University of Zurich[UZH]

Department of Informatics

Master Project Report

Simulation of the Ontogenetic Development of Human Biped Walking

Tobias Klauser
Winterthur, Switzerland
Registration Number: 05-535-927

Written at the Artificial Intelligence Laboratory

Supervized by:
Prof. Dr. Rolf Pfeifer
Dr. Hugo Gravato Marques

April 30, 2013

Abstract

Coordinating and learning to coordinate the movement of the human body is a non-trivial problem. One theory of how this could possibly be achieved suggests that complex movements are a result of the combination of simple, rudimentary building blocks called motor primitives.

In the context of this project thesis data from a study on the development of motor primitives for biped locomotion was applied to a musculoskeletal model of the human legs in the OpenSim simulation software. The model could not be made to reproduce walking behavior, however valuable insights were gained on the capabilities of the OpenSim environment. OpenSim is targeted at – and primarily used in – studies in biomechanics were detailed experimental data are available. Thus the application of OpenSim to the analysis of more abstract data is not easily possible and would call for the availability of additional high resolution data and the precise modeling of ground contact forces.

Contents

1 Introduction

Controlling and learning to control movements of a many degree of freedom system such as the human body is a non-trivial task. How does the central nervous system (CNS) choose from the infinitely many possibilities of how to achieve a certain movement? This problem is famously known as Bernstein's problem [2]. Understanding how natural agents such as humans and animals control their movements is one of the fundamental questions in embodied artificial intelligence [14, 15] but also holds insights for the field of robotics.

Complex movements of humans (and other animals such as cats [19]) are thought to develop based on a simple, rudimentary set of building blocks – commonly called motor primitives or muscle synergies – during ontogenetic development [5, 7, 10, 18]. These primitives are defined by the pattern of simultaneous activation of different muscles and are then combined in order to achieve a certain movement or pose.

Contrary to the hitherto belief that these innate simple reflexes get completely replaced by more complex ones with age, a recent publication by Dominici et al. [4] shows that certain basic patterns of muscle activation involved in biped walking are instead retained and are being built upon during development in order to achieve a variety of walking movement behaviors. The study measured EMG activity of 24 muscles (the same 12 muscles in each leg) simultaneously in neonates[1] ($\sim$ 3 days old), toddlers (11-14 months old), preschoolers (22-48 months old) and adults (25-40 years old). Additionally foot pressure was recorded as well as the limb kinematics.

This project thesis attempts to investigate the role these locomotor primitives play in the development of human bipedal walking by building a musculoskeletal simulation model in the OpenSim software [3] and then applying the basic activation patterns at different stages of human development (neonate, toddlers, preschooler and adults) identified by Dominici et al. [4] to it in order to achieve a walking behavior. Thereby we hope to understand the basis of the development of biped walking in order to possibly be applied to humanoid robots, especially anthropomimetic robots [21] which more closely mimic the internal structure of the human body with respect to muscles, tendons, bones and joints.

[1] Even though neonates of course can't walk by themselves, stepping can be evoked by holding them upright and then gently pushing the infant along the pathway. However this behavior typically disappears 4 to 6 weeks after birth.

The basic process of developing the necessary components and finally reproducing the walking behavior will consist of the following distinct steps:

Single joint model Implement a simple musculoskeletal model with one joint to get familiar with OpenSim. Actuate it using a reflex controller.

Full two-legged model Implement the full musculoskeletal model with two legs, fixed in space without ground forces. Actuate it using a simple reflex controller and pseudo-random data resembling the activation patterns from the study.

Extract data and investigate walking Extend the model to include the ground-contact forces accordingly. Extract the real activation pattern data from [4] and apply it to the model. Examine the walking behavior of the musculoskeletal model.

The remainder of this report is organized as follows: In chapter 2, the simulation software environment, the musculoskeletal model and the developed simulation controller are described. Furthermore the data preparation is explained. In chapter 3 the data produced and the simulation behaviors are presented. In chapter 4 the obtained results are discussed and the usability of the OpenSim environment for these kinds of experiments is evaluated. Finally in chapter 5 some conclusions are drawn.

2 Methods

2.1 Simulation Software

Simulation is performed using the OpenSim[1] software [3]. OpenSim is an open- source platform for modeling and analyzing neuromusculoskeletal structures and simulating their dynamic movement behavior. It's developed and maintained at the NIH Center for Biomedical Computation at Stanford University and is distributed as part of the Simtk.org repository.

In OpenSim, a simulation typically consists of a musculoskeletal model and a controller. The model can either be built individually or one of the models provided with OpenSim can be used. The model contains any combination of rigid bodies (such as bones), joints (simple or complex), constraints, springs, dampers, muscles and other actuators. A model is then assigned a controller which defines its dynamic behavior during simulation. There are predefined controllers available for forward dynamics (drive simulation with given muscle excitations), inverse kinematics (position model in simulation according to experimental markers and coordinate data at each time step) and inverse dynamics (determine generalized forces at each joint responsible for a given movement).

The software provides an extensive C++ application programming interface (API) in order to develop custom simulation controllers. The SimTK multi body dynamics API (Simbody) is used as the basic physics engine. Additionally OpenSim contains a graphical user interface (GUI) used to visualize models and simulations, edit models and muscle excitation data, as well as to access predefined simulation controllers. The GUI can be used independently from the simulation and the visualization is done using the data previously generated in the offline simulation.

The controllers provided with the OpenSim distribution turned out to require a very specific data collection as their input. This type of data is typically generated during biomechanics experiments, where muscle activations are measured using electromyography (EMG), the kinematics are tracked using a motion capturing system and force plates are used to register ground contact forces.

[1] Not to be confused with the 3D virtual world software OpenSimulator, which is sometimes also abbreviated as OpenSim.

In the context of this project, no such data was available or could easily be acquired. Instead we rather relied on relatively sparse muscle activation patterns and have no ground force measurements or tracked kinematic paths available. Also the interest is in general principles of movement rather than the detailed analysis of the walking behavior of an individual subject.

Thus a custom controller making use of the provided OpenSIM C++ API was developed which actuates the model based on the basic activation patterns extracted from [4]. This approach also allows to specifically tune the simulation behavior to the requirements of this project and the data available.

For all tasks performed as part of this project, version 3.0 of OpenSim was used as provided on `http://www.simtk.org`. The source version was used and built on Ubuntu Linux according to the provided instructions. Since the Linux version of OpenSim does not provide the GUI, the Windows version of OpenSim was used for visualizations.

2.2 Musculoskeletal Model

2.2.1 Simple Leg Model

In order to get familiar with modeling and controlling models in OpenSim, a simple and abstracted skeletal model of a human leg was created. It consists of three rigid bodies – *pelvis*, *femur* and *tibia* – and two joints – hip and knee. The pelvis was fixed in space, the femur attached to it using a pin joint (one degree-of-freedom) and the *tibia* attached to the *femur* using a pin joint. The model can be actuated using two muscles: *Rectus Femoris* (anterior) and *Biceps Femoris* (posterior), which are in this abstract model an agonist/antagonist pair of muscles and represent the largest muscles in the human lower extremity. They're attached to the pelvis on one end and to the femur and the tibia on the other end respectively. See figure 2.1 for a visualization of the model.

2.2.2 Two-legged Model

In the study by Dominici et al. [4] the EMG activity of a selected set of 12 muscles was recorded for each leg of the subject. The following muscles were included in the publication:

- *Tibialis Anterior (TA)*
- *Gastrocnemius Lateralis (LG)*

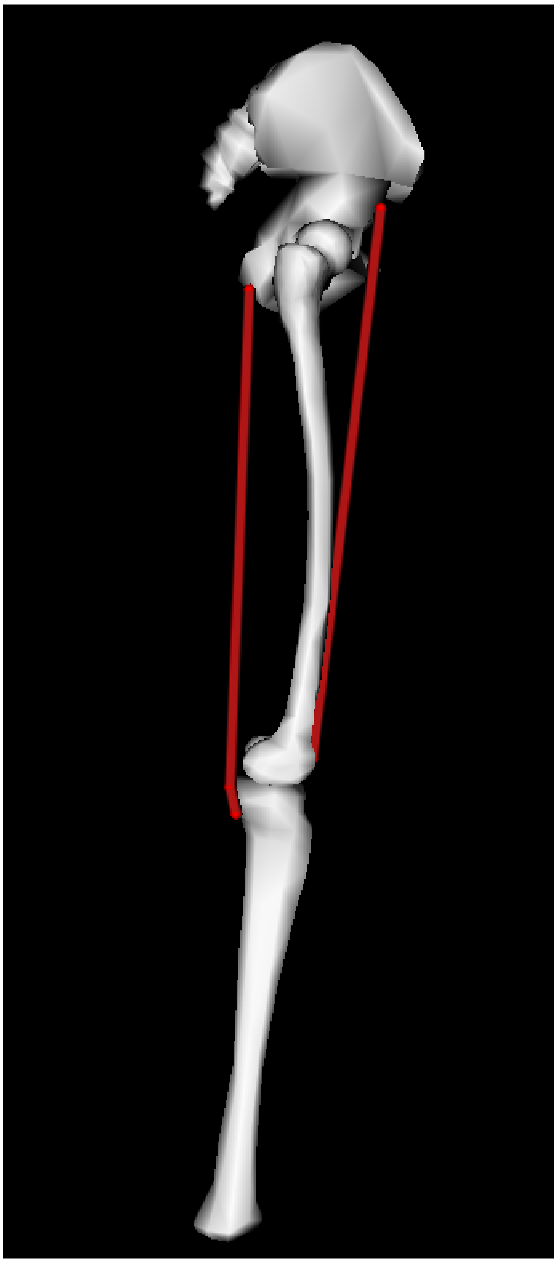

Figure 2.1: Visualization of the SimpleLeg model. The topmost bone (*pelvis*) is fixed in space, with the *femur* and *tibia* attached using pin joints. Actuation is achieved using the *Rectus Femoris* (right) and *Biceps Femoris* (left) muscles.

- *Gastrocnemius Medialis (MG)*
- *Soleus (Sol)*
- *Vastus Lateralis (VL)*
- *Vastus Medialis (VM)*
- *Rectus Femoris (RF)*
- *Hamstring (HS)* – a group of four tigh muscles and their respective tendons, namely *Semitendinosus*, *Semimembranosus*, *Biceps Femoris-Long Head* and *Biceps Femoris-Short Head*
- *Adductor Longus (Add)*
- *Tensor Fascia Latae (TFL)*
- *Gluteus Maximus (GM)*
- *Erector Spinae (ES)*

The musculoskeletal model to be used thus had to incorporate all these muscles for each leg, so the proper activation patterns could be applied.

Several musculoskeletal models of human legs are already provided with OpenSim 3.0. For this project the *Gait2354* model was used as a basis. This is a 23-degree-of-freedom model of the musculoskeletal system of the human leg. By default it contains 54 musculo-tendon actuators and represents a subject with a height of 1.8 m and a mass of 75.16 kg[2]. See figure 2.2 for a visualization of the model. Since the model represents an adult subject, the adult data from [4] was of primary interest. However OpenSim provides means to scale a given musculoskeletal model to different sizes, so the model could eventually be adapted to represent a neonate, toddler or preschooler respectively.

The model was then adapted in order for its final version to only incorporate the muscles for which publication data was available. Table 2.1 shows the muscles from the study and the *Gait2354* musculoskeletal model respectively and how their activation was determined from the study data in the final model. Some of the muscles (the *Hamstring* and *Gluteus Maximus* specifically) were measured in combination for the study, but the muscles were present individually in the model. They were left as is and later on the controller was adapted to distribute the entire activation among the individual muscles (see 2.4.2 for details). Furthermore for some of the muscles in the original *Gait2354* model no activation data was available from the study, thus the

[2] More information on the *Gait2354* model and its kinematic and dynamic properties as well as references to the studies the model is based on can be found on `http://simtk-confluence.stanford.edu:8080/display/OpenSim/Gait+2392+and+2354+Models`

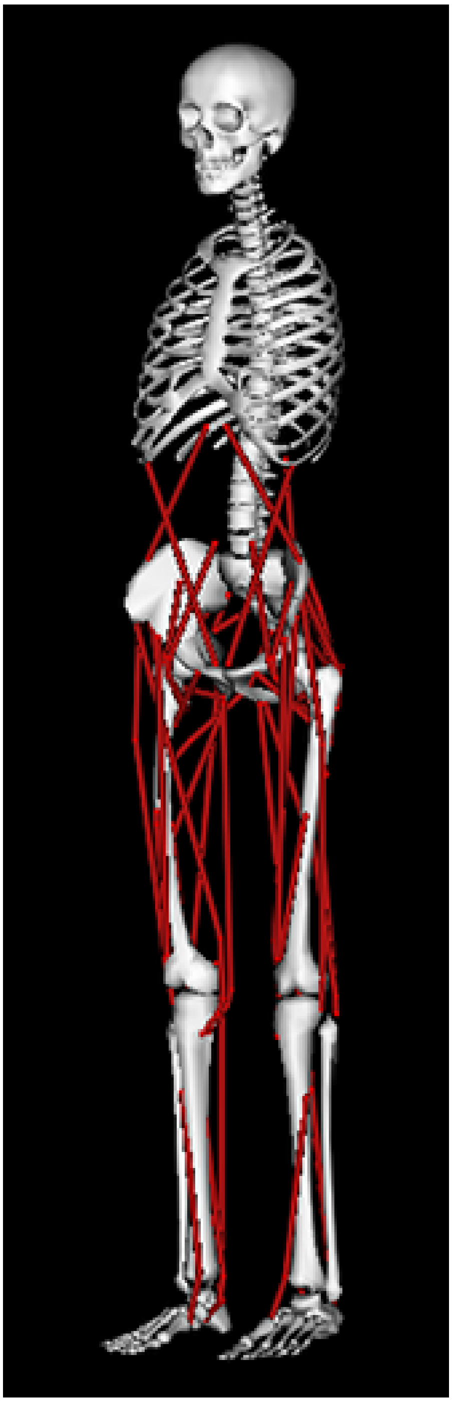

Figure 2.2: Visualization of the *Gait2354* model

respective muscles were removed from the model. The table shows only the muscles for one leg, the muscles of the second leg were handled correspondingly. The leg to which each muscle in the model belongs, is signified by it having a postfix *_l* for the left leg, or a postfix *_r* for the right leg.

Muscle	**Name in model**	**Name in [4]**	**Activation Usage**
Adductor Longus	add_long	Add	used directly
Gastrocnemius Lateralis	lat_gas	LG	used directly
Gastrocnemius Medialis	med_gas	MG	used directly
Rectus Femoris	rect_fem	RF	used directly
Soleus	soleus	Sol	used directly
Tensor Fasciae Latae	tfl	TFL	used directly
Tibialis Anterior	tib_ant	TA	used directly
Vastus Lateralis	vas_lat	VL	used directly
Vastus Medialis	vas_med	VM	used directly
Biceps Femoris-Long Head	bifemlh_r	part of HS	$\frac{1}{4}$ of HS activation
Biceps Femoris-Short Head	bifemsh_r	part of HS	$\frac{1}{4}$ of HS activation
Semimembranosus	semimem	part of HS	$\frac{1}{4}$ of HS activation
Semitendinosus	semiten	part of HS	$\frac{1}{4}$ of HS activation
Gluteus Maximus 1	glut_max1	part of GM	$\frac{1}{3}$ of GM activation
Gluteus Maximus 2	glut_max2	part of GM	$\frac{1}{3}$ of GM activation
Gluteus Maximus 3	glut_max3	part of GM	$\frac{1}{3}$ of GM activation
Adductor Magnus 2	add_mag2	-	removed from model
Erector Spinae	ercspn	ES	removed from model
Gluteus Medius 1	glut_med1	-	removed from model

Gluteus Medius 2	glut_med2	-	removed from model
Gluteus Medius 3	glut_med3	-	removed from model
Gracilis	grac	-	removed from model
Iliacus	iliacus	-	removed from model
Internal Oblique	intobl	-	removed from model
Inferio gemellus	gem	-	removed from model
Pectineus	pect	-	removed from model
Piriformis	peri	-	removed from model
Psoas Major	psoas	-	removed from model
Quadratus Femoris	quad_fem	-	removed from model
Sartorius	sar	-	removed from model
Tibialis Posterior	tib_post	-	removed from model
Vastus Intermedius	vas_int	-	removed from model

Table 2.1: Muscles in the *Gait2354* model and their correspondents in [4].

Because no data was available for the *External Oblique (OE)* muscle – one of the muscles which connects the torso to the pelvis – the torso was removed from the model entirely, because otherwise only the back muscles – the left and right *Erector Spinae (ES)* – would connect the torso to the pelvis, but would have no antagonist. Therefor *OE* and *ES* were removed, resulting in a total of 16 muscles per leg (32 total) in the final version of the model.

The joints were left as in the original *Gait2354 models.* For initial experiments the “joint” connecting the pelvis to the world was fixed in space using the zero-degree-of-freedom *WeldJoint* joint type of OpenSim.

Figure 2.3 shows the resulting musculoskeletal model.

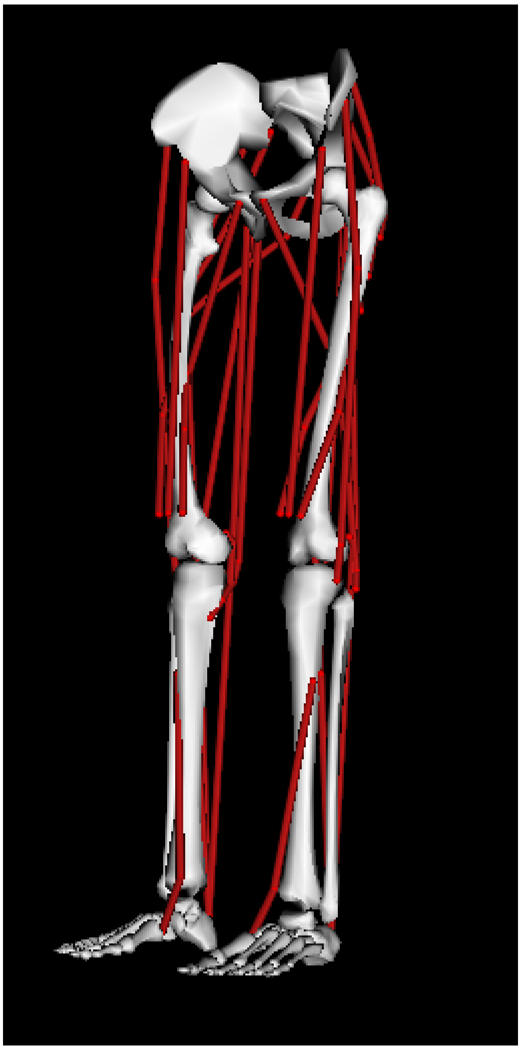

Figure 2.3: Visualization of the final musculoskeletal model used for the project.

2.3 Muscle Model

2.3.1 Thelen Muscle Model

For all muscles in the model, the Thelen mathematical muscle model [17] was used. This model is the default muscle model used by OpenSim 3.0[3] and is available in

[3] Note that up to version 2.4 OpenSim contained numerical errors in the implementation of the Thelen muscle model. This deprecated version is still available in the software in the C++ class `OpenSim::Thelen2003Muscle_Deprecated` but shouldn't be used for actual simulations.

the C++ class `OpenSim::Thelen2003Muscle`. This muscle model is employed in all musculoskeletal models provided with OpenSim.

The Thelen muscle model is based on the well-known and widely used Hill-type muscle-tendon model [8, 9, 16, 22], but incorporates adjustments in order to better account for age-related changes in the muscle properties.

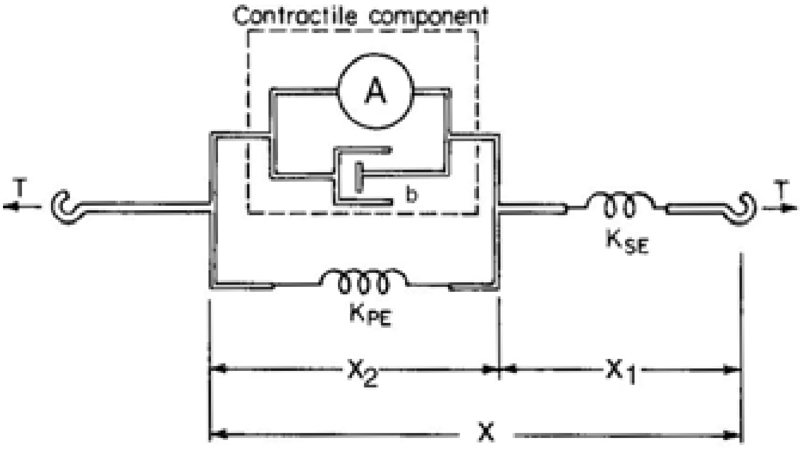

Figure 2.4: Schematic of the Hill-type mathematical muscle model. Illustration taken from [16]

In the Thelen as well as in the Hill model, the muscle-tendon complex of an actuator consists of three components: a *contractile element (CE)*, a *passive element (PE)*, and a *series elastic element (SEE or SE)*. See figure 2.4 for an illustration, where T is the tension, K_{SE} the spring constant of the *SE*, K_{PE} the spring constant of the *PE*, b the damping coefficient of the *CE* and A the active force of the muscle.

The following parameters are used to characterize each Thelen-type muscle: maximum isometric force, optimal muscle fiber length, tendon slack length, maximum contraction velocity, and pennation angle. During simulation, the muscle force is calculated using two states: the activation value and the muscle fiber length. All these parameters were left intact for the adjusted model.

2.4 Simulation Controller

2.4.1 Simple Leg Controller

The *Simple Leg Controller* was used in order to create a simulation using the Simple Leg model described in 2.2.1 in order to get familiar with the OpenSim workflow.

It implements a behavior similar to the stretch reflex (*myotatic reflex*): An increase of muscle length causes the muscle spindles to be stretched, leading them to increase neural activity. This increases the activity of alpha motor neurons, which cause the muscle fibers to contract in order to regain the original muscle length. Using a second set of neurons, the antagonist muscle is caused to relax [1].

OpenSim provides access to the lengthening speed of each muscle during simulation. This was used in the controller to deduce increases in muscle length and cause the corresponding reflex.

2.4.2 Locomotor Primitives Controller

All the predefined controllers available with OpenSim require a specific set of input data typically resulting from biomechanics studies using EMG, tracker systems and force plates. Since such data was not available in the context of this project, a custom controller for the simulation of the musculoskeletal model was developed in order to account for the limited availability of data.

The *Locomotor Primitives Controller* allows to actuate the model described in section 2.2.2 using only the activation patterns for its 32 muscles (16 per leg). The data can be fed into the controller using the *OpenSim Storage file format* (file extension `.sto`), which is the standard format to store and read data in OpenSim[4].

All the activation data from the storage file are read into the controller and are accessible using the column name, which corresponds to the muscle name according to table 2.1 with the proper postfix. The controller checks the availability of activation data for all muscles for the entire simulation time and is then able to retrieve the corresponding activation for each muscle at each time step.

The time step resolution in OpenSim by default is dynamic and determined by the integrator used. For this project a Runge-Kutta integrator in the implementation of `SimTK::RungeKuttaMersonIntegrator` was used. This is the suggested default integrator in OpenSim.

Since the time step is not constant when using this integration method, OpenSim allows to interpolate the data read from the storage file in order to provide activation values at all possible time steps during the simulation. The total duration of the simulation can be specified as a parameter to the controller, so interpolation of the data points is also crucial for that matter.

[4] The file format is explained in `http://simtk-confluence.stanford.edu:8080/display/OpenSim/Storage+(.sto)+Files`, but there are some subtleties to the be considered when generating `.sto` files outside of OpenSim. Thus looking at the implementation of the class `OpenSim::Storage` is suggested when generating `.sto` files using external scripts.

For most of the muscles in the model, the *Locomotor Primitives Controller* applies the activation value directly. For some special cases however (see column "Activation Usage" in table 2.1), the muscle activation is split among a group of muscles, because they were measured in combination. Furthermore, the controller allows to scale the activation for each muscle individually by a constant value.

2.5 Data Preparation

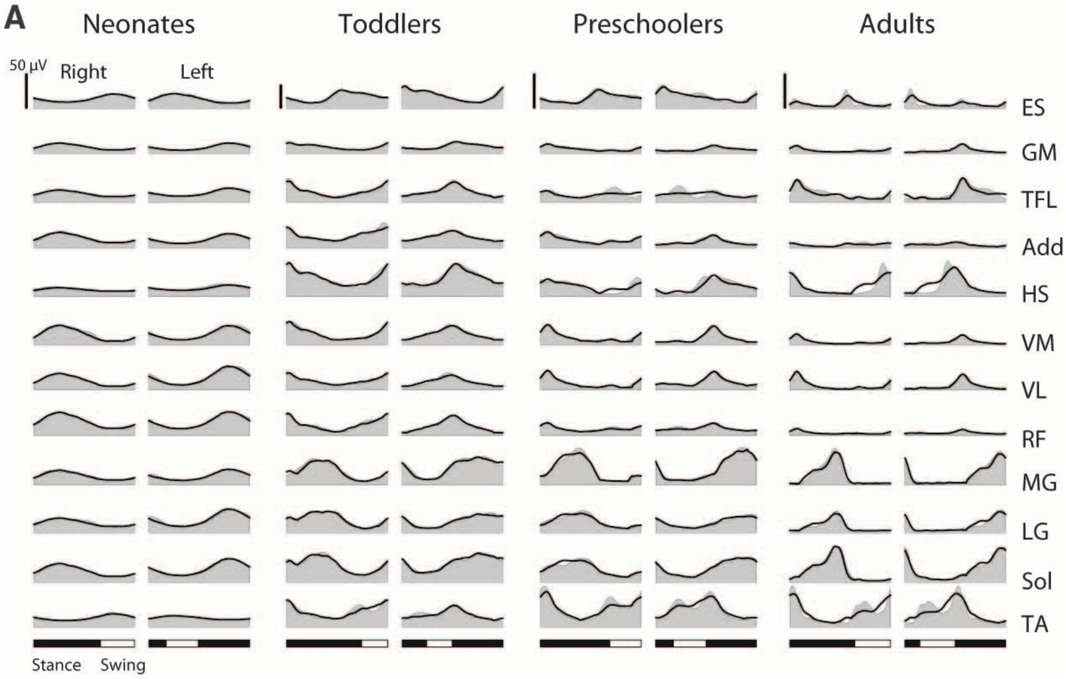

Figure 2.5: Muscle activation pattern curves for the 24 muscles (EMG profiles averaged over all subjects). Illustration extracted from [4]

Since the original data from the publication by Dominici et al. [4] could not be used, the respective pattern curves had to be extracted from the paper (see figure 2.5). This also lead to the fact, that neither data on ground contact forces nor kinematic tracker data was available, since they're not published.

In order to gather the activation data from the publication, a graph digitizer software (GraphClick by Arizona Software) was used. The data was then preprocessed by applying a curve interpolation in order to allow arbitrary, linearly spaced sampling of data points later on in the OpenSim model controller. Curve interpolation was done in a Python script using the *NumPy* and *SciPy* toolboxes.

3 Results

3.1 Reflexes in Simple Leg Model

The simulation of the *Simple Leg Model* led to the expected stretch reflex behavior. Given an initial activation of the *Rectus Femoris*, the *Biceps Femoris* was caused to shorten proportional to the speed of the shortening of the *Rectus Femoris*. Since this will again cause *Rectus Femoris* to be shortened, the reflex is triggered in the other direction. For initial tests, the knee joint in the model was fixed in all but one degree of freedom (pin joint) in order to permit flexion and tension. Unfortunately this didn't lead to a realistic behavior with respect to the movement of the lower leg, as the knee joint remained fixed during the entire simulation (see figure 3.1 for a sequence of two stretch cycles). However since this simulation served as a test of the principal methods and the results were satisfying enough, this problem was not further investigated.

3.2 Random Data in Two-Legged Model

In order to validate the principal working of the two-legged musculoskeletal model (see section 2.2.2) and to test the corresponding controller (see section 2.4.2), random data resembling muscle activation patterns was generated for all the 32 muscles of the model. In the initial version, the data contained discontinuities and the activation values were generally too high. As a consequence, the simulation of the model took very long or in some situations didn't complete at all.

Thus the script used to generate the data was adjusted in order to smoothen the data and provide smaller activation values. With this second set of data, the simulation succeeded, but of course showed unrealistic behavior. Since the simulation now completed with any type of generated pattern, the model and controller were considered ready for usage with the real data.

3.3 Locomotor Primitive Patterns in Two-Legged Model

Only results for an adult subject was obtained using the locomotor primitive patterns (rightmost two columns in figure 2.5). The simulation was run for 1.5, 2.0, 2.5 and 3.0 seconds respectively. The locomotor patterns were scaled to the according timespan. The visualization of a part of the simulation run is shown in figure 3.2.

It can be seen from the sequence that the muscle activations lead to an unrealistic movement of the legs (knees bending backwards, twisting of the feet). Part of the effect can be attributed to the improper scaling of the muscle activations. Thus different scalings factors were tried (0.25, 0.33, 0.5, 0.66 and 0.75) but with all of them the basic improper movement pattern remained.

When looking at the exerted muscle forces for a simulation run with the muscle activations scaled by 0.25 (figure 3.3), the activations are only partially reflected. For the *Biceps Femoris-Long Head* and the *Semimembranosus* (which are both part of the hamstring muscle group, see section 2.2.2) the reported forces match the muscle activation patterns (see figure 2.5) quite closely, while in the case of the *Soleus* and the *Rectus Femoris*, the patterns don't directly reflect the activations. Overall the exerted forces don't show unrealistically high values.

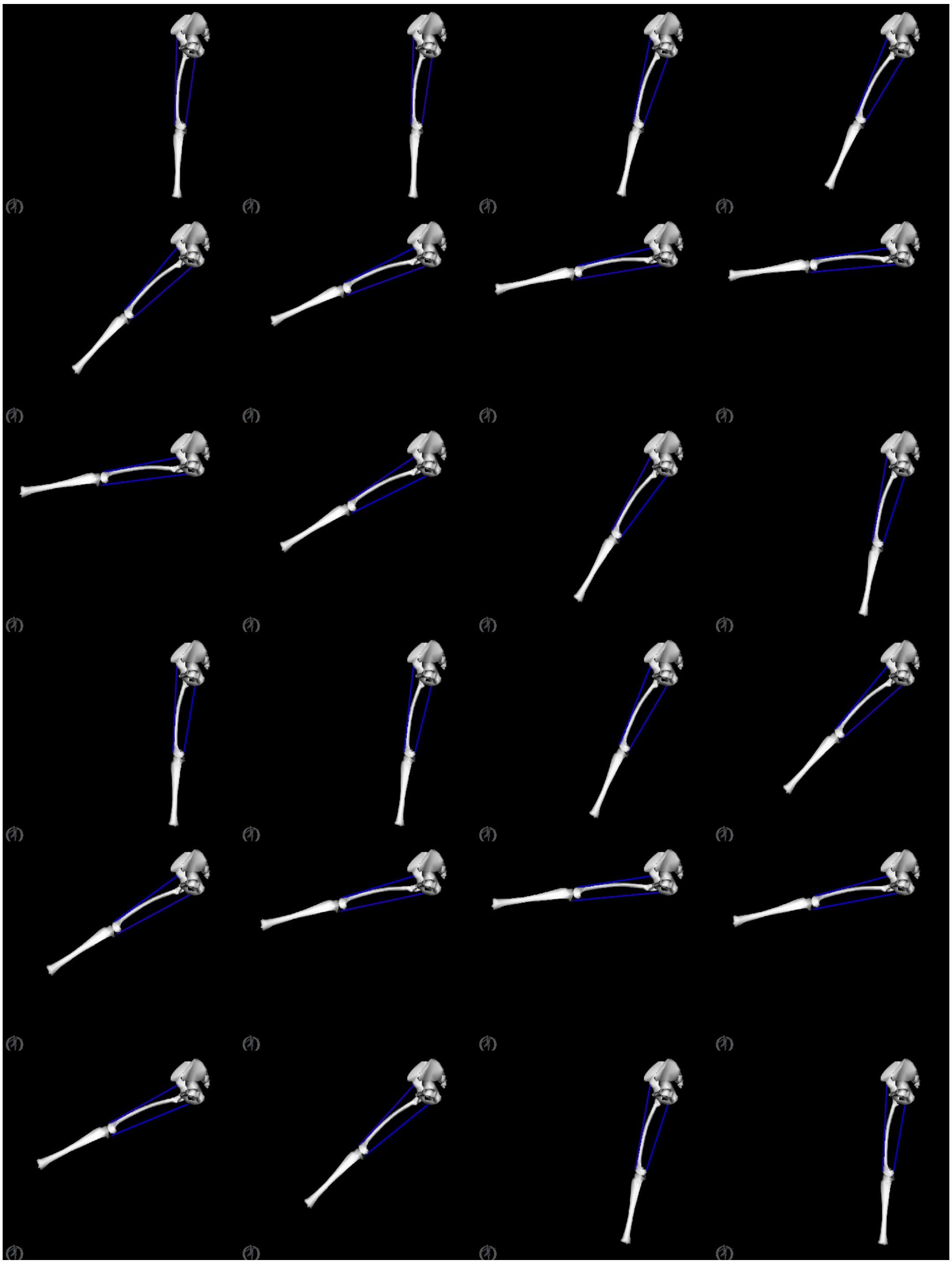

Figure 3.1: Sequence of the *SimpleLeg* model during simulation of two cycles of the stretch reflex.

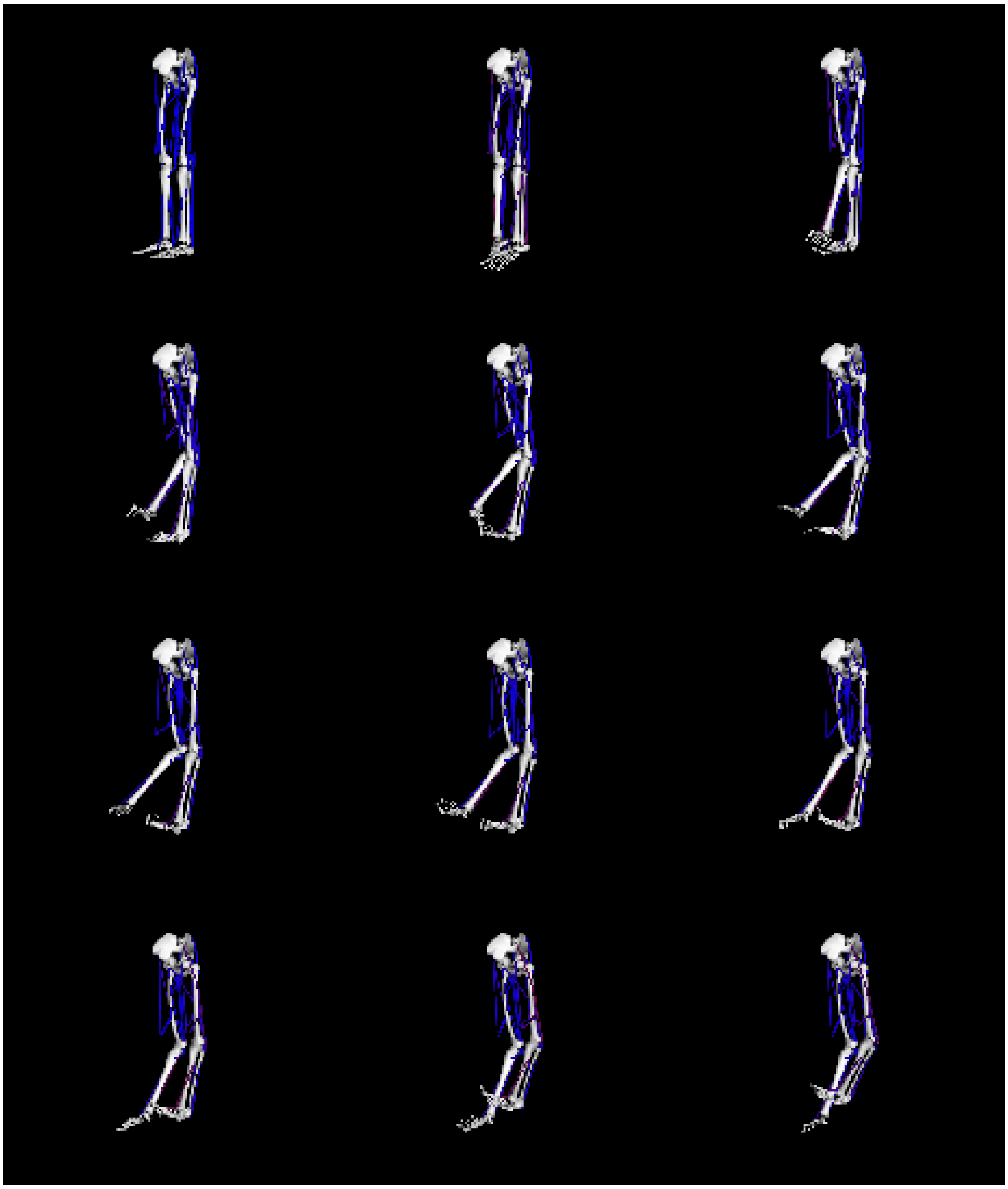

Figure 3.2: Sequence of the two-legged *LocomotorPrimitives* model. Only the first 1.2 seconds are visualized. Total duration of simulation: 3.0 seconds.

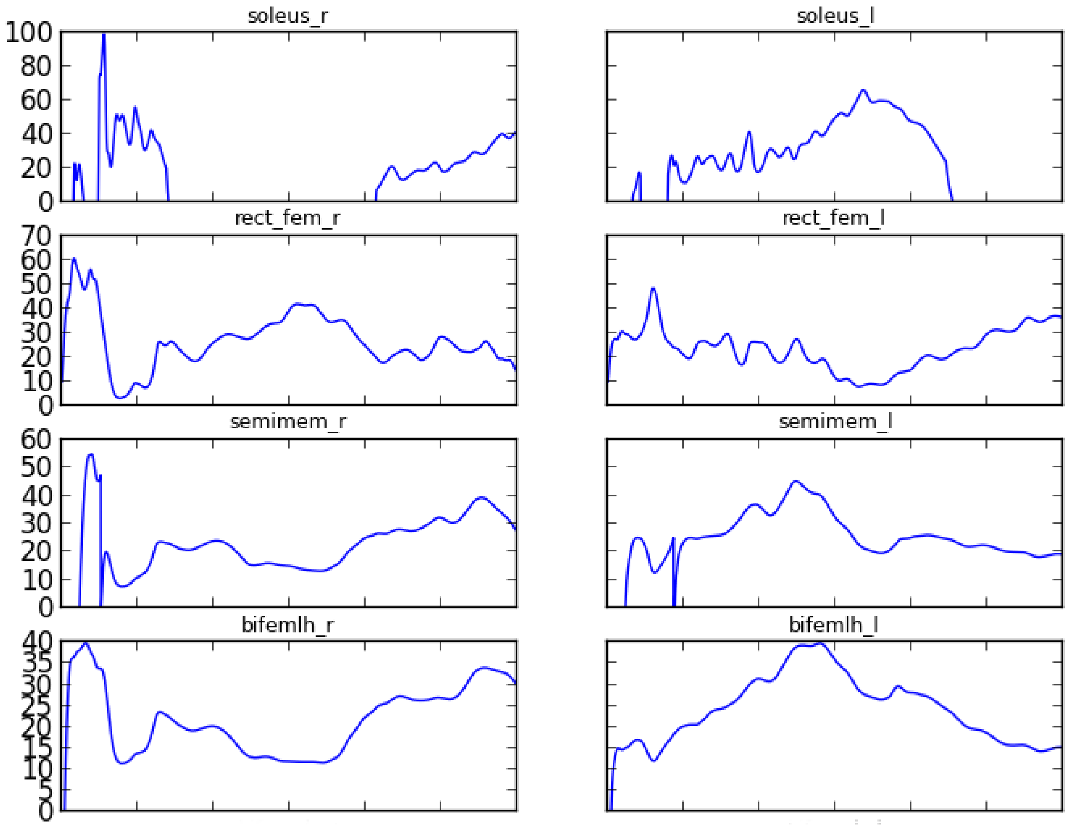

Figure 3.3: Muscle forces exerted on the model during simulation for the muscles *Soleus (soleus)*, *Rectus Femoris (rect_fem)*, *Semimembranosus (semimem)* and *Biceps Femoris-Long Head (bifemlh)* in both legs. X-axis is time from 0.0–3.0 seconds, Y-axis is force in N.

4 Discussion

The results acquired in the few experiments that were conducted didn't match the initial expectations of the project. Rather they showed the difficulties of working with OpenSim given only the locomotor primitive patterns as input data for the simulation. Since several insights into the working of OpenSim, its advantages and disadvantages were gained during the course of this project, mainly an evaluation of the OpenSim software shall be discussed here.

4.1 Simulation using Locomotor Primitives

The unrealistic behavior in the simulation can be attributed to three main reasons: lack of ground contact forces, scaling of the muscle activation and lack of joint constraints. As can be seen in the results for the exerted muscle forces, by scaling the muscle activations accordingly, at least the muscle forces can be limited to an appropriate range. One reason for the "twiddling" with the scaling of the activation is that OpenSim doesn't specify a unit for the muscle activation. Rather it's a normalized value in the range 0.0–1.0, but the exact mapping of measured values to this range is unclear and no documentation could be found. Also the muscle's default activation parameter was found to play a crucial role in the behavior of the simulation. As with the muscle activation, unfortunately no documentation of the proper settings of this parameters could be found with OpenSim.

The lack of data for ground contact forces most probably is the main reason for the problems with respect to the simulation. Ground contact forces were found to be crucial for the control of walking and the patterns of muscle activation involved in walking [11]. By not incorporating them in the simulation which is conducted using muscle activation data that was generated when ground contact forces were present, the experimental setup is not appropriately reproduced and the outcome of the simulation can become unrealistic.

4.2 Evaluation of OpenSim

OpenSim is a great tool to conduct biomechanics experiments – when experimental data is available. Its usage thus mainly is in areas where such data can be directly gathered. Its community provides a wide array of musculoskeletal models and tools for analyzing simulations. However in order to use these appropriately, the availability of experimental data – namely marker trajectories, ground reaction forces and electromyography measurements – is key. The experimental protocols suggested to be used with OpenSim[1] are rather strict e.g. with respect to the measurement of forces. Up to now OpenSim is mainly used in biomechanics or clinical studies (e.g. to compare the effects of different treatments to pathological gaits). In all of these cases subject data were available.

In this project however a lot of abstractions and simplifications (e.g. removing muscles, applying combined activations to certain muscles, neglecting ground contact forces) were made to a model whose purpose is to represent the human legs as anatomically correct and realistic as possible. Furthermore abstracted data was applied to it. This is not the indented usage of these models and the effects are unknown and cannot be predicted when lacking the proper knowledge about its exact working. Furthermore the models used are very sensitive to the type of muscle activation data that was provided (i.e. as could be seen in the random data tests). Unfortunately also the scales and units used for the input data and model parameters (e.g. joint parameters) is not always clearly disclosed.

OpenSim's open-source nature makes it possible to build models and simulations from ground up. The API provided as part of Simbody and OpenSim provide a good base for physics-based simulations of multi-body systems. It can be extended and adjusted if needed. Implementing e.g. custom simulation controllers or custom muscle models can relatively easily be added and just need to implement the according interfaces, as it was done in this project. However documentation on most of OpenSim's internals is rather scarce or only available in the source code.

Building an abstract model from ground up is not easy and the supporting tools provided by OpenSim are very rudimentary and are mostly targeted at changing existing models shipped with OpenSim. However if an abstract model shall be built – given the experiences gained in this project – it should be done bottom-up rather than top-down.

Thus OpenSim with the provided muscle-skeleton models is in its current state not directly feasible for simulations lacking suitable experimental data.

[1] http://simtk-confluence.stanford.edu:8080/display/OpenSim/Collecting+Experimental+Data

5 Conclusion

Musculoskeletal modeling and simulation is an instrumental tool in understanding muscle function involved in biped walking. However creating these simulations at an anatomically realistic level is impossible without proper experimental data. If phenomena shall be studied at a more abstract and fundamental level, the models and the input need to be abstracted accordingly.

The OpenSim software was used to analyze the feasibility of such an abstract-level musculoskeletal simulation using locomotor primitives patterns. However the focus of usage for OpenSim currently is in areas where adequate and precise experimental data is available or can easily be generated.

More abstract – bottom-up built – models such as the ones used by Geyer and Herr [6], Marques et al. [12, 13] or Wang et al. [20] are more suitable for these kinds of analyses. Such an abstracted model could possibly also be built inside OpenSim, however the tools it currently provides to achieve this are limited in their functionality.

Bibliography

[1] Mark F. Bear, Barry Connors, and Michael Paradiso. *Neuroscience: Exploring the Brain.* Third Edition. Philadelphia, PA, USA: Lippincott Williams & Wilkins, 2007.

[2] Nikolai A. Bernstein. *The co-ordination and regulation of movements.* Oxford: Pergamon Press, 1967.

[3] Scott L. Delp et al. "OpenSim: open-source software to create and analyze dynamic simulations of movement". In: *IEEE transactions on bio-medical engineering* 54.11 (Nov. 2007), pp. 1940–50.

[4] Nadia Dominici et al. "Locomotor primitives in newborn babies and their development". In: *Science (New York, N.Y.)* 334.6058 (Nov. 2011), pp. 997–9.

[5] Tamar Flash and Binyamin Hochner. "Motor primitives in vertebrates and invertebrates". In: *Current opinion in neurobiology* 15.6 (Dec. 2005), pp. 660–6.

[6] Hartmut Geyer and Hugh Herr. "A muscle-reflex model that encodes principles of legged mechanics produces human walking dynamics and muscle activities." In: *IEEE transactions on neural systems and rehabilitation engineering* 18.3 (June 2010), pp. 263–73.

[7] Corey B. Hart and Simon F. Giszter. "A neural basis for motor primitives in the spinal cord." In: *The Journal of neuroscience: the official journal of the Society for Neuroscience* 30.4 (Jan. 2010), pp. 1322–36.

[8] A. V. Hill. *First and last experiments in muscle mechanics.* Cambridge: Cambridge University Press, 1970.

[9] A. V. Hill. "The maximum work and mechanical efficiency of human muscles, and their most economical speed". In: *The Journal of physiology* i (1922).

[10] Yuri P. Ivanenko, Richard E. Poppele, and Francesco Lacquaniti. "Motor control programs and walking." In: *The Neuroscientist: a review journal bringing neurobiology, neurology and psychiatry* 12.4 (Aug. 2006), pp. 339–48.

[11] Yuri P. Ivanenko et al. "Control of Foot Trajectory in Human Locomotion: Role of Ground Contact Forces in Simulated Reduced Gravity". In: *Journal of Neurophysiology* (2002), pp. 3070–3089.

[12] Hugo Gravato Marques et al. “Self-organization of reflexive behavior from spontaneous motor activity”. In: *Biological cybernetics* (Sept. 2012).

[13] Hugo Gravato Marques et al. “Self-Organization of Spinal Reflexes Involving Homonymous, Antagonist and Synergistic Interactions”. In: *From Animals to Animats 12: 12th International Conference on Simulation of Adaptive Behavior, SAB 2012, Odense, Denmark, August 27-30, 2012. Proceedings* (2012), pp 269–278.

[14] Rolf Pfeifer and Josh C. Bongard. *How the Body Shapes the Way We Think: A New View of Intelligence (Bradford Books).* Cambridge, MA, USA: MIT Press, 2006.

[15] Rolf Pfeifer and Christian Scheier. *Understanding intelligence.* Cambridge, MA, USA: MIT Press, 1999.

[16] R. Shadmehr and S. P. Wise. “A Simple Muscle Model”. In: *Supplementary documents for “Computational Neurobiology of Reaching and Pointing”.* MIT Press, 2005.

[17] Darryl G. Thelen. “Adjustment of Muscle Mechanics Model Parameters to Simulate Dynamic Contractions in Older Adults”. In: *Journal of Biomechanical Engineering* 125.1 (2003), p. 70.

[18] Lena H. Ting et al. “Review and perspective: neuromechanical considerations for predicting muscle activation patterns for movement”. In: *International Journal for Numerical Methods in Biomedical Engineering* 28.10 (2012), pp. 1003–1014.

[19] Gelsy Torres-Oviedo, Jane M. Macpherson, and Lena H. Ting. “Muscle Synergy Organization Is Robust Across a Variety of Postural Perturbations”. In: *Journal of Neurophysiology* 96.3 (2006), pp. 1530–1546.

[20] Jack M. Wang et al. “Optimizing locomotion controllers using biologically-based actuators and objectives”. In: *ACM Transactions on Graphics* 31.4 (July 2012), pp. 1–11.

[21] Steffen Wittmeier, Cristiano Alessandro, and Michael Jäntsch. “Toward Anthropomimetic Robotics : Development, Simulation, and Control of a Musculoskeletal Torso”. In: *Artificial Life* 193 (2013), pp. 171–193.

[22] Felix E. Zajac. “Muscle and tendon: properties, models, scaling, and application to biomechanics and motor control.” In: *Critical reviews in biomedical engineering* (1989).