Dynamic Modeling of Human Gait Using a Model Predictive Control Approach

Jinming Sun
Marquette University

DYNAMIC MODELING OF HUMAN GAIT USING A MODEL PREDICTIVE CONTROL APPROACH

by

Jinming Sun, B.S., M.S.

A Dissertation Submitted to the Faculty of the Graduate School,
Marquette University,
in Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy

Milwaukee, Wisconsin

May 2015

ABSTRACT

DYNAMIC MODELING OF HUMAN GAIT USING A MODEL PREDICTIVE CONTROL APPROACH

Jinming Sun, B.S., M.S.

Marquette University, 2015

This dissertation aims to develop a dynamic model of human gait, especially the working principle of the central nervous system (CNS), using a novel predictive approach. Based on daily experience, it should be straightforward to understand the CNS controls human gait based on predictive control. However, a thorough human gait model using the predictive approach have not yet been explored. This dissertation aims to fill this gap. The development of such a predictive model can assist the developing of lower limb prostheses and orthoses which typically follows a trial and error approach. With the development of the predictive model, lower limb prostheses might be virtually tested so that their performance can be predicted qualitatively, future cost can be reduced, and the risks can be minimized.

The model developed in this dissertation includes two parts: a plant model which represents the forward dynamics of human gait and a controller which represents the CNS. The plant model is a seven-segment six-joint model which has nine degrees of freedom. The plant model is validated using data collected from able-bodied human subjects. The experimental moment profile of each joint is input to the model; the kinematic output of the model is consistent with the experimental kinematics which verifies the fidelity of the plant model.

The developed predictive human gait model is first validated by simulating able-bodied human gait. The simulation results show that the controller is able to simulate the kinematic output close to experimental data. The developed model was then validated by simulating variable speed able-bodied human gait. The simulation results showed the dynamic characteristics of variable speed gait could be qualitatively predicted by the developed model. Finally the gait of a unilateral transtibial amputee wearing passive prosthetic ankle joint is simulated to verify its ability to qualitatively predict the dynamic characteristics of pathological gait. This dissertation opens the door for modeling human gait from predictive control perspective. With the development of such a model, future prosthetic and orthotic designers can greatly reduce cost, avoid risk, and save time by using the virtual design and testing of prostheses and orthoses.

For my beloved family and country

ACKNOWLEDGEMENTS

Jinming Sun, B.S., M.S.

First of all, I would like to say I am so lucky to have Dr. Voglewede as my dissertation advisor, who also happens to be the best professor I have ever met. His passion for teaching, rigor for research, and enthusiasm for life deeply affects me. However, the most important lesson he ever teaches me is how to respect every person and appreciate every person is unique. Thank you, Phil, for your patience with a somewhat stubborn student. I would like to thank Dr. Kevin Craig, Dr. M. Barbara Silver-Thorn, Dr. Ronald Brown, and Dr. Schimmels for serving on my committee and providing invaluable guidance and suggestions. I feel honored to have them on my committee.

I would like to thank all the guys I have worked with in Wede Lab. I would like to thank Brian Korves, Joe Prisco, Michael Boyarsky, Bryan Bergelin, and JAZ for all the brainstorming sessions we had, all the classes we took together, and the numerous discussions about each other's research. My special thanks go to Brian Slaboch for being such a true friend for so many years. I miss all those good old days when we were having "Song of the Day" in the lab.

I would not survive this long journey without my friends along the way. I would like to thank Tao Yan and his family for offering me so much help when I first came to the States. I would like to thank Jiangbiao He for being such a great friend and teaching me so much about electrical engineering sometimes even late at night.

I would like to thank my family for their long and unconditional love all along the way. I would like to give my special thanks to my mom, Shan Jin, who gave life back to me when I screwed up. I would like to thank my grandpa, Shouhe Sun, for all the support and encouragement he gives me, and my uncle, Fang Sun, who always tells me to do the right thing at the right time.

Finally I would like to wish good luck to Liverpool football club, which is almost like a religion to me. Thank you for all the memorable nights you gave me both in Premier League and European football. Those unforgettable moments gave me a lot of mental support during this long journey. Hope you will end your league title droughts sooner rather than later.

You will never walk alone!

TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . . **i**

TABLE OF CONTENTS . . . . **ii**

LIST OF FIGURES . . . . **vi**

LIST OF TABLES . . . . **xi**

CHAPTER 1 Introduction **1**

1.1 Motivation and Problem Statement 1

1.2 Literature Review 3

1.2.1 Inverted Pendulum Model 4

1.2.2 Passive Dynamic Walker 5

1.2.3 Zero-Moment-Point Method 6

1.2.4 Optimization-Based Method 8

1.2.5 Control Based Methods 11

1.3 Overview of Dissertation 13

CHAPTER 2 Plant Model Development **15**

2.1 Structure of the Plant Model 16

2.2 Parameter Calculation and Optimization 19

2.3 Open Loop Simulation 22

2.4 Generality of the Open Loop Model 24

CHAPTER 3 Model Predictive Control Approach to Human Gait Modeling **26**

3.1 General Concept of MPC 26

3.2 Critical Aspects of MPC 29

3.2.1 Internal Model of MPC 29

3.2.2 Objective Function 30

3.2.3 Constraints 31

3.3 MPC Strategy in Human Gait Study 32

TABLE OF CONTENTS — *Continued*

3.3.1 Linear or Nonlinear Internal State Space Model 33

3.3.2 End-Point OR Continuous MPC Control 34

3.3.3 PID or MPC for HAT Orientation Control and Stance Knee Orientation Control During Single Support Phase 36

3.4 Summary 37

CHAPTER 4 Development of the Internal MPC Model 38

4.1 Guidelines for the Internal MPC Model 38

4.1.1 Simplicity is Critical 39

4.1.2 Single Support and Double Support Phase Should be Simulated Separately 39

4.1.3 Not Every Joint Moment Is Required 40

4.2 Internal MPC Model for Single Support Phase 40

4.3 Internal MPC Model for Double Support Phase 45

4.4 Conclusion 49

CHAPTER 5 MPC Control System 50

5.1 Overall Control Algorithms 50

5.2 MPC Related Parameters - Prediction Horizon, Control Horizon, and Constraints 52

5.3 Objective Function 54

5.4 Laguerre Functions as Control Inputs 55

5.4.1 Laguerre Functions 56

5.4.2 Application to Joint Moments 59

5.5 Auxiliary PID Control 60

5.6 Platform to Realize the MPC Control System 61

5.7 Summary 62

CHAPTER 6 Simulation of Able-Bodied Human Gait 63

6.1 Method of Able-Bodied Human Gait Simulation at SSWS 63

6.2 Simulation Results of SSWS Gait 64

6.3 Discussion of Simulation Results of SSWS Gait 67

TABLE OF CONTENTS — *Continued*

6.4 Summary 72

CHAPTER 7 Simulation of Variable Speed and Pathological Gait 73

7.1 Simulation of Fast and Slow Walking for Able-Bodied Individuals . . 73

7.1.1 Method 73

7.1.2 Literature 74

7.1.3 Simulation Results 75

7.1.4 Discussion 76

7.2 Simulation of Amputee Gait 81

7.2.1 Method and Literature 81

7.2.2 Simulation Results 83

7.2.3 Discussion 84

CHAPTER 8 Conclusion and Future Work 86

8.1 Contributions 86

8.2 Model Limitations 87

8.3 Future Work 88

8.4 Final Remarks 90

REFERENCES 91

APPENDIX A The Kinematic Results of the Open Loop Human Gait Model Simulation 98

APPENDIX B Kinematic Results of the Open Loop Human Gait Model for Three Other Subjects 104

B.1 Human Subject 2 104

B.2 Human Subject 3 111

B.3 Human Subject 4 118

APPENDIX C Simulink Forward Dynamics Plant Model 124

APPENDIX D Control System MATLAB Code 140

D.1 Control Program for the Single Support Phase Simulation at SSWS . 140

D.2 Optimization Code for the Single Support Phase Simulation at SSWS 150

D.3 Control Program for the Double Support Phase Simulation at SSWS 150

TABLE OF CONTENTS — *Continued*

D.4 Optimization Code for the Double Support Phase Simulation at SSWS 161

LIST OF FIGURES

1.1 Control-Oriented Gait Dynamic Model 2
1.2 Classification Chart of the Directions of Human Gait Research 4
1.3 Inverted Pendulum Model 4
1.4 A Simple Model of Passive Dynamic Walker [1] 6
1.5 Active Force/Moment Balanced by Inertia Force/Moment at ZMP Point 7
1.6 General Block Diagram of MPC Applied to Human Gait Analysis 12
2.1 Seven-Link and Six-Joint Gait Model 16
2.2 Model of the Joints 17
2.3 Model of the Ground Reaction Force 18
2.4 The Optimization Algorithm to Obtain the Internal Mechanical Parameters 20
3.1 Typical Block Diagram of Control Method Based on Past Error 26
3.2 The CNS Predicts and Make Adjustment in Advance to Avoid Possible Failure 27
3.3 Block Diagram of Model Predictive Control 27
3.4 Trade-Off Between Linear and Nonlinear Internal Model 34
3.5 Proposed Control Strategy of CNS 35
4.1 Nonlinear Internal MPC Model for Both Phases 38
4.2 Internal MPC Model for Single Support Phase 41
4.3 Anthropometric Parameters of the Internal MPC Model for Single Support Phase 42
4.4 Internal MPC Model for Double Support Phase 45
4.5 Anthropometric Parameters of the Internal MPC Model for Double Support Phase 46
5.1 Control Algorithm of the Entire System 51
5.2 Laguerre Functions With $a = 0.5$ 58
5.3 Laguerre Functions With $a = 0.8$ 58
5.4 Laguerre Functions Approximation With $M = 4$ and $a = 0.8$ 59

LIST OF FIGURES — *Continued*

5.5 HAT PID Control Block Diagram 61

5.6 Stance Knee PID Control for Single Support Phase Block Diagram . . . 61

6.1 Sagittal Plane Ankle Angle of Stance Leg - Simulation vs Experimental Data 65

6.2 Sagittal Plane Knee Angle of Stance Leg - Simulation vs Experimental Data 65

6.3 Sagittal Plane Hip Angle of Stance Leg - Simulation vs Experimental Data 66

6.4 Sagittal Plane Ankle Angle of Swing Leg - Simulation vs Experimental Data 66

6.5 Sagittal Plane Knee Angle of Swing Leg - Simulation vs Experimental Data 67

6.6 Sagittal Plane Hip Angle of Swing Leg - Simulation vs Experimental Data 67

6.7 Moment of Stance Ankle - Simulation vs Experimental Data 68

6.8 Moment of Stance Knee - Simulation vs Experimental Data 69

6.9 Moment of Stance Hip - Simulation vs Experimental Data 69

6.10 Moment of Swing Ankle - Simulation vs Experimental Data 70

6.11 Moment of Swing Knee - Simulation vs Experimental Data 70

6.12 Moment of Swing Hip - Simulation vs Experimental Data 71

7.1 Angular Position of Stance Ankle - Fast, Slow, and Self-Selected Speed Simulation 76

7.2 Angular Position of Stance Knee - Fast, Slow, and Self-Selected Speed Simulation 77

7.3 Angular Position of Stance Hip - Fast, Slow, and Self-Selected Speed Simulation 77

7.4 Angular Position of Swing Ankle - Fast, Slow, and Self-Selected Speed Simulation 78

7.5 Angular Position of Swing Knee - Fast, Slow, and Self-Selected Speed Simulation 78

7.6 Angular Position of Swing Hip - Fast, Slow, and Self-Selected Speed Simulation 79

7.7 Moment of Stance Ankle - Fast, Slow, and Self-Selected Speed Simulation 79

7.8 Moment of Stance Knee - Fast, Slow, and Self-Selected Speed Simulation 80

7.9 Moment of Stance Hip - Fast, Slow, and Self-Selected Speed Simulation . 80

7.10 Moment of Swing Ankle - Fast, Slow, and Self-Selected Speed Simulation 81

LIST OF FIGURES — *Continued*

7.11 Moment of Swing Knee - Fast, Slow, and Self-Selected Speed Simulation 81

7.12 Moment of Swing Hip - Fast, Slow, and Self-Selected Speed Simulation . 82

A.1 Stance Ankle Single Support Phase 98

A.2 Stance Knee Single Support Phase 98

A.3 Stance Hip Single Support Phase 99

A.4 Swing Ankle Single Support Phase 99

A.5 Swing Knee Single Support Phase 100

A.6 Swing Hip Single Support Phase 100

A.7 Stance Ankle Double Support Phase 101

A.8 Stance Knee Double Support Phase 101

A.9 Stance Hip Double Stance Phase 102

A.10 Swing Ankle Double Support Phase 102

A.11 Swing Knee Double Support Phase 103

A.12 Swing Hip Double Support Phase 103

B.1 Stance Ankle Single Support Phase 104

B.2 Stance Knee Single Support Phase 105

B.3 Stance Hip Single Support Phase 105

B.4 Swing Ankle Single Support Phase 106

B.5 Swing Knee Single Support Phase 106

B.6 Swing Hip Single Support Phase 107

B.7 Stance Ankle Double Support Phase 107

B.8 Stance Knee Double Support Phase 108

B.9 Stance Hip Double Stance Phase 108

B.10 Swing Ankle Double Support Phase 109

B.11 Swing Knee Double Support Phase 109

B.12 Swing Hip Double Support Phase 110

B.13 Stance Ankle Single Support Phase 111

B.14 Stance Knee Single Support Phase 111

LIST OF FIGURES — *Continued*

B.15 Stance Hip Single Support Phase 112

B.16 Swing Ankle Single Support Phase 112

B.17 Swing Knee Single Support Phase 113

B.18 Swing Hip Single Support Phase 114

B.19 Stance Ankle Double Support Phase 114

B.20 Stance Knee Double Support Phase 115

B.21 Stance Hip Double Support Phase 115

B.22 Swing Ankle Double Support Phase 116

B.23 Swing Knee Double Support Phase 116

B.24 Swing Hip Double Support Phase 117

B.25 Stance Ankle Single Support Phase 118

B.26 Stance Knee Single Support Phase 118

B.27 Stance Hip Single Support Phase 119

B.28 Swing Ankle Single Support Phase 119

B.29 Swing Knee Single Support Phase 120

B.30 Swing Hip Single Support Phase 120

B.31 Stance Ankle Double Support Phase 121

B.32 Stance Knee Double Support Phase 121

B.33 Stance Hip Double Stance Phase 122

B.34 Swing Ankle Double Support Phase 122

B.35 Swing Knee Double Support Phase 123

B.36 Swing Hip Double Support Phase 123

C.1 Overview of the Forward Dynamics Plant Model 125

C.2 Subsystem 1 - Planar Joint 126

C.3 Subsystem 2 - Stance Foot 127

C.4 Subsystem 2.1 - Toe Ground Reaction Force of the Stance Foot 128

C.5 Subsystem 2.2 - Heel Ground Reaction Force of the Stance Foot 129

C.6 Subsystem 3 - Stance Ankle Model and Its Joint Moment Control 130

LIST OF FIGURES — *Continued*

C.7 Subsystem 4 - Model of the Shank 131
C.8 Subsystem 5 - Stance Knee Model and Its Joint Moment Actuation . . . 132
C.9 Subsystem 5.1 - Stance Knee Joint Moment PID Control 133
C.10 Subsystem 5.2 - Stance Knee Joint Model 134
C.11 Subsystem 6 - Model of the Thigh 135
C.12 Subsystem 7 - Stance Hip Model and Its Joint Moment Actuation 136
C.13 Subsystem 7.1 - Stance Hip Joint Model 137
C.14 Subsystem 7.2 - Stance Hip Joint Moment PID Control 138
C.15 Subsystem 8 - HAT Model 139

LIST OF TABLES

2.1 Optimization Algorithm 21

2.2 Minimum and Maximum Allowable Internal Spring and Damping Parameters 21

2.3 Optimized Internal Mechanical Parameters 22

2.4 Percentage Error Between the Open Loop Simulation and Experimental Kinematic Data 23

2.5 Kinematics RMSE Between the Open Loop Simulation and Experimental Data for Three Other Subjects 24

5.1 The Value of the Proportional and Derivative Gains 61

6.1 Required Model Parameters and MPC Control References 63

6.2 Comparison of Model Output and Control Reference for SSWS 65

6.3 RMSD of Angular Position of Each Joint Between Simulation and Experimental Data 68

6.4 RMSE of Moment of Each Joint Between Simulation and Experimental Data 68

7.1 MPC Control Reference for Fast and Slow Speed Simulation References . 74

7.2 Model Output Compared to Control Reference for Fast Speed Gait . . . 75

7.3 Actual Model Output Compared to Control Reference for Slow Speed Gait 75

7.4 Actual Model Output Compared to Control Reference of the Prosthetic Limb for Pure Passive Prosthesis 83

7.5 Actual Model Output Compared to Control Reference of the Intact Limb for Pure Passive Prosthesis 83

7.6 Actual Model Output Compared to Control Reference of the Passive Prosthetic Limb for Prosthesis with Torsional Spring 83

7.7 Actual Model Output Compared to Control Reference of the Intact Limb for Prosthesis with Torsional Spring 84

CHAPTER 1

Introduction

1.1 Motivation and Problem Statement

Even though walking is one of the most common behaviors which a person performs thousands of times every day, the understanding of the human gait is still quite limited. Human gait is a very complex behavior which requires delicate coordination of the central nervous system (CNS), muscles and the limbs. How the CNS controls the dynamics of the limbs to generate biped gait is still not thoroughly understood. A good dynamic model of human gait should represent the forward dynamics of human gait as well as the neurological control to be robust to the variation of environments and disturbances. This dynamic model has not been fully developed yet.

This lack of understanding in human gait may hinder the development of gait related medical devices and treatments. From the design of medical devices perspective, for example, the current design of prostheses and orthoses (P&O) is still largely based on experience intuition followed by experimental verification. Most P&O have to be fabricated and tested on human subjects before any feedback can be obtained. This trial-and-error approach is expensive and inefficient. It is highly desirable to develop a model which represents the essentials of the dynamics of human gait and the control algorithm used by the CNS. If such a model could be developed, it can facilitate the design of P&O by helping designers better understand normal and pathological gait. Furthermore, P&O can be virtually tested before being prototyped and tested on human subjects, so that their performance can be predicted, the cost can be reduced, and the risks can be minimized.

Such a biped gait model is also highly desired for medical diagnoses and treatments. It opens the door for more analysis in the causes for abnormal gait. A good forward dynamic gait model can aid in diagnosis, pre-operative planning and treatment. With this model, doctors and therapists will be able to test their

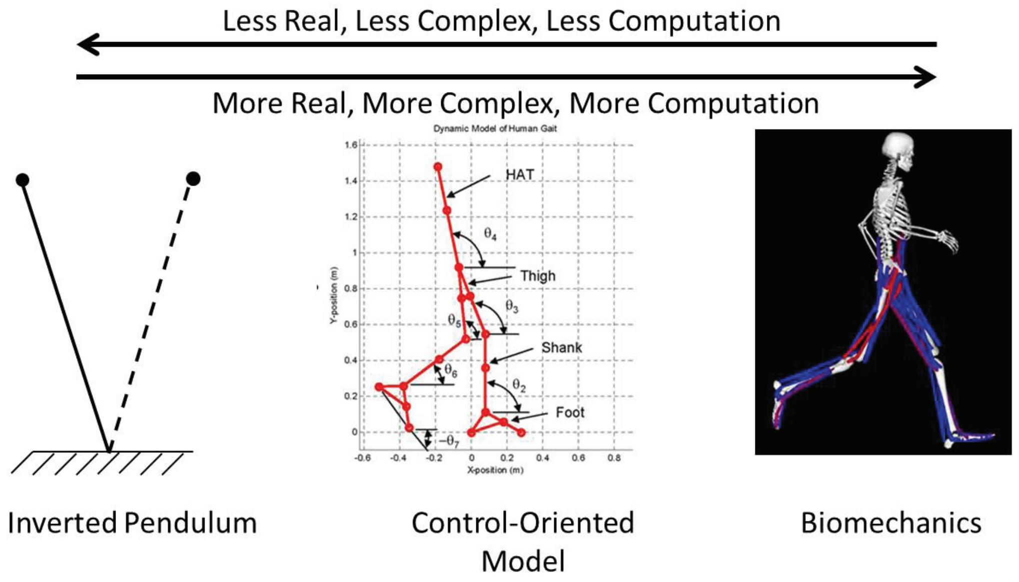

Figure 1.1: Control-Oriented Gait Dynamic Model

hypothesis without having to experiment on the patient. For example, doctors can look at how arthritis in joints or limitations in the range of motion affect the resulting gait, and then make the appropriate intervention whether it should be surgery or therapy.

As the development of an appropriate human gait model is highly desired in the design of medical devices and medical treatments, this dissertation seeks to develop a better human gait model from two perspectives: **The first objective is to build a control-oriented plant model with appropriate fidelity which represents the forward dynamics of human gait.** The complexity of this plant model should be between a high fidelity biomechanics model and a low fidelity inverted pendulum model, i.e., it should not be too complicated but still contain the essential principles of human gait (Fig. 1.1). From a simulation perspective, the plant model should also be able to be simulated in a reasonable time which should be less than one minute.

Even when a plant model is built, generation of human gait is still not guaranteed if an experimentally measured moment trajectory at each joint is input into the model. Human walking is an unstable process which is highly sensitive to input variation. Slight disturbances or variations in the input will cause the simulated human to fall. Therefore, a control algorithm is required to make the simulation of human gait possible.

Classical proportional-integral-derivative (PID) control is a widely used method both in industry and academia. This method adjusts the control input based on the feedback of the past error between the reference and the system output. However, this approach is not the only control method that will be used in this dissertation because people do not only make the adjustment based on the feedback of the past. More importantly, people look forward to predict what will happen if the current walking pattern is maintained, make the adjustment in advance so that any failure in walking will be avoided. The principles of model predictive control (MPC) are very similar to this walking strategy. **Therefore, the second objective of this dissertation is to combine classical feedback control with MPC and incorporate this control into the model to simulate the CNS, so that robust and adaptive, normal and pathological human gait can be generated.**

1.2 Literature Review

The current research of human gait can be broken into two areas: biomechanical gait analysis and biped robotics research (Fig. 1.2). The biomechanical gait analysis typically uses a musculoskeletal model which can give more details on the physiological aspect of human gait. The contribution of individual muscle, tendon and ligament to the human gait is considered in detail [2 - 7]. This type of musculoskeletal model normally has hundreds of degrees of freedom (DOF) which is overly sophisticated and distracts from the essential principles of the dynamics of human gait. In addition, the musculoskeletal model is computationally intensive and is unable to be simulated and controlled within a several days.

In the biped robotics research field, real-time control of human gait is normally the main focus and the dynamic models used are simpler than the ones used in biomechanics research. The research proposed in this dissertation falls into this category. Therefore, this review focuses on the biped robotics research literature. This field can be further divided into several subareas, where the classification chart is shown in Fig. 1.2. Xiang et al. [1] did a thorough explanation for each of the subareas. While each of these research areas has its own advantages,

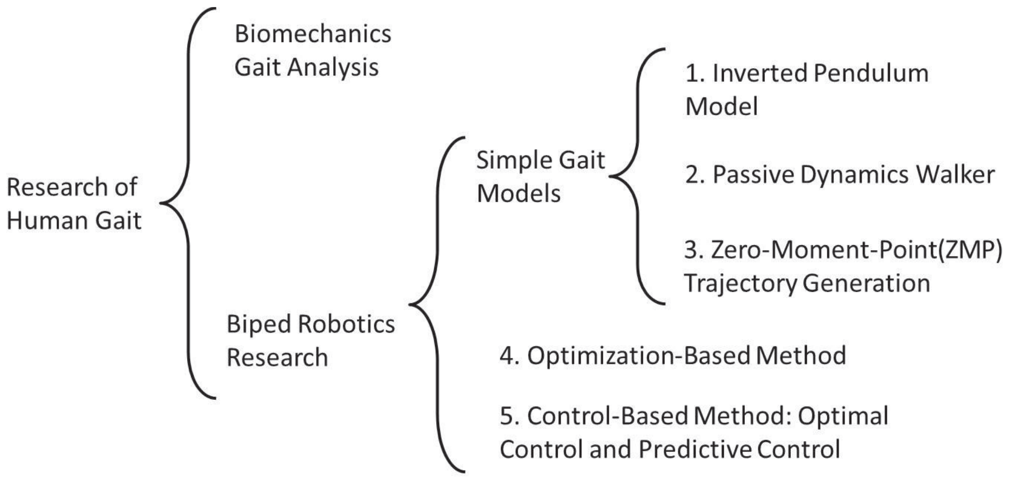
Research of Human Gait
Biomechanics Gait Analysis
Biped Robotics Research
Simple Gait Models
1. Inverted Pendulum Model
2. Passive Dynamics Walker
3. Zero-Moment-Point(ZMP) Trajectory Generation
4. Optimization-Based Method
5. Control-Based Method: Optimal Control and Predictive Control

Figure 1.2: Classification Chart of the Directions of Human Gait Research

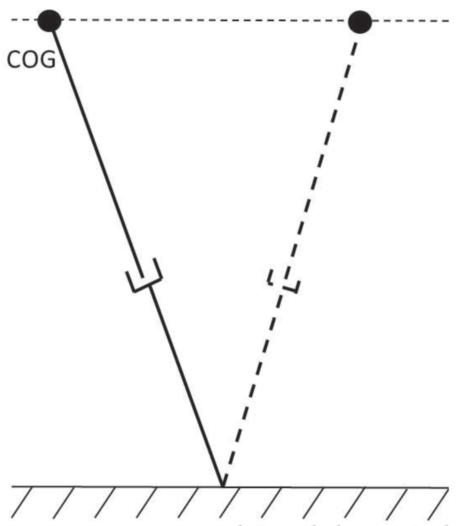

Figure 1.3: Inverted Pendulum Model

none of them has succeeded in building a human gait model which can both represent the forward dynamics principles of human walking and have a control system to make the walking simulation robust to system variation and disturbances. The following sections will review the current status of each of the subareas.

1.2.1 Inverted Pendulum Model

Walking involves energy transmission between potential energy and kinetic energy. Based on this concept, the simplest dynamics approximation is an inverted pendulum to simulate walking motion. This method uses a simple pendulum model with concentrated body mass at the center of gravity (COG). The COG trajectory along the walking direction is typically analytically derived by assuming the COG height to be fixed during the motion as shown in Fig. 1.3.

Kajita et al. were the first group to use the inverted pendulum to simulate biped gait. They used a planar inverted pendulum with a concentrated point mass and a massless leg with variable length which is similar to that illustrated in Fig. 1.3. They extended the model from the planar case to the 3D case with the same concepts [2, 3]. Kudoh and Komura [4] expanded this model by considering angular momentum around the COG. Albert and Gerth [5] further developed this method by considering the dynamics of the swing leg and proposed a two-mass inverted pendulum model and multiple-mass inverted pendulum model which represents both the stance leg and the swing leg. The latest development of this method is from Ha and Choi [6] where the height of the COG varied based on the zero-moment-point (ZMP) method. The principle of ZMP method will be explained in the following section.

The advantages of this method are the simplicity and its representation of the essential energy exchanging principles of walking. The disadvantage of this method is that the forward dynamics is over simplified, i.e., no knee joint, ankle joint and foot are modeled. Therefore, it is difficult to generate natural and realistic human gait. The passive dynamic walker is an improvement on this method in that biped gait can be generated without having to provide active power to the model.

1.2.2 Passive Dynamic Walker

The basic idea of passive dynamics walking is that a biped compass-like model can be purely driven by gravity to walk down a shallow slope without any actuation and control as shown in Fig. 1.4. The leg swings naturally as a pendulum. Conservation of angular momentum governs the transition of the swing foot with the ground and the stance leg. The most significant energy loss for this model is the impact which occurs when the swing foot contacts the ground. The energy source that compensates for this impact energy loss is the energy gained by moving down the slope.

McGeer was the pioneer in the passive dynamic walker approach. He proposed the concept and derived the governing equations in [7]. In addition, a prototype passive dynamic walker with knees was successfully built to validate the

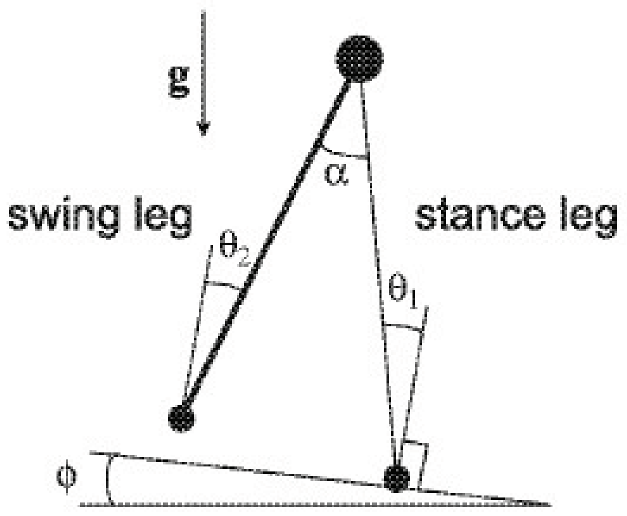

Figure 1.4: A Simple Model of Passive Dynamic Walker [1]

concept. Hurmuzlu [8] further expanded this concept to a five-link model with an upper body. The effect of the upper body on walking stability was studied. Springs and dampers were also introduced to generate additional gait patterns. Kuo [9] extended this concept from the planar case to the 3D case which allowed the model to tilt from side to side. To overcome this model's limitation that it can only walk down a slope, Collins et al. [10] added small actuators to compensate for the loss of gravity and achieve level walking. The prototype was successfully built and tested adding small amount of power at the ankle and hip joint.

The gait model proposed in this approach is simple and energy efficient and can provide some insight into the principles of human walking [11–13]. The disadvantage for this method is the same as simple inverted pendulum model; it is too simple as no knee joint, ankle joint and foot are modeled. It is difficult to rely on this model to generate natural and realistic biped gait. A more sophisticated model needs to be employed to represent the forward dynamics of human gait.

1.2.3 Zero-Moment-Point Method

The basic idea of the zero-moment-point (ZMP) method is to generate biped gait by enforcing the balance of the human body by following a set of pre-defined ZMP positions. The purpose of the control is to ensure the stability of the body rather than coordination of the entire gait. The ZMP is generally defined as a point

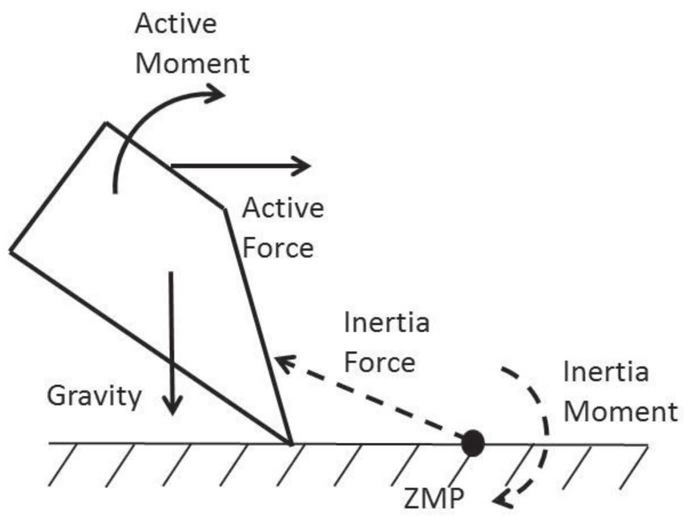

Figure 1.5: Active Force/Moment Balanced by Inertia Force/Moment at ZMP Point

on the ground where the resultant moments of the active forces should be zero, i.e., the body is dynamically balanced in the presence of active forces which include inertia, gravity and external forces from actuators but does not include the ground reaction forces. As shown in Fig. 1.5, from a dynamics perspective, all the active force and moment should be balanced by the inertial force and moment at the ZMP. The objective is to control the active forces to ensure that the ZMP is within the range of the predefined position and the center of pressure always falls within the contact surface region between the foot and the ground.

The first practical application of the ZMP method was made by Takanishi et al. [14] and Yamaguchi et al. [15], where a biped robot successfully achieved biped walking. A similar approach was also used by other researchers to develop dynamic walking robots [16–20]. Huang et al. [21] presented gait synthesis for a biped robot with 15 DOFs using the ZMP method. Both Shih [22] and Huang et al. [21] used cubic spline interpolations to generate smoother foot trajectories. Hirai et al. [23] presented the development of a Honda humanoid robot that had 26 DOFs using ZMP method to realize real-time control and Shih [24] proposed a ZMP method to generate and control the motion of a robot with 7 DOFs. Kajita et al. [25] further expanded the ZMP method by combining the inverted pendulum model with the ZMP to plan walking motion for a biped robot.

The advantage of the ZMP method is that it is computationally efficient so real-time control can be realized for biped robots. In addition, it contributes to the stability of human gait. The disadvantage of this method is that it is not inherently how humans walk as, first, the stability criteria is not human and, second, the predefined ZMP trajectory is believed to not exist in the CNS. A better approach is desired to better simulate the working principles of the CNS.

1.2.4 Optimization-Based Method

In contract to the inverted pendulum model which focuses on the dynamics of human gait and ZMP method which focuses on the stability, the optimization-based method concentrates on finding out which criteria the CNS uses to generate human gait. In general, an optimization problem is defined as:

$$Find\ x \tag{1.1}$$

$$To\ minimize\ f(x) \tag{1.2}$$

$$subject\ to\ g_i(x) \leq 0,\ and\ h_j(x) = 0 \tag{1.3}$$

where $f(x)$ is the objective function to be minimized, $g_i(x)$ are inequality constraints, and $h_j(x)$ are equality constraints. The designed variables x are typically the net moment at each joint. The objective function $f(x)$ utilized in gait analysis is normally a gait related performance measure which will be explained in the following sections. The constraints are gait related constraints such as the motion limitation and maximum possible moment at each joint. Once the optimal designed variables are obtained, they are substituted into a dynamic gait model to generate the resulting gait. The dynamic gait model is often simplified to a rigid-link model which has five or more DOFs. According to [1], the governing equations of motion (EOMs) to represent the mechanics of human gait are generally written as:

$$M(\mathbf{q})\ddot{\mathbf{q}}(t) + C(\dot{\mathbf{q}}, \mathbf{q}) + G(\mathbf{q}) = \tau(t) \tag{1.4}$$

where $\mathbf{q}$ is the joint angle profile, M is the inertia matrix, C is the Coriolis and centrifugal forces, G is the gravity force and external force, τ is the joint moments, and t is the time.

Depending on how one approaches Eqn. 1.4, there are two ways for gait simulation: inverse dynamics or forward dynamics. The inverse dynamics approach calculates the forces and moments from the experimental position, velocity and acceleration, i.e., the body motion [26]. These forces can then be utilized in an open loop fashion to drive the model forward. The approach is computationally efficient because the EOMs are not integrated in the solving process. However, this approach is not inherently how human walks because no feedback is provided. In reality, feedback is provided to the CNS. Therefore, people are able to adjust the net forces and moments at each joint so that specific kinematic objectives such as step length or walking velocity are achieved.

In contrast, a forward dynamics approach calculates the motion from the predefined forces and moments by integrating the left side of Eqn. 1.4 with specified initial conditions, which means this is a computational intensive method. For forward dynamics optimization, the forces are the design variables. The motion is obtained by integrating the EOMs with initial conditions. The optimal gait is determined by minimizing a human performance measure subject to certain constraints. In contrast to inverse dynamics, the advantage of this approach is that it inherently simulates how the control of human gait works.

Various performance measures have already been utilized in the optimization-based method. The most commonly used performance measures that are minimized as summarized in [1] are:

1. Dynamic effort:

$$f = \int_0^T \tau \cdot \tau dt \tag{1.5}$$

   which means the integration of all joint moments should be minimized over the total time, T.

2. Mechanical energy:

$$f = \int_0^T \tau \cdot \dot{q} dt \tag{1.6}$$

   which means the mechanical energy cost should be minimized.

3. Metabolic energy:

$$f = \int_0^T \dot{E} dt \tag{1.7}$$

which means the metabolic energy cost should be minimized. $\dot{E}$ represents the total energy the human body consumes during a certain distance of walking. It is different from Eqn. 1.6 that only part of metabolic energy is converted into mechanical energy.

4. Jerk:

$$f = \int_0^T \dot{\tau} \cdot \dot{\tau} dt \tag{1.8}$$

which means the rates of change in joint torque should be minimized.

5. Stability:

$$f = \int_0^T S dt \tag{1.9}$$

where S represents the stability quantity normally defined by ZMP method. Another definition can be the deviation of the trunk from vertical position.

The dynamic effort and mechanical energy measures are most frequently used in robotic field gait simulation [27–29]. The metabolic performance measure is normally used in biomechanical gait analysis [30, 31]. In reality, human gait may be governed by multiple performance measures functioning together. Some researchers conducted studies into the optimal combination of objective functions which are reviewed thoroughly in [32].

The advantage of the optimization-based method is that it can reveal some insight of the principles of human gait by using different performance measures. In addition, this method is able to handle large DOF models, which means it can be utilized on sophisticated human gait dynamic models. The disadvantage of this method is that it is computationally intensive. Therefore, it is not suitable for cases in which the simulation has to be completed in a reasonable timeframe. In addition, the optimization-based method requires experimental data are known as *a priori*. Therefore the optimization-based method is not predictive and cannot simulate pathological gait when the experimental data are difficult to be obtain.

1.2.5 Control Based Methods

Control based methods are one step further than the methods illustrated above in simulating the human CNS. In the biped robotics research, control-based methods are used to generate biped walking for humanoid robots, in which a robot can interact with its environment, react to external disturbances and execute a task in real-time. The traditional PID control widely used in industry cannot be applied to human gait analysis because of the reason already discussed; the PID method is based on the past error between the reference and the actual feedback. During human walking, people predict what will happen in the future and make adjustments in advance [33].

Compared to the other methods, the control-based method simulates the essential principles of the CNS. It is robust and flexible, can interact with environment and handle disturbances, and can be simulated in a reasonable time frame. The disadvantage of the control-based method is that a proper controller needs to be specified to ensure the stability and robustness of the model. Hurmuzlu et al. [8] reviewed various control methods for gait simulation. Three issues related to modeling, stability and control algorithms were discussed. Katic and Vukobratovic [34] reviewed intelligent control techniques such as neural networks, fuzzy logic, genetic algorithms, and their hybrid forms of control algorithms. Westervelt et al. [35] proposed a similar hybrid-zero-dynamics (HZD) feedback control method to simulate planar biped walking. Azevedo et al. [36] proposed a nonlinear predictive controller in which the optimal trajectories were obtained for the prediction horizon by minimizing the objective function. This approach can adapt to the environment and external disturbances.

Besides the above mentioned methods, the control methods currently used for gait simulation are previously optimal control approaches. The difference between the optimization-based method and the optimal control method is that: for the optimization-based method, the cost function is minimized once and the

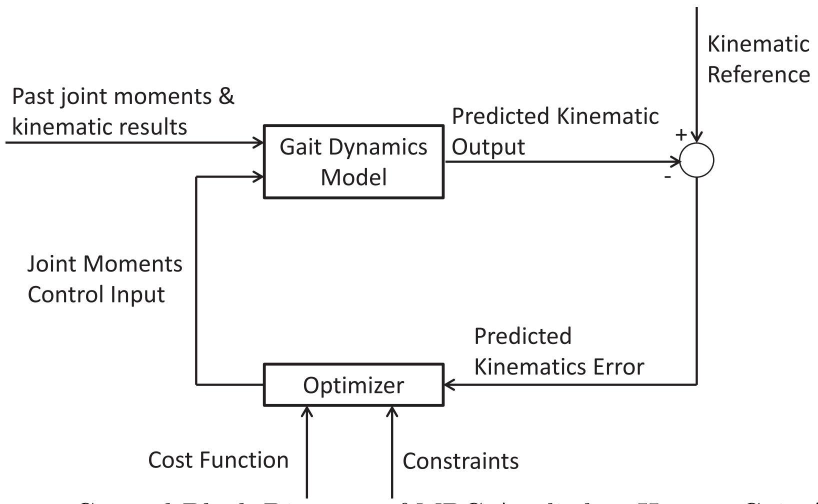
Kinematic Reference
Past joint moments & kinematic results
Gait Dynamics Model
Predicted Kinematic Output
+
-
Joint Moments Control Input
Predicted Kinematics Error
Optimizer
Cost Function
Constraints

Figure 1.6: General Block Diagram of MPC Applied to Human Gait Analysis

optimized trajectory is input to the model to get the gait. However in optimal control method, the input joint moments are unknowns in the EOMs and are continuously optimized for the next time step with the kinematic feedback provided.

One sub-area of the optimal control is called model predictive control (MPC). MPC is based on an iterative, finite horizon optimization of the motion. In this approach, the current state of the gait is discretized at time t to minimize a cost function for the optimal trajectory over a relatively short period of time in the future: $[t, t + t_N]$, where t_N represents the final time. Specifically, state trajectories are explored which emanate from the current state and find a control solution which can minimize a cost function up to time $[t + t_N]$. This optimization problem is repeated starting from the current state, yielding a new control and a new predicted state path. The futures states which are predicted keep shifting for the next time step. The general block diagram of MPC applied to human gait analysis is shown in Fig. 1.6.

Several researchers applied MPC method to simulate the CNS in human gait research. Kooij et al. [33] proposed a predictive control algorithm in which only three gait descriptors determine the nature of the gait are selected as the references: step time, step length and the velocity of the center of mass at push off. By using a seven-link eight DOF dynamics model and re-linearizing this model at each time interval, repetitive gait was reportedly generated. Ren et al. [29] utilized a similar seven-segment model as the plant with MPC as the control algorithm to simulate level walking. Different from Kooij et al. [33], the minimization of mechanical energy expenditure was employed as the major cost function. The references for the predictive control are also different, namely walking velocity, cycle period and double stance phase duration. Although repetitive walking was not generated, a complete cycle of human gait was successfully simulated. Their conclusion shows that minimizing energy expenditure should be the primary control object.

Other performance objectives have also been incorporated to improve simulation results. Gawthrop et al. [37] compared the predictive control method and the non-predictive control method, i.e., typical feedback PID control, to control a inverted pendulum. Results showed that the predictive control provides a better simulation than the traditional feedback control in that the time-delay is smaller. However, this work was not extended to full dynamic human gait model and its main concentration was on the balancing of the inverted pendulum. Karimian et al. [38] used MPC to control joint impedances of a 3D five-segment gait model. The cost function of the controller was energy consumption, vertical orientation of the body, and forward velocity of the center of mass. Results showed that the model was able to achieve level walking, stairs ascent and descent.

This literature shows that MPC should be a potential control algorithm for a human gait model. The advantage of this method is its flexibility and its simulation of the CNS. Different control objectives can be utilized and different gait dynamics can be employed to simulate the forward dynamics. Therefore, MPC will be used as the primary control algorithm of the model developed in this dissertation. However, challenges still exist in that proper control objectives need to be specified so that stable and repetitive gait can be generated. In addition, the control system must be robust and have good disturbance rejection. The solution of these challenges will be addressed in this dissertation.

1.3 Overview of Dissertation

This dissertation will follow the control-based method path and complete two objectives. **First, a forward dynamic human gait model with reasonable level of fidelity that can represent the essential principles of human walking will be developed.** This model will be used as the plant model of human gait in this dissertation. The MPC method will be used as the primary control method for the model. The hypothesis of this dissertation is that the control algorithm used in the CNS is similar to the theory of MPC. **Therefore, the second objective of this dissertation is to build a control system primarily using MPC to simulate the function of CNS, so that robust and adaptive, normal and pathological gait can be generated.** The proposed model which completes these two objectives will contribute to the understanding of human gait and aid the design of medical devices and medical treatments.

The rest of this dissertation is organized as follows. Chapter 2 explains the development of the human gait plant model and completes the first objective. Chapter 3 introduces the general concept of MPC and how it can be applied to the simulation of human gait. One important aspect of MPC is to develop an internal model for prediction purposes. Chapter 4 explains the development of the internal model. Chapter 5 combines all the elements developed in previous chapters into one human gait simulation system and explains in detail how this system works. Chapter 6 presents the simulation results of the able-bodied human gait and compares them to the experimental data. The results verify that the developed system is able to simulate human gait with appropriate fidelity within several hours. Chapter 7 presents the simulation results of the pathological gait with unilateral passive ankle and verifies that the developed model is able to qualitively predict pathological gait.

CHAPTER 2

Plant Model Development

As stated in Chap. 1, there are two major research objectives for this dissertation. The first objective is to develop a plant model with appropriate fidelity to represent the forward dynamics of human gait. The second objective is to develop and implement a control algorithm for the plant to predict able-bodied and transtibial amputee gait. This chapter will focus on the first objective.

When determining any model, the first step is to determine the level of fidelity required. In this particular research, the question becomes, how does one to determine an appropriate open loop model which can be used as a “good enough” plant to represent the dynamics of human gait. For purposes of this dissertation, it is assumed the model is sufficient when the experimental moment data of each joint is input into the plant model, it can respond with kinematic outputs that are similar to natural gait. From a controls perspective, this means the controller does not have to generate unrealistic moments to drive the plant model to achieve control objectives.

Based on this assumption, a plant model with appropriate fidelity was built and parameterized. This model is the first open loop seven link nine DOF human gait model that, given experimental moment reference input, can generate similar kinematics output as experimental results. In other words, no open loop human gait model exists in the current literature that can walk as naturally as the model developed in this work using such a simple structure.

The resulting open loop plant model will be explained in detail in the following section. First, the structure of the model will be explained. Second, the parameterization of the model is described. Finally, the model is simulated in open loop, and the outputs of the simulation are demonstrated and discussed.

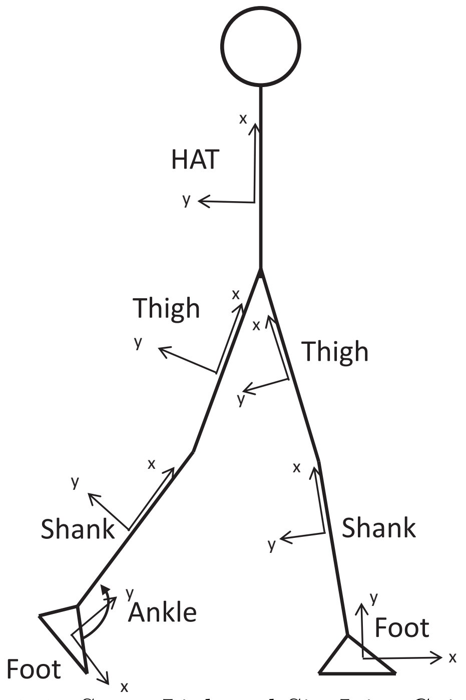

Figure 2.1: Seven-Link and Six-Joint Gait Model

2.1 Structure of the Plant Model

As shown in Fig. 2.1, the plant model developed has seven segments and nine degrees of freedom (DOF). The seven segments are feet, shanks and thighs on both sides and a single rigid body representing the head-arm-torso (HAT). The model was restricted to move only in the sagittal plane because the dynamic effects in the coronal and transverse planes are small compared with that in the sagittal plane for able-bodied gait [7]. The dynamic effect of the movement of the arms is also ignored [7].

The six joints of this model are hips, knees and ankles on both sides. All the joints are assumed to be revolute acting in the sagittal plane. As shown in Fig. 2.2, there is a rotational spring and a damper across each joint. The values of the spring stiffness, K, and damping coefficient, B, are conditionally linear with respect to the angular position of the joint. When the joint is within the range of motion, the spring stiffness and damping coefficient are constant. When the joint moves beyond the range of motion, the spring stiffness and damping coefficient increase exponentially.

The damper is used to model the viscous friction effect that physically exists

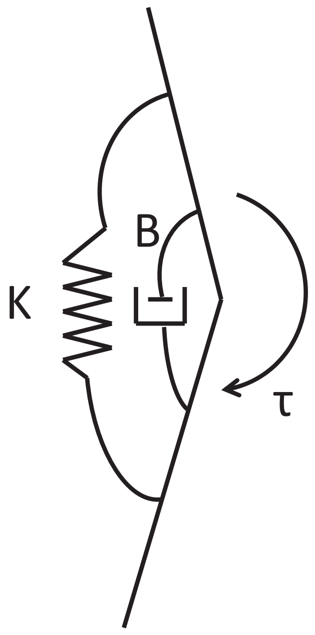

Figure 2.2: Model of the Joints

when the joint is moving. While the spring does not physically exist at each joint, a spring is added to the model to function like a passive feedback system. When the joint moves beyond the equilibrium position, which is defined as the human body standing upright, the spring pulls the joint back. Because human gait is an inherently unstable dynamic process, the existence of the spring is important in stabilizing the dynamics of human gait. This method is commonly used in modeling human gait which can be found in literature [33, 39].

There are three internal torque sources acting on each joint as shown in Fig. 2.2. One torque source is caused by the net effect of the muscles across the joint, τ. The internal spring and damper also exert internal torque on the joint. The three torque sources acting together cause the relative movement between two joints.

The model of the ground reaction force (GRF) is critical in the dynamics of human gait. This force is the only interaction the model has with the environment. This force also supports the human body and propels it forward. In this research, the GRF is modeled as two sets of springs and dampers at both heel and forefoot of each foot. One set acts horizontally and the other set acts vertically. This model is illustrated in Fig. 2.3. A spring was used because of the stiffness effect between the foot and the ground. A damper was used because of the shock absorption and energy dissipation function of the shoe, human tissue and other effects. As the GRF

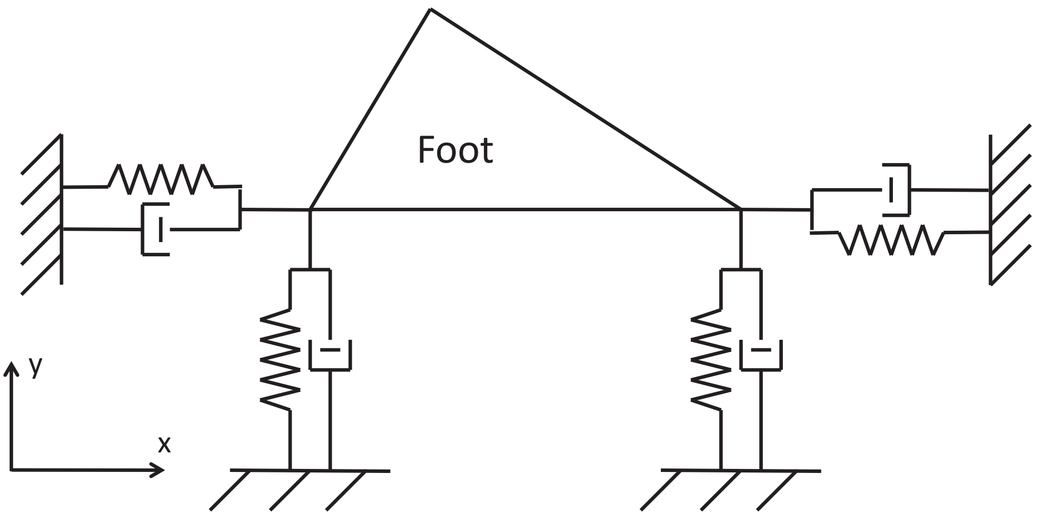

Figure 2.3: Model of the Ground Reaction Force

only acts when the foot is in contact with the ground, the GRF model must be conditional and is summarized in Eqn. 2.1 and 2.2.

$$F_y^{heel,toe} = \begin{cases} 0, & \text{if } y^{heel,toe} > 0 \\ K_y y^{heel,toe} + B_y \dot{y}^{heel,toe} & \text{if } y^{heel,toe} \leq 0 \end{cases} \tag{2.1}$$

$$F_x^{heel,toe} = \begin{cases} 0, & \text{if } y^{heel,toe} > 0 \\ K_x(x^{heel,toe} - x_0^{heel,toe}) + B_x \dot{x}^{heel,toe} & \text{if } y^{heel,toe} \leq 0 \end{cases} \tag{2.2}$$

where the x axis is defined as a space fixed coordinate system pointing from heel to toe along the sole surface, y axis is defined as perpendicular to x and pointing upward, therefore, $F_y^{heel,toe}$ and $F_x^{heel,toe}$ represent the GRF in vertical and anterior/posterior direction, K_y and K_x represent the spring stiffness in vertical and anterior/posterior direction, B_y and B_x represent the damping coefficient in vertical and anterior/posterior direction, $y^{heel,toe}$ and $x^{heel,toe}$ represent the vertical and anterior/posterior position of the heel or forefoot, $x_0^{heel,toe}$ represent the anterior/posterior position of the heel or toe when the foot has initial contact with the ground.

After the main structure of the model is determined, the parameters of the model need to be found. The anthropometry and internal mechanical parameters such as spring and damping values need to be determined. The next section will explain how these parameters are calculated or optimized.

2.2 Parameter Calculation and Optimization

The parameters that need to be determined can be categorized into two groups. The first is anthropometric parameters and the second is internal mechanical parameters which are the spring and damping values for each joint and GRF. The anthropometric parameters can be further divided into segment length, mass, mass moment of inertia and the position of the center of the mass.

The anthropometric parameter values were either obtained directly from human subject testing or calculated using the equations from [40]. A total of four able-bodied human subjects testing were performed in the Gait Lab at Medical College of Wisconsin. All of the human subjects were male with an average body mass of 86.8 kilograms and average height of 1.84 m. For each of the subjects, the data of 10 successful trials were collected. The open-loop plant model shown in this dissertation is parameterized according to one of the subjects whose body mass is 86.2 kilograms and height is 1.90 m and the data is averaged between the 10 successful trials. The experimental kinematic and kinetic data were obtained and used as the benchmark data in this dissertation. The segment lengths were directly measured. The segment mass cannot be measured directly. However, [40] provided the ratio of segments' mass to the whole body mass. Therefore, the segments mass can be calculated using Eqn. 2.3:

$$M_{segment} = \mu_{segment} M_{whole\_body} \tag{2.3}$$

where $M_{segment}$ is the mass of each of the segments, μ is the ratio provided by [40], and M_{whole_body} is the total mass of human body. Similarly, [40] provided the ratio of center of the mass to the segment length, $f_{segment}$. Therefore, the center of the mass can be calculated as:

$$y_c = f_{segment} L_{segment} \tag{2.4}$$

where f represents the ratio which is provided by [40]. Using the radius of gyration parameter per length, $\Re_{segment}$, provided by [40], the mass moment of inertia of each segment with respect to the center of the mass on sagittal plane can be calculated using Eqn. 2.5:

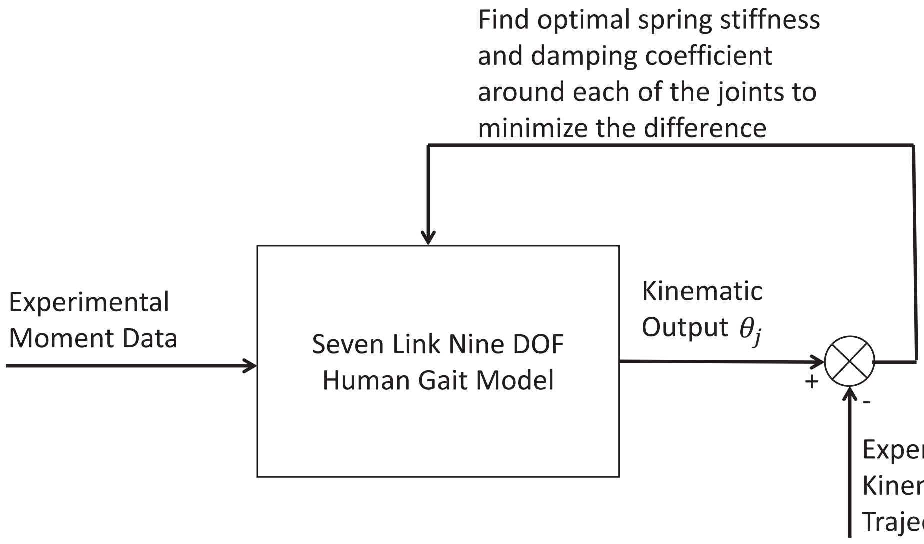
Find optimal spring stiffness and damping coefficient around each of the joints to minimize the difference
Experimental Moment Data
Seven Link Nine DOF Human Gait Model
Kinematic Output θ_j
+
-
Experimental Kinematics Trajectory

Figure 2.4: The Optimization Algorithm to Obtain the Internal Mechanical Parameters

$$I_{segment} = M_{segment}(\Re_{segment} L_{segment})^2 \tag{2.5}$$

After the anthropometry parameters are obtained or calculated, the internal mechanical parameters need to be determined. However, there is no equation or data in the literature can be directly used to obtain the internal mechanical parameters. Therefore, to obtain valid internal mechanical parameters, an optimization methods are utilized. The algorithm of the optimization is illustrated in Fig. 2.4 and the summary of the optimization procedure is listed in Tab. 2.1.

The experimental moment data at each joint are the input into the plant model. The design variables are the spring stiffness and damping coefficient for each joint and also the GRF. The cost function is the summation of the squared error between the experimental kinematic trajectory and the kinematic output of the model which is shown in Eqn. 2.6.

$$\min e = \sum_{j=1}^{6} w_j \left[\sum_{k=t_0}^{t_f} (\theta_{jk} - \theta_{rjk})^2 \right] \tag{2.6}$$

where j represent each of the joints, w_j is a weighting factor, θ_{jk} is the kinematic output of the model at time instant k, and θ_{rjk} is the experimental kinematics trajectory at time instant k. t_0 and t_f is the starting and stopping time of the simulation. The objective of this optimization is to obtain the optimal internal mechanical parameters so that the error between the kinematic output of the plant model and the experimental kinematic trajectory are minimal. More weighting was put on the stance leg because this is the side that bears body weight. When a control algorithm is augmented with the plant model, it requires more input effort on the stance side than the swing side to achieve any control objectives. Therefore, the kinematic output of the stance leg has more priority. This priority is achieved by giving a larger number in the weighting factor w_j. The constraints of the minimum and maximum allowable spring and damping parameters are listed in Tab. 2.2. The values of these constraints are determined to ensure the optimized parameters are inside a physically realistic range.

Table 2.1: Optimization Algorithm

Optimization Algorithm
Model: Seven segments six joints, and nine DOFs human gait model
Input: M_j
Output: θ_j
Design variables: K_j, D_j, $K_{GRF,V}$, $D_{GRF,v}$, $K_{GRF,H}$, $D_{GRF,V}$
Cost function: $\min E = \sum_{j=1}^{6} w_j [\sum_{k=t_0}^{k=t_f} (\theta_j - \theta_{rj})^2]$
Constraints: $K_j^{min} < K_j < K_j^{max}$
$D_j^{min} < D_j < D_j^{max}$
$K_{GRF,V}^{min} < K_j < K_{GRF,V}^{max}$
$K_{GRF,H}^{min} < K_j < K_{GRF,H}^{max}$
$D_{GRF,V}^{min} < D_j < D_{GRF,V}^{max}$
$D_{GRF,H}^{min} < D_j < D_{GRF,H}^{max}$

Table 2.2: Minimum and Maximum Allowable Internal Spring and Damping Parameters

	Component	Minimum	Maximum
Ankle	Spring (Nm/deg)	0	3
	Damper (Nm(deg/s))	0	3
Knee	Spring (Nm/deg)	0	3
	Damper (Nm/(deg/s))	0	3
Hip	Spring (Nm/deg)	0	5
	Damper (Nm/(deg/s))	0	5
GRF - Horizontal	Spring (N/m)	0	130000
	Damper (N/(m/s))	0	50000
GRF - Vertical	Spring (N/m)	0	130000
	Damper (N/(m/s))	0	50000

The optimal internal mechanical parameters were obtained and listed in Tab. 2.3. With the calculated anthropometric and internal mechanical parameters,

Table 2.3: Optimized Internal Mechanical Parameters

	Component	Single Support	Double Support
Stance Ankle	Spring (Nm/deg)	0.3903	0.1054
	Damper (Nm(deg/s))	2.205	0.0988
Swing Ankle	Spring (Nm/deg)	0.7055	0.136
	Damper (Nm/(deg/s))	0.0643	0.1403
Stance Knee	Spring (Nm/deg)	0.1669	0.0278
	Damper (Nm/(deg/s))	0.8772	0.0595
Swing Knee	Spring (Nm/deg)	0.3002	0.0549
	Damper (Nm/(deg/s))	0.0832	0.052
Stance Hip	Spring (Nm/deg)	2.0244	0.0607
	Damper (Nm/(deg/s))	0.0242	0.0439
Swing Hip	Spring (Nm/deg)	0.741	0.0502
	Damper (Nm/(deg/s))	0.0012	0.0000049
GRF - Horizontal	Spring (N/m)	117650	10182
	Damper (N/(m/s))	197.8251	1720.1
GRF - Vertical	Spring (N/m)	129480	31795
	Damper (N/(m/s))	16587	7619.4

an open loop simulation can be performed to verify the fidelity of the plant model.

2.3 Open Loop Simulation

A forward dynamics open loop simulation was performed using the plant model and the parameters described in previous sections. The results are encouraging in that, by inputting the experimental moment data into the model, it can respond very closely to the experimental kinematics reference, i.e., the plant model can "walk" for one cycle open loop. The figures in Appendix A show the kinematics output of the model compared with the experimental reference for each joint during single support phase and double support phase. The root mean square error (RMSE) is listed in Tab. 2.4. Comparing with the range of motion of each joint, it can be seen that the RMSE is very small.

Several things are worth noticing in the simulation results. Figs. A.1, A.2 and A.12 show that even though the kinematic outputs of the plant model follow the experimental reference closely at the beginning of the simulation, the slope, i.e., the angular speed, deviates from the experimental reference at the end. This discrepancy may be because the spring and damping values are assumed to be constant inside the range of motion of the joints during the simulation, while in human body, the impedance of the joint is nonlinear with respect to angular position and tends to change at the transition from the single support to the double support or vice versa. Adding the angular speed error of these joints at the end of the simulation into the cost function may achieve better results and will be investigated in the future.

Table 2.4: Percentage Error Between the Open Loop Simulation and Experimental Kinematic Data

	Single Support Phase (deg)	SSP RMSE (deg)	Double Support Phase (deg)	DSP RMSE (deg)	Range of Motion (deg)
Stance Ankle	1.67 (2.6%)	0.944	0.17 (0.26%)	0.674	65
Swing Ankle	2.64 (4.1%)	2.273	1.75 (2.7%)	1.688	65
Stance Knee	1.95 (1.4%)	1.829	0.13 (0.093%)	1.007	140
Swing Knee	0.19 (0.13%)	11.877	0.84 (0.60%)	0.887	140
Stance Hip	0.82 (0.51%)	1.184	0.51 (0.32%)	0.236	160
Swing Hip	1.07 (0.67%)	4.289	1.72 (1.1%)	0.638	160

In Fig. A.5, because the knee joint has limitation in the range of motion in the model, the kinematic output of the swing knee during single support phase cannot follow the experimental reference. The lower limit of the knee joint is assumed to be 0°; the knee can only flex in one direction but cannot extend in the other way. However, the experimental data showed the knee joint goes below 0° which is unrealistic. The reason is unclear. Therefore, it is understandable that the kinematics output of the swing knee does not follow the experimental reference at the end of the single support phase.

Figs A.10, A.11 and A.12 show there are two sudden changes in the angular velocity in plant model output. One is at 0.06 sec and the other at 0.15 sec. Such sudden changes do not exist in the experimental reference. This sudden change is due to the fact that at those two time points, the heel and forefoot of the swing leg have initial contact with the ground. The same GRF spring and damping values are used at the heel and forefoot. From the biomechanics point of view, those values at the forefoot should be much smaller than the heel. Different values could be used at the heel and forefoot to obtain better results.

Given experimental moment inputs, the kinematics output of the plant model is within 4% percent difference to the experimental reference. The RMSE shown in Tab. 2.4 are very small compared with the range of motion. This plant model can perform similarly as experimental results with a seven segment nine DOF structure. The simulation results showed that this plant model has appropriate fidelity to represent the forward dynamics of the human gait. The MPC control system developed in the rest of this dissertation will be built to control this plant model.

2.4 Generality of the Open Loop Model

Table 2.5: Kinematics RMSE Between the Open Loop Simulation and Experimental Data for Three Other Subjects

	Subject 2(deg)		Subject 3(deg)		Subject 4(deg)	
	SSP	DSP	SSP	DSP	SSP	DSP
Stance Ankle	2.178	2.897	2.217	2.347	2.007	13.139
Swing Ankle	3.484	6.982	5.146	3.962	5.200	3.834
Stance Knee	8.985	7.570	8.799	2.367	2.209	5.685
Swing Knee	7.696	9.122	13.324	1.719	2.825	9.616
Stance Hip	12.447	7.210	3.079	9.800	2.825	9.616
Swing Hip	8.774	0.797	2.135	7.444	11.330	2.604

To show the generality of the developed open loop plant model, the same modeling methodology is applied to three other able-bodied subjects where the experimental data was collected under the same configuration in the Gait Lab at Medical College of Wisconsin. By maintaining the same plant model structure as explained in Sec. 2.1, the anthropometric data is customized to each of the subjects. However the internal mechanical parameters utilized are the same as the ones in Sec. 2.2 so that the generality of the open-loop model can be tested.

The kinematic simulation results are compared to the experimental data in Appendix B. It can be seen that without optimizing internal mechanical parameters according to each of the subject, the simulation results are not as close to the experimental data as shown in Sec. 2.3. To quantify the difference, the RMSE values between the model output and experimental data for each subject are listed in Tab. 2.5. Therefore it can be concluded that the developed open-loop plant model cannot be universally applied to different subjects. The anthropometric parameters and especially the internal mechanical parameters must be customarily optimized for each individual subjects, which reduces the general applicability of the open-loop plant model. The possibility of developing a general open-loop plant model can be considered as future work.

CHAPTER 3

Model Predictive Control Approach to Human Gait Modeling

The model described in Chap. 2 functions as the plant for the developed model. To complete the MPC control system, a control algorithm needs to be developed to function as the CNS. Unlike classical feedback control which adjusts control inputs based on past error, MPC is a branch of modern control theory which predicts the output of the plant and adjusts control input in advance. Like many other control methods, MPC has many branches. This chapter discusses how the critical aspects of MPC associated with human gait and which branch of MPC was implemented. First, the fundamental principle of MPC is described and the rationale for the MPC is justified; Second, the critical aspects of MPC are investigated and associated to human gait and the rationale for nonlinear end-point MPC control is explained. Third, after investigating the dynamics of human gait, a hybrid control approach which contains end-point MPC control and continuous PID control is selected and the reason is justified.

3.1 General Concept of MPC

All control algorithms can be broadly categorized into two categories: control based on past error or control based on prediction. Most control methods fall into the first category where the control input is generated based on the past difference between reference signals and outputs of the plant. The block diagram of this type

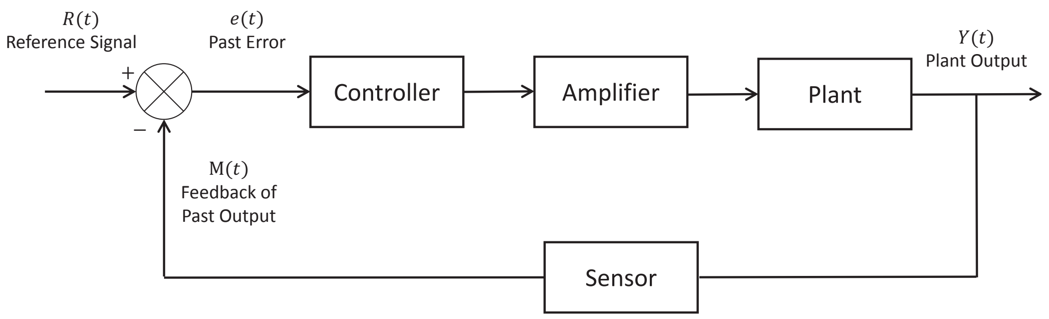

Figure 3.1: Typical Block Diagram of Control Method Based on Past Error

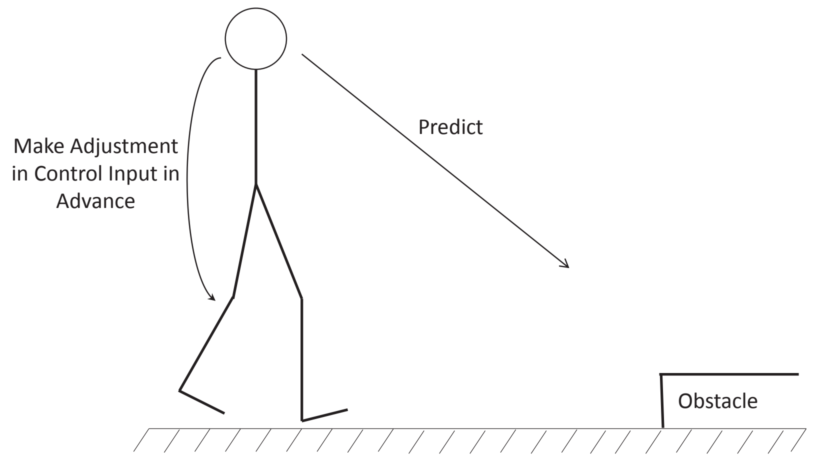

Figure 3.2: The CNS Predicts and Make Adjustment in Advance to Avoid Possible Failure

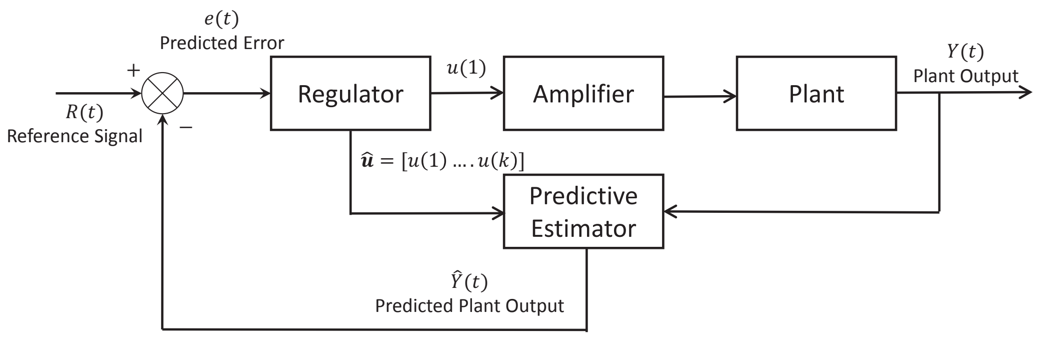

Figure 3.3: Block Diagram of Model Predictive Control

of control algorithm is shown in Fig. 3.1. PID control is the most common method in this type of control algorithm. It is widely used in industry because it is easy to understand, implement and adjust.

However, the essential principle of the CNS for human walking is different. Instead of controlling based on past error, the CNS uses feedback to predict what will happen in the future if the current walking pattern is maintained and make adjustments in the control inputs in advance to avoid any possible failure. For example, as shown in Fig. 3.2, the CNS makes the prediction that if the current walking pattern is maintained, an obstacle in the walking path will cause potential failure. Therefore, the CNS adjusts the joints moments so that the person can walk around or over the obstacle. If a PID control algorithm is employed in the CNS, the person would run into the obstacle first and then try to make adjustment; failure in walking will occur.

Therefore, it is hypothesized that the CNS employs a predictive control strategy during walking. Model Predictive Control is employed to simulate the CNS in this dissertation. MPC is a typical type of predictive control whose block diagram is shown in Fig. 3.3. The control strategy of MPC can be summarized as follows:

1. The future predicted outputs for a finite time horizon, P, called prediction horizon, are calculated at each time instant using an internal model. The internal model differs from the plant model developed in Chap. 2. The internal model is used by the MPC controller to predict future outputs while the plant model is used to represent the forward dynamics of the plant, which in this dissertation is the forward dynamics of the human gait. The predicted outputs, which can be expressed as $y(t + k \mid t)$, depend on the current states of the system and the future control inputs used. This process corresponds to the "Predictive Estimator" block in Fig. 3.3.

2. The future control signals for a finite time horizon, C, called control horizon, are calculated by optimizing a objective function to keep the plant as close as possible to the control reference. The objective function usually has the form of a quadratic function of the errors between the predicted outputs and the control reference. An explicit solution can be obtained if the objective function is quadratic, the internal model of MPC is linear, and there are no constraints. Otherwise an iterative method needs to be used. The control horizon is usually less than or equal to the prediction horizon ($C < P$). This process corresponds to the "Regulator" block in Fig. 3.3.

3. Once the control inputs are optimized, only the first time instant of the optimized control inputs is sent to actuators while the following ones are discarded. The control inputs of the second and subsequent time instants will be re-optimized for the following time steps because of the mismatch between the internal model used by MPC and the plant. If the MPC and plant are perfectly consistent and there is no noise, the control inputs need only optimized once and sent to the actuators. However, in real world applications the control inputs need to be re-calculated for every time step. The optimized control inputs, i.e., the joint moments, are generated by muscles which corresponds to the "Amplifier" in Fig. 3.3. The generated joint moments drive the plant, i.e., the human body, to move to the states of the next time step.

This process corresponds to the “Amplifier” and “Plant” blocks and their associated arrows in Fig. 3.3.

4. The optimized control inputs drive the plant to the next time step. The measured outputs of the plant are then fed back to the Predictive Estimator and the entire process is reiterated again from step 1.

3.2 Critical Aspects of MPC

Several aspects of the MPC control system are of critical importance. Therefore they need to be emphasized and discussed here as the choice of those critical aspects directly affect the performance of the system in this dissertation.

3.2.1 Internal Model of MPC

To implement MPC control, an internal model is used to predict the future plant outputs based on current plant states and future control inputs. The internal model plays a critical role in the control system. The developed internal model must be able to capture the dynamics of the plant to adequately predict the future outputs, and at the same time, be sufficiently simple to be simulated whithin several minutes for one iteration of simulation. In this research, this means the internal model needs to capture the essential forward dynamics of human gait and at the same time be simulated in a reasonable time frame.

In the chemical engineering industry, where MPC was originally developed, the most popular type of internal model is the an empirical model which is very simple to obtain as it only requires the measurement of the output when the plant is driven by a step or impulse input [41]. This type of model is widely accepted in industry because it is very intuitive and can be used for highly nonlinear processes. The drawbacks of empirical models are the large number of parameters needed and applicable to only open-loop stable processes. In addition, the most critical drawback of using an empirical model for this research is that it does not offer any insight into either the dynamics of human gait or the principles of the CNS.

Another possible type of internal MPC model is a State Space (SS) model which is widely used both in industry and academia. The SS model describes the plant process mathematically in the time domain. The general expression of a SS model is:

$$\begin{aligned} \dot{x}(t) &= \mathbf{f}(t, x(t), u(t)) \\ y(t) &= \mathbf{h}(t, x(t), u(t)) \\ x(t_0) &= x_0 \end{aligned} \tag{3.1}$$

where the first equation is called the state equation and second equation is called the output equation. $x(t)$ represents the states, $u(t)$ represents the inputs, $y(t)$ represents the outputs, and x_0 represents the initial states. Since MPC is a discrete time based control strategy, Eqn. 3.1 must be converted into a discrete form, which is expressed as:

$$\begin{aligned} x(k+1) &= \mathbf{f}(k, x(k), u(k)) \\ y(k) &= \mathbf{h}(k, x(k), u(k)) \\ x(k_0) &= x_0 \end{aligned} \tag{3.2}$$

where k is discrete time sample. Even highly nonlinear and multivariable processes can be represented by a SS model, which also has well developed stability and robustness criteria. More importantly, the SS model offers insight into the dynamic process of the plant. **Therefore, the SS approach will be utilized to build the internal MPC model in this dissertation.**

3.2.2 Objective Function

Once the internal model of MPC is developed, an objective function must be established to determine the optimal future inputs. The general aim for an objective function, J, is that the predicted future output along the prediction horizon P should be as close as possible to the reference, while the control inputs employed should be kept minimum. This philosophy can be expressed as [42]:

$$J(x(0), u) = \frac{1}{2} \sum_{k=N_0}^{N-1} [x(k)^T Q x(k) + u(k)^T R u(k)] + \frac{1}{2} x(N)^T Q_f x(N) \tag{3.3}$$

where N_0 is normally the current time, which is normally 0, N is the final time step,

Q is the weighting matrix for the predicted states along the prediction horizon, R is the weighting matrix for the control inputs, and Q_f is the weighting matrix for the final predicted states at the final time step.

There are three terms in Eqn. 3.3. The first term related to $x(k)$ is called the Stage Cost, the second term related to $u(k)$ is called the Control Input Cost, and the last term related to $x(N)$ is called the Terminal Cost. By tuning the relative ratios between the weighting matrices Q, R, and Q_f, the relative importance between the three different costs can be adjusted. For example, if Q_f is greater than Q, the objective function tightly enforces the final state of the plant to move to the reference value while stage cost during the process is ignored, and vice versa. This feature of MPC proves powerful in the development of the human gait model in this dissertation, and provides a significant advantage over traditional PID control.

3.2.3 Constraints

Another advantage of MPC over traditional PID control is that MPC is able to explicitly incorporate constraints into the controller. The control inputs for every physical system have limitations. In this dissertation, for example, the maximum moment inputs generated from the human joints such as ankle, knee, and hip are bounded. These constraints can be expressed as:

$$u_j^{min} \leq u_j(k) \leq u_j^{max} \tag{3.4}$$

If Eqn. 3.4 can be converted into linear inequality form which is expressed as:

$$Gu(k) \leq g \tag{3.5}$$

in which:

$$G = \begin{bmatrix} I \\ -I \end{bmatrix} \qquad g = \begin{bmatrix} \mathbf{u}_{max} \\ \mathbf{u}_{min} \end{bmatrix} \qquad \mathbf{u}_{max} = \begin{bmatrix} \mathbf{u}_1^{max} \\ \mathbf{u}_2^{max} \\ \vdots \\ \mathbf{u}_6^{max} \end{bmatrix} \qquad \mathbf{u}_{min} = \begin{bmatrix} \mathbf{u}_1^{min} \\ \mathbf{u}_2^{min} \\ \vdots \\ \mathbf{u}_6^{min} \end{bmatrix}$$

where I is the identity matrix.

Similar to constraints on the control input, it is also desirable to impose constraints on the states of the plant for safety and feasibility. In human gait, for example, there are limitations on the range of motion for each of the joints. This can be expressed as:

$$x_j^{min} \leq x_j(k) \leq x_j^{max} \tag{3.6}$$

Or in matrix form:

$$Hx(k) \leq f \tag{3.7}$$

where:

$$H = \begin{bmatrix} I \\ -I \end{bmatrix} \qquad f = \begin{bmatrix} \mathbf{x}_{max} \\ \mathbf{x}_{min} \end{bmatrix} \qquad \mathbf{x}_{max} = \begin{bmatrix} \mathbf{x}_1^{max} \\ \mathbf{x}_2^{max} \\ \vdots \\ \mathbf{x}_6^{max} \end{bmatrix} \qquad \mathbf{x}_{min} = \begin{bmatrix} \mathbf{x}_1^{min} \\ \mathbf{x}_2^{min} \\ \vdots \\ \mathbf{x}_6^{min} \end{bmatrix}$$

One distinction between control input constraints and state constraints is that control input constraints represent physical limitations, where the actuators are unable to generate control inputs beyond limitations. However, state constraints are desirable constraints that often can be relaxed for a certain range. For human gait, for example, some joints do not have a definite hard-stop constraint in their range of motion such as the hip. The developed MPC control system in this dissertation therefore must have hard constraints for the control input constraints and flexible constraints with modest of flexibility for the state constraints.

3.3 MPC Strategy in Human Gait Study

Before building the MPC control system, the structure of the MPC must be considered. The decisions must be made include whether to use linear or nonlinear state space representation for the internal MPC model, end-point or continuous MPC control, and traditional PID or MPC to control the orientation of HAT. These decisions directly affect the performance of the human gait model.

3.3.1 Linear or Nonlinear Internal State Space Model

There are two potential types of SS models to describe the target dynamic process: linear or nonlinear SS model. As previously described in Sec. 3.1, the discrete form of a *nonlinear* SS model can be expressed as:

$$\begin{aligned} x(k+1) &= \mathbf{f}(k, x(k), u(k)) \\ y(k) &= \mathbf{h}(k, x(k), u(k)) \\ x(k_0) &= x_0 \end{aligned} \tag{3.8}$$

The discrete form of *linear* SS model can be expressed as:

$$\begin{aligned} x(k+1) &= \mathbf{A}x(k) + \mathbf{B}u(k) \\ y(k) &= \mathbf{C}x(k) + \mathbf{D}u(k) \\ x(0) &= x_0 \end{aligned} \tag{3.9}$$

Every dynamic process is in fact a nonlinear process. Therefore, an inherent advantage of the nonlinear SS model is that it can describe the dynamic processes more accurately. However, for some simple engineering applications a linear SS model can describe the dynamic process very well because the nonlinear dynamics are subtle or outside the range of operation; such nonlinearities can be ignored without any obvious performance deterioration. For other situations, even though the dynamic process of the plant may be highly nonlinear, the plant performs around one operating point; therefore the nonlinear SS model can be linearized around that operating point and converted to linear model. In these cases, linear SS models are preferred because they can be easily integrated and can be implemented in real-time. This trade-off between the nonlinear and linear SS models is shown in Fig. 3.4.

For the internal SS model in this dissertation, an engineering decision needs to be made regarding whether a nonlinear or linear SS model will be utilized. As no plant processes need to be controlled in real-time, time is not a critical. Human gait is an highly nonlinear process that is inherently unstable; there are no steady state

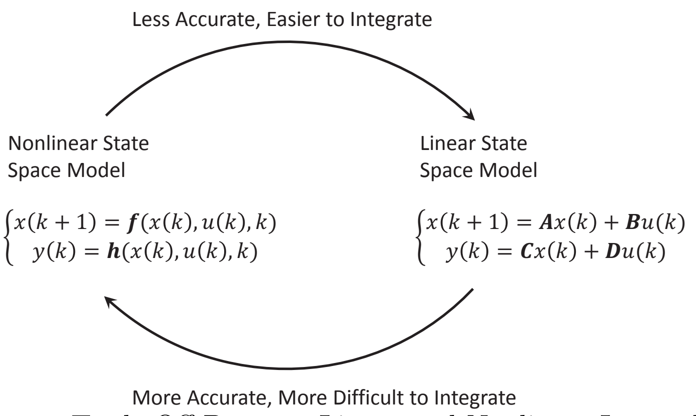
Less Accurate, Easier to Integrate
Nonlinear State Space Model
$\begin{cases} x(k+1) = \boldsymbol{f}(x(k), u(k), k) \\ y(k) = \boldsymbol{h}(x(k), u(k), k) \end{cases}$
Linear State Space Model
$\begin{cases} x(k+1) = \boldsymbol{A}x(k) + \boldsymbol{B}u(k) \\ y(k) = \boldsymbol{C}x(k) + \boldsymbol{D}u(k) \end{cases}$
More Accurate, More Difficult to Integrate

Figure 3.4: Trade-Off Between Linear and Nonlinear Internal Model

operating points about which linearization can be performed. A simple linear SS model is therefore not sufficient to represent the dynamics of human gait. **Therefore, a nonlinear SS model approach will be used to develop the internal MPC model of human gait.**

3.3.2 End-Point OR Continuous MPC Control

As previously described in Sec. 3.2, the objective function of MPC control takes the general form:

$$J(x(0), u) = \frac{1}{2} \sum_{k=N_0}^{N-1} [x(k)^T Q x(k) + u(k)^T R u(k)] + \frac{1}{2} x(N)^T Q_f x(N) \tag{3.10}$$

where Q is the weighting factor for the states, R is the weighting factor for the control inputs, and Q_f is the weighting factor for the final states at the end of the process. The first term is called Stage Cost which penalizes the error between the output and the reference during the process. The third term is called Terminal Cost which penalizes the error between the output and the reference at the end of the process. If the weighting matrix Q_f is made to be 0, the objective function becomes:

$$J(x(0), u) = \frac{1}{2} \sum_{k=N_0}^{N-1} [x(k)^T Q x(k) + u(k)^T R u(k)] \tag{3.11}$$

In this way, the terminal cost is ignored and the controller focuses on the outputs *during* the process. This control strategy is called Continuous MPC Control. By contrast, if the weighting matrix Q is made to be 0, the objective function becomes:

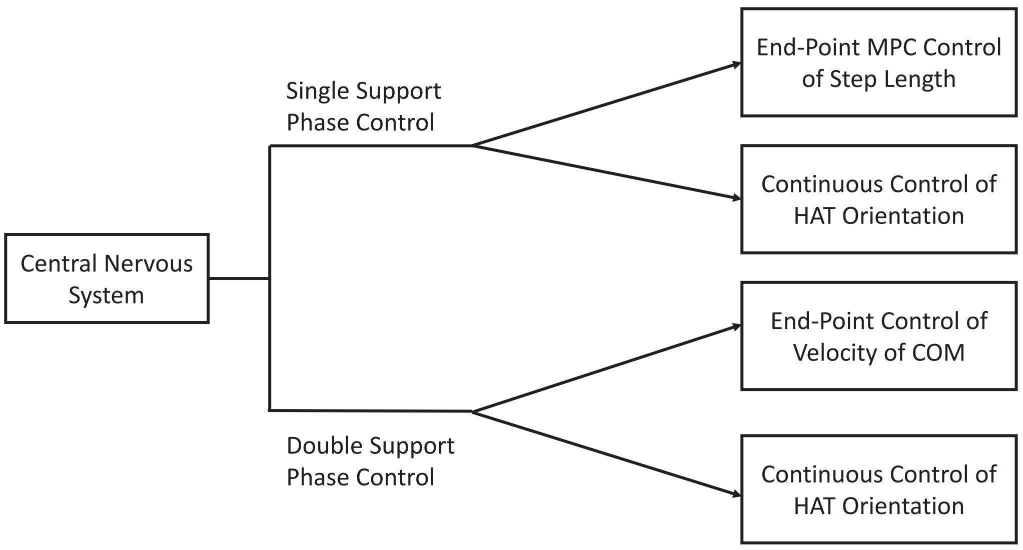
Central Nervous System
Single Support Phase Control
Double Support Phase Control
End-Point MPC Control of Step Length
Continuous Control of HAT Orientation
End-Point Control of Velocity of COM
Continuous Control of HAT Orientation

Figure 3.5: Proposed Control Strategy of CNS

$$J(x(0), u) = \frac{1}{2}\sum_{k=N_0}^{N-1} u(k)^T R u(k) + \frac{1}{2} x(N)^T Q_f x(N) \tag{3.12}$$

In this case, the outputs during the process are completely ignored and the controller focuses to bring the output of the reference at the end of the process. This control strategy is called End-Point MPC Control. MPC therefore can emphasize either the process or the final results.

Engineering judgment is needed to determine whether to emphasize on the end-point or continuous control. As the MPC is to approximate the CNS, the control objectives of the CNS need to be reviewed. Based on daily life experiences, the CNS does not appear to consciously control legs to follow a reference trajectory for the entire gait cycle. People walk naturally and subconsciously. It is proposed in this dissertation that the CNS only controls several critical gait related descriptors at the transitions between single and double support, while continuously keeping the HAT upright. This proposition is supported by the literature [33, 43]. During single support, the stance leg functions as an inverted pendulum without much control regulation so that the center of mass (COM) progresses forward with a consistent speed; the joint moments in the swing leg are controlled such that a target step length is achieved at the end of the phase. During double support, the joint moments are controlled such that the velocity of the COM achieves a certain value in preparation of the subsequent single support period. These two control objectives of the CNS are consistent with the philosophy of MPC end-point control. The CNS also maintains the HAT in an upright position during the entire gait cycle. This proposed control strategy of the CNS is shown in Fig. 3.5.

Therefore, based on the philosophy of MPC and the essential principle of the CNS, **by utilizing Eqn. 3.12, end-point MPC control is used to control the step length for Single Support Phase and the final velocity of the COM for Double Support Phase in order to simulate the function of the CNS.** The next question that needs to be answered is what control method should be used for continuous HAT orientation control during the entire gait cycle.

3.3.3 PID or MPC for HAT Orientation Control and Stance Knee Orientation Control During Single Support Phase

As stated in Winter [40], during able-bodied human walking, the HAT is continuously maintained upright. In addition, during single support, the stance knee is maintained straight such that the stance leg functions as an inverted pendulum to progress the body forward. There are two possible methods to control the HAT orientation and stance knee orientation. One possible solution is to use the MPC continuous control. As shown in Eqn. 3.11, this can be realized by setting the weighting matrix Q uniformly along the prediction horizon. The MPC controller predicts the orientation of the HAT or the stance knee along the prediction horizon and adjusts the control inputs such that the predicted orientation of the HAT and stance knee can be maintained vertical and extended. Another possible solution is to use traditional PID control algorithm to control the HAT and stance knee orientation by giving a constant upright and straight position reference to the controller. Although the two control methods may have similar performance, the control philosophies differ. In addition, if continuous MPC control is used, the determination of the weighting factors in Eqn. 3.12 is fairly complicated. As mentioned in Sec. 3.1, the continuous MPC controls the plant based on prediction of what will happen in the future, while PID controls the plant based on the past error between the plant output and reference.

Engineering judgment is therefore required to determine which method better approximates the CNS. In this dissertation, the author proposes that the continuous control of the HAT and stance knee orientation will be based on feedback of the HAT or stance knee's past deviation from upright or straight position but not from prediction. In another words, the deviation of the HAT from upright position or the deviation of the stance knee from being straight is fed back to the model and control inputs are generated based on this deviation, which is consistent with PID control philosophy. **Therefore, PID control is used to control the orientation of HAT and stance knee during single support.**

3.4 Summary

In this chapter, several critical aspects related to the proposed human gait model are discussed and the modeling methods are determined and justified. First, MPC will be used as the primary control method because the philosophy of MPC better approximates the CNS. Nonlinear first principle models will be used as the internal model of the MPC controller. End-point MPC control will be used to regulate step length for single support and velocity of the COM for double support phase. PID control will be used to control the orientation of the HAT and stance knee during single support, similar to the CNS feedback control regulation.

CHAPTER 4

Development of the Internal MPC Model

As previously mentioned in Sec. 3.2, the internal MPC model used to predict the future outputs of the plant plays a key role in the performance of the MPC system. This chapter is dedicated to the development of this model. First, the requirements of an appropriate internal MPC model is discussed and will serve as guidelines for model development. The second section describes the development of the internal MPC model for the single support phase; the third section discusses the development of the internal MPC model for the double support phase.

4.1 Guidelines for the Internal MPC Model

Before the internal MPC model can be developed, the measures of an appropriate model need to be clarified. Based on the philosophy of the MPC algorithm and the nature of human gait, a good internal MPC model needs to meet the following requirements: (1) the internal model should be simple, (2) the single support and double support phases should be simulated separately, and (3) several joint moments are not required to be modeled in the MPC internal model because they are controlled by the PID controller.

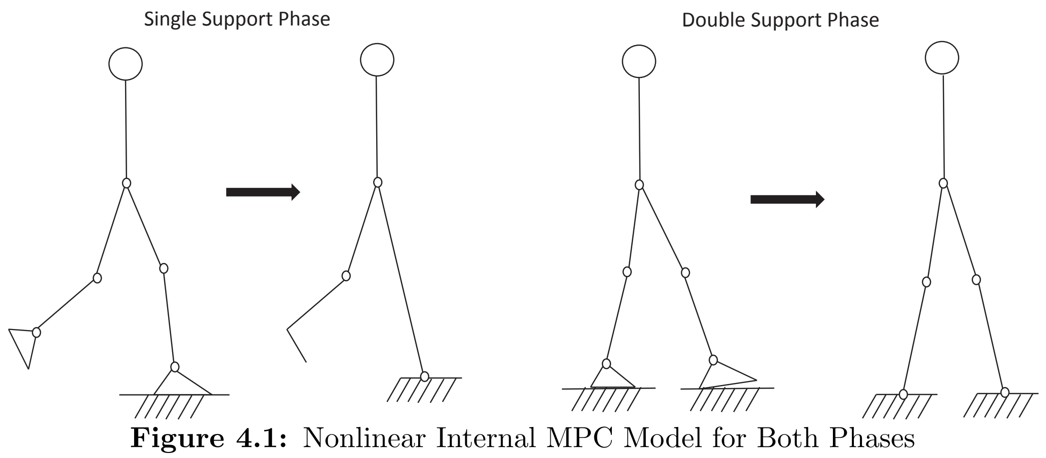

Figure 4.1: Nonlinear Internal MPC Model for Both Phases

4.1.1 Simplicity is Critical

One of the questions that may be asked is: “Since a plant model is already developed, why can’t this model be used as the internal MPC model?” To answer this question, the differences between the purposes of the two models need to be clarified. The plant model developed in Chap. 2 represents the dynamic process of human gait. Ideally the most realistic human gait model should be used in the developed system. The purpose of the plant model is to simulate the dynamic process of human gait as closely as possible with potential subsequent refinement. The purpose of the MPC internal model is to be a control-oriented model which can represent the essential forward dynamics of human gait and be simple enough to be implemented within several minutes for one iteration. If the plant model is used as the internal MPC model, the model run time will be too long; therefore virtual testing of the P&Os will be unpractical. From another perspective, the use of the plant model as the internal MPC model is contrary to the assumption that a simple control-oriented model is used by CNS to control the human gait. Therefore, “parsimony” serves as the most important requirement for the development of the internal MPC model that; as few variables and parameters as possible should be used to sufficiently represent the forward dynamics of human gait. This requirement is also consistent with daily experience that people can walk without much cognitive effort.

4.1.2 Single Support and Double Support Phase Should be Simulated Separately

As previously mentioned in Chap. 2, a gait cycle can be divided into single support and double support phases. Another question that needs to be addressed is: “Is it possible to develop one internal MPC model to represent both phases OR are two models, one for each phase, required?” This question can be answered from both dynamics’ and controls’ perspectives. As shown in Fig. 4.1, the single support and double support phases are dynamically two distinct processes; an open kinematic chain and a closed kinematic chain, respectively. The constraints and EOMs differ. From a controls perspective, single support and double support phases have different end-point control objectives, as previously described in Chap. 3.

Therefore, two different internal MPC models are developed for single support and double support phases, respectively.

4.1.3 Not Every Joint Moment Is Required

As previously mentioned in Chap. 3, during Single Support Phase, because HAT and stance knee are controlled continuously by the PID controller and not controlled by the MPC controller, they are assumed to be straight in the internal MPC model during the entire phase. Another question that should be raised is: “Since the orientation of the HAT and stance knee are controlled by the PID controllers, is it necessary to model all joints in the MPC internal model?” The HAT orientation is primarily controlled by the stance hip moment; similarly the stance knee is controlled largely by the stance knee moment. These two moment sources are not modeled in the internal MPC model. Likewise during double support phase, the orientation of HAT is controlled continuously by the PID controller and is again assumed to be upright during the entire phase. The bilateral hip moments are therefore regulated by the PID controller to control the HAT so they are not part of the Double Support Phase internal MPC model.

The explanation serve as the guidelines to develop the internal MPC model. The next two sections describe in detail the structures of the internal MPC model for single support and double support phases, respectively, as well as the development of the EOMs in SS form.

4.2 Internal MPC Model for Single Support Phase

The graphical representation of the internal MPC model used for single support is shown in Fig. 4.2. For convenience, the swing limb is assumed to be the right limb and stance limb is assumed to be the left limb. The orientation of HAT is controlled by a PID controller and assumed to remain upright in the internal model; it can only translate forward or backward and does not rotate. Therefore, the HAT is modeled as a point mass on top of both thighs. The stance knee is assumed to be fully extended during the single support. The moments at the stance hip and stance knee are not part of the internal MPC model since they will be controlled by a separate PID controller. As the stance foot is in contact with the ground throughout

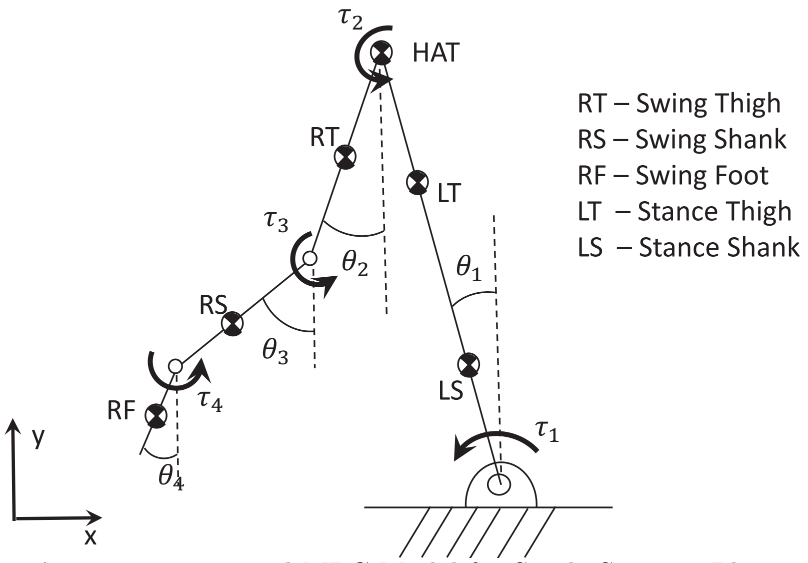

Figure 4.2: Internal MPC Model for Single Support Phase

single support, the stance foot is not modeled and is combined into the stance leg. The stance ankle is treated as a revolute joint, connecting the stance limb to the ground. The reaction force in the revolute joint is simply the GRF, the sole external force to advance the body forward. Each of the joints are modeled in the same way as for the forward dynamics plant model developed in Chap. 2. The respective spring stiffness and damping coefficients are the same as those in the plant model.

As previously discussed in Chap. 3, the stance leg essentially functions as an inverted pendulum to advance the body forward. The moment at the stance ankle does not significantly accelerate or decelerate the COM given the less than half second duration of single support; instead this moment is used to fine-tune and maintain the momentum of the body. The MPC mainly controls the joint moments of the swing side limb so that at the end of single support, a specified step length can be achieved.

EOMs are required to convert the previously described model into SS form to be used by the MPC controller. Lagrange's equation is utilized because it is the easiest method in developing EOMs for multiple DOF systems. Based on the description of the model, the four generalized coordinates are the angular position of the stance ankle, swing hip, swing knee, and swing ankle, labeled θ_1 to θ_4, respectively in Fig. 4.2. The four generalized "forces" are the moments at the stance ankle, swing hip, swing knee, and swing ankle, labeled τ_1 to τ_4 respectively in

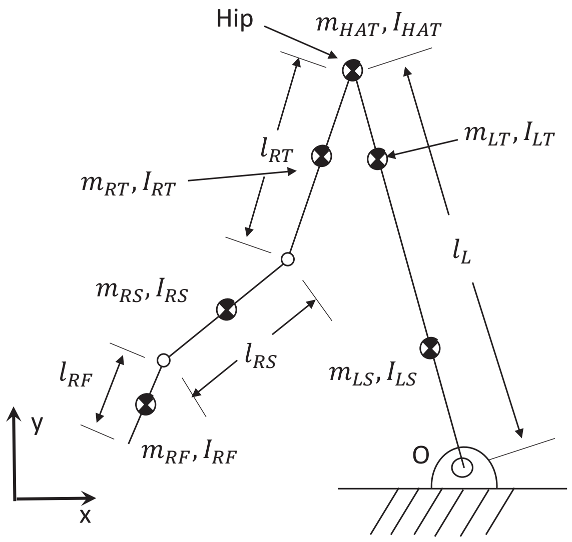

Figure 4.3: Anthropometric Parameters of the Internal MPC Model for Single Support Phase

Fig. 4.2. The absolute angular positions with respect to the global reference frame are used as the generalized coordinates to make the EOMs simpler. The anthropometric parameters of the model are defined symbolically as shown in Fig. 4.3, where m represents segment mass, l represents segment length, and I represents segment moment of inertia with respect to the center of mass; the subscription LT represents "left thigh", LS represents "left shank", L represents the left limb, RT represents "right thigh", RS represents "right shank", RF represents "right foot".

The first step in Lagrange's equation is to find the kinetic energy of the system. The kinetic energy of the stance thigh and shank can be expressed as:

$$T_1 = \frac{1}{2}(I'_{LS} + I'_{LT})\dot{\theta}_1^2 \tag{4.1}$$

where I'_{LS} and I'_{LT} represent the calculated moment of inertia of stance shank and thigh with respect to point O, as shown in Fig. 4.3, using the parallel axis theorem. The kinetic energy of the swing thigh can be expressed as the combination of its translational and rotational kinetic energy:

$$T_2 = \frac{1}{2}m_{RT}v_{RT}^2 + \frac{1}{2}I_{RT}\omega_{RT}^2 \tag{4.2}$$

where v_{RT} represents the translational velocity of the swing thigh. The angular velocity $\vec{\omega}_{RT}$ can be expressed as:

$$\omega_{RT} = \dot{\theta}_2 \tag{4.3}$$

Since the swing thigh is connected to the stance hip, the translational velocity of the right thigh can be expressed with respect to the hip of the stance leg as:

$$\vec{v}_{RT} = \vec{v}_{hip} + \vec{\omega}_{RT} \times \vec{r}_{RT/HAT} \tag{4.4}$$

where $\vec{r}_{RT/HAT}$ represents the position vector from the HAT to the COM of the swing thigh. After substituting the parameters into Eqn. 4.4:

$$\vec{v}_{RT} = (\frac{1}{2}l_{RT}\dot{\theta}_2\cos\theta_2 - l_L\dot{\theta}_1\cos\theta_1)\hat{\imath} + (\frac{1}{2}l_{RT}\dot{\theta}_2\sin\theta_2 - l_L\dot{\theta}_1\sin\theta_1)\hat{\jmath} \tag{4.5}$$

By substituting Eqn. 4.3 and Eqn. 4.5 into Eqn. 4.2, the kinetic energy of the swing thigh can be described as a function of θ_1, θ_2, $\dot{\theta}_1$, and $\dot{\theta}_2$ as:

$$T_2 = f_2(\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2) \tag{4.6}$$

Similarly, the kinetic energy of the swing shank is:

$$T_3 = \frac{1}{2}m_{RS}v_{RS}^2 + \frac{1}{2}I_{RS}\omega_{RS}^2 = f_3(\theta_1, \theta_2, \theta_3, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3) \tag{4.7}$$

and the kinetic energy of the swing foot is:

$$T_4 = \frac{1}{2}m_{RF}v_{RF}^2 + \frac{1}{2}I_{RF}\omega_{RF}^2 = f_4(\theta_1, \theta_2, \theta_3, \theta_4, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\theta}_4) \tag{4.8}$$

Because the HAT is assumed to be a point mass connected to the hip joint, it only has translational kinetic energy and can be expressed as:

$$T_5 = \frac{1}{2}m_{HAT}v_{HAT}^2 = \frac{1}{2}m_{HAT}l_L^2\dot{\theta}_1^2 \tag{4.9}$$

The total kinetic energy of the model is defined as the summation of the kinetic energy of the five segments as:

$$T = T_1 + T_2 + T_3 + T_4 + T_5 \tag{4.10}$$

The second step in Lagrange's equation is to find the potential energy of the system. The only potential energy in this model is the gravitational potential energy for each of the segments, V_1 through V_4. They can be expressed as:

$$\begin{aligned} V_1 = [m_{LS}c_{LS}l_{LS} + m_{LT}(c_{LT}l_{LT} + l_{LS}) + m_{HAT}(l_{LT} + l_{LS})]g\cos\theta_1 \\ V_2 = m_{RT}(l_L\cos\theta_1 - c_{RT}l_{RT}\cos\theta_2)g \\ V_3 = m_{RS}(l_L\cos\theta_1 - l_{RT}\cos\theta_2 - c_{RS}l_{RS}\cos\theta_3)g \\ V_4 = m_{RF}(l_L\cos\theta_1 - l_{RT}\cos\theta_2 - l_{RS}\cos\theta_3 - c_{RF}l_{RF}\cos\theta_4)g \end{aligned} \tag{4.11}$$

where c_{LS}, c_{LT}, c_{RT}, c_{RS}, and c_{RF} represent the position of the COM relative to the segment length. Therefore, the total potential energy of the model can be expressed as the summation of the terms in Eqn. 4.12.

$$V = V_1 + V_2 + V_3 + V_4 \tag{4.12}$$

After the kinetic and potential energy of each of the segments are defined, Lagrange's equation can be applied using matrix form:

$$\begin{bmatrix} \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_1}) - \frac{\partial}{\partial\theta_1} & \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_1}) - \frac{\partial}{\partial\theta_1} & \cdots & \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_1}) - \frac{\partial}{\partial\theta_1} \\ \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_2}) - \frac{\partial}{\partial\theta_2} & \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_2}) - \frac{\partial}{\partial\theta_2} & \cdots & \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_2}) - \frac{\partial}{\partial\theta_2} \\ \vdots & & \ddots & \vdots \\ \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_4}) - \frac{\partial}{\partial\theta_4} & \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_4}) - \frac{\partial}{\partial\theta_4} & \cdots & \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_4}) - \frac{\partial}{\partial\theta_4} \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix} + \begin{bmatrix} \frac{\partial}{\partial\theta_1} & \frac{\partial}{\partial\theta_1} & \cdots & \frac{\partial}{\partial\theta_1} \\ \frac{\partial}{\partial\theta_2} & \frac{\partial}{\partial\theta_2} & \cdots & \frac{\partial}{\partial\theta_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial}{\partial\theta_4} & \frac{\partial}{\partial\theta_4} & \cdots & \frac{\partial}{\partial\theta_4} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \\ \tau_4 \end{bmatrix} \tag{4.13}$$

To convert Eqn. 4.13 to SS form, the state vector is defined as:

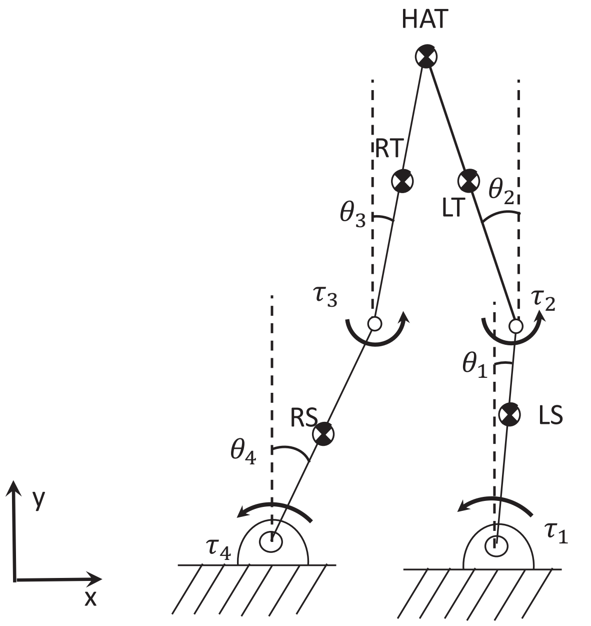

Figure 4.4: Internal MPC Model for Double Support Phase

$$\vec{x} = \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_7 \\ x_8 \end{Bmatrix} = \begin{Bmatrix} \theta_1 \\ \dot{\theta}_1 \\ \theta_2 \\ \dot{\theta}_2 \\ \vdots \\ \theta_4 \\ \dot{\theta}_4 \end{Bmatrix} \tag{4.14}$$

Eqn. 4.13 can then be converted to the SS form as:

$$\dot{\vec{x}}(t) = \vec{\mathbf{h}}(\vec{x}(t), \vec{\tau}(t)) \tag{4.15}$$

Eqn. 4.15 has the proper form to be used by the MPC controller.

Although the proposed internal MPC model for single support is significantly simplified compared to the plant model, the EOMs are still highly nonlinear; it is not practical, or even possible, to solve them analytically. These equations can be solved numerically in a reasonable amount of time.

4.3 Internal MPC Model for Double Support Phase

For the double support phase, a similar procedure is utilized. However, both feet are in contact with the ground during double support, forming a closed

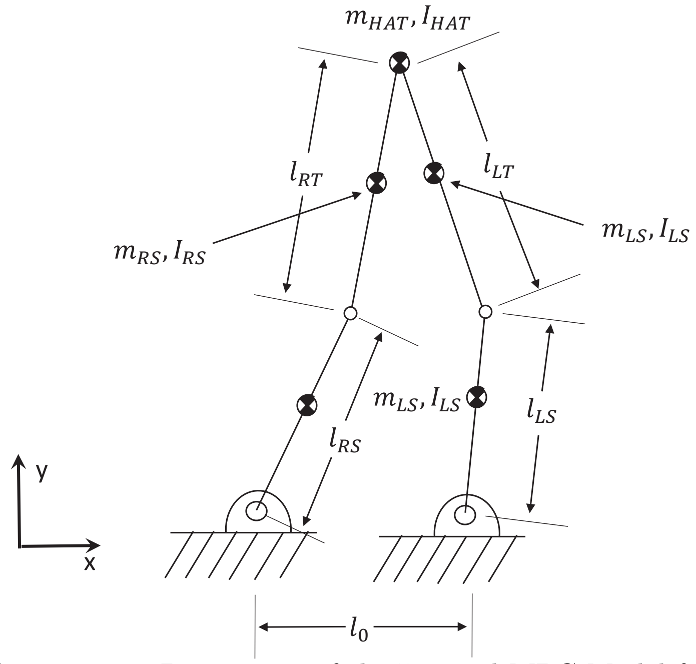

Figure 4.5: Anthropometric Parameters of the Internal MPC Model for Double Support Phase

kinematic chain as shown in Fig. 4.1. Because both feet have small displacement when in contact with the ground, they can be modeled as fixed and the ankles modeled as two revolute joints connected to the ground. The graphical representation of the corresponding internal model is shown in Fig. 4.4. The leading limb is assumed to be the right side limb and the trailing limb is assumed to be the left limb, which is the subsequent double support following the single support described in Sec. 4.2. The HAT is assumed to be upright during the entire double support phase; since it is controlled by a separate PID controller, it is modeled as a point mass in the internal MPC model. The configuration of the double support phase shown in Fig. 4.4 can be considered a classic five bar linkage with two DOFs. Constraints exist between the four angular positions, θ_1 to θ_4, and only two of them are independent variables.

There are two possible methods to build the EOMs with Lagrange's equation. The first method is to build the constraint equations and substitute them into the kinetic and potential energy terms to eliminate the dependent variables and only leave the unconstrained generalized coordinates. However, because of the complicated constraint equations, using unconstrained generalized coordinates is difficult.

An alternative method is to keep all the constrained generalized coordinates (θ_1 to θ_4) and introduce Lagrange multipliers to add the constraints into the EOMs. This method makes the EOMs concise and easily derived. The trade-off is that the burden on deriving the EOMs will be shifted to solving them. However, advances in computational power make solving the constrained equations feasible. Therefore, Lagrange multipliers are introduced into the EOMs and the four constrained generalized coordinates are utilized.

The anthropometric parameters of the model are defined symbolically as shown in Fig. 4.5. The closed kinematic chain generates two constraint equations in the x and y directions, respectively. Those constraint equations can be expressed as:

$$\begin{cases} l_{RS}\sin\theta_4 + l_{RT}\sin\theta_3 + l_{LT}\sin\theta_2 - l_{LS}\sin\theta_1 = l_0 \\ l_{RS}\cos\theta_4 + l_{RT}\cos\theta_3 - l_{LT}\cos\theta_2 - l_{LS}\cos\theta_1 = 0 \end{cases} \tag{4.16}$$

Differentiating Eqn. 4.16 with respect to time yields the velocity constraint equations:

$$\begin{cases} l_{RS}\cos(\theta_4)\dot\theta_4 + l_{RT}\cos(\theta_3)\dot\theta_3 + l_{LT}\cos(\theta_2)\dot\theta_2 - l_{LS}\cos(\theta_1)\dot\theta_1 = 0 \\ -l_{RS}\sin(\theta_4)\dot\theta_4 - l_{RT}\sin(\theta_3)\dot\theta_3 + l_{LT}\sin(\theta_2)\dot\theta_2 + l_{LS}\sin(\theta_1)\dot\theta_1 = 0 \end{cases} \tag{4.17}$$

Following similar procedure as in Sec. 4.2, the kinetic energy of the each segment is:

$$\begin{cases} T_1 = \frac{1}{2}I_{LS}\dot{\theta_1}^2 + \frac{1}{2}m_{LS}v_{LS}^2 2 \\ T_2 = \frac{1}{2}I_{LT}\dot{\theta_2}^2 + \frac{1}{2}m_{LT}v_{LT}^2 \\ T_3 = \frac{1}{2}I_{RT}\dot{\theta_3}^2 + \frac{1}{2}m_{RT}v_{RT}^2 \\ T_4 = \frac{1}{2}I_{RS}\dot{\theta_4}^2 + \frac{1}{2}m_{RS}v_{RS}^2 \\ T_{HAT} = \frac{1}{2}m_{HAT}v_{HAT}^2 \end{cases} \tag{4.18}$$

where the translational velocity terms are obtained as:

$$\begin{cases} \vec{v}_{LS} = \frac{1}{2}l_{LS}\dot{\theta}_1\cos\theta_1\hat{\imath} - \frac{1}{2}l_{LS}\dot{\theta}_1\sin\theta_1\hat{\jmath} \\ \vec{v}_{LT} = (l_{LS}\dot{\theta}_1\cos\theta_1 - \frac{1}{2}l_{LT}\dot{\theta}_2\cos\theta_2)\hat{\imath} - (l_{LS}\dot{\theta}_1\sin\theta_1 + \frac{1}{2}l_{LT}\dot{\theta}_2\sin\theta_2)\hat{\jmath} \\ \vec{v}_{RT} = (l_{LS}\dot{\theta}_1\cos\theta_1 - l_{LT}\dot{\theta}_2\cos\theta_2 - \frac{1}{2}l_{RT}\dot{\theta}_3\cos\theta_3)\hat{\imath} \\ \qquad\qquad (-l_{LS}\dot{\theta}_1\sin\theta_1 - l_{LT}\dot{\theta}_2\sin\theta_2 + \frac{1}{2}l_{RT}\dot{\theta}_3\sin\theta_3)\hat{\jmath} \\ \vec{v}_{RS} = (l_{LS}\dot{\theta}_1\cos\theta_1 - l_{LT}\dot{\theta}_2\cos\theta_2 - l_{RT}\dot{\theta}_3\cos\theta_3 - \frac{1}{2}l_{RS}\dot{\theta}_4\cos\theta_4)\hat{\imath} \\ \qquad\qquad +(-l_{LS}\dot{\theta}_1\sin\theta_1 - l_{LT}\dot{\theta}_2\sin\theta_2 + l_{RT}\dot{\theta}_3\sin\theta_3 + \frac{1}{2}l_{RS}\dot{\theta}_4\sin\theta_4)\hat{\jmath} \\ \vec{v}_{HAT} = (l_{LS}\dot{\theta}_1\cos\theta_1 - l_{LT}\dot{\theta}_2\cos\theta_2)\hat{\imath} - (l_{LS}\dot{\theta}_1\sin\theta_1 + l_{LT}\dot{\theta}_2\sin\theta_2)\hat{\jmath} \end{cases}$$

The only potential energy of each segment is gravitational energy which can be calculated as:

$$\begin{cases} V_1 = \frac{1}{2}m_{LS}l_{LS}g\cos\theta_1 \\ V_2 = m_{LT}(l_{LS}\cos\theta_1 + \frac{1}{2}l_{LT}\cos\theta_2)g \\ V_3 = m_{RT}(l_{RS}\cos\theta_4 + \frac{1}{2}l_{RT}\cos\theta_3)g \\ V_4 = \frac{1}{2}m_{RS}l_{RS}g\cos\theta_4 \\ V_{HAT} = m_{HAT}(l_{LS}\cos\theta_1 + l_{LT}\cos\theta_2)g \end{cases} \tag{4.19}$$

After the kinetic and potential energy of each segment is calculated, Lagrange's equation can be applied using a matrix formulation. Because two velocity constraint equations exist, two Lagrange multipliers λ_1 and λ_2 are introduced into the EOMs. The final EOMs are:

$$\begin{bmatrix} \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_1}) - \frac{\partial}{\partial\theta_1} & \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_1}) - \frac{\partial}{\partial\theta_1} & \cdots & \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_1}) - \frac{\partial}{\partial\theta_1} \\ \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_2}) - \frac{\partial}{\partial\theta_2} & \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_2}) - \frac{\partial}{\partial\theta_2} & \cdots & \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_2}) - \frac{\partial}{\partial\theta_2} \\ \vdots & & \ddots & \vdots \\ \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_4}) - \frac{\partial}{\partial\theta_4} & \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_4}) - \frac{\partial}{\partial\theta_4} & \cdots & \frac{d}{dt}(\frac{\partial}{\partial\dot{\theta}_4}) - \frac{\partial}{\partial\theta_4} \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_{HAT} \end{bmatrix}$$

$$+ \begin{bmatrix} \frac{\partial}{\partial\theta_1} & \frac{\partial}{\partial\theta_1} & \cdots & \frac{\partial}{\partial\theta_1} \\ \frac{\partial}{\partial\theta_2} & \frac{\partial}{\partial\theta_2} & \cdots & \frac{\partial}{\partial\theta_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial}{\partial\theta_4} & \frac{\partial}{\partial\theta_4} & \cdots & \frac{\partial}{\partial\theta_4} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_{HAT} \end{bmatrix} = \begin{bmatrix} \tau_1 + (-l_{LS}\cos\theta_1)\lambda_1 + l_{LS}\sin\theta_1\lambda_2 \\ \tau_2 + l_{LT}\cos\theta_2\lambda_1 + l_{LT}\sin\theta_2\lambda_2 \\ \tau_3 + l_{RT}\cos\theta_3\lambda_1 + (-l_{RT}\sin\theta_3)\lambda_2 \\ \tau_4 + l_{RS}\cos\theta_4\lambda_1 + (-l_{RS}\sin\theta_4)\lambda_2 \end{bmatrix} \tag{4.20}$$

Eqn. 4.20 together with the constraint of Eqn. 4.16 can be solved numerically making the prediction of the future outputs possible. The form of Eqn. 4.20 is still concise with constraints incorporated. As previously mentioned, the trade-off is more computational power is required to integrate Eqn. 4.20 than to substitute the constraints into the EOMs and eliminate the dependent variables. However, since real-time control is not a concern in this dissertation, additional computational time is not a problem.

4.4 Conclusion

In this chapter, the internal models used by MPC are developed. After investigating the philosophy of MPC and the nature of human gait, separate internal models are developed for single support and double support phases. The internal models for both phases are presented and EOMs are developed based on the proposed physical model. The derived equations facilitate the prediction of future outputs for the MPC controller.

CHAPTER 5

MPC Control System

The previous chapters developed a forward dynamics plant model of human gait that is used as the control target, established Model Predictive Control as the primary control algorithm, specified the controller configuration, and defined internal MPC models to be used for prediction purposes. With the elements of the system developed, this chapter describes the proposed human gait simulation in detail from a system level. First, the overall block diagram of the entire system is described. Second, MPC related parameters such as prediction horizon, control horizon, and constraints are discussed. Third, the objective functions of MPC are developed. A special mathematical function, Laguerre functions, is introduced as the form of optimized MPC control input to reduce the computation load. Lastly, the PID control used to maintain the orientation of HAT and stance knee during single support is explained.

5.1 Overall Control Algorithms

The control algorithms of the entire system were developed based on MPC, the essential principles of the CNS, and human gait dynamics as shown in Fig. 5.1. The system is divided into single support and double support phase control blocks.

For single support control, two controllers are combined to regulate the moment at each joint. The MPC functions as the main controller to regulate the stance ankle and swing leg moments to achieve a specified step length. The PID controllers function as the auxiliary controllers to regulate the stance hip and stance knee moments to maintain the upright orientation of the HAT and full extension of the stance knee. The single support phase model developed in Chap. 4 is used as the MPC internal model. The forward dynamics plant model developed in Chap. 2 is employed as the control target (i.e., the plant) and its kinematic output provides feedback to the MPC and PID controllers.

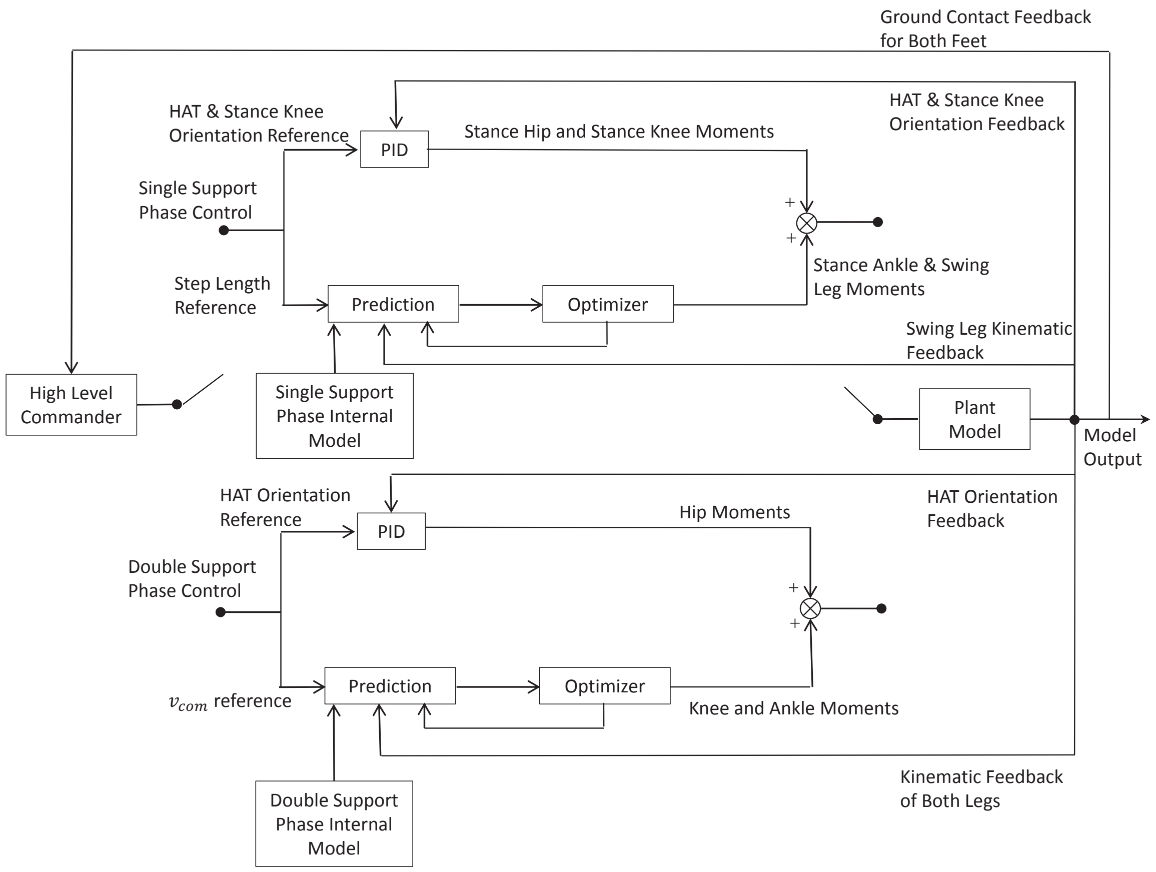
Ground Contact Feedback for Both Feet
HAT & Stance Knee Orientation Reference
PID
Stance Hip and Stance Knee Moments
HAT & Stance Knee Orientation Feedback
Single Support Phase Control
+
+
Stance Ankle & Swing Leg Moments
Step Length Reference
Prediction
Optimizer
Swing Leg Kinematic Feedback
High Level Commander
Single Support Phase Internal Model
Plant Model
Model Output
HAT Orientation Reference
PID
Hip Moments
HAT Orientation Feedback
Double Support Phase Control
+
+
v_{com} reference
Prediction
Optimizer
Knee and Ankle Moments
Kinematic Feedback of Both Legs
Double Support Phase Internal Model

Figure 5.1: Control Algorithm of the Entire System

Similarly for double support phase control, a MPC and PID controllers are combined together to regulate the moment at each joint. The MPC functions as the main controller to regulate the ankle and knee moments of both limbs to achieve specified velocity of the COM at the end of double support. The PID controllers function as auxiliary controllers to regulate the hip moments of both limbs to maintain the orientation of HAT. The double support phase model developed in Chap. 4 is used as the MPC internal model. The same forward dynamics plant model developed in Chap. 2 is again employed as the control target (i.e., the plant) and its kinematic output provides feedback to double support phase controllers.

The control structure for single support and double support is a similar. Differences firstly exist in the control reference. For single support, the control reference is step length while the control reference for double support is the terminal velocity of the COM. Difference also lies in the different internal models they use which are based on the distinct difference in dynamics between the two phases. The third difference lies in the different joint moments controlled by the MPC. For single support, the MPC does not control the stance knee moment; for double support, the MPC controls the moments at both knees. This approach is used because during single support the stance limb functions as an inverted pendulum and the stance knee remains fully extended.

The strategy of the fully extended stance knee is closer to feedback control rather than predictive control. Therefore, PID control is used to control the moment at stance knee. However, for double support, the moments for both knees need to be regulated to adjust the velocity of COM to the specified value; this strategy approximates predictive control.

With the overall control algorithm of the system determined, some of the critical parameters related to MPC controller still need to be developed. The next section discusses the determination of the prediction horizon, control horizon, and constraints.

5.2 MPC Related Parameters - Prediction Horizon, Control Horizon, and Constraints

Two of the important parameters that need to be determined in any MPC system are Prediction Horizon, P, and Control Horizon, C. Prediction Horizon determines how many future time steps the MPC predicts the plant states by applying candidate future control inputs into the internal MPC model. Control Horizon determines how many time steps the optimized future control inputs are generated. Normally longer Prediction Horizon and Control Horizon produce more accurate MPC performance but require more computational power. Therefore, there exists a trade-off between performance and control speed in MPC which is critical for the situations where real-time control is required.

Since end-point MPC is used for both Single Support and Double Support Phase simulation, the Prediction and Control Horizon are from the current time step to the end time step of the respective phases. Therefore the Prediction and Control Horizon are not constant and decrease as time progresses. They can be expressed in mathematical form as:

$$\mathbf{x_p} = \underbrace{\{x_{t+1}, x_{t+2}, \cdots, x_N\}}_{P} \tag{5.1}$$

$$\mathbf{u_C} = \underbrace{\{u_t, u_{t+1}, \cdots, u_{N-1}\}}_{C} \tag{5.2}$$

where t represents the current time step, N represents the final time step in the respective single support or double support phase, x_P is the predicted state, u_C is the optimized future control input, P represents the Prediction Horizon, and C represents the Control Horizon. To give an example how P and C changes as time proceeds, assume $t = 1$ and the final time step $N = 50$, the MPC controller predicts the future states and optimizes future control inputs from the first time step until the final 50th time step; therefore $P = C = 49$. When the current time proceeds to the next time step, $t = 2$, the MPC controller predicts the states and optimizes future control inputs from the second time step until the final 50th time step which makes $P = C = 48$. Therefore the Prediction Horizon and Control Horizon decrease as time proceeds.

After the Prediction Horizon and Control Horizon are determined, the remaining parameters that need to be determined are the constraints related to optimized control inputs. Like any other control algorithm, MPC should not generate control inputs beyond the capability of actuators which, in this dissertation, are maximum moments a specific joint is able to generate. This constraint can be expressed as:

$$u_j^{min} \leq u_j(k) \leq u_j^{max}, \quad k \in (0, N) \tag{5.3}$$

where u_j represents optimized joint moment control input at an individual joint, j, and time step, k. In addition, because muscles can only generate continuous joint moments, limitations on the incremental change in the optimized control inputs between each time step also exist. This can be expressed as:

$$\Delta u_j^{min} \leq \Delta u_j(k) \leq \Delta u_j^{max}, \quad k \in (0, N) \tag{5.4}$$

where Δu_j represents the incremental change in the optimized control input between two adjacent time steps at an individual joint. The limitation in optimized control inputs ensures that unrealistic control inputs cannot be generated from MPC.

5.3 Objective Function

The objective function is an essential part of the MPC algorithm. Its function is similar to that of the brain in the CNS; it determines what control inputs should be employed to generate the optimal gait. In this dissertation, the proposed objective for the single support phase is to achieve a specific step length; the objective for the double support phase is to achieve a specific velocity of the COM at the end of the phase. These two objectives serve as the primary propositions for the dissertation for consistency with the principles of the CNS. These objectives can be realized in the objective function in MPC with additional auxiliary constraints.

The objective function for the single support phase, J_{SSP}, is end-point control which follows the form of Eqn. 3.12 and can be expressed mathematically as:

$$\begin{aligned} J_{SSP}(x(k)) &= (f_{sl}(\theta_{1\ldots6,N}) - R_{sl})^2 \\ \text{subject to} &\begin{bmatrix} f_{VP}(\theta_{1\ldots6,k}) > 0 \\ f_{VP}(\theta_{1\ldots6,k+1}) > 0 \\ \vdots \\ f_{VP}(\theta_{1\ldots6,N-1}) > 0 \\ f_{VP}(\theta_{1\ldots6,N}) = 0 \end{bmatrix} \end{aligned} \tag{5.5}$$

where $\theta_{1\ldots6,k}$ represents the angular position of each of the joints (including ankle, knee, and hip) for both limbs at time step, k; f_{sl} represents the function which calculates the step length based on the angular position of each joint using the internal MPC model developed in Sec. 4.2; f_{VP} represents the function which calculates the vertical position of the swing foot based on the angular position of each joint; and R_{sl} represents the reference step length. The constraints, as shown in Eqn. 5.5, ensure the swing foot clears the ground during single support and contacts the ground at the end of single support. The purpose of this objective function is to minimize the error in step length between MPC prediction and the reference while ensuring that the swing foot contacts the ground only at the end of the phase. The control input constraints, as described in Sec. 5.2, also must be satisfied.

The objective function for the double support phase can be expressed as:

$$J_{DSP}(x(k)) = (f_{v_{COM}}(\dot{\theta}_{1\ldots6,N}) - R_{v_{COM}})^2 \tag{5.6}$$

where $\dot{\theta}_{1\ldots6,N}$ represents the angular velocity of each joint (including ankle, knee, and hip) for both limbs at the final time step N; $f_{V_{COM}}$ represents the function which calculates the velocity of the COM at the end of the double support based on the angular velocity of each joint using the internal model developed in Sec. 4.3; $R_{v_{COM}}$ represents the reference velocity of the COM. As previously discussed in Sec. 4.3, both feet in the internal MPC model are hinged to the ground; therefore constraints similar to Eqn. 5.5 are not required.

Eqn. 5.5 and 5.6 are the objective functions of the developed model and function similarly to the "brain" in the CNS. By minimizing these two functions, the joint moment control inputs can be generated and human gait can be simulated. Although the dynamics involved in the entire system are complicated, the core "brain", i.e., the objective functions, is surprisingly simple. Two simple criteria governing the operation of the entire system are also consistent with the assumption that the CNS utilizes simple criteria to control walking. Another advantage of this system is that future work can investigate alternative CNS principles by only changing the objective functions while leaving the remaining system unchanged.

5.4 Laguerre Functions as Control Inputs

With the objective functions determined, the control inputs can be generated. However, there is one potential problem with this system. Because MPC is a type of discrete control theory, the entire simulation needs to be divided into a finite number of time steps and MPC proceeds with each time step. If the moment at each joint at every time step is an individual design variable, the number of design variables for optimization is very large and impossible to manage in an optimization routine. For example, in the case where there are 47 time steps for single support simulation and 4 joint moments need to be optimized, the total number of design variables for the initial step of MPC optimization is $47 \times 4 = 188$. For the following kth time step, the total number of design variables to be optimized is $(47 - k + 1) \times 4 = 192 - 4k$ which is still a large number. To significantly reduce the number of design variables for the optimization, thereby reducing the computational burden and simulation run-time, Laguerre functions are used to parameterize the moment functions. Another advantage of using Laguerre Functions is that smoother moment function profiles can be generated, while if the control inputs are optimized for each time step there may be discontinuity in the generated control inputs.

5.4.1 Laguerre Functions

Laguerre functions are a group of time-domain functions which are solutions of Laguerre's Equation and mutually orthonormal. This dissertation uses Laguerre functions as a tool to parameterize the control inputs (i.e., the joint moments). The details of their derivation are beyond the scope of this dissertation. One can reference [42] for more details of the derivation.

The set of discrete-time Laguerre functions from initial time, 0, to final time step, N, can be expressed in a vector form as:

$$L = \begin{bmatrix} L(0) & L(1) & \cdots & L(k) & \cdots & L(N) \end{bmatrix} = \begin{bmatrix} l_1(0) & l_1(1) & \cdots & l_1(k) & \cdots & l_1(N) \\ l_2(0) & l_2(1) & \cdots & l_2(k) & \cdots & l_2(N) \\ \vdots & \vdots & & \ddots & & \vdots \\ l_M(0) & l_M(1) & \cdots & l_M(k) & \cdots & l_M(N) \end{bmatrix} \tag{5.7}$$

where M represents the number of Laguerre basis functions utilized; $l_i(k)$ represents the ith Laguerre function at the kth time step; $L(k)$ represents the vector form of values of all Laguerre functions at the kth time step; l_i represents the ith Laguerre basis function. All the Laguerre basis functions are functions which starts from the 0th to Nth time step. For each time step, the set of $L(k)$ satisfies the following difference equation:

$$L(k+1) = A_l L(k) \tag{5.8}$$

where matrix A_l is a $M \times M$ matrix and is a function of parameters a and $\beta = (1 - a^2)$:

$$A_l = \begin{bmatrix} a & 0 & 0 & \cdots & 0 & 0 \\ \beta & a & 0 & & 0 & 0 \\ -a\beta & \beta & a & & 0 & 0 \\ a^2\beta & -a\beta & \beta & \ddots & 0 & 0 \\ -a^3\beta & a^2\beta & -a\beta & \ddots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ (-1)^M a^{M-2}\beta & (-1)^{M-1}a^{M-3}\beta & (-1)^{M-2}a^{M-4}\beta & \cdots & \beta & a \end{bmatrix} \tag{5.9}$$

The initial condition, i.e., $L(0)$ is given by:

$$L(0) = \sqrt{\beta}\begin{bmatrix}1 & -a & a^2 & -a^3 & \cdots & (-1)^{M-1}a^{M-1}\end{bmatrix} \tag{5.10}$$

where a is an independent variable that is selected manually and has a direct impact on the shape of each individual Laguerre function. As an illustration of the impact of a, Fig. 5.2 ($a = 0.5$) and Fig. 5.3 ($a = 0.8$) show two groups of Laguerre functions which vary only in the values of a.

One critical feature of Laguerre functions is that they are mutually orthonormal. This orthonormality can be expressed as:

$$\sum_{k=0}^{N} l_i(k)l_j(k) = 0 \ \ for \ \ i \neq j \tag{5.11}$$

$$\sum_{k=0}^{N} l_i(k)l_j(k) = 1 \ \ for \ \ i = j \tag{5.12}$$

This orthonormality feature can be also explained as each Laguerre function, l_i, peaks at a different time step. For example, as shown in Fig. 5.2, l_1 peaks at the first time step, l_2 peaks from time steps 1 to 3, and l_3 peaks even later at time steps 4 to 6. This orthonormality feature allows Laguerre Functions to be used intensively in the area of system identification, where the discrete-time response of a dynamic system is represented by the combination of a group of Laguerre functions [42, 44].

The use of Laguerre Functions can greatly reduce the number of design variables, therefore saving computational power and reducing simulation time. To illustrate this reduction of design variables, consider an example with an optimal MPC control input profile (the solid line in Fig. 5.4) that needs to be generated. If

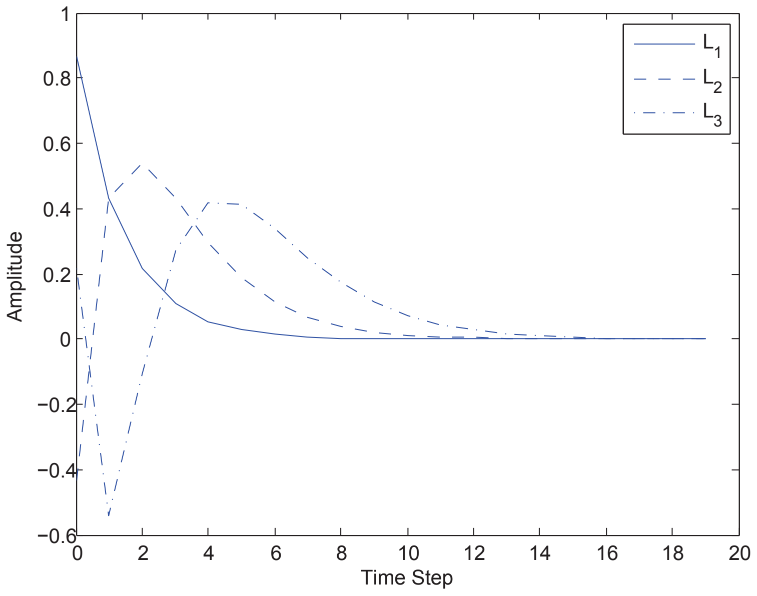

Figure 5.2: Laguerre Functions With $a = 0.5$

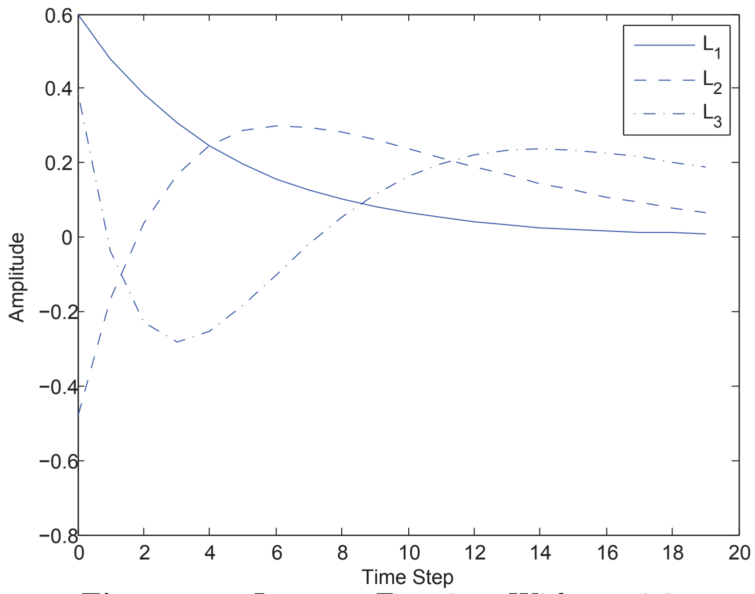

Figure 5.3: Laguerre Functions With $a = 0.8$

the original method is used, $\Delta u(k)$, for every time step needs to be optimized; the total number of design variables are 60. Even if the optimization sampling rate is changed so that a control input only needs to be optimized once every four time steps, the number of design variables are still 15. In contrast, if four Laguerre functions are used and the independent variable a is chosen to be 0.8, the optimized control input profile at time step k can be expressed as:

$$u(k) = c_1 l_1(k) + c_2 l_2(k) + c_3 l_3(k) + c_4 l_4(k) \tag{5.13}$$

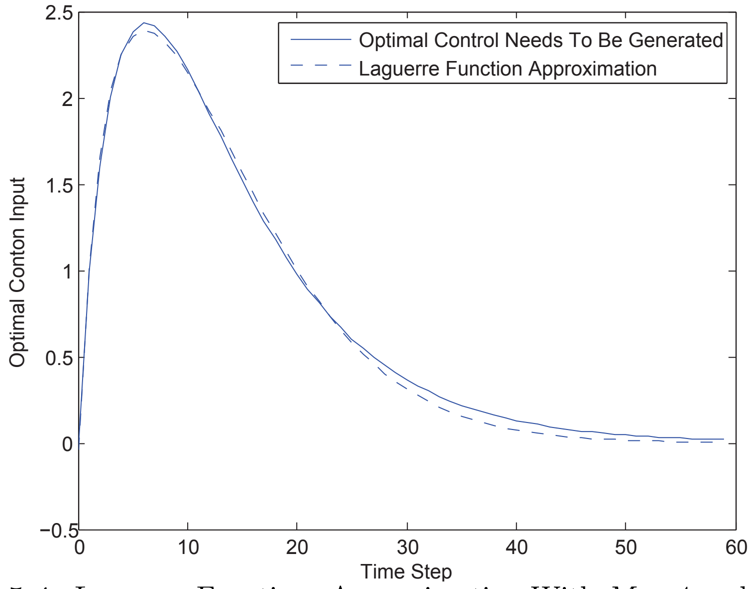

Figure 5.4: Laguerre Functions Approximation With $M = 4$ and $a = 0.8$

where only 4 coefficient parameters c_1, c_2, c_3, and c_4 need to be optimized. The optimization results using Laguerre function approximation is also shown in Fig. 5.4 (dashed line), nearly identical to the desired optimal control input. Therefore, the use of Laguerre functions to approximate the control input profile and the use of Laguerre function coefficient parameters as design variables significantly reduces the number of design variables while sacrificing performance only slightly. Note that each of the Laguerre coefficients, c_1 to c_4, still need to be optimized for every time step because the optimal control input may change due to discrepancies between the plant model and the internal MPC model and disturbances introduced during simulation.

5.4.2 Application to Joint Moments

To apply Laguerre functions to the simulation of human gait, each of the joint moment control inputs is decomposed into a combination of a group of Laguerre functions for both single support and double support phases. The values of M and a are determined by choosing the values that best approximate the experimental data of joint moments. The reason of doing this is the experimental joint moment data are known *a priori*. Therefore, if the Laguerre functions with a specific value of M and a are able to approximate the experimental data, the MPC controller should have the capability to generate able-bodied gait. After trial-and-error testing, it was decided that $M = 6$ and $a = 0.5$ reach the best balance that the specified Laguerre functions are able to generate various control input profiles while maintaining the simulation time within several hours. The number of design variables for the single support phase is 24: 6 Laguerre function coefficients for the stance ankle, swing ankle, knee, and hip, respectively. The number of design variables for the double support phase is also 24: 6 Laguerre function coefficients for both stance ankles and knees. Therefore, the task of MPC controller optimizing each joint moment control input for every time step is implemented by optimizing the Laguerre function coefficients for each time step.

With the form of control inputs determined, the structure of the MPC system is complete. However, MPC is not the only control algorithm used for the human gait simulation. The next section discusses the auxiliary PID controller which controls the HAT during the entire gait cycle as well as the stance knee during single support.

5.5 Auxiliary PID Control

In this dissertation, the main control algorithm approximating the CNS is MPC to achieve a certain step length for single support and a certain velocity of the COM for double support. In addition to MPC, the HAT is required to remain upright throughout the entire gait cycle and the stance knee remains extended during single support phase. The control philosophy on the HAT and stance knee differ in that no prediction needs to be made on future states and the only objective is to maintain the HAT and stance knee orientation at a constant reference value. The adjustment of the control input is based on the error between the current orientation and the reference, approximating feedback control instead of predictive control. PID control is a classic type of feedback control which determines control inputs based on past error from a constant reference. Therefore, PID control is selected as the auxiliary control algorithm as shown in the block diagram of the entire system in Fig. 5.1.

The detailed block diagrams of PID control on the HAT during the entire gait cycle and stance knee during single support are shown in Fig. 5.5 and . 5.6, respectively. The two PID controllers share the same structure, but different control

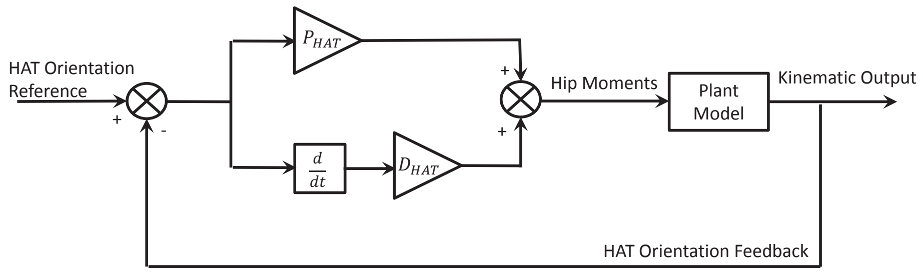

Figure 5.5: HAT PID Control Block Diagram

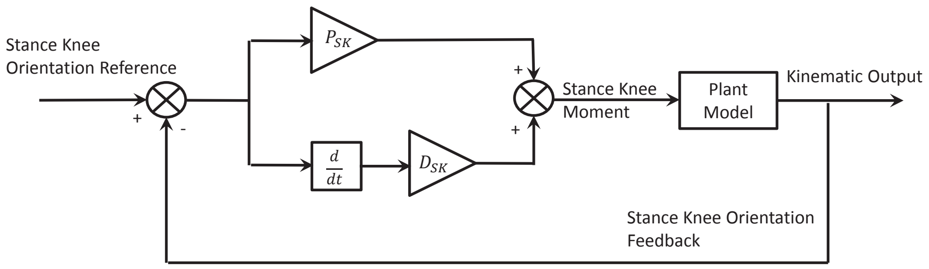

Figure 5.6: Stance Knee PID Control for Single Support Phase Block Diagram

references and PID gains. The control inputs are based on the error between the reference and feedback. Only proportional and derivative gains are used; the integral gain is 0. Human gait is an inherently dynamic process where no steady state exists, therefore integral gain is not necessary. The proportional and derivative gains are tuned by a trial and error, until their deviation from the references are less than 3 degree. The values of the proportional and derivative gains for both single support and double support simulations are shown in Tab. 6.2.

Table 5.1: The Value of the Proportional and Derivative Gains

		P Gain (Nm/rad)	D Gain (Nm/rad/s)
Single Support Phase	Stance Hip	2	0.2
	Stance Knee	1	0.1
Double Support Phase	Stance Hip	1	0.1
	Swing Hip	1	0.1

5.6 Platform to Realize the MPC Control System

As mentioned in Chap. 2, the human gait plant model is developed in the MATLAB/Simulink environment, specifically using the SimMechanics toolbox. It is beneficial to keep the MPC control system and the plant model to be on the same platform to make the two components combine seamlessly. In addition, MATLAB is a powerful computational platform. The MPC control system is also developed on the MATLAB/Simulink platform, as illustrated in Appendix. C.

5.7 Summary

In this chapter, the overall control algorithm of the human gait simulation system is explained; the critical MPC related parameters are determined; the objective function is developed with a Laguerre functions to reduce the number of design variables; PID control is selected as the auxiliary controller. With all the elements (Fig. 5.1) in this human gait simulation system developed and explained, the simulation can be performed. To verify the fidelity of the developed system, able-bodied human gait is simulated and the results are described in Chap. 6.

CHAPTER 6

Simulation of Able-Bodied Human Gait

The fidelity of the human gait model developed in first five chapters needs to be verified. In this chapter, the model is verified first by simulating gait at an able-bodied person's self selected walking speed (SSWS). The experimental data was acquired using a Vicon motion capture system at the Medical College of Wisconsin Department of Orthopaedic Surgery's Center for Motion Analysis. The fidelity of the model is addressed by comparing the kinematic and kinetic output from the simulation to the benchmark experimental data. The errors are quantified. These baseline results will then be expanded in Chap. 7 to verify the prediction capability of the model.

6.1 Method of Able-Bodied Human Gait Simulation at SSWS

To perform the simulation, some model parameters and MPC control references are required. The parameters include anthropometric, internal mechanical, and MPC control reference parameters, as summarized in Tab. 6.1

Table 6.1: Required Model Parameters and MPC Control References

Anthropometric Parameters	Segmental lengths, masses, moment of inertia with respect to segment COM, location of COM for each of segment (feet, shanks, and thighs for both legs, and HAT)
Internal Mechanical Parameters of the Plant	Internal spring stiffness and damping coefficient for each joint (ankles, knees, and hips) Internal spring stiffness and damping coefficient (horizontal and vertical) for both feet
MPC Control References	Step length and velocity of the COM

For the anthropometric parameters, the segment lengths were directly recorded on the human subjects. The segmental mass, mass moment of inertia, and location of COM are calculated using empirical anthropometric equations (Winter [40]). The internal mechanical parameters are obtained by optimizing

(Chap. 2). The same internal mechanical parameters are used for the internal MPC models for consistency between the plant model and internal MPC models. During the human subject testing, reflective markers are placed on the subject via a modified Helen-Hayes marker set to obtain kinematic testing data with Vicon MX cameras capturing the movement of the markers. The experimental step length and velocity of COM are calculated based on the captured kinematic data. Therefore, the calculated experimental step length and velocity of COM provide the MPC control references.

Force plates are embedded along the walkway to measure the GRF. The kinetic data including moments at each joint, power consumption, and other data are calculated using the standard inverse dynamic model (Vicon Plug-In Gait Model) [45]. These kinetic data serves as another benchmark data to evaluate the fidelity of the developed model.

6.2 Simulation Results of SSWS Gait

With values of the required parameters in Tab. 6.1 specified, the simulation of gait for an able-bodied subject at SSWS is performed. The performance of the model is assessed at three levels: (1) ability of the model to achieve the control references; (2) ability of the model to achieve kinematic experimental data; (3) ability of the model to achieve kinetic experimental data.

Note the accuracy of experimental kinematic and kinetic data differ. The kinematic data, i.e., the saggital plane angular position of each joint during one gait cycle, was directly captured using the Vicon cameras. In contrast the kinetic joint moment data are indirectly calculated using the Vicon Plug-In inverse dynamic gait model [45] based on the measured GRF from the foot plates. Therefore the kinematic experimental data have relatively high accuracy and is used as the primary benchmark data. Whereas the kinetic joint moment experimental data has lower accuracy and used as secondary benchmark data.

The MPC control reference for SSWS and the actual output of the model are shown in Tab. 6.2. The MPC controller achieved the control reference very accurately, with error smaller than 0.5% for step length during single support and velocity of COM for double support.

Table 6.2: Comparison of Model Output and Control Reference for SSWS

	MPC Control Reference	Actual Model Output	Percentage Error
Step Length (m) for Single Support Phase	0.7345	0.7372	0.37%
Velocity at COM (m/s) for Double Support Phase	1.3854	1.3788	0.48%

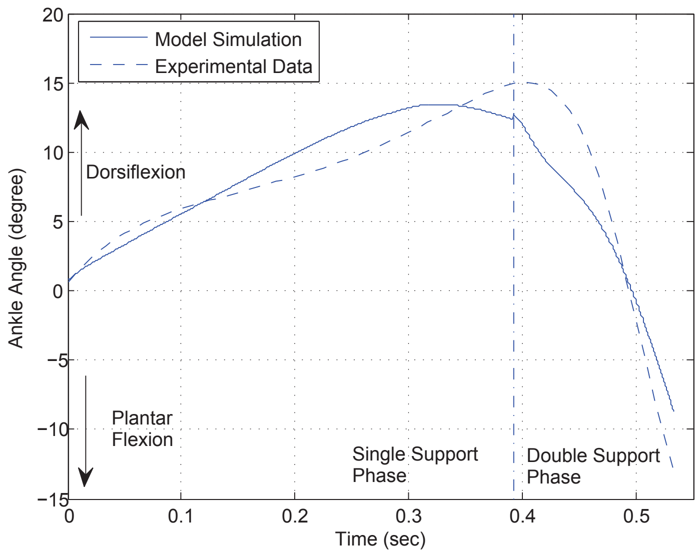

Figure 6.1: Sagittal Plane Ankle Angle of Stance Leg - Simulation vs Experimental Data

The simulation for one full step is performed in the sequence of single support and then double support. The state at the end of the single support phase

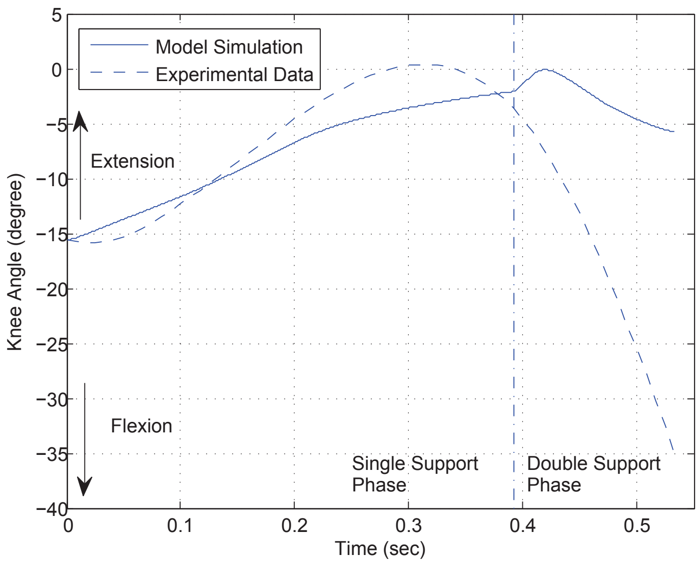

Figure 6.2: Sagittal Plane Knee Angle of Stance Leg - Simulation vs Experimental Data

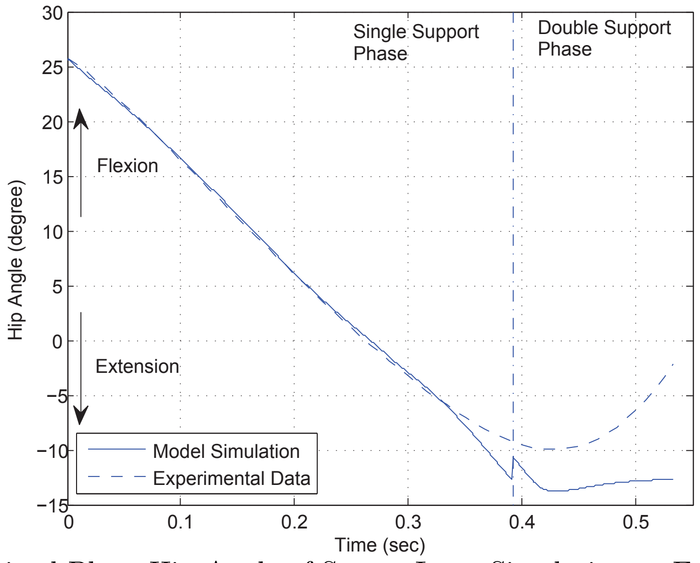

Figure 6.3: Sagittal Plane Hip Angle of Stance Leg - Simulation vs Experimental Data

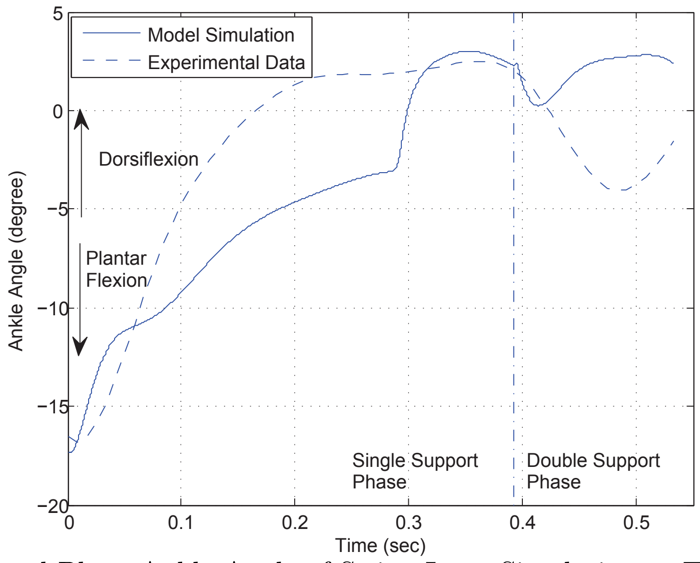

Figure 6.4: Sagittal Plane Ankle Angle of Swing Leg - Simulation vs Experimental Data

is used as the initial state for the double support phase, therefore the two phases are combined seamlessly. The kinematic simulation results for one full step, single support followed by double support, are shown from Fig. 6.1 to . 6.6. The experimental kinematic data are plotted as dashed line for comparison purposes. Each figure illustrates the sagittal plane angular position of the ankles, knees, and hips for both sides. To quantify errors between the simulation results and experimental data, the root mean square error (RMSE) for the full gait step is calculated and listed in Tab. 6.3.

Similarly, the kinetic joint moment simulation results for one step are

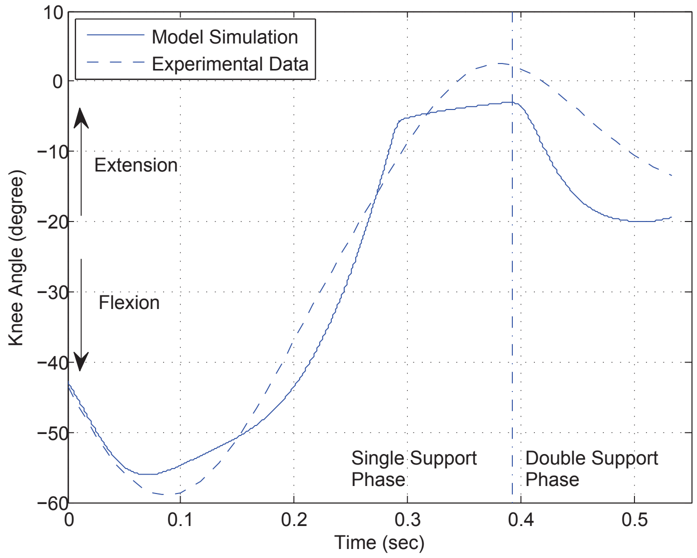

Figure 6.5: Sagittal Plane Knee Angle of Swing Leg - Simulation vs Experimental Data

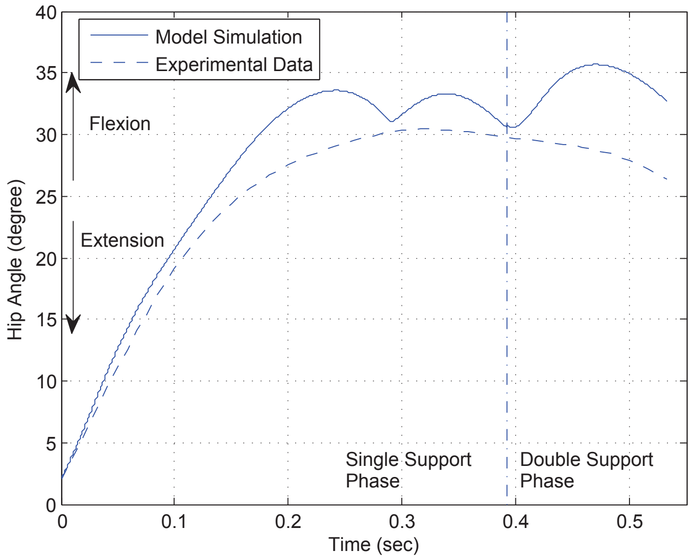

Figure 6.6: Sagittal Plane Hip Angle of Swing Leg - Simulation vs Experimental Data

illustrated as shown from Fig. 6.7 to . 6.12. The experimental data are plotted as dashed line for comparison purposes. To quantify the errors between the simulation results and experimental data, the RMSE for the step, single support, and double support phases is calculated and listed in Tab. 6.4.

6.3 Discussion of Simulation Results of SSWS Gait

Based on the model simulation results and experimental data, the MPC control references are achieved very accurately (smaller than 0.5% error). The angular position of each joint is consistent with the experimental data, with the

Table 6.3: RMSD of Angular Position of Each Joint Between Simulation and Experimental Data

	RMSE Full Step (°)	Single Support (°)	Double Support (°)
Stance Ankle	2.30	1.35	3.89
Stance Knee	8.94	2.24	17.07
Stance Hip	3.014	0.82	5.73
Swing Ankle	4.214	3.99	4.79
Swing Knee	6.063	3.93	9.84
Swing Hip	3.820	2.77	5.84

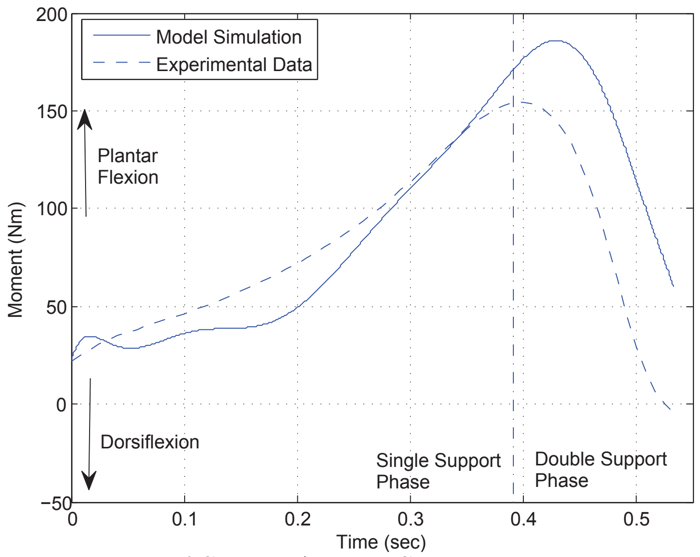

Figure 6.7: Moment of Stance Ankle - Simulation vs Experimental Data

largest errors less than 5° except the stance and swing knee during double support (Tab. 6.3). The joint moment model outputs, however, are less consistent with errors between 28% and 110%.

The developed model is first and foremost a predictive model. The purpose of the control system is to drive the plant model to achieve the control references

Table 6.4: RMSE of Moment of Each Joint Between Simulation and Experimental Data

	Full Step (Nm)	Single Support (Nm)	Double Support (Nm)	Average Percentage Error (%)
Stance Ankle	34.74	12.89	64.40	27.81
Stance Knee	21.49	8.47	39.55	65.11
Stance Hip	18.83	11.12	31.73	48.69
Swing Ankle	3.56	2.92	4.95	83.24
Swing Knee	25.11	13.37	43.66	109.93
Swing Hip	20.41	2.86	39.63	38.86

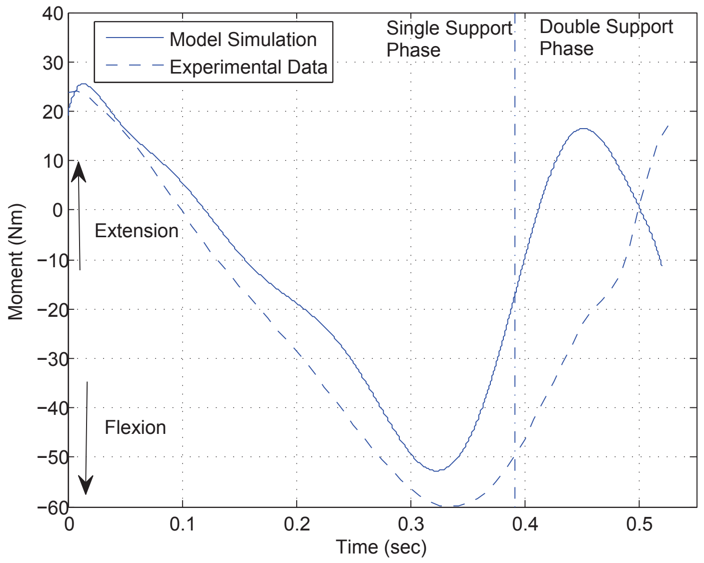

Figure 6.8: Moment of Stance Knee - Simulation vs Experimental Data

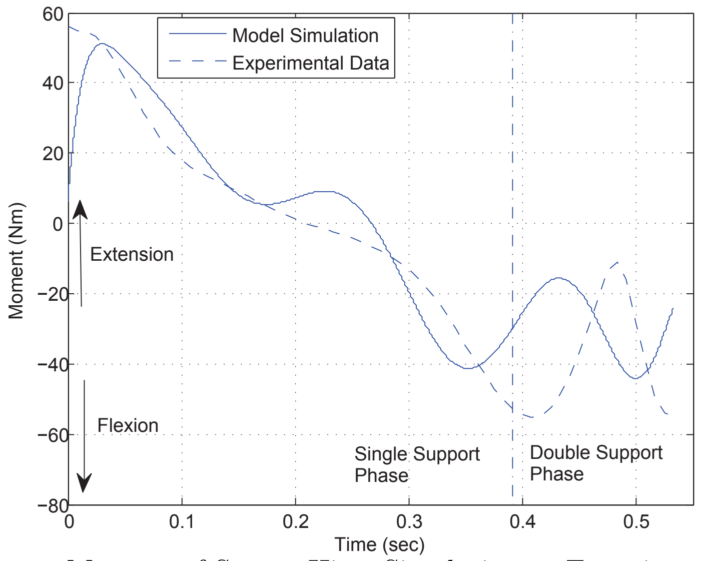

Figure 6.9: Moment of Stance Hip - Simulation vs Experimental Data

regardless of the kinematic trajectories or the kinetic moments. Therefore it is expected that the model will achieve the control reference accurately while the kinematic and kinetic output can deviate from the experimental data. However, the model still predicted the sagittal plane kinematics fairly well with typical joint angle errors less than 5%.

In the author's opinion, the kinetic joint moment output of the model deviates from the experimental data because of three reasons. The first reason lies in the difference between the dynamics of the gait plant model utilized in this dissertation and the dynamics of real human gait. The second reason lies in that the

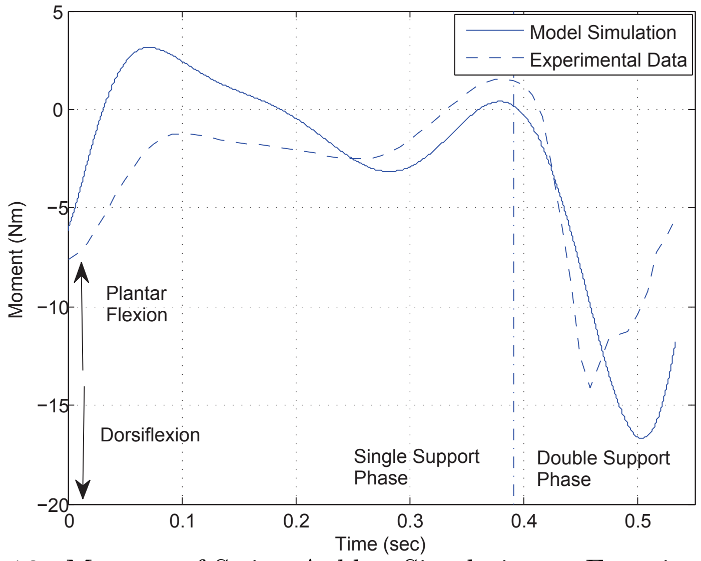

Figure 6.10: Moment of Swing Ankle - Simulation vs Experimental Data

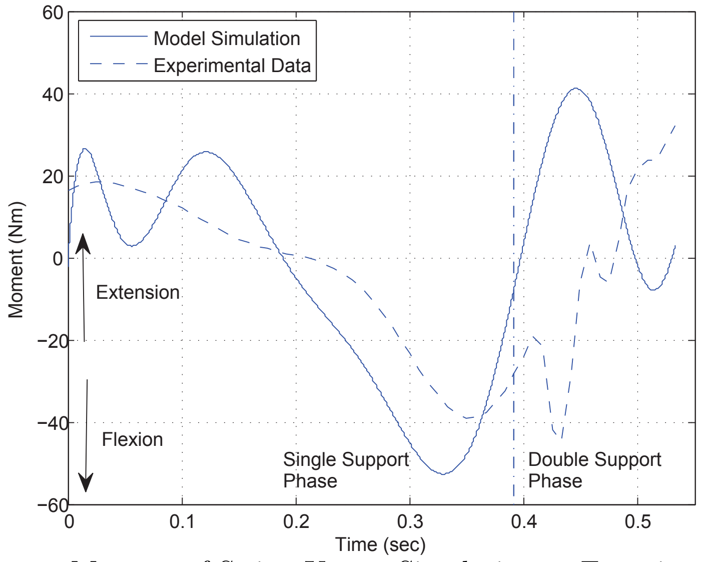

Figure 6.11: Moment of Swing Knee - Simulation vs Experimental Data

experimental kinetic data (i.e., the joint moments) is indirect data calculated based on Plug-In Gait Model. Therefore, the experimental data does not have high fidelity. The third reason is that the objective function utilized by the model could be off from the the realistic objective function used by the CNS. These three reasons combined together cause the kinetic model output to deviate from the experimental data.

As shown in Fig. 6.2, the model kinematic output of the stance knee greatly deviates from experimental data during the double stance phase where the stance knee remains extended and does not flex in preparation for swing. During double

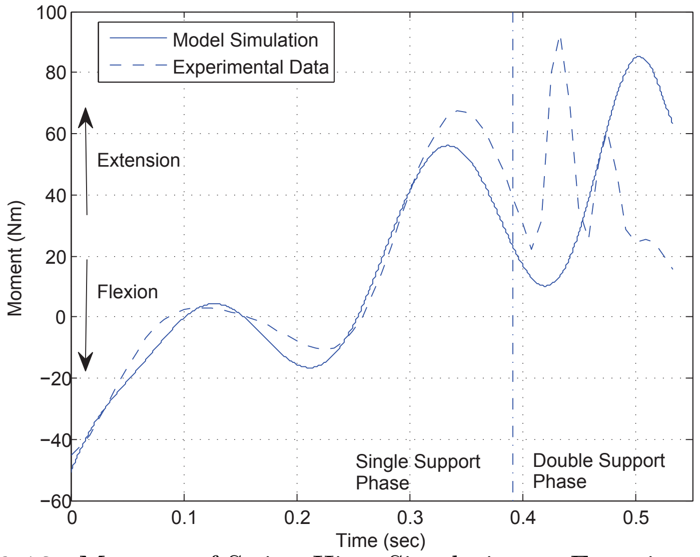

Figure 6.12: Moment of Swing Hip - Simulation vs Experimental Data

support, extension of trailing knee increases the speed of the COM while knee flexion decreases speed [46]. Therefore in contrast to the experimental data, the stance knee in the model extends to control the reference speed of the COM. Stance knee flexion would be unable to achieve the 1.3854 m/s reference speed. This deviation of the stance knee leads to the argument again that the developed model is first and foremost a predictive model which does not know the kinematic trajectory *a priori*.

As shown in Fig. 6.5, the kinematic output of the swing knee is consistent with the experimental data until approximately 0.3 sec when the modeled swing knee fails to further extend. This deviation is attributed to the range of motion (ROM) limit in the plant model as described in Chap. 2. When the stance knee joint approaches the ROM limits the stiffness and damping coefficient of the internal spring and damper greatly increase to enforce the ROM constraints. However, it is interesting to notice that the experimental data does move beyond 0°.

As illustrated in Fig. 6.7 to . 6.12, at the heel contact (i.e., transition from single support to double support), the model output moment suddenly changes, contrary to the experimental data. This deviation is because the momentum of each segment suddenly changes due to heel strike. Therefore the control inputs generated by the model must change dramatically to achieve the control reference. This sudden change is again contrary to the experimental data. Regardless of these phase transition errors, the overall joint moment outputs still demonstrate the similar trends as the experimental data.

6.4 Summary

In this chapter, the fidelity of the model is tested for gait at comfortable speed. It is shown that the model is able to achieve the control reference accurately, the kinematic output of the model is consistent with the experimental data, and the kinetic output of the model is able to follow the same trend as the experimental data. In the next chapter, the predict capability of the model is tested first by predicting human gait at variable speed. Then, the model is further tested by predicting pathological gait with unilateral passive ankle joint.

CHAPTER 7

Simulation of Variable Speed and Pathological Gait

As stated in Chap. 1, the ultimate purpose of this dissertation is not only to simulate able-bodied gait but to also build a human gait model with prediction capability. Therefore human gait related virtual testing can be performed to expedite P&O equipment design and development, reduce cost, and minimize the risks involved with human subject testing.

To show that developed model is able to **qualitatively** predict various types of human gait, the model is tested under three different conditions. For the first two conditions, the model is used to predict fast and slow walking over level ground. Kinematic and kinetic data details in the literature [47–49] show that fast and slow walking vary with respect to SSWS. These simulation results show that the developed model is able to qualitatively predict these patterns without *a priori* experimental kinematics and kinetics knowledge.

In addition to the simulation of various walking speed for able-bodied individuals, amputee gait with a unilateral passive prosthetic ankle joint is also simulated. Unilateral transtibial amputation is one of the most common amputations performed in the U.S. Various literature [50–52] study the difference in gait patterns between amputated gait using passive transtibial prostheses and able-bodied gait. Model simulation results are compared to the literature. The results show that the model is able to predict the walking patterns of a unilateral trantibial amputee with a passive prosthesis.

7.1 Simulation of Fast and Slow Walking for Able-Bodied Individuals

7.1.1 Method

To perform fast and slow speed walking simulations for an able-bodied subject, model parameters and MPC control references need to be determined. The purpose of the simulations of fast and slow walking is to further verify the capability of the model to qualitatively predict kinematic and kinetic performance; the same model parameters from Chap. 6 are used to contrast these results with the subject-specific SSWS simulation. The only model inputs that need to be revised are the MPC control references, step length and velocity of the COM. The fact that very few parameters are required in order to modify the simulation for different gaits implies an advantage of the developed model than biomechanics gait model described in Chap. 2 where usually hundreds of parameters need to be determined.

Fast and slow speed walking trials were not conducted for the able-bodied subject test in Chap. 6. However, a linear relationship has been identified between step length and walking speed [53]. The MPC control references are based on this linear relationship as described in Tab. 7.1.

Table 7.1: MPC Control Reference for Fast and Slow Speed Simulation References

	Step Length (m)	Velocity at COM (m/s)
Fast Speed Walking	0.813	1.6
Slow Speed Walking	0.666	1.2
SSWS	0.7345	1.3854

The model is verified for these parameters from three perspectives. First, the simulated step length and velocity of the COM output from the simulation are contrasted with the desired control references. The kinematic outputs from the simulation of the fast and slow trials are compared to the SSWS simulation in Chap. 6. Finally the kinetic outputs from the simulation are compared to the SSWS simulation.

7.1.2 Literature

Kinematic and kinetic data for self-selected, fast, and slow walking trials reported in the literature are reviewed. These reported trends are then used to qualitatively verify the model. The effects of walking speed in joints kinematics and kinetics for able-bodied individuals in [47, 54] indicate that:

1. The kinematic trajectories of the hip, knee, and ankle do not vary with walking speed. Only minor increases in peak knee flexion during loading response are observed as the gait speed increases.

2. The peak kinetic joint moments of the hip, knee, and ankle are dependent on gait speed [47]. Both the peak extension and flexion moment of the hip during single and double support increases with gait speed.

3. The peak knee flexion moment during loading response (early double support) and pre-swing (late double support) increases as gait speed increases [47]. During swing, the peak knee extension moment increases with gait speed.

4. Minor increases in peak ankle dorsiflexion moment with gait speed are observed [47].

The results of the slow and fast walking simulations are compared to the SSWS simulation in the next section to verify the ability of the developed model to capture these kinematic and kinetic dependencies on gait speed.

7.1.3 Simulation Results

The results of the simulation of fast and slow walking are again evaluated from three perspectives. First, the ability of the model to achieve the control reference is evaluated. The kinematic and kinetic model output of the fast and slow gait simulations are contrasted with the SSWS simulation results to assess the ability of the model to qualitatively predict the variations in kinematic and kinetic trajectories with speed, as reported in the literature.

Table 7.2: Model Output Compared to Control Reference for Fast Speed Gait

	MPC Control Reference	Actual Model Output	Percentage Error
Step Length (m)	0.8134	0.8135	0.012%
Velocity at COM (m/s)	1.6000	1.6026	0.16%

Table 7.3: Actual Model Output Compared to Control Reference for Slow Speed Gait

	MPC Control Reference	Actual Model Output	Percentage Error
Step Length (m)	0.6664	0.6663	0.015%
Velocity at COM (m/s)	1.2000	1.1986	0.12%

The comparison of the MPC control reference and the model output for fast and slow gait simulations are shown in Tab. 7.2 and . 7.3 respectively. The model

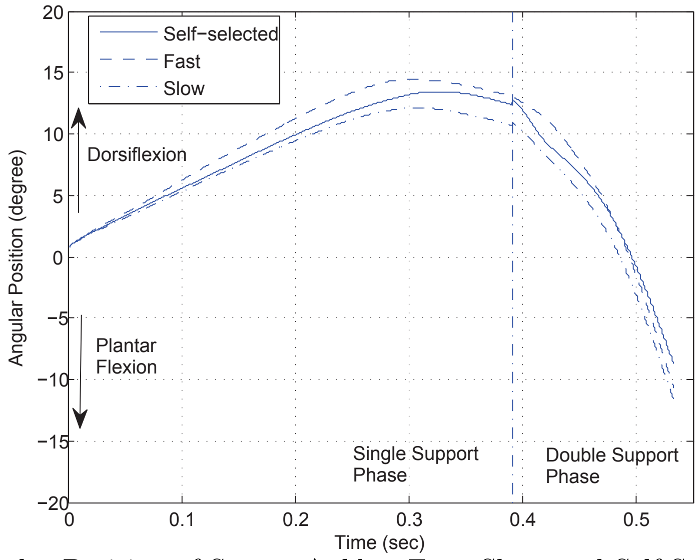

Figure 7.1: Angular Position of Stance Ankle - Fast, Slow, and Self-Selected Speed Simulation

achieved the control reference for both speed and COM velocity with errors smaller than 0.2%. This indicates the developed controller is able to generate appropriate joint moment inputs to drive the plant model to achieve the control references.

The kinematic simulation results for one step from single support to double support, for fast, slow, and SSWS are shown from Fig. 7.1 to . 7.6. In each figure, the self-selected speed simulation results are plotted as solid lines. The fast speed simulation results are plotted as dashed lines and the slow speed simulation results are plotted as a combination of dashed and dotted lines. The differences in kinematic and kinematic simulation output between speed are not quantified, as experimental data are only available for the specific subject.

Similarly, the kinetic joint moment simulation results for one step are shown in Fig. 7.7 to . 7.12. In each of the figure, the SSWS simulation results are plotted as solid lines. The fast speed simulation results are plotted as dashed lines and the slow speed simulation results are plotted as a combination of dashed and dotted lines. The differences in gait moments with the gait speed are not quantified because of the same reason.

7.1.4 Discussion

Based on the simulation results illustrated in the previous section, the following points can be summarized:

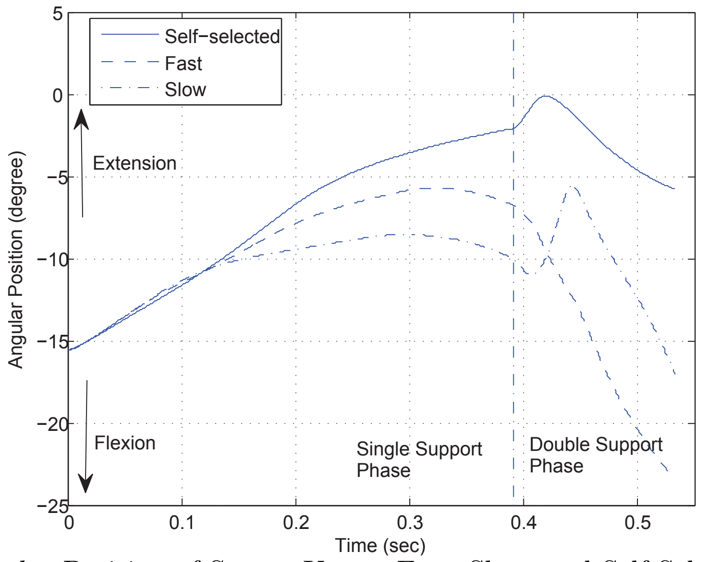

Figure 7.2: Angular Position of Stance Knee - Fast, Slow, and Self-Selected Speed Simulation

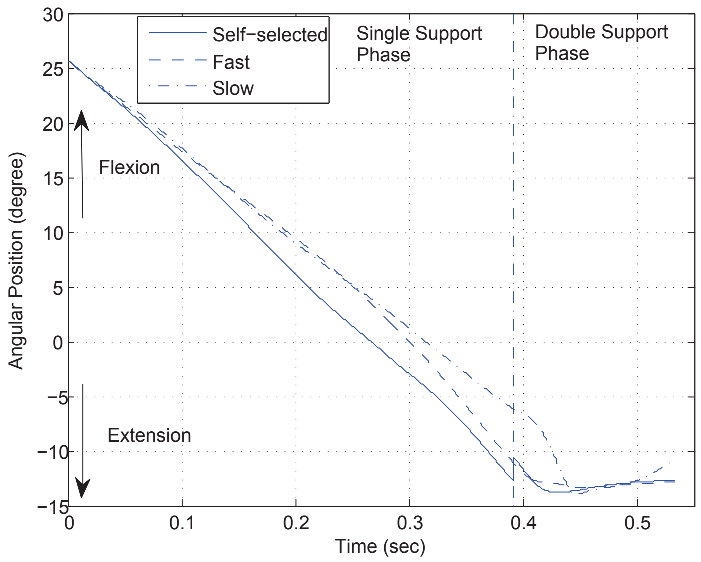

Figure 7.3: Angular Position of Stance Hip - Fast, Slow, and Self-Selected Speed Simulation

1. From control system performance perspective the developed model achieved the MPC control reference very well, with the errors in step length and the COM velocity smaller than 0.2%. Therefore it can be concluded the controller made the appropriate adjustment on joint moments control input to achieve difference control reference.

2. From kinematics perspective, it can be seen from Fig. 7.1, Fig. 7.3, Fig. 7.4, and Fig. 7.6 that, in the simulation, the ankle and hip kinematic output show

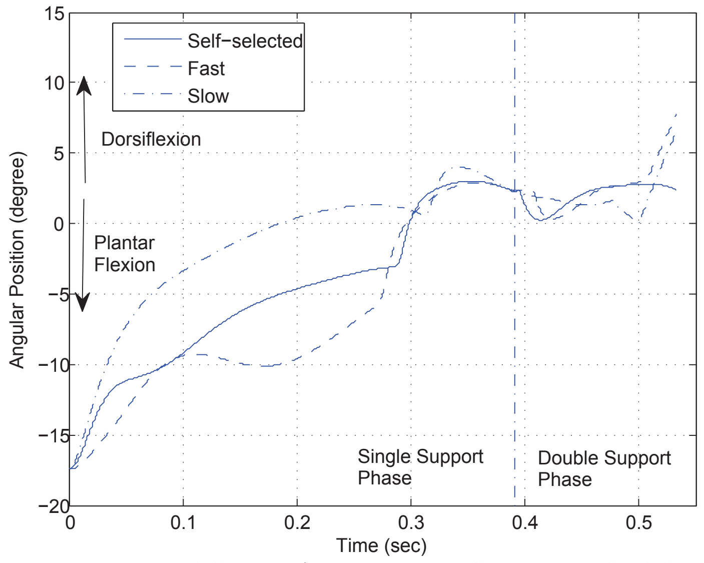

Figure 7.4: Angular Position of Swing Ankle - Fast, Slow, and Self-Selected Speed Simulation

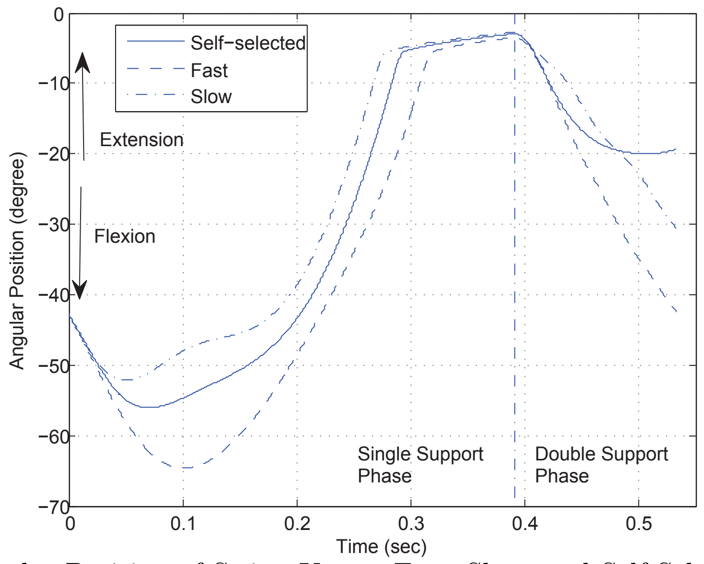

Figure 7.5: Angular Position of Swing Knee - Fast, Slow, and Self-Selected Speed Simulation

no significant difference between the three various gait speed. This is consistent with the literature where both [47] and [54] stated the there exists extremely similar experimental kinematic trajectory at hip and ankle between various gait speeds.

3. However, for the kinematic simulation output at the knee joint, it can be seen from Fig. 7.2 that the stance knee has more flexion for fast and slow speed than self-selected speed. From Fig. 7.5, it can be seen that as the gait speed

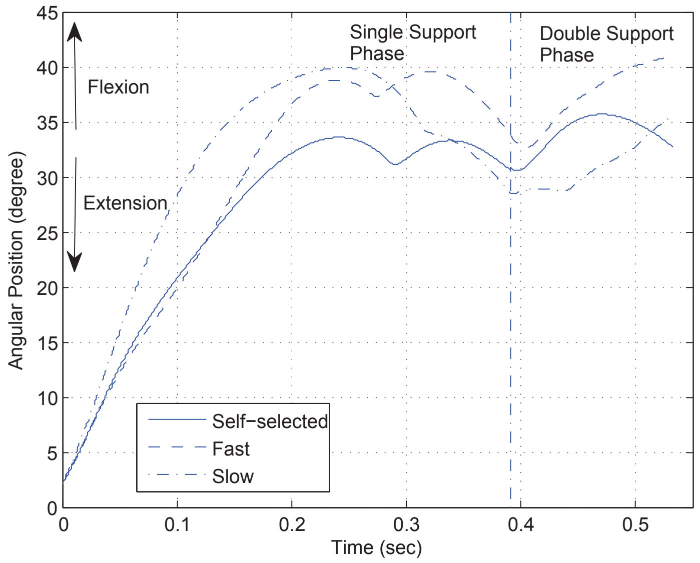

Figure 7.6: Angular Position of Swing Hip - Fast, Slow, and Self-Selected Speed Simulation

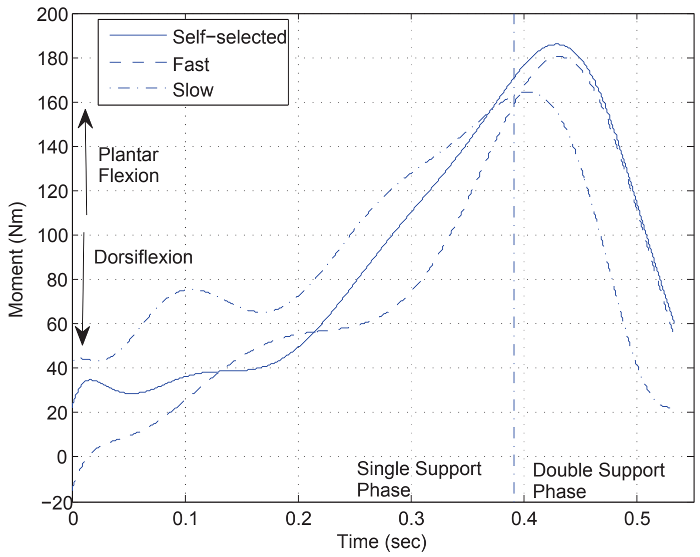

Figure 7.7: Moment of Stance Ankle - Fast, Slow, and Self-Selected Speed Simulation

increases, the knee flexion during initial swing and loading response increases correspondingly. This swing knee kinematic variation was reported by Winter [54]. This means the developed model successfully predicts that the swing knee flexion will increase as the gait speed increases.

4. From kinetics perspective, Lelas et al. [47] reported a rule of thumb that the peak joint moments in knee and hip in both extension and flexion direction increase with the increase in gait speed. This characteristics is reflected from Fig. 7.8, Fig. 7.9, Fig. 7.11, and Fig. 7.12 that the slow gait speed knee and hip joint moments for both extension and flexion direction in simulation are

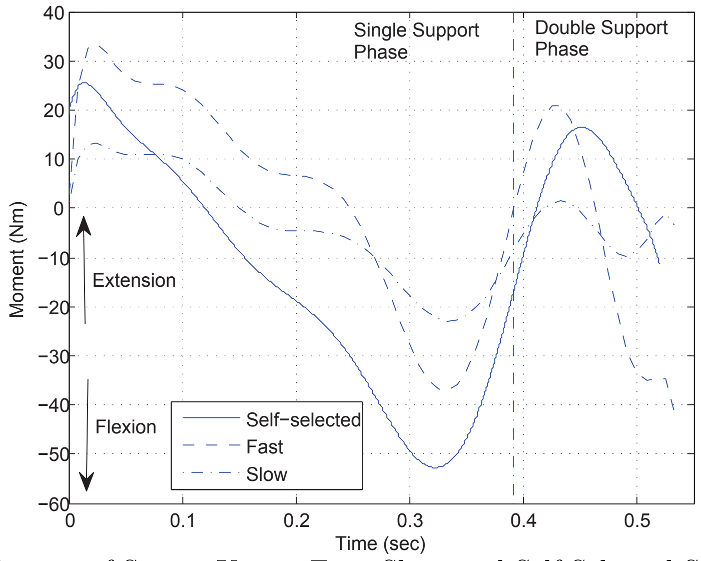

Figure 7.8: Moment of Stance Knee - Fast, Slow, and Self-Selected Speed Simulation

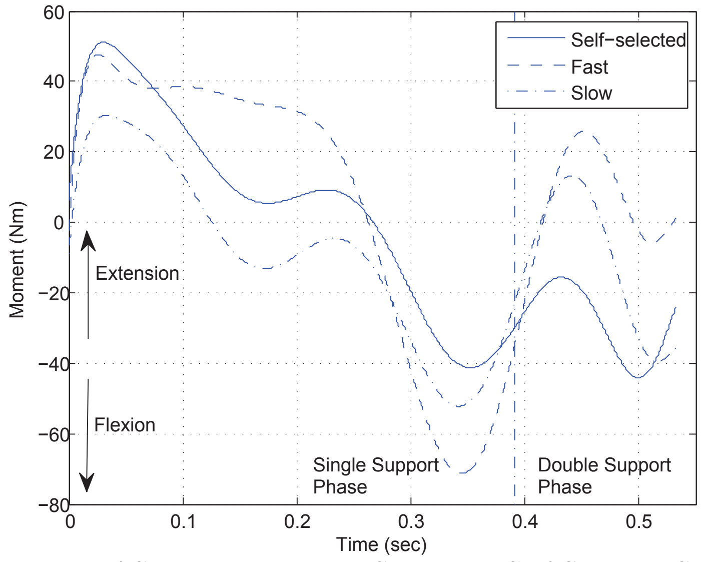

Figure 7.9: Moment of Stance Hip - Fast, Slow, and Self-Selected Speed Simulation

generally smaller than self-selected and fast gait speed.

In conclusion, the developed model successfully achieve the various speed simulation by hitting the control reference very accurately without *a priori* knowledge of the joint moments. The kinematic output of the model successfully predicted the general similarity in joint trajectories between fast, slow, and self-selected gait speed. The model also successfully predicted the increase in knee joint flexion as the gait speed increases. The differences in kinetic joint moment outputs are not as clear as the kinematic outputs but the model is still able to predict the general trend of increase in knee and hip joint moments as the gait

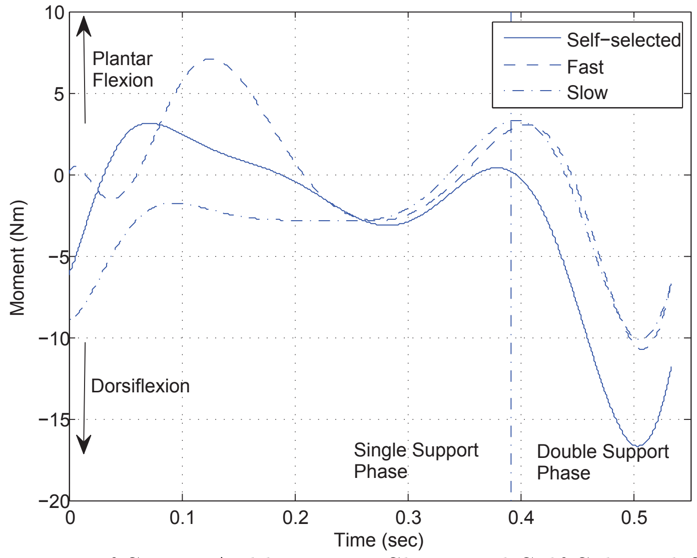

Figure 7.10: Moment of Swing Ankle - Fast, Slow, and Self-Selected Speed Simulation

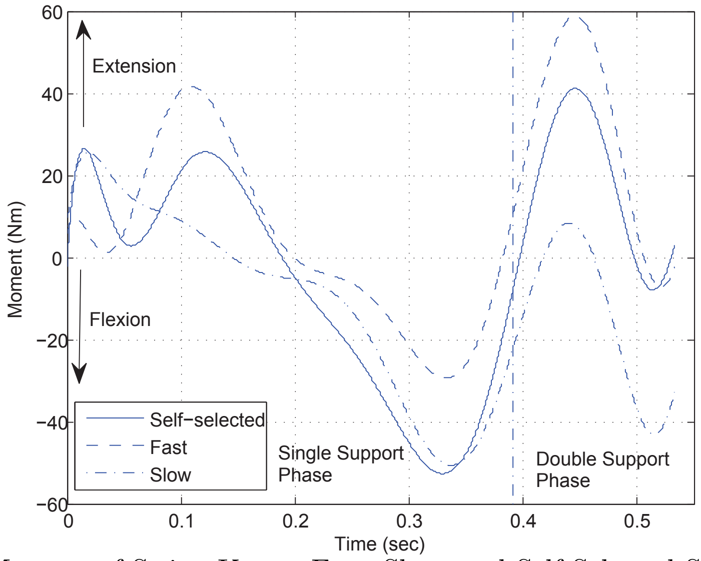

Figure 7.11: Moment of Swing Knee - Fast, Slow, and Self-Selected Speed Simulation

speed increases. Therefore, the simulation results shown in this section show that the developed model has prediction capability to predict the kinematic and kinetic characteristics of various gait speeds.

7.2 Simulation of Amputee Gait

7.2.1 Method and Literature

The simulation of amputee gait differs from the simulation of gait for an able-bodied subject in Chap. 6. The amputee subject's prosthesis incorporates a passive prosthetic ankle joint that cannot generate power. This section uses the

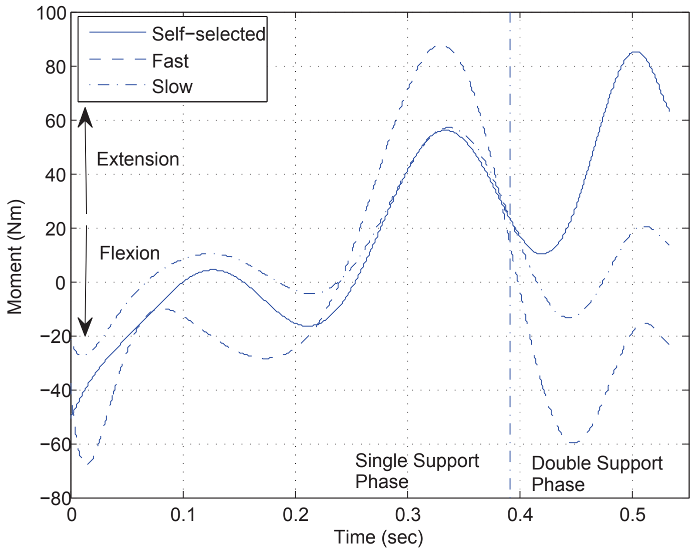

Figure 7.12: Moment of Swing Hip - Fast, Slow, and Self-Selected Speed Simulation

developed model to simulate this prosthesis and predict these gait.

A unilateral passive prosthetic ankle is simulated by setting the moment control input for the prosthetic ankle to zero. This means no active moment can be generated at the ankle; the only moment source in the ankle is from the internal elastic spring and damping due to the angular motion of the ankle. For the MPC control reference, no literature is available establishing the possible step length and velocity walking target for the MPC controller for a transtibial amputee. Literature [46] shows that the CNS works differently from able-bodied gait. How the control reference for step length and velocity changed for amputee gait needs to be investigated. However in this dissertation there will be no change in the target step length and walking velocity for a transtibial amputee. Therefore only the performance of the control system, i.e. how well the model achieves the control reference, is shown in this dissertation.

The performance of two types of unilateral prosthetic ankles are simulated. First, the prosthetic ankle is simulated as a pure passive revolute joint. Then the prosthetic ankle is simulated as a passive revolute joint with a torsional spring on the joint which is consistent with most of the designed passive prostheses. The value of the spring stiffness is obtained from a prototype prosthesis designed by Bergelin et al. [55]. The model is expected to predict the prosthetic ankle with torsional spring achieve control reference better than pure passive revolute joint prosthesis.

7.2.2 Simulation Results

The MPC control reference compared to the actual model output with the pure passive ankle on the stance and swing side respectively for one full stride is shown in Tab. 7.4 and 7.5. It can be noticed that each of the model output achieves the MPC control reference accurately except for one: The target velocity of the COM with the passive ankle on the stance side. With the target velocity being 1.3854m/s, the model only achieves 0.8987m/s which is 42.62% less. The reason of this discrepancy is analyzed in the Discussion subsection.

Table 7.4: Actual Model Output Compared to Control Reference of the Prosthetic Limb for Pure Passive Prosthesis

	MPC Control Reference	Actual Model Output	Percentage Error
Step Length (m)	0.7345	0.7373	0.38%
Velocity of the COM (m/s)	1.3854	0.8987	-42.62%

Table 7.5: Actual Model Output Compared to Control Reference of the Intact Limb for Pure Passive Prosthesis

	MPC Control Reference	Actual Model Output	Percentage Error
Step Length (m)	0.7345	0.7344	-0.014%
Velocity of the COM (m/s)	1.3854	1.4025	1.23%

Table 7.6: Actual Model Output Compared to Control Reference of the Passive Prosthetic Limb for Prosthesis with Torsional Spring

	MPC Control Reference	Actual Model Output	Percentage Error
Step Length (m)	0.7345	0.7344	-0.014%
Velocity of the COM (m/s)	1.3854	1.0955	-23.37%

The MPC control reference compared to the actual model output with the prosthesis with a torsional spring is shown in Tab. 7.6 and 7.7. While other control references were achieved, the velocity at COM achieved when the prosthesis is on the stance side is 1.0955m/s, which is -23.37% less than the control reference.

Table 7.7: Actual Model Output Compared to Control Reference of the Intact Limb for Prosthesis with Torsional Spring

	MPC Control Reference	Actual Model Output	Percentage Error
Step Length (m)	0.7345	0.7357	0.16%
Velocity of the COM (m/s)	1.3854	1.3394	-3.38%

7.2.3 Discussion

An important function of the ankle joint is to propel the COM of the body and swing leg to a certain velocity during late stance and pre-swing phase in order to get ready for the swing. Because of this demand to propel the whole body, for able-bodied person's gait, the moment the ankle joint normally achieves is more than 150Nm. Therefore, one can assume with the transtibial amputee's ankle joint being passive, the moment cannot achieve the able-bodied level and provide enough propulsion to make the body achieve the required speed before swing phase. This trend is successfully predicted by the developed model as shown in Tab. 7.4 and 7.6

Because the pure passive prosthetic ankle is not able to provide any active moment during late stance and pre-swing, the simulation predicts that the COM cannot achieve the target velocity of 1.3854m/s with the actual model output being only 0.8987m/s as shown in Tab. 7.4. The other MPC control reference is achieved accurately by the model as shown in Tab. 7.4 and Tab. 7.5 since the moments from the prosthetic ankle is not of critical importance in achieving that target. With the prosthesis having a torsional spring on the prosthetic ankle joint, the torsional spring is able to absorb energy during early and mid stance and release the stored energy during push-off [55]. As shown in Tab. 7.6, this trend is predicted by the model that the prosthetic ankle with a torsional spring achieves closer velocity model output to control reference than the pure passive prosthetic ankle, while the intact limb is not affected.

The step length and velocity at COM control references used in this chapter are the same as in Chap. 6 for comparison purpose. However it is known that the CNS use a different control strategy for pathological gait [46]. How the MPC control references changes with the CNS using a different control strategy is a potential direction for research. The MPC control references can be conveniently changed in the developed model to test the proposed CNS control strategy.

In conclusion, the developed model successfully predicts that without active power from the ankle, the amputee human gait is not able to achieve the same level of velocity at push-off as able-bodied gait. The model also successfully predicts that with help of a designed torsional spring, the amputee gait achieved a closer performance to able-bodied gait though still not the same.

CHAPTER 8

Conclusion and Future Work

In this dissertation, a human gait model with prediction capability is developed. The developed model includes a human gait plant model to function as the control target and a control system, a combination of classical PID control and MPC control, to simulate the CNS. The performance of the developed model is verified by performing simulation under three conditions: Self-selected speed able-bodied gait, various speed able-bodied gait, and amputee gait with a passive prosthesis. The simulation results showed that the self-selected speed able-bodied gait simulation output is close to the experimental data and the developed model has the ability to qualitatively predict the characteristics of kinematics and kinetics for various cadences.

8.1 Contributions

The major contributions of this dissertation are:

1. A human gait plant model is developed which has only seven segments and nine DOFs but is able to represent the forward dynamics of human gait. Therefore the developed plant model can be used as the control target for the system.

2. The CNS controls human gait based on a combination of prediction and feedback. A novel control system is developed that combines MPC and classical PID control algorithms to simulate the CNS of the able-bodied people.

3. By controlling only two critical gait related parameters, step length and velocity of the COM at push-off, the able-bodied gait can be simulated.

4. A simple internal MPC model is used for prediction and can be represented by a simple pendulum model.

5. The human gait model successfully simulated able-bodied gait with the kinematics percentage error less than 5% from experimental data. The model was also able to predict the characteristics of various speeds without those characteristics known *a priori*.

8.2 Model Limitations

The simulation results are evaluated from three perspectives for all conditions - the capability of the model to achieve the MPC control references, the kinematic results, and the kinetic results either compared to the experimental data or between the simulations under different conditions. In Chap. 6 and Chap. 7, from MPC control references and kinematics perspectives the model achieved good results. The MPC control references are achieved accurately and the kinematics results either follow the experimental data closely or predict the characteristics clearly. However, the kinetics results are not fully consistent with the experimental data or not very illuminating.

The discrepancies in the kinetic results can be from three reasons: The first reason is because of the discrepancies between the real human gait dynamics and the developed plant model. As one widespread motto in engineering field said, "No model is real, every model is wrong". One can never build a model that is completely the same as the target plant. The same problem exists in this dissertation in that a plant model has to be used to represent human gait because of the nature of the project. Even though the plant model is verified in Chap. 2 that it can closely represent the forward dynamics of human gait, there still exists some discrepancies. These discrepancies cause the joint moments to drive the plant model to be different from those to drive the real gait dynamics.

The second reason is that different motor behavior can be employed to achieve similar kinematics output. Bateni and Olney [51] reported that even within the amputee subjects the motor control strategy employed varies between individuals. For example, in the pathological gait simulation in Chap. 7, the developed MPC controller uses more extension moment in the prosthetic knee to prevent the body collapse because of a lack of support from prosthetic passive ankle. Instead, the experimental data [51, 56] shows that the amputees more often choose to use more extension moment in the intact knee to achieve the same function. Because there is no prescribed kinetic pattern for the developed model to follow, the resultant kinetic simulation results can be different from experimental data even though the same behavior is achieved.

The third reason lies in the discrepancies in the objective function utilized between the developed model and the CNS. The objective functions in the developed MPC control system only regulates the step length and velocity of the COM. This objective function is surprisingly simple nevertheless is able to generate gaits under various condition. However, this simple objective function may not fully represent the control target of the CNS. For example, the CNS may consider some other gait related factor such as dynamic effort or metabolic energy consumption as part of the objective function which is not captured in the developed model. The differences in the objective function can cause the deviation in the kinetic results. Therefore, the author suggests the achievement of MPC control reference and kinematics results should be emphasized more than the kinetic results.

8.3 Future Work

This dissertation is far from concluding a research topic. Instead, the purpose of this dissertation is to open the door of a whole new world of using a predictive control method to predict able-bodied and pathological gait in order to reduce cost, minimize risk, and facilitate the development of P&O. The core part of this dissertation is to use a predictive control method to simulate human gait. As explained in Chap. 1, the author believes that the CNS employs the same predictive control philosophy when regulating human gait instead of using widespread and classical feedback control. This predictive control philosophy should be maintained as the baseline principle for future work. In addition, this predictive control philosophy not only can be employed for human gait simulation research but also can be employed to P&O development. As the powered P&Os are developing rapidly in the U.S and the microprocessors are becoming more powerful and cheaper, the predictive control method should be considered to be the potential control strategy for future powered P&Os.

While the core part of predictive control should be maintained for future research, the other parts of the developed model in this dissertation have room for improvement. First, the plant model can be improved to more closely represent the forward dynamics of human gait. As explained in Chap. 1, the current research of human gait model can be broken into two categories: Biomechanics gait models and biped robotics gait models. The author does not believe there is a need to adopt biomechanics model for future research because this type of model usually has hundreds of DOFs and requires weeks to perform simulation which deviates from the essential purpose of this dissertation. However, numerous methods can be adopted from the biped robotics gait model to make the plant model better represent the real human gait. For example, numerous literature suggests arms have important dynamic effect in human walking which helps human gait maintain balance. The future work should considering including the swing of arms into the plant model. The other possible direction is to build a more advanced GRF model where currently the GRF is modeled as four groups of springs and dampers both horizontally and vertically at both toe and heel. Various literature suggests their GRF model achieves good simulation results and may be worth being considered [57–59]. The author believes with more realistic plant model built, the developed model should achieve better performance.

The second part the author believes can be improved is the structure of the internal model. For example, for each segment the mass and moment of inertia can be lumped to several points or distribute continuously along the segment instead of the current configuration as a point mass to achieve better prediction for the MPC controller. However, in developing the internal MPC model, one basic principle is that the structure of the internal model should be maintained simple. Therefore the forward dynamics of the internal model is relatively simple to integrate to maintain the simulation time to be at a reasonable level, which spans from several hours to several days depending on the computational hardware.

The third part which can be improved is the MPC minimization algorithm and constraints. The current model uses step length and velocity of the COM at push-off as the minimization criteria and several other constraints as explained in Chap. 5. Some other minimization criteria are worth trying in combination with the existing step length and velocity at COM. For example, the minimization of dynamic effort and metabolic energy are two widely utilized criteria in optimization-based approach in human gait research. It should be noteworthy adding those minimization criteria to the existing model and verify if better simulation results are obtained.

8.4 Final Remarks

This dissertation developed a human gait model using a novel approach which uses a combination of MPC and feedback control to function as the CNS to control the plant gait model. The developed model is able to simulate able-bodied human gait at self-selected speed. The kinematic and kinetic results are close to the experimental data. More importantly, the developed model is able to simulate able-bodied human gait at various speed and pathological gait with unilateral passive ankle. The simulation results show that the developed model is able to qualitatively predict the characteristics of able-bodied gait at various speed and the transtibial pathological gait with passive prosthesis.

REFERENCES

[1] Yujiang Xiang, Jasbir Arora, and Karim Abdel-Malek. Physics-based modeling and simulation of human walking: a review of optimization-based and other approaches. *Structural and Multidisciplinary Optimization*, 42(1):1–23, July 2010.

[2] Shuuji Kajita, Osamu Matsumoto, and Muneharu Saigo. Real-time 3d walking pattern generation for biped robot with telescopic legs. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2299–2306, Seoul, Korea, May 2001.

[3] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kiyoshi Fujiwara, Kazuhito Yokoi, and Hirohisa Hirukawa. A realtime pattern generator for biped walking. In *Proceedings of 2002 IEEE International Conference on Robotics and Automation*, pages 31–37, Washington, D.C., May 2002.

[4] Shunsuke Kudoh and Taku Komura. C2 continuous gait-pattern generation for biped robots. In *Proceedings of 2003 IEEE International conference on Intelligent Robots and Systems*, pages 1135–1140, Las Vegas, NV, 2002.

[5] Amos Albert and Wilfried Gerth. Analytic path planning algorithms for bipedal robots without a trunk. *Journal of Intelligent and Robotic Systems*, 36(2):109–127, February 2003.

[6] Taesin Ha and Chong-ho Choi. An effective trajectory generation method for bipedal walking. *Robotics and Autonomous Systems*, 55(10):795–810, June 2007.

[7] Tad McGeer. Passive dynamic walking. *The International Journal of Robotics Research*, 9(2):62–82, April 1990.

[8] Yildirim Hurmuzlu, Frank Genot, and Bernard Brogliato. Modeling, stability and control of biped robots a general framework. *Automatica*, 40(10), October

2004.

[9] Arthur D. Kuo. Stabilization of lateral motion in passive dynamic walking. *International Journal of Robotics Research*, 18(9):917–930, September 1999.

[10] Steve Collins, Andy Ruina, Russ Tedrake, and Martijn Wisse. Efficient bipedal robots based on passive-dynamic walker. *Science*, 307(5712), February 2005.

[11] Arthur D. Kuo. A simple model of bipedal walking predicts the preferred speed-step length relationship. *Journal of Biomechanical Engineering*, 123(3):264–269, June 2001.

[12] Arthur D. Kuo. Energetics of actively powered locomotion using the simplest walking model. *Journal of Biomechanical Engineering*, 124(1):113–120, February 2002.

[13] Arthur D. Kuo, J. Maxwell Donelan, and Andy Ruina. Energetic consequences of walking like an inverted pendulum: Step-to-step transitions. *Exercise and Sport Sciences Reviews*, 33(2):88–97, April 2005.

[14] A. Takanishi, M. Ishida, Y. Yamazaki, and I. Kato. The realization of dynamic walking by biped walking robot wl40rd. In *Proceedings of the 1985 International Conference on Advanced Robotics*, pages 459–466, 1985.

[15] Jin-ichi Yamaguchi, Atsuo Takanishi, and Ichiro Kato. Development of a biped walking robot compensating for three-axis moment by trunk motion. In *Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 561–566, Yokohama, Japan, July 1993.

[16] J. Furusho and A. Sano. Sensor-based control of a 9-link biped. *International Journal of Robotics Research*, 9(2):83–98, April 1990.

[17] Tsutomo Mita, Toru Yamaguchi, Toshio Kashiwase, and Taro Kawase. Realization of a high-speed biped using modern control-theory. *International Journal of Control*, 40(1):107–119, January 1984.

[18] Hirofumi Miura and Isao Shimoyama. Dynamic walk of a biped. *International Journal of Robotics Research*, 3(2):60–74, June 1984.

[19] Yuan F. Zheng and F. Sias. Design and motion control of practical biped robots. *International Journal of Robotics and Automation*, 3(2):70–77, 1988.

[20] Yuan F. Zheng and Jie Shen. Gait synthesis for the sd-2 biped robot to climb sloping surface. *IEEE Transactions on Robotics and Automation*, 6(1):86–96, February 1990.

[21] Qiang Huang, Kazuhito Yokoi, Shuuji Kajita, Kenji Kaneko, Hirohiko Arai, Noriho Koyachi, and Kazuo Tanie. Planning walking patterns for a biped robot. *IEEE Transactions on Robotics and Automation*, 17(3), June 2001.

[22] Ching-long Shih. Gait synthesis for a biped robot. *Robotica*, 15(6):599–607, November 1997.

[23] Kazuo Hirai. The Honda humanoid robot: Development and future perspective. *Industrial Robot: An International Journal*, 26(4):260–266, 1999.

[24] Ching-long Shih. The dynamics and control of a biped walking robot with seven degrees of freedom. *Journal of Dynamic Systems, Measurement and Control*, 118(4):683–690, December 1996.

[25] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kiyoshi Fujiwara, Kensuke Harada, Kazuhito Yokoi, and Hirohisa Hirukawa. Biped walking pattern generation by using preview control of zero-moment point. In *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, pages 1620–1626, Taipei, China, September 2003.

[26] Subhra Chowdhury and Neelesh Kumar. Estimation of forces and moments of lower limb joints from kinematics data and inertia properties of the body by using inverse dynamics technique. *Journal of Rehabiliation Robotics*, 1(2):93–98, November 2013.

[27] Guy Bessonnet, Stephane Chesse, and Philippe Sardain. Optimal gait synthesis of a seven-link planar biped. *International Journal of Robotics Research*, 23(10-11):1059–1073, October-November 2004.

[28] Guy Bessonnet, P. Seguin, and Philippe Sardain. A parametric optimization approach to walking pattern synthesis. *International Journal of Robotics Research*, 24(7):523–536, July 2005.

[29] Lei Ren, Richard K. Jones, and David Howard. Predictive modeling of human walking over a complete gait cycle. *Journal of Biomechanics*, 40(7):1567–1574, 2007.

[30] Frank C. Anderson and Marcus G. Pandy. Dynamic optimization of human walking. *Journal of Biomechanical Engineering*, 123(5):381–390, October 2001.

[31] Marcus G. Pandy. Computer modeling and simulation of human movement. *Annual Review of Biomedical Engineering*, 3:245–273, 2001.

[32] R.T. Marler and J.S. Arora. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26(6):369–395, March 2004.

[33] Herman van der Kooij, Ron Jacobs, Bart Koopman, and Frans van der Helm. An alternative approach to synthesizing bipedal walking. *Biological Cybernetics*, 88(1):46–59, January 2003.

[34] Dusko Katic and Miomir Vukobratovic. Survey of intelligent control techniques for humanoid robots. *Journal of Intelligent and Robotic System*, 37(2):117–141, June 2003.

[35] E.R. Westervelt, J.W. Grizzle, and D.E. Koditschek. Hybrid zero dynamics of planar biped walkers. *IEEE Transactions on Automatic Control*, 48(1):42–56, December 2001.

[36] Christine Azevedo, Philippe Poignet, and Bernard Espiau. Artificial locomotion control: From human to robots. *Robotics and Autonomous Systems*, 47(4):565–573, June 2004.

[37] Peter Gawthrop, Ian Loram, and Martin Lakie. Predictive feedback in human simulated pendulum balancing. *Biological Cybernetics*, 101(2):131–146, July 2009.

[38] M. Karimian, F. Towhidkhah, and M. Rostami. Application of model predictive impedance control in analysis of human walking on rough terrains. *International Journal of Applied Electromagnetics and Mechanics*, 24(3):147–162, 2005.

[39] Elena Borzova and Yildirim Hurmuzlu. Passively walking five-link robot. *Automatica*, 40:621–629, 2004.

[40] David.A. Winter. *Biomechanics and Motor Control of Human Movement.* Wiley-Interscience, second edition, 1990.

[41] Eduardo F. Camacho and Carlos Bordons. *Model Predictive Control.* Springer, second edition, 2007.

[42] Liuping Wang. *Model Predictive Control System Design and Implementation Using MATLAB.* Springer, first edition, 2009.

[43] B.W. Verdaasdonk, H.F.J.M. Koopman, and F.C.T. van der Helm. Energy efficient walking with central patter generators: from passive dynamic walking to biologically inspired control. *Biological Cybernetics*, 101(1):49–61, July 2009.

[44] B. Wahlberg. System identification using laguerre models. *IEEE Transactions on Automatic Control*, 36(5):551–562, May 1991.

[45] Vicon Motion Systems Ltd. *Plug-In Gait Manual.*

[46] Jacquelin Perry and Judith Burnfield. *Gait analysis: Normal and Pathological Function.* Slack Incorporated, second edition, 2010.

[47] Jennifer L. Lelas, Gregory J. Merriman, Patrick O. Riley, and D. Casey Kerrigan. Predicting peak kinematic and kinetic parameters from gait speed. *Gait and Posture*, 17(2):106–112, April 2003.

[48] Jonathan Shemmell, Jennifer Johansson, Vanessa Portra, Gerald L. Gottlieb, James S. Thomas, and Daniel M. Corcos. Control of interjoint coordination during the swing phase of normal gait at different speeds. *Journal of Neuroengineering and Rehabilitation*, 4(10), April 2007.

[49] J.P. Paul. The effect of walking speed on the force actions transmitted at the hip and knee joints. In *Proceedings of the royal society of medicine*, pages 200–202, February 1970.

[50] D.A. Winter and S.E. Sienko. Biomechanics of below-knee amputee gait. *Journal of Biomechanics*, 21(5):361–367, 1988.

[51] H. Bateni and S. Olney. Kinematic and kinetic variations of below-knee amputee gait. *Journal of Prosthetics and Orthotics*, 14(1):2–13, 2002.

[52] H.B. Skinner and D.J. Effeney. Gait analysis in amputees. *American Journal of Physical Medicine and Rehabilitation*, 64(2).

[53] Rawesak Tanawongsuwan and Aaron F. Bobick. Modeling the effects of walking speed on appearance-based gait recognition. In *Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition*, pages 783–790, June 2004.

[54] D.A. Winter. Biomechanical motor patterns in normal walking. *Journal of Motor Behavior*, 15(4):302–330, 1983.

[55] Bryan.J. Bergelin and Philip.A. Voglewede. Design of an active ankle-foot prosthesis utilizing a four-bar mechanism. *Journal of Mechanical Design*, 134(6), June 2012.

[56] David J. Sanderson and Philip E. Martin. Lower extremity kinematic and kinetic adaptations in unilateral below-knee amputees during walking. *Gait and Posture*, 6:126–136, 1997.

[57] R.R. Neptune, I.C. Wright, and A.J. van den Bogert. A method for numerical simulation of single limb ground contact events: application to heel-toe running. *Computer Methods in Biomechanics and Biomedical Engineering*, 3(4):321–334, February 2000.

[58] O. Bruneau and F.B. Ouezdou. Dynamic walk simulation of various bipeds via ankle trajectory. In *Proceedings of the 1998 IEEE/RSJ International*

Conference on Intelligent Robots and Systems, pages 58–63, Victoria, B.B., Canada, October 1998.

[59] R.W. Soames. Foot pressure patterns during gait. *Journal of Biomedical Engineering*, 7(2):120–126, April 1985.

APPENDIX A

The Kinematic Results of the Open Loop Human Gait Model Simulation

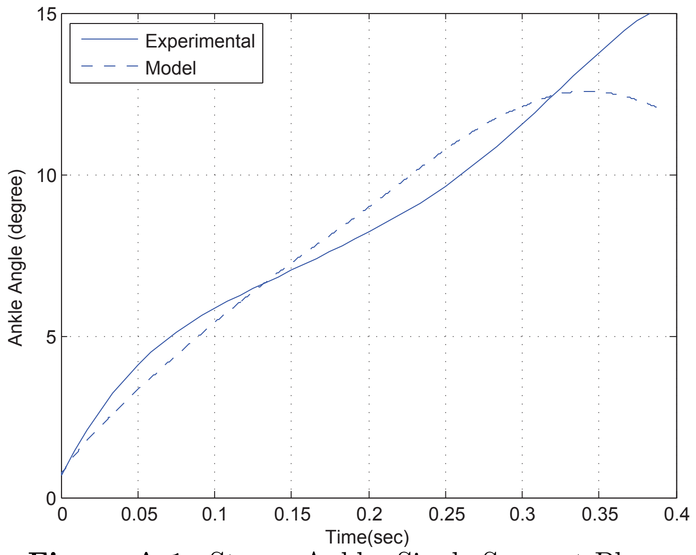

Figure A.1: Stance Ankle Single Support Phase

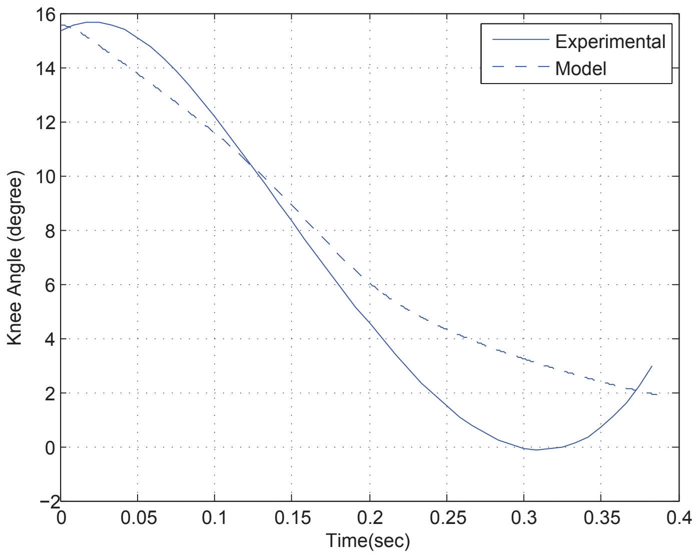

Figure A.2: Stance Knee Single Support Phase

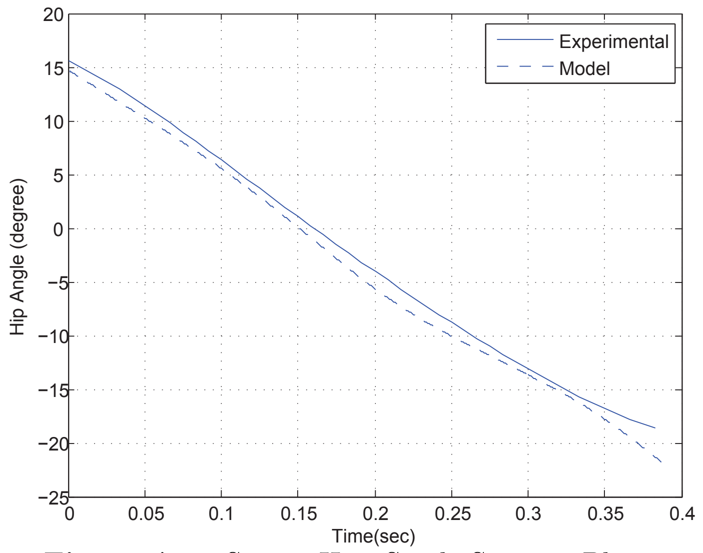

Figure A.3: Stance Hip Single Support Phase

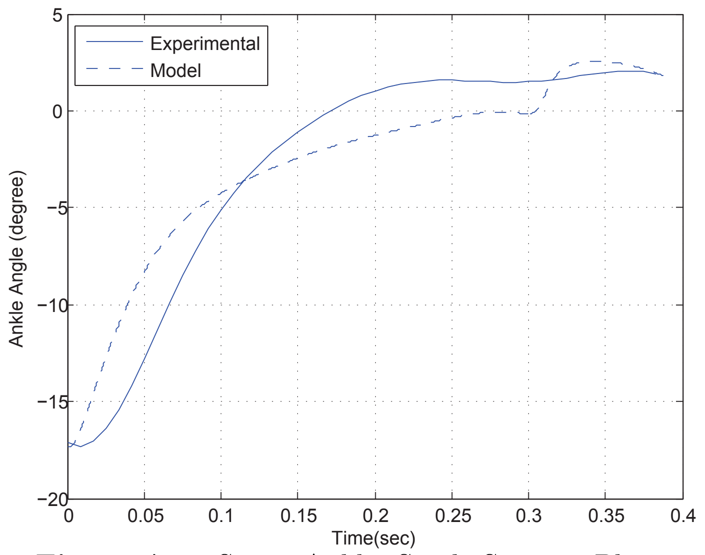

Figure A.4: Swing Ankle Single Support Phase

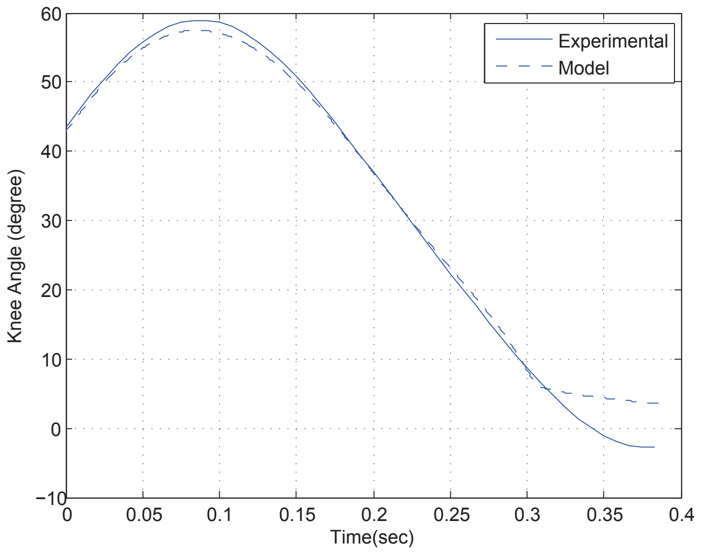

Figure A.5: Swing Knee Single Support Phase

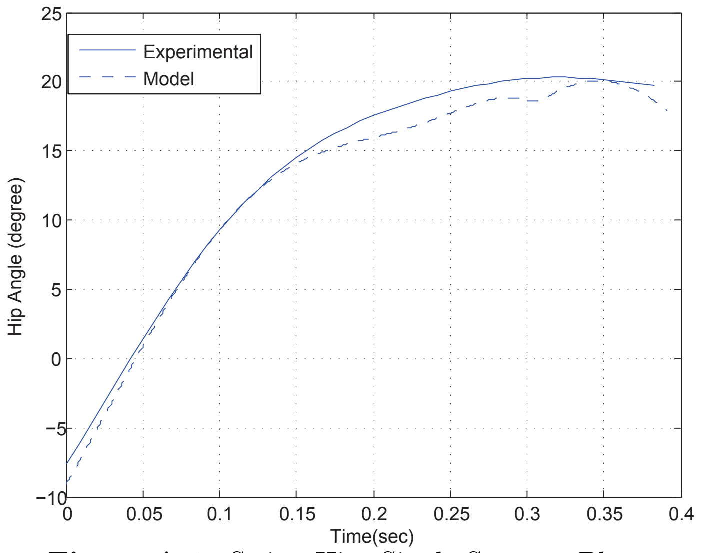

Figure A.6: Swing Hip Single Support Phase

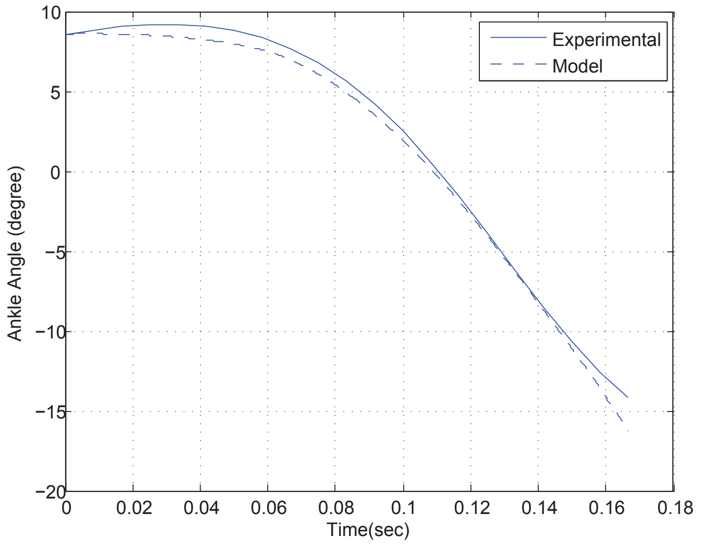

Figure A.7: Stance Ankle Double Support Phase

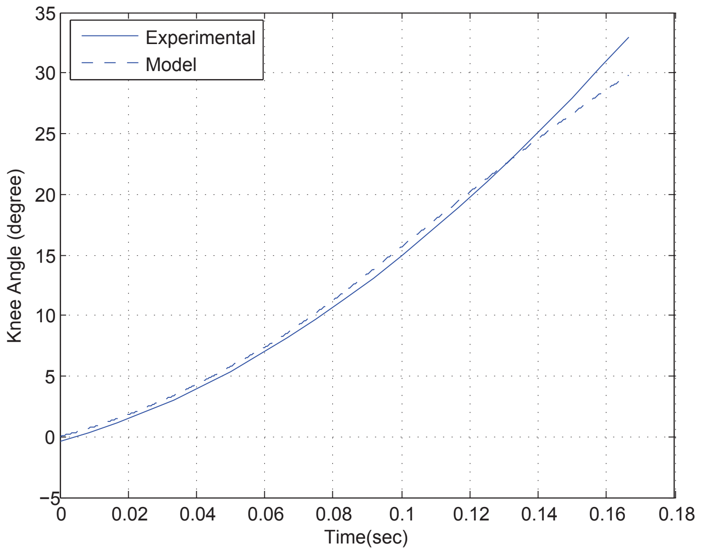

Figure A.8: Stance Knee Double Support Phase

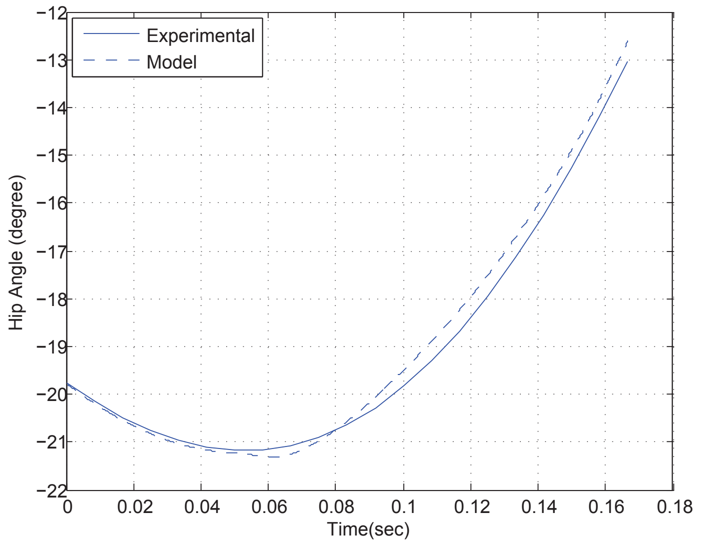

Figure A.9: Stance Hip Double Stance Phase

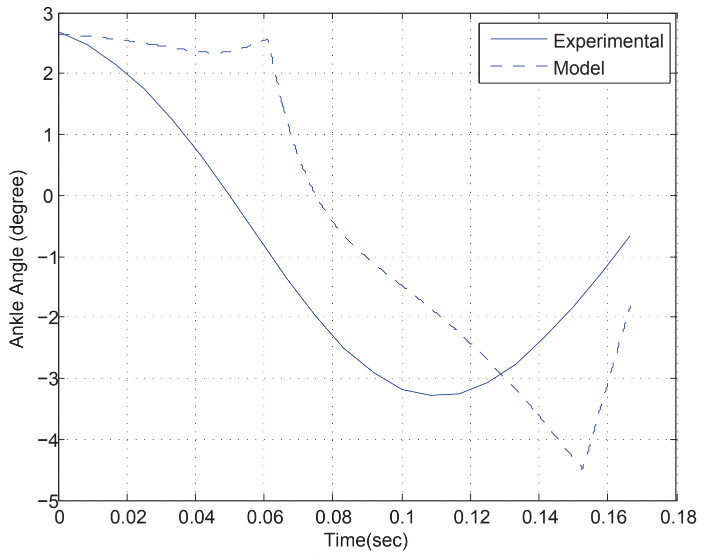

Figure A.10: Swing Ankle Double Support Phase

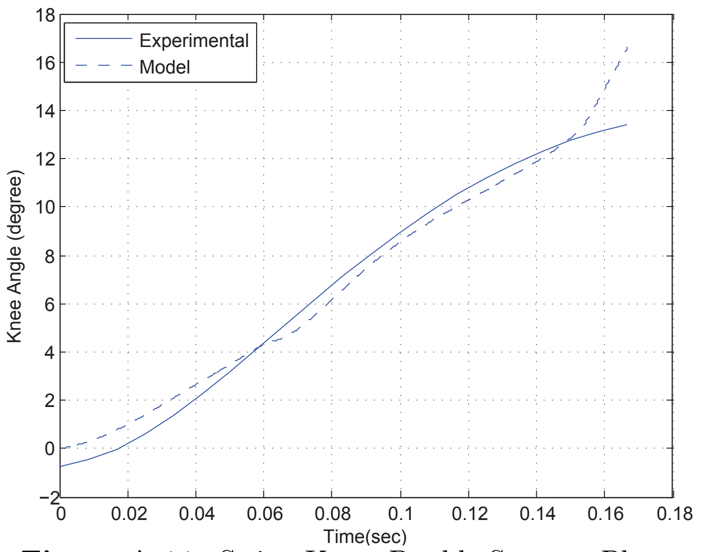

Figure A.11: Swing Knee Double Support Phase

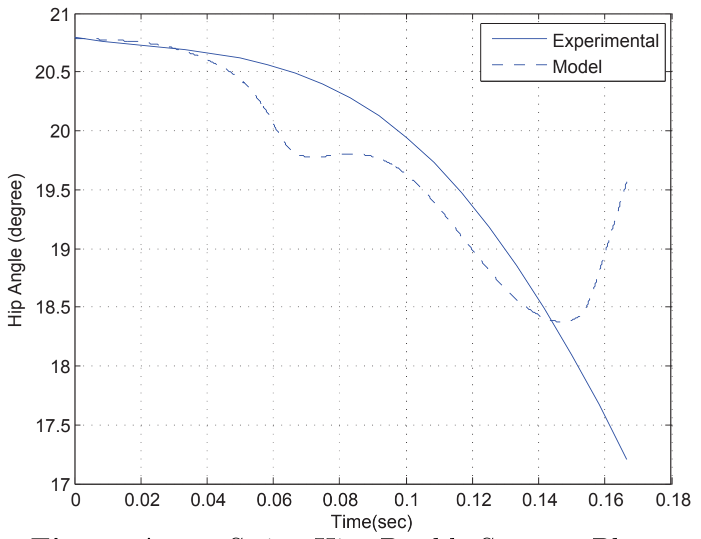

Figure A.12: Swing Hip Double Support Phase

APPENDIX B

Kinematic Results of the Open Loop Human Gait Model for Three Other Subjects

This appendix shows the simulation of the open-loop plant model of three other able-bodied subjects. The purpose of this appendix is to show the generality of the developed open-loop plant model explained in Chap. 2. The anthropometric parameters are customized to each subject while the internal mechanical parameters are the same as the ones used in Chap. 2.

B.1 Human Subject 2

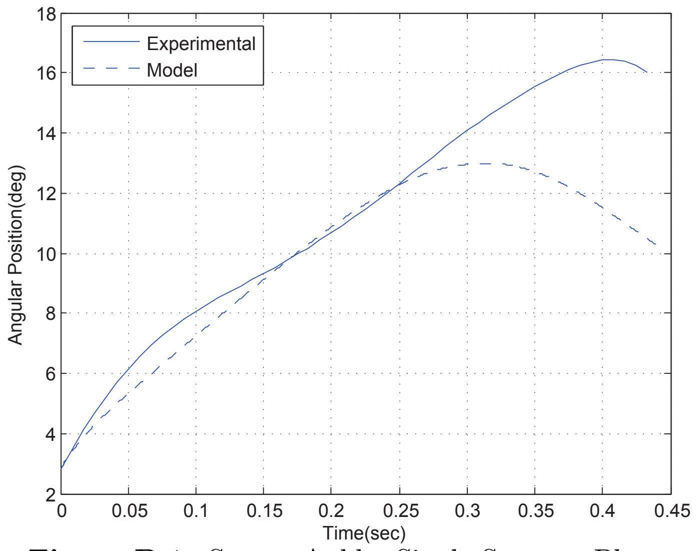

Figure B.1: Stance Ankle Single Support Phase

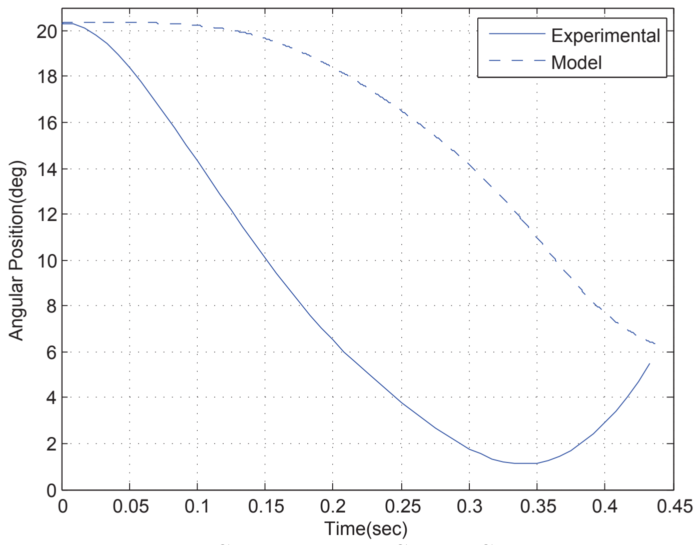

Figure B.2: Stance Knee Single Support Phase

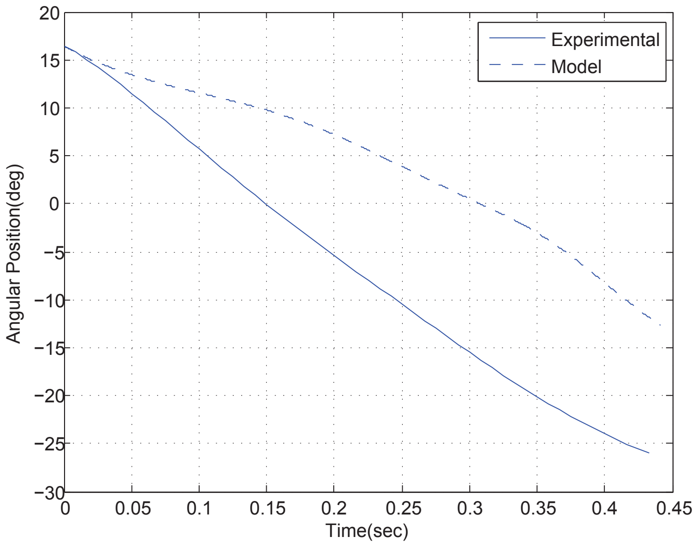

Figure B.3: Stance Hip Single Support Phase

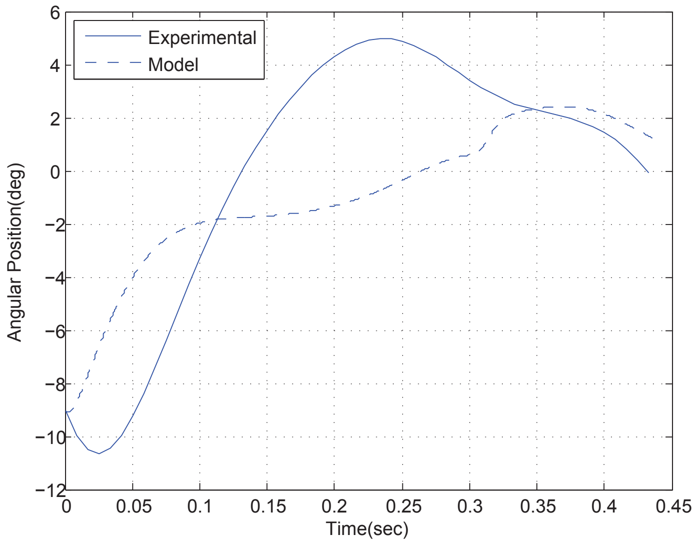

Figure B.4: Swing Ankle Single Support Phase

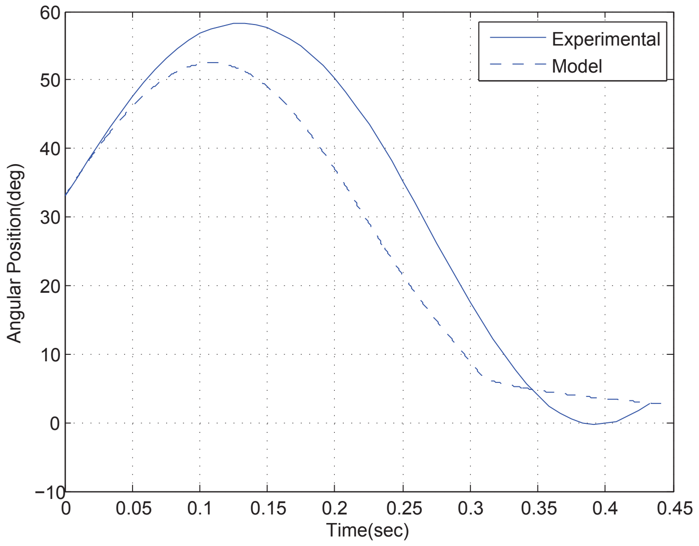

Figure B.5: Swing Knee Single Support Phase

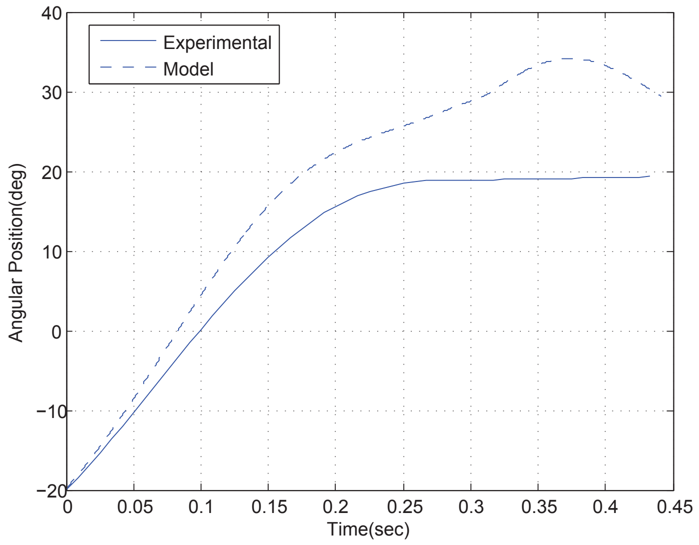

Figure B.6: Swing Hip Single Support Phase

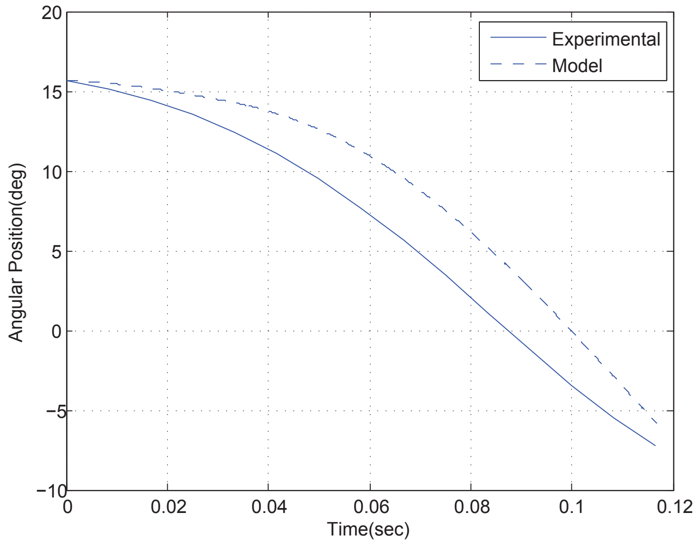

Figure B.7: Stance Ankle Double Support Phase

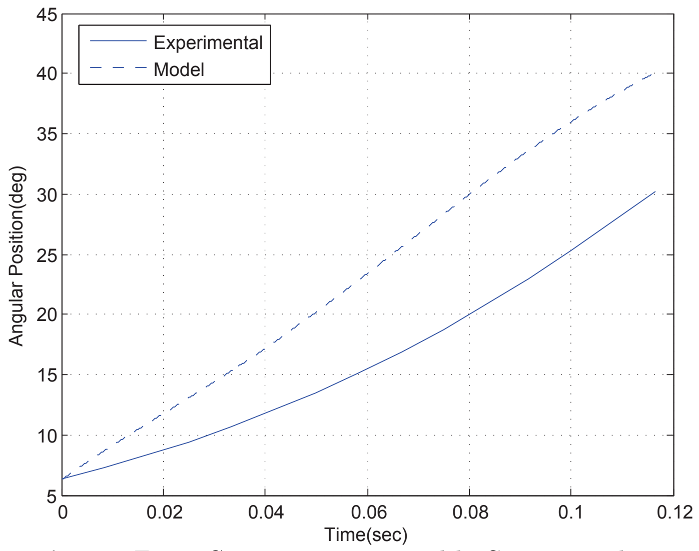

Figure B.8: Stance Knee Double Support Phase

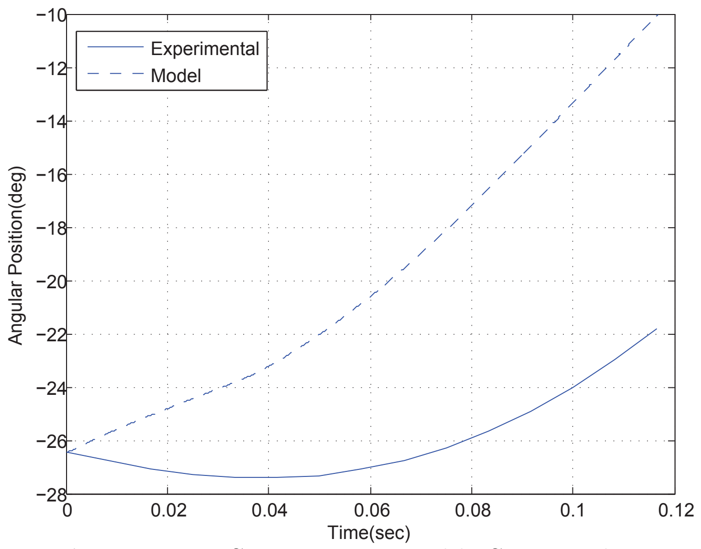

Figure B.9: Stance Hip Double Stance Phase

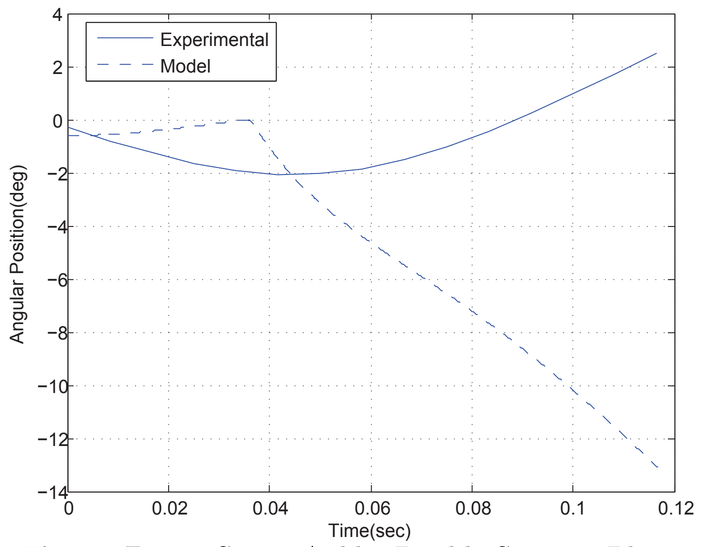

Figure B.10: Swing Ankle Double Support Phase

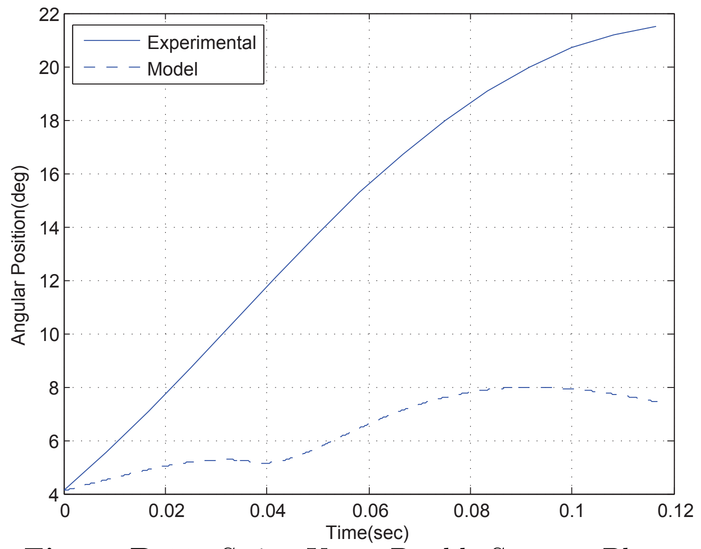

Figure B.11: Swing Knee Double Support Phase

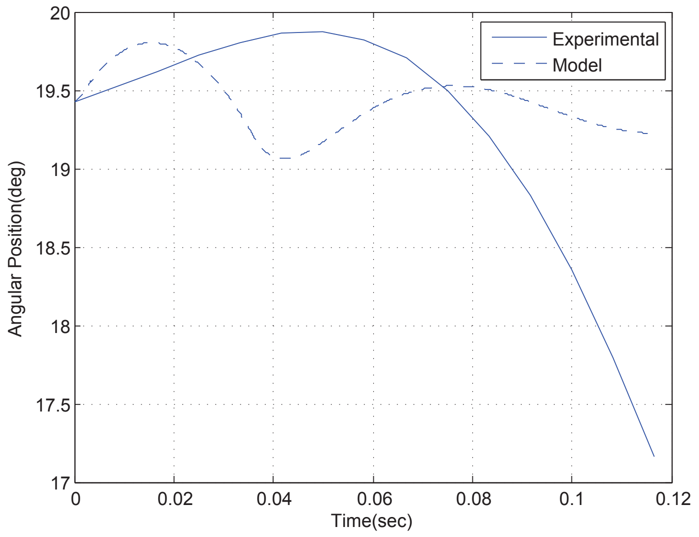

Figure B.12: Swing Hip Double Support Phase

B.2 Human Subject 3

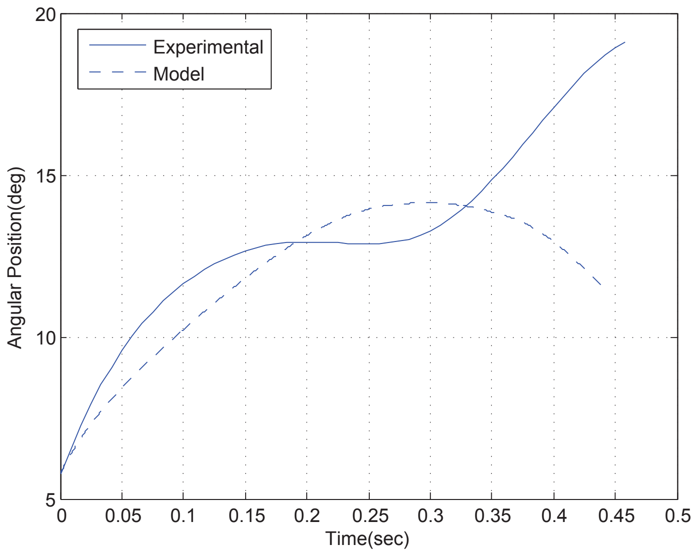

Figure B.13: Stance Ankle Single Support Phase

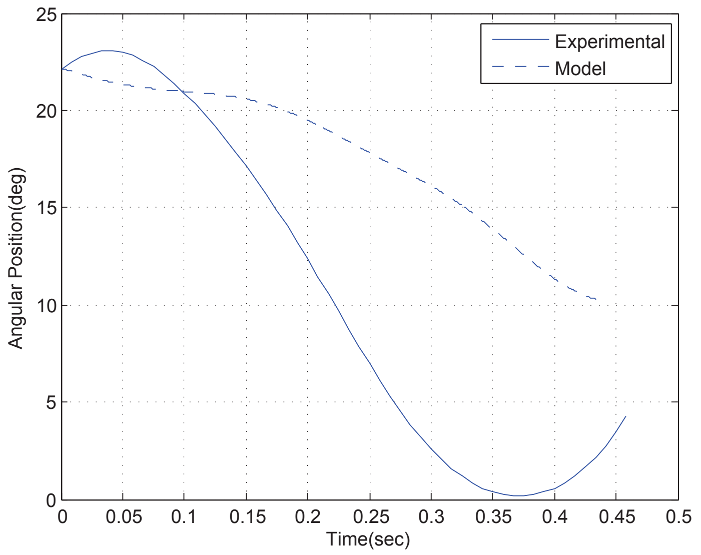

Figure B.14: Stance Knee Single Support Phase

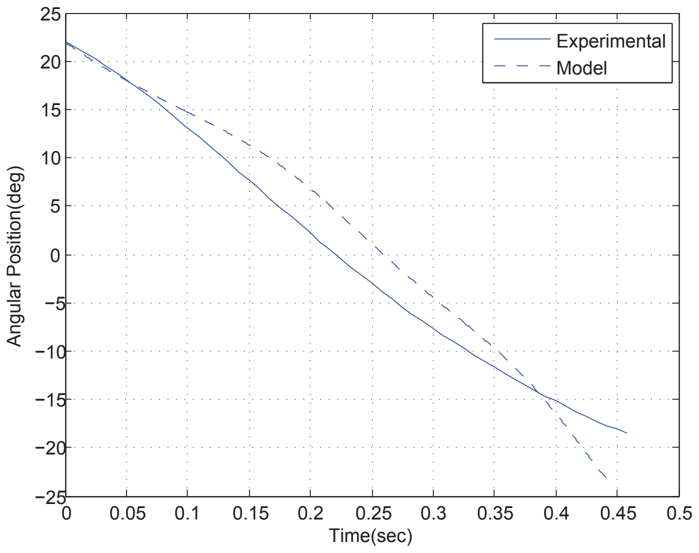

Figure B.15: Stance Hip Single Support Phase

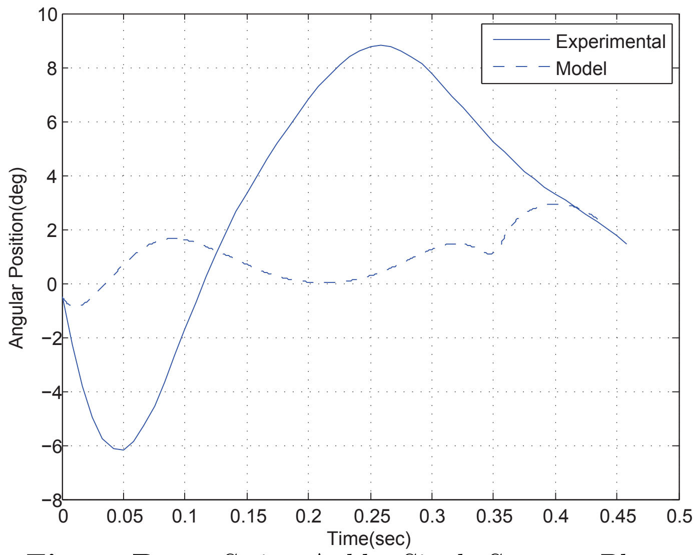

Figure B.16: Swing Ankle Single Support Phase

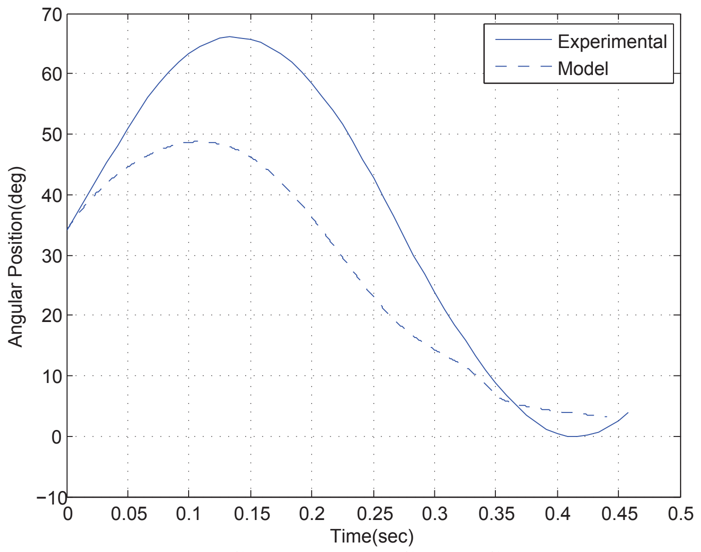

Figure B.17: Swing Knee Single Support Phase

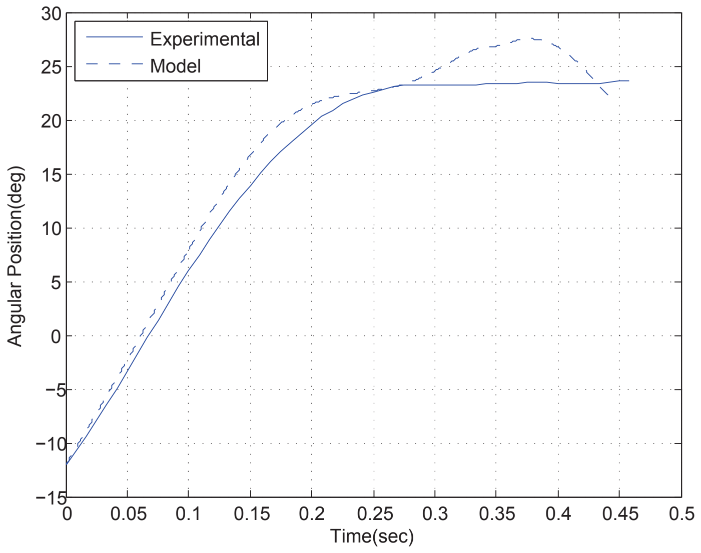

Figure B.18: Swing Hip Single Support Phase

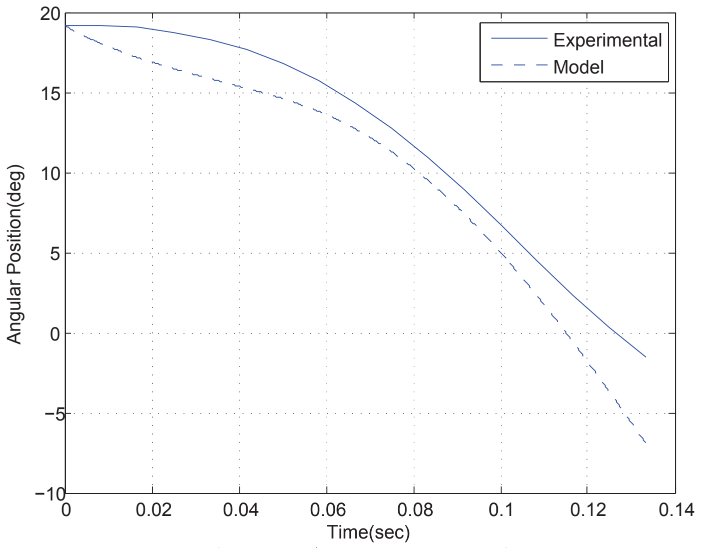

Figure B.19: Stance Ankle Double Support Phase

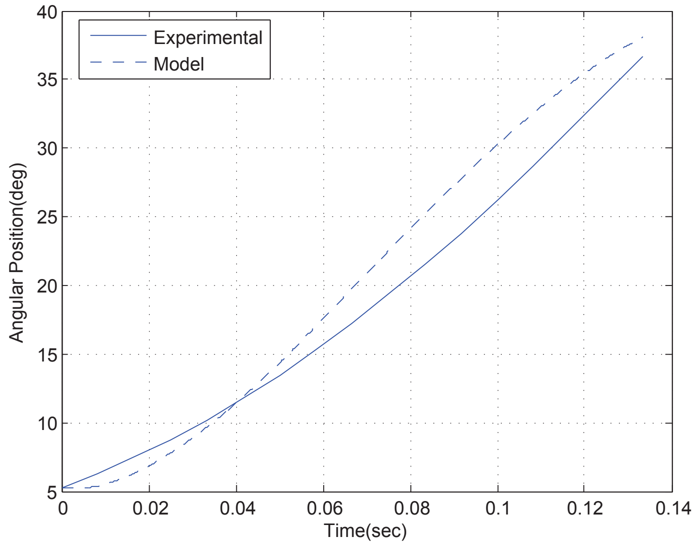

Figure B.20: Stance Knee Double Support Phase

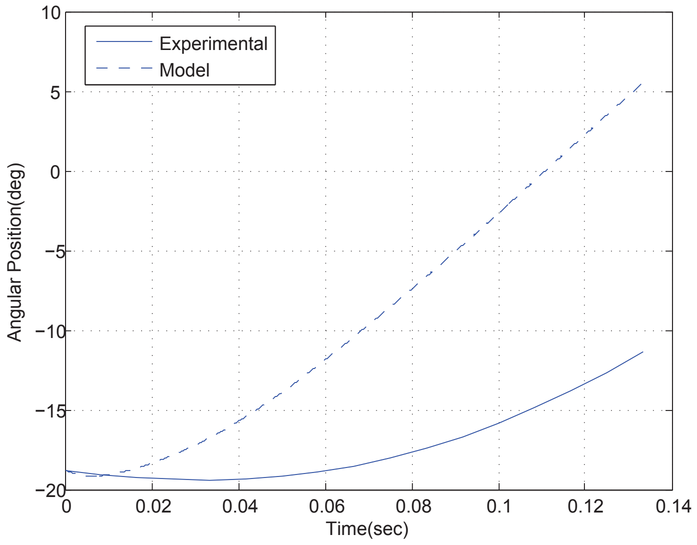

Figure B.21: Stance Hip Double Support Phase

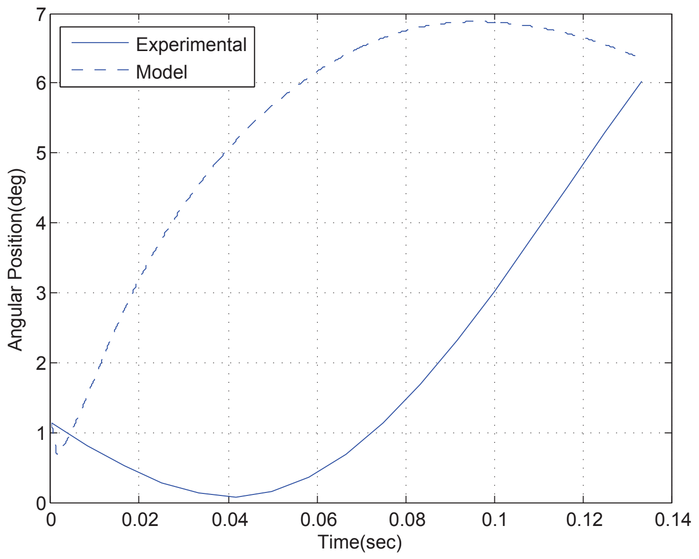

Figure B.22: Swing Ankle Double Support Phase

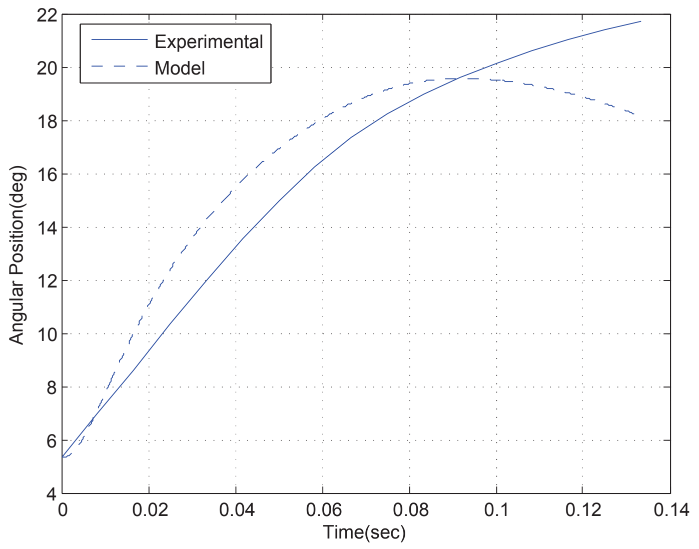

Figure B.23: Swing Knee Double Support Phase

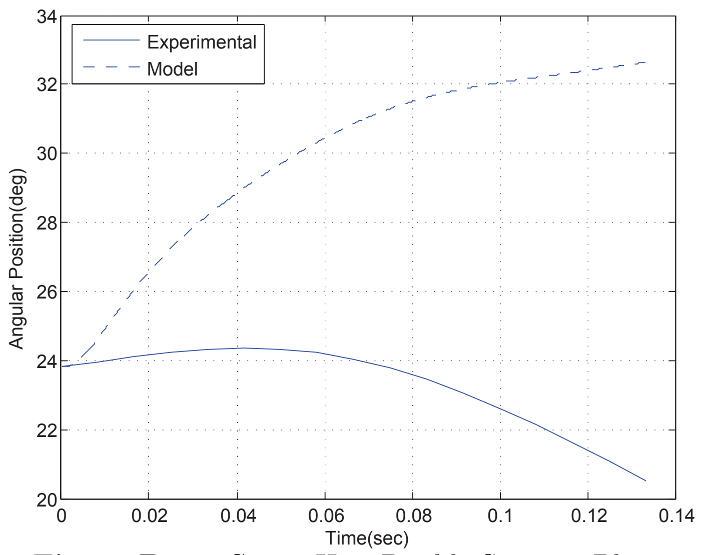

Figure B.24: Swing Hip Double Support Phase

B.3 Human Subject 4

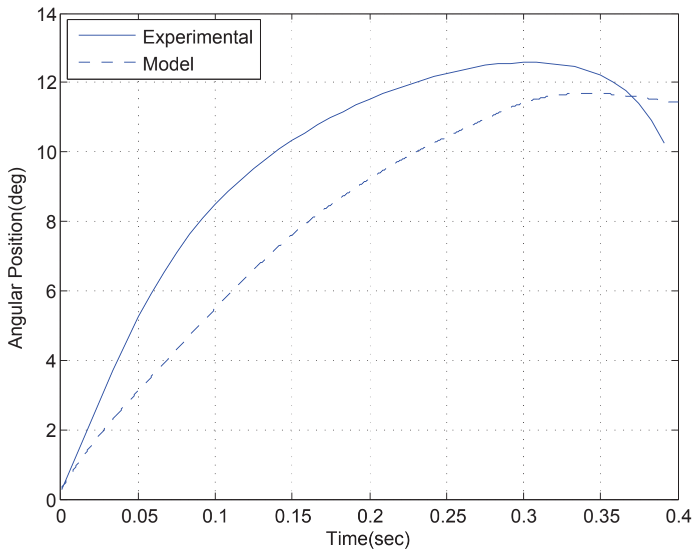

Figure B.25: Stance Ankle Single Support Phase

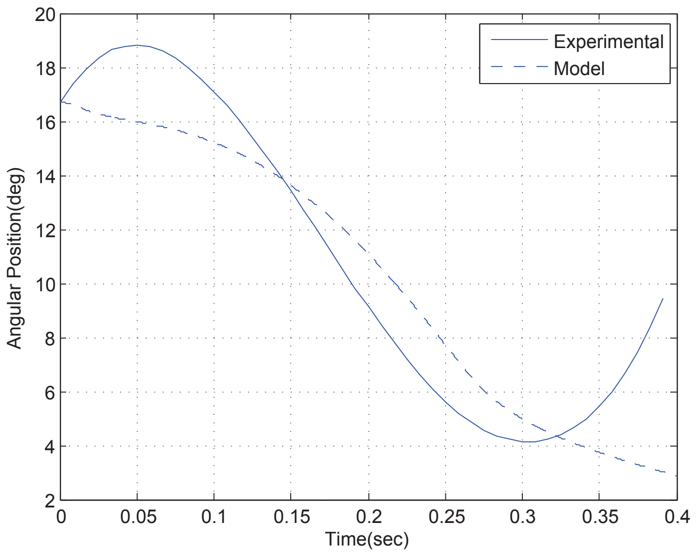

Figure B.26: Stance Knee Single Support Phase

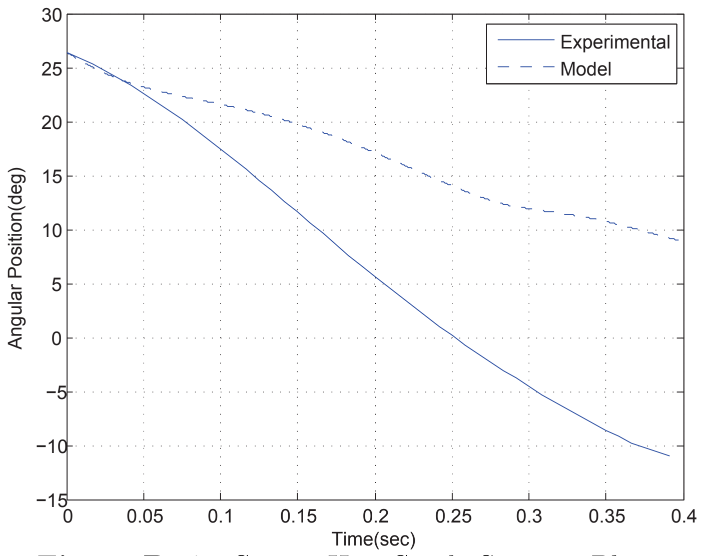

Figure B.27: Stance Hip Single Support Phase

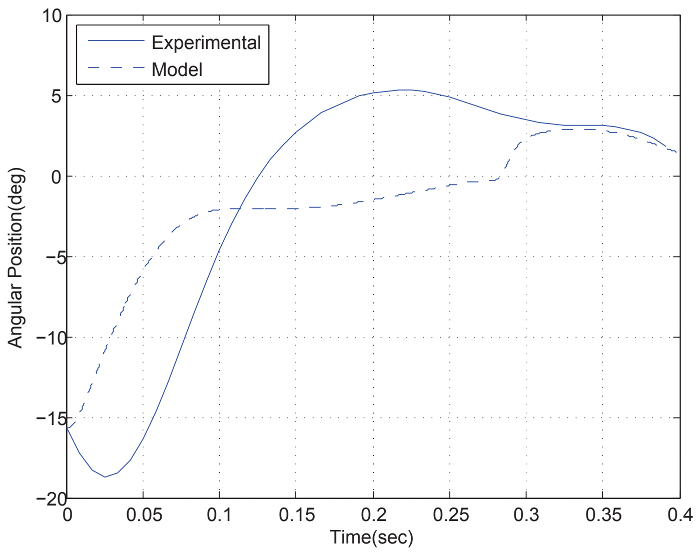

Figure B.28: Swing Ankle Single Support Phase

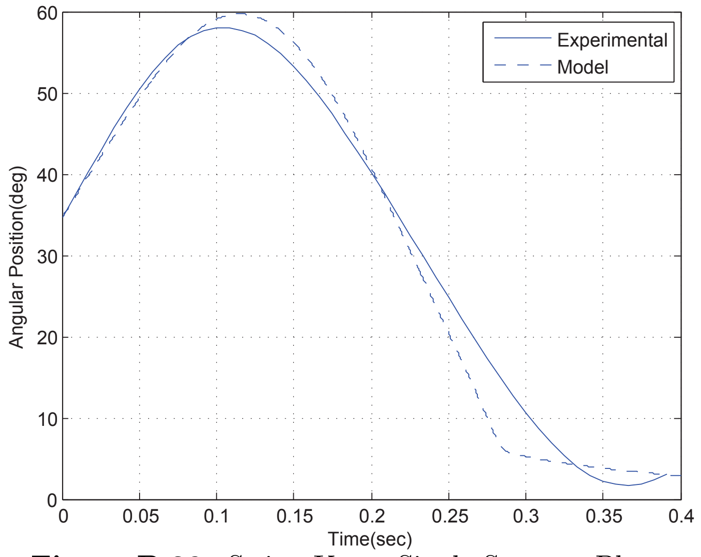

Figure B.29: Swing Knee Single Support Phase

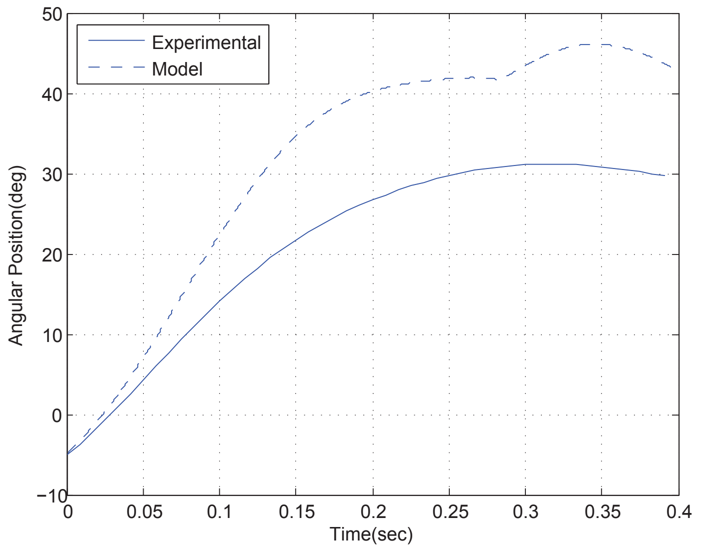

Figure B.30: Swing Hip Single Support Phase

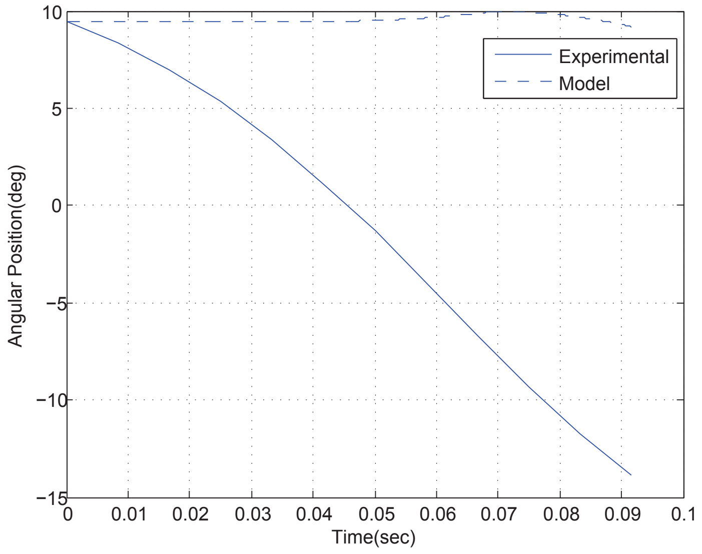

Figure B.31: Stance Ankle Double Support Phase

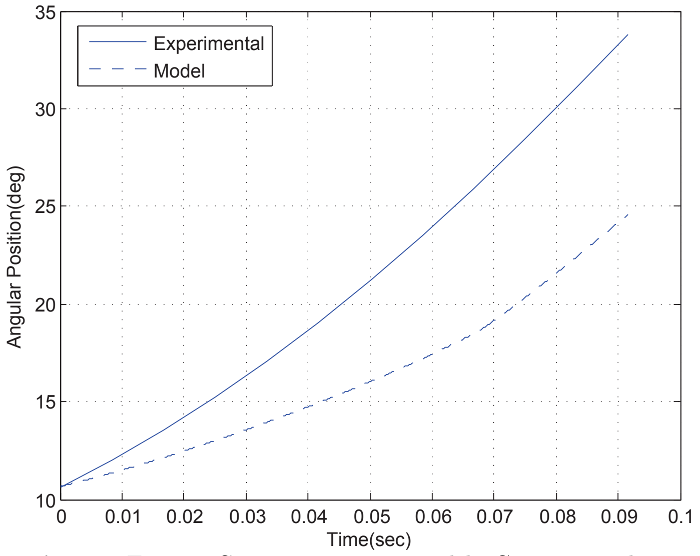

Figure B.32: Stance Knee Double Support Phase

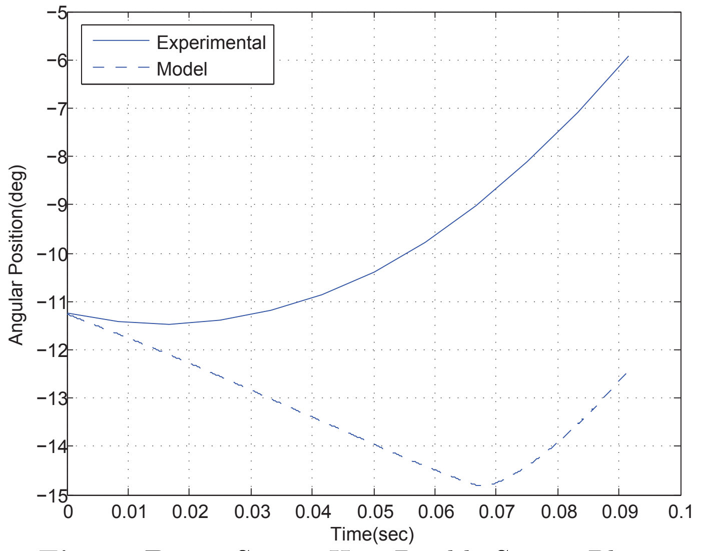

Figure B.33: Stance Hip Double Stance Phase

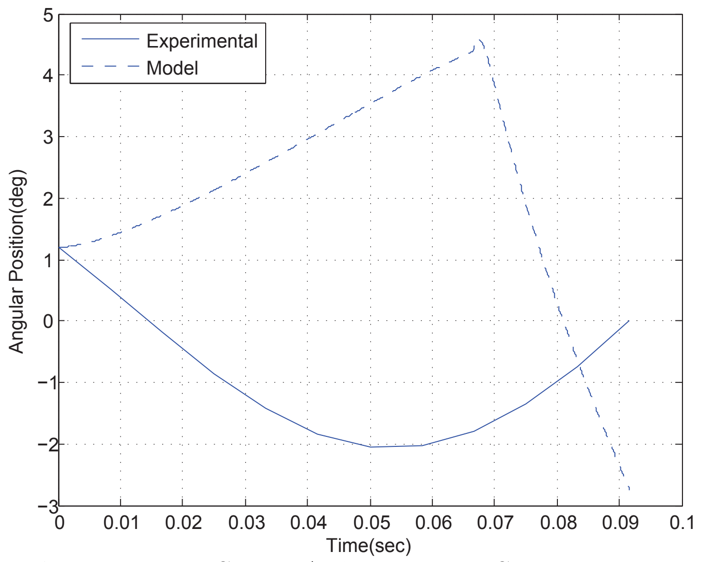

Figure B.34: Swing Ankle Double Support Phase

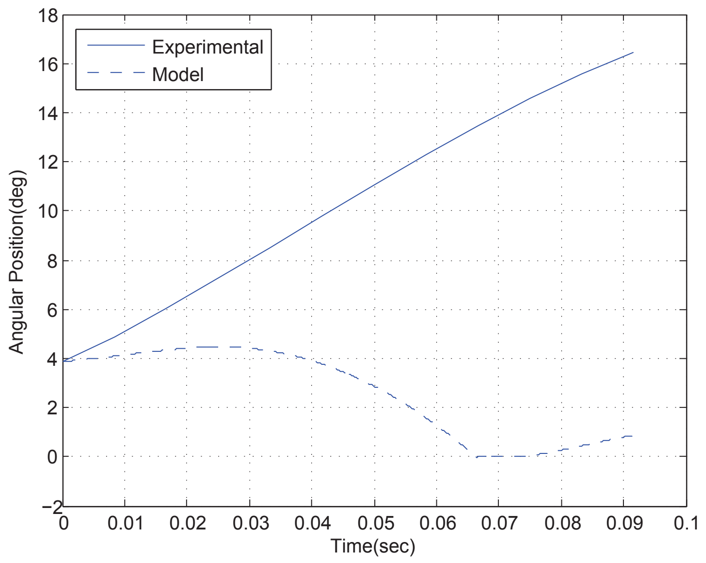

Figure B.35: Swing Knee Double Support Phase

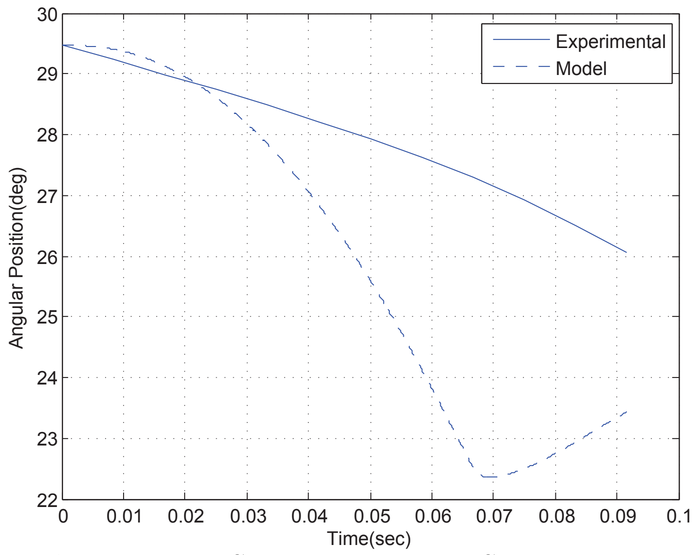

Figure B.36: Swing Hip Double Support Phase

APPENDIX C

Simulink Forward Dynamics Plant Model

This appendix illustrates the forward dynamic plant model developed using MATLAB Simulink, as explained in Chap. 2. Fig. C.1 shows the overall structure of the model and the following figures show the subsystems corresponding to the labeled numbers in Fig. C.1. The model parameters and their determined values are not shown in this appendix. Due to the symmetry, the structures of the model for both left and right sides are similar; only one side is illustrated.

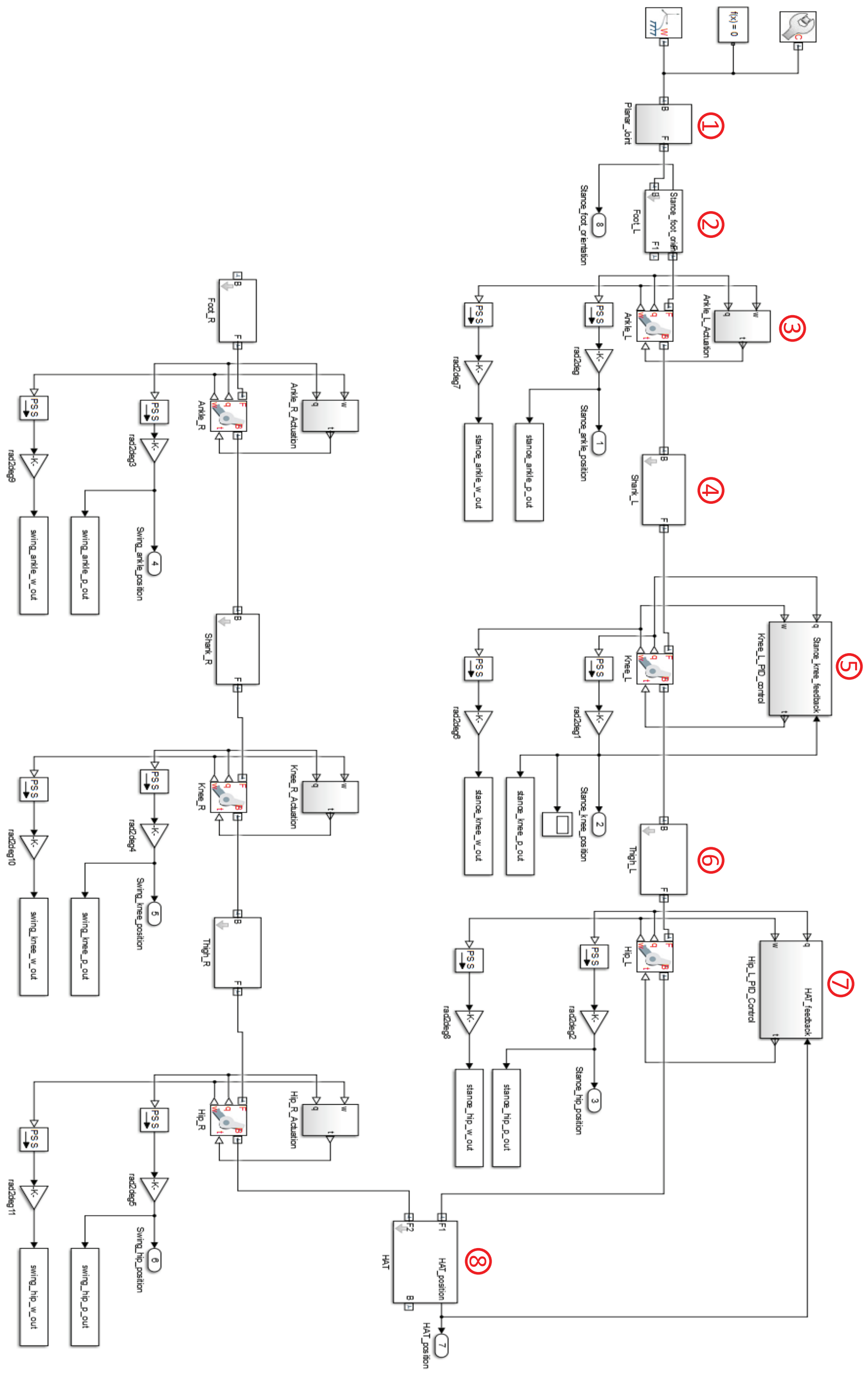

Figure C.1: Overview of the Forward Dynamics Plant Model

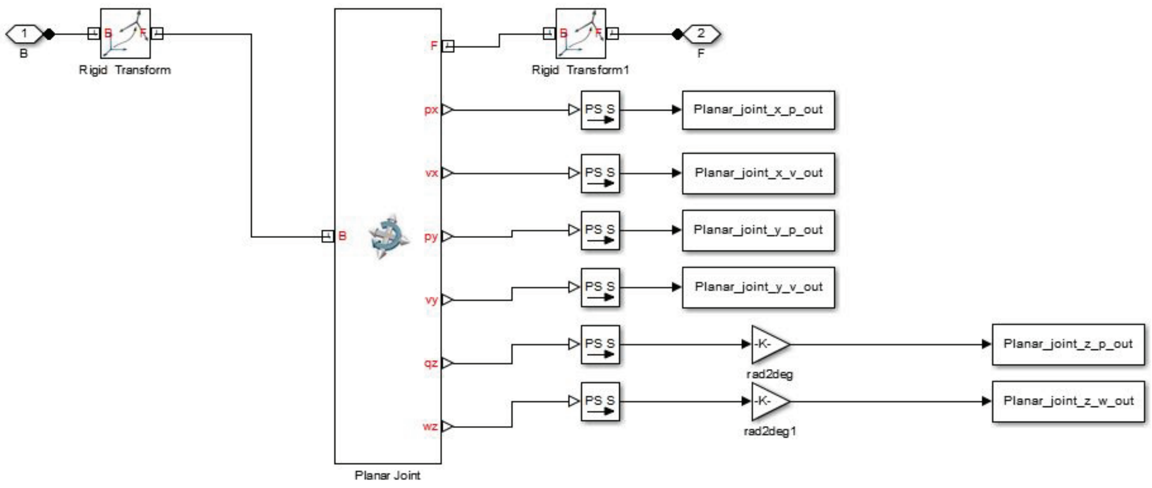

Figure C.2: Subsystem 1 - Planar Joint

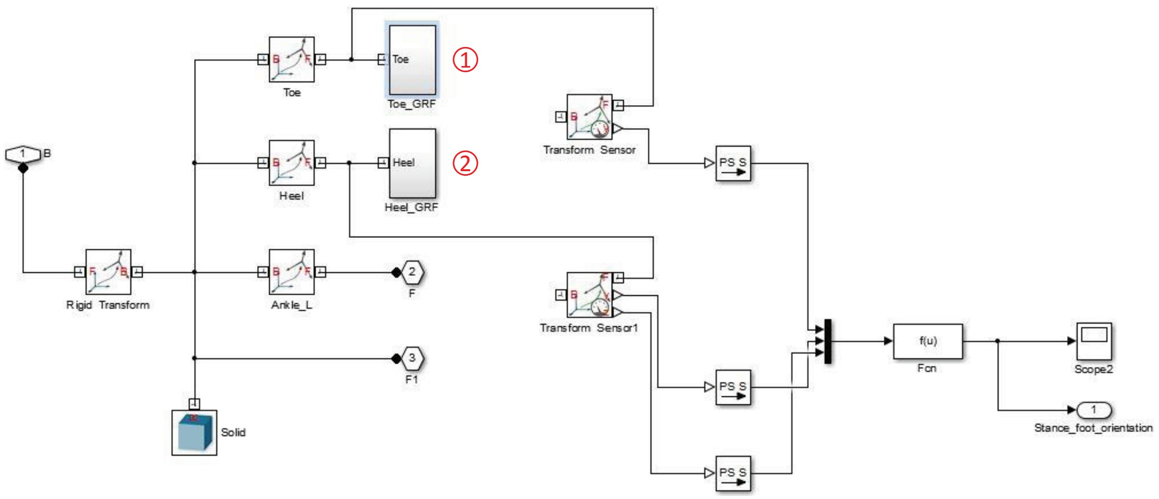

Figure C.3: Subsystem 2 - Stance Foot

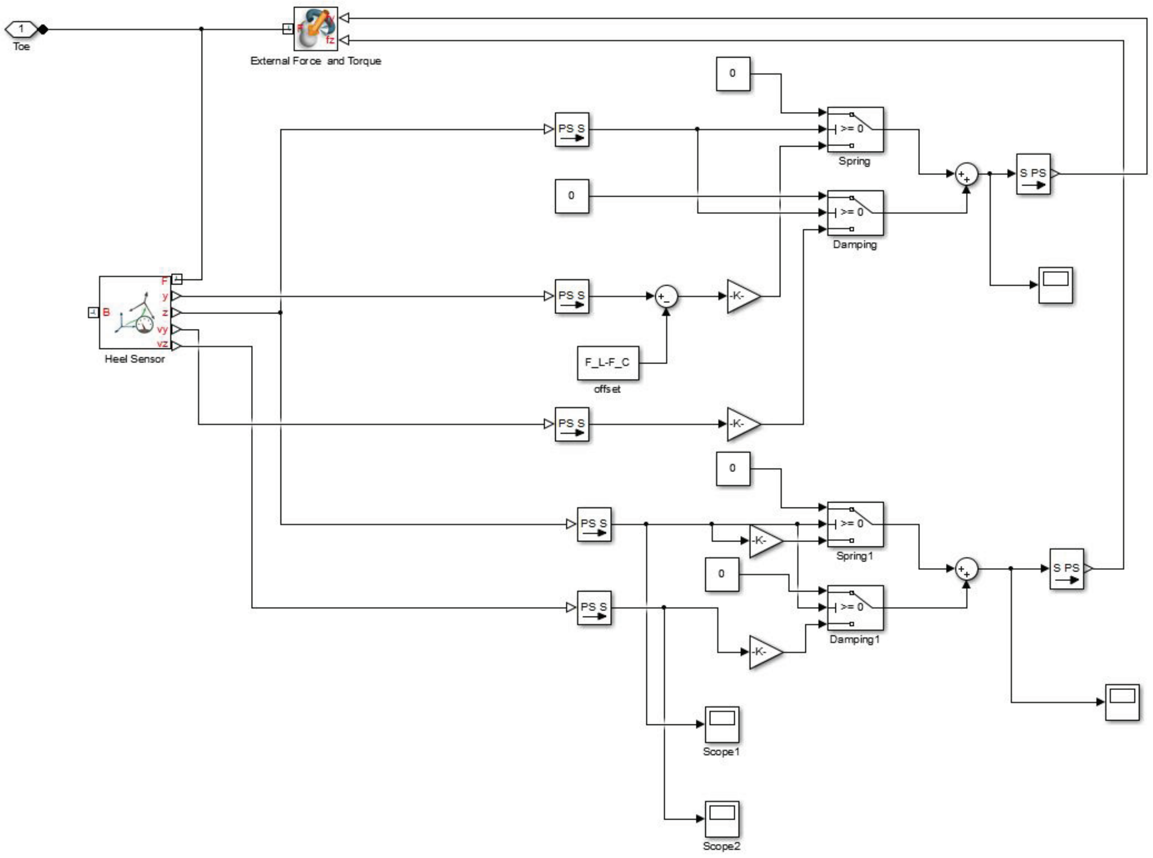

Figure C.4: Subsystem 2.1 - Toe Ground Reaction Force of the Stance Foot

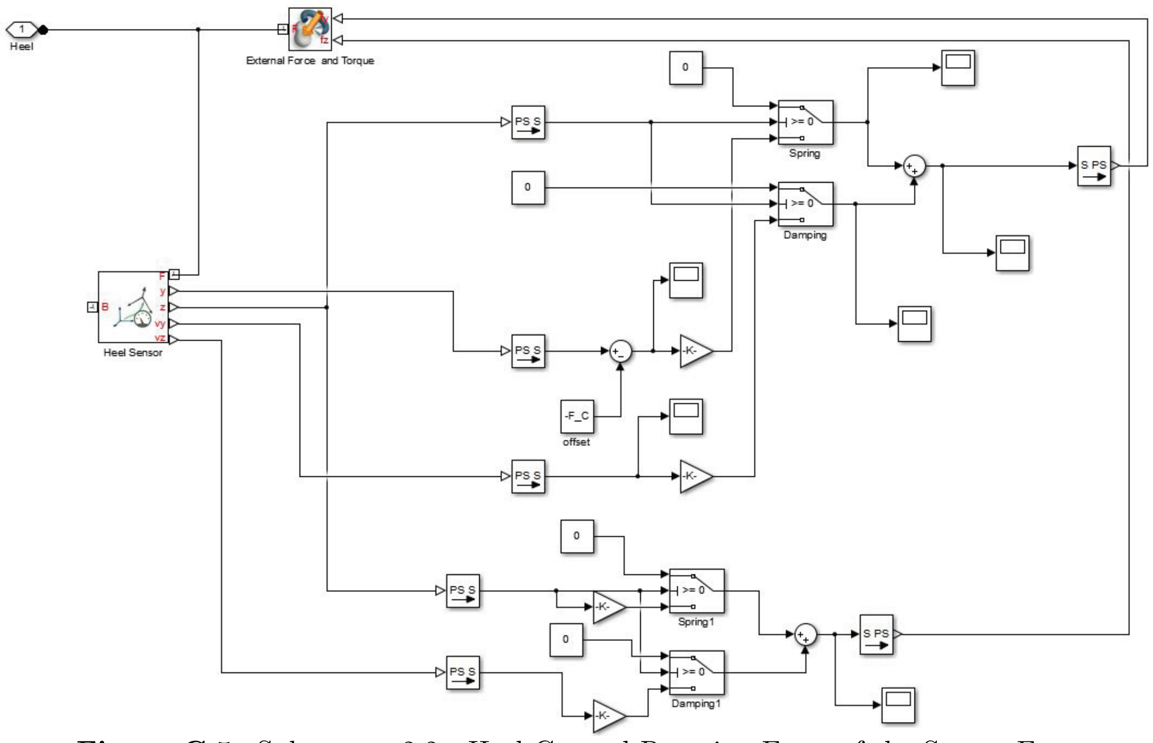

Figure C.5: Subsystem 2.2 - Heel Ground Reaction Force of the Stance Foot

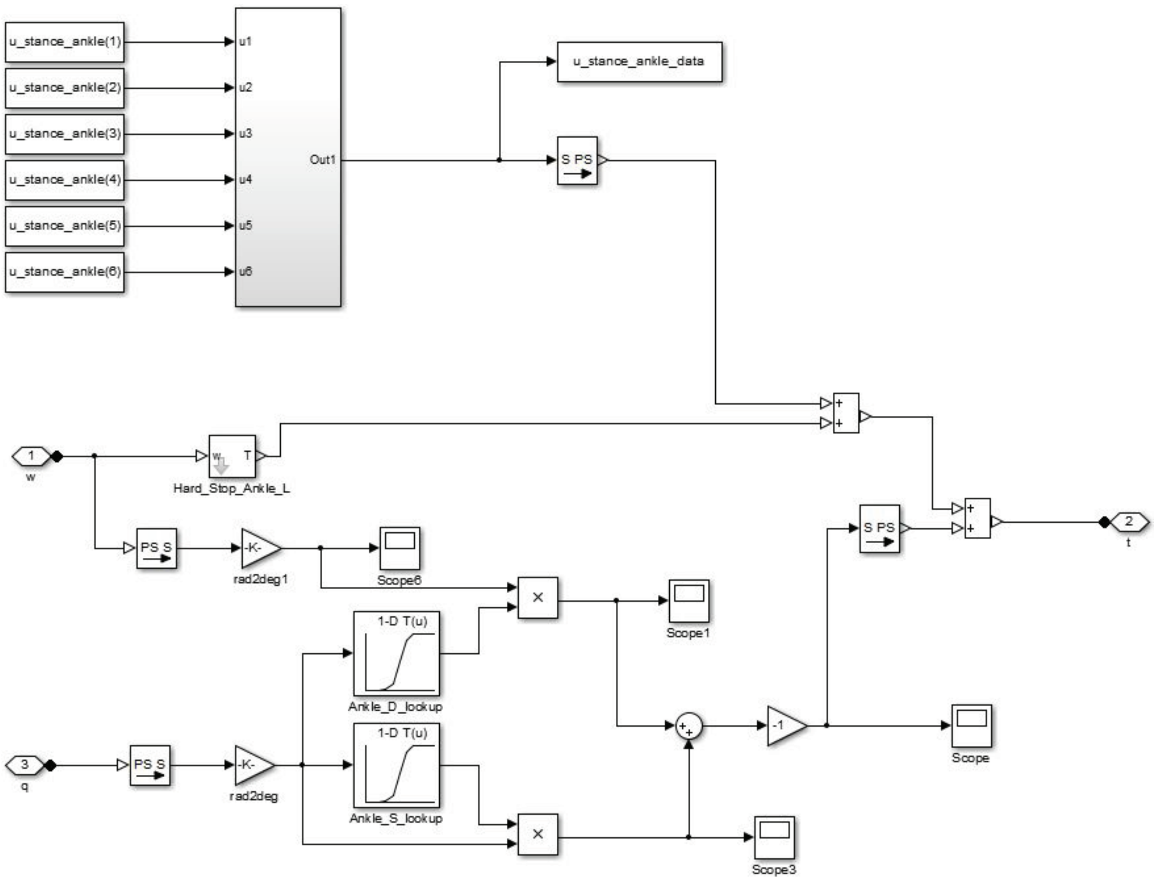

Figure C.6: Subsystem 3 - Stance Ankle Model and Its Joint Moment Control

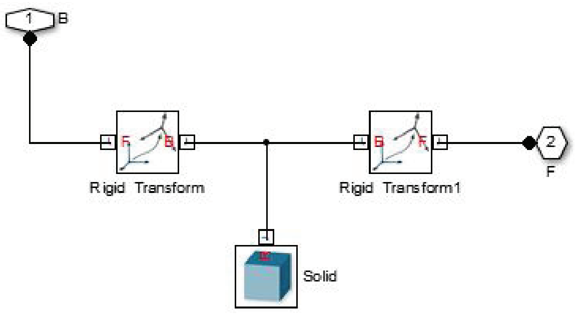

Figure C.7: Subsystem 4 - Model of the Shank

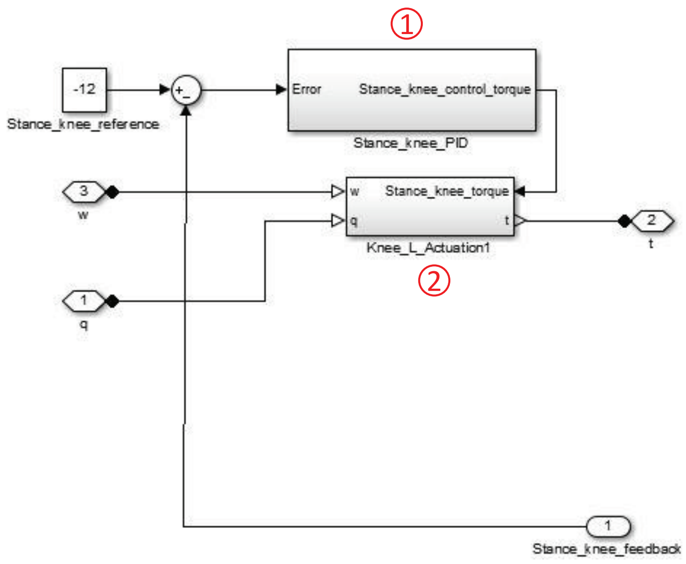

Figure C.8: Subsystem 5 - Stance Knee Model and Its Joint Moment Actuation

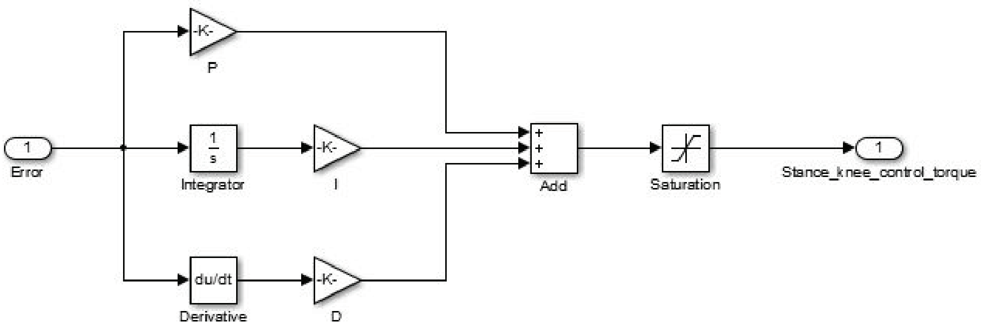

Figure C.9: Subsystem 5.1 - Stance Knee Joint Moment PID Control

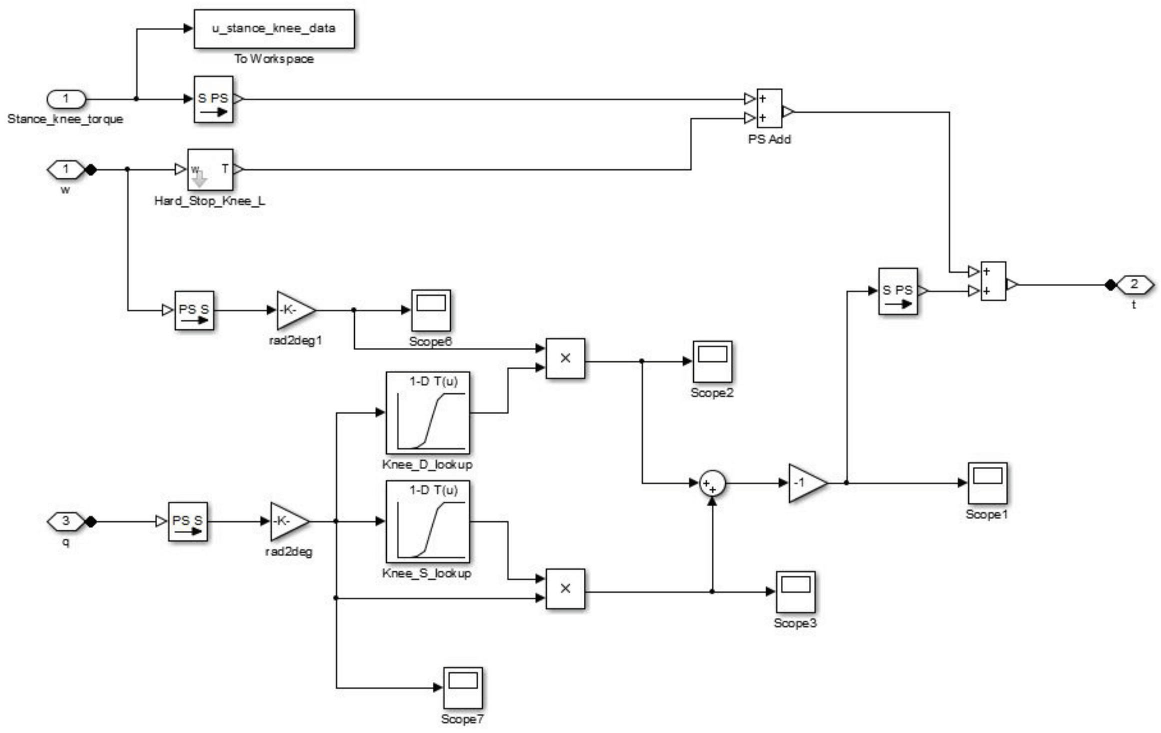

Figure C.10: Subsystem 5.2 - Stance Knee Joint Model

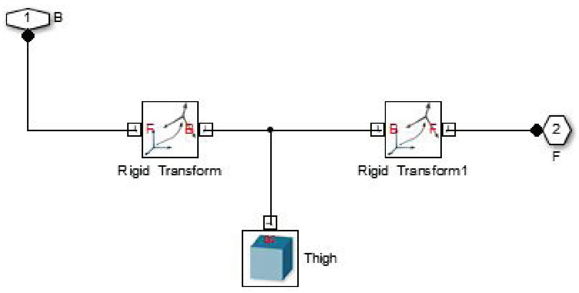

Figure C.11: Subsystem 6 - Model of the Thigh

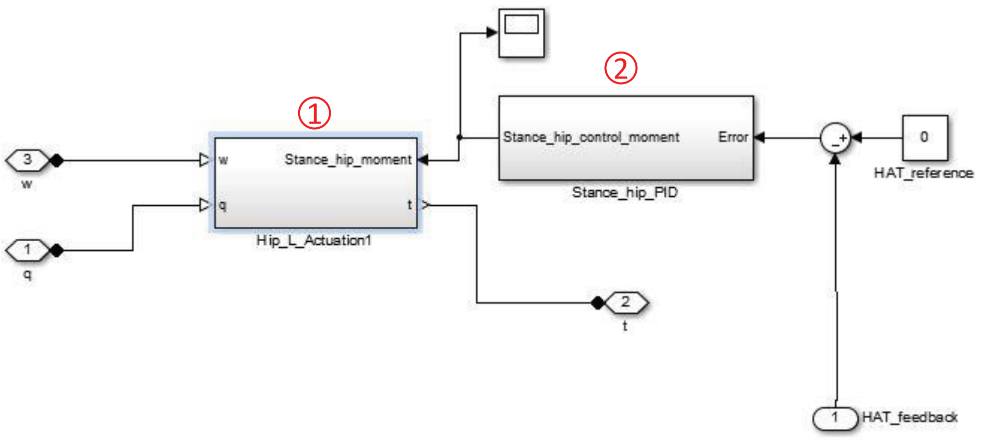

Figure C.12: Subsystem 7 - Stance Hip Model and Its Joint Moment Actuation

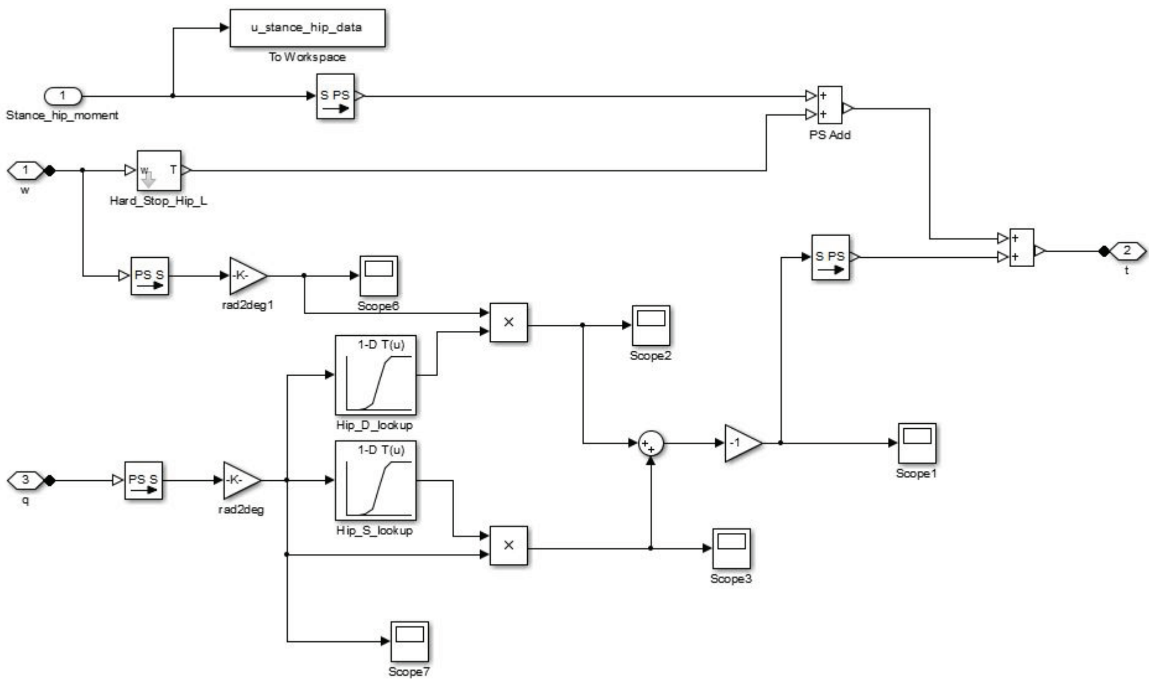

Figure C.13: Subsystem 7.1 - Stance Hip Joint Model

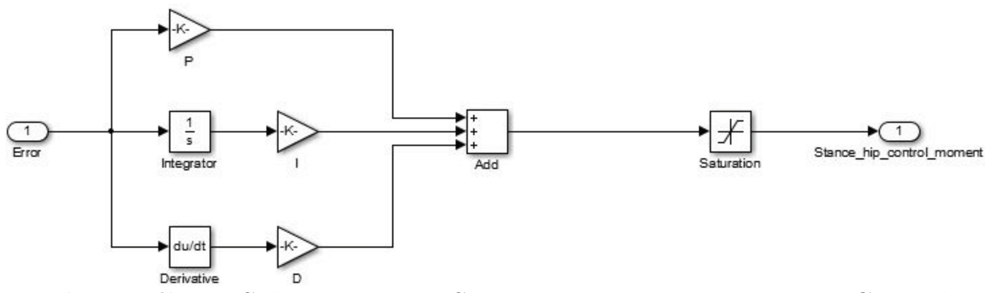

Figure C.14: Subsystem 7.2 - Stance Hip Joint Moment PID Control

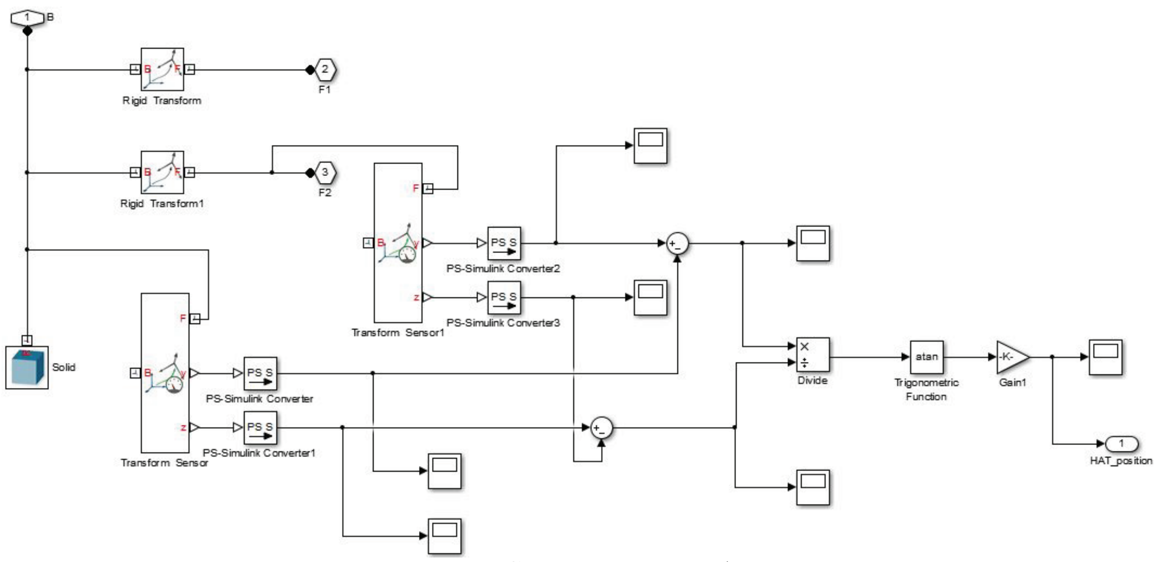

Figure C.15: Subsystem 8 - HAT Model

APPENDIX D

Control System MATLAB Code

This appendix shows the control program source code developed using MATLAB. Section. D.1 shows the control program for the Single Support Phase simulation at SSWS. Section. D.2 shows the optimization source code for the Single Support Phase simulation at SSWS. Section. D.3 shows the control program for the Double Support Phase simulation at SSWS. Section. D.4 shows the optimization algorithm source code for the Double Support Phase simulation at SSWS. The variable speed and pathological gait simulation source codes are similar to SSWS, therefore they are not shown in this appendix.

D.1 Control Program for the Single Support Phase Simulation at SSWS

```
clear all
clc

% Record the history of the command window
diary on
diary CommandWindowHistory.txt

% Start a timer to record how long the first iteration
% of optimization takes
tic

% Start the parallel computing
parpool

% Current time is set to be 0
tnow = 0;

% Open human gait plant model and internal MPC model
open_system('Internal_MPC_model_20_original');
save_system('Internal_MPC_model_20_original',
'Internal_MPC_model_20');
open_system('Human_gait_plant_model_13_original');
save_system('Human_gait_plant_model_13_original',
'Human_gait_plant_model_13');
```

```
sys = 'Internal_MPC_model_20';
open_system(sys);

% Set the designed variables for the MPC optimization
p = sdo.getParameterFromModel(sys,{'lagStanceAnkle',
'lagSwingAnkle','lagSwingKnee','lagSwingHip',
'w_stance_ankle_i','w_swing_knee_i','w_swing_hip_i'});
% Set the maximum and minimum allowable values of the
% design variables
p(1).Minimum = [-100 -300 -100 -30 -10];
p(2).Minimum = -20*ones(1,14);
p(3).Minimum = -40*ones(1,18);
p(4).Minimum = -55*ones(1,18);
p(1).Maximum = [100 300 100 30 10];
p(2).Maximum = 20*ones(1,14);
p(3).Maximum = 40*ones(1,18);
p(4).Maximum = 55*ones(1,18);
temp1 = p(5).Value*0.95;
temp2 = p(5).Value*1.05;
if temp1 < temp2
p(5).Minimum = temp1;
p(5).Maximum = temp2;
else
p(5).Minimum = temp2;
p(5).Maximum = temp1;
end
temp1 = p(6).Value*0.8;
temp2 = p(6).Value*1.2;
if temp1 < temp2
p(6).Minimum = temp1;
p(6).Maximum = temp2;
else
p(6).Minimum = temp2;
p(6).Maximum = temp1;
end
temp1 = p(7).Value*0.8;
temp2 = p(7).Value*1.2;
if temp1 < temp2
p(7).Minimum = temp1;
p(7).Maximum = temp2;
else
p(7).Minimum = temp2;
p(7).Maximum = temp1;
end
% Set the scale of the design variables
```

```
p(1).Scale = [100 300 100 30 10];
p(2).Scale = 20*ones(1,14);
p(3).Scale = 40*ones(1,18);
p(4).Scale = 55*ones(1,18);
p(5).Scale = abs(p(5).Value);
p(6).Scale = abs(p(6).Value);
p(7).Scale = abs(p(7).Value);

% Create a simulation tester
simulator = sdo.SimulationTest(sys);

% Define the optimization algorithm
evalDesign = @(p) objfun8(p,simulator);

% Select options for the optimizer
opt = sdo.OptimizeOptions;
opt.MethodOptions.Algorithm = 'interior-point';
opt.MethodOptions.UseParallel = 'always';
%opt.MethodOptions.DiffMaxChange = 4;
opt.MethodOptions.TypicalX = [];
opt.MethodOptions.FinDiffType = 'central';
opt.MethodOptions.Hessian = 'bfgs';
opt.MethodOptions.TolFun = 0.1;
opt.MethodOptions.ObjectiveLimit = 5;
opt.MethodOptions.MaxIter = 20;
opt.MethodOptions.TolX = 0.02;
opt.UseParallel = 'always';
opt.OptimizedModel = sys;

% Start the optimization
[pOpt,optInfo] = sdo.optimize(evalDesign,p,opt);

% Save the optimization results of the first iteration
save('Internal_MPC_model_20_iter_1');

u1 = pOpt(1).Value;
u2 = pOpt(2).Value;
u3 = pOpt(3).Value;
u4 = pOpt(4).Value;
u5 = pOpt(5).Value;
u6 = pOpt(6).Value;
u7 = pOpt(7).Value;

save('Optimized_control_input_DS_iter_1','u1','u2','u3','u4',
'u5','u6','u7');
```

```
clear all

% Load the human gait anthropometric data and the optimized
% control input
Plant = 'Human_gait_plant_model_13';
load Optimized_control_input_DS_iter_1

% Send the optimized values of the design variables to the
% human gait plant model
sdo.setValueInModel(Plant, 'u_stance_ankle',u1);
sdo.setValueInModel(Plant, 'u_swing_ankle',u2);
sdo.setValueInModel(Plant, 'u_swing_knee',u3);
sdo.setValueInModel(Plant, 'u_swing_hip',u4);
sdo.setValueInModel(Plant, 'w_stance_ankle',u5);
sdo.setValueInModel(Plant, 'w_swing_knee',u6);
sdo.setValueInModel(Plant, 'w_swing_hip',u7);

u_stance_ankle_data_timeseries = timeseries;
u_stance_ankle_data_timeseries.Time = 0;
u_stance_ankle_data_timeseries.Data = 0;
u_swing_ankle_data_timeseries = timeseries;
u_swing_ankle_data_timeseries.Time = 0;
u_swing_ankle_data_timeseries.Data = 0;
u_swing_knee_data_timeseries = timeseries;
u_swing_knee_data_timeseries.Time = 0;
u_swing_knee_data_timeseries.Data = 0;
u_swing_hip_data_timeseries = timeseries;
u_swing_hip_data_timeseries.Time = 0;
u_swing_hip_data_timeseries.Data = 0;

clear u1 u2 u3 u4 u5 u6 u7;

% Simulate plant model using the opitmized control inputs
sim(Plant);
save('Human_gait_plant_model_13_iter_1');

% Save the states of the plant model at the end of the first
% time step. They will be used as the initial state of the
% plant model for the next time step
stance_ankle_p_out1 = stance_ankle_p_out(831);
stance_ankle_w_out1 = stance_ankle_w_out(831);
stance_hip_p_out1 = stance_hip_p_out(831);
stance_hip_w_out1 = stance_hip_w_out(831);
stance_knee_p_out1 = stance_knee_p_out(831);
stance_knee_w_out1 = stance_knee_w_out(831);
swing_ankle_p_out1 = swing_ankle_p_out(831);
```

```
swing_ankle_w_out1 = swing_ankle_w_out(831);
swing_knee_p_out1 = swing_knee_p_out(831);
swing_knee_w_out1 = swing_knee_w_out(831);
swing_hip_p_out1 = swing_hip_p_out(831);
swing_hip_w_out1 = swing_hip_w_out(831);
Planar_joint_x_p_out1 = Planar_joint_x_p_out(831);
Planar_joint_x_v_out1 = Planar_joint_x_v_out(831);
Planar_joint_y_p_out1 = Planar_joint_y_p_out(831);
Planar_joint_y_v_out1 = Planar_joint_y_v_out(831);
Planar_joint_z_p_out1 = Planar_joint_z_p_out(831);
Planar_joint_z_w_out1 = Planar_joint_z_w_out(831);

save('IC_for_next_opt_iter_1','Planar_joint_x_p_out1',
'Planar_joint_x_v_out1','Planar_joint_y_p_out1',
'Planar_joint_y_v_out1','Planar_joint_z_p_out1',
'Planar_joint_z_w_out1','stance_ankle_p_out1',
'stance_ankle_w_out1','stance_hip_p_out1',
'stance_hip_w_out1','stance_knee_p_out1',
'stance_knee_w_out1','swing_ankle_p_out1',
'swing_ankle_w_out1','swing_hip_p_out1',
'swing_hip_w_out1','swing_knee_p_out1','swing_knee_w_out1');
save('Control_input_history_iter_1','u_stance_ankle_data',
'u_stance_knee_data','u_stance_hip_data',
'u_swing_ankle_data','u_swing_knee_data','u_swing_hip_data');

toc

for i = 1:46
tic

% Clear all the variables except for the counter i to perform
% optimization for internal model again
clearvars -except i

% Regress the current simulation time
tnow = .0083*i;

% Open human gait plant model and internal MPC model
sys = 'Internal_MPC_model_20';
open_system(sys);

% Set the designed variables for the MPC optimization
filename = ['Optimized_control_input_DS_iter_' num2str(i)];
load(filename);
% Set the maximum, minimum allowable values and the scales
% of the design variables
```

```
p = sdo.getParameterFromModel(sys,{'lagStanceAnkle',
'lagSwingAnkle','lagSwingKnee','lagSwingHip'});
p(1).Minimum = [u1(1)-10 u1(2)-30 u1(3)-10 u1(4)-3 u1(5)-1];
p(2).Minimum = u2-2*ones(1,14);
p(3).Minimum = u3-4*ones(1,18);
p(4).Minimum = u4-5*ones(1,18);
p(1).Maximum = [u1(1)+10 u1(2)+30 u1(3)+10 u1(4)+3 u1(5)+1];
p(2).Maximum = u2+2*ones(1,14);
p(3).Maximum = u3+4*ones(1,18);
p(4).Maximum = u4+5*ones(1,18);

p1_Min = [-100 -300 -100 -30 -10];
p1_Max = [100 300 100 30 10];

for kk = 1:5
if p(1).Minimum(kk) < p1_Min(kk)
p(1).Minimum(kk) = p1_Min(kk);
end
if p(1).Maximum(kk) > p1_Max(kk)
p(1).Maximum(kk) = p1_Max(kk);
end
if abs(p(1).Maximum(kk)) >= abs(p(1).Minimum(kk))
p(1).Scale(kk) = abs(p(1).Maximum(kk));
else
p(1).Scale(kk) = abs(p(1).Minimum(kk));
end
end

for kk = 2:4
if kk == 2
laguerre_coefficient_no_count = 14;
p_Min = -20;
p_Max = 20;
elseif kk == 3
laguerre_coefficient_no_count = 18;
p_Min = -40;
p_Max = 40;
else
laguerre_coefficient_no_count = 18;
p_Min = -55;
p_Max = 55;
end
for kkk = 1:laguerre_coefficient_no_count
if p(kk).Minimum(kkk) < p_Min
p(kk).Minimum(kkk) = p_Min;
end
```

```
if p(kk).Maximum(kkk) > p_Max
p(kk).Maximum(kkk) = p_Max;
end
if abs(p(kk).Maximum(kkk)) >= abs(p(kk).Minimum(kkk))
p(kk).Scale(kkk) = abs(p(kk).Maximum(kkk));
else
p(kk).Scale(kkk) = abs(p(kk).Minimum(kkk));
end
end
end

p(1).Value = u1;
p(2).Value = u2;
p(3).Value = u3;
p(4).Value = u4;

% Load the saved states of the plant model at the end
% of the previous sample time
filename = ['IC_for_next_opt_iter_' num2str(i)];
load(filename);

% Define the initial condition for the internal MPC
% model for the next optimization step
sdo.setValueInModel(sys,'beginTime',tnow);
sdo.setValueInModel(sys,'p_stance_ankle_i',
stance_ankle_p_out1);
sdo.setValueInModel(sys,'w_stance_ankle_i',
stance_ankle_w_out1);
sdo.setValueInModel(sys,'p_stance_knee_i',
stance_knee_p_out1);
sdo.setValueInModel(sys,'w_stance_knee_i',
stance_knee_w_out1);
sdo.setValueInModel(sys,'p_stance_hip_i',
stance_hip_p_out1);
sdo.setValueInModel(sys,'w_stance_hip_i',
stance_hip_w_out1);
sdo.setValueInModel(sys,'p_swing_ankle_i',
swing_ankle_p_out1);
sdo.setValueInModel(sys,'w_swing_ankle_i',
swing_ankle_w_out1);
sdo.setValueInModel(sys,'p_swing_knee_i',
swing_knee_p_out1);
sdo.setValueInModel(sys,'w_swing_knee_i',
swing_knee_w_out1);
sdo.setValueInModel(sys,'p_swing_hip_i',
swing_hip_p_out1);
```

```
sdo.setValueInModel(sys,'w_swing_hip_i',
swing_hip_w_out1);
sdo.setValueInModel(sys,'x_planar_p_i',
Planar_joint_x_p_out1);
sdo.setValueInModel(sys,'x_planar_v_i',
Planar_joint_x_v_out1);
sdo.setValueInModel(sys,'y_planar_p_i',
Planar_joint_y_p_out1);
sdo.setValueInModel(sys,'y_planar_v_i',
Planar_joint_y_v_out1);
sdo.setValueInModel(sys,'z_planar_p_i',
Planar_joint_z_p_out1);
sdo.setValueInModel(sys,'z_planar_v_i',
Planar_joint_z_w_out1);
save_system('Internal_MPC_model_20');

% Create a simulation tester
simulator = sdo.SimulationTest(sys);

% Define the optimization algorithm
evalDesign = @(p) objfun8(p,simulator);

% Select options for the optimizer
opt = sdo.OptimizeOptions;
opt.MethodOptions.Algorithm = 'interior-point';
opt.MethodOptions.UseParallel = 'always';
% opt.MethodOptions.DiffMaxChange = 2;
% opt.MethodOptions.TypicalX = u;
opt.MethodOptions.FinDiffType = 'central';
opt.MethodOptions.TolFun = 0.1;
% opt.MethodOptions.ScaleProblem = 'obj-and-constr';
% opt.MethodOptions.AlwaysHonorConstraints = 'none';
opt.MethodOptions.ObjectiveLimit = 5;
opt.MethodOptions.MaxIter = 20;
opt.MethodOptions.TolX = 0.02;
opt.UseParallel = 'always';
opt.OptimizedModel = sys;

clear u1 u2 u3 u4 u5 u6 u7 j
clear stance_ankle_p_out1 stance_ankle_w_out1
stance_hip_p_out1 stance_hip_w_out1 stance_knee_p_out1
stance_knee_w_out1 swing_ankle_p_out1 swing_ankle_w_out1
swing_hip_p_out1 swing_hip_w_out1 swing_knee_p_out1
swing_knee_w_out1 Planar_joint_x_p_out1 Planar_joint_x_v_out1
clear Planar_joint_y_p_out1 Planar_joint_y_v_out1
 Planar_joint_z_p_out1 Planar_joint_z_w_out1
```

```
clear u_stance_ankle_data_end u_stance_hip_data_end
u_stance_knee_data_end u_swing_ankle_data_end
u_swing_hip_data_end u_swing_knee_data_end
clear kk kkk laguerre_coefficient_no_count p1_Max p1_Min
p_Max p_Min

[pOpt, optInfo] = sdo.optimize(evalDesign,p,opt);

filename = ['Internal_MPC_model_20_iter_' num2str(i+1)];
save(filename)

u1 = pOpt(1).Value;
u2 = pOpt(2).Value;
u3 = pOpt(3).Value;
u4 = pOpt(4).Value;
filename = ['Optimized_control_input_DS_iter_' num2str(i+1)];
save(filename,'u1','u2','u3','u4');

% Load the reuqired parameter to be able to run the plant
% model simulation
clearvars -except i
%      load Double_stance_phase_plant_model_4_parameter
Plant = 'Human_gait_plant_model_13';
% load IC_for_next_opt
filename = ['Optimized_control_input_DS_iter_' num2str(i+1)];
load(filename);
filename = ['Control_input_history_iter_' num2str(i)];
load(filename);

% Simulate model and save the end states to be used as the
% initial conditions for the next time step
sdo.setValueInModel(Plant,'BeginTimePlantModel',i*.0083);
sdo.setValueInModel(Plant,'u_stance_ankle',u1);
sdo.setValueInModel(Plant,'u_swing_ankle',u2);
sdo.setValueInModel(Plant,'u_swing_knee',u3);
sdo.setValueInModel(Plant,'u_swing_hip',u4);

u_stance_ankle_data_timeseries = timeseries;
u_stance_ankle_data_timeseries.Time = 0:.00001:i*.0083;
u_stance_ankle_data_timeseries.Data =
  u_stance_ankle_data(1:(i*830+1));
u_swing_ankle_data_timeseries = timeseries;
u_swing_ankle_data_timeseries.Time = 0:.00001:i*.0083;
u_swing_ankle_data_timeseries.Data =
  u_swing_ankle_data(1:(i*830+1));
u_swing_knee_data_timeseries = timeseries;
```

```
u_swing_knee_data_timeseries.Time = 0:.00001:i*.0083;
u_swing_knee_data_timeseries.Data =
  u_swing_knee_data(1:(i*830+1));
u_swing_hip_data_timeseries = timeseries;
u_swing_hip_data_timeseries.Time = 0:.00001:i*.0083;
u_swing_hip_data_timeseries.Data =
  u_swing_hip_data(1:(i*830+1));

clear u1 u2 u3 u4
clear u_stance_ankle_data u_stance_knee_data u_stance_hip_data
u_swing_ankle_data u_swing_knee_data u_swing_hip_data

sim(Plant);

% Save all the variable after one iteration simulation of
% plant model into a file
filename = ['Human_gait_plant_model_13_iter_' num2str(i+1)];
save(filename);

stance_ankle_p_out1 = stance_ankle_p_out((i+1)*830+1);
stance_ankle_w_out1 = stance_ankle_w_out((i+1)*830+1);
stance_hip_p_out1 = stance_hip_p_out((i+1)*830+1);
stance_hip_w_out1 = stance_hip_w_out((i+1)*830+1);
stance_knee_p_out1 = stance_knee_p_out((i+1)*830+1);
stance_knee_w_out1 = stance_knee_w_out((i+1)*830+1);
swing_ankle_p_out1 = swing_ankle_p_out((i+1)*830+1);
swing_ankle_w_out1 = swing_ankle_w_out((i+1)*830+1);
swing_knee_p_out1 = swing_knee_p_out((i+1)*830+1);
swing_knee_w_out1 = swing_knee_w_out((i+1)*830+1);
swing_hip_p_out1 = swing_hip_p_out((i+1)*830+1);
swing_hip_w_out1 = swing_hip_w_out((i+1)*830+1);
Planar_joint_x_p_out1 = Planar_joint_x_p_out((i+1)*830+1);
Planar_joint_x_v_out1 = Planar_joint_x_v_out((i+1)*830+1);
Planar_joint_y_p_out1 = Planar_joint_y_p_out((i+1)*830+1);
Planar_joint_y_v_out1 = Planar_joint_y_v_out((i+1)*830+1);
Planar_joint_z_p_out1 = Planar_joint_z_p_out((i+1)*830+1);
Planar_joint_z_w_out1 = Planar_joint_z_w_out((i+1)*830+1);

filename = ['IC_for_next_opt_iter_' num2str(i+1)];
save(filename,'Planar_joint_x_p_out1','Planar_joint_x_v_out1',
'Planar_joint_y_p_out1','Planar_joint_y_v_out1',
'Planar_joint_z_p_out1','Planar_joint_z_w_out1',
'stance_ankle_p_out1','stance_ankle_w_out1',
'stance_hip_p_out1','stance_hip_w_out1',
'stance_knee_p_out1','stance_knee_w_out1',
'swing_ankle_p_out1','swing_ankle_w_out1',
'swing_hip_p_out1','swing_hip_w_out1','swing_knee_p_out1',
'swing_knee_w_out1');

filename = ['Control_input_history_iter_' num2str(i+1)];
save(filename,'u_stance_ankle_data','u_stance_knee_data',
'u_stance_hip_data','u_swing_ankle_data','u_swing_knee_data',
'u_swing_hip_data');

toc

end
```

D.2 Optimization Code for the Single Support Phase Simulation at SSWS

```
function design = objfun8(p,simulator)

% Define design variables
simulator.Parameters = p;
simulator = sim(simulator);

% Obtain the optimization results
StepLength1 = get(simulator.LoggedData,'StepLength');
VerticalPos1 = get(simulator.LoggedData,'VerticalPos');
VerticalPosEnd1 = get(simulator.LoggedData,'VerticalPosEnd');

[p(1).Value p(2).Value p(3).Value p(4).Value]

design.F = (0.7345-StepLength1)^2*100000
design.Cleq = [-VerticalPos1;VerticalPosEnd1 -0.01];

end
```

D.3 Control Program for the Double Support Phase Simulation at SSWS

```
clear all
clc

% Record the history of the command window
diary on
diary CommandWindowHistory.txt
```

```
% Start a timer to record how long the first iteration of
% optimization takes
tic

% Start the parallel computing
parpool

% Current time is set to be 0
tnow = 0;

% Open human gait plant model and internal MPC model
open_system('Double_support_phase_internal_model_22_original')
save_system('Double_support_phase_internal_model_22_origina',
   'Double_support_phase_internal_model_22');
open_system('Double_stance_phase_plant_model_15_original');
save_system('Double_stance_phase_plant_model_15_original',
   'Double_stance_phase_plant_model_15');

sys = 'Double_support_phase_internal_model_22';
open_system(sys);

% Set the designed variables for the MPC optimization
p = sdo.getParameterFromModel(sys,{'lagStanceAnkle',
   'lagStanceKnee','lagSwingAnkle','lagSwingKnee',
   'w_stance_ankle_i', 'w_stance_knee_i','w_stance_hip_i',
   'w_swing_knee_i','w_swing_hip_i'});
% Set the maximum and minimum allowable values of the
% design variables
p(1).Minimum = -150*ones(1,10);
p(2).Minimum = -30*ones(1,21);
p(3).Minimum = -8*ones(1,18);
p(4).Minimum = -15*ones(1,15);
p(1).Maximum = 150*ones(1,10);
p(2).Maximum = 30*ones(1,21);
p(3).Maximum = 8*ones(1,18);
p(4).Maximum = 15*ones(1,15);
temp1 = p(5).Value*0.9;
temp2 = p(5).Value*1.1;
if temp1 < temp2
p(5).Minimum = temp1;
p(5).Maximum = temp2;
else
p(5).Minimum = temp2;
p(5).Maximum = temp1;
end
```

```
temp1 = p(6).Value*0.9;
temp2 = p(6).Value*1.1;
if temp1 < temp2
p(6).Minimum = temp1;
p(6).Maximum = temp2;
else
p(6).Minimum = temp2;
p(6).Maximum = temp1;
end
temp1 = p(7).Value*0.9;
temp2 = p(7).Value*1.1;
if temp1 < temp2
p(7).Minimum = temp1;
p(7).Maximum = temp2;
else
p(7).Minimum = temp2;
p(7).Maximum = temp1;
end
temp1 = p(8).Value*0.9;
temp2 = p(8).Value*1.1;
if temp1 < temp2
p(8).Minimum = temp1;
p(8).Maximum = temp2;
else
p(8).Minimum = temp2;
p(8).Maximum = temp1;
end
temp1 = p(9).Value*0.9;
temp2 = p(9).Value*1.1;
if temp1 < temp2
p(9).Minimum = temp1;
p(9).Maximum = temp2;
else
p(9).Minimum = temp2;
p(9).Maximum = temp1;
end
% Set the scale of the design variables
p(1).Scale = 150*ones(1,10);
p(2).Scale = 30*ones(1,21);
p(3).Scale = 8*ones(1,18);
p(4).Scale = 15*ones(1,15);
p(5).Scale = abs(p(5).Value);
p(6).Scale = abs(p(6).Value);
p(7).Scale = abs(p(7).Value);
p(8).Scale = abs(p(8).Value);
p(9).Scale = abs(p(9).Value);
```

```
% Create a simulation tester
simulator = sdo.SimulationTest(sys);

% Define the optimization algorithm
evalDesign = @(p) objfun4(p,simulator);

% Select options for the optimizer
opt = sdo.OptimizeOptions;
opt.MethodOptions.Algorithm = 'interior-point';
opt.MethodOptions.UseParallel = 'always';
% opt.MethodOptions.DiffMaxChange = 4;
opt.MethodOptions.TypicalX = [];
opt.MethodOptions.FinDiffType = 'central';
opt.MethodOptions.Hessian = 'bfgs';
opt.MethodOptions.TolFun = 0.1;
opt.MethodOptions.ObjectiveLimit = 5;
opt.MethodOptions.MaxIter = 20;
opt.MethodOptions.TolX = 0.02;
opt.UseParallel = 'always';
opt.OptimizedModel = sys;

% Start the optimization
[pOpt,optInfo] = sdo.optimize(evalDesign,p,opt);

% Save the optimization results of the first iteration
save('Double_support_phase_internal_model_22_iter_1');

u1 = pOpt(1).Value;
u2 = pOpt(2).Value;
u3 = pOpt(3).Value;
u4 = pOpt(4).Value;
u5 = pOpt(5).Value;
u6 = pOpt(6).Value;
u7 = pOpt(7).Value;
u8 = pOpt(8).Value;
u9 = pOpt(9).Value;
save('Optimized_control_input_DS_iter_1','u1','u2','u3',
   'u4','u5','u6','u7','u8','u9');

clear all

% Load the human gait anthropometric data and the optimized
% control input
Plant = 'Double_stance_phase_plant_model_15';
load Optimized_control_input_DS_iter_1
```

```
% Send the optimized values of the design variables to
% the human gait plant model
sdo.setValueInModel(Plant,'u_stance_ankle',u1);
sdo.setValueInModel(Plant,'u_stance_knee',u2);
sdo.setValueInModel(Plant,'u_swing_ankle',u3);
sdo.setValueInModel(Plant,'u_swing_knee',u4);
sdo.setValueInModel(Plant,'w_stance_ankle',u5);
sdo.setValueInModel(Plant,'w_stance_knee',u6);
sdo.setValueInModel(Plant,'w_stance_hip',u7);
sdo.setValueInModel(Plant,'w_swing_knee',u8);
sdo.setValueInModel(Plant,'w_swing_hip',u9);

u_stance_ankle_data_timeseries = timeseries;
u_stance_ankle_data_timeseries.Time = 0;
u_stance_ankle_data_timeseries.Data = 0;
u_stance_knee_data_timeseries = timeseries;
u_stance_knee_data_timeseries.Time = 0;
u_stance_knee_data_timeseries.Data = 0;
u_swing_ankle_data_timeseries = timeseries;
u_swing_ankle_data_timeseries.Time = 0;
u_swing_ankle_data_timeseries.Data = 0;
u_swing_knee_data_timeseries = timeseries;
u_swing_knee_data_timeseries.Time = 0;
u_swing_knee_data_timeseries.Data = 0;

clear u1 u2 u3 u4 u5 u6 u7 u8 u9

% Simulate plant model using the opitmized control inputs
sim(Plant);
save('Double_stance_phase_plant_model_15_iter_1')

% Save the states of the plant model at the end of the
% first time step. They will be used as the initial state
% of the plant model for the next time step
stance_ankle_p_out1 = stance_ankle_p_out(831);
stance_ankle_w_out1 = stance_ankle_w_out(831);
stance_hip_p_out1 = stance_hip_p_out(831);
stance_hip_w_out1 = stance_hip_w_out(831);
stance_knee_p_out1 = stance_knee_p_out(831);
stance_knee_w_out1 = stance_knee_w_out(831);
swing_ankle_p_out1 = swing_ankle_p_out(831);
swing_ankle_w_out1 = swing_ankle_w_out(831);
swing_knee_p_out1 = swing_knee_p_out(831);
swing_knee_w_out1 = swing_knee_w_out(831);
swing_hip_p_out1 = swing_hip_p_out(831);
```

```
swing_hip_w_out1 = swing_hip_w_out(831);
Planar_joint_x_p_out1 = Planar_joint_x_p_out(831);
Planar_joint_x_v_out1 = Planar_joint_x_v_out(831);
Planar_joint_y_p_out1 = Planar_joint_y_p_out(831);
Planar_joint_y_v_out1 = Planar_joint_y_v_out(831);
Planar_joint_z_p_out1 = Planar_joint_z_p_out(831);
Planar_joint_z_w_out1 = Planar_joint_z_w_out(831);

save('IC_for_next_opt_iter_1','Planar_joint_x_p_out1',
   'Planar_joint_x_v_out1','Planar_joint_y_p_out1',
   'Planar_joint_y_v_out1','Planar_joint_z_p_out1',
   'Planar_joint_z_w_out1','stance_ankle_p_out1',
   'stance_ankle_w_out1','stance_hip_p_out1',
   'stance_hip_w_out1','stance_knee_p_out1',
   'stance_knee_w_out1','swing_ankle_p_out1',
   'swing_ankle_w_out1','swing_hip_p_out1',
   'swing_hip_w_out1','swing_knee_p_out1',
   'swing_knee_w_out1');
save('Control_input_history_iter_1',
   'u_stance_ankle_data','u_stance_knee_data',
   'u_stance_hip_data','u_swing_ankle_data',
   'u_swing_knee_data','u_swing_hip_data');

toc

for i = 1:19
tic

% Clear all the variables except for the counter
% i to perform optimization for internal model again
clearvars -except i

% Regress the current simulation time
tnow = .0083*i;

% Open human gait plant model and internal MPC model
sys = 'Double_support_phase_internal_model_22';
open_system(sys);

% Set the designed variables for the MPC optimization
filename =
   ['Optimized_control_input_DS_iter_' num2str(i)];
load(filename);
% Set the maximum, minimum allowable values and the
% scales of the design variables
p = sdo.getParameterFromModel(sys,{'lagStanceAnkle',
   'lagStanceKnee','lagSwingAnkle','lagSwingKnee'});
p(1).Minimum = u1-15*ones(1,10);
p(2).Minimum = u2-3*ones(1,21);
p(3).Minimum = u3-1*ones(1,18);
p(4).Minimum = u4-1.5*ones(1,15);
p(1).Maximum = u1+15*ones(1,10);
p(2).Maximum = u2+3*ones(1,21);
p(3).Maximum = u3+1*ones(1,18);
p(4).Maximum = u4+1.5*ones(1,15);

for kk = 1:4
if kk == 1
laguerre_coefficient_no_count = 10;
p_Min = -150;
p_Max = 150;
elseif kk == 2
laguerre_coefficient_no_count = 21;
p_Min = -30;
p_Max = 30;
elseif kk == 3
laguerre_coefficient_no_count = 18;
p_Min = -8;
p_Max = 8;
else
laguerre_coefficient_no_count = 15;
p_Min = -15;
p_Max = 15;
end
for kkk = 1:laguerre_coefficient_no_count
if p(kk).Minimum(kkk) < p_Min
p(kk).Minimum(kkk) = p_Min;
end
if p(kk).Maximum(kkk) > p_Max
p(kk).Maximum(kkk) = p_Max;
end
if abs(p(kk).Maximum(kkk)) >= abs(p(kk).Minimum(kkk))
p(kk).Scale(kkk) = abs(p(kk).Maximum(kkk));
else
p(kk).Scale(kkk) = abs(p(kk).Minimum(kkk));
end
end
end

p(1).Value = u1;
p(2).Value = u2;
p(3).Value = u3;
```

```
p(4).Value = u4;

% Load the saved states of the plant model at the
% end of the previous sample time
filename = ['IC_for_next_opt_iter_' num2str(i)];
load(filename);

% Define the initial condition for the internal MPC
% model for the next optimization step
sdo.setValueInModel(sys,'beginTime',tnow);
sdo.setValueInModel(sys,'p_stance_ankle_i',
  stance_ankle_p_out1);
sdo.setValueInModel(sys,'w_stance_ankle_i',
  stance_ankle_w_out1);
sdo.setValueInModel(sys,'p_stance_knee_i',
  stance_knee_p_out1);
sdo.setValueInModel(sys,'w_stance_knee_i',
  stance_knee_w_out1);
sdo.setValueInModel(sys,'p_stance_hip_i',
  stance_hip_p_out1);
sdo.setValueInModel(sys,'w_stance_hip_i',
  stance_hip_w_out1);
sdo.setValueInModel(sys,'p_swing_ankle_i',
  swing_ankle_p_out1);
sdo.setValueInModel(sys,'w_swing_ankle_i',
  swing_ankle_w_out1);
sdo.setValueInModel(sys,'p_swing_knee_i',
  swing_knee_p_out1);
sdo.setValueInModel(sys,'w_swing_knee_i',
  swing_knee_w_out1);
sdo.setValueInModel(sys,'p_swing_hip_i',
  swing_hip_p_out1);
sdo.setValueInModel(sys,'w_swing_hip_i',
  swing_hip_w_out1);
sdo.setValueInModel(sys,'Planar_joint_x_p_i',
  Planar_joint_x_p_out1);
sdo.setValueInModel(sys,'Planar_joint_x_v_i',
  Planar_joint_x_v_out1);
sdo.setValueInModel(sys,'Planar_joint_y_p_i',
  Planar_joint_y_p_out1);
sdo.setValueInModel(sys,'Planar_joint_y_v_i',
  Planar_joint_y_v_out1);
sdo.setValueInModel(sys,'Planar_joint_z_p_i',
  Planar_joint_z_p_out1);
sdo.setValueInModel(sys,'Planar_joint_z_w_i',
  Planar_joint_z_w_out1);
```

```
save_system('Double_support_phase_internal_model_22');

% Create a simulation tester
simulator = sdo.SimulationTest(sys);

% Define the optimization algorithm
evalDesign = @(p) objfun4(p,simulator);

% Select options for the optimizer
opt = sdo.OptimizeOptions;
opt.MethodOptions.Algorithm = 'interior-point';
opt.MethodOptions.UseParallel = 'always';
% opt.MethodOptions.DiffMaxChange = 2;
% opt.MethodOptions.TypicalX = u;
opt.MethodOptions.FinDiffType = 'central';
% opt.MethodOptions.TolFun = 0.1;
% opt.MethodOptions.ScaleProblem = 'obj-and-constr';
% opt.MethodOptions.AlwaysHonorConstraints = 'none';
opt.MethodOptions.ObjectiveLimit = 5;
opt.UseParallel = 'always';
opt.OptimizedModel = sys;

clear u1 u2 u3 u4 u5 u6 u7 u8 u9 j
clear stance_ankle_p_out1 stance_ankle_w_out1
  stance_hip_p_out1 stance_hip_w_out1 stance_knee_p_out1
  stance_knee_w_out1 swing_ankle_p_out1 swing_ankle_w_out1
  swing_hip_p_out1 swing_hip_w_out1 swing_knee_p_out1
  swing_knee_w_out1 Planar_joint_x_p_out1
  Planar_joint_x_v_out1
clear Planar_joint_y_p_out1 Planar_joint_y_v_out1
  Planar_joint_z_p_out1 Planar_joint_z_w_out1
clear u_stance_ankle_data_end u_stance_hip_data_end
  u_stance_knee_data_end u_swing_ankle_data_end
  u_swing_hip_data_end u_swing_knee_data_end
clear kk kkk laguerre_coefficient_no_count p1_Max
  p1_Min p_Max p_Min

[pOpt,optInfo] = sdo.optimize(evalDesign,p,opt);

filename =
  ['Double_support_phase_internal_model_22_iter_'
  num2str(i+1)];
save(filename)

u1 = pOpt(1).Value;
u2 = pOpt(2).Value;
```

```
u3 = pOpt(3).Value;
u4 = pOpt(4).Value;
filename =
  ['Optimized_control_input_DS_iter_' num2str(i+1)];
save(filename,'u1','u2','u3','u4');

% Load the reuqired parameter to be able to run the
% plant model simulation
clearvars -except i
%     load Double_stance_phase_plant_model_4_parameter
Plant = 'Double_stance_phase_plant_model_15';
% load IC_for_next_opt
filename = ['Optimized_control_input_DS_iter_' num2str(i+1)];
load(filename);
filename = ['Control_input_history_iter_' num2str(i)];
load(filename);

% Simulate model and save the end states to be used as
% the initial conditions for the next time step
sdo.setValueInModel(Plant,'BeginTimePlantModel',i*.0083);
sdo.setValueInModel(Plant,'u_stance_ankle',u1);
sdo.setValueInModel(Plant,'u_stance_knee',u2);
sdo.setValueInModel(Plant,'u_swing_ankle',u3);
sdo.setValueInModel(Plant,'u_swing_knee',u4);

u_stance_ankle_data_timeseries = timeseries;
u_stance_ankle_data_timeseries.Time = 0:.00001:i*.0083;
u_stance_ankle_data_timeseries.Data =
  u_stance_ankle_data(1:(i*830+1));
u_stance_knee_data_timeseries = timeseries;
u_stance_knee_data_timeseries.Time = 0:.00001:i*.0083;
u_stance_knee_data_timeseries.Data =
  u_stance_knee_data(1:(i*830+1));
u_swing_ankle_data_timeseries = timeseries;
u_swing_ankle_data_timeseries.Time = 0:.00001:i*.0083;
u_swing_ankle_data_timeseries.Data =
  u_swing_ankle_data(1:(i*830+1));
u_swing_knee_data_timeseries = timeseries;
u_swing_knee_data_timeseries.Time = 0:.00001:i*.0083;
u_swing_knee_data_timeseries.Data =
  u_swing_knee_data(1:(i*830+1));

clear u1 u2 u3 u4
clear u_stance_ankle_data u_stance_knee_data
  u_stance_hip_data u_swing_ankle_data
  u_swing_knee_data u_swing_hip_data
```

```
sim(Plant);

% Save all the variable after one iteration simulation
% of plant model into a file
filename =
  ['Double_stance_phase_plant_model_15_iter_' num2str(i+1)];
save(filename);

stance_ankle_p_out1 = stance_ankle_p_out((i+1)*830+1);
stance_ankle_w_out1 = stance_ankle_w_out((i+1)*830+1);
stance_hip_p_out1 = stance_hip_p_out((i+1)*830+1);
stance_hip_w_out1 = stance_hip_w_out((i+1)*830+1);
stance_knee_p_out1 = stance_knee_p_out((i+1)*830+1);
stance_knee_w_out1 = stance_knee_w_out((i+1)*830+1);
swing_ankle_p_out1 = swing_ankle_p_out((i+1)*830+1);
swing_ankle_w_out1 = swing_ankle_w_out((i+1)*830+1);
swing_knee_p_out1 = swing_knee_p_out((i+1)*830+1);
swing_knee_w_out1 = swing_knee_w_out((i+1)*830+1);
swing_hip_p_out1 = swing_hip_p_out((i+1)*830+1);
swing_hip_w_out1 = swing_hip_w_out((i+1)*830+1);
Planar_joint_x_p_out1 =
  Planar_joint_x_p_out((i+1)*830+1);
Planar_joint_x_v_out1 =
  Planar_joint_x_v_out((i+1)*830+1);
Planar_joint_y_p_out1 =
  Planar_joint_y_p_out((i+1)*830+1);
Planar_joint_y_v_out1 =
  Planar_joint_y_v_out((i+1)*830+1);
Planar_joint_z_p_out1 =
  Planar_joint_z_p_out((i+1)*830+1);
Planar_joint_z_w_out1 =
  Planar_joint_z_w_out((i+1)*830+1);

filename = ['IC_for_next_opt_iter_' num2str(i+1)];
save(filename,'Planar_joint_x_p_out1',
  'Planar_joint_x_v_out1','Planar_joint_y_p_out1',
  'Planar_joint_y_v_out1','Planar_joint_z_p_out1',
  'Planar_joint_z_w_out1','stance_ankle_p_out1',
  'stance_ankle_w_out1','stance_hip_p_out1',
  'stance_hip_w_out1','stance_knee_p_out1',
  'stance_knee_w_out1','swing_ankle_p_out1',
  'swing_ankle_w_out1','swing_hip_p_out1',
  'swing_hip_w_out1','swing_knee_p_out1',
  'swing_knee_w_out1');
```

```
filename =
   ['Control_input_history_iter_' num2str(i+1)];
save(filename,'u_stance_ankle_data',
   'u_stance_knee_data','u_stance_hip_data',
   'u_swing_ankle_data','u_swing_knee_data',
   'u_swing_hip_data');

toc

end
```

D.4 Optimization Code for the Double Support Phase Simulation at SSWS

```
function design = objfun4(p,simulator)

% Define design variables
simulator.Parameters = p;
simulator = sim(simulator);

% Obtain the optimization results
StepLength1 = get(simulator.LoggedData,'vel_COM');

[p(1).Value p(2).Value p(3).Value p(4).Value]

design.F = (1.3854-StepLength1)^2*100000

end
```