+1}$ location the GA searches values of $\theta_{1(i+1)}$, $\theta_{2(i+1)}$ and $\theta_{3(i+1)}$ in the ± 0.2 radians range of $\theta_{1i}$, $\theta_{2i}$ and $\theta_{3i}$.
The fitness function used for the GA to calculate inverse kinematics is as follows:

\[
F_1 = 10 \ast (|COM_x(\text{desired}) - COM_x(\text{candidate})| + |COM_y(\text{desired}) - COM_y(\text{candidate})|) \\
+ 20 \ast ((\theta_1(\text{previous}) - \theta_1(\text{candidate}))^2 + (\theta_2(\text{previous}) - \theta_2(\text{candidate}))^2 \\
+ (\theta_3(\text{previous}) - \theta_3(\text{candidate}))^2)
\]  

(6.8)

The fitness function explained in Eqn. 6.8 consists of two important parts. Part I calculates difference between COM location calculated using candidate solution set for \(\theta_1\), \(\theta_2\) and \(\theta_3\) and actual COM location for which inverse kinematics needs to be calculated. Part II calculates the squared difference between \(\theta_1\), \(\theta_2\) and \(\theta_3\) of previous step with candidate set \(\theta_1\), \(\theta_2\) and \(\theta_3\).
Chapter 7

Numerical Simulations And Results

7.1 Trajectory Generation and Tracking

GA was used to generate values for coefficient $a_k$ for $5^{th}$ degree polynomial in $t$. Then these joint angle trajectories are provided to the PD controller. To study flexibility of the algorithm numerical results for different cases of different initial and final joint angle trajectories, constraints is presented in this chapter.

7.1.1 CASE: I

This involves using OptiTrack data to define initial and final joint angles. Constraints used for this case are as follows:

- Constraint 1: $-212 \leq \tau_1 \leq 212$
- Constraint 2: $-290 \leq \tau_2 \leq 290$
- Constraint 3: $-200 \leq \tau_3 \leq 200$
- Constraint 4: $0 \leq \theta_1 \leq \pi/2$
- Constraint 5: $\pi/2 \leq \theta_2 \leq \pi$
- Constraint 6: $0 \leq \theta_3 \leq \pi/2$
- Constraint 7: $COM_x \leq 0.05$
• Constraint 8: $t_f > 1$

![Figure 7-1: Performance of GA for CASE I](image)

Fig. 7-1 shows comparison between convergence of maximum, minimum and average performance index for each generation. Performance index is the value of fitness function for each generation. Maximum performance index refers to the candidate with highest total fitness function value, from population of 1001 candidate solution. Likewise, minimum performance index refers to the candidates with minimum total fitness function value, from same generation candidate pool. The average performance index represents, the average of fitness function value for each candidate from one generation. The convergence of maximum, average and minimum performance index is an indication of how good Genetic Algorithm is working. It shows the searching capacity as well as ability to move towards global minima. Sudden change in maximum performance index shows the working of mutation. This mutation is very important, specially when there are multiple local minima in the system. It introduces a mutated candidate in the solution space which may get the Genetic Algorithm out of local minima.

Convergence of minimum performance index is clearly visible in Fig. 7-2. As the change in minimum performance index tends to very small compared to actual value of performance index, it is an indication of Genetic Algorithm reaching global minima.
The value attained by the performance index before algorithm stopped was 10240. Also, the plot shows all the minimum performance indexes evaluated for each function evaluation.

Fig. 7-3 shows comparison between actual human joint angle trajectories with joint angle trajectories generated using Genetic Algorithm. The actual joint angle trajectories were recorded over 3.19 seconds. Whereas, Genetic Algorithm generated joint angle trajectories for 1.004 seconds. For comparison purposes, the actual human joint angle trajectories were normalized for 1.004 seconds. Ankle, knee and hip joint angle trajectories have specific trend. This trend directly affects the COM trajectory. In order to achieve a human like COM trajectory, it is very important to get those key trends in all joint angle trajectories. For instance, actual ankle angle trajectory and ankle angle trajectory generated using Genetic Algorithm do not overlap perfectly but both of them do share similarity in terms of how they change with respect to other joint angle trajectories. Also, actual hip angle trajectory and hip angle trajectory generated using Genetic Algorithm have significant difference between the time at which the lowest value of joint angle is achieved but both of them do achieve approximately similar lowest value and the change with respect to other joint angle trajectories is also similar. Similarly, actual knee angle trajectory and knee angle trajectory generated using Genetic Algorithm decreases gradually.
Figure 7-3: Comparison of Actual Joint Angle Trajectories Vs Joint Angle Trajectories Generated Using GA for CASE I

Fig. 7-4 shows how Genetic Algorithm is able to generate COM trajectory that closely matches with actual human COM trajectory. The COM trajectory generated using Genetic Algorithm shows that the objective is successfully been achieved. The fitness function designed for the Genetic Algorithm worked well and as expected. The x-coordinate of the COM moved towards \( x = 0 \) in the early stage of the STS to achieve stability. Once it reached close to \( x = 0 \), the y-coordinate started to increase prompting completion of stabilization phase and starting of standing phase. The COM trajectory shares a close relation with joint angle trajectories discussed in Fig. 7-3. For instance, the lowest point is achieved approximately around the same time as hip angle achieves the lowest point in the trajectory. This is because of the fact that hip angle controls the movement of entire body above waist joint, which contributes most to the COM location. Also, around the same time ankle angle is also actuated which contributes to the lowering of COM location further. After the COM location reaches close to heel of the human at \( x = 0, y = 0 \), the body becomes stable. This location may change based on the speed with which the STS is being carried out. If fast STS is performed then COM will start to move in positive y direction, before it reaches \( x = 0 \). Whereas, if slow STS is performed then COM
will go pass \( x = 0 \), before moving in positive y direction. The case presented by Fig. 7-4 is a typical medium speed case of STS. Once the lowest point is reached based on the speed of STS, the COM trajectory starts to progress in positive y. This happens as a result of increase in hip and ankle angle and decrease in knee angle at the same time. It is important to note that the only input to the Genetic Algorithm is initial and final joint angles. The COM trajectory seen in Fig. 7-4 is a result of carefully designed fitness function for the Genetic Algorithm.

Figure 7-4: Comparison of Actual COM Trajectory Vs COM Trajectory Generated Using GA for CASE I

Once the joint angle trajectories are generated using Genetic Algorithm for each individual joint, they are used as reference trajectories for the controller to track. Fig. 7-5 shows how closely the system with a PD controller was able to track the reference joint angle trajectories. The dotted lines represents trajectories which system achieved during STS. The deviation from the reference trajectory is very small, as evident from the Fig. 7-5. Also, it is important to note that, the dotted trajectories are smooth in nature. This is very important when it comes to human robot interaction devices.
Fig. 7-5 compares torque applied by actual human during STS and torque values generated by system in order to track the reference trajectories. The values represented by the dotted lines are required to perform STS using exoskeleton only on one side.

Figure 7-6: Output of PD Controller for CASE I

Figure 7-6: Comparison of Actual Joint Torque Trajectories Vs Joint Torque Trajectories Generated Using GA for CASE I
In other words if the exoskeleton is used to assist both legs during STS, only half of the torque values represented in the plot are required.

It is clear from the figures that GA was able to generate trajectories that were close enough to actual joint angle trajectories. Also, controller output plot shows that controller was able to track the joint angle trajectories closely.

7.1.2 CASE : II

Constraints used for this case are as follows:

- Constraint 1: $-212 \leq \tau_1 \leq 212$
- Constraint 2: $-290 \leq \tau_2 \leq 290$
- Constraint 3: $-200 \leq \tau_3 \leq 200$
- Constraint 4: $0 \leq \theta_1 \leq \pi/2$
- Constraint 5: $\pi/2 \leq \theta_2 \leq \pi$
- Constraint 6: $0 \leq \theta_3 \leq \pi/2$
- Constraint 7: $COM_x \leq 0.1$
- Constraint 8: $t_f > 1$

Similar to CASE I, Fig. [7-7] shows comparison between convergence of maximum, minimum and average performance index for each generation. The difference between Fig. [7-1] and Fig. [7-7] is insignificant compared to actual value of the performance index, even though COM constraint is changed. In larger context, this is a good sign. It shows that Genetic Algorithm is sensitive to the changes to constraints but not does not make it unstable.
Convergence of minimum performance index is shown in Fig. 7-8. As the change in minimum performance index tends to very small compared to actual value of performance index, it is an indication of Genetic Algorithm reaching global minima. There is a significant difference between converged value for CASE I and II. Because of change in constraints, the path followed by COM changes and that causes change in minimum value achieved by fitness function.
The value attained by the performance index before algorithm stopped was 8230. Also, the plot shows all the minimum performance indexes evaluated for each function evaluation.

Fig. 7-9 shows comparison between actual human joint angle trajectories with joint angle trajectories generated using Genetic Algorithm. The difference between Fig. 7-3 and Fig. 7-9 is that, the lowest value achieved by hip and ankle joint. This affects the COM trajectory to a great extent. Apart from that there is no significant difference in shape of the trajectories. This is a very useful result. This allows us to manipulate the constraints based on each individual and then generate trajectories for that particular individual.

![Figure 7-9: Comparison of Actual Joint Angle Trajectories Vs Joint Angle Trajectories Generated Using GA for CASE II](image)

Results presented in Fig. 7-10 shows how much COM constraint affects the COM trajectory. It is clear that COM trajectory drifted more in positive x direction compared to previous case. Which is an outcome of moving COM constraint in positive x direction.
Fig. 7-10: Comparison of Actual COM Trajectory Vs COM Trajectory Generated Using GA for CASE II

Fig. 7-11 and Fig. 7-12 are very similar to CASE I. Since the change in trajectory is not very significant for the controller, with same gain values controller was able to track these new joint angle trajectories.

Figure 7-11: Output of PD Controller for CASE II
Even-though the constraints on COM were changed algorithm was still able to generate set of coefficient that still exhibited a pattern close to pattern shown by a healthy human being performing STS.

7.2 Inverse Kinematics

GA is again used to find joint angle trajectories from COM position. In this the goal is to minimize the sum of squared difference between consecutive joint angles and squared distance between two consecutive COM position.

7.2.1 CASE I

The search space is enforced with same joint angle constraints that a human joint experiences. Which means, for each iteration of Genetic Algorithm the constraints on parameters were 0 to $\pi/2$, $\pi/2$ to $\pi$ and 0 to $\pi/2$. To find right value for $\theta_1$, $\theta_2$ and $\theta_3$, Genetic Algorithm searched for the entire space defined by the constraints. This required lot of computational
time. Also, since the space was so large, narrowing down on exact value became more difficult. The Fig. 7-13 shows joint angle trajectory generated by performing inverse kinematics using Genetic Algorithm and COM trajectory.

As discussed earlier, it is very hazardous to use irregular trajectories in human robot interaction devices. The joint angle trajectories shown in Fig. 7-13 are not smooth and formed the basis for refined constraints on parameters. The results were encouraging and showed the applicability of Genetic Algorithm for inverse kinematics problem.

Fig. 7-14 shows how closely the COM trajectory obtained using joint angle trajectories obtained using inverse kinematics follow the desired COM trajectory. The COM trajectory was overlapping desired COM trajectory. There are some sudden changes in the COM trajectories as a result of irregularities in joint angle trajectories.
Figure 7-14: Comparison of Actual COM Trajectory vs COM Trajectory Obtained from Joint Angles Obtained by Inverse Kinematics for CASE I

Fig. 7-15 shows the error in each joint angle trajectory compared to target joint angle trajectories. For the most part error stays within ± 0.1 radians for all three joint angles.

Figure 7-15: Difference Between Actual Joint Angle Trajectories and Joint Angle Trajectories Obtained by Inverse Kinematics for CASE I
7.2.2 CASE II

Lessons learned from CASE I were incorporated and the search space was enforced for each iteration using joint angles calculated during the previous step with ± 0.2 radians. These constraints were developed based on the fact that consecutive COM locations are very close and in order to reach new position the change in each joint angle should not be more than a certain value. Fig. 7-16 shows the outcome of the Genetic Algorithm with refined constraints. There is significant improvement in the smoothness of the trajectory compared to CASE I. The main reason is the confined search space. New constraints made sure that there is very less opportunity for the algorithm to search in spaces where there is small chance of presence of right solution.

Figure 7-16: Comparison of Actual Joint Angle Trajectories Vs Joint Angle Trajectories Obtained by Inverse Kinematics for CASE II

Fig. 7-17 shows how closely the COM trajectory obtained using joint angle trajectories obtained using inverse kinematics follow the desired COM trajectory. The COM trajectory was overlapping desired COM trajectory. Also, sudden changes in the COM trajectories as a result of irregularities in joint angle trajectories were significantly improved compared to CASE I.
Figure 7-17: Comparison of Actual COM Trajectory vs COM Trajectory Obtained from Joint Angles Obtained by Inverse Kinematics for CASE II

Figure 7-18: Difference Between Actual Joint Angle Trajectories and Joint Angle Trajectories Obtained by Inverse Kinematics for CASE II

Fig. 7-18 shows the error in each joint angle trajectory compared to target joint angle
trajectories. For the most part error stays within ± 0.07 radians for all three joint angles.

    Significant improvements in terms of smoothness of joint angle trajectories and maximum
value of error for each joint angle trajectories were observed after refining constraints.
Chapter 8

Discussion, Conclusion and Future work

Natural STS is a very complex process in which ankle, knee and hip joints coordinate with each other to complete the task. Human not always chooses the trajectory requiring minimum joint torque but chooses the one that achieves the stability faster. This is shown by human tendency to lean the entire body above waist joint forward and then actuate knee and ankle joint.

This also shows that individual joint angles are of not that importance but COM of the body is an indication for STS process. The trajectory of COM followed during STS tells us a lot about STS. For instance based on start of upward shift in COM can tell us about speed with which STS is being performed. Faster STS COM trajectories tend to start upward shift earlier than slower STS. Changes in joint angle trajectories may have multiple reason and hence COM trajectory is needed to make conclusion regarding changes in trajectories.

GA provides a fast and acceptable solution. The constraints on the fitness function for GA plays an important role in guiding the COM trajectory along a right path. Also the search space constraints also make it faster to reach global minima. These constraints are based on available hardware at this point and can definitely be updated in future to as the available hardware improves. This will push us one more step forward in replicating what human tries to do during STS.

Inverse kinematics is a well studied problem. Due to possibility of more than one solution there are very few mathematical methods that can do the job. Another way to approach the problem is to precalculate all the possible joint angle values to 4 digit precision and then find the set of joint angles that belongs to particular COM location. This process takes more time.
and can not be implemented for real time systems. GA algorithm does the exact same thing of exploring space but without calculating extra joint angle values that are not required which makes GA faster. There are other search algorithms like GA that can be explored to compare the time each algorithm takes.

Human body and the exoskeleton system are approximated as an inverse 3 DOF system. The dynamics of the system is highly non linear. Feedback linearization does a good job of linearizing the system. Once the system is linearized the problem becomes easy and a PD controller does perform well with linear systems. The gains of PD controller largely affects the torque input to the system. In human robot interaction, a system should have sufficient damping for human to cope with the system. A sudden jerk caused by high gains of the system may pose a threat to the human safety.

Results generated in this research allows us to explore the control COM trajectory more. This shows us the possibility to manipulate the fitness function and there by generating different COM trajectories for same human being. These trajectories can be faster or slower. Also, inverse kinematics results allows us to use COM trajectory as a reference trajectory and control the lower limb exoskeleton using COM.

8.1 Future Work

The fitness function for Genetic Algorithm needs to be refined, to allow generation of joint angle trajectories for different speeds. Amending the constraints on the fitness function used to calculate joint angle trajectories is necessary for future advancement. Gain scheduling or fuzzy logic can be used, to manipulate the gains of the PD controller in order to minimize the torque values. Robustness analysis for the controller needs to be performed. The inverse kinematics results are very primitive and need more work to reduce the error. Refining the constraints on the search space and exploring other possible avenues of search algorithms may give us better results.
References


