5-2012

Biarticular Muscles Influence Postural Responses: Implications for Treatment of Stiff-Knee Gait

Ashley Elizabeth Clark
aclark54@utk.edu

To the Graduate Council:

I am submitting herewith a thesis written by Ashley Elizabeth Clark entitled "Biarticular Muscles Influence Postural Responses: Implications for Treatment of Stiff-Knee Gait." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Biomedical Engineering.

Jeffrey A. Reinbolt, Major Professor

We have read this thesis and recommend its acceptance:

William R. Hamel, J.A.M. Boulet

Accepted for the Council:
Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Biarticular Muscles Influence Postural Responses: Implications for Treatment of Stiff-Knee Gait

A Thesis Presented for the
Master of Science
Degree
The University of Tennessee, Knoxville

Ashley Elizabeth Clark
May 2012

I dedicate this thesis to my
loving and supporting family.

ACKNOWLEDGEMENTS

I would like to sincerely thank my family, friends, and colleagues for their support and encouragement throughout my graduate work.

The dedication and guidance of Dr. Jeffrey A. Reinbolt was invaluable during this research. His enthusiasm, creativity, and experience in the field provided encouragement and direction throughout my education and research in biomechanics. I am extremely grateful to him for providing me with the opportunity to participate in research with the Reinbolt Research Group.

I also extend gratitude to Ajay Seth and the Neuromuscular Biomechanics Lab at Stanford University for their assistance, collaboration, and suggestions. Their dedication and professionalism was a great resource during my graduate work.

I would also like to thank the MABE faculty and staff for their support, mentorship, and instruction in the department and in the classroom. I would particularly like to thank Dr. William Hamel and Dr. J.A.M. Boulet for dedicating their time and commitment to serve on my graduate committee.

I wish to thank all of my colleagues in the Reinbolt Research Group for their support and collaboration during research and coursework. I am grateful to the group for providing an enjoyable graduate experience—both in lab and otherwise.

Finally, I would like to thank my parents, Gary and Barbara Clark, for their continued support throughout all of my endeavors. They are my true mentors and I continue to appreciate all that they have done.

ABSTRACT

Stiff knee gait is a prevalent and troublesome movement disorder among children with cerebral palsy, where peak knee flexion is diminished during swing phase. Rectus femoris transfer surgery, a common treatment for stiff-knee gait, reattaches the distal tendon of this biarticular, or two joint, muscle to a new site, such as the sartorius insertion on the tibia. Biarticular muscles play a unique role in motor control. As a biarticular muscle, rectus femoris may offer unrecognized benefits to maintain balance. This study uses musculoskeletal modeling and simulation to investigate the role of this biarticular muscle on balance recovery following support-surface translations. The hypothesis is that a preoperative simulation has increased balance recovery compared with two postoperative cases, and that a unilateral transfer simulation has improved balance recovery relative to a bilateral transfer.

The influence of rectus femoris transfer surgery on balance recovery was assessed with forward dynamic simulations of a patient with cerebral palsy. A 3-dimensional musculoskeletal model was scaled to represent the size of the patient using previously collected gait analysis data. This pre-surgical model was altered to represent unilateral and bilateral rectus femoris tendon transfers to the sartorius. The mechanism used to maintain balance was based on a muscle stretch-reflex control model, where reflex properties were found using optimization. Each 6s simulation included 0.25s of quiet standing, 0.35s of support-surface translation (6

cm in the anterior and posterior directions, with a peak velocity of 23 cm/s), and 5.4s of balance recovery. Balance recovery was evaluated by recording whole-body center of mass displacements relative to the support surface.

The preoperative simulations of balance recovery following support-surface translations maintained balance while both postoperative simulations did not. Moreover, the unilateral simulation maintained balance longer than the bilateral case in both support-surface translation directions. These findings support the hypothesis that the preoperative simulation has the best balance recovery, followed by the unilateral rectus femoris tendon transfer, and finally the bilateral transfer. This study's results suggest that rectus femoris tendon transfer reduces balance recovery compared with the preoperative case, illustrating the biomechanical advantage that biarticular muscles have in motor control.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I
INTRODUCTION

Stiff-Knee Gait: Troublesome Movement Abnormality in Children with Cerebral Palsy

Stiff-knee gait is a prevalent and troublesome movement abnormality among children with cerebral palsy, characterized by diminished and delayed peak knee flexion during the swing phase of gait. Cerebral palsy is a non-progressive, non-contagious disorder resulting from neural impairments, and can affect a patient's muscle tone, movement, and/or motor skills. While several types of cerebral palsy are common (spastic, dyskinetic, ataxic, hypotonic), there is no cure for the disorder. In 2003, the estimated lifetime costs for persons born in 2000 with cerebral palsy in the United States totaled $11.5 billion, with an average lifetime cost of $417,000 per patient (Honeycutt et al., 2004). A more recent study suggests an increase in the average lifetime costs associated with cerebral palsy, with estimates around $1.2 million per person in Europe (Kruse et al., 2009).

Stiff-knee gait is a symptom of spastic cerebral palsy which affects the sagittal-plane motion of the knee during gait (J. R. Gage, Perry, Hicks, Koop, & Werntz, 1987). More specifically, stiff-knee gait is characterized by diminished knee flexion during the swing phase of gait. Patients experiencing stiff-knee gait typically adopt energy-inefficient movements to compensate for reduced toe-clearance to avoid tripping or falling (J. R. Gage et al., 1987; Goldberg, Ounpuu, & Delp, 2003). The causes of stiff-knee gait are not well understood, but several factors are

believed to contribute the gait abnormality. Over-activity of the rectus femoris during swing phase is thought to induce an excessive knee extension moment, and is attributed as a primary cause of this gait pattern (Fox, Reinbolt, Ounpuu, & Delp, 2009; J. R. Gage et al., 1987; Goldberg et al., 2003; Perry, 1987).

While there is no cure for cerebral palsy, various surgical procedures can treat symptoms of the condition. Rectus femoris transfer surgery is a common treatment for stiff-knee gait which aims to decrease the muscle's knee extension moment and augment the knee flexion moment (Fox et al., 2009; J. R. Gage et al., 1987; Perry, 1987). This procedure reattaches the distal tendon of the rectus femoris from the patella to a new site posterior to the knee, such as the insertion of the sartorius on the tibia. While most studies report improved knee flexion, the outcomes of the procedure are inconsistent and variable (Fox et al., 2009; Goldberg et al., 2003). Some suggest that the transferred rectus femoris does not induce a knee flexion moment as intended, but rather, alters the capacity of the muscle to generate a knee extension moment while preserving its hip flexion moment (Goldberg, Anderson, Pandy, & Delp, 2004; Riewald & Delp, 1997).

Biarticular Muscles and Their Role in Cerebral Palsy

Biarticular muscles are defined as muscles that act across two joints, while monoarticular muscles act across a single joint. More specifically, the rectus femoris is considered to be a bifunctional biarticular muscle, demonstrating opposite actions at each of the two spanned joints: a hip flexor and a knee extensor. Due to their dichotomous nature, biarticular muscles play a unique role in motor control and are

hypothesized to act as energy transfer straps across joints (Metaxiotis, Wolf, & Doederlein, 2004). Bifunctional biarticular muscles, in particular, are thought to be responsible for the regulation of the distribution of moments across multiple joints (Schenau, Pratt, & Macpherson, 1994).

Previous studies suggest that the high level of control required by biarticular muscles makes them the most susceptible to the effects of cerebral palsy, and consequently affects their ability to work as energy transfer straps and adapt to fine control changes with quick movements (James R. Gage, Deluca, & Renshaw, 1995; Metaxiotis et al., 2004). Lieber (1990) suggests consideration of muscular strengthening exercises as a treatment possibility before surgically altering muscles. Furthermore, Metaxiotis et al. (2004) suggests that computer modeling may lend valuable insight to the function of biarticular muscles, particularly following a conversion of function.

The Importance of Simulation in Biomechanics

The human musculoskeletal system is a very complex multi-joint linkage system. Additional consideration of the neurological inputs required to produce motion can make evaluation of musculoskeletal disorders very difficult. While many experimental data from clinical studies have aided in the evaluation and treatment of movement abnormalities such as cerebral palsy, there exist limitations which make it difficult to entirely comprehend the biomechanical relationships throughout the musculoskeletal system. Several biomechanical parameters, such as muscle activations and muscle forces, are typically not available through experiments, but

are easily derived through computer simulation. Additionally, some scientific questions, such as surgical technique or outcomes, cannot practically be evaluated in an experimental environment with human subjects.

Computer simulation can offer a means of integrating experimental data, anatomical models, and dynamic principles to thoroughly evaluate scientific questions. Based on anatomical and physiological data, simulations can estimate important parameters of the musculoskeletal system (i.e.: muscle & tendon properties, muscle activations, muscle & joint forces, etc.). Furthermore, biomechanical dynamic simulations can help researchers to understand the mechanisms of movement abnormalities, and can act as a tool to predict treatment outcomes in "what if" scenarios.

Need for Study

Stiff-knee gait is a debilitating gait abnormality among patients with cerebral palsy, commonly attributed to spasticity of the rectus femoris during the initiation of knee flexion in swing (Perry, 1987). A common treatment for this gait disorder is a distal transfer of the rectus femoris, but successful improvement in knee flexion following this procedure is variable (Fox et al., 2009; Goldberg et al., 2003). While the treatment aims to correct the swing phase of gait, it may be compromising another phase such as stance phase. Studies have suggested that the maintaining balance and posture can act as a precursor for other voluntary movements such as independent standing or walking over uneven terrain, thus it is important to consider the effect on all phases of gait (Ting, 2007; M. Woollacott et al., 2005).

Furthermore, it has been suggested that biarticular muscles, such as the rectus femoris, play a unique role in motor control and are among the first affected muscles in the case of cerebral palsy (James R. Gage et al., 1995). These studies lead to the hypothesis that as a biarticular muscle, rectus femoris may be more beneficial preserved as a stabilizer, rather than transferred as a knee flexor. This study will investigate the role of the rectus femoris muscle, before and following transfer surgery, in terms of postural balance in a child patient with cerebral palsy.

Focus of Thesis

The focus of this thesis is to use musculoskeletal modeling and forward dynamic simulation to investigate the influence of biarticular muscles in the human musculoskeletal system. More specifically, this work focuses on the effect of rectus femoris transfer surgery on the postural stability of a child with cerebral palsy. The objective of this study is to compare whole-body center of mass (CoM) displacements in response to support-surface translations for separate simulations of a nominal model (pre-rectus femoris transfer surgery) and various altered models (post surgery). The hypothesis is that a preoperative simulation has less CoM displacement (increased balance recovery) compared to a postoperative case, and furthermore that a unilateral rectus femoris transfer simulation has better balance recovery relative to a bilateral transfer. If this hypothesis proves true, this research would suggest a unique significance of biarticular muscles and their role in postural stability and motor control. Furthermore, a confirmed hypothesis might provide incentive to investigate alternative methods for treatment of stiff-knee gait

in patients with cerebral palsy; while a rejected hypothesis would provide support for rectus femoris transfer surgery as a method of improving knee flexion in patients with stiff-knee gait while maintaining stability.

Figure 1. Patient with cerebral palsy displaying symptoms of stiff-knee gait. Image courtesy of Connecticut Children's Medical Center.

CHAPTER II
BACKGROUND

Motion Capture

Motion capture, or motion tracking, involves the use of external devices to record the position and orientation of an object in physical space. There are many applications of this technology in military, entertainment, sports, medical and scientific industries. A common type of motion capture system is based on a passive optical system, involving retroreflective markers and motion cameras. With this system, retroreflective markers are strategically fixed to an object, and surrounding high-speed, high-resolution video cameras record the position of the markers throughout time. The resulting data set can then be processed and analyzed to recreate the motion of the object in a virtual setting.

A popular application of this technology involves the entertainment industry (video games & films), and uses motion capture on human subjects to generate realistic animations in movies such as Pirates of the Caribbean, Avatar, and Tron: Legacy. In a biomechanics application, the markers are generally fixed to anatomical landmarks (i.e.: greater trochanter, medial/lateral condyles, medial/lateral malleoli), and to the bodies of interest, to track a subjects' movement throughout time. The resulting data can then be used for studying gait analysis, sports performance, and other types of biomechanical research.

Biomechanical Model

Motion capture technology is frequently used in biomechanical research to construct biomechanical models of human subjects. The position of external markers can be used to estimate the position of internal landmarks such as joint centers, and help to define segment properties such as length and position. Furthermore, the anatomical markers enable the creation of local reference frames on each individual segment to define the position and orientation of each body segment within a Newtonian laboratory reference frame. The motion data collected from the subject are then used to prescribe the motion of the biomechanical model.

Elastic Foundation Model

Elastic foundation contact is modeled as a set of independent linear springs which disperse the contact area to calculate the contact pressure for each discrete element, independent from its neighbors (Figure 2) (Johnson, 1985; Perez-Gonzalez et al., 2008). The local pressure, p_i, associated with each discrete spring is determined using Equation (1), where k is the contact stiffness per unit area expressed by Equation (2):

$$p_i = k\delta_i \tag{1}$$

$$k = \frac{1-v}{(1+v)(1-2v)}\frac{E}{h} \tag{2}$$

where E is the elastic modulus, v the Poisson coefficient of the elastic foundation material, and h is the thickness of the elastic foundation. The force experienced in

each spring is a function of the local pressure and the associated area of influence from the active springs, A_i:

$$\overrightarrow{P_i} = A_i \overrightarrow{p_i} \tag{3}$$

$\overrightarrow{P_i} = A_i \overrightarrow{p_i}$and the total contact force is calculated as a discrete integral of the contributing springs:

$$\vec{P} = \sum_i \overrightarrow{P_i} = \sum_i A_i \overrightarrow{p_i} \tag{4}$$

The sum of the contributing areas from active springs, A_i, provides the contact area of the model, while the maximum value of the local deformations, δ_i, is the contact deformation of the elastic foundation model (Perez-Gonzalez et al., 2008).

Optimization

Mathematical optimization describes the process of calculating a global maximum or global minimum value of an objective function by systematically adjusting input values from a pre-defined set of design variables. An example of optimization in a biomechanics application would be an objective function of the errors between the biomechanical model and the experimental motion data. These errors represent the model's generalized coordinates and kinematic parameters such as segment lengths, joint positions, and joint orientations. Optimization techniques can be used to adjust the design variables of the model to minimize these errors and identify a solution that more accurately matches the experimental data.

Forward Dynamics

Forward dynamics is based on Newton's second law of motion,

$$\mathbf{F} = m\,\mathbf{a} \tag{5}$$

where F is the net force applied, m is the mass of the body, and **a** is the body's acceleration. Forward dynamics describes the process of solving for a motion, given forces as an input. Figure 3 details the process of a forward dynamics problem. Either a neural command or external force induces activation and contraction dynamics within the musculotendon system. These forces are experienced in the musculoskeletal system, where the geometrical properties of the lines of action, moment arms, and axes of rotation can be used to determine joint moments. Given these moments, a multi-joint dynamics analysis can be used to calculate the joint accelerations. Finally, integration of the joint accelerations results in the velocity, and additional integration yields position. The resulting positions determine the observed movement.

OpenSim: Open Source Dynamic Simulation Software

Dynamic simulation software is becoming increasingly popular and available as a resource for a wide range of uses. More recently, biomechanical dynamic simulation software packages have become a valuable tool for modeling, simulating, controlling, and analyzing the musculoskeletal system. The difficulty arises when laboratories develop their own simulation software packages and do not make these resources available to the biomechanics community for use and evaluation. Many

commercial software packages exist such as Visual 3-D (C-Motion Inc.), Anybody (Anybody Technology), and SIMM (Musculographic Inc.); however, these packages are often costly and provide limited access to source code for extensibility of the tools.

OpenSim is a freely available, open source biomechanical dynamic simulation software package developed at Stanford University by the Neuromuscular Biomechanics Lab (NMBL). OpenSim allows users to build musculoskeletal models and create simulations of movement through tools such as forward and inverse dynamics. The source code for OpenSim is available in ANSI C++ (Figure 4) while the graphical user interface (GUI) is written in Java (Figure 5). The plug-in capabilities offered through OpenSim encourage users to develop customized controllers, analyses, and models while encouraging collaboration throughout the biomechanics research community (Delp et al., 2007). OpenSim is a powerful tool that has enhanced research in rehabilitation medicine, sports performance, and other biomechanical focuses. The code that drives OpenSim undergoes continuous development, testing, and analysis through multi-institutional collaborations (Delp et al., 2007).

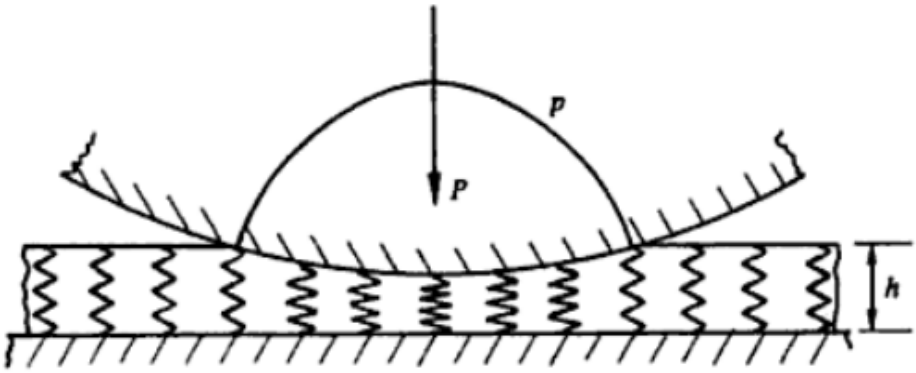

Figure 2. Elastic foundation model demonstrating bed of springs.

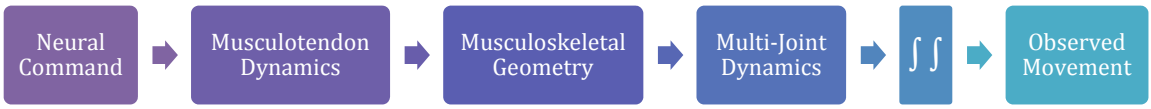

Figure 3. Forward dynamics flowchart.

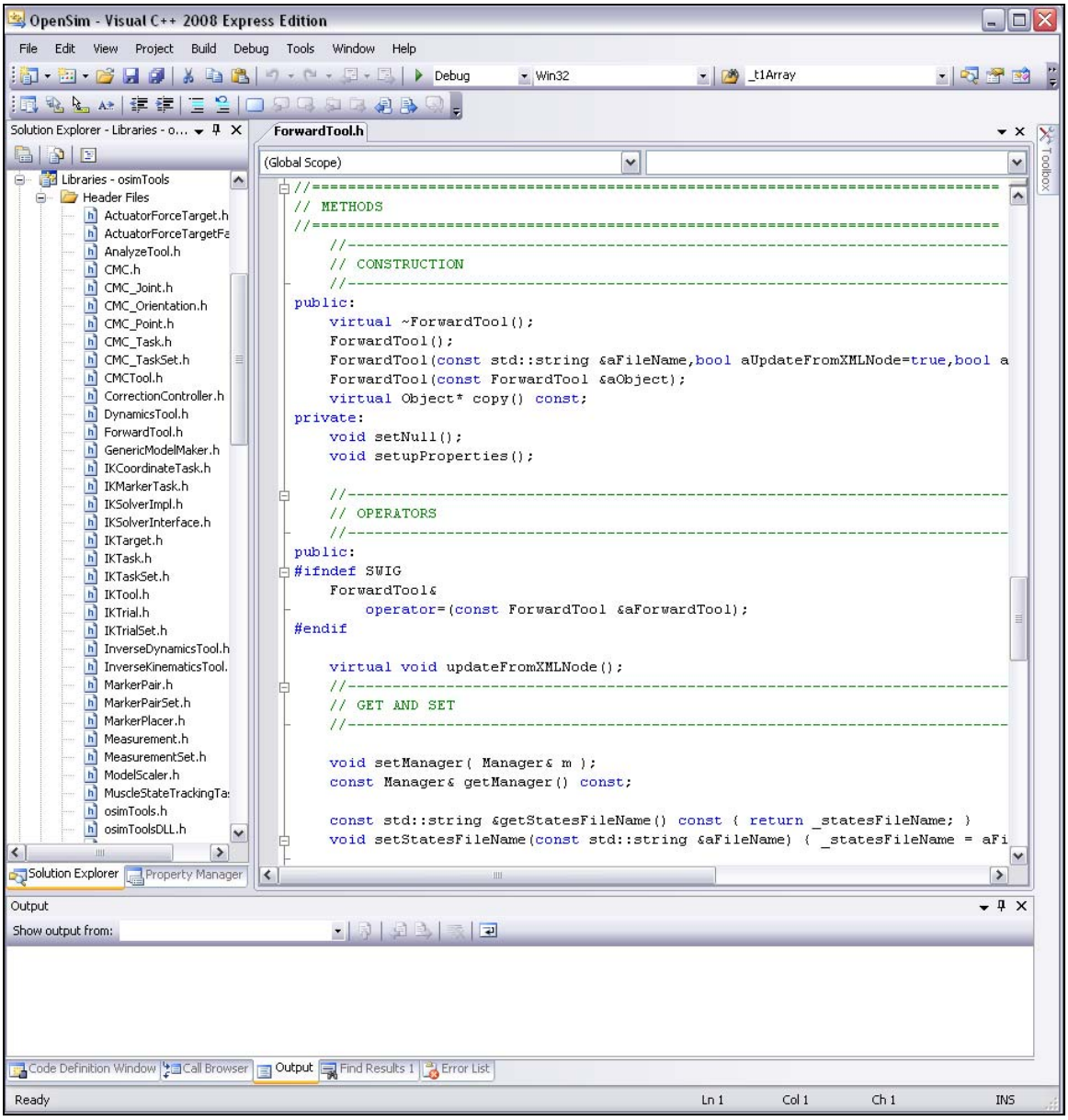

Figure 4. OpenSim source code in Microsoft Visual C++.

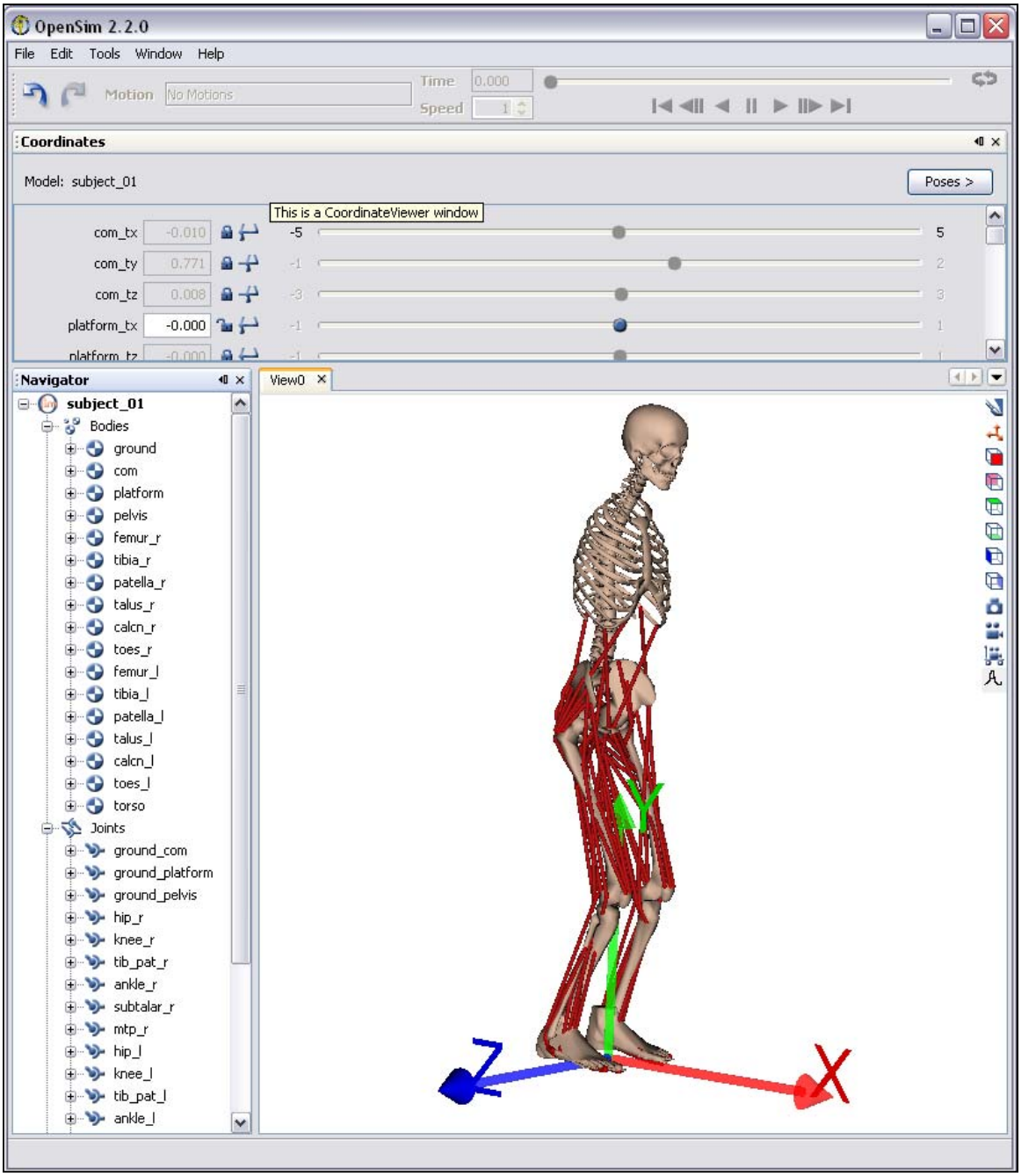

Figure 5. OpenSim graphical user interface (GUI).

CHAPTER III
METHODS

Three-Dimensional Musculoskeletal Model

Three-dimensional musculoskeletal models were constructed in OpenSim to evaluate the proposed hypothesis. A nominal model was used for the preoperative cases, opposed to postoperative cases following virtual tendon transfer. All models include 10 rigid body segments: head and trunk, pelvis, and a right and left femur, tibia, patella, and foot segments (Figure 6). The arms are not included in the models, but the mass of the arms is included in the head and trunk body segment. The lower extremity joints are modeled as follows with 19 degrees of freedom (DOF): the hips are ball-and-socket joints (Anderson & Pandy, 1999), each knee is a pin joint with tibiofemoral and patellofemoral kinematics defined by knee flexion angle (Delp et al., 1990), and the ankle joints are pin joints (Inman, 1976). The feet were scaled to the patient's size, and are modeled using elastic foundation contact geometry with an underlying 1 square-meter platform.

Muscle-tendon actuators were implemented to act as muscles and move the model. The paths of the actuators are defined using points of origin and insertion, and intermediate via points to account for muscle wrapping around bones. The force-generating properties of the muscle-tendon actuators are defined by scaling a generic Hill-type muscle model which consists of a tendon in series with a muscle (Figure 7) (Hill, 1938; Zajac, 1989). The tendon is modeled as a non-linear elastic element, while the muscle is represented by a passive elastic element in parallel

with an active contractile element (CE). Each muscle-tendon actuator is scaled based on four properties and associated relationships: peak isometric muscle force (F_0^M), optimal muscle-fiber length (L_0^M), pennation angle (α), tendon slack length (L_s^T), the normalized passive and active muscle fiber force-length relationships and the normalized tendon force-length relationship. The physiological cross-sectional area (PCSA) was used to determine the peak isometric muscle force (Friederich & Brand, 1990; Wickiewicz, Roy, Powell, & Edgerton, 1983) and the fiber length and pennation angle data were taken from Friederich and Brand (1990).

Nominal Model

The nominal, *preoperative model* is based on anthropometric data from a 12-year old male patient with cerebral palsy, who suffers from stiff-knee gait (Delp, Arnold, Liu, Anderson, & Thelen, 2006; Goldberg, Ounpuu, Arnold, Gage, & Delp, 2006). The patient is modeled at 1.45 meters tall with a mass of 35.76 kilograms. The initial states of the models are based on motion capture data, collected at Connecticut Children's Medical Center in Hartford, CT, to determine the joint angles and body positions. The nominal model consists of 92 muscle-tendon actuators, with muscle and tendon properties adapted from Delp *et al.* (1990, 2007) (Figure 8; Appendix A). Similar neuromusculoskeletal models have been used in previous studies concerning cerebral palsy (Delp et al., 2006; Fox et al., 2009; Goldberg et al., 2006; Reinbolt, Fox, Arnold, Ounpuu, & Delp, 2008; Reinbolt, Fox, Schwartz, & Delp, 2009).

Tendon Transfer

The postoperative models were modified from the nominal model to simulate a virtual rectus femoris transfer. The simulated rectus femoris transfer involved a relocation of the original insertion of the rectus femoris from the patella to the effective insertion of the sartorius, posterior to the knee (Figure 8-b). The tendon slack length of the transferred muscle was scaled to ensure the muscle fibers operated near their preoperative length ranges (Fox et al., 2009). The *unilateral model* involved a virtual transfer on the left limb only, while the *bilateral model* reflects a virtual transfer on both limbs.

Monoarticular Comparison

The rectus femoris (biarticular) and vastus medialis (monoarticular) are the two quadriceps muscles primarily responsible for sagittal plane motion (Hernandez, Dhaher, & Thelen, 2008). In an effort to draw conclusions about the significance of biarticular muscles, the vastus medialis was used to comparatively investigate the influence of monoarticular muscles on postural balance. The *monoarticular model* was modified from the nominal model by removing the vastus medialis from the left limb. The model involved removal of the vastus medialis as an extreme comparison because no corrective procedure involving surgical transfer of this monoarticular muscle is used in practice. Consequently, the resulting model contains 91 muscle-tendon actuators.

Foot & Platform Interface

An elastic foundation contact model was implemented to model the foot/platform interface. The feet were modeled as infinitely stiff with a dissipation of 0.50. The frictional coefficients considered were static (0.90), dynamic (0.90) and viscous (0.60). The contact mesh of the foot was modeled based on cadaveric foot geometry and then scaled to match the subject's size (Erdemir, Sirimamilla, Halloran, & van den Bogert, 2009) (Figure 9). The platform was modeled as a 1 meter x 1 meter halfspace which interacted with the feet meshes. This interaction can be considered similar to that of rubber on concrete.

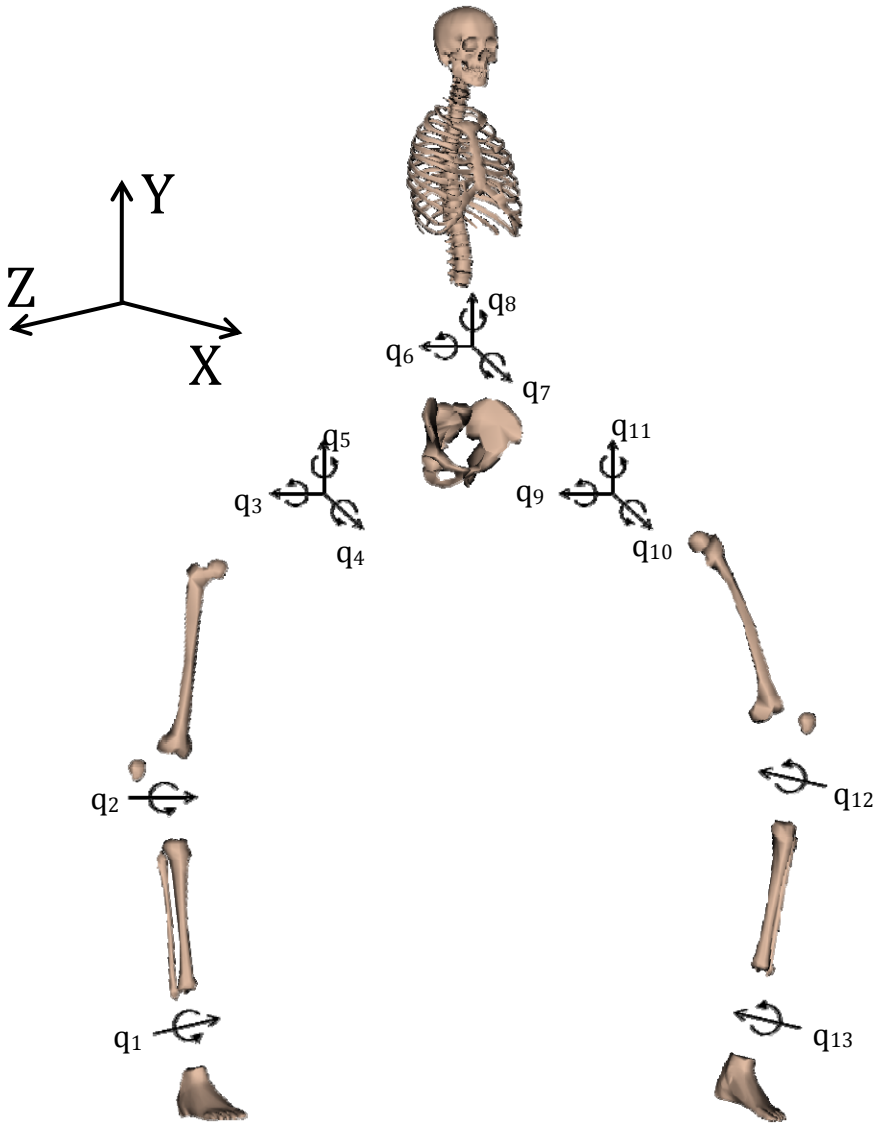

Figure 6. Three-dimensional, 10 segment, 19 DOF kinematic model linkage illustrating pin and ball-and-socket joints. The ground-pelvis joint adds 3 translational and 3 rotational DOFs to the model.

Table 1. Degrees of freedom for biomechanical model. The ground-pelvis joint adds 3 translational and 3 rotational DOFs to the model.

DOF	Description
q_1	Right ankle plantarflexion-dorsiflexion angle
q_2	Right knee flexion-extension angle
q_3	Right hip flexion-extension angle
q_4	Right hip adduction-abduction angle
q_5	Right hip internal-external rotation angle
q_6	Trunk anterior-posterior tilt angle
q_7	Trunk elevation-depression angle
q_8	Trunk internal-external rotation angle
q_9	Left hip flexion-extension angle
q_{10}	Left hip adduction-abduction angle
q_{11}	Left hip internal-external rotation angle
q_{12}	Left knee flexion-extension angle
q_{13}	Left ankle plantarflexion-dorsiflexion angle

Table 2. Mass and mass centers of each body in the model.

		Mass Center (m)		
Body	**Mass (kg)**	**x**	**y**	**z**
Toes Right	0.0999	0.0300	0.00520	-0.0152
Calcaneus Right	0.577	0.867	0.0260	0.00
Talus Right	0.0461	0.00	0.00	0.00
Patella Right	0.0461	0.00	0.00	0.00
Tibia Right	1.71	0.00	-0.152	0.00
Femur Right	4.29	0.00	-0.156	0.00
Pelvis	5.43	-0.0613	0.00	0.00
Femur Left	4.29	0.00	-0.156	0.00
Tibia Left	1.71	0.00	-0.152	0.00
Patella Left	0.0461	0.00	0.00	0.00
Talus Left	0.0461	0.00	0.00	0.00
Calcaneus Left	0.5767	0.867	0.0260	0.00
Toes Left	0.0999	0.0300	0.00520	0.0152
Torso	15.8	0.00	0.277	0.00

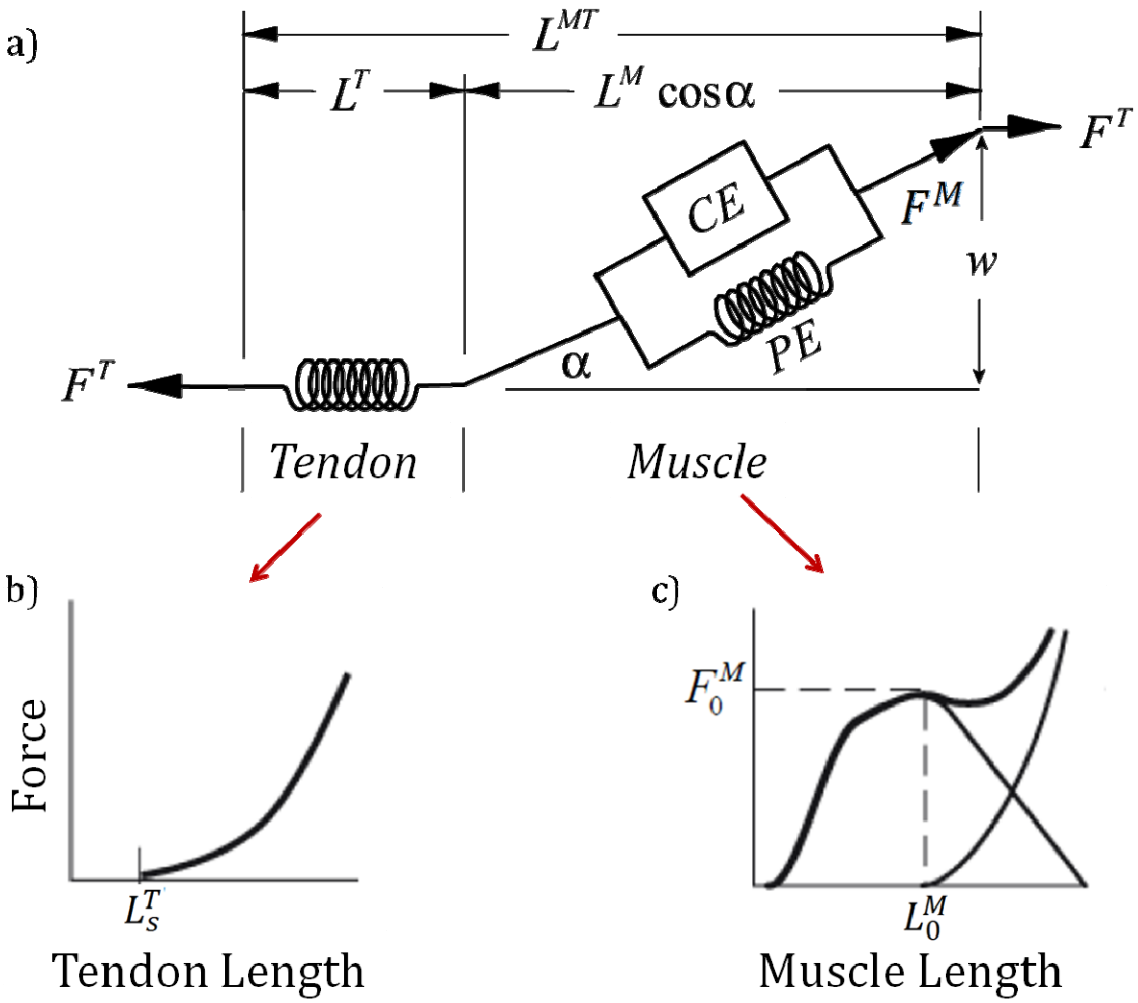

Figure 7. (a) Muscle-tendon actuator using a generic Hill-type muscle model with (b) normalized tendon force-length curve, and (c) normalized active and passive muscle force-length curve.

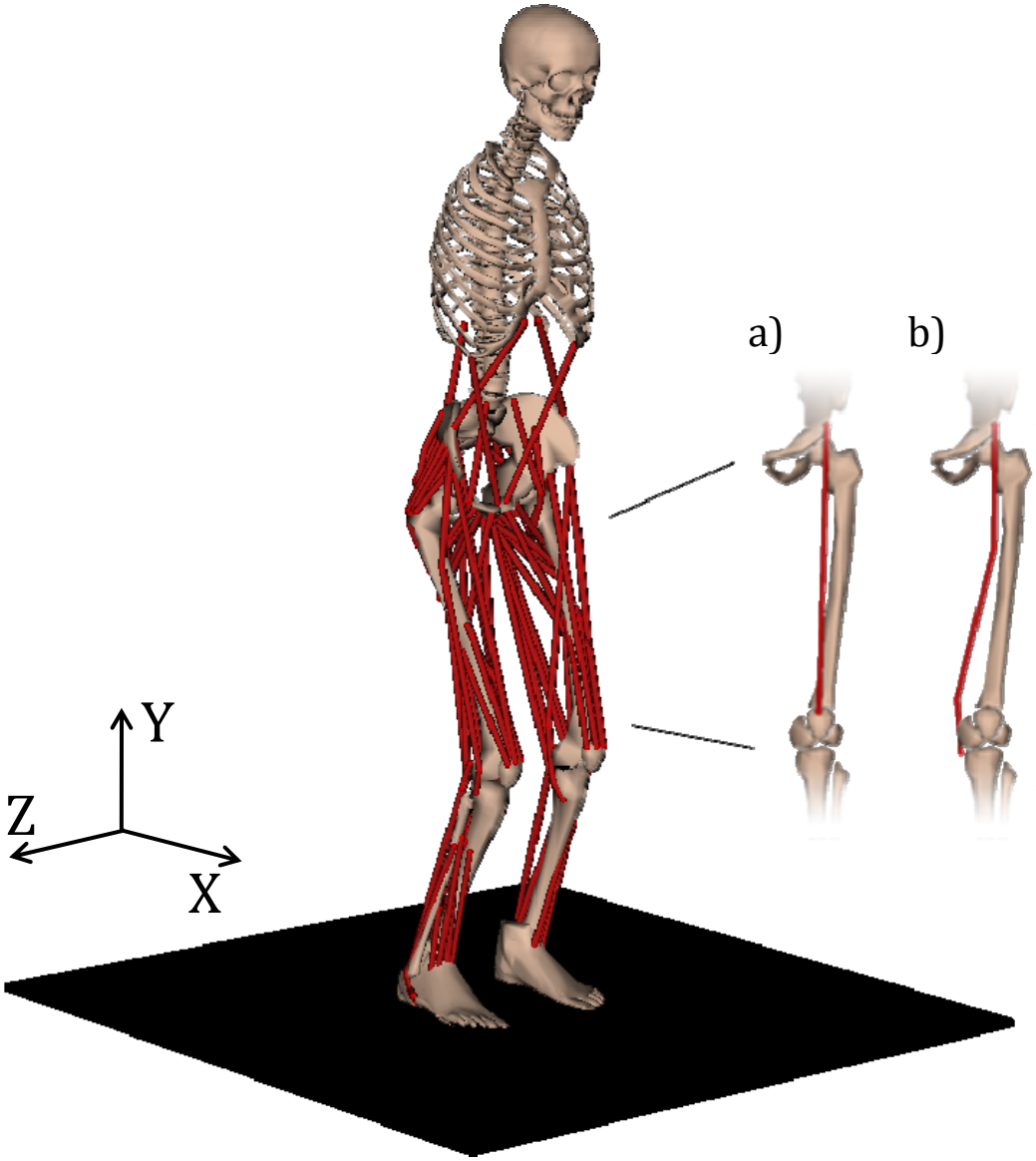

Figure 8. 3-dimensional, 10 segment, 19 DOF musculoskeletal model of a patient with cerebral palsy with 92 muscle-tendon actuators (shown in red) and biarticular attachments for the rectus femoris muscle (a) pre- and (b) post-surgical transfer to the sartorius.

Figure 9. Foot contact mesh based on cadaveric geometry and scaled to subject's size.

Support-Surface Translation

The motion of the support-surface was based on previous clinical studies investigating postural responses to perturbations (Welch & Ting, 2008; M. Woollacott et al., 2005). Each 6 second simulation involved 0.25 seconds of quiet standing, 0.35 seconds of support-surface translation, and 5.4 seconds of balance recovery. The support-surface was prescribed to translate 6 centimeters in the anterior and posterior directions (±X) with a peak velocity of 23 cm/s (Figure 10). The translation of the support-surface was implemented using a generalized cross validation (GCV) spline function within the model file which calls for coordinates to define the time and position of the support-surface. The GCV spline function used a half order of 3 and 501 coordinate pairs, each with a weighting of 1.0. The support-surface was constrained to only move in the anterior/posterior (±X) directions.

Stretch Reflex Controller

In lieu of a brain and neural command, a stretch reflex controller was used to manage the muscles' response to posture perturbations. The controller was adapted from Feng and Mak (1998) and is dependent on static and dynamic control parameters, and a general reflex gain, to determine the stretch reflex. The static and dynamic thresholds, T_L and T_V, are defined in Equations (6) and (7) where L_0 is the resting length of a muscle, L_R is the range of muscle stretch, $0 < L_R < L_0$, and V_{MAX} is the maximal contracting velocity of the muscle. The static (λ_L) and dynamic (λ_V) control parameters adjust the sensitivity of the stretch reflex.

$$T_L = L_0 - \lambda_L \cdot L_R \tag{6}$$

$$T_V = \lambda_V \cdot V_{MAX} \cdot \frac{L_R}{L_0} \tag{7}$$

The general reflex gain (λ_G) is nonlinear, and is incorporated into the stretch reflex with the static and dynamic control parameters as seen below in Equation (8).

$$u(t) = \begin{cases} \lambda_G\left[(L - T_L)/(L_0 + L_R) + \left(\dot{L} - T_V\right)/(V_{MAX})\right] \\ 0 \end{cases}$$
$$if \begin{cases} L > T_L \; and \; \dot{L} > T_V \\ otherwise \end{cases} \tag{8}$$

The static and dynamic control parameters and general reflex gain were all determined using MATLAB's nonlinear least-squares optimizer algorithm. The objective function of the optimizer was to maintain the CoM at its initial position throughout the simulation. The optimizer began with an initial guess of zero for all parameters, and was run to maintain a 16 second long simulation of quiet-standing for the preoperative model. This controller is analogous (but not identical) to monosynaptic reflexes and afferent mechanisms (e.g., muscle spindles and Golgi tendon organs) responsible for lower-level motor control, and should not be affected following surgical transfer. For this reason, the same control parameters and reflex gain were used for both the preoperative and postoperative simulations of quiet standing and support-surface translations.

C++ Main Program

Forward dynamics analysis is implemented through a main program written in C++ using Microsoft Visual C++ (Figure 11; Appendix B). The main program is responsible for driving the forward dynamics analysis, and also incorporates a header file which contains the stretch reflex controller algorithm (Figure 12; Appendix C). The inputs for each simulation include the control parameters and reflex gain, model file, and simulation time. The program runs the forward dynamics analysis, applying the stretch reflex controller to each muscle in the model. The results of the analysis are output in various file forms in a specified directory. The controls file (_controls.sto) provides the muscle excitations while the states file (_posture_states.sto) provides joint positions and velocities, muscle fiber lengths, and muscle activations throughout the simulation. The motion file (_posture_states_degrees.mot) is similar to the states file, but allows for visualization in the OpenSim GUI. The pseudo-code for the C++ main program is shown in Figure 13 while that for the reflex controller header file is detailed in Figure 14.

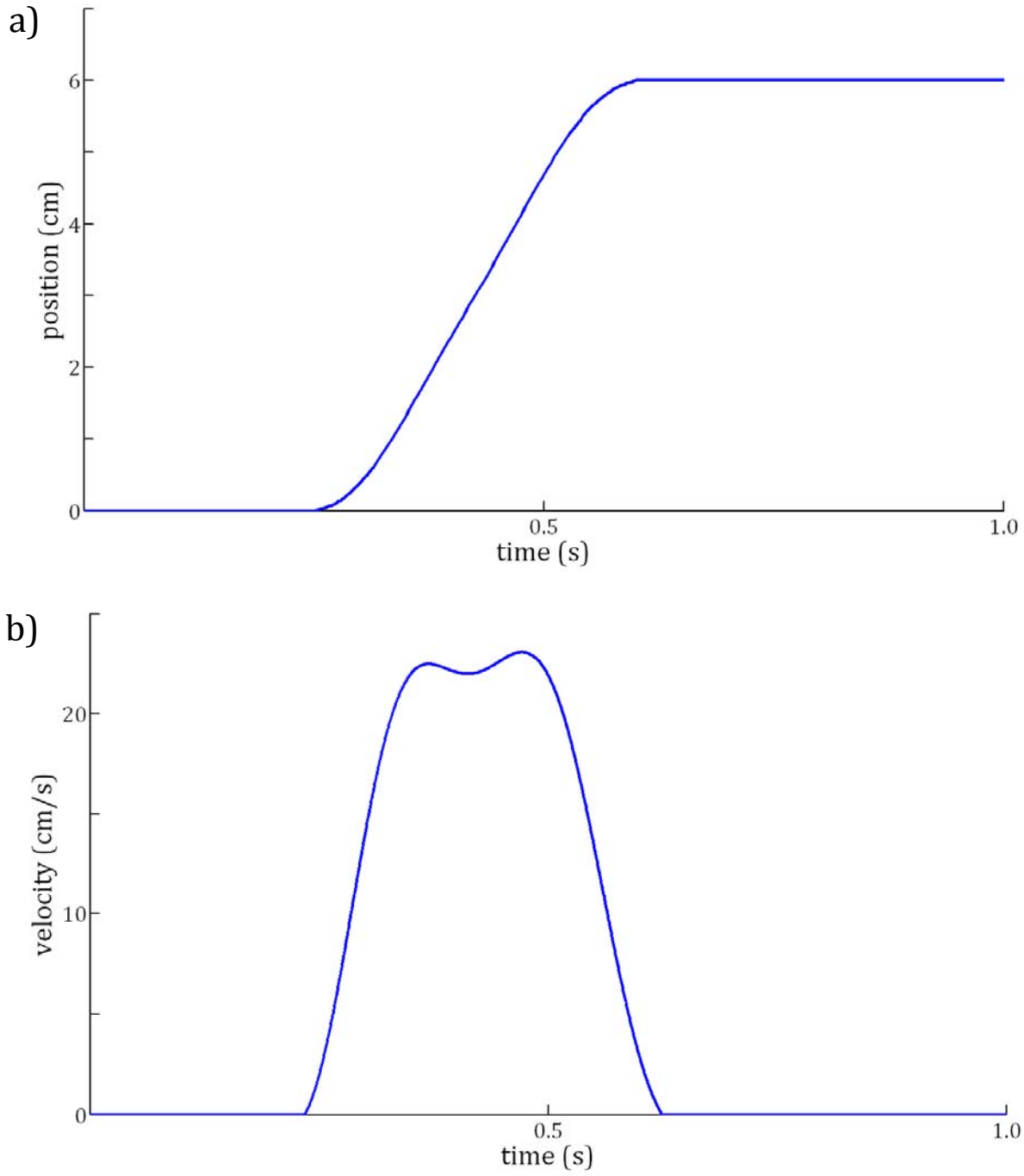

Figure 10. Support-surface translation (a) position and (b) velocity profiles.

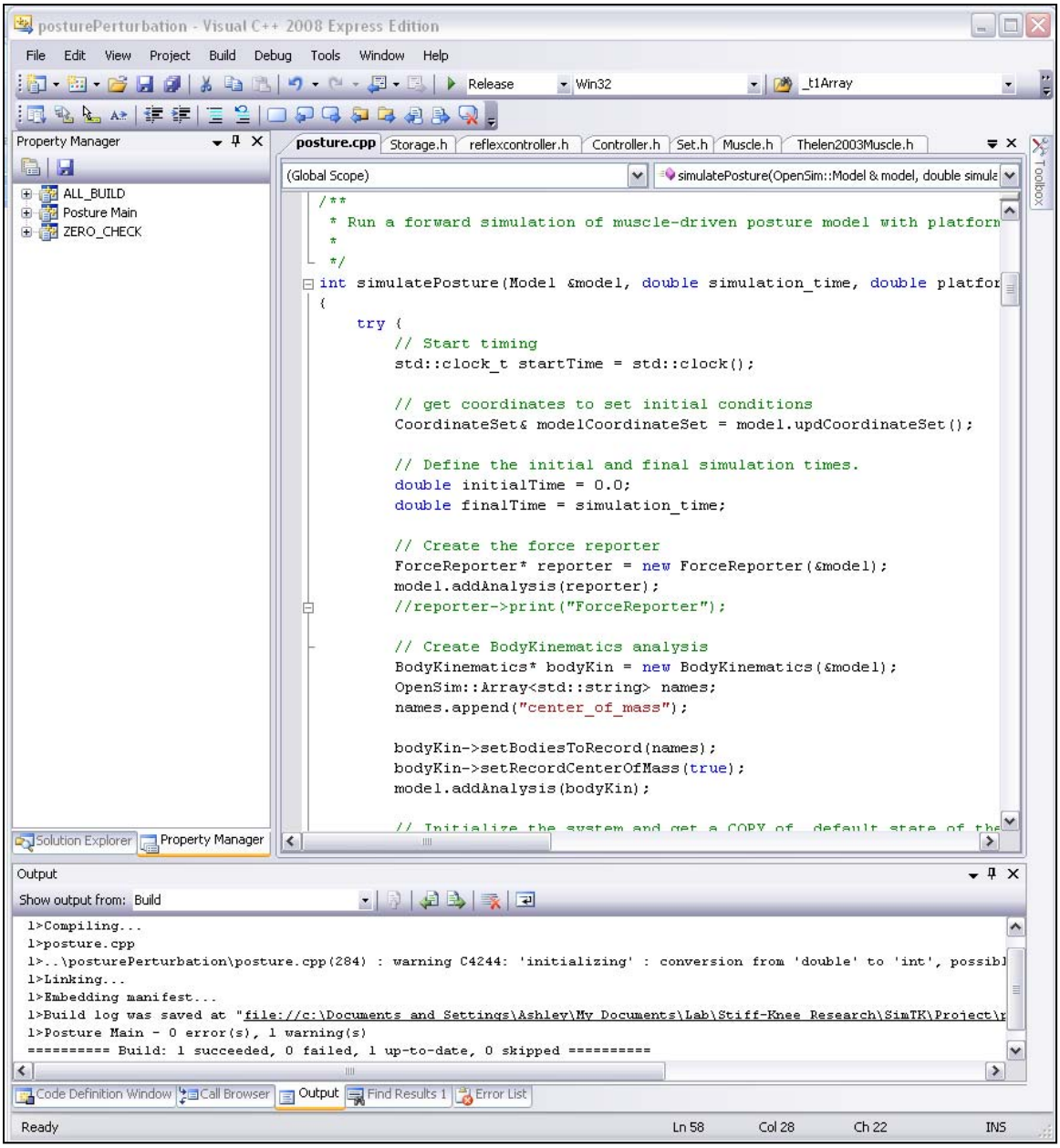

Figure 11. Screenshot of the C++ code to run a forward dynamics posture simulation.

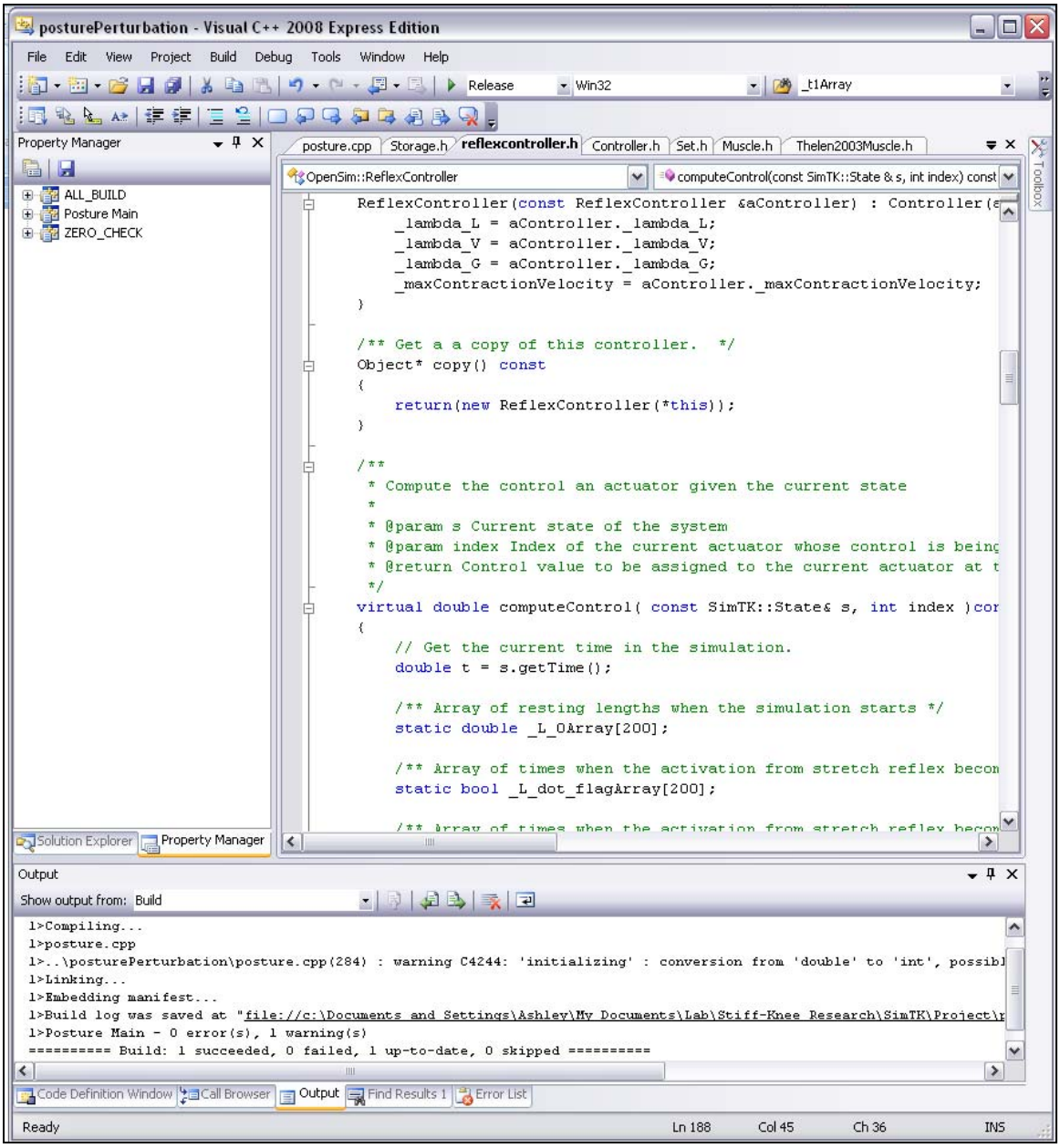

Figure 12. Screenshot of the reflex controller header file for the posture main program.

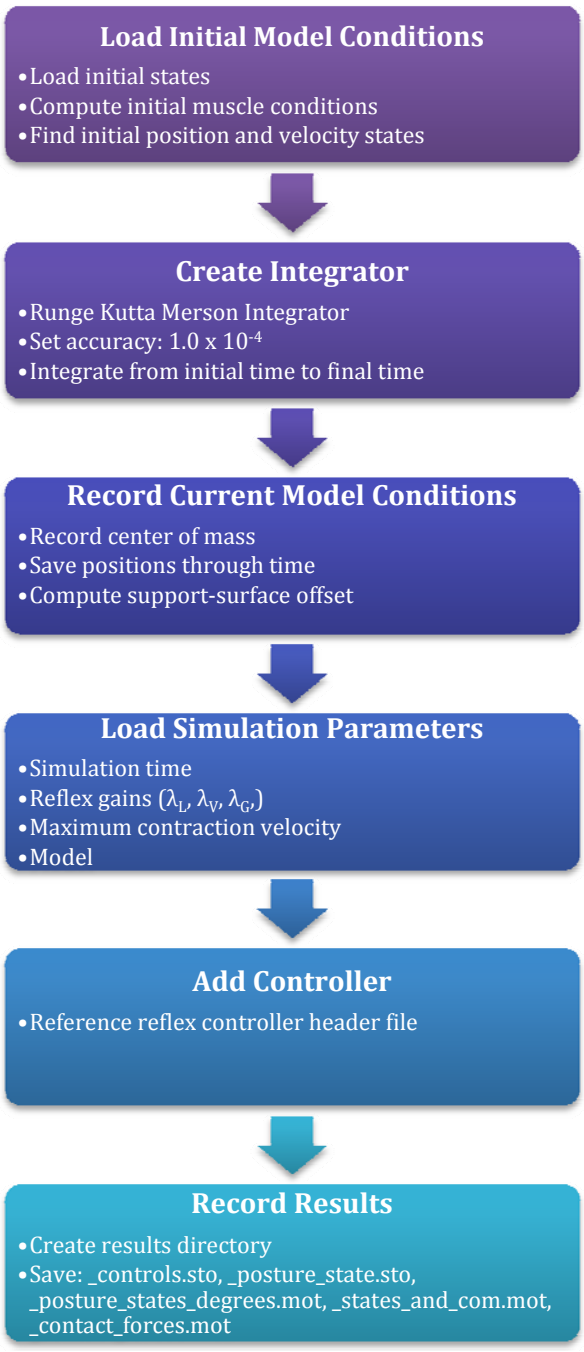

Figure 13. Pseudo-code of C++ main program.

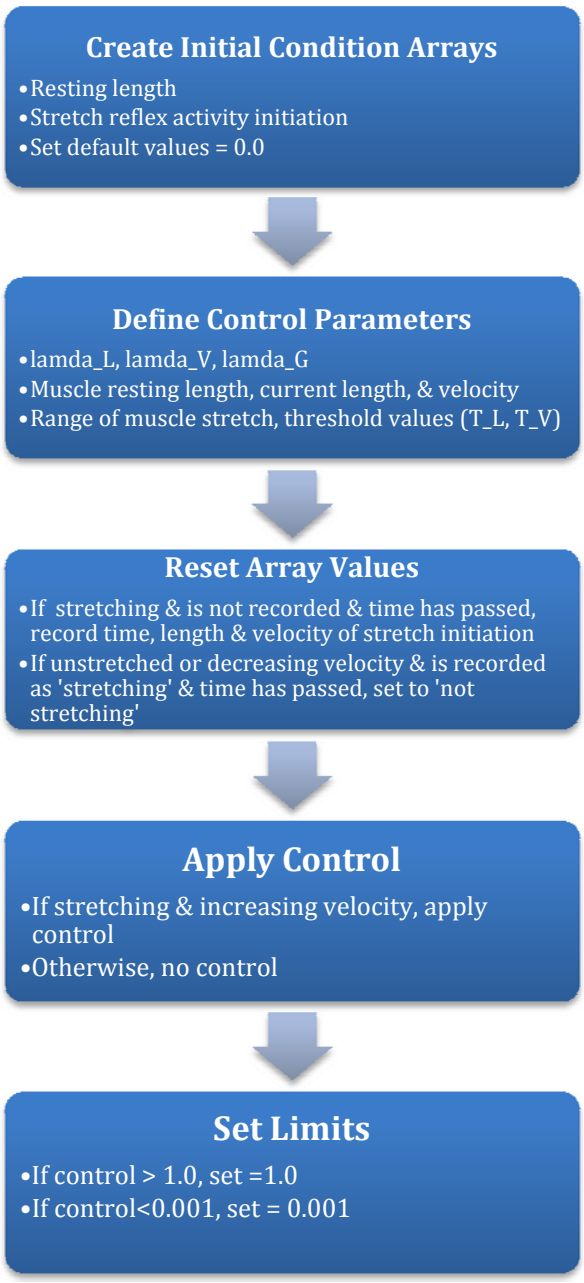

Figure 14. Pseudo-code of reflex controller header file.

CHAPTER IV
RESULTS

Initial States

Due to the spring nature of the elastic foundation contact platform, the initial state conditions for the models were finalized through a forward dynamics analysis on the nominal model. This forward dynamics analysis involved dropping the model from above and allowing it to "settle" into the elastic foundation contact interface. The vertical position of the pelvis began at 76.3 cm from the ground, and settled at 76.2 cm (Figure 15). The resulting vertical pelvis position was used in the initial states file to minimize effects from the elastic foundation contact.

Reflex Control Parameters & Gain

The static and dynamic control parameters and general reflex gain which define the stretch-reflex controller were all determined using the nonlinear least-squares optimizer algorithm in MATLAB. Following each output from the optimization process, the control parameters and reflex gain were implemented in the main program and the CoM displacement was evaluated. The simulation was run on a nominal model with no support-surface translation, to ensure that the control parameters and reflex gain could maintain a static posture. The optimization was discontinued after output parameters were constant, and a 16 second static simulation resulted in negligible change (<0.1 cm) in the CoM position

(Figure 16). The final values used to define the control parameters and reflex gain are highlighted in Table 3.

Preoperative Model vs. Postoperative Model

This study compares whole-body CoM displacements in response to support-surface translations for separate simulations of preoperative and postoperative models to evaluate the effect of rectus femoris transfer surgery on the postural stability of a child with cerebral palsy. Balance was maintained for the preoperative model simulations of anterior and posterior support-surface translation, while balance was not maintained for the postoperative model(s) simulations (Figure 17). The preoperative model successively minimized CoM sway and recovered balance during each 6 second simulation. Postoperative model simulations were not able to recover their original CoM positions, and lost balance after 5 seconds in the case of anterior support-surface translation, and 2 seconds for a posterior translation of the support-surface. This is supported numerically with a lower RMS difference from zero CoM displacement for the preoperative model compared to the postoperative models (Appendix D).

Unilateral vs. Bilateral

The effect of rectus femoris transfer surgery on simulated CoM was evaluated with a unilateral transfer model as well as a bilateral transfer model. The unilateral model consistently demonstrated improved balance recovery and lower RMS difference from zero CoM displacement over the bilateral model in simulations

of anterior and posterior translation of the support-surface. Anterior translation of the support-surface resulted in loss of balance for the unilateral model simulation around 5 seconds, while control was lost for the bilateral model simulation around 4.5 seconds. Additionally, posterior translation of the support-surface resulted in loss of balance after 4 seconds with the unilateral model and before 2 seconds with the bilateral model.

Monoarticular vs. Biarticular

In order to draw conclusions regarding the significance of biarticular muscles in postural control, a monoarticular model (lacking the left vastus medialis muscle) was also simulated and analyzed with a support-surface translation in the anterior direction. Balance was maintained and CoM sway was minimized throughout the 6 second simulation with the monoarticular model, similar to the preoperative model (Figure 18). This result is numerically supported in noting that the RMS difference from zero CoM displacement for the monoarticular model simulation is lower than other models for anterior support-surface translation simulations.

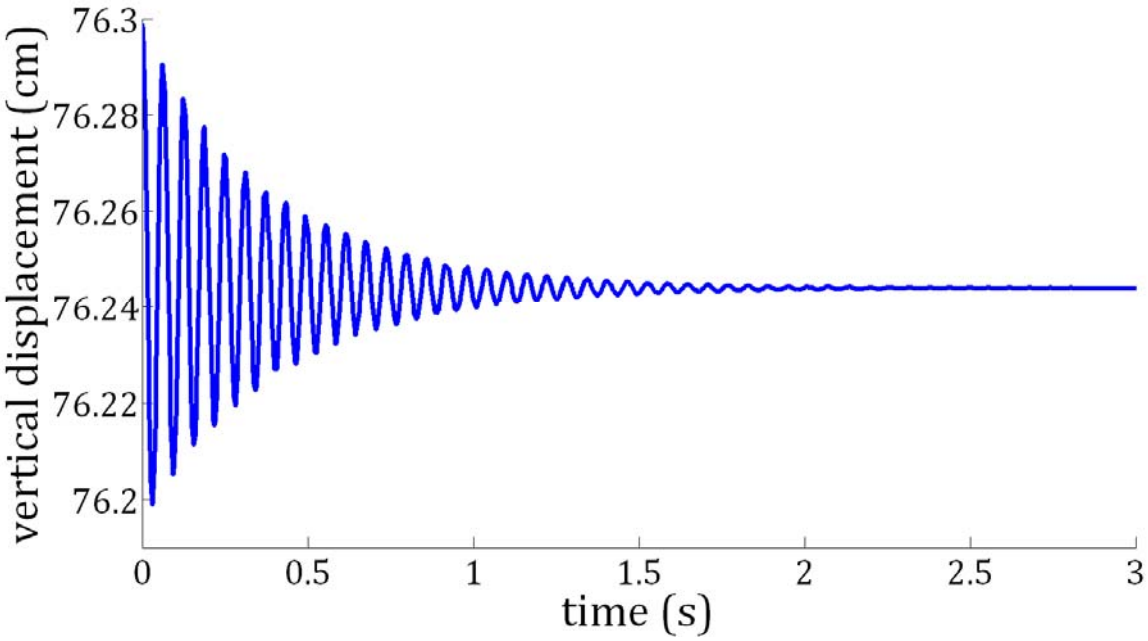

Figure 15. Vertical displacement of the pelvis of the nominal model to determine initial conditions given elastic foundation foot contact.

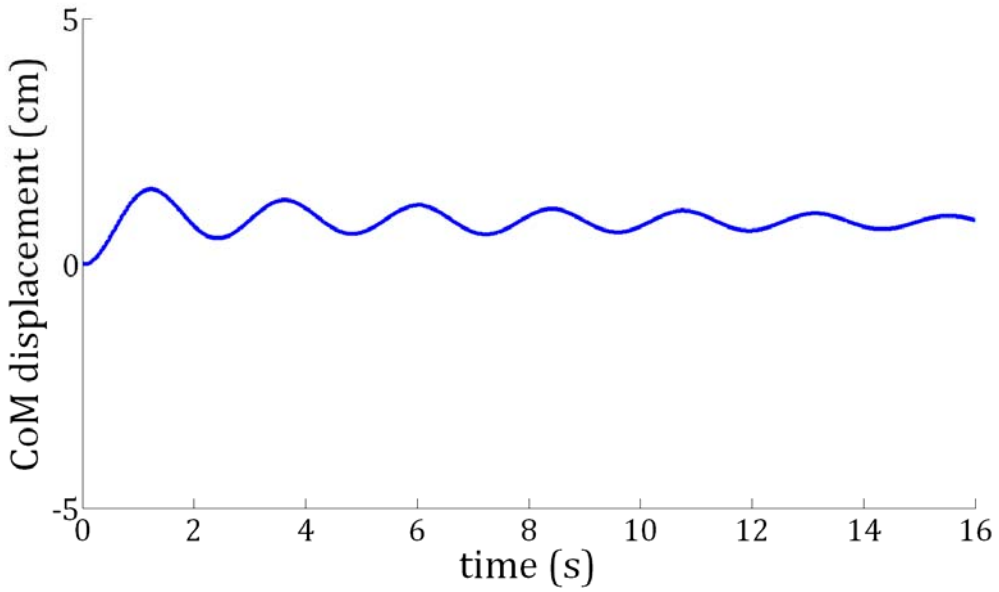

Figure 16. Anterior/posterior (+/-) center of mass displacement for static simulation while determining control parameters and reflex gain.

Table 3. Input and output values for optimization to determine control parameters and reflex gain. Final values used for the simulations are highlighted in gray.

Simulation Time	**Input Guess Values**			**Optimizer Output Values**		
	λ_L	λ_V	λ_G	λ_L	λ_V	λ_G
0.25 s	0.00	0.00	0.00	2.34	0.0614	8.56
0.50 s	2.34	0.0614	8.56	2.35	0.0573	8.57
1.0 s	2.35	0.0573	8.57	2.50	0.0417	8.91
2.0 s	2.50	0.0417	8.91	3.87	0.0222	7.86
4.0 s	3.87	0.0222	7.86	5.43	0.0169	8.96
8.0 s	5.43	0.0169	8.96	5.42	0.0186	8.95
16.0 s	5.42	0.0186	8.95	5.41	0.0196	8.95

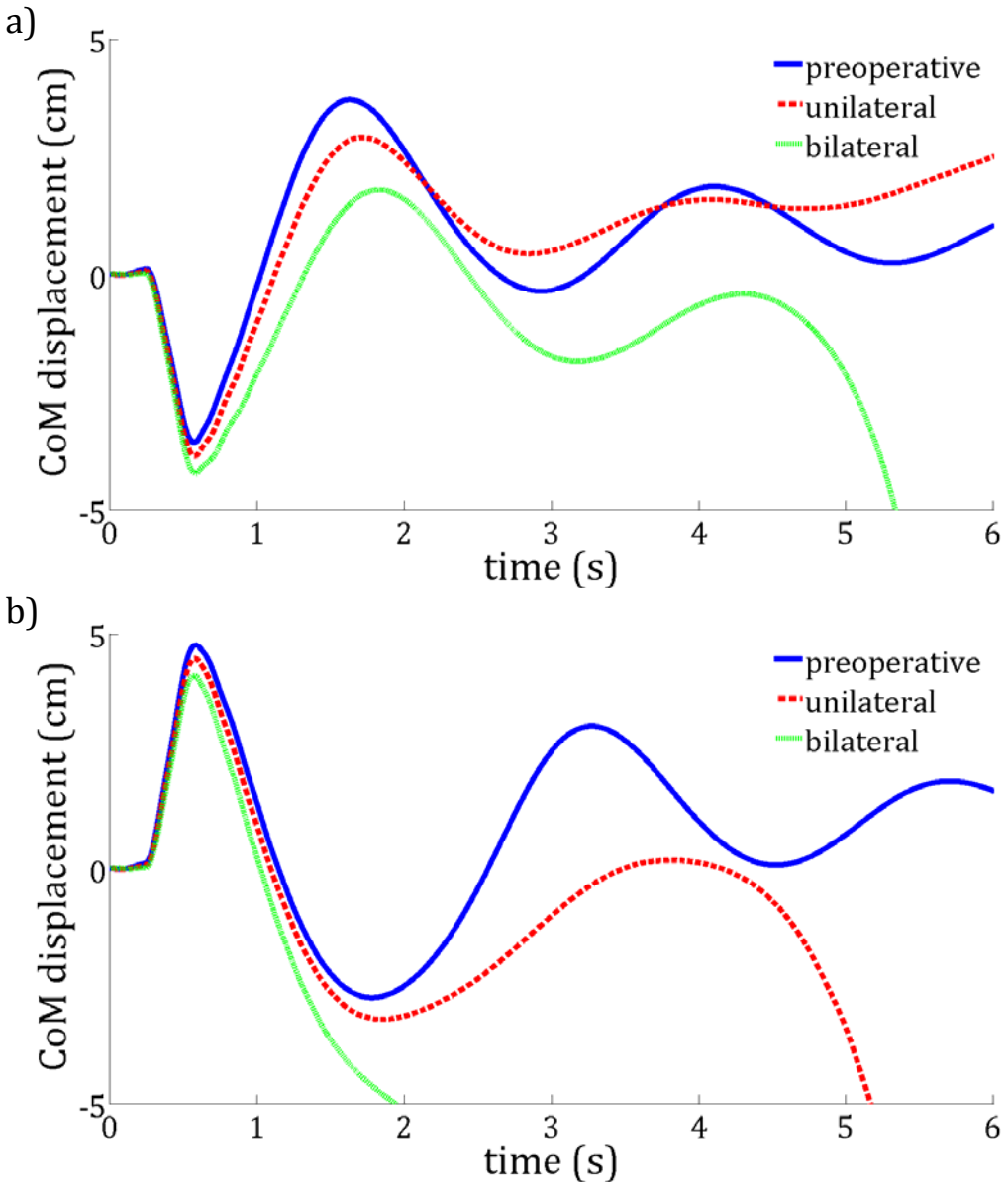

Figure 17. Anterior/posterior (+/-) center of mass displacements relative to the support-surface translating (a) anterior and (b) posterior for simulations of preoperative, unilateral, and bilateral tendon transfer. The gray shade highlights the duration of support-surface translation.

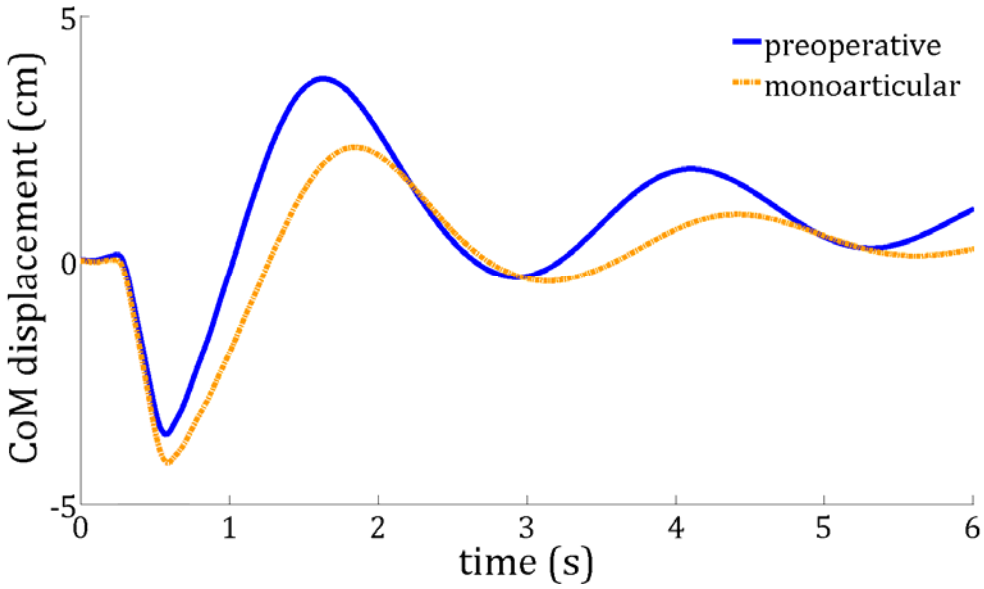

Figure 18. Anterior/posterior (+/-) center of mass displacement relative to the support-surface translating anterior for simulations of the preoperative and monoarticular comparison models. The gray shade highlights the duration of support-surface translation.

CHAPTER V
DISCUSSION

Analysis of Results

Simulations for both directions of support-surface translation support the hypothesis that CoM displacement is reduced with the preoperative model as compared to the postoperative cases, and that balance recovery is improved with the unilateral model relative to the bilateral model. While the preoperative model demonstrated CoM sway throughout the simulations, CoM displacement gradually decreased and an upright posture was maintained. Conversely, increased CoM sway, and loss of balance stability occurred with the postoperative models. In general, the postoperative model simulations seemed to have less postural control when the support-surface was posteriorly translated.

Furthermore, stability with the monoarticular model was similar to the preoperative model suggesting that the absence of the vastus medialis had little to no effect on the postural stability within the model, and that the rectus femoris is more influential in postural stability than the vastus medialis. Given that the rectus femoris and vastus medialis are the quadriceps muscles responsible for sagittal plane motion, this result provides reasonable support that biarticular muscles assume more responsibility for control than monoarticular muscles.

Assumptions & Research Challenges

The findings of this study should be interpreted within the context of the presented assumptions and research challenges. Limitations in the biomechanical model, controller, and lack of subject variability are addressed.

Biomechanical Model Selection

While providing many opportunities to perform "what if" studies, computer modeling does offer a few challenges in terms of the degree of realism that can be presented. The model scenarios used in this study were fairly generic and did not incorporate any skeletal abnormalities commonly found in children with cerebral palsy such as tibial torsion (Novacheck, Trost, & Sohrweide, 2010). Furthermore, arms were not included in the biomechanical model due to a lack of data, but the mass properties of the arms are included in the torso body. Hamner, Seth, and Delp (2010) found that inclusion of arms in a running simulation accounted for less than 1% of the mass center acceleration and suggested that their contribution to propulsion is minimal. While a patient would likely use arms to aid in balance recovery, this study was focused on muscular properties and contributions to postural stability. Thus, consideration for bone deformities or inclusion of arms would create additional design variables, complicating the results and analysis to elucidate the role of biarticular muscles in the musculoskeletal system.

Reflex Controller

As the models used in these simulations are purely musculoskeletal and do not incorporate a neural command, a simple stretch reflex controller was applied to

each muscle in the model. This controller is analogous (but not identical) to monosynaptic reflexes and afferent mechanisms (e.g., muscle spindles and Golgi tendon organs) responsible for lower-level motor control. Trial simulations clearly demonstrated that the controller is very sensitive to the control parameters and reflex gain, and that many iterations of optimization were necessary to determine appropriate values to maintain stability. Additionally, while the stretch-reflex controller was not the focus of this study, it only incorporated a single reflex and did not account for the time delay between neural activation and stretch activity. While these assumptions are adequate for a comparative study, a more intricate controller might prove more versatile for future studies.

Subject Variability

The effects of cerebral palsy on stiff-knee gait are variable and many patients experience different symptoms and undergo a variety of treatments. However, the aim of this study was to investigate the role of biarticular muscles in postural control, and while consideration of patient variability may aid in treatment analysis, it would complicate the evaluation of biarticular muscles in the musculoskeletal system. For this reason, this study used kinematic and anthropometric data from a single patient suffering from stiff-knee gait and evaluated a single surgical treatment.

CHAPTER VI
CONCLUSION

Importance of Biarticular Muscles

The results of this study lead to the preliminary conclusion that distal transfer of the rectus femoris to the insertion of the sartorius compromises the postural stability of patients experiencing stiff-knee gait from spastic cerebral palsy. The biarticular quadriceps muscle, rectus femoris, proved a significant role in balance recovery in response to support-surface translations while the monoarticular vastus medialis was not significant. An upright posture and stability were both maintained with the preoperative model while balance was lost within the 6 second simulation for both postoperative models.

Future Work

There are several opportunities for future development based upon the results of this study. Alternative treatment procedures, controller development, and multidirectional support-surface translations are the main focus for the continuation of this research.

Alternative Treatment Procedures

This study focused on one of many treatment solutions involving biarticular muscles for abnormal gait as a symptom of cerebral palsy. Alternative treatment options involve distal transfer of the rectus femoris to the iliotibial band for stiff-knee gait and hamstring lengthening procedures for crouch gait (Baumann, Ruetsch, &

Schurmann, 1980; Fox et al., 2009). Investigation of these alternative procedures would contribute additional insight to the significance of biarticular muscles and their role in postural control.

Controller Development

While it was not the focus of this study, the controller that was used is a very basic stretch-reflex control model applied uniformly to each muscle in the musculoskeletal system. Incorporating a time delay for the muscle activity (such as 50 ms), and applying control to groups of muscles (i.e.: hip flexors vs. extensors, etc.) has potential to add accuracy and depth to the control model. Furthermore, a feedback control model seems ideal for a study involving unexpected perturbations. Enhancing the control model increases the versatility of the study and allows for application in other investigations.

Multidirectional Translations

This study served as an initial investigation of computer simulation of musculoskeletal response to support-surface translations and was based off of similar experimental studies (Welch & Ting, 2008; M. Woollacott et al., 2005; M. H. Woollacott & Shumway-Cook, 2005). The support-surface translations were prescribed at a single magnitude (6 cm) in either the anterior or posterior direction. There is an opportunity for future investigations to incorporate multidirectional (medio-lateral, diagonal, rotational, etc.) as well as various magnitudes of translations of the support-surface.

LIST OF REFERENCES

Anderson, F. C., & Pandy, M. G. (1999). A Dynamic Optimization Solution for Vertical Jumping in Three Dimensions. *Comput Methods Biomech Biomed Engin, 2*(3), 201-231. doi: I291L991001 [pii]

Baumann, J. U., Ruetsch, H., & Schurmann, K. (1980). Distal Hamstring Lengthening in Cerebral-Palsy - an Evaluation by Gait Analysis. *International Orthopaedics, 3*(4), 305-309.

Delp, S. L., Anderson, F. C., Arnold, A. S., Loan, P., Habib, A., John, C. T., . . . Thelen, D. G. (2007). OpenSim: open-source software to create and analyze dynamic simulations of movement. *IEEE Trans Biomed Eng, 54*(11), 1940-1950. doi: 10.1109/TBME.2007.901024

Delp, S. L., Arnold, A. S., Liu, M. Q., Anderson, F. C., & Thelen, D. G. (2006). *Simulation-Based Treatment Planning for Stiff-Knee Gait.* Paper presented at the American Society of Biomechanics, Blacksburg, VA.

Delp, S. L., Loan, J. P., Hoy, M. G., Zajac, F. E., Topp, E. L., & Rosen, J. M. (1990). An interactive graphics-based model of the lower extremity to study orthopaedic surgical procedures. *IEEE Trans Biomed Eng, 37*(8), 757-767. doi: 10.1109/10.102791

Erdemir, A., Sirimamilla, P. A., Halloran, J. P., & van den Bogert, A. J. (2009). An elaborate data set characterizing the mechanical response of the foot. *J Biomech Eng, 131*(9), 094502. doi: 10.1115/1.3148474

Feng, C. J., & Mak, A. F. T. (1998). *Neuromuscular model for the stretch reflex in passive movement of spastic elbow joint.* Paper presented at the 20th Annual

International Conference of the IEEE Engineering in Medicine and Biology Society.

Fox, M. D., Reinbolt, J. A., Ounpuu, S., & Delp, S. L. (2009). Mechanisms of improved knee flexion after rectus femoris transfer surgery. *J Biomech, 42*(5), 614-619. doi: 10.1016/j.jbiomech.2008.12.007

Friederich, J. A., & Brand, R. A. (1990). Muscle-Fiber Architecture in the Human Lower-Limb. *Journal of Biomechanics, 23*(1), 91-95.

Gage, J. R., Deluca, P. A., & Renshaw, T. S. (1995). Gait Analysis: Principles and Applications Emphasis on its use in Cerebral Palsy. *The Journal of Bone and Joint Surgery, 77-A*(10), 1607-1623.

Gage, J. R., Perry, J., Hicks, R. R., Koop, S., & Werntz, J. R. (1987). Rectus femoris transfer to improve knee function of children with cerebral palsy. *Dev Med Child Neurol, 29*(2), 159-166.

Goldberg, S. R., Anderson, F. C., Pandy, M. G., & Delp, S. L. (2004). Muscles that influence knee flexion velocity in double support: implications for stiff-knee gait. *Journal of Biomechanics, 37*(8), 1189-1196. doi: 10.1016/j.jbiomech.2003.12.005

Goldberg, S. R., Ounpuu, S., Arnold, A. S., Gage, J. R., & Delp, S. L. (2006). Kinematic and kinetic factors that correlate with improved knee flexion following treatment for stiff-knee gait. *J Biomech, 39*(4), 689-698. doi: 10.1016/j.jbiomech.2005.01.015

Goldberg, S. R., Ounpuu, S., & Delp, S. L. (2003). The importance of swing-phase initial conditions in stiff-knee gait. *Journal of Biomechanics, 36*(8), 1111-1116. doi: S0021929003001064 [pii]

Hamner, S. R., Seth, A., & Delp, S. L. (2010). Muscle contributions to propulsion and support during running. *Journal of Biomechanics, 43*(14), 2709-2716. doi: 10.1016/j.jbiomech.2010.06.025

Hernandez, A., Dhaher, Y., & Thelen, D. G. (2008). In vivo measurement of dynamic rectus femoris function at postures representative of early swing phase. *J Biomech, 41*(1), 137-144. doi: 10.1016/j.jbiomech.2007.07.011

Hill, A. V. (1938). The heat of shortening and the dynamic constants of muscle. *Proceedings of the Royal Society of London Series B-Biological Sciences, 126*(843), 136-195.

Honeycutt, A., Dunlap, L., Chen, H., al Homsi, G., Grosse, S., & Schendel, D. (2004). Economic Costs Associated with Mental Retardation, Cerebral Palsy, Hearing Loss, and Vision Impairment---United States *Morbidity and Mortality Weekly Report* (Vol. 53, pp. 57-59): Centers for Disease Control and Prevention.

Inman, V. T. (1976). *The Joints of the Ankle*. Baltimore, MD.

Johnson, K. L. (1985). *Contact Mechanics*. Cambridge.

Kruse, M., Michelsen, S. I., Flachs, E. M., Bronnum-Hansen, H., Madsen, M., & Uldall, P. (2009). Lifetime costs of cerebral palsy. *Dev Med Child Neurol, 51*(8), 622-628. doi: 10.1111/j.1469-8749.2008.03190.x

Lieber, R. L. (1990). Hypothesis: biarticular muscles transfer moments between joints. *Dev Med Child Neurol, 32*(5), 456-458.

Metaxiotis, D., Wolf, S., & Doederlein, L. (2004). Conversion of biarticular to monoarticular muscles as a component of multilevel surgery in spastic diplegia. *J Bone Joint Surg Br, 86*(1), 102-109.

Novacheck, T. F., Trost, J. P., & Sohrweide, S. (2010). Examination of the child with cerebral palsy. [Review]. *Orthop Clin North Am, 41*(4), 469-488. doi: 10.1016/j.ocl.2010.07.001

Perez-Gonzalez, A., Fenollosa-Esteve, C., Sancho-Bru, J. L., Sanchez-Marin, F. T., Vergara, M., & Rodriguez-Cervantes, P. J. (2008). A modified elastic foundation contact model for application in 3D models of the prosthetic knee. *Medical Engineering & Physics, 30*(3), 387-398. doi: DOI 10.1016/j.medengphy.2007.04.001

Perry, J. (1987). Distal rectus femoris transfer. *Dev Med Child Neurol, 29*(2), 153-158.

Reinbolt, J. A., Fox, M. D., Arnold, A. S., Ounpuu, S., & Delp, S. L. (2008). Importance of preswing rectus femoris activity in stiff-knee gait. *Journal of Biomechanics, 41*(11), 2362-2369. doi: DOI 10.1016/j.jbiomech.2008.05.030

Reinbolt, J. A., Fox, M. D., Schwartz, M. H., & Delp, S. L. (2009). Predicting outcomes of rectus femoris transfer surgery. *Gait & Posture, 30*(1), 100-105. doi: DOI 10.1016/j.gaitpost.2009.03.008

Riewald, S. A., & Delp, S. L. (1997). The action of the rectus femoris muscle following distal tendon transfer: does it generate knee flexion moment? *Dev Med Child Neurol, 39*(2), 99-105.

Schenau, G. J. v. I., Pratt, C. A., & Macpherson, J. M. (1994). Differential use and control of mono- and biarticular muscles. *Human Movement Science, 13*, 495-517.

Ting, L. H. (2007). Dimensional reduction in sensorimotor systems: a framework for understanding muscle coordination of posture. [Research Support, N.I.H., Extramural

Review]. *Prog Brain Res, 165*, 299-321. doi: 10.1016/S0079-6123(06)65019-X

Welch, T. D., & Ting, L. H. (2008). A feedback model reproduces muscle activity during human postural responses to support-surface translations. *J Neurophysiol, 99*(2), 1032-1038. doi: 10.1152/jn.01110.2007

Wickiewicz, T. L., Roy, R. R., Powell, P. L., & Edgerton, V. R. (1983). Muscle Architecture of the Human Lower-Limb. *Clinical Orthopaedics and Related Research*(179), 275-283.

Woollacott, M., Shumway-Cook, A., Hutchinson, S., Ciol, M., Price, R., & Kartin, D. (2005). Effect of balance training on muscle activity used in recovery of stability in children with cerebral palsy: a pilot study. *Dev Med Child Neurol, 47*(7), 455-461.

Woollacott, M. H., & Shumway-Cook, A. (2005). Postural dysfunction during standing and walking in children with cerebral palsy: what are the underlying

problems and what new therapies might improve balance? *Neural Plast, 12*(2-3), 211-219; discussion 263-272. doi: 10.1155/NP.2005.211

Zajac, F. E. (1989). Muscle and tendon: properties, models, scaling, and application to biomechanics and motor control. *Crit Rev Biomed Eng, 17*(4), 359-411.

APPENDIX

A. Muscle-tendon Length Properties

Number	Muscle	Optimal Fiber Length (m)	Tendon Slack Length (m)
1	glut_med1_r	0.04700941	0.06853708
2	glut_med2_r	0.07427225	0.04658496
3	glut_med3_r	0.05693475	0.04671117
4	glut_min1_r	0.06040022	0.01421182
5	glut_min2_r	0.04986999	0.02315392
6	glut_min3_r	0.03384964	0.04542977
7	semimem_r	0.07302276	0.32768962
8	semiten_r	0.18133122	0.23636209
9	bifemlh_r	0.09897910	0.30965021
10	bifemsh_r	0.15418315	0.08912321
11	sar_r	0.46470127	0.08936563
12	add_long_r	0.12604081	0.10046731
13	add_brev_r	0.12050907	0.01812166
14	add_mag1_r	0.07929207	0.08726646
15	add_mag2_r	0.11188879	0.11096409
16	add_mag3_r	0.12112047	0.24039178
17	tfl_r	0.08566467	0.38323668
18	pect_r	0.09045311	0.02984953
19	grac_r	0.31706787	0.12610654
20	glut_max1_r	0.12555508	0.11052383
21	glut_max2_r	0.13098572	0.11316453
22	glut_max3_r	0.12834970	0.12924101
23	iliacus_r	0.08818192	0.08818192
24	psoas_r	0.08777509	0.14044014
25	quad_fem_r	0.04788190	0.02128085
26	gem_r	0.02141315	0.03479638
27	peri_r	0.02281441	0.10090989

Number	Muscle	Optimal Fiber Length (m)	Tendon Slack Length (m)
28	rect_fem_r	0.10403328	0.28289752
29	vas_med_r	0.08121978	0.11498530
30	vas_int_r	0.07967188	0.12454455
31	vas_lat_r	0.07659759	0.14316454
32	med_gas_r	0.04885955	0.31758708
33	lat_gas_r	0.05206978	0.30916429
34	soleus_r	0.04052820	0.20264100
35	tib_post_r	0.02536956	0.25369562
36	flex_dig_r	0.02845807	0.33480077
37	flex_hal_r	0.03604234	0.31851373
38	tib_ant_r	0.08080175	0.18386521
39	per_brev_r	0.04128302	0.13293134
40	per_long_r	0.04066237	0.28629629
41	per_tert_r	0.06497645	0.08224867
42	ext_dig_r	0.08500692	0.28752340
43	ext_hal_r	0.09278526	0.25495049
44	glut_med1_l	0.04700941	0.06853708
45	glut_med2_l	0.07427225	0.04658496
46	glut_med3_l	0.05693475	0.04671117
47	glut_min1_l	0.06040022	0.01421182
48	glut_min2_l	0.04986999	0.02315392
49	glut_min3_l	0.03384964	0.04542977
50	semimem_l	0.07302276	0.32768962
51	semiten_l	0.18133122	0.23636209
52	bifemlh_l	0.09897910	0.30965021
53	bifemsh_l	0.15418315	0.08912321
54	sar_l	0.46470127	0.08936563
55	add_long_l	0.12604081	0.10046731

Number	Muscle	Optimal Fiber Length (m)	Tendon Slack Length (m)
56	add_brev_l	0.12050907	0.01812166
57	add_mag1_l	0.07929207	0.05468418
58	add_mag2_l	0.11188879	0.11096409
59	add_mag3_l	0.12112047	0.24039178
60	tfl_l	0.08566467	0.38323668
61	pect_l	0.09045311	0.02984953
62	grac_l	0.31706787	0.12610654
63	glut_max1_l	0.12555508	0.11052383
64	glut_max2_l	0.13098572	0.11316453
65	glut_max3_l	0.12834970	0.12924101
66	iliacus_l	0.08818192	0.08818192
67	psoas_l	0.08777509	0.14044014
68	quad_fem_l	0.04788190	0.02128085
69	gem_l	0.02141315	0.03479638
70	peri_l	0.02281441	0.10090989
71	rect_fem_l	0.10403328	0.28289752
72	vas_med_l	0.08121978	0.11498530
73	vas_int_l	0.07967188	0.12454455
74	vas_lat_l	0.07659759	0.14316454
75	med_gas_l	0.04885955	0.31758708
76	lat_gas_l	0.05206978	0.30916429
77	soleus_l	0.04052820	0.20264100
78	tib_post_l	0.02536956	0.25369562
79	flex_dig_l	0.02845807	0.33480077
80	flex_hal_l	0.03604234	0.31851373
81	tib_ant_l	0.08080175	0.18386521
82	per_brev_l	0.04128302	0.13293134
83	per_long_l	0.04066237	0.28629629

Number	Muscle	Optimal Fiber Length (m)	Tendon Slack Length (m)
84	per_tert_l	0.06497645	0.08224867
85	ext_dig_l	0.08500692	0.28752340
86	ext_hal_l	0.09278526	0.25495049
87	ercspn_r	0.10399923	0.02599981
88	ercspn_l	0.10399923	0.02599981
89	intobl_r	0.08666603	0.08666603
90	intobl_l	0.08666603	0.08666603
91	extobl_r	0.10399923	0.121334244
92	extobl_l	0.10399923	0.121334244

B. Main Program in Microsoft Visual C++

```
// Posture.cpp

 * Copyright (c)  2010 Stanford University
 * Use of the OpenSim software in source form is permitted provided
   that the following conditions are met:
 * 1. The software is used only for non-commercial research and
   education. It may not be used in relation to any commercial
   activity.
 * 2. The software is not distributed or redistributed.  Software
   distribution is allowed only through https://simtk.org/home/opensim.
 * 3. Use of the OpenSim software or derivatives must be acknowledged
   in all publications,presentations, or documents describing work in
   which OpenSim or derivatives are used.
 * 4. Credits to developers may not be removed from executables created
   from modifications of the source.
 * 5. Modifications of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
   FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE
   COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT,
   INDIRECT,INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
   (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
   SERVICES; LOSS OF USE, DATA, OR PROFITS; HOWEVER CAUSED AND ON ANY
   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR
   BUSINESS INTERRUPTION) OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
   ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
   OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 *  Main application for performing a posture control study.
 *
 *   Author:  Ajay Seth, Jeff Reinbolt, Ashley Clark
 *
//=============================================================================
//=============================================================================
#include <OpenSim/OpenSim.h>
#include <OpenSim/Common/IO.h>
#include <ctime>
#include "ReflexController.h"

using namespace OpenSim;
using namespace std;
using namespace SimTK;

bool PRINT_FLAG=true;

//_____________________________________________________________________________
/**
```

```
 *  Run  a  forward  simulation  of  muscle-driven  posture  model  with
platform perturbation
 *
 */
int  simulatePosture(Model  &model,  double  simulation_time,  double
platform_dx,  double  platform_dz,  double  perturbation_start,  double
perturbation_dt, string outPrefix)
{
try {
   // Start timing
   std::clock_t startTime = std::clock();

   // get coordinates to set initial conditions
   CoordinateSet& modelCoordinateSet =    model.updCoordinateSet();

   // Define the initial and final simulation times.
   double initialTime = 0.0;
   double finalTime = simulation_time;

   // Create the force reporter
   ForceReporter* reporter = new ForceReporter(&model);
   model.addAnalysis(reporter);

   // Create BodyKinematics analysis
   BodyKinematics* bodyKin = new BodyKinematics(&model);
   OpenSim::Array<std::string> names;
   names.append("center_of_mass");

   bodyKin->setBodiesToRecord(names);
   bodyKin->setRecordCenterOfMass(true);
   model.addAnalysis(bodyKin);

// Initialize the system and get a COPY of default state of the system.
   model.initSystem();

// create prescribed motion constraint automatically
   const Coordinate &platform_x =
   model.getCoordinateSet().get("platform_tx");

   SimTK::State si = model.getMultibodySystem().realizeTopology();
   model.initStateWithoutRecreatingSystem(si);

   ForceSet &forces = model.updForceSet();
   for(int i=0; i< forces.getSize(); i++){
      Muscle* m = dynamic_cast<Muscle*>(&forces[i]);
      if(m){
            double act =  0.02;
            m->setActivation(si, act);
            }
      }

// Load initial states.
   Storage initialStates("Initial_states_s26_com_rev2.sto");

// Set initial states on the model.
```

```
   initialStates.getData(0,si.getNY(),&si.updY()[0]);

// Compute initial conditions for muscles.
   model.computeEquilibriumForAuxiliaryStates( si );

// Create the integrator and manager for the simulation.
   SimTK::RungeKuttaMersonIntegrator                          integrator(
model.getMultibodySystem() );
   integrator.setAccuracy( 1.0e-4 );
   Manager manager(model, integrator);

// Print out the initial position and velocity states.
   if(PRINT_FLAG) {
      for( int i = 0; i < modelCoordinateSet.getSize(); i++ ) {
            double  units  =   modelCoordinateSet[i].getMotionType()  ==
Coordinate::Rotational ? SimTK_RADIAN_TO_DEGREE : 1.0;
            std::cout << "Initial " << modelCoordinateSet[i].getName()
            << " = " << modelCoordinateSet[i].getValue( si ) * units
            << ", and speed = "
            <<  modelCoordinateSet[i].getSpeedValue(  si  )  *  units  <<
std::endl;
            }
      }

//si.getQ().dump("Initial q's");
//si.getZ().dump("Initial z's");

// Integrate from initial time to final time.
   manager.setInitialTime( initialTime );
   manager.setFinalTime( finalTime );
   std::cout << "\n\nIntegrating from " << initialTime
   << " to " << finalTime << std::endl;
   manager.integrate( si );

// Save the simulation results.
   if(PRINT_FLAG) {

   model.printControlStorage(outPrefix+model.getName()+"_controls.sto"
);
      Storage statesDegrees( manager.getStateStorage() );

   statesDegrees.print(outPrefix+model.getName()+"_posture_states.sto"
);
      model.updSimbodyEngine().convertRadiansToDegrees(    statesDegrees
);
      statesDegrees.setWriteSIMMHeader( true );

   statesDegrees.print(outPrefix+model.getName()+"_posture_states_degre
es.mot" );
      }

// muscAnalysis->printResults("posture", "MuscleAnalysis");
   if(PRINT_FLAG) bodyKin->printResults(outPrefix+model.getName(), "");

// Center of mass position
```

```
   double *CoM=NULL;
   int ny = 0;
   int nyy = 0;
   int ncom = 0;
   Storage *posStore;
   posStore = bodyKin->getPositionStorage();

   Storage statesAndCOM;

// OPEN THE FILE
   FILE *fp = fopen("CoM.txt", "w");
   if(fp==NULL) return(-1);

// Save positions through time
   double timeStep = 0.01;
   int frames = simulation_time/timeStep;
   for(int i=0; i<=frames; i++) {
      double atTime = i*timeStep;
      ny = posStore->getDataAtTime(atTime, ny, &CoM);

// Get states to get platform position offset
   Storage statesDegrees( manager.getStateStorage() );
   model.updSimbodyEngine().convertRadiansToDegrees( statesDegrees );
      double *statesAtTime=NULL;
      nyy = statesDegrees.getDataAtTime(atTime, nyy, &statesAtTime);
      int tx_index = statesDegrees.getStateIndex("com_tx");
      int ty_index = statesDegrees.getStateIndex("com_ty");
      int tz_index = statesDegrees.getStateIndex("com_tz");
      OpenSim::Array<std::string> columnLabels;
      columnLabels = statesDegrees.getColumnLabels();


   statesAtTime[tx_index]=CoM[0];statesAtTime[ty_index]=CoM[1];statesAt
Time[tz_index]=CoM[2];
      statesAndCOM.append(atTime,nyy,statesAtTime);
      statesAndCOM.setColumnLabels(columnLabels);
      statesAndCOM.setName("States and COM");

      double                          platform_tx                          =
statesAtTime[statesDegrees.getStateIndex("platform_tx")];
      double                          platform_tz                          =
statesAtTime[statesDegrees.getStateIndex("platform_tz")];

// Convert to platform position
   CoM[0] -= platform_tx;
// CoM[1] is platform y-position is affected by massive (10Mg) platform
   CoM[2] -= platform_tz;

// Set state vector
   StateVector vec;
   vec.setStates(simulation_time,ny,CoM);

      fprintf(fp, "%f %f %f\n", CoM[0], CoM[1], CoM[2]);
      }
```

```
      statesAndCOM.setWriteSIMMHeader( true );

   statesAndCOM.print(outPrefix+model.getName()+"_states_and_com.mot"
);


// CLEANUP
   fclose(fp);
   if(CoM!=NULL) { delete[] CoM;  CoM=NULL; }

// Save the reulstant contact forces
   if(PRINT_FLAG)                                                reporter-
>getForceStorage().print(outPrefix+model.getName()+"_contact_forces.mot
");

   std::cout   <<   "Elapsed   time   =   "   <<   1.e3*(std::clock()-
startTime)/CLOCKS_PER_SEC << "ms\n" << std::endl;

// Model must disown or program will try to delete components twice and
crash
   model.disownAllComponents();
   }
   catch ( std::exception ex ) {

// In case of an exception, print it out to the screen.
   std::cout << ex.what() << std::endl;

// Return 1 instead of 0 to indicate that something
// undesirable happened.
   return 1;
    }

// If this program executed up to this line, return 0 to
// indicate that the intended lines of code were executed.
   return 0;

}
// END simulate posture with muscles


int main(int argc, char **argv)
{
// Default settings
// Length of simulation
   double simulation_time = 1.0; //seconds
// maginitude of platform peturbation displacement
   double platform_dx = 0.0;
// maginitude of platform perturbation displacement
   double platform_dz = 0.0;
// perturbation start time
   double perturbation_start = 0.0;
// time interval for platform movement
   double perturbation_dt = 0.0;
// Set gain for the controller.
   double lambda_L = 5.41368553976471;
```

```
   double lambda_V = 0.0196182496996364;
   double lambda_G = 8.95178523300332;
   double  maxContractionVel  =  10.0;  //  max  contraction  velocity  in
opt_fiber_length/s

   if (argc < 2)
   {
      cout   <<   "Using   default   parameters   for   simulation_time,
platform_dx,    platform_dz,    perturbation_start,    perturbation_dt,
reflex_gain, and maxContractionVel." << endl;
   }
   else {         //Command line parameter
      int i;
      for(i=1; i<=argc-1; i++) {
            if(i==1) simulation_time = atof(argv[i]);
            if(i==2) platform_dx = atof(argv[i]);
            if(i==3) platform_dz = atof(argv[i]);
            if(i==4) perturbation_start = atof(argv[i]);
            if(i==5) perturbation_dt = atof(argv[i]);
            if(i==6) lambda_L = atof(argv[i]);
            if(i==7) lambda_V = atof(argv[i]);
            if(i==8) lambda_G = atof(argv[i]);
            if(i==9) maxContractionVel = atof(argv[i]);
            if(i > 9) break;
      }

   }

   cout << "Current settings: simulation_time = " << simulation_time <<
", platform_dx = " << platform_dx
      <<  ",  perturbation_start  =  "  <<  perturbation_start  <<  ",
perturbation_dt = " << perturbation_dt <<
      ",  maxContractionVel = " << maxContractionVel << endl;

   Model model("s26_preop_v22_tx_tz_com.osim");

// SET OUTPUT PRECISION
   IO::SetPrecision(20);

   State &s = model.initSystem();

   CoordinateSet& modelCoordinateSet =    model.updCoordinateSet();

   if(model.getMuscles().getSize()){ // If a model with muscles
      // Print the control gains and block mass.
   std::cout << std::endl;

// Create the controller.
//   ReflexController   *controller   =   new   ReflexController(model,
reflex_gain, maxContractionVel);
   ReflexController *controller = new ReflexController(model, lambda_L,
lambda_V, lambda_G, maxContractionVel);

// Add the controller to the Model.
   model.addController(controller);
```

```
    }

    char dirString[256];
    sprintf(dirString, "Results/");
    string resultsDir = string(dirString);
    if(PRINT_FLAG) {
       IO::makeDir(resultsDir);
    }

    int failed = simulatePosture(model, simulation_time, platform_dx,
platform_dz, perturbation_start, perturbation_dt, resultsDir);
    return failed;
}
```

C. *Reflex Controller Header File in Microsoft Visual C++*

```
//ReflexController.h
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
/*
 * Copyright (c)  2010 Stanford University
 * Use of the OpenSim software in source form is permitted provided
   that the following conditions are met:
 * 1.  The software is used only for non-commercial research and
   education.  It may not be used in relation to any commercial
   activity.
 * 2.  The software is not distributed or redistributed.   Software
   distribution is allowed only through https://simtk.org/home/opensim.
 * 3. Use of the OpenSim software or derivatives must be acknowledged
   in all publications,presentations, or documents describing work in
   which OpenSim or derivatives are used.
 * 4. Credits to developers may not be removed from executables created
   from modifications of the source.
 * 5.  Modifications of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
   FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE
   COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT,
   INDIRECT,INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
   (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
   SERVICES; LOSS OF USE, DATA, OR PROFITS; HOWEVER CAUSED AND ON ANY
   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR
   BUSINESS INTERRUPTION) OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
   ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
   OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * Main application for performing a posture control study.
 *
 * Author:  Ajay Seth, Jeff Reinbolt, Ashley Clark
 *
//=============================================================================
//=============================================================================
#ifndef _ReflexController_h_
#define _ReflexController_h_

//=============================================================================
// INCLUDE
//=============================================================================
#include <OpenSim/Simulation/Control/Controller.h>

//=============================================================================
//=============================================================================
/**
 * ReflexController is a concrete controller that excites muscles in
   response
```

```
 * to muscle lengthening to simulate a stretch reflex.
 *
 * @author  Ajay Seth
 * @version 1.0
 */
namespace OpenSim {

class ReflexController : public Controller {
public:
/**
 * Constructor
 *
 * @param aModel Model to be controlled
 * @param gain, reflex gain by which the excitation of the muscle is
increased by muscle fiber lengthening speed
 */

ReflexController( Model& aModel, double lambda_L, double lambda_V,
double lambda_G, double maxContractVel ) :
      Controller( aModel ), _lambda_L(lambda_L),  _lambda_V(lambda_V),
_lambda_G(lambda_G),  _maxContractionVelocity(maxContractVel) {
   }

/** Copy constructor */
ReflexController(const      ReflexController      &aController)      :
Controller(aController){
      _lambda_L = aController._lambda_L;
      _lambda_V = aController._lambda_V;
      _lambda_G = aController._lambda_G;
      _maxContractionVelocity = aController._maxContractionVelocity;
   }

/** Get a a copy of this controller.  */
Object* copy() const
   {
   return(new ReflexController(*this));
   }

/**
 * Compute the control an actuator given the current state
 *
 * @param s Current state of the system
 * @param index Index of the current actuator whose control is being
   calculated
 * @return Control value to be assigned to the current actuator at the
   current time
 */
virtual double computeControl( const SimTK::State& s, int index )const
   {
// Get the current time in the simulation.
   double t = s.getTime();

/** Array of resting lengths when the simulation starts */
   static double _L_0Array[200];
```

```
/** Array of times when the activation from stretch reflex becomes
positive */
   static bool _L_dot_flagArray[200];

/** Array of times when the activation from stretch reflex becomes
positive */
   static double _t1Array[200];

/** Array of muscle lengths at times when the activation from stretch
reflex becomes positive */
   static double _L_t1Array[200];

/** Array of muscle velocities at times when the activation from
stretch reflex becomes positive */
   static double _L_dot_t1Array[200];

// Get a pointer to the current muscle whose control is being
// calculated.
   Thelen2003Muscle*                          mus                          =
dynamic_cast<Thelen2003Muscle*>(&_actuatorSet.get(index));

   if(mus == NULL)
      return 0;

// Get the speed of contraction
   _model->getMultibodySystem().realize(s, SimTK::Stage::Velocity);
//double speed = mus->computeLengtheningSpeed(s); // * working reflex

// Set default values for arrays
   if(t==0.0) {
      _L_0Array[index] = mus->getFiberLength(s);
      _L_dot_flagArray[index] = false;
      _t1Array[index] = index;
      _L_t1Array[index] = 0.0;
      _L_dot_t1Array[index] = 0.0;
   }

// Now, compute the control value for the current muscle.
// apply reflex if the muscle is lengthening (i.e. speed is positive)

// Stretch reflex model from Feng & Mak, 1998
   double lambda_L = _lambda_L; // 10.0;
   double lambda_V = _lambda_V; // 0.1;
   double lambda_G = _lambda_G; // 0.25;
   double L_0 = _L_0Array[index]; // Is this resting length?
   double L = mus->getFiberLength(s);
   double L_dot = mus->getFiberLengthDeriv(s);

   double L_R = L - L_0;
   if(L_R < 0) {
      L_R = 0; //if stretched, set to zero (unstretched)
   } else if (L_R > L_0) {
      L_R = L_0; //if fiber length > 2x resting length, set to resting
length
   }
```

```
   double V_max = mus->getVmax();
   double T_L = L_0-(lambda_L*L_R);
   double T_V = lambda_V*V_max*(L_R/L_0); //Feng & Mak 1998

   if(L_dot > 0.0 && _L_dot_flagArray[index] == false && t >
_t1Array[index]) {
      _L_dot_flagArray[index] = true;
      _t1Array[index] = t;
      _L_t1Array[index] = L;
      _L_dot_t1Array[index] = L_dot;
   } else if (L_dot <= 0.0 && _L_dot_flagArray[index] == true && t >
_t1Array[index]) {
      _L_dot_flagArray[index] = false;
   }
   double t1 = _t1Array[index];
   double L_t1 = _L_t1Array[index];
   double L_dot_t1 = _L_dot_t1Array[index];

//Feng & Mak 1998
   double newControl;
   if(L>T_L && L_dot>T_V) {
      newControl = lambda_G*((L-T_L) / (L_0+L_R) + (L_dot-T_V) /
V_max);
   } else if((t-t1)<0.050 && (L<=T_L || L_dot<=T_V)) {
      newControl = lambda_G*((L_t1-T_L) / (L_0+L_R) + (L_dot_t1-T_V) /
V_max);
   } else {
      newControl = 0;
      }

// Return the final computed control value for the current muscle.
   if (newControl > 1.0)
      newControl = 1.0;
   else if (newControl < 0.001)
      newControl = 0.001;

   return newControl;
   }

protected:

/**
 * ModelComponent interface for setting up the underlying computational
system
 *
 * @param aModel Model to be controlled
 */
   virtual void setup(Model &model)
   {
   Controller::setup(model);
// reset the set of actuators to be controlled by this controller
   _actuatorSet.setSize(0);

   std::cout << std::endl;
```

```
// Select muscles from these actuators of the model and set their
indices
// for the reflex controller
//Set<Actuator> &acts = _model->updActuators();
   Set<Actuator> &muscles = (Set<Actuator> &)(_model->updMuscles());
   setActuators(muscles);
   _numControls = _actuatorSet.getSize();
   printf("  ReflexController::setup  added  %d  Muscle  reflexes\n",
_numControls );
   }


// This section contains the member variables of this controller class.
private:
/** Reflex gain for the controller */
   double _lambda_L;
   double _lambda_V;
   double _lambda_G;

/** Maximum contraction velocity by which to normalize contraction
velocity */
   double _maxContractionVelocity;

};


}; //namespace
//=============================================================================
//=============================================================================

#endif // _ReflexController_h_
```

D. Model Characteristics & Simulation Results

Model characteristics and RMS difference from zero CoM displacement for preoperative, unilateral, bilateral, and monoarticular simulations. Simulations which recovered an upright posture are highlighted in gray.

Model Name	Model Characteristics	RMS Difference	
		Anterior Translation	Posterior Translation
Preoperative	Nominal	1.67	1.98
Unilateral	Left transferred rectus femoris	1.77	3.81
Bilateral	Both transferred rectus femoris	5.27	3.87
Monoarticular	No vastus medialis	1.37	--

E. Glossary

Abduction	Movement away from the midline of the body in the coronal plane.
Acceleration	The time rate of change of velocity.
Adduction	Movement towards the midline of the body in the coronal plane.
Ankle motion	The ankle angles reflect the motion of the foot segment relative to the shank segment.
Anterior	The front or before, also referred to as ventral.
Balance	A generic term describing the dynamics of body posture to prevent falling. It is related to the inertial forces acting on the body and the inertial characteristics of body segments.
Center of mass (CoM)	The point at which the entire mass of a body may be considered concentrated for some purposes. The point such that the first moment of a physical object about every line through the point is zero.
Constraint functions	Specific limits that must be satisfied by the optimal design.
Degree of freedom (DOF)	A single coordinate of relative motion between two bodies. Such a coordinate responds without constraint or imposed motion to externally applied forces or torques. For translational motion, a DOF is a linear coordinate along a single direction. For rotational motion, a DOF is an angular coordinate about a single, fixed axis.
Design variables	Variables that change to optimize the design.
Distal	Away from the point of attachment or origin.
Dorsiflexion	Movement of the foot towards the anterior part of the tibia in the sagittal plane.
Eversion	A turning outward.
Extension	Movement that rotates the bones comprising a joint away from each other in the sagittal plane.
Femur	The longest and heaviest bone in the body. It is located between the hip joint and the knee joint.
Flexion	Movement that rotates the bones comprising a joint towards each other in the

	sagittal plane.
Force	A push or a pull and is produced when one object
Force plate	A transducer that is set in the floor to measure about some specified point, the force and torque applied by the foot to the ground. These devices provide measures of the three components of the resultant ground reaction force vector and the three components of the resultant torque vector.
Gait	A manner of walking or moving on foot.
Generalized coordinates	A set of coordinates (or parameters) that uniquely describes the geometric position and orientation of a body or system of bodies. Any set of coordinates that are used to describe the motion of a physical system.
Graphical User Interface (GUI)	A software interface designed to standardize and simplify the use of computer programs, as by using a mouse to manipulate text and images on a display screen featuring icons, windows, and menus.
Hip motion	The hip angles reflect the motion of the thigh segment relative to the pelvis.
Inferior	Below or at a lower level (towards the feet).
Inverse dynamics	Analysis to determine the forces and torques necessary to produce the motion of a mechanical system, given the topology of how bodies are connected, the kinematics, the mass properties, and the initial condition of all degrees of freedom.
Inversion	A turning inward.
Kinematics	Those parameters that are used in the description of movement without consideration for the cause of movement abnormalities. These typically include parameters such as linear and angular displacements, velocities and accelerations.
Kinetics	General term given to the forces that cause movement. Both internal (muscle activity, ligaments or friction in muscles and joints) and external (ground or external loads) forces are included. The moment of force produced by muscles crossing a joint, the mechanical power flowing to and from those

	same muscles, and the energy changes of the body that result from this power flow are the most common kinetic parameters used.
Knee abduction-adduction	Motion of the long axis of the shank within the coronal plane as seen by an observer positioned along the anterior-posterior axis of the thigh.
Knee flexion-extension	Motion of the long axis of the shank within the sagittal plane as seen by an observer positioned along the medial-lateral axis of the thigh.
Knee internal-external rotation	Motion of the medial-lateral axis of the shank with respect to the medial-lateral axis of the thigh within the transverse plane as viewed by an observer positioned along the longitudinal axis of the shank.
Knee motion	The knee angles reflect the motion of the shank segment relative to the thigh segment.
Lateral	Away from the body's longitudinal axis, or away from the mid-sagittal plane.
Markers	Active or passive objects (balls, hemispheres or disks) aligned with respect to specific bony landmarks used to help determine segment and joint position in motion capture.
Medial	Toward the body's longitudinal axis, or toward the mid-sagittal plane.
Model parameters	A set of coordinates that uniquely describes the model segments lengths, joint locations, and joint orientations, also referred to as joint parameters. Any set of coordinates that are used to describe the geometry of a model system.
Moment of force	The moment of force is calculated about a point and is the cross product of a position vector from the point to the line of action for the force and the force. In two-dimensions, the moment of force about a point is the product of a force and the perpendicular distance from the line of action of the force to the point. Typically, moments of force are calculated about the center of rotation of a joint.

Motion capture	Interpretation of computerized data that documents an individual's motion.
Objective functions	Figures of merit to be minimized or maximized.
Parametric	Of or relating to or in terms of parameters, or factors that define a system.
Passive markers	Joint and segment markers used during motion capture that reflect visible or infrared light.
Pelvis	Consists of the two hip bones, the sacrum, and the coccyx. It is located between the proximal spine and the hip joints.
Posterior	The back or behind, also referred to as dorsal.
Posture	The orientation of any body segment relative to the gravitational vector; an angular measure from the vertical.
Proximal	Toward the point of attachment or origin.
Range of motion	Indicates joint motion excursion from the maximum angle to the minimum angle.
Sagittal plane	The plane that divides the body or body segment into the right and left parts.
Subtalar joint	Located between the distal talus and proximal calcaneous, also known as the talocalcaneal joint.
Superior	Above or at a higher level (towards the head).
Swing phase	The period of time when the foot is not in contact with the ground.
Talus	The largest bone of the ankle transmitting weight from the tibia to the rest of the foot.
Tibia	The large medial bone of the lower leg, also known as the shinbone. It is located between the knee joint and the talocrural joint.
Transverse plane	The plane at right angles to the coronal and sagittal planes that divides the body into superior and inferior parts.
Velocity	The time rate of change of displacement.

Vita

Ashley Elizabeth Clark was born in 1987 in Bradenton, Florida to Gary and Barbara Clark. She has two younger brothers, Robert and Matthew. In 1994, her family moved to Manhattan, KS where Ashley graduated from Manhattan High School in 2005. Following high school, Ashley began her studies in Biological & Agricultural Engineering at Kansas State University. During college, Ashley participated in a study abroad experience in Monterrey, Mexico at the Instituto Tecnológico de Estudios Superiores de Monterrey. While at KSU, she was involved in organizations such as American Society of Agricultural and Biological Engineers, Society of Women Engineers, Women Mentoring Women, Engineering Ambassadors, and Steel Ring Engineering Honor Society. Ashley was also actively involved as an undergraduate research assistant in a water quality lab. Her research was presented and recognized at local, regional, and international conferences, and contributed towards her honors research. Ashley graduated with honors from Kansas State University in May 2009 with a Bachelor of Science in Biological & Agricultural Engineering, a secondary degree in Biological Engineering, and minor degrees in Spanish and Biology. In August 2011 Ashley began her graduate studies in Biomedical Engineering and was appointed a graduate research assistantship in the Reinbolt Research Group at the University of Tennessee. Her research in neuromuscular biomechanics has been presented and recognized at national and international conferences. Following completion of her Master of Science degree, Ashley will continue her education at the University of Kansas School of Medicine.