Linear Quadratic Regulator Control of an Under Actuated Five-Degree-of-Freedom Planar Biped Walking Robot

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Matthew Thomas Leines

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

Adviser, Jiann-Shiou Yang

September 2010

(The essential components of the copyright notice are the copyright symbol © or the word “copyright,” the full legal name of the author, and the year of publication.)

Acknowledgements

I'd like to thank my parents for housing me again for the duration of this degree program, my brother for putting up with my distraction needs by proxy, my baby nephew for somehow understanding that I could only get away from the computer to play for 5 minutes at a time, and to all my friends and family who listened to my exclamations of how close to done I was and only offered a smile when I somehow flew by another self-imposed deadline.

I also want to thank Dr. Jiann-Shiou Yang and Dr. Debao Zhou at the University of Minnesota, Duluth, for all their help throughout this effort. Their understanding and patience since I entered active duty made it possible for me to finish this more than any other factor.

I want to give special thanks to Dr. Yang for his support throughout this effort and especially when he picked up this topic when my previous advisors left the campus, without it, none of this would have been possible.

Dedication

This thesis is dedicated to Jess and Gavin. One for waiting for me, the other for accepting that although Iron Man's power suit would not be forthcoming, Iron Man's pants were just as good.

Abstract

Modern robotic systems are fully actuated with full information of themselves and their immediate surroundings. If faced with a failure or damage, robotic systems cannot function properly and can quickly damage themselves. A Linear Quadratic Regulator (LQR) control system is proposed to allow an under actuated (damaged) robotic system (five-degrees-of-freedom, planar, biped, walking robot) to continue to follow a human-like walking gait in a series of Matlab and Simulink simulations. The proposed LQR controller keeps joint position errors to below 4 degrees for the fully actuated system, performing the entire gait within the given step time and length. The under actuated control system can match the fully actuated system using a separate LQR controller. Use of time-varying control and Markovian jump methods can compile both controllers into a dynamically adaptive whole, capable of full to partial gait during both locked joint and free joint failures with brakes applied as needed.

_____

Signature of Faculty Advisor

Table of Contents

List of Tables

List of Figures

1. Introduction

The word “robot” was first coined in the 1921 play R.U.R. (Rossum’s Universal Robots) by Karel Capek, where in its original Czech the word “robota” refers to the kind of labor serfs would perform on their master’s lands [1]. The first “modern” robot, digitally operable and programmable, called Unimate, was installed at a General Motors plant in 1961 and was used to lift and stack hot metal from a die casting machine [2]. Since Unimate industrial robotics haven’t changed much in physical structure, in general incorporating multiple electromechanical motors and an end effector tool to complete tasks. What has changed significantly in the last 50 years are the programming complexity, performed tasks and the human imagination behind these machines.

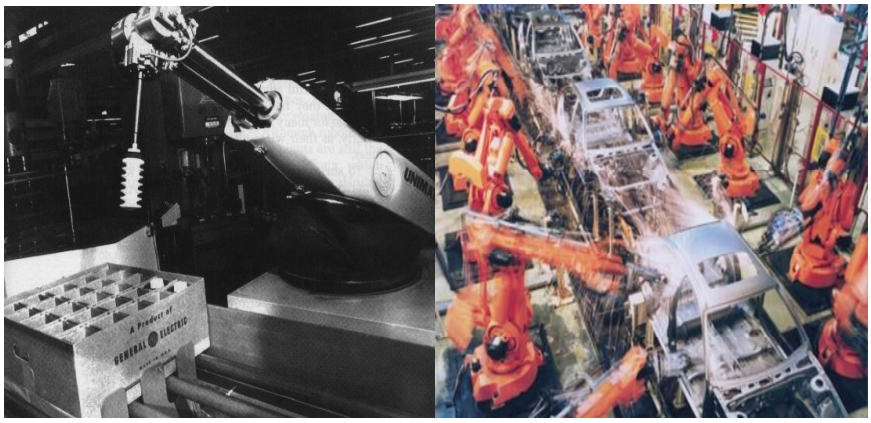


Figure 1. (Left) The Unimate robot [2]. (Right) Modern car assembly robots with welding end effectors [3].

1.1 Background

Most modern robotic machines are used in service and industrial capacities. Examples of service robots are becoming more common as products like the Roomba can

autonomously vacuum floors and a Robomow can trim a lawn [4, 5]. Industrial robots are commonly in the form of an articulated arm. Most industrial robots have 3 or more degrees-of-freedom (DOF) from electromechanical, hydraulic or pneumatic actuators giving them a great deal of freedom-of-motion, speed and strength as needed. The additions of end effectors (a none to multiple DOF, job specific device or tool) allow industrial robots to perform specific tasks very efficiently and are commonly found performing automotive assembly, electronic manufacture and packaging. Research robot systems come in a wider variety of forms used to push the technological limits, develop new features and control schemes and determine new skills that could be automated.

Robotic systems are popular for a number of reasons; strength, speed, precision, reliability and durability to name a few. With better microcontrollers, small but powerful electric motors and modern semiconductor technology, robotic systems are becoming capable of more and more tasks. Industrial robots can have more complicated end effectors and more degrees of freedom, making them much more flexible and able to handle multiple types of jobs [6]. Systems have been developed to do things that are deemed too dangerous for humans like traversing the rubble of a collapsed building to locate survivors, or assisting an operator to disarm a bomb [7, 8]. The increasing complexity of robotic systems and their programming is beginning to open up the field to things that couldn't be considered before.

Human imagination gives insight to where people envision the future of robotics. Imagery is very widespread in cinema of human functioning machines that are here to either help (C3PO in Starwars, 1977) or hurt us (The Terminator, 1984). Literature goes back even farther with Robert A. Heinlein's classic 1959 novel "Starship Troopers,"

describing soldiers fighting on far off worlds in powered armor space suits that greatly enhance strength and lift capacities or Isaac Asimov's 1950 novel "I, Robot" and the laws and ethics of humanoid machines interacting in society. It seems that science fiction at least foresees machines that are as capable, or even more so, of anything a human could do.

Developments in the robotics field have taken some baby steps toward the kind of autonomy required to make fiction a reality. Systems like the DaVinci can now be used to perform surgery over a distance with a real doctor using teleoperation [9]. Honda's human-like ASIMO can walk up stairs, dance and even run [10]. Cyberdyne's Hybrid Assistive Limb (HAL) allows a paraplegic person to wear a powered suit and by detecting weak muscle signals, allow that person to walk [11]. DARPA's on its third working prototype to replace an amputee's arm with a fully articulate robotic limb, with 22 degrees-of-freedom (DOF), and control it by wiring directly into the nerve endings [12]. Boston Dynamics' BigDog (another DARPA funded project) is an autonomous 4 legged platform capable of carrying several hundred pounds of payload, walk over very rough terrain and recover from severe disturbances like slipping on ice or getting kicked over [13]. Instead of enhancing or replacing human actions, other projects like Ishikawa Komuro Laboratory's robot hand and vision system shows off completely inhuman dexterity and speed [14].

These are all up and coming technologies in the field of robotics, but to reach the lofty goals of science fiction, there are still many difficult challenges to overcome. This thesis aims to add to the effort of solving just one of these challenges.

1.2 Problem Statement

Modern robots are fully actuated systems with full information of themselves and their immediate surroundings. This knowledge is what allows determination of the robots' physical and motion parameters, the dynamics, to a level of precision allowing prediction and control of the machine. Without this complete knowledge the system dynamics change and robotic systems cannot function properly, sometimes causing damage to themselves and their surroundings without intervention. Examples of this kind of disruption include failed actuators, unaccounted for objects, a person within the work space, or structural damage, each of which causes a significant change that the system was not designed to operate in.

Loss of use of a joint or a robot with less control than actuation points can be referred to under actuated. A control system that can adapt from fully actuated to under actuated would have impacts from advanced prosthetics to automation and industrial robotics. For example, imagine an aircraft that could still retain complete control after the loss of a rudder, without burdening the pilot's skills. An under actuated control system could allow damaged or altered systems to retain some or all of their intended functions or at least continue partial function while retaining enough control to prevent further damage.

1.3 Approach

To understand this problem, a set of simulations are conducted. Using human-like size and mass parameters, bipedal robot dynamics are modeled. With Matlab

software, the model is simulated and given a modeled walking gait to perform. By using a linear quadratic regulator (LQR) controller to ensure the proper walking gait is performed, this simulation creates a fully actuated system. Once the fully actuated system is established, it is made under actuated by removing control over a specific joint actuator. This is done to represent damage to the robotic system in the form of a failed or unresponsive actuator motor. The now destabilized system can't perform the walking gait with the original LQR controller, so a new LQR controller is developed and simulated with the under actuated model. Using under actuated control, the walking gait is performed without the use of the uncontrolled joint and overall effectiveness is compared with the fully actuated system performance.

Once control systems exist that can perform the walking gait for both the fully and under actuated systems, a method to combine or switch between the two controllers is needed. This can be done using time-varying control and Markovian jump methods, combining various controllers into operation states as a single dynamically adaptive controller.

1.4 Overview

The following chapters of this thesis outline the modeling of the bipedal system, development of the controller and simulation results of the fully and under actuated systems. Chapter 2 discusses the modeling of the biped walking simulation. The first section describes the creation and implementation of the proposed walking gait. The next section describes the proposed robot model in detail, including the full dynamics model.

This chapter concludes with a simulation of the walking gait using the robot model through the use of Matlab software.

Chapter 3 covers the background and steps used to utilize the system dynamics in creating an optimized controller to implement the walking gait. It starts by describing the conversion of the system dynamics into a state space representation that can be utilized to create controllers. The next section introduces LQR control in detail. This chapter concludes with details on how the proposed LQR controller was optimized for use with the Matlab simulations of the fully and under actuated systems.

Chapter 4 describes the Matlab simulations results. Data and analysis of results from both the fully actuated controller simulation and the under actuated controller simulation are discussed in detail. The last chapter concludes with a summary of the results and recommendations for future work.

2. Modeling the Biped

To study under actuated control, first a model of the walking gait and a physical system to move with it is needed. This chapter starts with the basics of creating a biped walking gait and the one modeled in the experiments. Following walking is the robot model as used in the experiments and an explanation of the dynamics of the planar 5 degree-of-freedom (5DOF) system. The chapter concludes with simulations of the walking gait and robot model as used in Matlab, Simulink and SimMechanics software.

2.1 Introduction to Biped Walking

2.1.1 Defining the Simulation Environment

Bipedal motion can be described as representative of human locomotion. Human walking locomotion looks to the outside observer as an effortless and almost smooth transition of weight from one foot to the other. If one thinks about the motions needed to move as they walk, it quickly becomes apparent that quite a lot is going on to produce that motion. The brain coordinates the flexing of the spine, tilt of the hips, lifting of the leg, support and firmness of the ankle, sense of balance and all the muscles and tendons needed to perform and control such a movement. Corrections for ground surface conditions like gravel or ice, outside influences like a stiff wind or running water, and more radical conditions like landing a jump, or tripping and missing a step are all very quickly adjusted for by the brain with input from the senses. While all of these things are

possible to model and simulate to some degree, a very realistic and situation flexible model of human walking, let alone any motion, would be a considerable challenge.

To simulate only walking motion in a robotic system requires that several constraints and assumptions be made. To reduce the complexity of the final control systems, a "simple as possible" biped walking gait is modeled and simulated in this thesis. The gait will be performed on a perfectly level and firm surface (no compression, absorption, angled or elevation related surface forces) with a perfect friction coefficient (no slipping, a foot placed on the ground stops exactly where placed) and the only outside influence is the force of gravity (acting down towards and perpendicular to the surface) on the model. With this simulation environment defined, we have simplified the controllable conditions for full three dimensional walking motions.

2.1.2 Reducing Model Complexity

Unfortunately, a human in motion represents a system with upward of a few hundred DOF, depending on how it's modeled. Since each DOF mathematically represents the size of the required matrices in the dynamics and controller calculations, they must be kept at a minimum and further simplification is needed. To reduce the number of DOF required to model walking motion, the simulation is reduced to two dimensions in the sagittal plane. As opposed to the transverse plane (which bisects a body horizontally parallel to the ground plane, creating top and bottom) and the coronal plane (which bisects a body vertically, passing through both shoulders, creating a front and a back), the sagittal plane vertically bisects the center of the model body from head to

foot, creating left and right halves. Modeling in the sagittal plane creates a side view where forward movement is shown as progressing either to the left or right of the page, shown in Fig. 2.

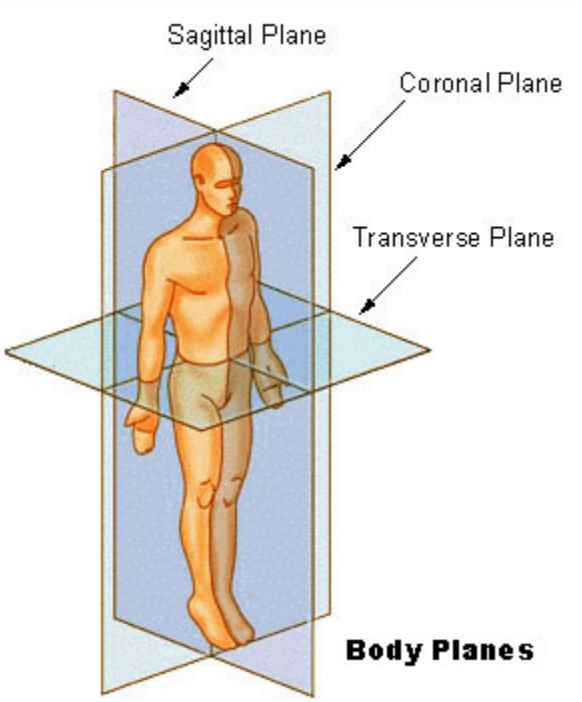


Figure 2. Illustration of the sagittal, coronal and transverse planes [16].

Even reduced to planar motion, this model could require dozens of DOF, (just consider how many joints make up the flexing motion of a human spine or how arm swing affects balance) and must be further reduced. Considering what is needed at a minimum for walking motion, the contributing motions of the arms, head, neck, chest and back are ignored. The ankles and feet have a great deal of impact on three dimensional motions and balance, but in the sagittal plane and on this perfect simulated surface the contribution is considered negligible, so they are mostly ignored. What are left to consider in this physical model are the hip, knee and “ankle” joints. The foot as modeled

is represented as a simple post and the joint is therefore an interaction between the ground plane and the end of this post, similar to ankle movement. This simplification of the foot and ankle creates a single DOF as long as the foot post is in contact with the ground plane and it is assumed that force can be exerted against the ground as if an ankle were present. Finally, all modeled joints have had their normal range of motion in the sagittal plane reduced to simple rotational joints, which have only one DOF each. This leaves the proposed model with a total of 6 DOF, as represented in Fig. 3, with both feet on the ground, and when one foot is off the ground, the model is further reduced to 5DOF.

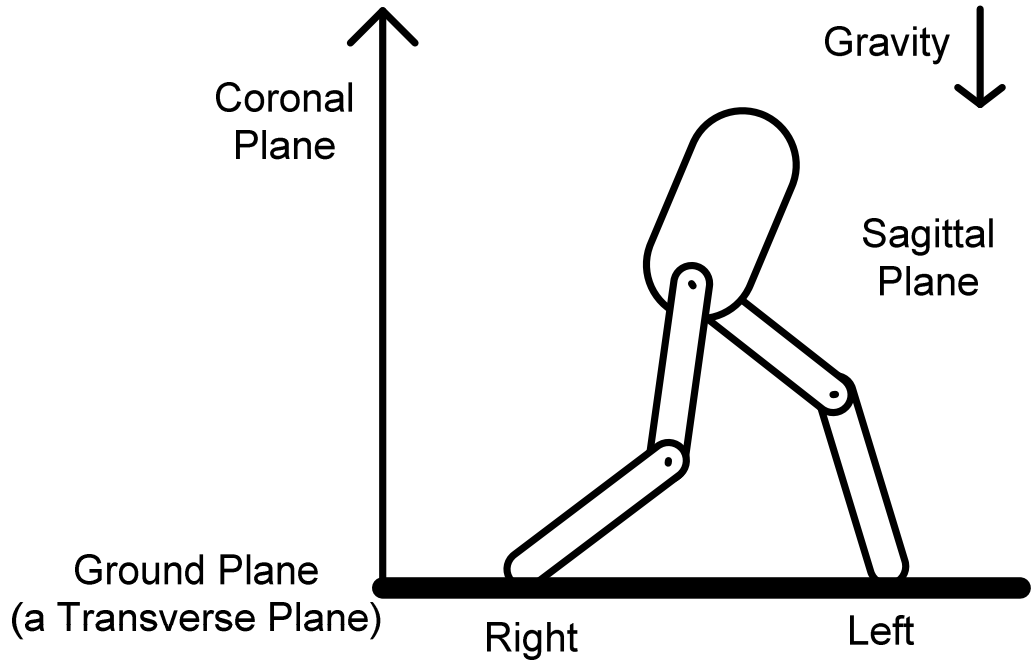


Figure 3. Proposed biped model shown in sagittal plane on simulation surface reduced to a maximum of 6 DOF (both hips, both knees, and both feet posts touching the ground).

2.1.3 Defining a Walking Gait

A walking gait can be divided into two distinct phases of motion; the single support phase (SSP) and the double support phase (DSP). In biped walking motion, the SSP begins when the model's weight is shifted to a single foot, and continues as the other foot is raised and moved to its new position, this represents a single step. Once the raised foot is placed on the ground plane again, the SSP ends. The DSP starts at this point when weight is again redistributed between both feet, stabilizing the biped's balance and ends as the opposite foot is lifted into its SSP. A continuous flow from SSP to DSP to the other foot SSP to DSP again represents 2 full steps and 1 full gait cycle [16-19]. It is important to note that if the stabilizing effect of the DSP places the feet and legs into the same but mirrored positions from left to right SSP, then the gait is periodic in nature, making it possible to model the entire walking gait by defining only a single SSP and DSP [18, 20]. This process is illustrated in Fig. 4.

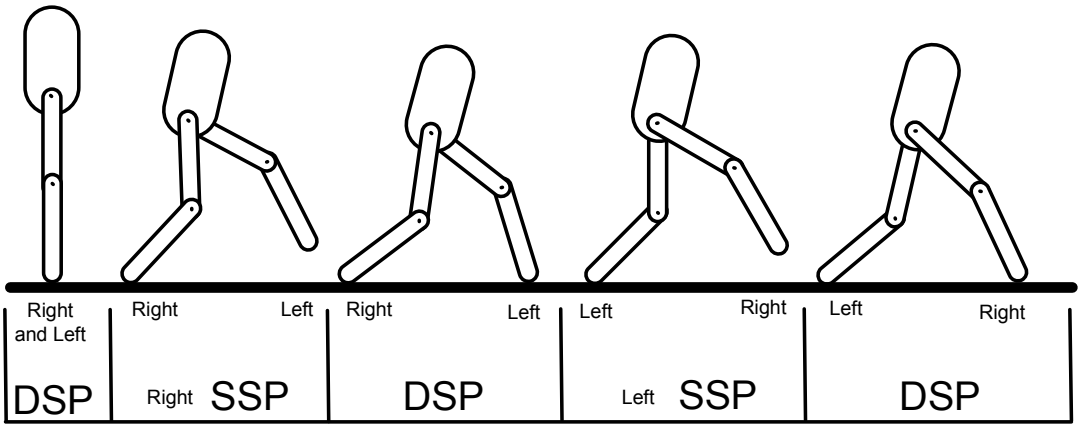


Figure 4. Illustration of walking gait passing from SSP to DSP for one full gait cycle.

Creating a walking gait with these properties is a field of study in itself, with many publications available. For example, in [18] a method for creating an arbitrary

walking gait for a planar 5 DOF robot is defined using over 20 variables but gives choices on step length, hip height, torso tilt, foot lift, etc… and handles both SSP and DSP gait phases. Attempts to model the SSP as shown in [16-25], while highly detailed and life-like, were eventually found to be unnecessarily complicated for the needs of this simulation (which is to test the proposed controller). A related but simpler method was used.

2.1.4 Modeling the SSP

The SSP gait used in this simulation was eventually developed with a protractor, with the gait angles mapped out on paper to be visually periodic. The requirements to keep the gait periodic are to ensure that final SSP angles of one leg are the visual mirror of the starting SSP angles of the other, and vice versa. Then over the duration of the SSP the joints are steadily shifted from initial to final angles. This generates the trajectories of the desired motion, that is, the angles and velocities of the various joints at any time during the period of motion. This method of trajectory planning is called finding a Path Polynomial Solution [26]. To implement this, the initial and final angle positions, joints, rotation direction, and walking step parameters must all be defined.

First, angle variables are assigned to the joints shown in the Fig. 3 model; a direction for initial motion and a positive direction orientation are chosen to allow easy understanding of the trajectory information. This is shown in Fig. 5.

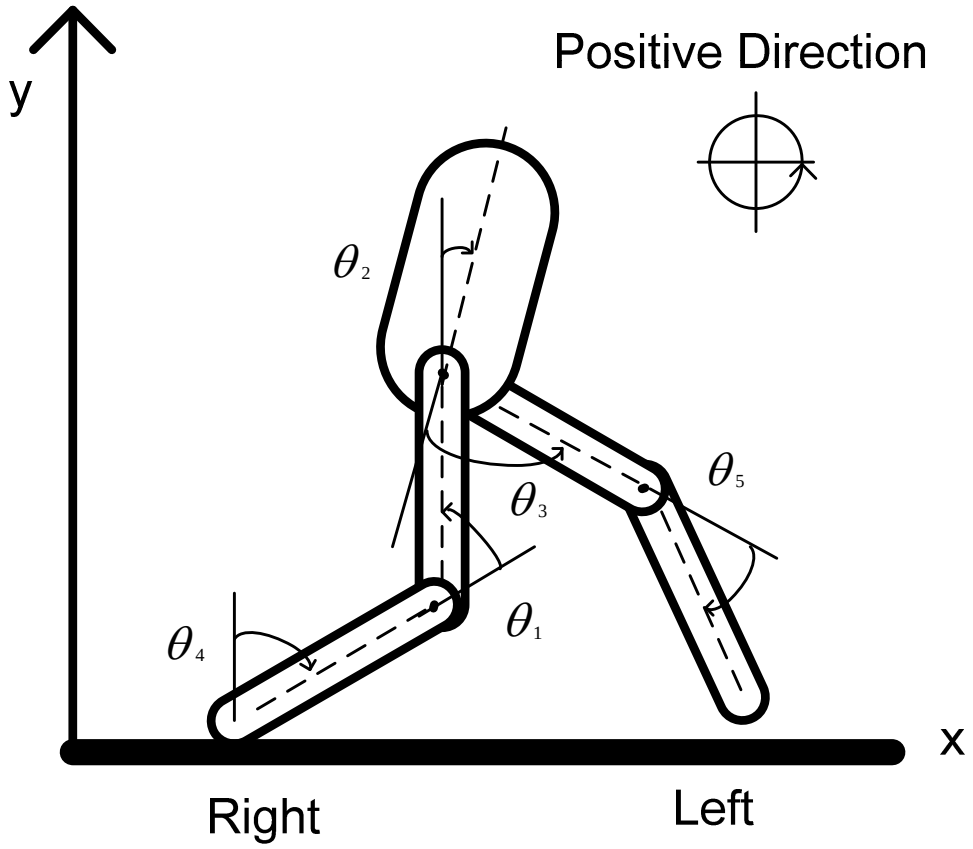


Figure 5. An exaggerated position showing location of joints, direction of motion during the right leg SSP.

As in Fig. 5, the joint trajectories for the SSP are defined in (1).

$$\begin{aligned}
\theta_1 &= \text{angle of right knee} & \dot{\theta}_1 &= \text{velocity of right knee} \\
\theta_2 &= \text{angle of right hip} & \dot{\theta}_2 &= \text{velocity of right hip} \\
\theta_3 &= \text{angle of left hip} & \dot{\theta}_3 &= \text{velocity of left hip} \\
\theta_4 &= \text{angle of right foot} & \dot{\theta}_4 &= \text{velocity of right foot} \\
\theta_5 &= \text{angle of left knee} & \dot{\theta}_5 &= \text{velocity of left knee}
\end{aligned} \tag{1}$$

To plot out the motion of the model, the terms in (1) must be solved for any point in time. For a periodic gait, the starting and ending trajectories are known. To define the desired motion trajectory, the known initial conditions are

$$\begin{aligned} t_0 &= \text{SSP start time} \\ t_f &= \text{SSP end time} \\ q_0 &= \text{initial joint angle} \\ \dot{q}_0 &= \text{initial joint velocity} \\ q_f &= \text{final joint angle} \\ \dot{q}_f &= \text{final joint velocity} \end{aligned} \tag{2}$$

By using standard position and velocity polynomials defined in [18], the q terms in (2) can be written as

$$\begin{aligned} q_0 &= a_0 + a_1 t_0 + a_2 t_0^2 + a_3 t_0^3 \\ \dot{q}_0 &= a_1 + 2a_2 t_0 + 3a_3 t_0^2 \\ q_f &= a_0 + a_1 t_f + a_2 t_f^2 + a_3 t_f^3 \\ \dot{q}_f &= a_1 + 2a_2 t_f + 3a_3 t_f^2 \end{aligned} \tag{3}$$

Where a_0, a_1, a_2, and a_3 are unknown constants. Assuming the value of t_0 in (2) is 0 and substituting to solve the unknowns in (3) gives

$$\begin{aligned} a_0 &= q_0 \\ a_1 &= \dot{q}_0 \\ a_2 &= \frac{3(q_f - q_0) - t_f(2\dot{q}_0 + \dot{q}_f)}{t_f^2} \\ a_3 &= \frac{2(q_0 - q_f) - t_f(\dot{q}_f + \dot{q}_0)}{t_f^3} \end{aligned} \tag{4}$$

Using the terms in (2) for each joint with (4) allows the calculation of the joint trajectories for any point in time from t_0 to t_f, solving for the terms in (1).

With this information, a path between any two joint positions for a given length of time during the SSP can be plotted as long as the desired model orientations are known. Unlike more advanced methods, using this method generates only very basic point to point motions for each joint individually. That means the models' separate joint movements must be coordinated and optimized by the designer through observation and trial and error to ensure a life-like gait. The gait requirements to test the proposed controller are sufficiently simple to alleviate concern in this simulation. If more rigorous testing, advanced structure or gait study were required, a more advanced method is highly recommended.

2.1.5 Modeling the DSP

Over time, the SSP is considerably longer in duration than the DSP [19] making up around 90% of the gait cycle. Considering that the DSP is a small portion of the walking gait, a distinct and separate phase, and a very different control problem [19, 28-30] than the SSP (related to an inverted pendulum [27]), only the SSP is modeled in this simulation. The DSP is only assumed to result in the balancing and repositioning of the feet as needed to correct placement error and maintain the periodic nature of the gait.

2.2 The Biped Robot Model

2.2.1 Finding Human Biped Parameters

Now that the basics of walking are understood, a physical model of the biped must be developed to perform it. Because of the reductions in complexity required to make simulation possible, it was decided early to design the biped around human physical parameters, in hopes of returning some of the lost realism. These parameters represent average male human physical characteristics as determined in [31], but must be appropriately adapted to fit the proposed planar model. These selected values are listed in Table 1.

Table 1. Average male body length, mass and center of mass values as selected from [31].

Segment	Length (mm)	Mass (%)	Center of Mass position (%)
Head	242.9	6.94	50.02
Trunk	603.3	43.46	51.38
Upper Arm	281.7	2.71	57.72
Forearm	268.9	1.62	45.74
Hand	187.9	0.61	36.24
Thigh	422.2	14.16	40.95
Shank	440.3	4.33	43.95
Foot	258.1	1.37	44.15

In Table 1, the segment refers to the portion of the human body represented. The length refers to the longitudinal length as measured vertically if a person was standing strait up with legs together, feet flat on the ground and with arms flat against the sides of the body. The total body mass must be chosen and the various segment masses are represented as a percent of the total body mass. The center of mass (CM) position is represented as the location defined by the percent of length as measured from the top of the segment towards the bottom. Left and right arm and leg segments are assumed to be identical.

To modify these values to conform to the reduced complexity model in Fig. 3, the arms, head and trunk are combined in mass and referred to as the torso. Furthermore, the torso's length is taken as the combination of the head and trunk length. The torso's CM is placed the same as the trunk, as this is the very close to the natural CM of the entire human body and makes a good representation for the combined torso masses. The new shank contains the mass of the Table 1 shank and foot, but only the length and CM of the original shank. This was done to account for the mass of the foot and ankle but not contribute complexities from considering the length (height) changes while rolling off of a foot while walking and to keep with the "foot as a post" analogy explained for Fig. 3. Finally, to move toward dynamics and kinematics terminology, the individual segments will be referred to as links. These changes are summarized in Table 2.

Table 2. New physical parameters as modified from Table 1 to fit Fig. 3 biped model.

Links	Length (mm)	Mass (%)	CM Position (%)
Torso	846.2	60.26	51.38
Thigh	422.2	14.16	40.95
Shank	440.3	5.7	43.95

2.2.2 Assigning Physical Attributes for Motion

To begin to understand how the biped will move, the physical attributes developed in Table 2 must be applied to the Fig. 5 model. To define a position only the length of the individual links are needed, but to define motion also requires the link masses and the calculated inertias. All are needed to perform the walking gait trajectories as defined in (1).

The length of each link is provided by Table 2 with an overall height of 1708 mm and with suggestions from [31] a total biped mass of 77 kg is chosen. This gives the biped the rough physical equivalent of a 5 ft. 7 in, 170 lb. male human. Applying the needed parameters to the Fig. 5 model is shown in Fig. 6.

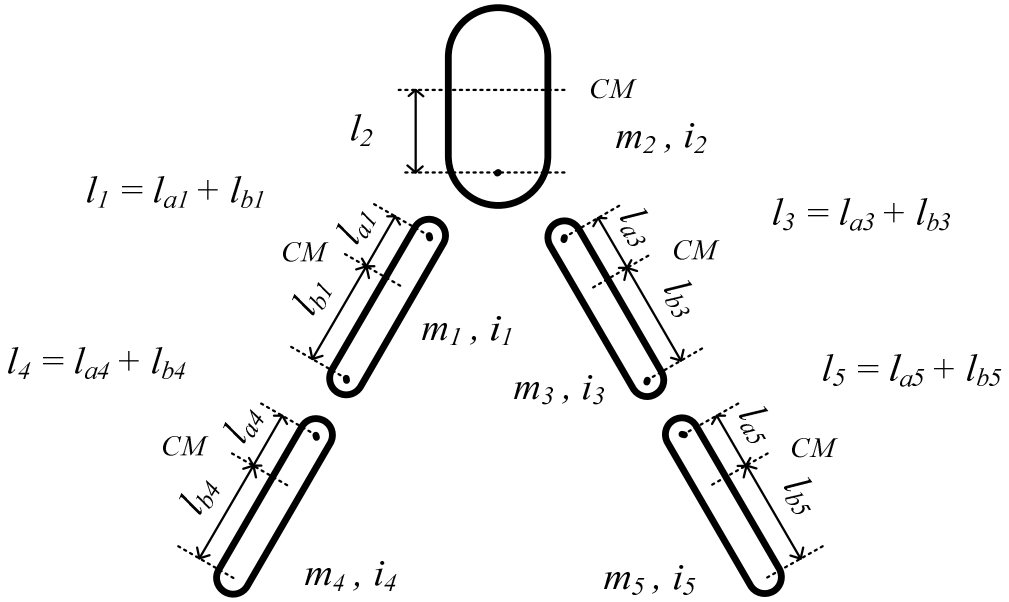


Figure 6. Biped robot general proportions

The Fig. 6 variables are defined in (5) including gravity.

$m_1 =$ mass of the right thigh
$m_2 =$ mass of the torso
$m_3 =$ mass of the left thigh
$m_4 =$ mass of the right shank
$m_5 =$ mass of the left shank

$i_1 =$ moment of inertia of the right thigh
$i_2 =$ moment of inertia of the torso
$i_3 =$ moment of inertia of the left thigh
$i_4 =$ moment of inertia of the right shank
$i_5 =$ moment of inertia of the left shank

(5)

$l_1 =$ link length of the right thigh
$l_2 =$ link length of the torso
$l_3 =$ link length of the left thigh
$l_4 =$ link length of the right shank
$l_5 =$ link length of the left shank

$CM =$ position of link center of mass
$l_a =$ length from top of link to CM
$l_b =$ length from CM to bottom of link
$g =$ force of gravity, $9.81 \frac{\text{meters}}{\text{second}^2}$

The length values are given and mass values derived from the Table 2 parameters using the selected total mass, but the inertia values require further calculation. For ease of calculation, each link is assumed to be equivalent to a thin rod of uniform mass giving the moment of inertia calculation the form of (6).

$$i = \frac{ml^2}{12} \tag{6}$$

This would put the equivalent CM in the center of the link. Since the actual link CM isn't in the physical center of the link [31], the parallel axis theorem is used to add a term to (6), accounting for the CM being off center and changing (6) into (7) for each leg link.

$$i = \frac{ml^2}{12} + m\left(\frac{l}{2} - l_a\right)^2 \tag{7}$$

The torso uses (7) with a l_a equal to 0, as its length l_2 is described as shown in Fig. 6. Table 3 summarizes the results of these calculations and displays the physical characteristics of the biped as used in the simulation.

Table 3. Length, Mass and Moment of Inertia of each Link as used in the simulated Biped Model.

Link	***l* (m)**	***l_a* (m)**	***l_b* (m)**	***m* (kg)**	***i* (kgm^2)**
Right Thigh (1)	0.4222 (l_1)	0.1729 (l_{1a})	0.2493 (l_{1b})	10.9 (m_1)	0.1778 (i_1)
Torso (2)	0.8462 (l_2)	0.4348 (l_{2a})	0.4114 (l_{2b})	46.4 (m_2)	11.075 (i_2)
Left Thigh (3)	0.4222 (l_3)	0.1729 (l_{3a})	0.2493 (l_{3b})	10.9 (m_3)	0.1778 (i_3)
Right Shank (4)	0.4403 (l_4)	0.2035 (l_{4a})	0.2468 (l_{4b})	4.38 (m_4)	0.0739 (i_4)
Left Shank (5)	0.4403 (l_5)	0.1935 (l_{5a})	0.2468 (l_{5b})	4.38 (m_5)	0.0739 (i_5)

Finally, with the physical proportions and parameters of the biped model defined, the biped dynamics can be modeled.

2.3 The Robot Dynamics Model

A standard way to derive the dynamic equations for mechanical systems is by using the Euler-Lagrange (E-L) approach [27, 29, 32], in which the torque required to move a robot with n rigid links is found by determining the various forces acting on system at a given time. This approach is preferred because it is relatively strait forward to convert into a state space representation, useful in developing a controller, which is discussed in the next chapter. As software was used to automate the development and utilization of these methods, the derived equations in this chapter are not directly used to model the biped but are representative of what is going on behind the scenes.

2.3.1 Right SSP Dynamics

To find the right leg SSP torques needed at each joint to perform the desired motions described in (1), the E-L approach is used to describe the dynamics of the system. The general form equation is (8).

$$\tau = D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + f(\dot{q}) \tag{8}$$

Where,

$D(q)\ddot{q}$ = n x n matrix for the linking structure inertial forces

$C(q,\dot{q})\dot{q}$ = n x n matrix for the centripetal and coriolis forces

$g(q)$ = n x 1 vector for the gravitational forces

$f(\dot{q})$ = n x 1 vector for the joint frictional forces

q = n x 1 vector for the joint angles at a given time

$\dot{q}$ = n x 1 vector for the joint velocities at a given time

$\ddot{q}$ = n x 1 vector for the joint accelerations at a given time

τ = n x 1 vector for the resulting applied joint torques

In this simulation, n is equal to the DOF of the biped system (n = 5). The descriptions of the component matrices and vectors in (8) are lengthy, so three layers of substitutions are used to keep the equations displayed within a single page and to reduce calculation and transcription errors. The following E-L matrices are the first layer of substitutions.

For the proposed 5 DOF planar biped, the inertial forces are described by (9).

$$D(q)=\begin{bmatrix} \alpha_{R11} & \alpha_{R12}cos(\theta_1-\theta_2) & \alpha_{R13}cos(\theta_1-\theta_3) & \alpha_{R14}cos(\theta_1-\theta_4) & \alpha_{R15}cos(\theta_1-\theta_5) \\ \alpha_{R21}cos(\theta_2-\theta_1) & \alpha_{R22} & 0 & \alpha_{R24}cos(\theta_2-\theta_4) & 0 \\ \alpha_{R31}cos(\theta_3-\theta_1) & 0 & \alpha_{R33} & \alpha_{R34}cos(\theta_3-\theta_4) & \alpha_{R35}cos(\theta_3-\theta_5) \\ \alpha_{R41}cos(\theta_4-\theta_1) & \alpha_{R42}cos(\theta_4-\theta_2) & \alpha_{R43}cos(\theta_4-\theta_3) & \alpha_{R44} & \alpha_{R45}cos(\theta_4-\theta_5) \\ \alpha_{R51}cos(\theta_5-\theta_1) & 0 & \alpha_{R53}cos(\theta_5-\theta_3) & \alpha_{R54}cos(\theta_5-\theta_4) & \alpha_{R55} \end{bmatrix} \quad (9)$$

The centripetal and coriolis forces are described by (10).

$$C(q,\dot{q})=\begin{bmatrix} 0 & \dot{\theta}_2\alpha_{R12}\sin(\theta_1-\theta_2) & \dot{\theta}_3\alpha_{R13}\sin(\theta_1-\theta_3) & \dot{\theta}_4\alpha_{R14}\sin(\theta_1-\theta_4) & \dot{\theta}_5\alpha_{R15}\sin(\theta_1-\theta_5) \\ \dot{\theta}_1\alpha_{R21}\sin(\theta_2-\theta_1) & 0 & 0 & \dot{\theta}_4\alpha_{R24}\sin(\theta_2-\theta_4) & 0 \\ \dot{\theta}_1\alpha_{R31}\sin(\theta_3-\theta_1) & 0 & 0 & \dot{\theta}_4\alpha_{R34}\sin(\theta_3-\theta_4) & \dot{\theta}_5\alpha_{R35}\sin(\theta_3-\theta_5) \\ \dot{\theta}_1\alpha_{R41}\sin(\theta_4-\theta_1) & \dot{\theta}_2\alpha_{R42}\sin(\theta_4-\theta_2) & \dot{\theta}_3\alpha_{R43}\sin(\theta_4-\theta_3) & 0 & \dot{\theta}_5\alpha_{R45}\sin(\theta_4-\theta_5) \\ \dot{\theta}_1\alpha_{R51}\sin(\theta_5-\theta_1) & 0 & \dot{\theta}_3\alpha_{R53}\sin(\theta_5-\theta_3) & \dot{\theta}_4\alpha_{R54}\sin(\theta_5-\theta_4) & 0 \end{bmatrix} \quad (10)$$

The gravitational forces are described by (11).

$$g(q)=\begin{bmatrix} \beta_{R1}\sin(\theta_1) \\ \beta_{R2}\sin(\theta_2) \\ \beta_{R3}\sin(\theta_3) \\ \beta_{R4}\sin(\theta_4) \\ \beta_{R5}\sin(\theta_5) \end{bmatrix} \quad (11)$$

The applied torques can be described by (12).

$$\tau=\begin{bmatrix} U_2-U_3 \\ U_3+U_4 \\ U_5-U_4 \\ U_1-U_2 \\ -U_5 \end{bmatrix} \quad (12)$$

And the trajectory information (angle, velocity and acceleration, respectively) are described by (13).

$$q = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \end{bmatrix} \qquad \dot{q} = \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \\ \dot{\theta}_5 \end{bmatrix} \qquad \ddot{q} = \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{\theta}_3 \\ \ddot{\theta}_4 \\ \ddot{\theta}_5 \end{bmatrix} \tag{13}$$

Note that the θ angles and velocities are the same as defined in (1) only in matrix form. Also note that the accelerations may be derived by taking the derivative of the velocity equations in (3) and applying (4). With four unknowns in (3) it was unnecessary to also determine joint accelerations to solve for (4), it is also difficult to visualize position information by choosing initial and final accelerations other than zero. Finally, it is important to note that given acceleration, velocity and position information can be derived by integrating the acceleration equation, and that this is the method preferred for use in the software simulation.

The frictional forces related to joint torques are considered zero in this simulation. This was done to keep the joint actuators generalized and avoid optimizing the controllers to specific real actuators with set coefficients of friction. This helps simplify the dynamics calculations by removing a matrix from (8) but makes the resulting controllers more difficult to optimize. Without the natural dampening effect of joint friction, small controller error corrections can easily create an under damped system requiring a higher level of optimization in the controller.

With the matrices of (9)-(13) defined, (12) can further be calculated as (14).

$$\tau = \begin{bmatrix} U_2 - U_3 \\ U_3 + U_4 \\ U_5 - U_4 \\ U_1 - U_2 \\ -U_5 \end{bmatrix} = \begin{bmatrix} \left\{ \begin{aligned} &\alpha_{R11}\ddot{\theta}_1 + \alpha_{R12}\left(\ddot{\theta}_2 \cos(\theta_1 - \theta_2) + \dot{\theta}_2^2 \sin(\theta_1 - \theta_2)\right) \\ &+\alpha_{R13}\left(\ddot{\theta}_3 \cos(\theta_1 - \theta_3) + \dot{\theta}_3^2 \sin(\theta_1 - \theta_3)\right) \\ &+\alpha_{R14}\left(\ddot{\theta}_4 \cos(\theta_1 - \theta_4) + \dot{\theta}_4^2 \sin(\theta_1 - \theta_4)\right) \\ &+\alpha_{R15}\left(\ddot{\theta}_5 \cos(\theta_1 - \theta_5) + \dot{\theta}_5^2 \sin(\theta_1 - \theta_5)\right) + \beta_{R1} \sin\theta_1 \end{aligned} \right\} \\ \left\{ \begin{aligned} &\alpha_{R22}\ddot{\theta}_2 + \alpha_{R21}\left(\ddot{\theta}_1 \cos(\theta_2 - \theta_1) + \dot{\theta}_1^2 \sin(\theta_2 - \theta_1)\right) \\ &+\alpha_{R24}\left(\ddot{\theta}_4 \cos(\theta_2 - \theta_4) + \dot{\theta}_4^2 \sin(\theta_2 - \theta_4)\right) + \beta_{R2} \sin\theta_2 \end{aligned} \right\} \\ \left\{ \begin{aligned} &\alpha_{R33}\ddot{\theta}_3 + \alpha_{R31}\left(\ddot{\theta}_1 \cos(\theta_3 - \theta_1) + \dot{\theta}_1^2 \sin(\theta_3 - \theta_1)\right) \\ &+\alpha_{R34}\left(\ddot{\theta}_4 \cos(\theta_3 - \theta_4) + \dot{\theta}_4^2 \sin(\theta_3 - \theta_4)\right) \\ &+\alpha_{R35}\left(\ddot{\theta}_5 \cos(\theta_3 - \theta_5) + \dot{\theta}_5^2 \sin(\theta_3 - \theta_5)\right) + \beta_{R3} \sin\theta_3 \end{aligned} \right\} \\ \left\{ \begin{aligned} &\alpha_{R44}\ddot{\theta}_4 + \alpha_{R41}\left(\ddot{\theta}_1 \cos(\theta_4 - \theta_1) + \dot{\theta}_1^2 \sin(\theta_4 - \theta_1)\right) \\ &+\alpha_{R42}\left(\ddot{\theta}_2 \cos(\theta_4 - \theta_2) + \dot{\theta}_2^2 \sin(\theta_4 - \theta_2)\right) \\ &+\alpha_{R43}\left(\ddot{\theta}_3 \cos(\theta_4 - \theta_3) + \dot{\theta}_3^2 \sin(\theta_4 - \theta_3)\right) \\ &+\alpha_{R45}\left(\ddot{\theta}_5 \cos(\theta_4 - \theta_5) + \dot{\theta}_5^2 \sin(\theta_4 - \theta_5)\right) + \beta_{R4} \sin\theta_4 \end{aligned} \right\} \\ \left\{ \begin{aligned} &\alpha_{R55}\ddot{\theta}_5 + \alpha_{R51}\left(\ddot{\theta}_1 \cos(\theta_5 - \theta_1) + \dot{\theta}_1^2 \sin(\theta_5 - \theta_1)\right) \\ &+\alpha_{R53}\left(\ddot{\theta}_3 \cos(\theta_5 - \theta_3) + \dot{\theta}_3^2 \sin(\theta_5 - \theta_3)\right) \\ &+\alpha_{R54}\left(\ddot{\theta}_4 \cos(\theta_5 - \theta_4) + \dot{\theta}_4^2 \sin(\theta_5 - \theta_4)\right) + \beta_{R5} \sin\theta_5 \end{aligned} \right\} \end{bmatrix} \tag{14}$$

The second layer of substitutions defined in (15) expands the α and β terms shown in (9), (10), (11) and (14).

$$\begin{aligned}
\alpha_{R11} &= B + 2Gl_1 + Al_1^2 \\
\alpha_{R12} &= Hl_1 \\
\alpha_{R13} &= Il_1 \\
\alpha_{R14} &= Gl_4 + L + Al_1l_4 - Jl_1 \\
\alpha_{R15} &= -Kl_1 \\
\alpha_{R21} &= Hl_1 \\
\alpha_{R22} &= C \\
\alpha_{R24} &= Hl_4 \\
\alpha_{R31} &= Il_1 \\
\alpha_{R33} &= D \\
\alpha_{R34} &= Il_4 \\
\alpha_{R35} &= M \\
\alpha_{R41} &= -Jl_1 + L + Gl_4 + Al_1l_4 \\
\alpha_{R42} &= Hl_4 \\
\alpha_{R43} &= Il_4 \\
\alpha_{R44} &= E - 2Jl_4 + Al_4^2 \\
\alpha_{R45} &= -Kl_4 \\
\alpha_{R51} &= -Kl_1 \\
\alpha_{R53} &= M \\
\alpha_{R54} &= -Kl_4 \\
\alpha_{R55} &= F \\
\beta_{R1} &= O - Nl_1 \\
\beta_{R2} &= -P \\
\beta_{R3} &= Q \\
\beta_{R4} &= R - Nl_4 \\
\beta_{R5} &= S
\end{aligned} \tag{15}$$

The third layer of substitutions in (16) expands the A – S terms in (15).

$$
\begin{aligned}
A &= m_1 + m_2 + m_3 + m_4 + m_5 & K &= m_5 l_{5a} \\
B &= m_1 l_{1a}^2 + i_1 + m_4 l_1^2 & L &= m_4 l_{4a} l_1 \\
C &= m_2 l_2^2 + i_2 & M &= m_5 l_{5a} l_3 \\
D &= m_3 l_{3a}^2 + i_3 + m_5 l_3^3 & N &= \left(m_1 + m_2 + m_3 + m_4 + m_5\right) g \\
E &= m_4 l_{4a}^2 + i_4 & O &= m_1 l_{1a} g + m_4 l_1 g \\
F &= m_5 l_{5a}^2 + i_5 & P &= m_2 l_2 g \\
G &= -\left(m_4 l_1 + m_1 l_{1a}\right) & Q &= m_3 l_{3a} g + m_5 l_3 g \\
H &= m_2 l_2 & R &= m_4 l_{4a} g \\
I &= -\left(m_5 l_3 + m_3 l_{3a}\right) & S &= m_5 l_{5a} g \\
J &= m_4 l_{4a}
\end{aligned}
\tag{16}
$$

Finally, all the variables defined in (16) are calculated values defined in (5) and listed in Table 3. Substituting (9)-(16) into (8) creates an equation that can produce a torque vector for the 5DOF biped given a joint trajectory for the right leg SSP. The equations shown in this section are only the end results of a considerable amount of calculation. A complete derivation of the system dynamics is available in [27].

2.3.2 Left SSP Dynamics

Similarly, to find the joint torques needed for motion for the left leg SSP, the E-L approach is again used. Beginning with (8), the component matrices are represented as: The inertial forces are described by (17).

$$
D(q) = \begin{bmatrix}
\alpha_{L11} & 0 & \alpha_{L13}\cos(\theta_1 - \theta_3) & \alpha_{L14}\cos(\theta_1 - \theta_4) & \alpha_{L15}\cos(\theta_1 - \theta_5) \\
0 & \alpha_{L22} & \alpha_{L23}\cos(\theta_2 - \theta_3) & 0 & \alpha_{L25}\cos(\theta_2 - \theta_5) \\
\alpha_{L31}\cos(\theta_3 - \theta_1) & \alpha_{L32}\cos(\theta_3 - \theta_2) & \alpha_{L33} & \alpha_{L34}\cos(\theta_3 - \theta_4) & \alpha_{L35}\cos(\theta_3 - \theta_5) \\
\alpha_{L41}\cos(\theta_4 - \theta_1) & 0 & \alpha_{L43}\cos(\theta_4 - \theta_3) & \alpha_{L44} & \alpha_{L45}\cos(\theta_4 - \theta_5) \\
\alpha_{L51}\cos(\theta_5 - \theta_1) & \alpha_{L52}\cos(\theta_5 - \theta_2) & \alpha_{L53}\cos(\theta_5 - \theta_3) & \alpha_{L54}\cos(\theta_5 - \theta_4) & \alpha_{L55}
\end{bmatrix}
\tag{17}
$$

The centripetal and coriolis forces are described by (18).

$$C(q,\dot{q})=\begin{bmatrix} 0 & 0 & \dot{\theta}_3\alpha_{L13}\sin(\theta_1-\theta_3) & \dot{\theta}_4\alpha_{L14}\sin(\theta_1-\theta_4) & \dot{\theta}_5\alpha_{L15}\sin(\theta_1-\theta_5) \\ 0 & 0 & \dot{\theta}_3\alpha_{L23}\sin(\theta_2-\theta_3) & 0 & \dot{\theta}_5\alpha_{L25}\sin(\theta_2-\theta_5) \\ \dot{\theta}_1\alpha_{L31}\sin(\theta_3-\theta_1) & \dot{\theta}_2\alpha_{L32}\sin(\theta_3-\theta_2) & 0 & \dot{\theta}_4\alpha_{L34}\sin(\theta_3-\theta_4) & \dot{\theta}_5\alpha_{L35}\sin(\theta_3-\theta_5) \\ \dot{\theta}_1\alpha_{L41}\sin(\theta_4-\theta_1) & 0 & \dot{\theta}_3\alpha_{L43}\sin(\theta_4-\theta_3) & 0 & \dot{\theta}_5\alpha_{L45}\sin(\theta_4-\theta_5) \\ \dot{\theta}_1\alpha_{L51}\sin(\theta_5-\theta_1) & \dot{\theta}_2\alpha_{L52}\sin(\theta_5-\theta_2) & \dot{\theta}_3\alpha_{L53}\sin(\theta_5-\theta_3) & \dot{\theta}_4\alpha_{L54}\sin(\theta_5-\theta_4) & 0 \end{bmatrix} \quad (18)$$

The gravitational forces are described by (19).

$$g(q)=\begin{bmatrix} \beta_{L1}\sin(\theta_1) \\ \beta_{L2}\sin(\theta_2) \\ \beta_{L3}\sin(\theta_3) \\ \beta_{L4}\sin(\theta_4) \\ \beta_{L5}\sin(\theta_5) \end{bmatrix} \quad (19)$$

The applied torques can be described by (20).

$$\tau=\begin{bmatrix} U_2-U_3 \\ U_3+U_4 \\ U_5-U_4 \\ -U_2 \\ U_6-U_5 \end{bmatrix} \quad (20)$$

And the trajectory information (angle, velocity and acceleration, respectively) are the same as described by (13) as long as the trajectories are calculated for a left leg SSP.

With the matrices of (13), (17)-(19) defined, (20) can further be calculated as (21).

$$\tau=\begin{bmatrix}U_2-U_3\\U_3+U_4\\U_5-U_4\\-U_2\\U_6-U_5\end{bmatrix}=\begin{bmatrix}\left\{\begin{aligned}&\alpha_{L11}\ddot{\theta}_1+\alpha_{L13}\left(\ddot{\theta}_3\cos\left(\theta_1-\theta_3\right)+\dot{\theta}_3^2\sin\left(\theta_1-\theta_3\right)\right)\\&+\alpha_{L14}\left(\ddot{\theta}_4\cos\left(\theta_1-\theta_4\right)+\dot{\theta}_4^2\sin\left(\theta_1-\theta_4\right)\right)\\&+\alpha_{L15}\left(\ddot{\theta}_5\cos\left(\theta_1-\theta_5\right)+\dot{\theta}_5^2\sin\left(\theta_1-\theta_5\right)\right)+\beta_{L1}\sin\theta_1\end{aligned}\right\}\\\left\{\begin{aligned}&\alpha_{L22}\ddot{\theta}_2+\alpha_{L23}\left(\ddot{\theta}_3\cos\left(\theta_2-\theta_3\right)+\dot{\theta}_3^2\sin\left(\theta_2-\theta_3\right)\right)\\&+\alpha_{L25}\left(\ddot{\theta}_5\cos\left(\theta_2-\theta_5\right)+\dot{\theta}_5^2\sin\left(\theta_2-\theta_5\right)\right)+\beta_{L2}\sin\theta_2\end{aligned}\right\}\\\left\{\begin{aligned}&\alpha_{L33}\ddot{\theta}_3+\alpha_{L31}\left(\ddot{\theta}_1\cos\left(\theta_3-\theta_1\right)+\dot{\theta}_1^2\sin\left(\theta_3-\theta_1\right)\right)\\&+\alpha_{L32}\left(\ddot{\theta}_2\cos\left(\theta_3-\theta_2\right)+\dot{\theta}_2^2\sin\left(\theta_3-\theta_2\right)\right)\\&+\alpha_{L34}\left(\ddot{\theta}_4\cos\left(\theta_3-\theta_4\right)+\dot{\theta}_4^2\sin\left(\theta_3-\theta_4\right)\right)\\&+\alpha_{L35}\left(\ddot{\theta}_5\cos\left(\theta_3-\theta_5\right)+\dot{\theta}_5^2\sin\left(\theta_3-\theta_5\right)\right)+\beta_{L3}\sin\theta_3\end{aligned}\right\}\\\left\{\begin{aligned}&\alpha_{L44}\ddot{\theta}_4+\alpha_{L41}\left(\ddot{\theta}_1\cos\left(\theta_4-\theta_1\right)+\dot{\theta}_1^2\sin\left(\theta_4-\theta_1\right)\right)\\&+\alpha_{L43}\left(\ddot{\theta}_3\cos\left(\theta_4-\theta_3\right)+\dot{\theta}_3^2\sin\left(\theta_4-\theta_3\right)\right)\\&+\alpha_{L45}\left(\ddot{\theta}_5\cos\left(\theta_4-\theta_5\right)+\dot{\theta}_5^2\sin\left(\theta_4-\theta_5\right)\right)+\beta_{L4}\sin\theta_4\end{aligned}\right\}\\\left\{\begin{aligned}&\alpha_{L55}\ddot{\theta}_5+\alpha_{L51}\left(\ddot{\theta}_1\cos\left(\theta_5-\theta_1\right)+\dot{\theta}_1^2\sin\left(\theta_5-\theta_1\right)\right)\\&+\alpha_{L52}\left(\ddot{\theta}_2\cos\left(\theta_5-\theta_2\right)+\dot{\theta}_2^2\sin\left(\theta_5-\theta_2\right)\right)\\&+\alpha_{L53}\left(\ddot{\theta}_3\cos\left(\theta_5-\theta_3\right)+\dot{\theta}_3^2\sin\left(\theta_5-\theta_3\right)\right)\\&+\alpha_{L54}\left(\ddot{\theta}_4\cos\left(\theta_5-\theta_4\right)+\dot{\theta}_4^2\sin\left(\theta_5-\theta_4\right)\right)+\beta_{L5}\sin\theta_5\end{aligned}\right\}\end{bmatrix} \tag{21}$$

The second layer of substitutions defined in (22) expands the α and β terms shown in (17), (18), (19) and (21).

$$
\begin{aligned}
\alpha_{L11} &= B & \alpha_{L31} &= Gl_3 & \alpha_{L51} &= Gl_5 \\
\alpha_{L13} &= Gl_3 & \alpha_{L32} &= Hl_3 & \alpha_{L52} &= Hl_5 \\
\alpha_{L14} &= L & \alpha_{L33} &= D + 2Il_3 + Al_3^2 & \alpha_{L53} &= -Kl_5 + M + Il_5 + Al_3l_5 \\
\alpha_{L15} &= Gl_5 & \alpha_{L34} &= -Jl_3 & \alpha_{L54} &= -Jl_5 \\
\alpha_{L22} &= C & \alpha_{L35} &= Il_5 + M + Al_3l_5 - Kl_3 & \alpha_{L55} &= F - 2Kl_5 + Al_5^2 \\
\alpha_{L23} &= Hl_3 & \alpha_{L41} &= L & \beta_{L1} &= O \\
\alpha_{L25} &= Hl_5 & \alpha_{L43} &= -Jl_3 & \beta_{L2} &= -P \\
 & & \alpha_{L44} &= E & \beta_{L3} &= Q - Nl_3 \\
 & & \alpha_{L45} &= -Jl_5 & \beta_{L4} &= S \\
 & & & & \beta_{L5} &= S - Nl_5
\end{aligned}
\tag{22}
$$

The third layer of substitutions in (16) expands the $A - S$ terms in (22) and relates (17)-(22) to the calculated values defined in (5) and listed in Table 3. Substituting (13), (16) and (17)-(22) into (8) creates an equation that can produce a torque vector for the 5DOF biped given a joint trajectory for the left leg SSP. The equations shown in this section are only the end results of a considerable amount of calculation. A complete derivation of the system dynamics is available in [27].

2.3.3 Under Actuated Dynamics

In the case of under actuated dynamics, the standard model must be modified to account for the uncontrolled joint's and dependent link's impact on the whole system. While becoming under actuated doesn't have any real impact on the true system dynamics, once the dynamic equations are utilized to develop a controller the difference becomes critical.

Starting with the general form equation (8), the fully actuated system dynamics can partitioned into active and passive joints shown in (23).

$$\begin{bmatrix}\tau_a\\ \tau_u\end{bmatrix}=\begin{bmatrix}D_{ar}(q) & D_{ac}(q)\\ D_{ur}(q) & D_{uc}(q)\end{bmatrix}\begin{bmatrix}\ddot{q}_r\\ \ddot{q}_c\end{bmatrix}+\begin{bmatrix}C_{ar}(q,\dot{q}) & C_{ac}(q,\dot{q})\\ C_{ur}(q,\dot{q}) & C_{uc}(q,\dot{q})\end{bmatrix}\begin{bmatrix}\dot{q}_r\\ \dot{q}_c\end{bmatrix}+\begin{bmatrix}g_a(q)\\ g_u(q)\end{bmatrix}+\begin{bmatrix}f_a(\dot{q})\\ f_u(\dot{q})\end{bmatrix}$$

(23)

Where,

$D(q)$ = represents the linking structure inertial forces

$C(q,\dot{q})$ = represents the centripetal and coriolis forces

$g(q)$ = represents the gravitational forces

$f(\dot{q})$ = represents the vector for the joint frictional forces

q = represents the vector for the joint angles at a given time

$\dot{q}$ = represents the vector for the joint velocities at a given time

$\ddot{q}$ = represents the vector for the joint accelerations at a given time

τ = represents the vector for the resulting applied joint torques

$_a$ = represents active joint forces

$_u$ = represents free passive (unlocked) joint forces

$_c$ = represents controlled joints

$_r$ = represents remaining (uncontrolled) joints

Assuming there is no torque applied to passive joints in (23), τ_u can be assumed to equal 0. Then by factoring out $\ddot{q}_r$ and substituting the result back in, (23) can be rewritten as (24).

$$\tau_a=\bar{D}(q)\ddot{q}_c+\bar{C}(q,\dot{q})\dot{q}_c+\bar{H}(q,\dot{q})\dot{q}_r+\bar{g}(q)+\bar{f}(\dot{q})$$

Where,

$$\begin{aligned}\bar{D}(q) &= D_{ac}(q)-D_{ar}(q)D_{ur}^{-1}(q)D_{uc}(q)\\ \bar{C}(q,\dot{q}) &= C_{ac}(q,\dot{q})-D_{ar}(q)D_{ur}^{-1}(q)C_{uc}(q,\dot{q})\\ \bar{H}(q,\dot{q}) &= C_{ar}(q,\dot{q})-D_{ar}(q)D_{ur}^{-1}(q)C_{ur}(q,\dot{q})\\ \bar{f}(\dot{q}) &= f_a(q)-D_{ar}(q)D_{ur}^{-1}(q)f_u(q)\\ \bar{g}(q) &= g_a(q)-D_{ar}(q)D_{ur}^{-1}(q)g_u(q)\end{aligned}$$

(24)

Further simplification can represent (24) as (25).

$$\tau_a = \bar{D}(q)\ddot{q}_c + \bar{b}(q,\dot{q})$$

Where,

$$b(q,\dot{q}) = C(q,\dot{q})\dot{q} + g(q) + f(\dot{q}) = \begin{bmatrix} b_a(q,\dot{q}) \\ b_u(q,\dot{q}) \end{bmatrix}, \text{ representing 'other' forces} \quad (25)$$

$$\bar{b}(q,\dot{q}) = \mathrm{b}_a(q,\dot{q}) - \mathrm{D}_{au}(q)\mathrm{D}_{uu}^{-1}(q)\mathrm{b}_u(q,\dot{q})$$

Performing this partition and substitution converts the fully actuated system dynamics into its possible under actuated forms. Doing so effectively reduces the corresponding matrices order by the number of passive joints. For example, a 5DOF system with [5x5] fully actuated system matrices with one passive joint, would be represented with [4x4] under actuated system matrices once the partition and substitution were performed. In this way, the contributions of the passive joints to the dynamics are included in the remaining controllable joint dynamics in the under actuated system model [33].

2.4 Simulating Walking Motion

In this experiment, the proposed 5DOF planar biped in Fig. 6 and the proposed walking gait are both modeled in a Matlab software simulation. Matlab scripts were created to work with Simulink (a graphical, block diagram based Matlab program) and the SimMechanics Toolbox to render the dynamics and kinematics of the biped system. SimMechanics allows the definition of physical structures using parameters such as mass, length and inertia, and can model many properties of links, joints and actuators in a simulated environment with data and graphical outputs. This section will describe the use of this powerful software to construct the biped model and walking gait trajectories,

and how the simulation is used to create ideal joint torque data that performs that motion. Since the right and left SSPs trajectories are mirror images of one another and the corresponding behavior is nearly identical, only the right leg SSP is fully defined in this paper.

2.4.1 Walking Gait Generation in Matlab

To generate the walking gait used in the simulation, a Matlab code is written that takes the variables in (2) and produces the desired joint trajectories (1) using the simple path polynomial solution developed in (4). The right leg SSP chosen values for (2) are listed in Table 4 and 5, joint trajectories are given in both radians (what the simulation uses) and in degrees for ease of visualization. The trajectories are chosen to maintain a periodic gait while keeping the torso upright.

Table 4. Right SSP joint trajectory generation initial conditions in radians.

Initial Conditions	θ_1	θ_2	θ_3	θ_4	θ_5
q_0 (rad)	0.6632	-0.6632	0	0	-0.6632
q_f (rad)	0.6632	0	0.6632	-0.6632	-0.6632
$\dot{q}_0$ (rad/s)	0.03	0.03	0.03	-0.03	-0.03
$\dot{q}_f$ (rad/s)	0.03	0.03	0.03	-0.03	-0.03

Table 5. Right SSP joint trajectory generation initial conditions in degrees.

Initial Conditions	θ_1	θ_2	θ_3	θ_4	θ_5
q_0 (deg)	38	-38	0	0	-38
q_f (deg)	38	0	38	-38	-38
$\dot{q}_0$ (deg/s)	1.7	1.7	1.7	-1.7	-1.7
$\dot{q}_f$ (deg/s)	1.7	1.7	1.7	-1.7	-1.7

The velocities ($\dot{q}$) are chosen to be an arbitrary small number (0.03 radians per second in these simulations), to indicate that the beginning and end of each SSP over a continuous and periodic walking gait are in a reasonably slow but non-stopped state, similar to a real walking gait. Mapping these initial angles to the model in Fig. 6 gives a right leg SSP starting and ending position illustrated in Fig. 7.

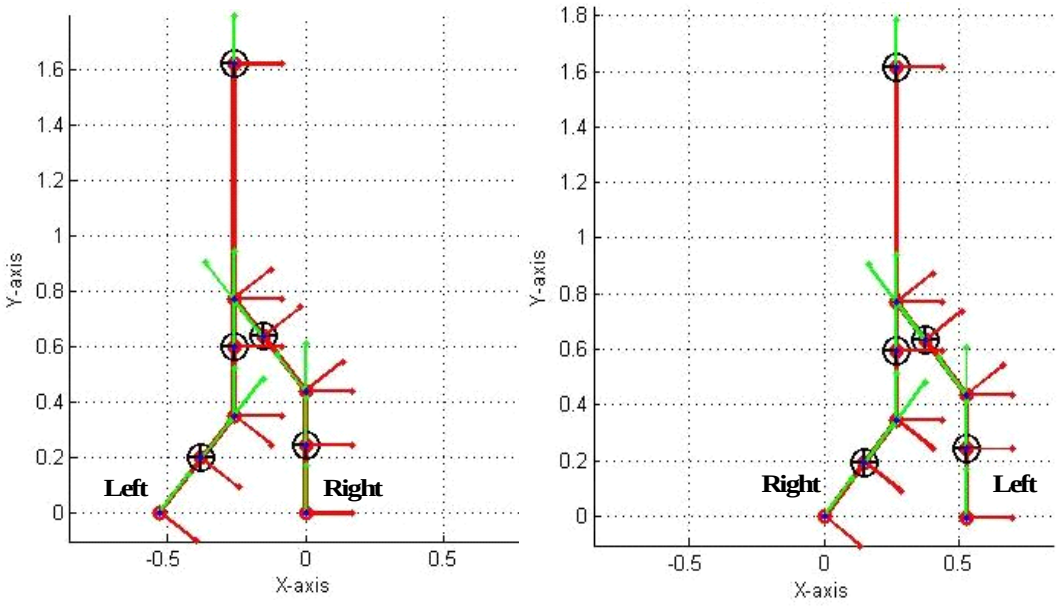


Figure 7. Initial state of right SSP (left) and end state of right SSP (right) generated from gait generation initial condition parameters as displayed in SimMechanics.

Assuming the SSP is 90% of the total gait time [19], a full gait cycle of 2 seconds is selected. By applying it to the Fig. 4 format, an approximate SSP duration is given as 0.9 seconds and the DSP is 0.1 seconds, making t_0 = 0, and a t_f = 0.9 for the right leg SSP. Choosing a reasonable calculation interval (to control simulation runtime) of 0.1 seconds and solving (1) for each interval and each joint from SSP initial to end states gives the trajectory path information listed in Tables 6 and 7.

Table 6. Right SSP joint trajectory path in radians and rad/s as calculated in Matlab simulation.

Time (s)	θ_5	θ_3	θ_2	θ_1	θ_4		$\dot{\theta}_5$	$\dot{\theta}_3$	$\dot{\theta}_2$	$\dot{\theta}_1$	$\dot{\theta}_4$
0	-0.6632	0	-0.6632	0.6632	0		0	0	0	0	0
0.1	-0.6653	0.0248	-0.6384	0.6653	-0.0248		-0.0207	0.2482	0.2482	0.0207	-0.2482
0.2	-0.6658	0.0863	-0.5769	0.6658	-0.0863		-0.0052	0.6147	0.6147	0.0052	-0.6147
0.3	-0.6652	0.1739	-0.4893	0.6652	-0.1739		0.0059	0.8765	0.8765	-0.0059	-0.8765
0.4	-0.6639	0.2773	-0.3859	0.6639	-0.2773		0.0126	1.0336	1.0336	-0.0126	-1.0336
0.5	-0.6625	0.3859	-0.2773	0.6625	-0.3859		0.0148	1.086	1.086	-0.0148	-1.086
0.6	-0.6612	0.4893	-0.1739	0.6612	-0.4893		0.0126	1.0336	1.0336	-0.0126	-1.0336
0.7	-0.6606	0.5769	-0.0863	0.6606	-0.5769		0.0059	0.8765	0.8765	-0.0059	-0.8765
0.8	-0.6611	0.6384	-0.0248	0.6611	-0.6384		-0.0052	0.6147	0.6147	0.0052	-0.6147
0.9	-0.6632	0.6632	0	0.6632	-0.6632		-0.0207	0.2482	0.2482	0.0207	-0.2482

Table 7. Right SSP joint trajectory path in degrees and deg/s as calculated from Table 6.

Time (s)	θ_5	θ_3	θ_2	θ_1	θ_4		$\dot{\theta}_5$	$\dot{\theta}_3$	$\dot{\theta}_2$	$\dot{\theta}_1$	$\dot{\theta}_4$
0	-37.998	0	-37.998	37.998	0		0	0	0	0	0
0.1	-38.118	1.420	-36.577	38.118	-1.420		-1.186	14.220	14.220	1.186	-14.220
0.2	-38.147	4.944	-33.053	38.147	-4.944		-0.297	35.219	35.219	0.297	-35.219
0.3	-38.113	9.963	-28.034	38.113	-9.963		0.338	50.219	50.219	-0.338	-50.219
0.4	-38.038	15.888	-22.110	38.038	-15.888		0.721	59.220	59.220	-0.721	-59.220
0.5	-37.958	22.110	-15.888	37.958	-22.110		0.847	62.223	62.223	-0.847	-62.223
0.6	-37.884	28.034	-9.963	37.883	-28.034		0.721	59.220	59.220	-0.721	-59.220
0.7	-37.849	33.053	-4.944	37.849	-33.053		0.338	50.219	50.219	-0.338	-50.219
0.8	-37.878	36.577	-1.420	37.878	-36.577		-0.297	35.219	35.219	0.297	-35.219
0.9	-37.998	37.998	0	37.998	-37.998		-1.186	14.220	14.220	1.186	-14.220

Note that even when the initial velocities are set to be a non-zero value, the software still assumes that at exactly time = 0, velocities are zero, but at near zero they

are as stated in Tables 4 and 5. Also note that to reach the desired positions in the given step time, the velocities for a given joint does not necessarily end at the stated final velocities. This would indicate that the gait as defined would start the left thigh, torso and right foot as entering the DSP with much more velocity than originally intended. This relates to the additional forward momentum the walking movement produced in completing the SSP. Since the DSP is assumed to be able to stabilize the gait motion and the joint angle information is correct to produce a periodic gait, it is still sufficient to test the controller as intended. Finally, as the gait progresses through the time steps, it visually appears as more of a "lunging" stride as the lifted foot slides just above the ground plane during the SSP and is illustrated in Fig. 8. These details of the gait could be addressed more thoroughly by using more advanced gait generation methods in [28].

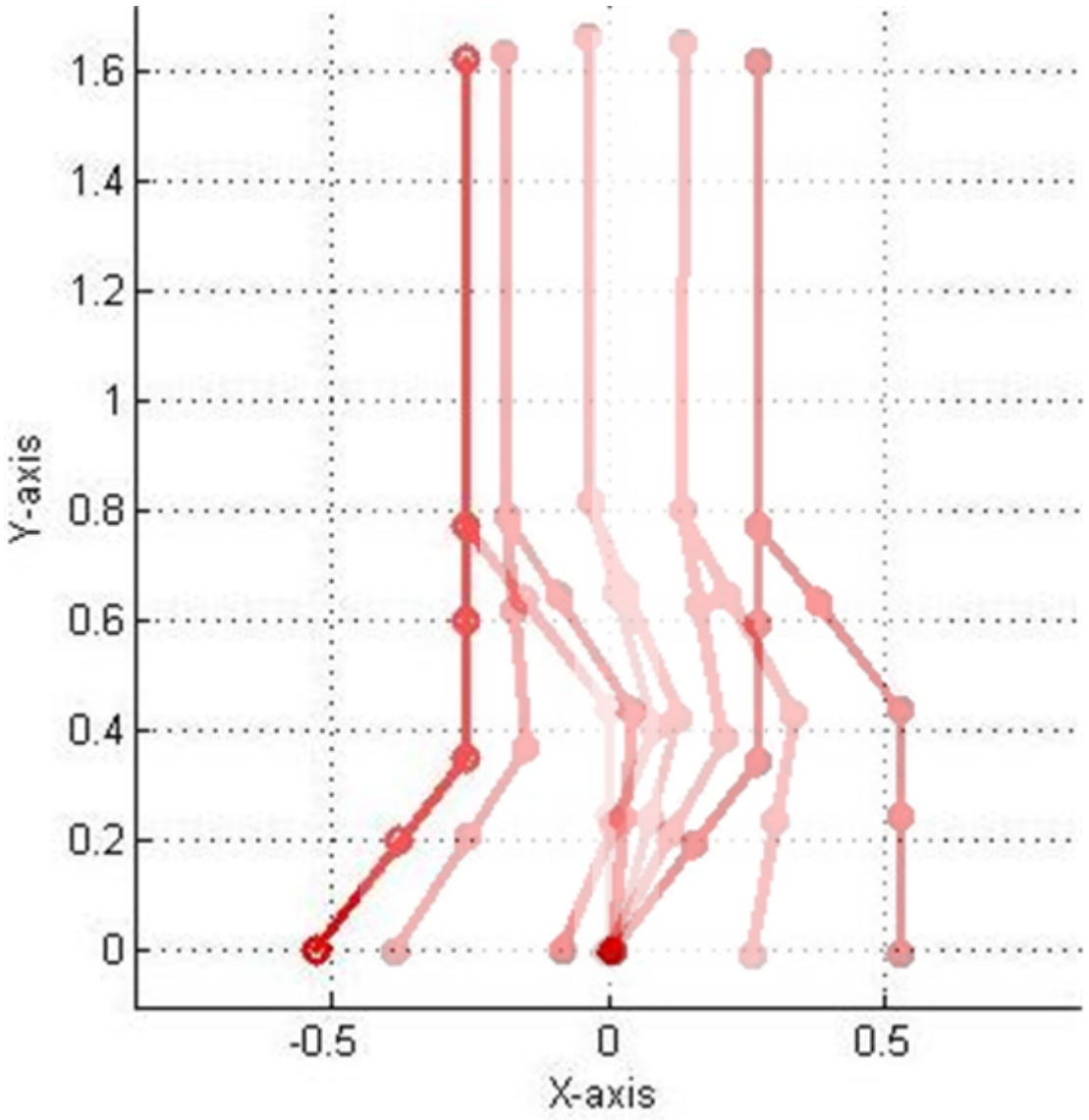


Figure 8. Illustration of Matlab calculated walking gait motion.

2.4.2 The SimMechanics Biped Model

The Matlab SimMechanics Toolbox makes it possible to construct models of physical structures in Simulink by defining their parameters. Using basic SimMechanics Toolbox blocks for joints, links and actuators, the biped robot model is constructed.

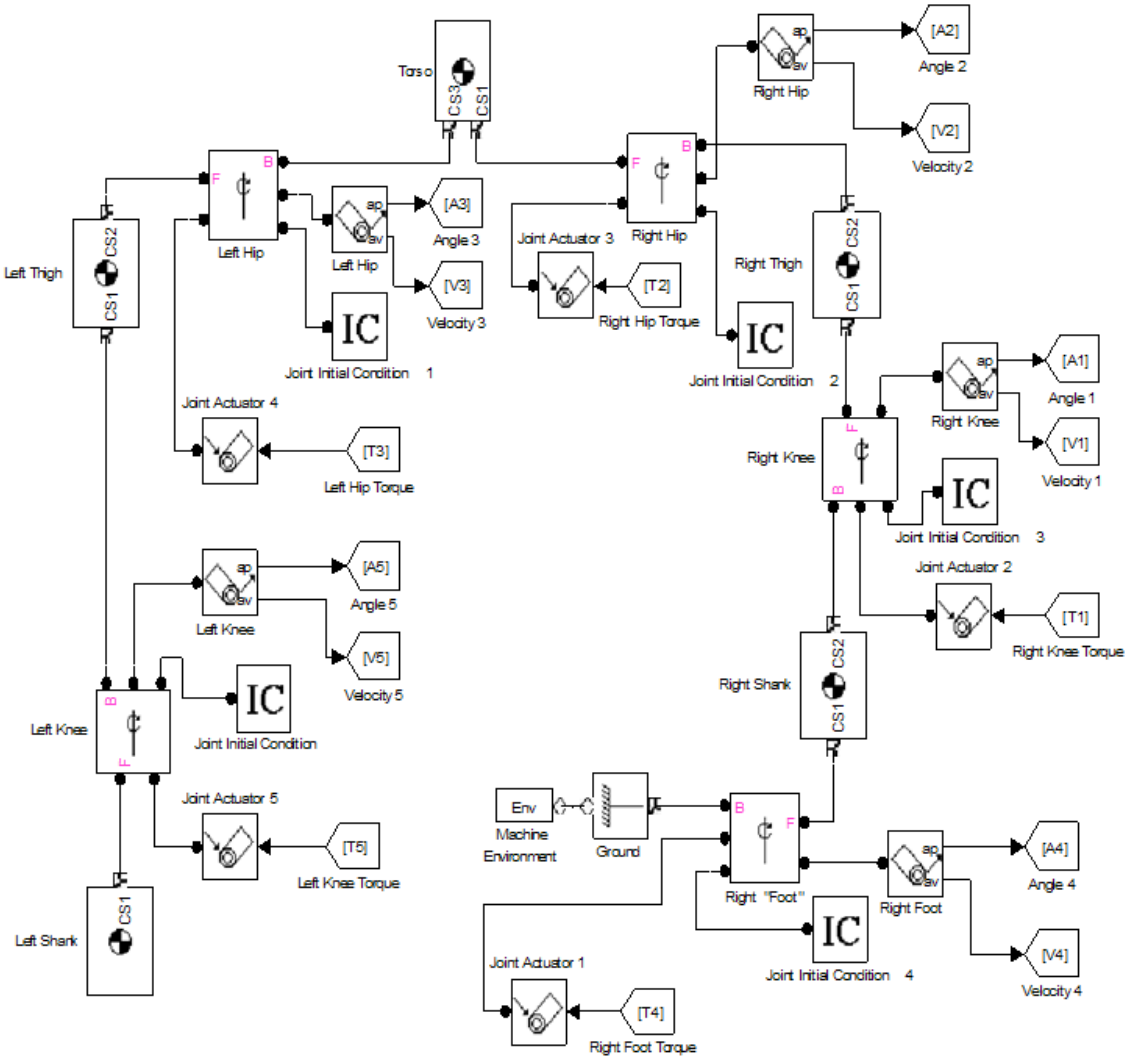


Figure 9. Block diagram of right SSP biped robot system, as modeled in SimMechanics.

Link blocks in SimMechanics must be connected together with joint blocks and to perform kinematics simulations, like this effort, the mechanical system must have a joint fixed to a point in the simulation space, labeled in Fig. 9 as the Ground block. The ground point in the simulation is (0, 0) on the sagittal plane and, for the right leg SSP, is the point where the right foot touches the ground plane. The Machine Environment block provides the type of simulation run, direction and force of the gravity vector and the

visualization windows used to generate the stick diagrams of the model and motion simulations.

Each joint block gives information on the type of joint and, since these are all 1DOF rotational joints, an axis for direction of rotation. The link blocks are where the physical parameters in Table 3 can be input. Each link block requires mass, moment of inertia, and length information input into the proper orientation as a position vector. Links also need to know how they are translated from a proceeding or following link, location of their center of gravity and what coordinate system their parameters are defined relative too. All of this information is used to define the physical structure of the robot system in the simulation space and display it as seen in Fig. 7.

Each joint is also attached to an actuator, initial condition and measurement blocks. Actuators represent electric motors, hydraulics or other force creating devices and apply that force to the links at the joint. Each actuator is fed trajectory or force information from outside the robot system block. A measurement block is attached to each joint block to monitor the output trajectory or force information resulting from inputs to the actuators and can send it out of the robot system block for further use and analysis. Finally, each joint block is also attached to an initial conditions block that provides the starting position and velocity information for that joint.

The biped model is initially defined in what is called a home position, represented as the far left pose in Fig. 4. Designating a good home position in robotics has several advantages over an arbitrary or more functioning position by allowing simplified dynamics calculations. By standing straight up with both legs together, the biped link length, position and inertial vectors are all easily defined in single y-axis directions in the

simulation space. If the home position was at the varied angles of the periodic walking gait, all the dynamics vector definitions must account for the lengths, inertia, etc. in x and y directions for this planar biped, and x, y, z directions in a 3 dimensional simulation space. Once the dynamics are defined in the simplified home position, the initial condition blocks place the biped into its proper starting gait position for the simulations.

2.4.3 Simulating Ideal Trajectories to Find Ideal Torques

To demonstrate the biped model and walking gait, a simulation is run to find the ideal joint torques needed to produce the desired walking gait. This torque information will be useful in developing the controller in the next chapter. Utilizing the developed biped model in Fig. 9, and the trajectory information generated from the Matlab code in Table 6, an inverse kinematics simulation is run. Inverse kinematics are useful in determining the desired pose and from that, the torques required to meet that pose can be recorded. The resulting ideal motions are displayed in Fig. 10, and a plot of the produced ideal joint torque information is shown in Fig. 11.

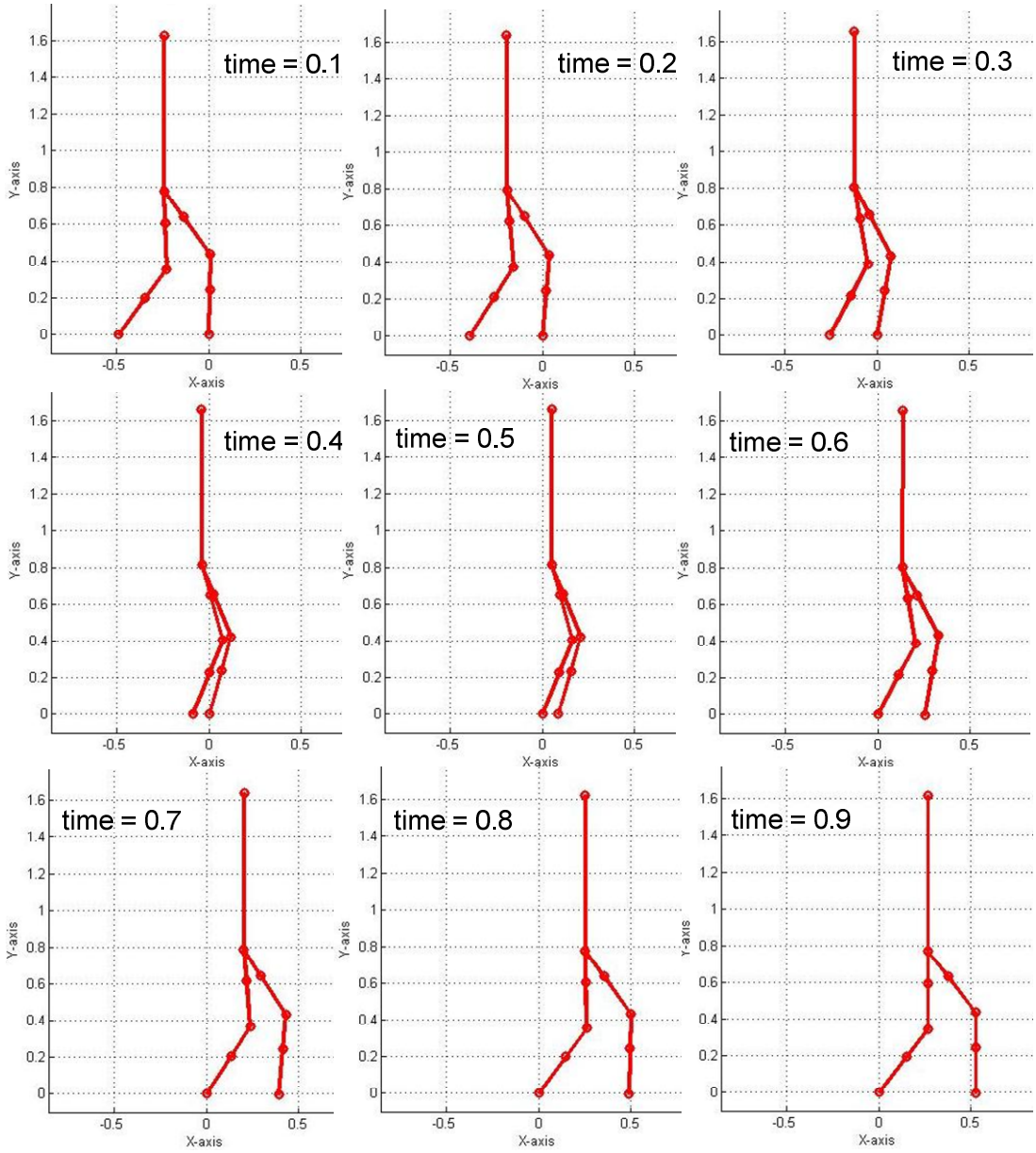


Figure 10. Ideal motion over time (in seconds) of right leg SSP using the inverse kinematics simulation.

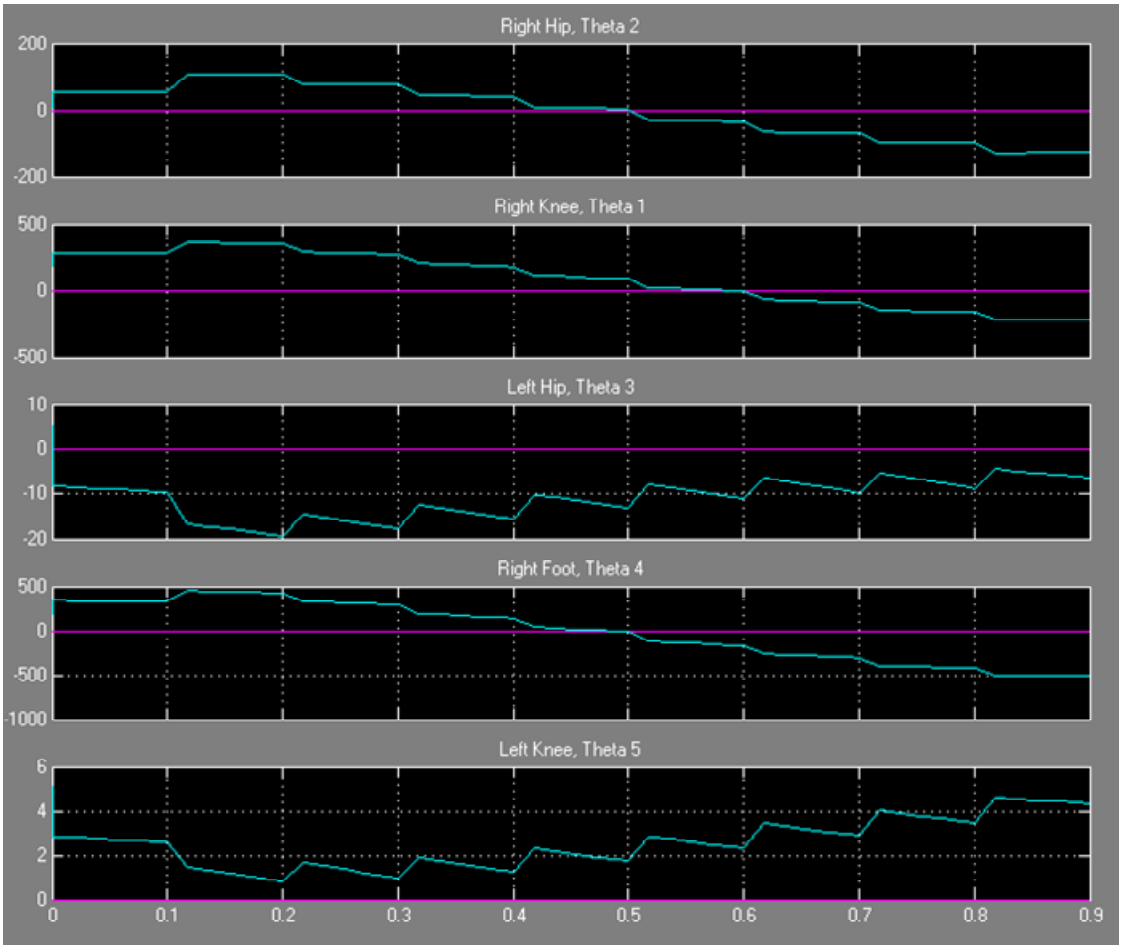


Figure 11. Resulting ideal joint torques during the right leg SSP using the inverse kinematics simulation.

Fig. 11 shows the torques at each time step for each joint, calculated during the inverse kinematics simulation. The torque forces labeled on the y-axis are in Newton-meters and the x-axis is the time in seconds. At each time step, the new desired trajectory information is input into the biped's dynamics in Fig. 9. The torque required to reach that trajectory is calculated and output to the Fig. 11 readout. The torque appears applied in a stair step fashion since the trajectory information is input to the actuators at each time step. As each new trajectory is input, the output torque displays a ramp up period, directly related to the inertia parameter for the link that joint applies torque to. The graph

scale on each joint's y-axis is also important to note; the amount of torque in Newton-meters required to move a particular joint is directly related to the amount of structure mass that joint is trying to move. It can be seen that the left knee and hip, which are swinging free in the air, need much less torque when compared to the right hip which positions the torso's mass or the right foot, which during this right leg SSP, holds the entire biped system upright.

With this simulation, the SimMechanics biped model is shown to perform the same function as the dynamics calculations in (14). It also shows that the SimMechanics biped can move in a human-like way, and that the gait trajectories visually create a human-like walking motion.

3. Controlling Biped Motion

Now that the robot dynamics are modeled and a human-like walking gait has been successfully applied to it, a controller can be developed to automate the system. To build the controller, first a state space representation of the model must be developed. From state space, a Linear Quadratic Regulator (LQR) controller can be developed. Using state feedback control of the resulting LQR the control laws are defined and optimized. Finally, a series of simulations are run to measure the effectiveness of the controllers compared to the ideal walking gait.

3.1 State Space Representation

A common way to develop a controller starts with converting the model into a state-space format. The general form representation of a state space model is the general form shown in Fig. 12.

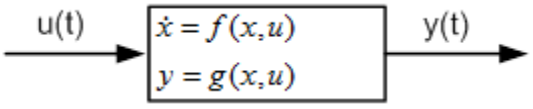


Figure 12. General form of a state space model representation.

Where *u(t)* represents m system control inputs, *y(t)* represents *k* system outputs and the block represents a system whose behavior can be described as a first order differential equation in terms of the internal state *x(t)* and inputs *u(t).* A special case of the general

form state space is for linear time-invariant (LTI) systems is shown in (26) and is extensively used in this controller design [34, 35].

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

Where,
x = The state vector
y = The output vector
u = The input vector

(26)

and,
A must be an n by n matrix, where n is the number of states
B must be an n by m matrix, where m is the number of inputs
C must be an k by n matrix, where k is the number of outputs
D must be an k by m matrix, representing direct feed through

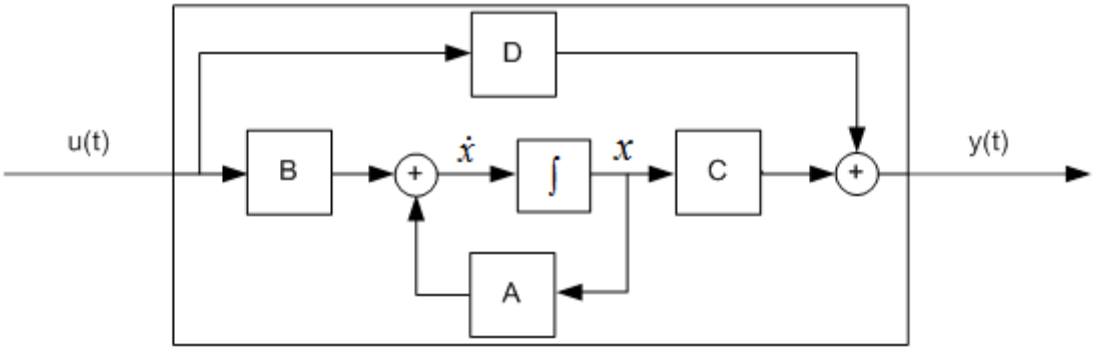


Figure 13. Visualization of state space model as described in (26).

The 5DOF biped model is a similar system to an inverted pendulum only more complicated and as such is inherently nonlinear [27]. This means that the biped system is not directly applicable to the LTI state space form unless it is first linearized around an operating point. By choosing a time *t*, a state vector *x* is defined using (13) and (4) and then solving for (14) produces torques creating controller input vector *u*. In this way, *x* and *u* are defined in (27).

$$x = \begin{bmatrix} q \\ \dot{q} \end{bmatrix}, \text{ a (10 x 1) state vector (trajectories)}$$
$$u = [\tau], \text{ a (5 x 1) input vector (torques)} \tag{27}$$

This defines the number of states n = 10, number of inputs m = 5, and number of outputs k = 10 for this 5DOF system as defined in (26).

With x and u defined, the system can be linearized around the operating point defined at time t. As long as the system nonlinearities are not too great near the operation point, the system will be a close enough approximation of a LTI system to use one of many LTI controller design methods successfully. If the system nonlinearities cause too great a change too fast, the LTI approximation will not produce an effective controller and another operating point must be chosen and subsequent controller designed to attempt to successfully control the system. If control of the system is sufficiently nonlinear, it could lead to the need for several controllers over the course of the walking gait and is explored in the results section.

Now that the biped is approximated as a LTI system, the biped model can be converted into the LTI state space form. Using a LTI equivalent position/velocity state space from [34, 35], the remaining matrices can be filled in from (8), where:

$$Ax = \begin{bmatrix} \dot{q} \\ -D^{-1}(q)\{C(q,\dot{q})\dot{q} + f(\dot{q}) + g(q)\} \end{bmatrix}, \text{ where A is a (10 x 10) state matrix and x the state vector, resulting in a (10 x 1) vector}$$
$$B = \begin{bmatrix} 0 \\ D^{-1}(q) \end{bmatrix}, \text{ a (10 x 5) input matrix} \tag{28}$$
$$C = [I], \text{ a (10 x 10) output identy matrix}$$
$$D = [0], \text{ a (10 x 5) null direct feed through matrix}$$

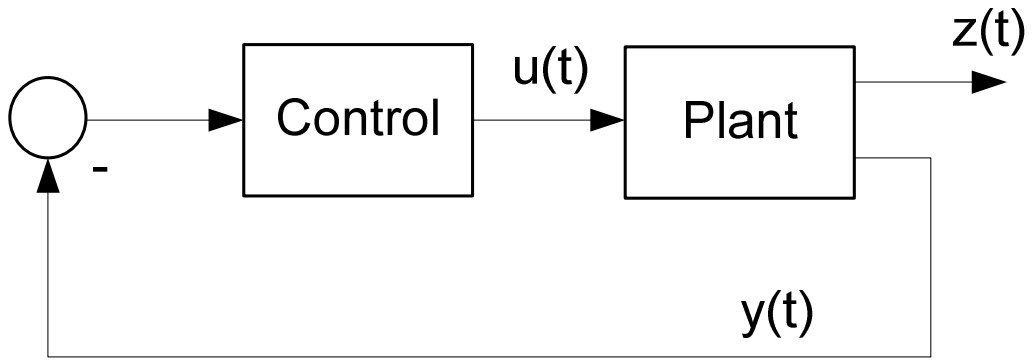
It is shown in (28) that matrices *A* and *B* represent the robot dynamics model, *C* chosen such that full state information (trajectories) will be output and *D* chosen so that no input torque will be fed forward to the output *y*. From this model, many LTI controller development and analysis techniques can be utilized.

3.2 Introduction to LQR Control

With the model dynamics converted into state space form, a controller can now be developed. A $H\infty$ controller was originally developed to test the fully and under actuated systems, modified from the methods described in [33, 39]. Unfortunately, once the complexity was increased to 5DOF and the $H\infty$ controller developed, optimizing it to perform the walking gait was taking upwards of 92 hours to simulate on a 2006 laptop, per iteration. This was due primarily to performing a (5x5) determinant calculation required for each of the state space matrices. This slowed the optimization process, which is primarily trial and error, to the point in which another solution was required. While several other control methods have been used in similar systems [17, 19, 25, 27-30, 34, 40], due to the nature of applying the walking gait, in which an ideal trajectory signal is created and the controller will attempt to match it, a type of regulator was deemed a good choice and a linear quadric regulator (LQR) was selected.

3.2.1 Basic LQR Control

The core of LQR control is a standard negative feedback loop described in Fig. 14.



Where,

$\dot{x} = Ax + Bu$, Defining the plant or model system

$y = Cx + Du$, Controlled plant output

$z = Gx + Hu$, Controlled output (if different from y)

Figure 14. General form of a LQR feedback control loop.

Optimization of this control scheme is typically to find a controller matrix *C* that reduces the output *z* of the system to as small a number as possible. The design criterion is to find *u(t)* that minimizes the general form quadratic cost function (29).

$$\text{(Energy) } J_{LQR} = \int_0^\infty x^T Qx + \gamma u^T Ru + 2x^T Nu \; dt \tag{29}$$

Where,

x = the state vector

u = the control vector

T = stands for transpose

γ = a positive trade-off constant

Q, R, N = positive definite weighting matrices

Note that $x^T Qx$ is the energy of the controlled output, $u^T Ru$ is the energy of the control signal, $2x^T Nu$ is the correlation factor between *x* and *u*, and γ establishes the tradeoff between minimizing the controlled output and control signal. The weighting matrices *Q*, *R*, and *N* set the importance of state feedback, controller outputs and correlation factors respectively.

When using LQR one attempts to minimize both $x^T Qx$ and $u^T Ru$ energies but reducing one increases the other. The use of γ allows adjustment over the tradeoff of each energy as desired for the specific control problem. If γ is chosen to be very large, J_{LQR} is minimized by using little control signal energy but at the expense of requiring a large controlled output energy. Conversely, if γ is chosen very small, then J_{LQR} is minimized by using very small controlled output energy at the expense of large control signal energy.

Assuming that the entire state *x* can be measured and is available for control allows state feedback control to be used. Using (28), *C* is assumed an identity matrix and matrix *D* is considered 0, reducing *y(t)* in Fig. 14 to *x*. This allows the criterion in (29) to be optimally solved with (30).

$$u = -Kx \tag{30}$$

Where,
K = the controller gain matrix

This converts the block diagram in Fig. 14 into the Fig. 15 representation, leading to the most basic form controller. [36-38]

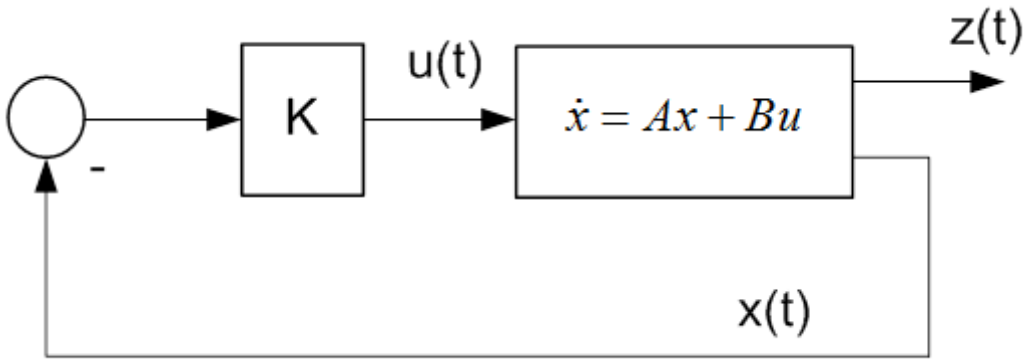


Figure 15. LQR model solving for State-Feedback Control.

3.2.2 LQR Control with a Regulator

In these simulations it is desirable to have the output *z,* rather than minimized as small as possible with the basic LQR, converge to a constant set point as fast as possible. This can be described with (31) and is known as set point control.

If,	and,	Then,	
$x \to x^*$, (converges)	$x^* = Fx$	$0 = Ax^* + Bu^*$	(31)
$u \to u^*$, (converges)	$u^* = Lu$	$r = Gx^* + Hu^*$	

Where r is the desired set point, x^* is the desired state, u^* is the desired control and F and L are general form equation gains.

Using Fig. 14 with (31), if the number of plant inputs are equal to the number of controlled outputs, then (32) is true.

$$\begin{bmatrix} A & B \\ G & H \end{bmatrix}\begin{bmatrix} x^* \\ u^* \end{bmatrix} = \begin{bmatrix} 0 \\ r \end{bmatrix} \Leftrightarrow \begin{bmatrix} x^* \\ u^* \end{bmatrix} = \begin{bmatrix} A & B \\ G & H \end{bmatrix}^{-1}\begin{bmatrix} 0 \\ r \end{bmatrix} \tag{32}$$

When the number of plant inputs is greater than the number of controlled outputs, then the system is considered over actuated and multiple solutions could exist for (32). If the

number of plant inputs is less than the number of controlled outputs, then the system is considered under actuated, and (32) may not have any solution at all. If an under actuated solution does exist, then it will only exist for certain values of *r* and in those cases the definitions of x^* and u^* in (32) will still hold true. This expands the general form of set point control in Fig. 15 to as illustrated in Fig. 16.

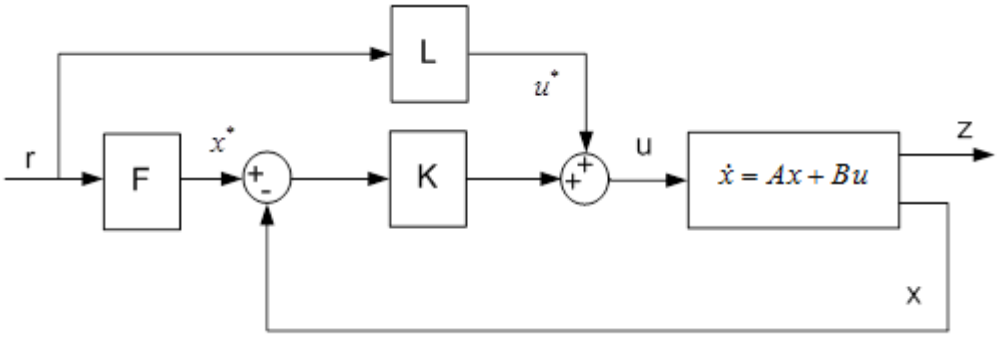


Figure 16. LQR model solving for State-Feedback Control with a set point regulator.

Using (29) and the state-feedback controller form in Fig. 15, including the regulator into the system allows (30) to take the form of (33), allowing *z* to converge to set point *r*.

$$\begin{aligned} u &= -K(x - x^*) + u^* \\ &= -Kx + (KF + L)r \end{aligned} \tag{33}$$

It is important to note that in (33) the $(x - x^*)$ term is the system error or in other words, the difference between the desired reference state and the current plant state. This error signal is input into the controller to influence the state of the plant and completes the feedback loop. Also, including the assumptions used to create the basic state feedback form in Fig. 15 and assuming that the gain *F* allows full information (equal to one) and *L*

is zero since torque feed forward is undesirable, (33) can be rewritten as (34) minimizing the criterion in (29).

$$\begin{aligned} u &= -Kx + Kr \\ &= K(r - x) \end{aligned} \tag{34}$$

Thus making the optimal controller the gain of error signal and reducing Fig. 16 to as shown in Fig. 17.

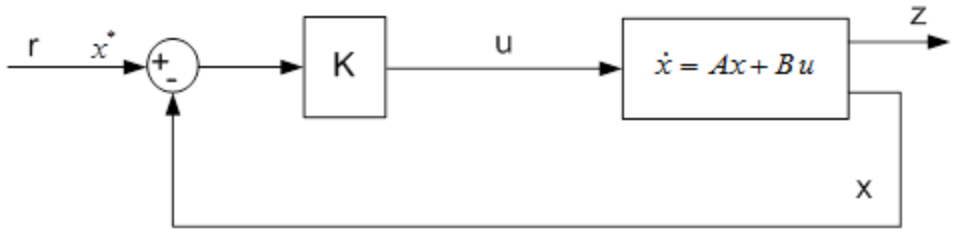


Figure 17. Optimal LQR model using state feedback controller.

The system block diagram in Fig. 17 is now in the form utilized for the remaining biped controller system simulations. [36-38]

3.3 LQR Controlled Motion Simulation

3.3.1 Creating the LQR Control System

The proposed LQR controller is created with a Matlab script. The Matlab command to linearize a system, called 'linmod', requires the system model and an operation point (desired system state and desired control inputs of the model at a given time t) to create the LTI state space model matrices *A*, *B*, *C* and *D* shown in (28). Utilizing the Fig. 9 SimMechanics robot model, the ideal trajectory information x^* from

Table 6 and the resulting ideal input torque information u^* from Fig. 11, enough information is available to linearize the system dynamics using linmod and also create a regulator input *r* as shown in Fig. 17 at any desired time during the walking gait. With the (28) matrices calculated, the terms *Q*, *R*, *N* and γ in (29) are chosen to develop the LQR quadratic cost function. The Matlab command 'lqr' creates the LQR controller by using the state space dynamics model matrices *A* and *B*, and the quadratic cost function terms, returning with the controller gain matrix *K* shown in Fig. 17.

Initially, the *Q* and *R* matrices are considered to be identity matrices, allowing unbiased weighting of controlled outputs and control inputs to be available in balancing out the quadratic cost function (29). Also, γ is also chosen to be 1 initially to offer minimal user balancing of the control inputs vs. the controlled outputs energy. The γ term is extensively used later to optimize the LQR controller and is detailed more in the results section. Finally, weighting matrix *N* is chosen to be zero, removing any correlation effects between the state and input vectors. This may later be desirable in optimizing the controller, but in this initial development, it will be ignored, removing the term from the cost function.

In this case, where the state space matrices were only achieved by linearizing around an operating point (gait pose and motion dynamics at that time *t*), the developed LQR controller is only truly effective near that operating point. As the gait progresses the system nonlinearities may reduce the effectiveness of the LQR controller, requiring additional controllers to be developed and applied. This is also explored in the results section.

Since in this simulation the ideal trajectories are available and completely measurable, utilizing full state feedback control is the preferred controller form. In using state feedback in this way, any controller feed forward information (state space matrix *D*) and any feed forward ideal torque information is considered undesirable and are removed or zeroed out as appropriate to conform the final control system flow to that shown in Fig. 17. This flow is illustrated in the actual SimMechanics top level block diagram displayed in Fig. 18.

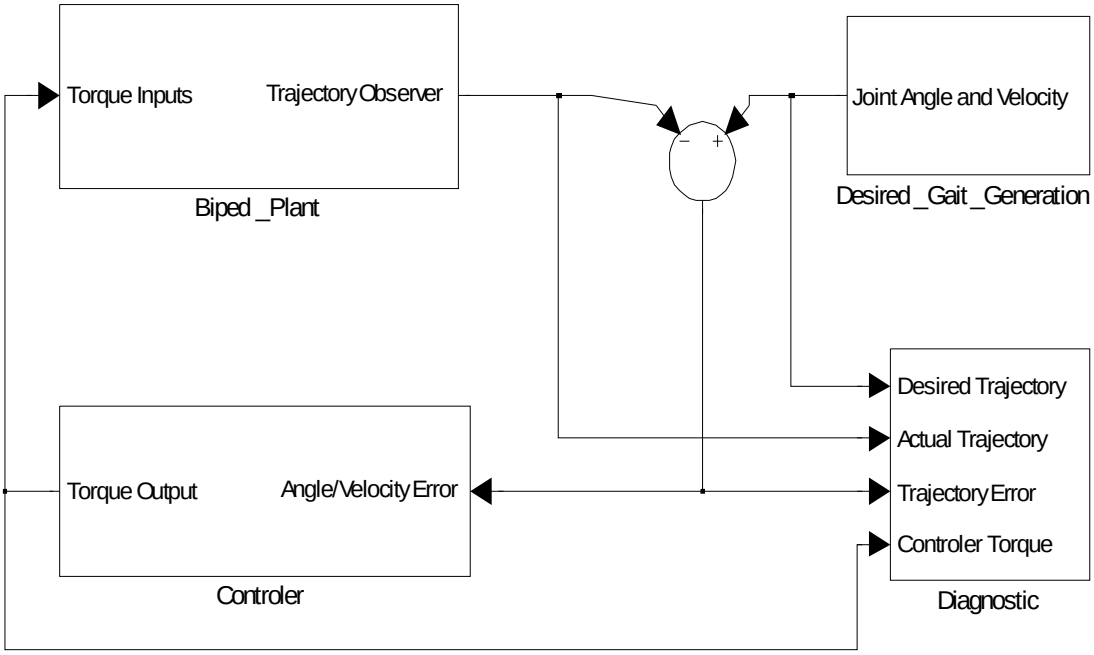


Figure 18. Block diagram of complete SimMechanics system with LQR controller, regulator, and biped robot model.

In Fig. 18, the regulator, labeled Desired_Gait_Generation, inputs into the system the full desired system state in the form of a (10x1) trajectory vector as shown in (27) and can be considered variable x^* from Fig. 16 and 17. The biped robot model, labeled Biped_Plant, is identical to the model shown in Fig. 9, only now the simulation will be

run in forward dynamics. This is done because the desired pose has already been calculated (inverse kinematics simulation in section 2.4.3) and now want the biped to reach the series of desired endpoints provided by the regulator (forward kinematics).

The outputs of the biped model block are the measured state trajectories and are equivalent to the state vector *x* from Fig. 16 and 17. Importantly, the biped model block output *x* and the regulator input x^* are summed creating the error signal that feeds into the controller gain and satisfies the minimization criterion (34) for LQR control. The outputs of the biped model are also absent of the previously described output *z* term in Fig. 16 and 17. This is done for simplicity, in this simulation output *z* is no different from *y* in Fig. 14, and is reduced to *x* as stated in (30) by using state feedback control.

The controller block is a series of LQR gain matrices created for a variety of operation points and switched in or out as needed for a particular time step. This allows the option of using multiple controllers at a range of times during the walking gait as the particular effectiveness of the developed LQR controller(s) are as of yet unknown and may require testing and optimization. The outputs of the controller block are the input torque information vector *u* from Fig. 16 and 17, and adjust the biped's movement to meet next regulator desired set point.

Finally, the only Fig. 18 deviation from the Fig. 17 form is the diagnostic block, which provides an output source for simulation data useful for analysis. The diagnostics are not part of the control system feedback loop, and have no impact on form or function.

3.3.2 The Performance Index

Optimization of the LQR controller was achieved primarily by visually observing how the stick motion model performs the walking gait as compared to the ideal in Fig. 7 and 9. Once the controlled system began to visually perform close to the ideal, recording the joint trajectory errors over the course of the walking gait also became useful for comparison. Trial and error of the various optimization efforts and walking gait results lead to many variations of the initial controller, and to help compare results a performance index (PI) was used.

$$\sum_{0}^{t}\left(\Theta_1^2+\Theta_2^2+\Theta_3^2+\Theta_4^2+\Theta_5^2\right) \tag{35}$$

The PI produces a number defined by (35), where Θ is the error signal ($(r-x)$ from (34) and shown in Fig. 17 and 18) for a given joint at a given time. The sum total of these error magnitudes over the SSP time t is the PI number for that iteration of the LQR optimal controller. The PI is calculated separately for each the position errors and the velocity errors. This gives a way to quantize the effectiveness of a particular LQR controller, especially when the resulting walking gait is visually close to its previous iterations.

Furthermore, the PI is only valid for direct comparison with results using the same total SSP time and time step (the interval that control and regulator inputs are applied over the SSP). Since the PI is a sum total of recorded errors at each time step, adding more time steps with either a longer SSP time or faster or slower time step interval will change the PI value even when using the same controller(s). Because of this, the PI is

only used to compare series of test results where the total SSP time, time step and number of joints measured are held constant.

4. Simulations and Results

With the model linearized around a variety of operating points and controllers developed to manage the walking gait at those times, tests can now be run. A series of trial and error simulations are conducted to optimize the controllers based on the resulting walking gaits when compared to the ideal. Use of the PI helps determine the controller effectiveness at keeping overall trajectory errors to a minimum and quantifies what level of errors allows for a visually acceptable walking gait.

The first tests showed that the model would collapse without the influence of a controller in the first 0.02 seconds of the SSP. This showed that not only was a controller necessary to perform the walking gait, but the first controller was needed almost immediately once the SSP began. This made it necessary to introduce the controller (or the first controller if multiple were needed) from the instant the SSP started.

As the walking gait progresses, a new regulator trajectory is input, state vector output from the biped model and controller input applied to the model at each designated time step. The time step used has two impacts on simulation results: different time steps cause control to be applied and new target trajectories are given from the regulator at faster or slower intervals. This means the results of a particular controller can be different at different time step intervals, but experimentation showed that the changes were minor overall and using intervals reducing the computational speed of the simulations took priority.

A minimally acceptable time step used throughout the experiments is 0.1 seconds. This was chosen because a longer interval did begin to have a negative impact on what was before a successful walking gait, and a shorter interval had a marginal impact at the cost of increasing the total simulation time. In a case using a time step of 0.001 seconds, the simulation could take a few hours to complete, as opposed to about 15 seconds using 0.1.

The primary tool to optimize the controllers is the use of γ and is explored in detail in the following results. Initial tests quickly found that the same optimized controller(s), regardless of γ, couldn't effectively maintain the walking gait for both the fully actuated and under actuated biped models and separate simulation models had to be developed. Therefore, testing and optimization results are broken down and compared in two separate cases, the fully actuated model and the under actuated model.

4.1 Fully Actuated Biped Motion

The fully actuated biped model is no different than that described in the previous sections. As such, the fully actuated model results are included here to establish the effectiveness of using a LQR controller on the proposed 5DOF model and baseline the controller's ability to perform the walking gait compared to the ideal. These results will help determine how effective LQR is with the under actuated case and characterize how much of that effectiveness is lost when the system becomes under actuated.

4.1.1 Initial Tests

Initial tests with the fully actuated biped quickly showed that using $\gamma = 1$ created a very poor controller producing a dramatically flawed and off balance interpretation of the ideal walking gait. Further tests determined that the controller optimization was highly sensitive to large values of γ and therefore the amount of energy in the control output of the quadratic cost function (29). This meant that effective control would depend on minimizing the control output energy by using a very small γ. Tests were run to determine how small γ was required to be and are summarized in Fig. 19.

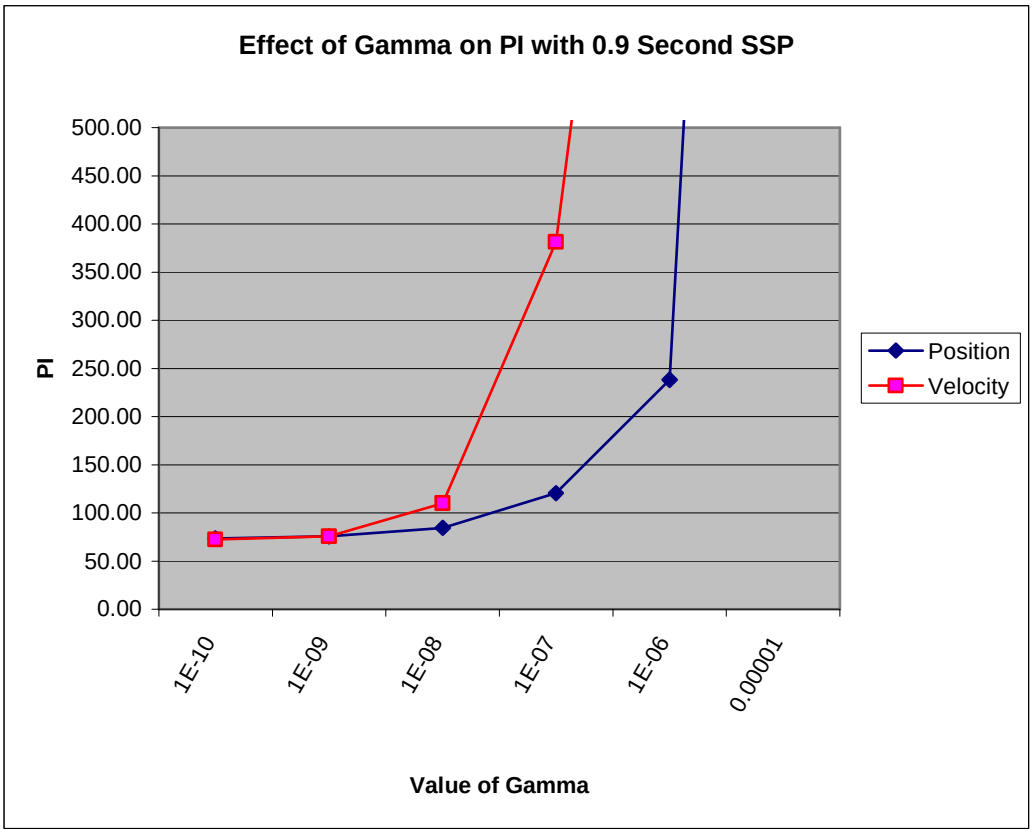


Figure 19. Effect of γ on the PI with total SSP time = 0.9 seconds and step time = 0.1 seconds.

Fig. 19 shows that γ needs to indeed be a very small number, with the threshold of a visually acceptable walking gait occurring at approximately $\gamma = 10^{-6}$. This can be verified by observing the final position of the walking gait compared to the ideal for each value of γ. These comparisons are provided in Fig. 20, 21 and 22.

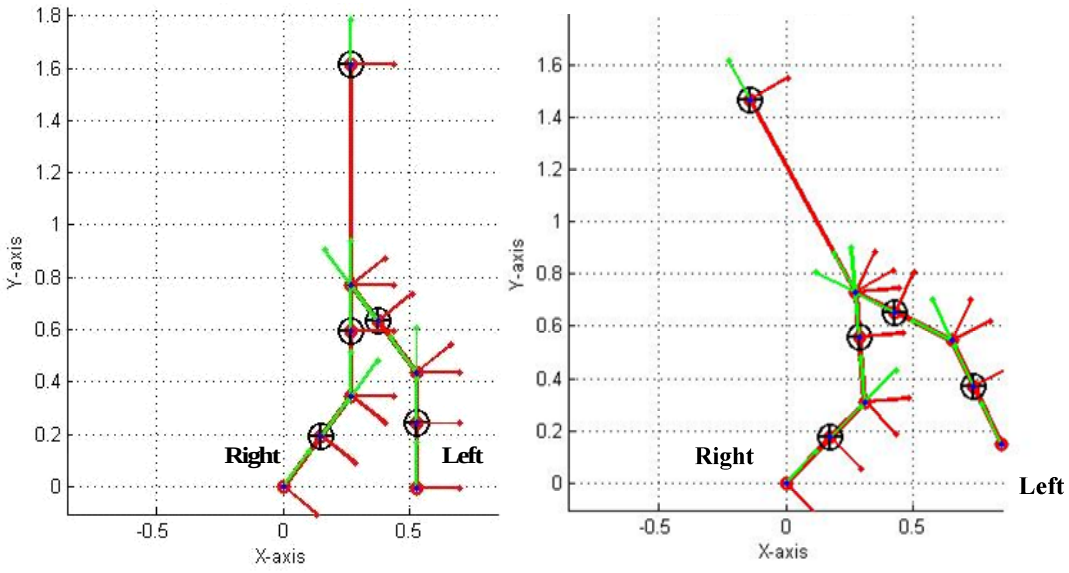


Figure 20. Final biped pose with $\gamma = 10^{-5}$, total SSP time = 0.9 seconds and step time = 0.1 seconds (Right) when compared to the ideal pose provided from the regulator (Left).

In Fig. 20 it can be easily seen that the final biped pose has significant errors compared to the ideal. Using larger γ values causes an even greater exaggeration of the final pose until the biped finally becomes uncontrollable before the end of the SSP. In this case, the biped is fully controlled and approaches this pose in a steady, deliberate fashion. Unfortunately, it doesn't actually complete the SSP and is possibly off balance as it widely over steps the placement of the left foot and enters the DSP with the torso mass well off center. As such, this case is considered to be an obvious failure of the optimized LQR controller to manage the walking gait.

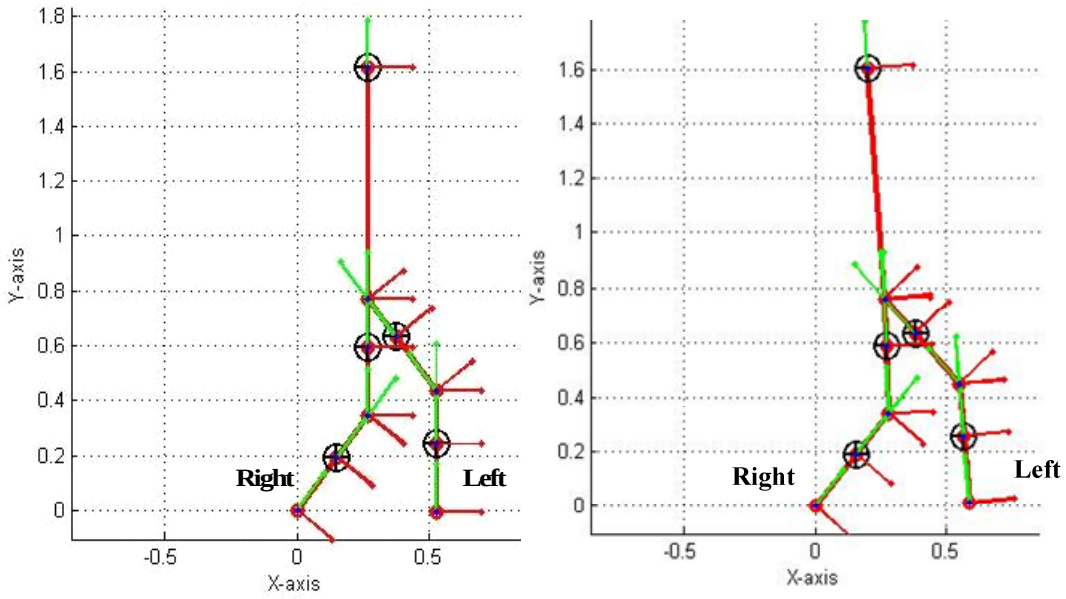


Figure 21. Final biped pose with γ = 10^{-6}, total SSP time = 0.9 seconds and step time = 0.1 seconds (Right) when compared to the ideal pose provided from the regulator (Left).

Fig. 21 shows a much better final biped pose when compared to the ideal. Some errors are still observable in that the left foot overstepped its placement some and the torso mass is slightly tilted back as the SSP ends. But the left foot did connect or nearly connected with the ground plane, completing the SSP step, thus allowing possible posture correction in the DSP. This means, that depending on how robust a developed DSP is, this could be an acceptable SSP controller. This case is notable as it defines the crossover point between the previously unsuccessful controllers and successful control of the fully actuated biped model changing only γ from the initial controller design. This is also achieved by using only one controller linearized at the time = 0 starting point.

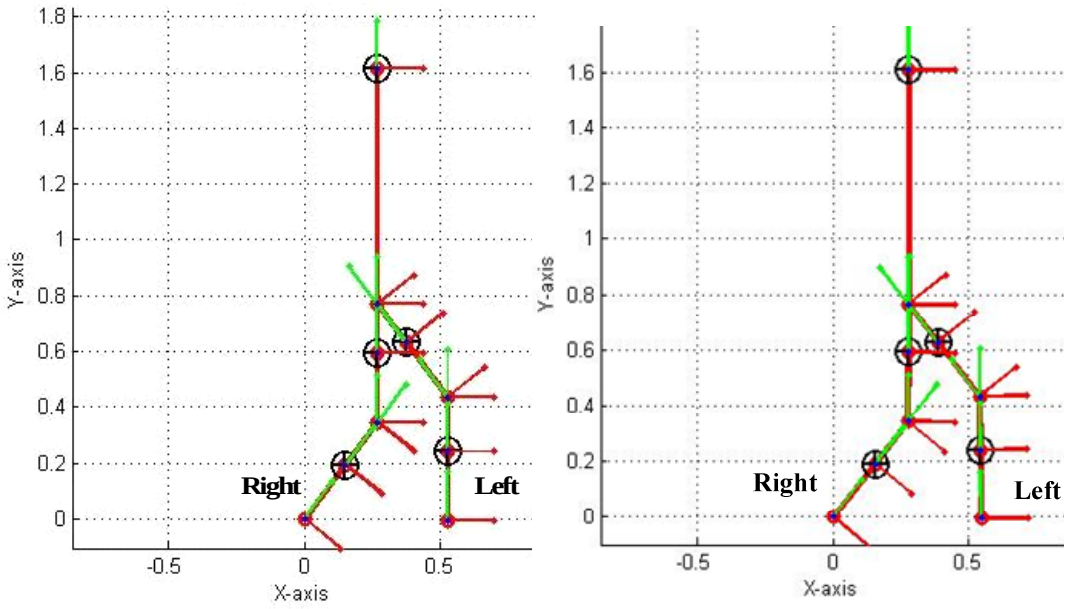


Figure 22. Final biped pose with γ = 10^{-7}, total SSP time = 0.9 seconds and step time = 0.1 seconds (Right) when compared to the ideal pose provided from the regulator (Left).

The final pose in Fig. 22 is visually near identical to the ideal. The errors displayed in Fig. 21, appear nonexistent here and only by using the PI are the differences apparent. The torso appears upright and the left foot is touching the ground plane on or very near the ideal location as the SSP ends. As such, this case is considered the largest approximate value of γ useful in matching the ideal walking gait. The further γ value tests from Fig. 19 are not shown as they are visually identical to this case. Although smaller γ values do continue to reduce the PI value, and therefore errors, they do so at less and less amounts and at the cost of increasing simulation times. Tiny values of γ could also translate into a high level of precision or complexity in a real world system. Considering this information, this case is chosen as the best (largest) overall value of γ

for fully actuated control of the proposed 5DOF biped and will be used for the remaining comparisons.

The threshold between barely completing the SSP and a near match of the ideal gait can be placed by comparing the PI for each value of γ. The PI for each case in Fig. 19 is displayed in Table 8.

Table 8. PI data points from Fig. 19.

Performance Index	$\gamma = 10^{-5}$	$\gamma = 10^{-6}$	$\gamma = 10^{-7}$	$\gamma = 10^{-8}$	$\gamma = 10^{-9}$	$\gamma = 10^{-10}$
Position	2372.13	237.95	120.57	84.40	75.72	73.40
Velocity	13962.41	1295.08	381.30	109.96	75.70	72.58

Using the above visual comparisons, the Table 8 data puts the threshold of a successful walking gait at approximately a position PI = 237 and a controlled gait very close to the ideal at an approximate position PI between 237 and 120. It can also be noted that as the position PI improves, the velocity PI improves even more so as γ is reduced. This data indicates that although the stated priority has been to match the pose of the ideal walking gait, the gait trajectories include velocity information and that errors in the controlled gait velocity do have a noticeable impact on the success of the controller.

4.1.2 Error Observations

With the best γ value chosen, it is useful to look into where and when the errors are accumulating over the course of the walking gait. This is shown in Fig. 23 and 24.

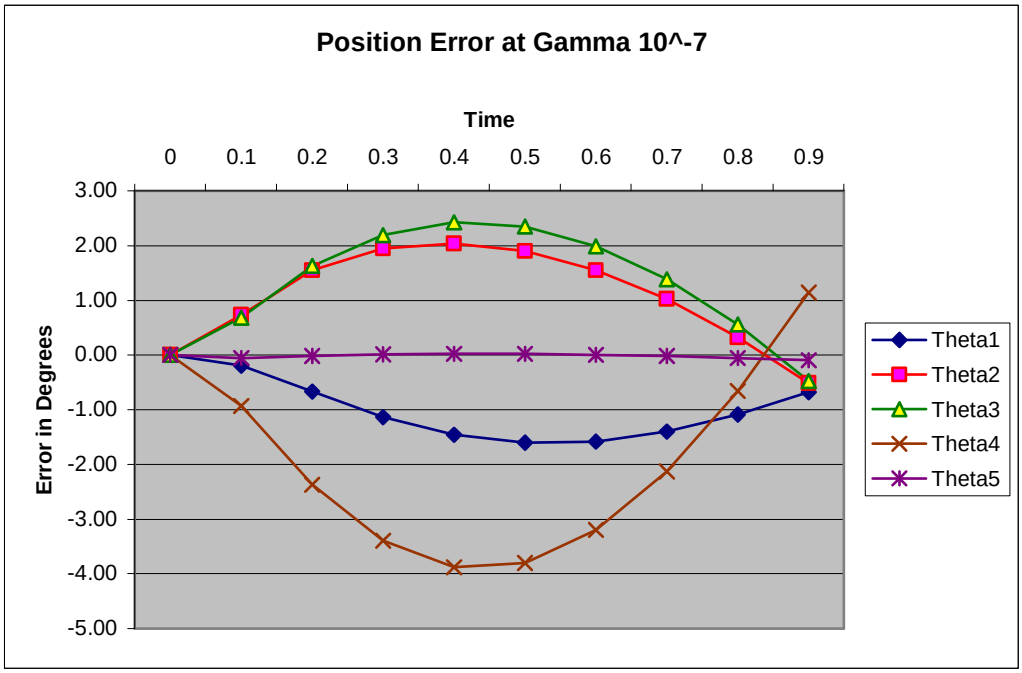


Figure 23. Position errors of each joint with $\gamma = 10^{-7}$ over the course of the fully actuated 0.9 second SSP.

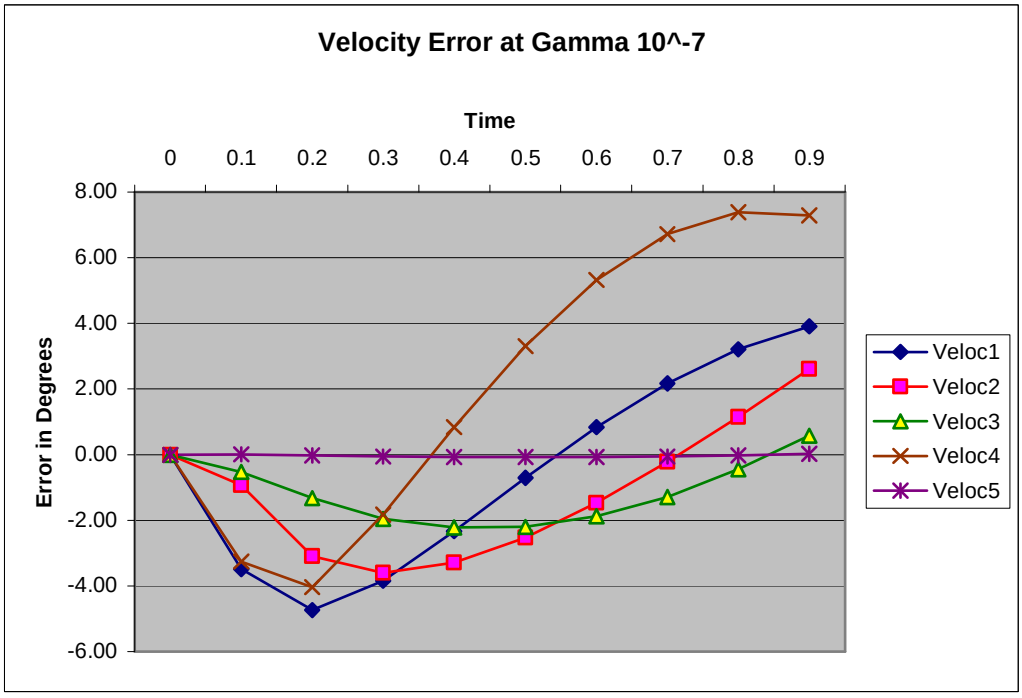


Figure 24. Velocity errors of each joint with $\gamma = 10^{-7}$ over the course of the fully actuated 0.9 second SSP.

Fig. 23 and 24 the plotted ‘Theta’ and ‘Veloc’ data correspond to the joint angle and joint velocity errors as attributed to specific joints in Fig. 6. From this the peaks and convergence of the various joint errors can be easily seen.

From Fig. 23, the position errors peak at approximately half way through the SSP and then converge at the end of the SSP. It is notable that ‘Theta4,’ associated with the right foot, experiences the largest error and the most error at the end of the SSP. This isn’t unexpected, as the right foot is holding up the combined mass of entire robot and is affected by the contributing dynamic forces from the rest of the biped over the course of the SSP. With this information, it can be said that for the fully actuated biped the joint position end errors for the right SSP are kept below1.2 degrees and the peak errors below 3.9 degrees.

For Fig. 24 the velocity errors also show some interesting behavior. First is that the velocity errors don’t exhibit the same convergence as the position errors, instead they seem dispersed at the end of the SSP. Also there are roughly two peak error times, one early and one late, in the SSP. This could indicate that the controller overshot its first major correction, and since the second peak is near the same or larger, may be exhibiting under damped behavior. It is also possible that the duration of the SSP is too short to see the joint velocities errors begin to converge. Again it is the most heavily stressed joint, ‘Veloc4’ the angular velocity of the right foot, that shows the greatest peak and SSP end errors. From this information it can be said that for the fully actuated biped, the joint velocity end errors for the right SSP are kept below 7.3 degrees per second and the peak errors below 7.4 degrees per second.

This can be compared to the smallest PI using the smallest tested value for γ shown in Fig. 25.

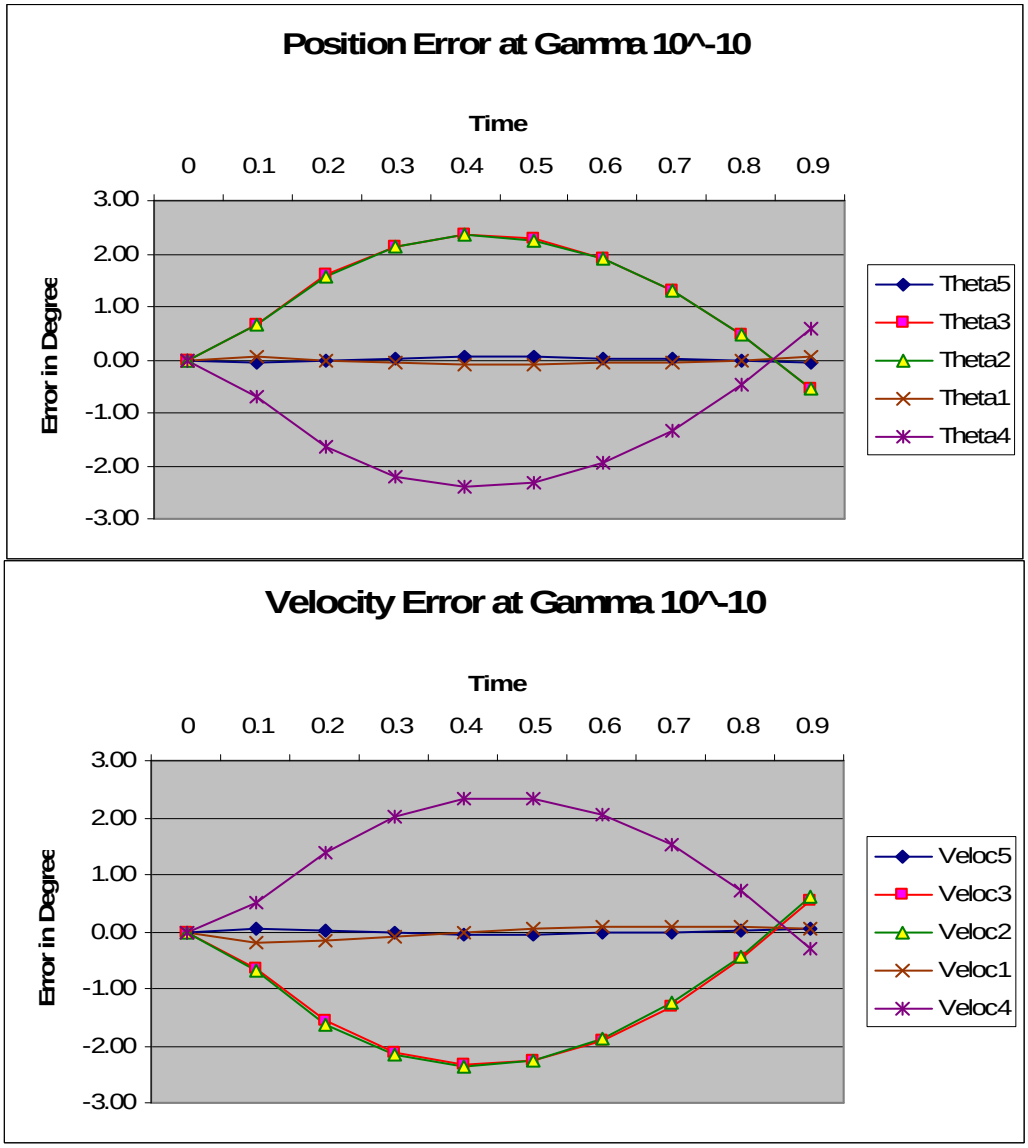


Figure 25. Position and velocity errors of each joint with γ = 10^{-10} over the course of the fully actuated 0.9 second SSP.

From Fig. 25 it can be quickly seen that compared to the Fig. 23 and 24 cases, the velocity errors are under considerably better control. Visual observation of the gait

shows no perceivable difference between these two values of γ and the ideal. The PI and joint errors show end position errors below 0.6 degrees, peak position errors below 2.5 degrees, end velocity errors below 0.6 degree per second and peak velocity errors below 2.4 degrees per second. A marked improvement, but at a computational cost, increasing the simulation run time by a factor of 50 (2.5 hours simulation time as opposed to 180 seconds). In is important to note that this control scheme can theoretically bring the errors down to this level, but the practicality of doing so in a real world system is beyond the scope of this paper. For this experiment, the minimally acceptable gait was achieved with $\gamma = 10^{-7}$ and will be utilized as the baseline for the remaining experiments.

4.1.3 Impact of Additional Controllers

Observation of the data in Fig. 23 and 24 suggest that some improvement may be possible by utilizing more than one controller. In the previous experiments, a controller was developed by linearizing around the time = 0 operating point. That same time = 0 controller was then reapplied at each successive time step until the end of the SSP. The nonlinearities of the system would follow the error and appear to grow and then reduce as the gait approaches the end of the SSP. Perhaps additional control could help reduce the SSP errors, especially with the more irregular velocity errors. A series of tests were run to determine the effect of a second controller and its placement to have the most impact on reducing errors. These results are summarized in Fig. 26.

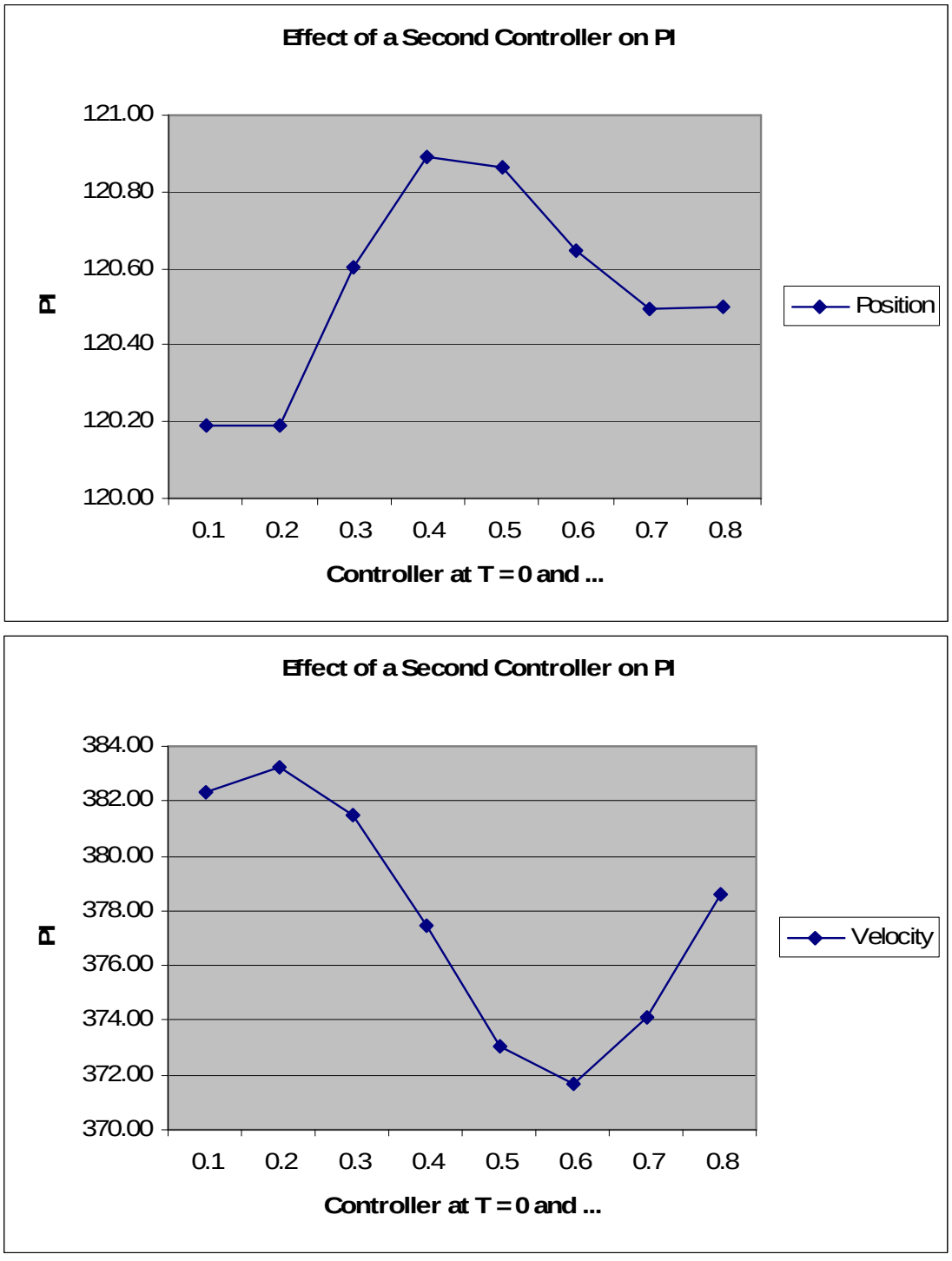


Figure 26. Changes in position and velocity PI during the right SSP with $\gamma = 10^{-7}$, a controller at time = 0 and the bottom labeled time step, total SSP time = 0.9 seconds and step time = 0.1 seconds.

The data in Fig. 26 shows the impact on the PI of adding a second controller in addition to the time = 0 linearized controller. For example, the data point at 0.6 is the resulting PI from using the time = 0 controller for each time step up to time = 0.5, then the second controller linearized for time = 0.6 is used for that time step and the remaining steps in the SSP. In addition to the data in Fig. 26 a controller linearized at every time step was run as a comparison. The changes in PI between one, the best second and several controllers are displayed in Table 9.

Table 9. PI comparison from using multiple controllers with $\gamma = 10^{-7}$, time step = 0.1 seconds and total SSP time = 0.9 seconds.

Performance Index	Controller at $t = 0$ only	Controller at $t = 0$ and $t = 0.6$	New controller at every time step
Position	120.57	120.65	120.93
Velocity	381.30	371.64	368.65

As suggested in Fig. 26 and confirmed in Table 9, the improvement on overall errors over the course of the SSP by using a second or several additional controllers is minimal. Since the changes in position PI are very small, the best case for a second controller is when it's applied at time = 0.6 seconds, where it has the most error reduction on the velocity PI.

Compared to the observations made in Fig. 19 and Table 8, the impact on PI of adding any additional controllers beyond the initial time = 0 one is negligible. The amount of change in position PI needed to visually see improvements appears to be at least 50 or more and velocity PI at least 100. Since additional controllers relate to additional hardware and complexity in a real world system and the impact on error

reduction so minimal, it is concluded that the initial time = 0 linearized controller is sufficient to perform the walking gait in the fully actuated system for this value of γ.

4.1.4 Impact of SSP Duration

A second possible way to reduce the walking gait errors, especially the velocity PI, may be achieved by slowing the walking gait down. If the velocity errors shown in Fig. 24 are primarily due to the controller being unable to converge in the given SSP duration, then slowing down the walking gait may show some improvement. To test this, another series of experiments are run with a total SSP duration of 1.8 seconds. These results are shown in Fig. 27-30, Table 10 and 11.

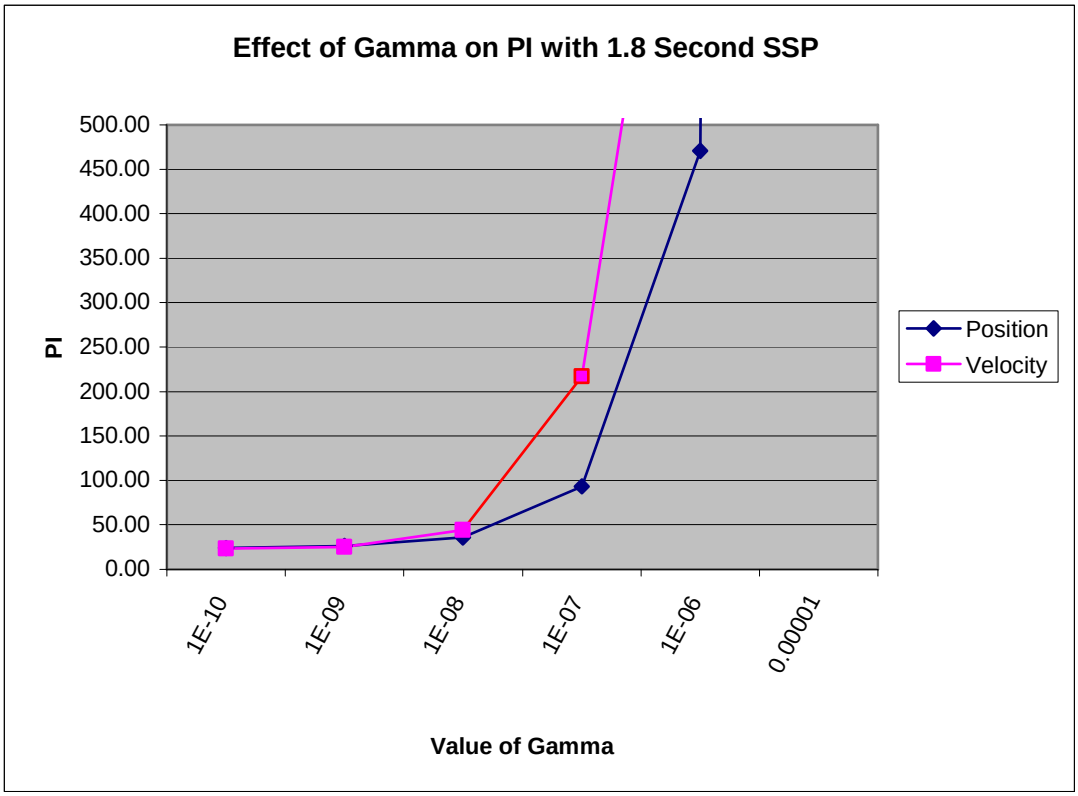


Figure 27. Effect of γ on the PI with total SSP time = 1.8 seconds and step time = 0.1 seconds.

In Fig. 27, the PI is again used to observe how errors accumulate over the course of the SSP. Visual observation of the walking gaits resulting from the various γ values show results similar to the 0.9 second SSP duration. Using $\gamma = 10^{-6}$ produces a similar barely successful and slightly off the ideal pose visually, although not as good as the 0.9 second SSP duration case. Using $\gamma = 10^{-7}$ again produces a final gait pose indistinguishable from the ideal. It is noticeable from Fig. 27 that the PI gets lower and the velocity PI converges faster in the 1.8 second SSP duration case.

Table 10. PI data points from Fig. 27.

Performance Index	$\gamma = 10^{-5}$	$\gamma = 10^{-6}$	$\gamma = 10^{-7}$	$\gamma = 10^{-8}$	$\gamma = 10^{-9}$	$\gamma = 10^{-10}$
Position	25399.84	470.98	93.13	35.93	26.22	24.14
Velocity	38221.63	1054.14	216.86	44.01	25.03	23.37

A closer look at the data points in Table 10 shows that not only is the PI worse at larger values of γ, but once past the threshold of a successful SSP completion between $\gamma = 10^{-6}$ and 10^{-7}, it reduces faster and to a lower overall error than the 0.9 second SSP duration case. This case puts the first successful optimized controller at an identical $\gamma = 10^{-7}$ value. This shows that the system is more sensitive to changes in γ with a slower SSP duration.

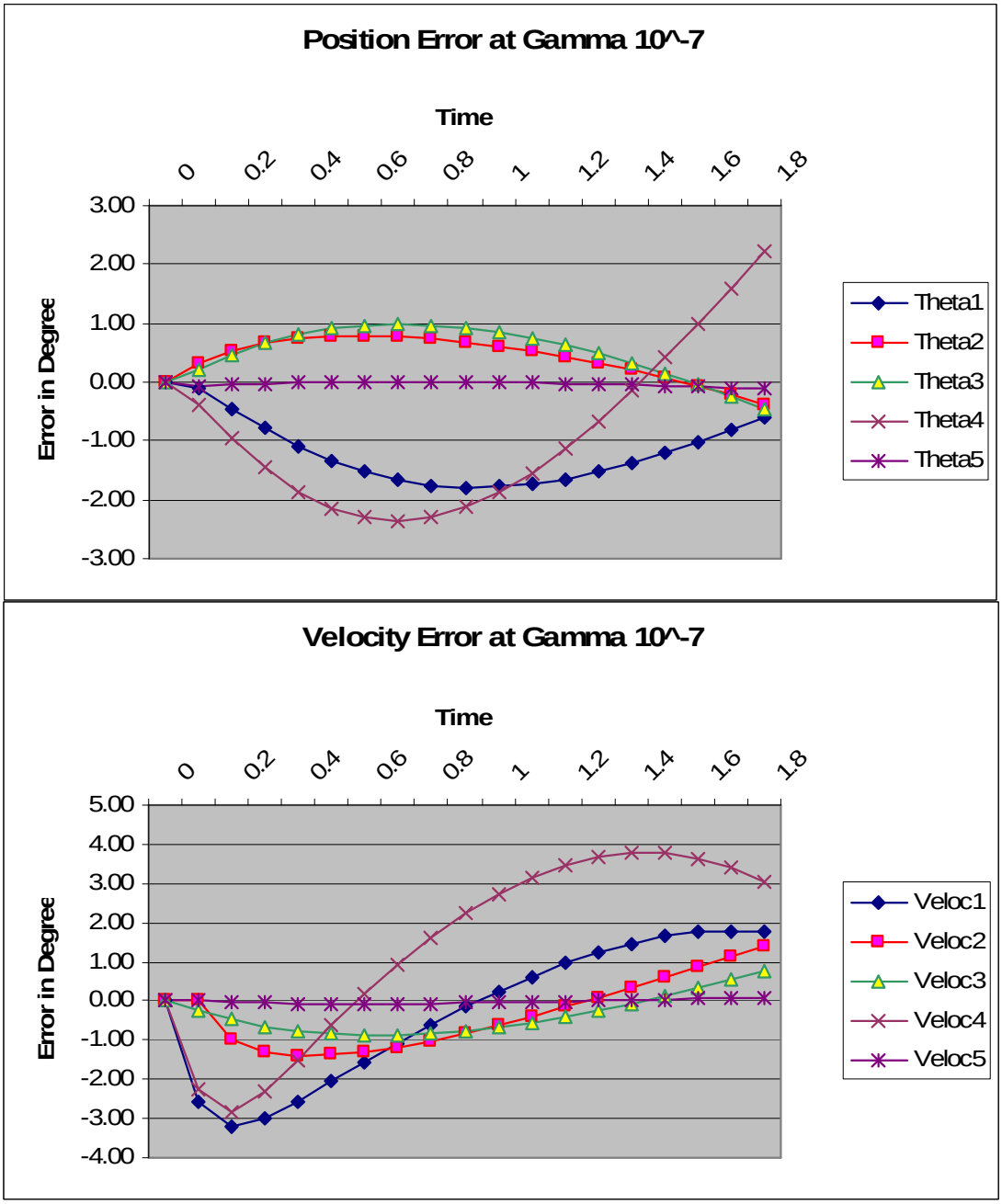
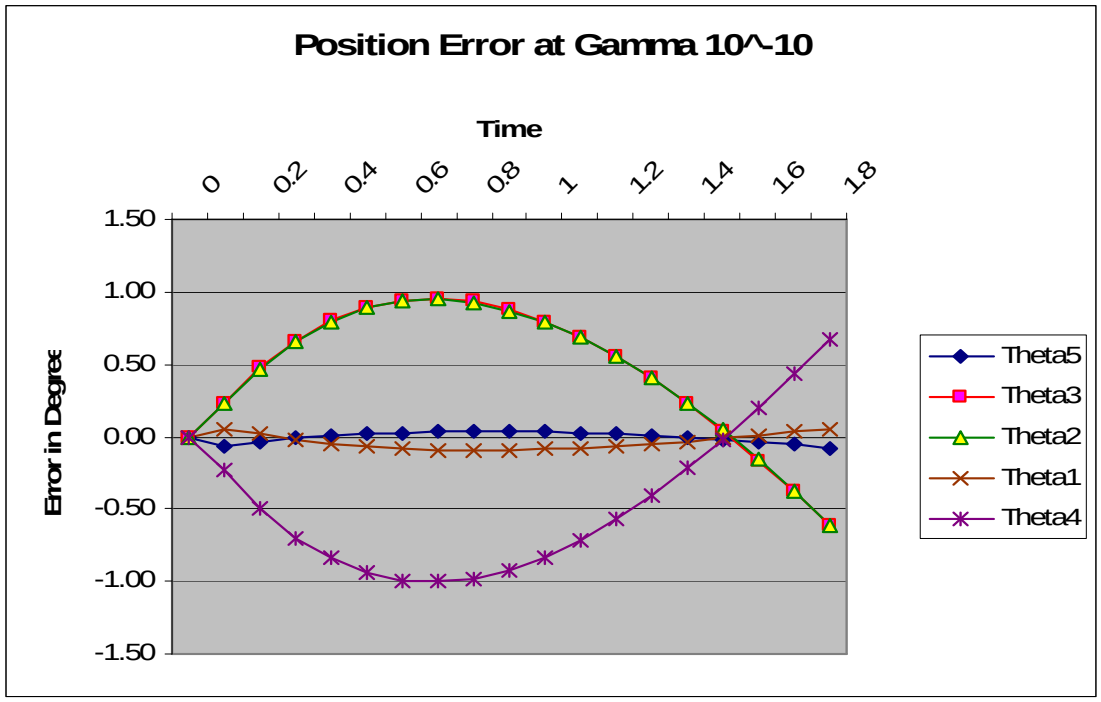


Figure 28. Position and velocity errors of each joint with $\gamma = 10^{-7}$ over the course of the fully actuated 1.8 second SSP.

In Fig. 28, similar error behavior is observed compared to the 0.9 second SSP duration time. It is noted that with the exception of the end position error of the right

foot, all the other errors (peaks and SSP end) are the same or reduced when compared to the 0.9 second SSP duration case. For comparison, the position peak and SSP end errors are below 2.2 degrees (due to the overshoot of the right foot) and the velocity peak error is below 3.9 degrees per second and the SSP end error is below 3.1 degrees per second. This means that roughly half the error is generated when the SSP time is doubled.



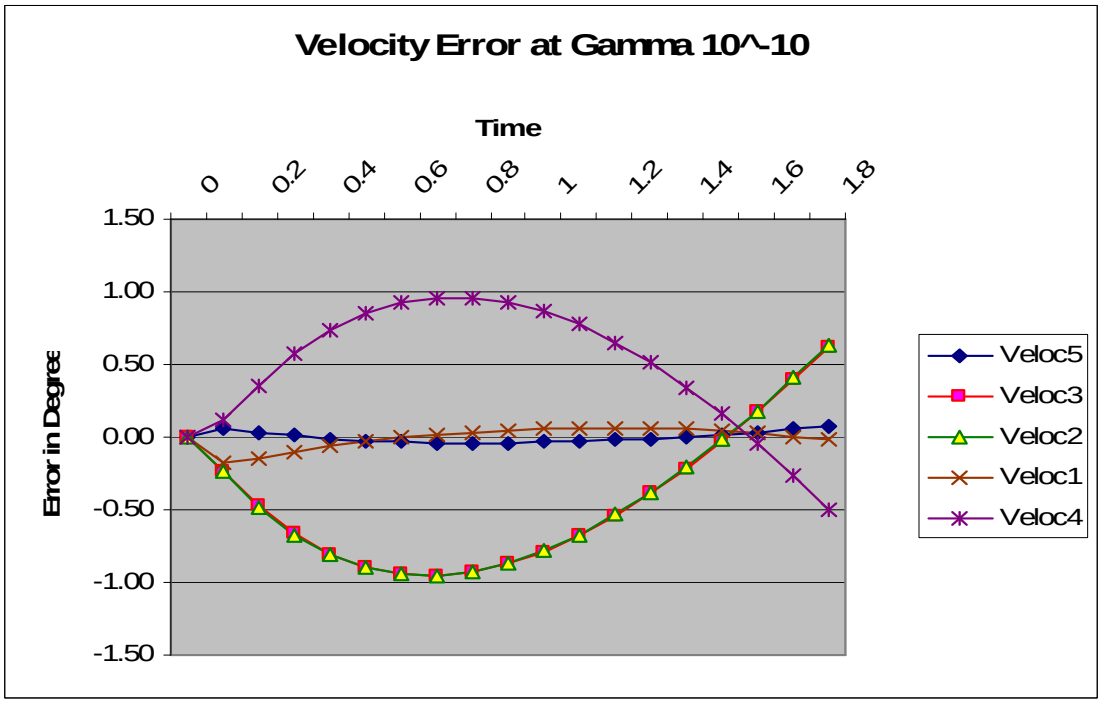


Figure 29. Position and velocity errors of each joint with $\gamma = 10^{-10}$ over the course of the fully actuated 1.8 second SSP.

Again, in Fig. 29 the smallest γ tested is shown to see just how small the errors will get with this control system. In this best possible case, position peak errors are below 1.1 degrees and SSP end errors are below 0.7 degrees. Similarly, velocity peak errors are below 1.0 degrees per second and SSP end errors are kept below 0.7 degrees per second. Compared to the 0.9 second SSP duration case, the end errors are nearly identical but the peak errors are again roughly half the faster SSP case's errors.

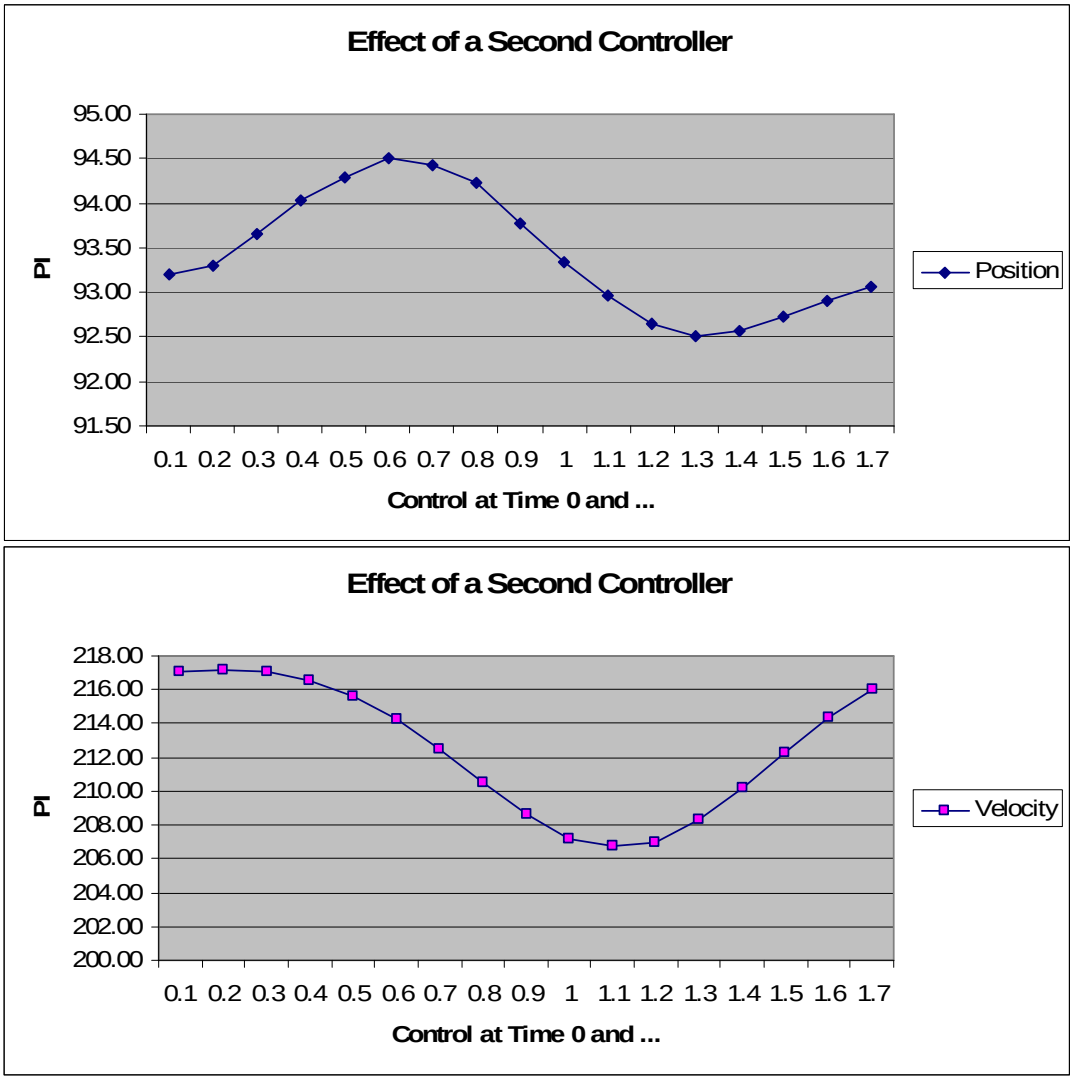


Figure 30. Changes in position and velocity PI during the right SSP with $\gamma = 10^{-7}$, a controller at time = 0 and the bottom labeled time step, total SSP time = 1.8 seconds and step time = 0.1 seconds.

As shown in Fig. 30, using the longer 1.8 second SSP duration, tests were run to determine the impact of additional controllers. The data shows slightly more effect on position error and slightly less effect on velocity error. Again, a second controller at any time step has an overall negligible impact on the errors. This is confirmed by taking the best second controller case (most impact at time = 1.2) and comparing it to single controller and control at every time step cases shown in Table 11.

Table 11. PI comparison from using multiple controllers with γ = 10^{-7}, time step = 0.1 seconds and total SSP time = 1.8 seconds.

Performance Index	Controller at t = 0 only	Controller at t = 0 and t = 1.2	New controller at every time step
Position	93.13	92.65	95.53
Velocity	216.86	207.00	204.31

Although some small amount of error can be reduced by adding additional controllers, it has no noticeable effect on the success of the controller to visually complete the SSP at a given γ value. This small improvement is considered to not be worth the real world increase in cost and complexity having additional controllers would add, as a much more significant improvement could be achieved by making γ smaller.

Overall, it appears that while some small improvement in system performance can be made by adding more controllers and some more significant improvements made by slowing the duration of the SSP, neither compares to the scale of improvements made by further reduction of γ. In both SSP duration cases, a $\gamma = 10^{-7}$ provides the largest γ value that is visually very close to the ideal pose and smaller values of γ begin to have a

noticeable increase in simulation runtime. In the 1.8 second SSP duration case, while significant error improvements can be made, the tradeoff is a rather slow walking gait (a full gait cycle in Fig. 4 would take 4 seconds to complete) which may not be acceptable depending on the goals of the gait designer. With these tradeoffs in mind, the baseline case that is assumed to have the lowest level of complexity and be most applicable is the $\gamma = 10^{-7}$ case with a SSP duration of 0.9 seconds and a single optimized controller linearized to the time = 0 initial position. This will be the starting point to develop an under actuated controller.

$$\begin{bmatrix} 3167.5 & 8.7467 & 8.2534 & -15.156 & -4.9979 & 3162.5 & 0.5267 & 0.8354 & 0.864 & 0.7099 \\ 5.7133 & 3205.3 & 37.675 & -39.829 & -25.214 & 0.5443 & 3164.4 & 5.4575 & 5.4483 & 5.0556 \\ 4.7164 & 33.603 & 2890.7 & 248.45 & 231.739 & 0.4022 & 1.8362 & 3190.3 & 33.163 & 40.552 \\ 5.7452 & 63.258 & -267.01 & 3502.7 & 705.919 & 0.515 & 0.4674 & 84.320 & 3282.1 & 158.95 \\ 5.6902 & 65.92 & -275.16 & 921.15 & 4234.17 & 0.2625 & -1.4961 & 107.61 & 156.85 & 3376.5 \end{bmatrix}$$

Figure 31. Best case fully actuated controller matrix K, with $\gamma = 10^{-7}$, SSP duration of 0.9 seconds and linearized to the time = 0 initial position.

4.2 Under Actuated Biped Motion

4.2.1 Failure Modes

To model under actuated control, type I and type II conditional failure cases are defined. A type I fault represents the case where a joint becomes passive and fails by locking into a fixed position [41]. The type II fault represents the case where the joint becomes passive in a loose or free swinging and uncontrolled manner [42-45]. These two cases represent the failure modes of the actuators that make up the joints in the biped

model. If both cases can be made stable with the use of the proposed LQR controllers, then the use of time-varying control [43-45] and Markovian Jump methods [46, 47] can be employed to create a dynamically adaptable controller state matrix, switching between the control modes as needed to continue the walking gait [33].

4.2.2 Modeling Failure Modes

In the following set of tests, a single joint is made passive to represent each of the actuator failure cases. To prove the concept, the joint thought to have the least overall impact on the right SSP gait was chosen to be the left knee and will be the passive joint in all of the following tests. To simulate and test the under actuated biped model, the Fig. 8 model was changed as represented in Fig. 32.

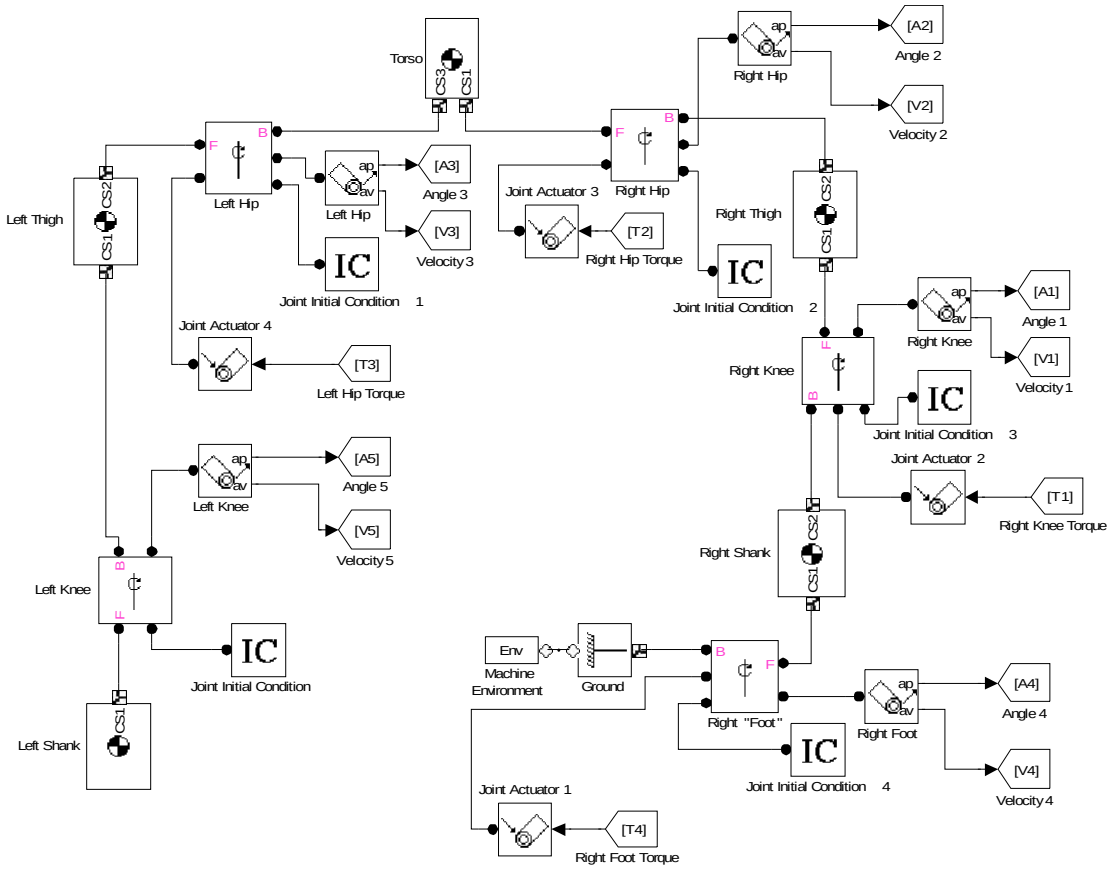


Figure 32. Block diagram of under actuated right SSP biped robot system, as modeled in SimMechanics.

As shown in Fig. 32, the under actuated Matlab simulated biped model is nearly identical to the fully actuated model. The only significant change is that the actuator for the left knee has been removed. This creates the type II case where the passive joint will swing free as the model performs the walking gait. With no actuator, the resulting controller is not able to influence the passive joint, but because the sensors remain in place, may still detect the position and velocity and calculate the associated errors.

To create the type I case, the actuator is left in but with no inputs. This has effect of locking the joint in its initial condition position relative to its linking joint, in this case the left thigh. Through testing, this was found to have near identical results to simply

creating a single link out of the left thigh and left shank also creating the type I case situation. Early tests to optimize a type I case controller quickly found the results to concur with the fully actuated results in the previous section. This showed that the locked joint, at any initial angle, had no impact on the remaining joints and the resulting fully actuated optimized controllers' ability to complete the walking gait, as long as the locked joint links didn't interact with the ground plane. In essence using LQR in the presence of a type I locked joint, with a fully actuated controller optimized as described in the previous section, can successfully perform the ideal walking gait under similar conditions.

Early tests of the type II case were not as successful. The first attempts to utilize a fully actuated controller in this case often resulted in uncontrollable results regardless of the γ used and the best case visual results were similar to the Fig. 20 failed test. Optimizing a fully actuated controller for use with the type II case simply couldn't account for the changes in the model dynamics accumulated over the course of the waking gait and as a result the controller would incorrectly compensate, causing the entire system to fail. The rest of this section explores the additional tests for the type II failure case using an optimized under actuated controller.

4.2.3 Initial Tests

Utilizing the partition and substitution in (23)-(25) to convert the fully actuated dynamics model in Matlab, an under actuated model was developed. To develop and test

under actuated optimized controllers, tests similar to those used in the previous sections were used. First, the proper use of γ was determined.

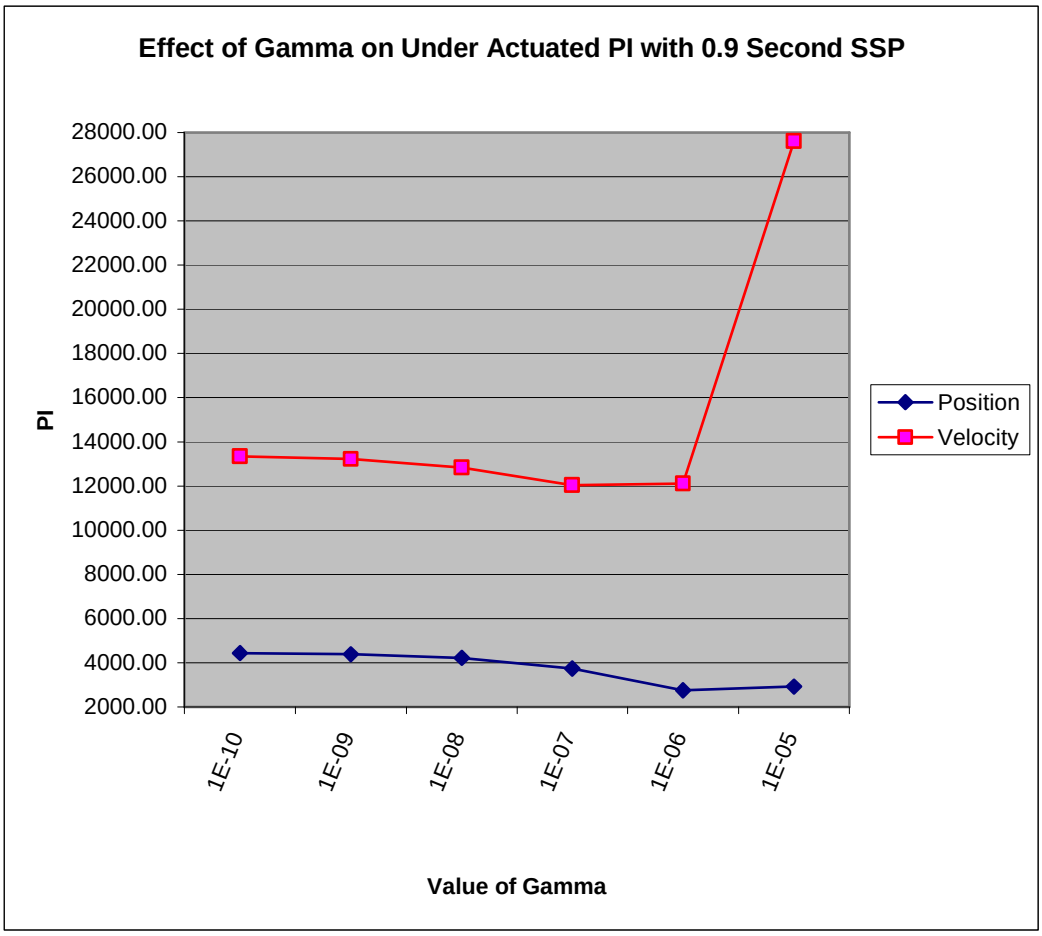


Figure 33. Effect of γ on the PI with total SSP time = 0.9 seconds and step time = 0.1 seconds for the under actuated system.

In Fig. 33 it suggests that additional errors accumulated in the under actuated model have a significant impact on the ideal use of γ. Using this information, the best case with the lowest errors would be at $\gamma = 10^{-6}$, which in the fully actuated case was not a successful performance of the SSP. This can visualized in Fig. 34.

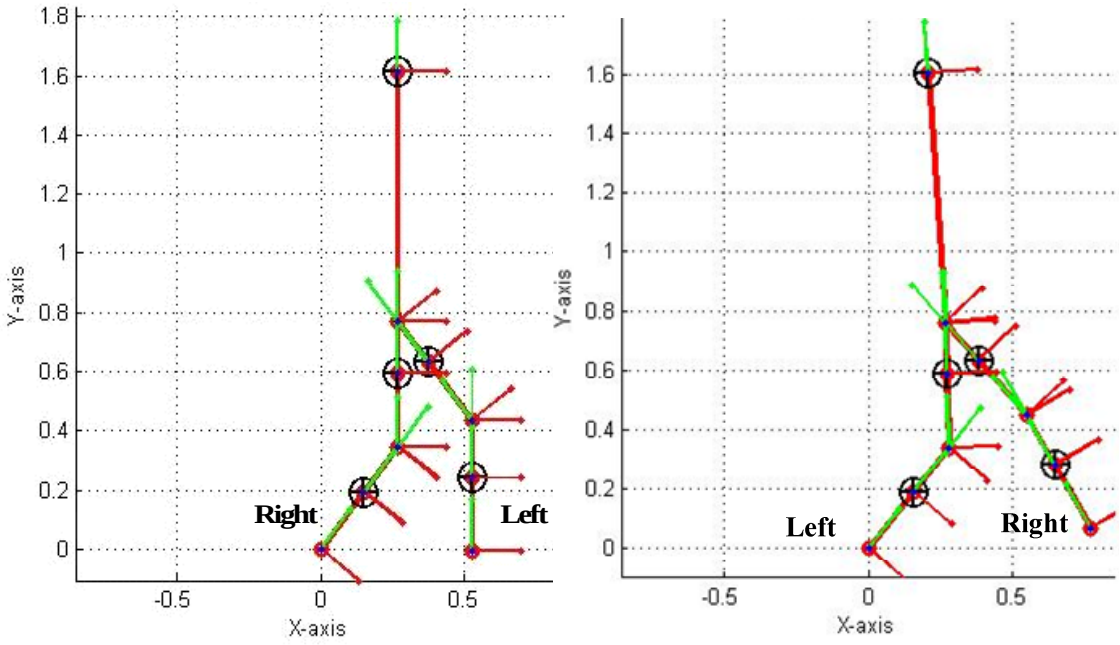


Figure 34. Final under actuated biped pose with γ = 10^{-6}, total SSP time = 0.9 seconds and step time = 0.1 seconds (Right) when compared to the ideal pose provided from the regulator (Left).

As seen in Fig. 34, the under actuated model best error rate at $\gamma = 10^{-6}$ is nearly identical to the fully actuated case shown in Fig. 21. As in the fully actuated case, the resulting final pose is poor and the simulation doesn't actually complete the SSP and is considered a failed case of the optimized under actuated controller.

To understand this similarity despite the vast difference in the PI between the fully and under actuated simulations a closer look was taken at the motion of the under actuated gait, illustrated in Fig. 35 and 36.

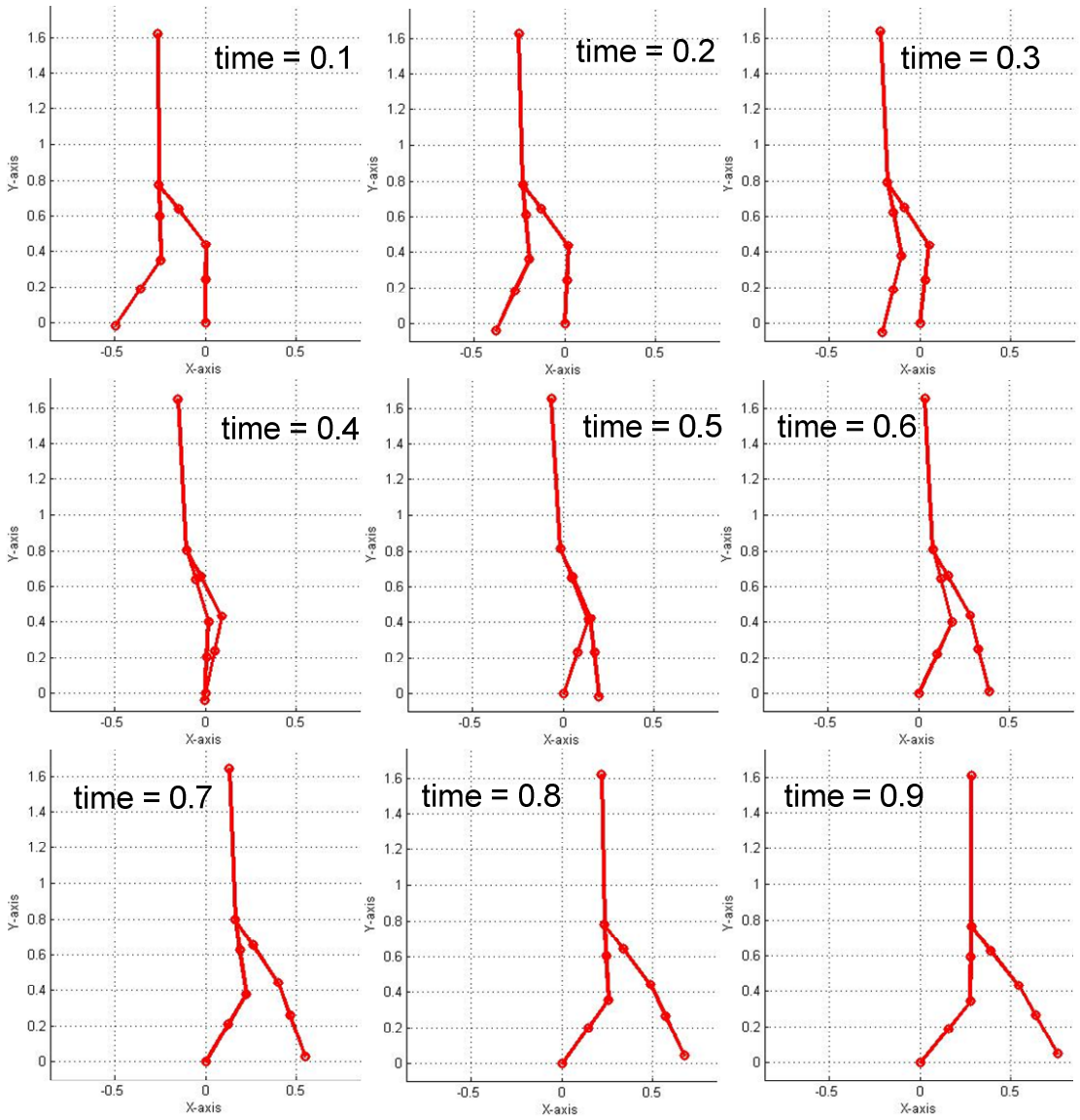


Figure 35. Under actuated motion over time (in seconds) of right leg SSP with $\gamma = 10^{-7}$.

In Fig. 35 a $\gamma = 10^{-7}$ was used to look at the under actuated walking motion as this was the most successful fully actuated case even though it disagrees with the Fig. 33 data. It can be seen that the pose at t = 0.9 is better than the $\gamma = 10^{-6}$ case with the torso fully upright, but the left leg with the uncontrolled left knee has swung the left shank link both through the ground plane and then above and past its contact point on the ground plane.

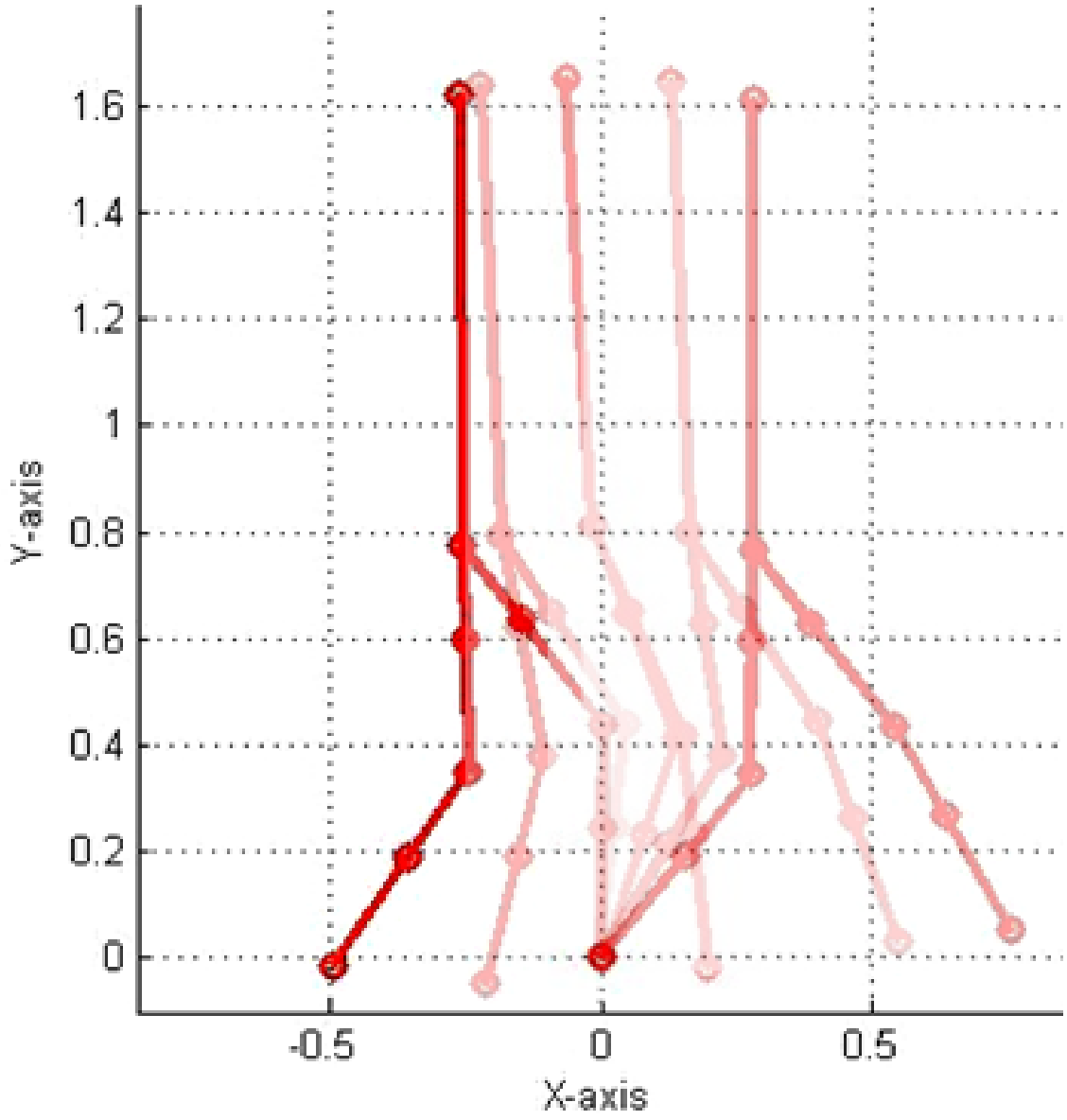


Figure 36. Illustration of Matlab under actuated walking motion of right leg SSP with $\gamma = 10^{-7}$.

Fig. 36 gives an even closer look at the $\gamma = 10^{-7}$ under actuated motion. It is clear that the left leg is swinging through the ground plane. While in the type I case (joint is locked) that has been assumed undesirable as the fixed link would be pressed against the ground in a real world system possibly disrupting the entire system, in this type II case it

is assumed acceptable. This is because with the passive joint free, in a real world system, it is assumed the link wouldn't press against the ground plane as much as be dragged along it. With this in mind it is considered acceptable to intersect with the ground plane in these simulations for the type II case as it is expected that it would have minimum impact on the control of the rest of the system.

The other item of concern is that the free joint and link (left knee/shank) continued on past its ideal point of contact on the ground plane. As with $\gamma = 10^{-6}$ in both the fully and under actuated models, this would normally be a failure of the optimized controller. In this simulation, it is much more important that the remaining controllable joints continue to function properly and reach the desired end pose despite the presence of a passive joint. This is due to the time-varying control and Markovian Jump methods, addressed at the end of this chapter [33], that directly address a way to approach this item. As such, the under actuated case with $\gamma = 10^{-7}$ visually appears to be the largest γ where the remaining controlled joints reach a position very close to the ideal.

Another point of interest is that the information in Fig. 33 appears to have little correlation with the fully actuated case and that the lowest PI case was not necessarily the best γ to use. To address this, the PI was plotted again without the passive joint errors included. This is shown in Fig. 37.

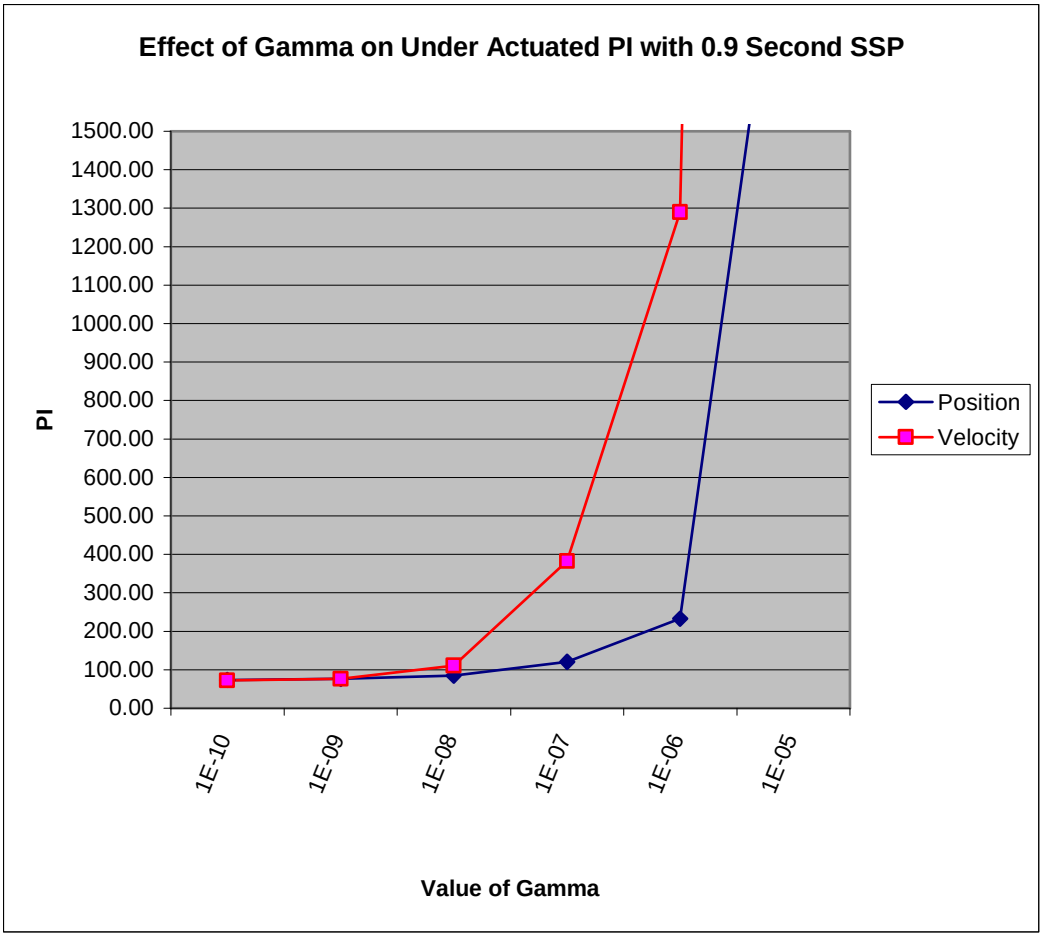


Figure 37. Using only the controllable joint errors, effect of γ on the PI with total SSP time = 0.9 seconds and step time = 0.1 seconds for the under actuated system.

With Fig. 37 it can quickly be seen that the controllable joints in the under actuated system have similar convergence behavior to the fully actuated case. This is compared numerically in Table 12.

Table 12. PI data points from Fig. 37.

Performance Index	$\gamma = 10^{-5}$	$\gamma = 10^{-6}$	$\gamma = 10^{-7}$	$\gamma = 10^{-8}$	$\gamma = 10^{-9}$	$\gamma = 10^{-10}$
Position	2340.88	232.88	120.53	84.61	75.73	73.39
Velocity	13965.49	1289.79	382.47	110.15	75.76	72.61

Comparing the Table 12 data with Table 8 for the under actuated system with the same time step and gait duration shows near identical error behavior. This suggests that the optimized under actuated controller is managing the gait for the remaining controllable joints in a similar fashion to the fully actuated case, despite the effects of the free swinging passive joint.

4.2.2 Error Observations

To look closer at where the errors are accumulating during the walking gait, and to compare with the fully actuated case, a detailed look at the errors over time is required.

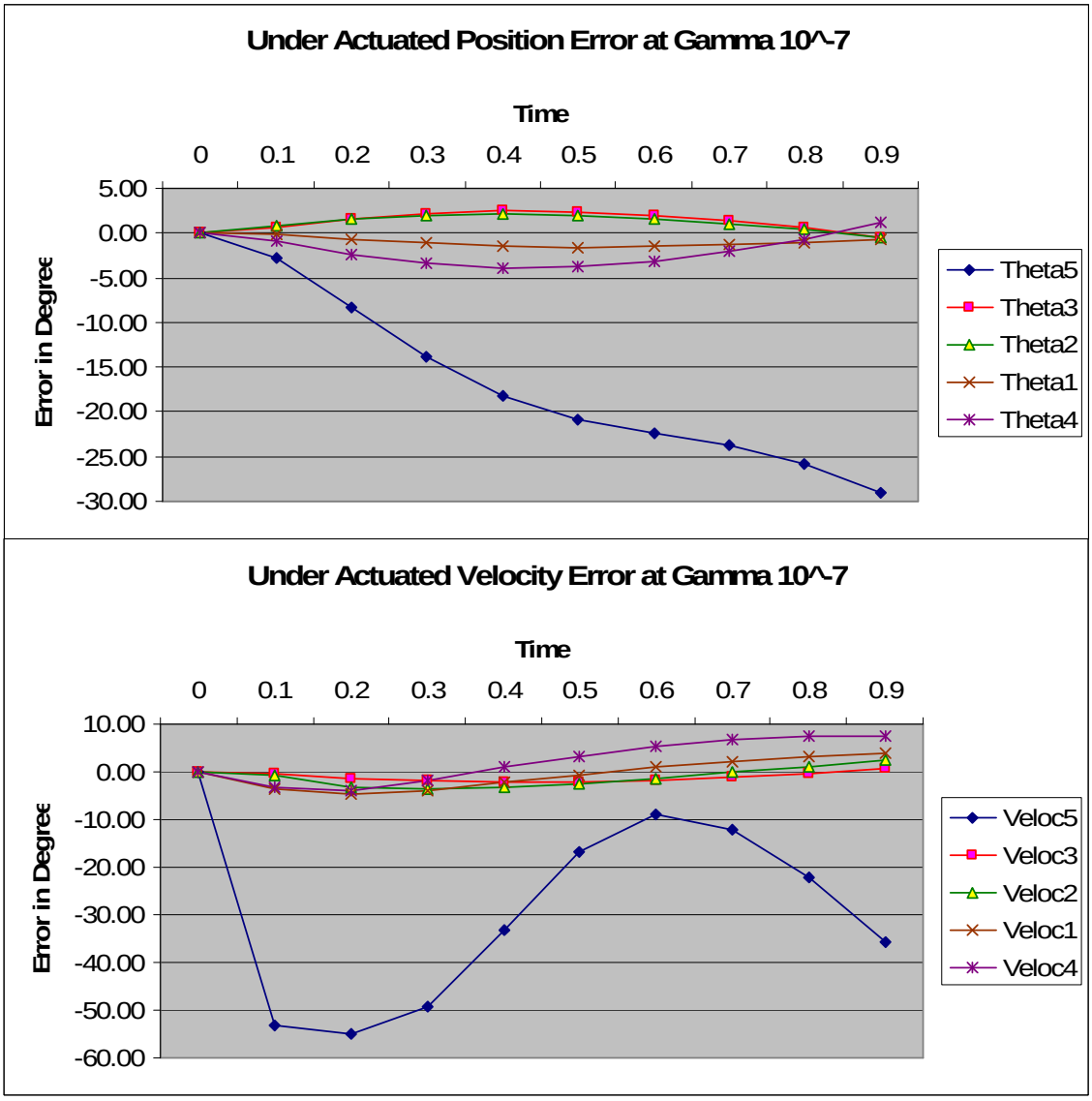


Figure 38. Position and velocity errors of each joint with γ = 10^{-7} over the course of the under actuated 0.9 second SSP.

From Fig. 38 the differences between the controlled and passive joints are clear. The passive joint is the most significant source of errors over the course of the SSP. This accounts for the very high PI numbers in Fig. 33. The passive joint also shows little if any of the convergence behavior seen in the errors of the controlled joints, clearly the optimized controller has no influence on the passive joint in this simulation. On the other

hand, the presence of the mounting passive joint errors doesn't seem to have any noticeable effect on the remaining controlled joints. From Fig. 38 it is difficult to see if the under actuated errors of the controlled joints have a close correlation with the fully actuated joints in the previous simulations. A closer look is provided in Fig. 39.

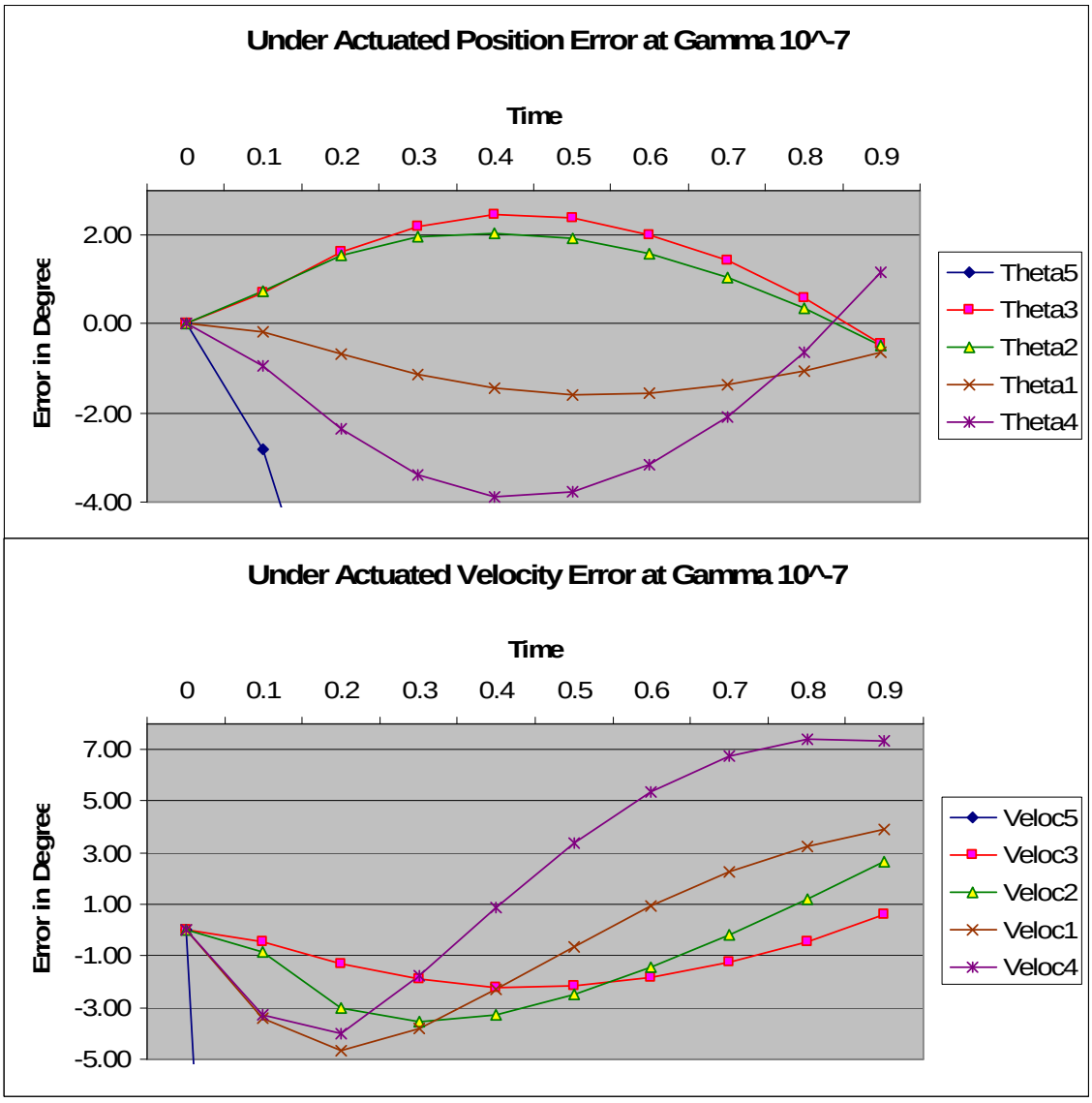


Figure 39. Position and velocity errors of the controlled joints with $\gamma = 10^{-7}$ over the course of the under actuated 0.9 second SSP.

The error information in Fig. 39 bares a strong resemblance to the fully actuated case in shown in Fig. 23 and 24. From this it can be shown that for the controlled joints in the under actuated biped, the best case $\gamma = 10^{-7}$ optimized controller can keep peak position errors below 3.9 degrees, peak velocity errors below 7.5 degrees per second, end position errors below 1.2 degrees and end velocity errors below 7.4 degrees per second. Also in similar to the fully actuated case, the worst errors experienced during the SSP are on the most burdened joints, specifically Theta 4, the right foot.

Testing the limits of the simulation, in terms of computation time, the same data was collected on the under actuated $\gamma = 10^{-10}$ case.

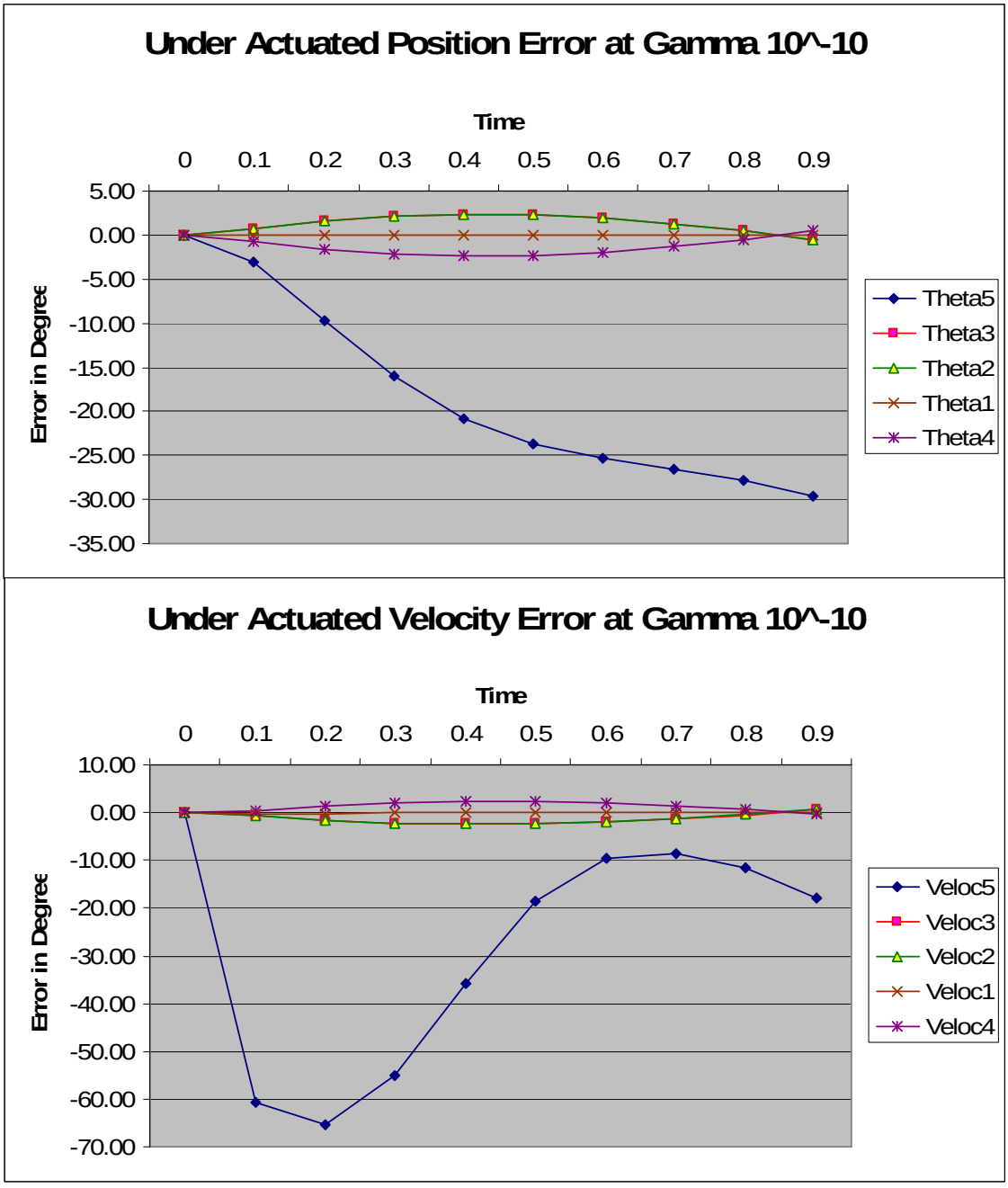


Figure 40. Position and velocity errors of each joint with γ = 10^{-10} over the course of the under actuated 0.9 second SSP.

In Fig. 40, the errors of the passive joint appear slightly larger, but overall have very similar behavior to the $\gamma = 10^{-7}$ case. This suggests that the behavior of the passive joint minimally affected by changes in γ if at all. Also similar to the fully actuated case, the

controlled joint errors are reduced compared to the larger γ cases and have a similar convergence behavior.

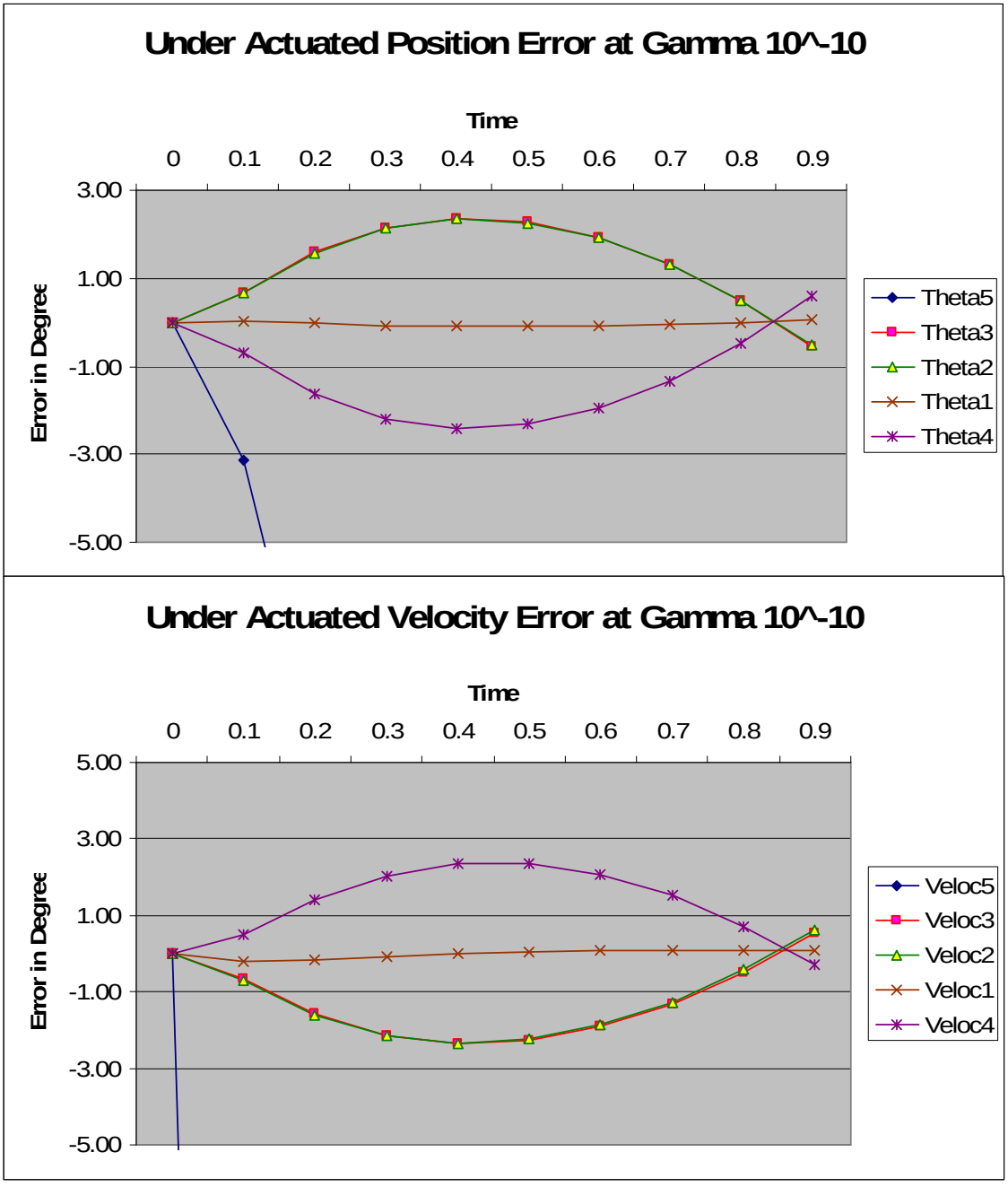


Figure 41. Position and velocity errors of the controlled joints with $\gamma = 10^{-10}$ over the course of the under actuated 0.9 second SSP.

Fig. 41 shows a closer look at the Fig. 40 data, also concurring with fully actuated data in Fig. 25. From this data, it can be shown that for the controlled joints in the under actuated biped, the $\gamma = 10^{-10}$ optimized controller can keep peak position errors below 2.5 degrees, peak velocity errors below 2.5 degrees per second, end position errors below 0.6 degrees and end velocity errors below 0.6 degrees per second.

For comparison, the best and smallest γ error cases for the fully and under actuated systems are summarized in Table 13.

Table 13. Summary of best error data points for fully and under actuated optimized controllers.

$\gamma = 10^{-7}$ **Errors(Degrees)**	Peak Position	Peak Velocity	SSP End Position	SSP End Velocity
Fully Actuated	3.88	7.38	1.14	7.29
Under Actuated	3.87	7.43	1.17	7.33
$\gamma = 10^{-10}$ **Errors(Degrees)**	Peak Position	Peak Velocity	SSP End Position	SSP End Velocity
Fully Actuated	2.41	2.35	0.58	0.6
Under Actuated	2.41	2.35	0.58	0.6

The data presented in Table 13 shows just how close the two simulations are in terms of results. Not only is the under actuated controller succeeding in preventing worse errors in the presence of a passive joint, but in this simulation and using the left knee as the passive joint, it is nearly identical to the best $\gamma = 10^{-7}$ case but it is identical in the $\gamma = 10^{-10}$ case. Overall it has been shown that by using the under actuated methods described in this paper, a successful under actuated controller can be developed for this SSP system.

$$\begin{bmatrix} 3185.8 & 19.621 & -6.0568 & -14.11 & 3163.2 & 3.5554 & 3.5021 & 3.4742 \\ 16.648 & 2875.2 & 277.72 & 240.91 & 0.9427 & 3188.6 & 31.366 & 38.928 \\ 46.647 & -286.34 & 3539.8 & 725.62 & -0.6656 & 84.5252 & 3283.1 & 161.16 \\ 58.758 & -283.05 & 936.24 & 4241.8 & -2.0881 & 107.54 & 157.13 & 3377.4 \end{bmatrix}$$

Figure 42. Best case under actuated controller matrix K, with $\gamma = 10^{-7}$, SSP duration of 0.9 seconds and linearized to the time = 0 initial position.

4.3 Markovian Jump Methods

Now that a system has been described that can successfully operate in a fully actuated situation or with both a type I and a type II actuator failure, how to develop a fault tolerant system can be approached. One method in the literature involves using a Markovian Jump to test a time-varying control scheme [33]. In essence, a Markov theory based controller coordinates switching between controller modes by representing the system as a state machine and state change probabilities [33, 46, 47]. Using Markov based methods with a system that can detect faults and apply a brake to a faulty joint is theorized to be controllable in this way [33, 43-45].

The concept can be illustrated with Fig. 43 using the 5DOF biped as an example.

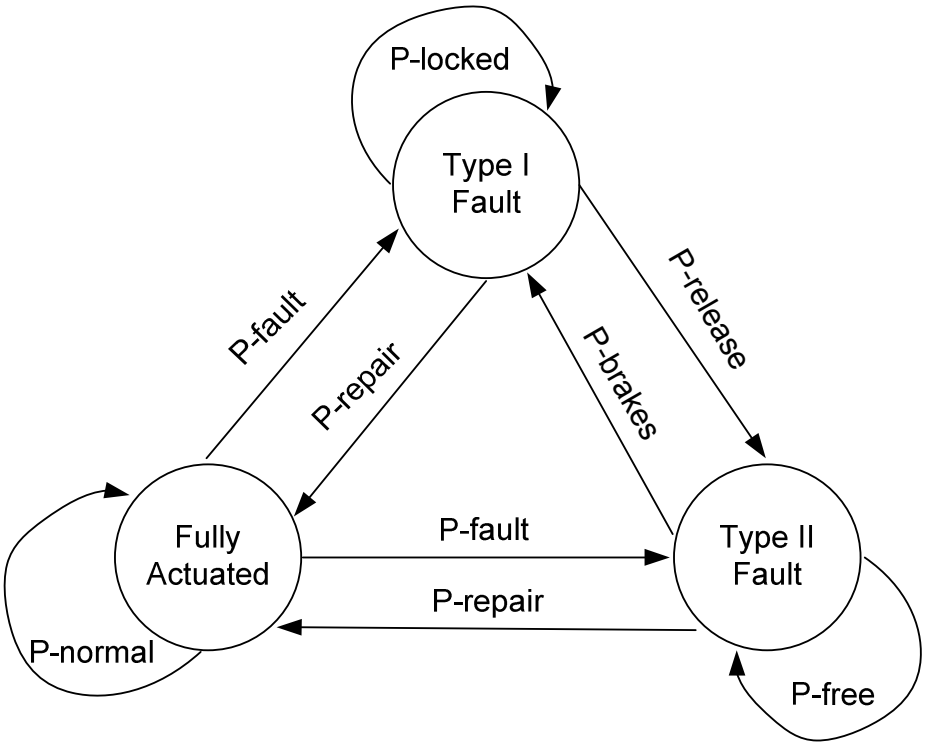


Figure 43. Rough concept model of using a Markovian Jump for control.

For this example, the biped model could be broken down into three distinct operation states, each designated as one of the circles in Fig. 43; a fully actuated controller performing the normal walking gait, a different fully actuated controller operating with one joint locked in place, either with a brake applied or from a type I fault, and a under actuated controller operating with a joint swinging free with a type II fault. To cross over to a different state requires some kind of event, and in Markov theory that event has an associated probability of happening (that is what the P- * stands for). If the fully actuated system detects a joint fault, either a lock or a free actuator failure, it can shift its control scheme to the appropriate controller. If the fault is temporary or is somehow repaired or removed, the system can shift its control scheme back to the normal fully actuated state.

If the fault is a type II, another sort of control can be utilized by applying brakes to the free joint in a controlled way. If the free joint swings into a tolerably good position, applying the brakes can lock that joint into the best position to complete the walking gait, moving it to a type I fault. Once the locked position is no longer ideal, the brakes can be released, moving the state back to a type II fault and allowing the joint to swing free into a better position where brakes could again be reapplied. In this way, even a type II fault could be controlled in a way that the walking gait could be successfully completed [43-45].

For the proposed biped model, the passive knee joint could be kept loose at the beginning of the SSP, allowing the shank to be dragged across the ground plane. The shank would then be swung forward and as it neared its ideal foot placement angel, could be locked to provide the needed support for the DSP and following SSP. By repeating this process, a relatively stable walking gait could possibility be performed, visually it may even resemble a dragging limp that injured bipeds display.

Use of Markovian methods could very well be utilized to provide a control architecture linking the three controllers developed for the proposed 5DOF biped. The full development and application of a Markov controller is not further explored in this effort as it has been considered out of scope for this thesis. More detail on how to develop and apply Markovian Jump methods on a related system is available in the literature [33, 43-47].

5. Conclusions

5.1 Summary of Observations

The demonstrated simulation is flexible, able to follow any input trajectory, not just a walking gait, and can simulate both the right leg SSP and the left leg SSP equally well (although separately with no model of the DSP to link the two). The simulation has also been shown to operate the proposed dynamics in a predictable and repeatable way which should allow a path to considerably more diverse experiments.

LQR control of the fully actuated 5DOF planar biped robot has been completely successful, performing the proposed waking gait with only a single controller linearized at the time = 0 home position. The fully actuated controller utilizes both position and velocity information in a single controller through the use of the γ in the cost function. Within this simulation, the sensitivity of using γ, multiple controllers at different time step increments, length of total step time (SSP time), and number of time step increments in the SSP time have been quantified. Although small improvements to the SSP errors can be made using these additional variables, none compared to the scope of improvement found adjusting γ.

The 0.9 second SSP walking gait peak performance errors were kept below 4 degrees position and 7.5 degrees per second velocity, entering the DSP with errors below 1.5 degrees position and 7.5 degrees per second velocity utilizing only the best case $\gamma = 10^{-7}$. An extreme case where $\gamma = 10^{-10}$, can better control the biped SSP errors

but at the cost of considerably longer simulation times and is considered to represent more costly equipment in a real world system. The $\gamma = 10^{-10}$ controller can keep the 0.9 second SSP walking gait peak errors were kept below 2.5 degrees position and 2.5 degrees per second velocity, entering the DSP with errors below 0.6 degrees position and 0.7 degrees per second velocity representing the best possible performance tested.

Under actuated control was successfully demonstrated performing the SSP under identical conditions as the fully actuated, despite the presence of a single uncontrolled passive joint in the simplest case. The original goals of the under actuated system to at least retain control of the biped with passive joints despite error rates were exceeded. The under actuated controller was shown capable of matching the fully actuated error levels and visual performance of the remaining controllable joints using the same γ values in the presence of a type II (free, unlocked) joint. Under actuated control dynamics were also shown to not be necessary for the control of the biped in the presence of a type I (locked) joint although a different optimized fully actuated controller must be used.

Methods involving time-varying control and Markovian Jumps to develop and test an overarching control scheme are discussed but not developed further in this thesis. Uses of such methods are thought to be able to switch between fully and under actuated controllers as necessary using joint brakes to manage the walking gait. If successful, such a control scheme could be developed and optimized to dynamically adapt to damage or faults during the operation of the biped system maintaining normal or near normal operations.

5.2 Future Work

Over the course of this simulation's development a list of items presented themselves as opportunities for additional study. The following are a description of the most prevalent ideas the author considered needing more attention to detail if more time and resources were available. It is also the hope of the author that this thesis will inspire the reader with numerous additional ideas.

The initial development of the biped's simulation environment involved needing a ground plane to push off of to perform the SSP. The constraints placed on the simulation by using SimMechanics software required that one point in the simulation space must be fixed to mechanical structure to properly perform kinematics calculations. This was an immediate benefit since that point could be considered the placement of the supporting biped foot. Unfortunately, the remainder of the ground plane is permeable to the biped model, allowing failed simulations to fall through the ground plane and the uncontrolled shank link in the simulation also was observed drifting through the ground plane, requiring additional assumptions on its true behavior to be made. It is possible to make an interactive ground plane in SimMechanics as shown used in several demos and its use would have benefited the realism of the simulations. Another off-shoot of an interactive ground plane would be to have variable terrain to walk the biped across, like stairs or a ramp, or even more dynamic surfaces like mud or shallow flowing water.

A very early decision in developing the biped model was to remove friction from the dynamics. Friction would have added another set of dynamics matrices, increasing

the overall biped complexity, and may have required that a specific type of actuator be further modeled to match the assumed friction coefficients. It is speculated that the lack of friction did lessen the model complexity but at the cost of possibly creating a more complicated controller. This was suggested by some of the test data in that some of the optimized controllers were prone to under damped behavior and that controllers that did work had to have very small values of γ. Adding friction to the model may have allowed simpler optimization of controllers and would have added further realism to the model.

The walking gait, as noted in the paper, was finally developed to be a very simple point to point, sliding gait. The literature is full of much more detailed and even algorithm based methods to develop and tailor make a walking gait for a particular system's needs. The gait used in the paper was good enough for the simulation needs, but if additional terrain, joint study, or biped model were used a completely new gait would have to developed from scratch. Using the more advanced methods in the literature would save a considerable amount of work and could also branch into a completely different type of control problem: a running gait.

The propose LQR controller did better than expected for both the fully and under actuated cases, using only a single controller for an SSP utilizing position and velocity inputs. On common way to control systems is to have separate position and velocity controllers. It is suspected that this would have allowed simpler (larger γ) controllers to be used with the same effect. It would be interesting to see which method would produce a simpler or better performing control system, or the impact on a robustness study.

Controller robustness in general was not studied in this paper. Comparisons with different controllers, multiple controllers, or other methods would be interesting to see in

a model of this complexity, especially in the under actuated cases. Only some preliminary tests (not discussed in this paper) revealed a small indication that a simple 2 joint double inverted pendulum SimMechanics model had some inherent controller robustness when exposed to a variable external disturbance.

As discussed in the last part of the results chapter, future work on this system will absolutely need to look into development of a time-varying controller utilizing Markovian Jump methods. This is the logical next step to test under actuated control in a system of this complexity. It is also a way to incorporate these controller schemes into much more complicated systems operation in a damage or failure prone environment.

The proposed and tested under actuated model showed better than expected results with the least impacting, single joint fault cases. That said, there are 4 other joints that could be points of singular faults, and especially in the supporting leg of a SSP, should be much more difficult to control cases. Also multiple joint failures and critical points during the walking gait failures could all be further studied. These would be especially interesting to look at using the time-varying Markovian developed controllers.

Very early controller studies in this paper used $H\infty$ controllers to perform the walking gait. Unfortunately, the long computation time to perform those methods prevented effective optimization of useful controllers, in some cases nearly 100 hours per iteration. Comparison between $H\infty$ and other control methods against the successes of the proposed LQR controllers would be interesting, especially in a robustness study.

Finally the ultimate goal of these efforts and the possible future work ideas would be to build and test a fully 3 dimensional simulation model and translate the results to a fully built real world physical robot model. While this would be significantly more

complicated than the efforts in this paper, the results, good or bad, could lead to highly damage/fault tolerant robotic systems.

Bibliography

[1] K. Capek, " R. U. R. (Rossum's Universal Robots)," New York, New York: Penguin Group, 2004, orig. Prague: Aventinum, 1921.

[2] The Robot Hall of Fame (n.d.) [Online]. Available: http://www.robothalloffame.org/unimate.html

[3] Industrial Robots (n.d.) [Online]. Available: http://awarobtics.edublogsorg/

[4] Robots for the Home (n.d.) [Online]. Available: http://www.iRobot.com/

[5] Robomow. It mows. You don't. (n.d.) [Online]. Available: http://www.robomow.com/robomow/

[6] Roboworx (n.d.) [Online]. Available: http://www.robots.com/

[7] St. Petersburg Times (Mar. 2, 2003) [Online]. Available: http://www.sptimes.com/2003/03/02/Floridian/Robots_to_the_rescue.shtml

[8] EOD-Robot: tEODor (n.d.) [Online]. Available: http://www.armedfroces-int.com/suppliers/telerob-gmbh.html

[9] da Vinci Surgical System (n.d.) [Online]. Available: http://www.intuitivesurgical.com/index.aspx

[10] ASIMO (n.d.) [Online]. Available: http://asimo.honda.com/

[11] Robot Suit HAL (n.d.) [Online]. Available: http://www.cyberdyne.jp/english/robotsuithal/index.html

[12] J. Beard, "DARPA's Bio-Revolution: Prosthetics," (2008) [Online]. Available: http://www.darpa.mil/Docs/Biology-biomedical_services_200807171322092.pdf

[13] BigDog (n.d.) [Online]. Available: http://www.bostondynamics.com/robot_bigdog.html

[14] Sensor Fusion (n.d.) [Online]. Available: http://www.k2.t.u-tokyo.ac.jp/fusion/index-e.html

[15] Absolute Astronomy (n.d.) [Online]. Available: http://www.absoluteastronomy.com/topics/Fly_(exercise)

[16] C. Zhu, Y. Tomizawa, X. Luo, and A. Kawamura, "Biped Walking with Variable ZMP, Frictional Constraint, and Inverted Pendulum Model," *in IEEE Proc. International Conf on Robotics and Biomimetics*., Shenyang, China, 2004, pp. 425-430.

[17] F. Silva, and J. Machado, "Position/Force Control of Biped Walking Robots," *in 2000 IEEE Int. Conf. System, Man and Cybernetics*., pp. 3288-3293.

[18] X. Mu, and Q. Wu, "Sagittal Gait Synthesis for a Five-Link Biped Robot," *Proc. of the 2004 American Control Conference*, Boston, MA, 2004, pp. 4004-4009.

[19] X. Mu, and Q. Wu, "Development of a complete dynamic model of a planar five-link biped and sliding mode control of its locomotion during the double support phase," *Int. J. Control*, May 2004, vol. 77, No. 8, 789-799.

[20] B. Ma, and Q. Wu, "Parametric study of repeatable gait for a planar five-link biped," *Robotica*, United Kingdom, Cambridge University Press, 2002, vol. 20, pp. 493-498.

[21] Q. Wu, and C. Chan, "Design of Joint Angle Profiles for a Planar Five-Link Bipedal System," ASME Trans. *Dynamic Systems, Measurement, and Control*, vol. 127, pp. 192-196, June 2005.

[22] Z. Peng, Q. Huang, X. Zhao, T. Xiao, and K. Li, "Online Trajectory Generation Based on Off-line Trajectory for Biped Humanoid," *in IEEE Proc. Int. Conf. on Robotics and Biomimetics*., Shenyang, China, 2004, pp. 752-756.

[23] F. Asano, Z-W. Luo, and M. Yamakita, "Some Extensions of Passive Walking Formula to Active Biped Robots," *Proc. 2004 IEEE Int. Conf. Robot. Automat.*, New Orleans, LA, Apr. 2004, pp. 3797-3802.

[24] C. Fu, M. Shuai, and K. Chen, "Proving Asymptotic Stability of Dynamic Walking for a Five-Link Biped Robot with Feet," *in 2006 IEEE Int. Robotics, Automation and Mechatronics*., Bangkok, June 2006.

[25] D. Sharon, and M. Panne, "Position/Force Control of Biped Walking Robots," *Proc. 2005 IEEE Int. Conf. Robot. Automat.*, Barcelona, Spain, Apr. 2005, pp. 1989-1994.

[26] J. Craig, "Intro to Robotics," 2nd ed., Prentice Hall, 1988, 2005, ch. 7.

[27] J-S. Yang, 1994, "On Trajectory Planning, Dynamics, and Control for a Five-Degree-of-Freedom Biped Locomotion System," *Proc. American Control Conference*, Baltimore, Maryland, pp. 3105-3109

[28] Mu, X., and Wu, Q., 2004, "Dynamic Modeling and Sliding Mode Control of a Five-Link Biped during the Double Support Phase," *Proc. American Control Conference*, Boston, MA, 2004, pp. 2609-2614.

[29] C. Golliday, and H. Hemami, "An Approach to Analyzing Biped Locomotion Dynamics and Designing Robot Locomotion Controls," *IEEE Trans. Automatic Control*, vol. AC-22, No. 6, pp. 963-972, Dec. 1977.

[30] J-S. Yang, 1994, "A Control Study of a Kneeless Biped Locomotion System," *Journal of the Franklin Institute*, **331B**(2), pp. 125-143.

[31] P. Leva, 1996, "Adjustments to Zatsiorsky-Seluyanov's Segment Inertia Parameters," *J. Biomechanics*, **29**(9), pp. 1223-1230

[32] M. Spong, and M. Vidyasasar, "Robot Dynamics and Control," Wiley, 1989

[33] A. Siqueira, and M. Terra, 2004, "A Nonlinear and Markovian $H\infty$ Controls of Under Actuated Manipulators," *IEEE Transactions on Control Systems Technology*, **12**(6), pp. 811-826.

[34] F. Lewis, S. Jagannathan, and A. Yesildirek, "Neural Network Control of Robots and Nonlinear Systems," Taylor and Francis, London 1998, ch. 3.

[35] A. Bryson, "Applied Linear Optimal Control, "Stanford CA: Cambridge University Press, 2002.

[36] G. Franklin, J. Powell, and A. Emami-Naeine, " Feedback Control of Dynamic Systems," Prentice Hall, Upper Saddle River, NJ, 4th ed., 2002.

[37] J. Vegte, "Feedback Control Systems," Prentice Hall, NJ, 3rd ed., 1994.

[38] K. Zhou, J. Doyle, and K. Glover, "Robust and Optimal Control," Prentice Hall, NJ, 1996.

[39] A. Siqueira, and M Terra, "Nonlinear $H\infty$ Control Applied to Biped Robots," *Proc. 2006 IEEE Int. Conf. Control Applications*, Munich, Germany, Oct. 2006, pp. 2190-2195.

[40] T. Komura, H. Leung, S. Kudoh, and J. Kuffner, "A Feedback Controller for Biped Humanoids that Can Counteract Large Perturbations During Gait," *Proc. 2005 IEEE Int. Conf. Robot. Automat.*, Barcelona, Spain, Apr. 2005, pp. 1989-1994.

[41] J-M. Yang, "Fault-Tolerant Gait Generation for Locked Joint Failures," *in 2003 IEEE Int. Conf. System, Man and Cybernetics*., pp. 2237-2242.

[42] A. Luca, R. Mattone, and G. Oriolo, "Stabilization of Under Actuated Robots: Theory and Experiments for a Planar 2R Manipulator," *Proc. 1997 IEEE Int. Conf. Robotics and Automation*, Albuquerque, NM, 1997, pp. 3274-3280.

[43] H. Arai and S. Tachi, “Position Control of a Manipulator with Passive Joints Using Dynamic Coupling,” *IEEE Trans. Robot. Automat*., vol. 7, pp. 528-534, Aug. 1991.

[44] H. Arai, K. Tanie, and S. Tachi, “Dynamic Control of a Manipulator with Passive Joints in Operational Space,” *IEEE Trans. Robot. Automat*., vol. 9, pp. 85-93, Feb 1993.

[45] H. Arai, K. Tanie, and N. Shiroma, “Time-scaling Control of an Under Actuated Manipulator,” *Proc. 1998 IEEE Int. Conf. Robotics and Automation*, Leuven, Belgium, 1998, pp. 2619-2626.

[46] O. Costa, and J. Val, “Full Information $H\infty$ -Control for Discrete-Time Infinite Markov Jump Parameter Systems.” *J. Math. Anal. Appl.*, vol. 202, No. 0335, pp. 578-603, 1996.

[47] O. Costa, E. Filho, E. Boukas, and R. Marques, “Constrained quadratic state feedback control of discrete-time Markovian jump linear systems,” *Automatica*, vol. 35, pp. 617-626, 1999.