Predictive simulation of virtual characters' motion using a muscle model

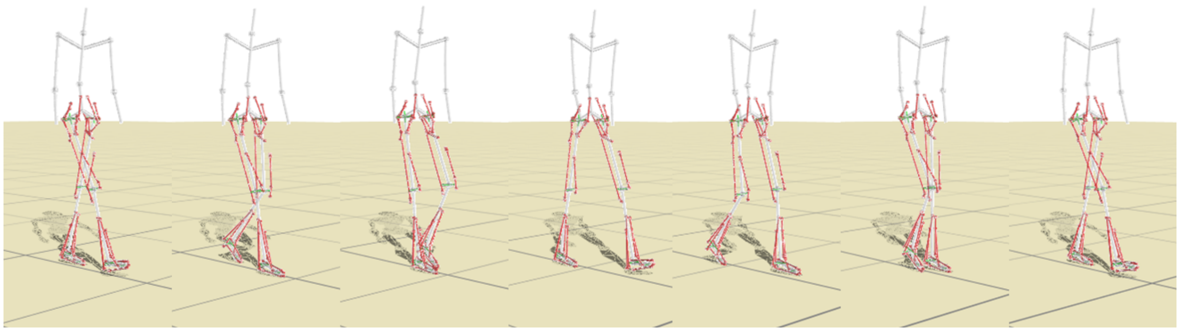
Anna-Katharina Bergmann
Claude Bernard Lyon 1 University (Lyon, France)

March - September 2017

Supervisor: Nicolas Pronost

Coordinator: Raphaëlle Chaine

Location: SAARA team, LIRIS laboratory, Claude Bernard Lyon 1 University



Résumé: Le contrôle physique temps-réel de mouvement de locomotion de personnages virtuels par modèle musculaire est un thème de recherche qui connaît un interêt croissant depuis que les capacités de calcul peuvent supporter la complexité requise. Il permet d'étudier des mouvements physiquement réalistes, interactifs avec l'environnement en temps-réel et ce de manière prédictive, et comporte ainsi des applications multiples, notament dans le diagnostic médical. Ce rapport présente un projet de conversion d'un contrôleur par modèle de moments articulaires en musculaire, réalisé au sein de mon stage de fin de master 2 "Image, Développement et Technologies 3D" à l'Université Claude Bernard Lyon 1, effectué dans l'équipe SAARA du laboratoire LIRIS. Notre modèle de bipède humanoïde 3D est animé en 15 articulations avec un total de 28 degrés de liberté, activés par des unités musclo-tendineuses de Hill. Les démarches sont obtenues en approximant des angles cible des articulations, contenus dans des poses clés, pour lesquels les moments articulaires sont calculés par PD-contrôleur puis convertis en forces musculaires, en respectant les contraintes entrainées par la dynamique de contraction et d'activation musculaire ainsi que des bras de levier. Une optimisation hors ligne permet de calibrer les paramètres inhérents aux muscles et aux contrôleurs en minimisant l'erreur entre pose clé et pose courante. Le contrôleur ainsi obtenu pourra servir de base au projet OMEGA pour réaliser des simulations musculosquelettiques prédictives de démarches pathologiques par optimisation.

Abstract: Muscle-based control of biped gaits in physical animation is a research area that knows a growing interest since the computation capacities are able to support the required complexity. It allows the predictive study of physically realistic motion, interactive with the physical environment in real-time, and hence allows many applications as in medical diagnosis. This report presents a project in which a torques-based controller was converted to muscle-based, within my end of master internship, in "Computer Graphics and 3D Development and Technologies", at Claude Bernard Lyon 1 University in the SAARA team of the LIRIS laboratory. Our 3D biped model is animated in 15 joints with 28 degrees of freedom, actuated by Hill-type muscle-tendon units. The gaits are obtained by approximating target joints angles, within key poses, for which torques are computed by PD-controllers and then converted to muscle forces, under the constraints of contraction dynamics and muscle activation dynamics as well as moment arms. Offline optimization is used to calibrate the parameters inherent to the muscles and the controller, by minimizing the error between target and current poses. The obtained controller may be used as a basis for the OMEGA project for optimization-based forward musculoskeletal simulation of pathological gait.

Keywords: physics-based animation, muscle actuation, musculoskeletal model, biped controller

Contents

1 Introduction

Physics-based animation serves the purpose to simulate motion of articulated characters in a natural looking manner that is entirely interactive with their physical surroundings, responding to external forces in real time. Such simulations require little to no motion data and have a good capacity of adaptation, allowing the reproduction of given gaits as well as the emergence of new ones.

There are two main models in the treatment of motion in physics-based animation, called inverse dynamics and forward dynamics. In inverse dynamics, a given input data of motion and external forces are applied to the model, and the aim is to compute the forces necessary to achieve that motion. On the other hand in forward dynamics, the set of forces (internal to the character and external) acting upon the character are given, and the resulting motions are generated. The input set of forces can be set manually or obtained from recordings ([Cruz Ruiz et al. 2016]) or can be computed by optimization process to match a target task. Such tasks can be defined by few constraints such as maintaining balance, or reaching a target speed. A way of defining a target motion without as much data as in motion capture, is with a finite state machine of a few key poses. A controller then computes target forces, for example a torque for each of the character's joints, and the resulting motion can be observed.

The virtual characters are modeled by a hierarchy of rigid bodies connected by joints. Each joint connects exactly one "parent" body to one "child" body, forming a tree structure (not allowing any loops). The joints are idealized in that they have 1 to 3 degrees of freedom (DOFs) and can produce any arbitrary torque in those directions whenever required, regardless of whether it would be possible for a human being to perform them in reality, ignoring real-life parameters like fatigue, pain, or individual physical condition. This simplification allows the existence of physical simulations that are stable, able to maintain balance, to keep a target heading and speed, track a target trajectory, and follow simple but also more complicated movements like skipping or even somersaults.

However such models lack realism, as the virtual character will act rather like an ideal robot than an actual human with individual medical characteristics. With the aim of creating more visually realistic motion, muscle-based control is preferred over direct control upon joint torques. Muscle-based models have many applications among which medical diagnosis, analysis of athletic performance, rehabilitation therapies, or post-surgery predictions. In animation, muscle-based controllers imply multiple complications as compared to joint-torques controllers, yet they have shown to successfully produce natural looking motion using realistic forces, precise enough for medical use ([Geijtenbeek and Pronost 2012]).

With the aim of an application in bio-mechanical analysis of human gaits that are not ideal, customized and representative of different pathologies, this internship consisted in developing a muscle-based model of a human being and a muscle-based controller producing motion from muscle contraction forces, and evaluating this model and its ability to track a desired motion.

1.1 SAARA team

The SAARA team stands for Simulation, Analysis and Animation for Augmented Reality and is part of the LIRIS laboratory for Graphics Information and Information Systems, located in Lyon, France. It is one of three research teams that belong to the area of Simulation, Virtuality and Computational Sciences and is currently made up of two professors, seven associate professors and six PhD students. Research activities focus on virtual simulations of real-life humans, or animals, in different levels of detail according to the topics, with the constraints of real-time, interactivity and physical realism. Many projects have medical applications, such as facilitating training of medical procedures or simulating dynamic phenomena in human organs. Another main topic is the capture and analysis of human motion recorded by sensors and the interaction with them. Finally, the present project is integrated in the area of interactive, real-time physical animations. It is part of a research project entitled OMEGA: Optimization-based forward musculoskeletal simulation of pathological gait, in collaboration with the Hannover Medical School. The present internship is one of the starting points of the project and will be followed by a three-year PhD.

1.2 Related work

Muscle-based controllers have emerged quite recently in animation due to the large amount of calculations implied, but they have already shown applications to different areas and popularity due their ability to produce realistic motion with respect to specific tasks. [Cruz Ruiz et al. 2016] present a review of the fundamental concepts, the state of research, the published works and future directions for development of muscle-based control. They provide a classification of the existent control methods and their key aspects, as well as commonly used neuromuscular models. In most of the literature, as well as in all the works discussed here, all muscles are represented by Hill-type muscle models as presented by [Zajac 1989]. An overview of this model is given in section 4.1.

In the field of biomechanics, software like OpenSim [Delp et al. Nov. 2007] or AnyBody [*The AnyBody^{TM} Modeling System* 2013], use inverse musculoskeletal simulations to let users develop models that can be precise, apply specific motion on them, and observe the effects on the obtained actuation values. This allows users to analyze motion obtained from patients with different conditions and possible pathologies, as well as specific characteristics of their musculature or other anatomical traits.

In animation, [Geyer and Herr 2010] developed a predictive model of human gait controlled by muscle reflexes using a total of seven muscles on each leg that represent existent muscle groups in the human body, and that control the movements of the hips, knees and ankles in the sagittal plane. Their model has shown to reproduce human walking dynamics, while adapting in real-time to the physical world it is emerged in, with a tolerance to ground disturbances, and adaptation to slopes without the need of reparametrization.

[Wang et al. 2012] extended this model with one extra muscle on each leg, resulting in a biped with 5 uniarticular and 3 biarticular muscles on each leg. The model is a 3-dimensional, physically-animated humanoid. The DOFs of the hips, knees and ankles in the sagittal plane are activated by the contraction of the muscles and all of the remaining DOFs are actuated by direct application of torques computed by the controller. The limitation of the controller to a straight line on a flat ground, with muscle actuation only for specific DOFs, provides it with very good lateral balance and upper-body stability. Whereas this makes the controller respond less naturally to perturbations, it still presents walking and running gaits that don't require motion data and yet show realistic motion, thanks to the constraints on torque generation due to muscle physiology. The parameters of the controllers are computed by offline optimization, requiring the fulfillment of specific locomotion task terms while minimizing effort, by use of a biological model of metabolic energy expenditure.

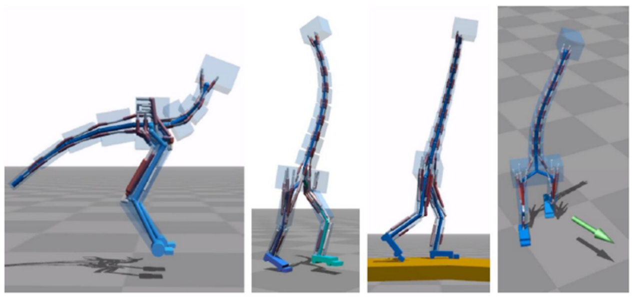


Figure 1: *From left to right: to reach a high steady target speed, the first creature developed a hopping gait, while the second shows a running posture; this creature adapts its gait and neck posture in order to adapt to slopes and heading disturbances [Geijtenbeek et al. 2013]*

[Geijtenbeek et al. 2013] present a muscle-based controller in which not only the parameters of the controller, but also some of the muscles' physiological properties are calculated by offline optimization. This includes rest length and maximum force, which therefore are not required to be known a priori. This results in a great capacity of adaptation to different bipedal models, and allows experiments on imaginary creatures that are absent in nature and for which no data is available. Another important addition is the optimization of muscle routing, which can greatly improve control and be useful in cases of disproportionate humans or creatures where these parameters are unknown. The creatures are formed with the combination of four possible models of upper and lower body. In each model, upper and lower body can be deformed in different manners but that keep the same muscle set. The placement of these muscles varying between the models entails the emergence of different gait styles. Figure 1 shows examples of different creatures and the gaits they developed to reach their target tasks.

A finite state machine consisting of four states (stance, lift-off, swing, and stance preparation) is used to track the model's state and adapt the control accordingly. Based on an input target speed and heading, the controller computes target poses with use of the parameters found by optimization and computes a neural excitation signal for each muscle in order to reach these poses. As will be further detailed in sections 3 and 5, this consists in the consecutive computations of desired joint torques, desired muscle forces, desired muscle activation and excitation. For the arms, a state-dependent constant muscle excitation is used. Additionally, passive spring-damper control with optimized gains is used for spine joints in the axial direction, for the axial rotations of the hips and ankles and the planar rotation of the ankles of the humanoid bipeds, as well as for some creatures' tails.

In order to find the muscle excitation signals that would cause the relevant muscle(s) to move the rigid bodies closer to their targets, the controller uses a muscle-based variation of Jacobian transpose control [???]. This attempts to find muscle torques that emulate the effect of a virtual force or torque applied to a specific body by use of a Proportional-Derivative (PD) controller. The same method is used to convert the muscle forces, which result from the excitation signals and depend on the muscle physiology, to joint torques which are applied to the joints.

All of these methods present models that are partially muscle-controlled. In [Geyer and Herr 2010] and [Wang et al. 2012] the DOFs of the upper body, and of the coronal and axial planes of the lower body are actuated by the application of target torques, and in the case of [Geijtenbeek et al. 2013], passive spring-dampers are used for a few specified DOFs. The controllers rely on key poses and state-machines to compute excitation signals for the muscles and different feedback methods are used subsequently to compute the force each muscle has to produce. These feedback methods include, among others, trunk and head orientation, head stability, or avoidance of hip or ankle hyperextension.

[Lee et al. Nov. 2014] developed a controller for detailed humanoid models based on models available in OpenSim, comprising between 25 and 39 DOFs, and 62 to 120 muscles in total. In opposition to the works mentioned so far, here the forces generated by the muscles are directly transferred onto the bones they are attached to or in contact with. The movements of every DOF of the biped are calculated by solving an equation of motion combining the muscle force as well as the remaining forces and elements of the physical world. Unlike the work of [Wang et al. 2012], feedback gain parameters do not require tuning for every individual target motion, thanks to the optimization of the controller. It is not sensitive to parameters of the model such as the number of muscles or skeletal structures and hence is not dependent on one specific musculoskeletal model. It therefore provides a wide variety of gaits, each of which can be adapted to new body conditions such as muscle weakness, tightness or dislocated joints, as well as new objectives like pain reduction and efficiency maximization. The very detailed models and adaptable reference motions allow the reproduction of a very large variety of gaits, including the subtle nuances of pathologic gait conditions that match real patient data.

1.3 SimBiCon

Physics-based simulations and motion control are complex in animation due to the high quantity of parameters and calculations required. Therefore, for this internship, the framework SimBiCon [Yin et al. 2007] was used as a starting point. SimBiCon is a framework for simple biped control that relies on torque actuation, and allows the generation of a variety of gaits ranging from simple regular forward walking to more elaborate movements like skipping. Such controllers can be defined by as few as two symmetrical key poses, in combination with a few other parameters. Within the time phase of one step, the angles of every degree of freedom of every joint are defined by a spline curve passing through control points that can be modified in real-time. This allows interactive modification of the gaits. The 2009 version, available open-source [Coros et al. 2009b], uses Open Dynamics Engine for the physical dynamics of the rigid bodies, and was used for this internship and converted to muscle-based control.

The SimBiCon controller relies on three main components: a finite state machine of target poses, additional torso and swing hip control and balance feedback. At each simulation step (at a frequency of 2 KHz), feedback torques are computed for all the joints to reach the target pose. Additional torso and swing hip control allow to steady the orientation of the trunk and to adapt the placement of the swing foot according to the current torso pitch angle. Finally, balance feedback provides a further correction of the swing hip angle, by considering the horizontal distance between the stance ankle and the center of mass (COM), and its current velocity. For instance, this grants a better possibility to maintain balance by better foot placement to recover from pushes. These torques are then calculated by PD-controllers before being applied to the model.

For the requirements of this internship, i.e. converting the direct joint torque actuation by muscle actuation, these feedback torques were used to calculate the neural muscle excitation signals by the controller. The resulting muscle forces were then converted to joint torques using Jacobian transpose control, like [Geijtenbeek et al. 2013], hence adding the constraints of muscle physiology to the resulting motion.

The SimBiCon framework has been extended in several other versions in the past, as by [Coros et al. 2009a], where additions were made to add specific tasks to the controllers, like reaching a point in space, and adapting to a target speed or heading. A further extension was made by [Coros and Beaudoin 2010], with the possibility to customize the characters' skeletons in real time, in their sizes and proportions, possibly resulting in asymmetrical bipeds. [Carensac 2015] used SimBiCon to develop a controller for bipeds that are partially immersed in water. His work includes many additions such as stance foot ground contact control, or adaptation to a target speed by step-size variations.

1.4 General Organization and Working Method

The internship lasted over a total of 24 weeks, from March to September with a 3-week break in August. A reunion with the supervisor was held every Wednesday. A first presentation and demonstration was made on April 5th as part of the launching reunion of the OMEGA project in presence of members from the Hannover Medical School, and another smaller presentation was made on May 4th for the supervisor of the OMEGA PhD thesis Saïda Bouakaz.

The framework was developed fully in C++ on Microsoft Visual Studio Community 2017, partly on a dedicated computer at the LIRIS laboratory and partly on my personal laptop, with help of the version control tool of the Lyon 1 University, on a personal repository. A personal daily journal was used to organize and keep track of the different tasks constituting the project.

2 Human Biped Model

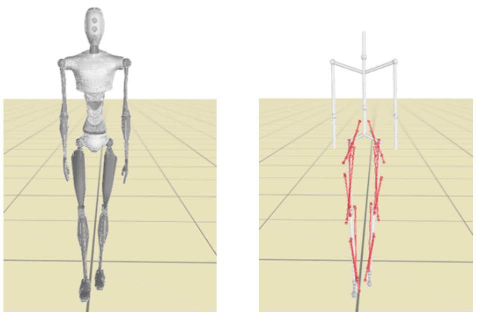


Figure 2: *left: the SimBiCon biped in its original view, with meshes for all the rigid bodies, right: an added view for the muscles, with white spheres for the joints, white segments between the joints and the COMs of the adjacent rigid bodies, and red spheres and segments for the muscles*

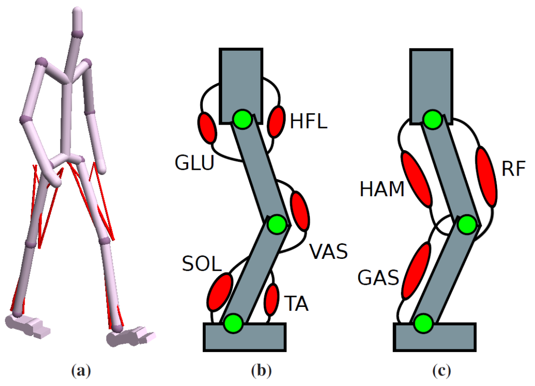


Figure 3: *(a) a humanoid model with 16 Hill-type muscles for the actuation of the lower body, (b) five uniarticular muscles, (c) three biarticular muscles. [Wang et al. 2012]*

The humanoid biped as it is defined in the 2007 version of SimBiCon has a total height of about 175 cm and a total mass of 70.4 Kg. It is represented by a tree structure of rigid bodies connected by joints. The root is the pelvis, the leaves are the hands, toes and head. It possesses 14 joints spread into three categories according to the number of DOFs: ball and socket joints with 3 DOFs (hips, shoulders, neck and pelvis), universal joints with 2 DOFs (ankles) and hinge joints with 1 DOF (elbows, knees and toes). In order to make joints move, muscles can be placed following different models. Figure 2 shows the simbicon biped with the additions of muscles placed according to the model [Wang et al. 2012] shown in figure 3.

This model includes gluteal muscles (GLU) for hip extension, hip flexor muscles (HFL), and the vasti (VAS) for knee extension. The tibialis anterior (TA) and the soleus (SOL) respectively generate dorsiflexion and plantarflexion torques at the ankle. The biarticular muscles include the hamstring (HAM), for hip extension and knee flexion, the rectus femoris (RF), for hip flexion and knee extension, and finally the gastrocnemius (GAS), for knee flexion and ankle plantarflexion.

In the human body, every muscle is attached to bones in two ends called the origin and the insertion. In nature, these ends can form points, lines or areas, however in this model, muscles are represented by thin and weightless action lines, and hence the origin and insertion are simplified as points. Between these points, the trajectory of the muscle can be defined by a sequence of via points, each of them linking it to a bone. The muscle is then formed by the segments between these points, which can span over articulations. The force exerted by the muscle will be evenly applied along these segments. Figure 4 shows an example of a biarticular muscle m^1 on the upper leg.

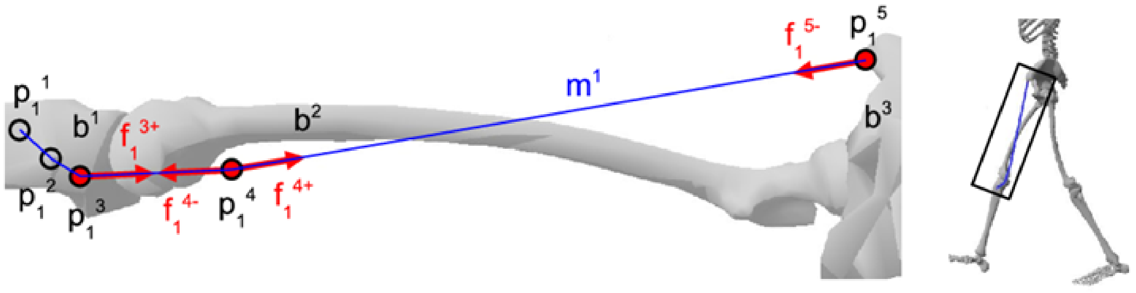


Figure 4: *A biarticular muscle m^1 attached to the tibia bone b^1 in the points p_1^1, p_1^2 and p_1^3, to the femur bone b^2 in the point p_1^4 and to the pelvis b^3 in the point p_1^5. The points p_1^1, p_1^2 and p_1^3 being on the same rigid body, the contraction force produced by the muscle will be locally canceled, but will pull the pairs p_1^3, p_1^4 and p_1^5 closer together, thus making the hip and knee joints turn. [Lee et al. Nov. 2014]*

3 System Overview

The global system used for our muscle-based simulator, of which an overview is given in figure 5, is based on that of [Wang et al. 2012] and [Geijtenbeek et al. 2013]. At each calculation step, the simulator provides information of its current state s, to the controller. This includes several different data such as the angles and angular velocities of all the joints, and the current contact forces. According to this information, the controller computes a neural excitation signal u for every muscle of the model, as will be explained in further detail in section 5. This excitation signal will in turn be converted to an activation signal a, then to an output force F_{MTU}, with the use of the current muscle length l_M and finally to a joint torque τ, which will be applied to the model. These three calculation steps in the musculoskeletal model will be presented in section 4.

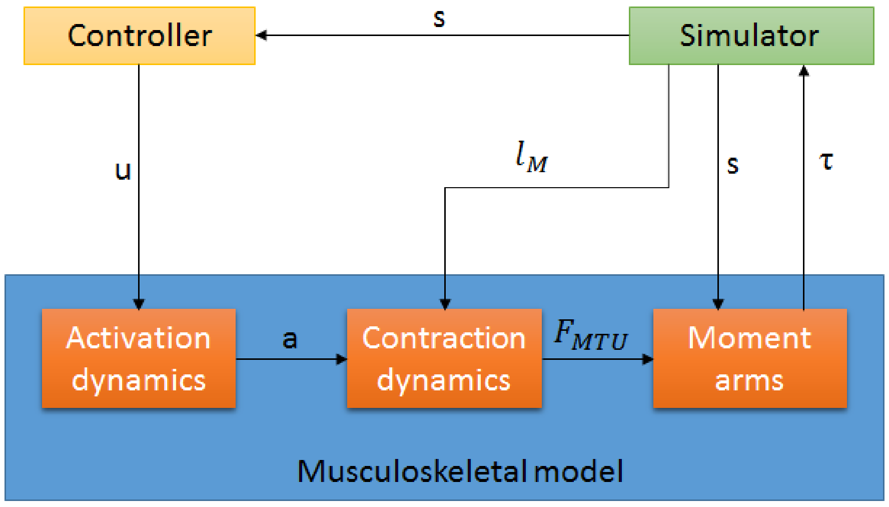


Figure 5: *control system overview. s: simulation state (body position, joint configuration, contact state, etc.), u: neural excitation signal, a: muscle activation, τ: vector torques for all joint DOFs, l_M: muscle length*

4 Musculoskeletal Model

The implementation of this model and the different calculation steps for the torque generation to replace the direct computation of the torques as was done in SimBiCon originally, was one of the main parts of this internship, and setting it up within the SimBiCon biped model was one of the earliest goals. We defined a muscle with the following information: a name, the rigid bodies and local coordinates defining the via points, and constant terms that will be explained in this section, namely the maximum isometric force, the optimal fiber length, the tendon slack length and the neural delay. These were all added to the existing files used in SimBiCon to load the biped model, with the information of the rigid bodies and joints. All this information is loaded at the beginning of the simulation from a specified ".rbs" file, whereas the information to define the controller (see section 5) is loaded from an ".sbc" file and a ".rs" file for the initial reduced state.

4.1 Contraction Dynamics

A muscle takes an input activation signal of values between 0 and 1 and outputs a force measured in Newton. A muscle is composed of fibers and tendons attaching it to bone and produces force in a more complex manner than can be represented by a simple formula. The model we use for the computation of this force is Hill's model for muscle-tendon units (MTUs) as presented by [Zajac 1989]. This is the model used in all of the works discussed in section 1.2, and is generally very popular in animation and biomechanics. A representation of this model is given in figure 6. It consists of a serial spring element SE that represents the tendon, and of two elements representing the muscle fibers: a contractile element CE and a passive parallel element PE. The two latter elements produce forces F_{CE} and F_{PE} respectively. The total force F_{MTU} produced by the MTU is given by the following equation:

$$F_{MTU} = F_{max}(F_{CE} + F_{PE})cos(\alpha) \tag{1}$$

where F_{max}, the maximum isometric force, is a constant representing the maximal force that can be produced by the muscle when the joint angle and muscle length do not change during contraction. α represents the angle formed between the fibers and the tendon, called the pennation angle. When a muscle contracts, the fibers and tendons shorten and the pennation angle increases. The optimal pennation angle is defined as the orientation at which the produced force reaches F_{max}. According to [Zajac 1989], this optimal pennation angle is very small in most human muscles, and is expected to barely affect the static and dynamical properties of the MTU. For this reason, and to simplify the model, we therefore assume here that this angle is always zero, as did [Geijtenbeek et al. 2013].

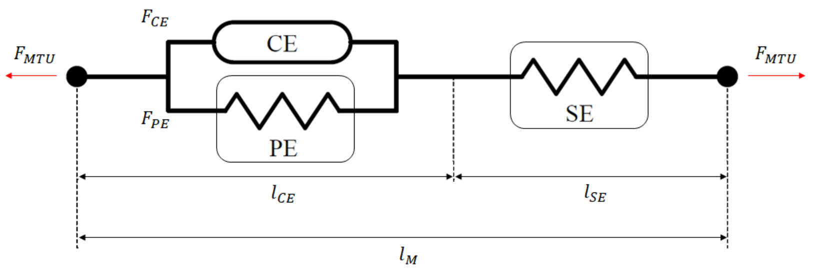


Figure 6: *Hill-type model for contraction dynamics of muscle tissue [Zajac 1989]. Total muscle force F_{MTU} is the sum of passive force F_{PE} and active force F_{CE}, created by the passive element PE and the contractile element CE respectively. F_{CE} depends on muscle fiber length l_{CE} and velocity v_{CE}, and the activation state of the muscle fibers a(t). The element representing the tendon is called the muscle series elastic element SE, or sometimes serial spring element (SSE or SEE). The total muscle length l_M is the sum of the fiber length l_{CE} and the tendon length l_{SE}.*

The contractile element represents the active forces created by the contractile proteins in the muscle. Its output force is a function of the activation signal and the current length and contraction velocity of the muscle:

$$F_{CE} = a f_{al}(\tilde{l}_{CE}) f_v(\tilde{v}_{CE}) \tag{2}$$

where $\tilde{l}_{CE}$ and $\tilde{v}_{CE}$ are respectively the fiber length l_{CE} and the fiber contraction velocity v_{CE}, normalized by the optimal fiber length l_{CE}^{opt}, which is defined as the fiber length at which the maximum isometric force is reached. The passive element represents the passive force that results from the elongation of the connective tissue components in the MTU. Its output force is calculated by:

$$F_{PE} = f_{pl}(\tilde{l}_{CE}) \tag{3}$$

The functions f_{al}, f_v and f_{pl} are respectively referred to as the active force-length, active force-velocity, and passive force-length relationships. Their formulas can vary slightly between studies but their general shapes all follow those presented by [Zajac 1989]. Here, we chose the formulas used by [Lee et al. Nov. 2014] given below and shown on figure 7.

$$f_{al}(\tilde{l}_m) = e^{-(\tilde{l}_m - 1)^2/\gamma} \tag{4}$$

$$f_v(\tilde{v}_m) = \begin{cases} \frac{\tilde{v}_m + \tilde{v}_m^{max}}{\tilde{v}_m^{max} - (\tilde{v}_m / A_f)} & \text{if} \quad \tilde{v}_m \leq 0 \\ \frac{f_m^{len} \tilde{v}_m (2 + (2/A_f)) + \tilde{v}_m^{max} (\tilde{f}_m^{len} - 1)}{\tilde{v}_m (2 + (2/A_f)) + \tilde{v}_m^{max} (\tilde{f}_m^{len} - 1)} & \text{if} \quad \tilde{v}_m > 0 \end{cases} \tag{5}$$

$$f_{pl}(\tilde{l}_m) = \begin{cases} \frac{e^{(k_{pe}(\tilde{l}_m - 1)/\epsilon_m)} - 1}{e^{k_{pe}} - 1} & \text{if} \quad \tilde{l}_m > 1 \\ 0 & \text{if} \quad \tilde{l}_m \leq 1 \end{cases} \tag{6}$$

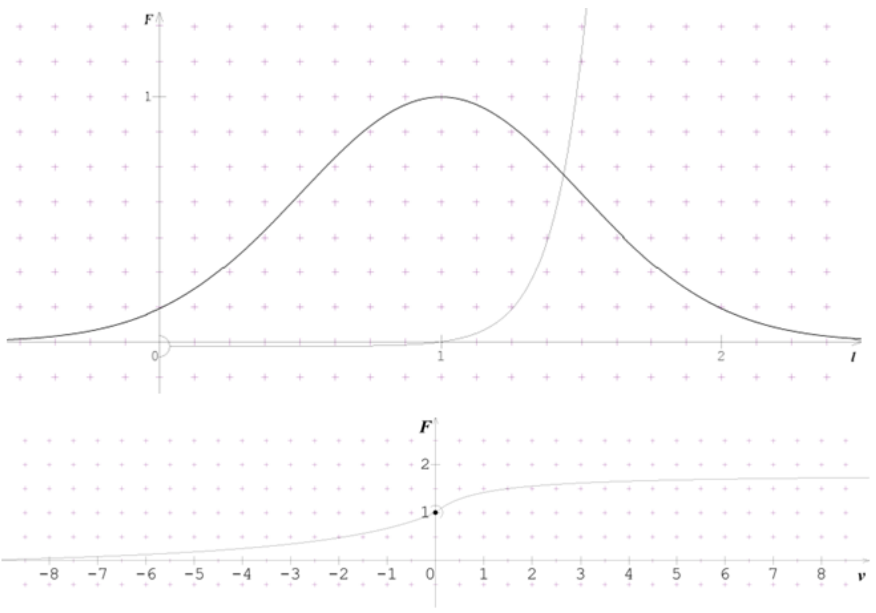


Figure 7: *top: active force-length relationship (black line) and passive force-length relationship (grey line). bottom: active force-velocity relationship. On the x-axis, l and v are normalized by l_{CE}^{opt}, on the y-axis, F is normalized by F_{max}*

These formulas contain a number of constant parameters: γ, k_{pe} and A_f are shape factors for the three different curves that affect their width and steepness, $\tilde{v}_m^{max}$ is the normalized maximum contraction velocity of the muscle fiber, measured in optimal fiber lengths per second, f_m^{len} is the normalized maximum lengthening muscle force, and ϵ_m is the passive muscle strain. The values for all these parameters have been taken from [Lee et al. Nov. 2014]. The according curves obtained vary in shape depending on individual characteristics of the subject (such as age or fitness). The active force-length relationship slightly simplifies that of [Zajac 1989], in which these curves are not symmetrical (the positive slope is steeper than the negative).

F_{max} is a muscle-specific constant, for which starting values were taken from a humanoid model from OpenSim, and then optimized (see section 6). This is also the case for the optimal fiber length, and the tendon slack length l_{SE}^{slack}, which is defined as the length below which the tendon is slack and doesn't produce any force. These two parameters allow the computation of the current fiber length, given the total muscle length in the simulation which is calculated by summing the lengths of the segments between the via points.

4.1.1 Tendon compliance

To calculate the current fiber length and hence the contraction velocity from the total muscle length, we looked at possible methods that differ in their model of the tendon compliance, in other words the tendon's ability to stretch and transmit force. One common simple model is to consider the tendon as infinitely stiff, which means its length is permanently set to its constant slack length and only the fibers can extend. Another approach is to do the opposite and to consider the tendon as infinitely compliant, in which case it is the fibers that remain constant at their optimal length and the tendon that stretches, without transmitting any force. We used a calculation based on that of [Thelen et al. 2003] in OpenSim that uses both of these models alternatively. For this, a first approximation of the tendon length is calculated

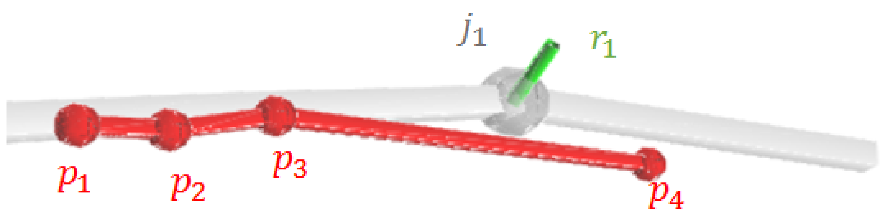


Figure 8: *Example of a muscle path with four via points p_1 to p_4, overlapping one joint j_1 with moment arm r_1*

as the difference between the total muscle length and the optimal fiber length. When this is shorter that the tendon slack length, the tendon is considered stiff, otherwise it is considered compliant. A compliant tendon implies that the fibers remain at their optimal length and hence the output force is always the maximum isometric force (see Figure 7). On the other hand, a stiff tendon can lead to big and unrealistic variations in the fiber length and very low output forces. Therefore, the altering tendon compliance model as used by Thelen offers a good compromise.

4.2 Moment Arms

The next step of the musculoskeletal model is to convert the forces applied along a muscle to joint torques. When a muscle m contains two consecutive via points p_k and p_{k+1} that are on different rigid bodies and hence overlap a joint c centered on point j_c (and eventually other joints as well), the segment $s_k = p_{k+1} - p_k$ is used to calculate the moment arm r_c:

$$r_c = (p_k - j_c) \times \frac{s_k}{||s_k||} \tag{7}$$

The moment arm gives the direction of the torque that will be applied to the joint when the force F_{MTU} is generated by the muscle. An example is given on figure 8. This vector is recalculated at each time step, as both its orientation and magnitude vary according to the configuration of the points j_c, p_k and p_{k+1}. The generated torque is then calculated by:

$$\tau_c = r_c F_{MTU} \tag{8}$$

In our model, the joints only allow rotations around their specified DOFs and the placement of the muscles does not guarantee or imply that the moment arms will be aligned with the DOFs. Therefore, the joint torque τ_c around r_c will be projected onto the joints DOFs, which can result in some force losses for joints will less than three DOFs. The better the moment arm and the DOF are aligned, the more force will be conserved, and the greater the angle between them, the more force will be lost, resulting in a total loss of force for a perpendicular configuration (see section 7.2).

To be able to perform movements in all the joint's DOFs, there needs to be at least one muscle and its antagonist for a movement in each direction. Therefore each joint is overlapped by at least two different muscles (the set of muscles overlapping the joint c is defined as M_c), each of them generating a torque $\tau_{c,m}$. The total torque is obtained by summing all of them:

$$\tau_c = \sum_{m \in M_c} \tau_{c,m} \tag{9}$$

The effect of having several muscles actuating the same joints is further discussed in section 7.4.

4.3 Activation Dynamics

At each time step, the controller outputs a neural excitation signal u for each muscle (as can be seen on figure 5), which is converted to a muscle activation signal a. This conversion does not happen instantly as it is deferred from the brain over the spine to the muscles. This process is called activation dynamics and is modeled by the following equation:

$$a_{t+1} = a_t + c_a h(u_t - a_t) \tag{10}$$

where t is the current time step, $h = 1/2000s$ the constant step size, and c_a is the constant activation and deactivation rate. The latter was set at 100Hz following [Geijtenbeek et al. 2013] as a simplification, as according to [Zajac 1989], the rate constant for activation is greater than for deactivation and activation has saturation.

5 Control

The controller in the initial SimBiCon framework computes feedback torques for each joint, in order to reduce the error between the current and desired poses. For this, it uses a PD-controller, reducing the difference between the current and target orientation of the joint's child body, and its angular velocity:

$$\tilde{\tau} = k_p \mathrm{d}(\tilde{Q}_{rel}, Q_{rel}) + k_d(\tilde{\omega}_{rel} - \omega_{rel}) \tag{11}$$

k_p and k_d are the PD-gains that are constant for each joint and subject to offline optimization, Q_{rel} and $\tilde{Q}_{rel}$ are quaternions for the current and desired orientations of the child body expressed in the frame relative to the parent body and $\tilde{\omega}_{rel}$ and ω_{rel} are the desired and current angular velocities also expressed in the relative frame. d is a distance function, defined using the exponential map that represents the rotation from the current to the target orientation. For this project, the aim was to add a process to this controller to convert these feedback torques $\tilde{\tau}$ to neural excitation signals.

5.1 Excitation Dynamics

Similarly to [Geijtenbeek et al. 2013], we compute the desired muscle activation a_m for the muscle m, step by step from the desired (vector) torque $\tilde{\tau}_k$ for the joint k by calculating the desired scalar torque $\tilde{T}_k$ for the direction of the moment arm r_k, then the total desired force $\tilde{F}_m$ for the muscle m by average of the desired torques for the n joints it overlaps, and finally get the desired activation $\tilde{a}_m$ with the maximum isometric force of the muscle m, F^m_{max}. This desired activation is not directly used as the neural excitation but delayed in time by Δ_t, a quantity which roughly depends on the distance between the muscle and the brain. Like [Geijtenbeek et al. 2013], we use Δ_t = 20ms for muscles attached to the foot, Δ_t = 10ms for muscles attached to lower leg, and Δ_t = 5ms for all the remaining muscles. The following represent the ordered sequence of these four calculation steps:

1. $\tilde{T}_k = \frac{r_k}{||r_k||} \cdot \tilde{\tau}_k$
2. $\tilde{F}_m = \frac{1}{n} \sum\limits_{k=1}^{n} \frac{\tilde{T}_k}{||r_k||}$
3. $\tilde{a}_m = \frac{\tilde{F}_m}{F^m_{max}}$
4. $u_m = \tilde{a}_{m,(t-\Delta_t)}$

These calculations were added to a new, "muscular" controller after completion of the musculoskeletal model. The time-delay Δ_t, constant for each muscle, was added to the model files, and the delay itself was made with the use of a buffer. In the musculoskeletal model, these calculations are followed consecutively by equations (10), (2), (8) and (9) to calculate the final vector torque that will be projected onto the DOFs and applied to the biped. The pseudo-code below gives the sequence of calculations performed at each simulation step. The first instruction is the feedback torques computation from SimBiCon, which computes the pose, the torque for each joint using the PD-controller (see equation 11), and overrides the hip torques.

```
// compute the feedback torques between the current and desired poses
$feedbackTorques \leftarrow$ computeTorques()

// compute the muscle activation for each muscle accordingly
// and the resulting torques
for each $muscle$ in $character\_muscles$ do
    // steps 1 through 4 above
    computeDesiredTorque()
    computeDesiredForce()
    computeDesiredActivation()
    computeExcitation()
    // equation 10
    updateActivation()
    // update muscle's segments, length, contraction velocity, moment arms
    updateFields()
    // equation 2
    computeForce()
    // equation 8
    $muscle.torque \leftarrow$ convertForceToTorque()
end for

// let each muscle add its torque to all joints it covers (equation 9)
for each $muscle$ in $character\_muscles$ do
    for each $joint$ in $muscle.coveredJoints$ do
        $muscleTorques[joint] \leftarrow muscleTorques[joint] + muscle.torque$
    end for
end for
```

6 Optimization

Both the controller and the model have a large number of constant parameters that have a great influence on the resulting dynamics and that can vary quite a lot between different simulations (for different gaits and bipeds for example). A number of these are set by preliminary optimization. [Wang et al. 2012] optimize a total of 30 controller parameters for the stance phase and 26 for the swing phase. For the lower body these are the parameters of the PD-controller and the target angles. To this are added 33 parameters for the initial state. Similarly, [Geijtenbeek et al. 2013] optimize parameters for state transition, target features, feedback control and the initial character state. For the lower body, muscle properties (F_{max}, l_{CE}^{opt} and l_{SE}^{slack}) are also subject to optimization, and for all the muscles the position of their attachment points can be changed within predefined lines, areas, or volumes. We chose to optimize the muscle properties and PD-gains for different subsets of the body, which in total contains 56 uniarticular muscles (see section 7.5), resulting in up to 100 parameters. To further optimize the controller for a fixed biped model, we also planned to optimize the control points that define the spline curves of the target angles of every target pose.

Whereas both [Wang et al. 2012] and [Geijtenbeek et al. 2013] use Covariance Matrix Adaptation for their optimization, due to time constraints, we chose a simple random process within a defined interval and with a given precision. The simulation frequency is set to 2KHz and the simulations were run to evaluate an objective function over a time interval of 10 seconds, which is then repeated for up to 5000 iterations.

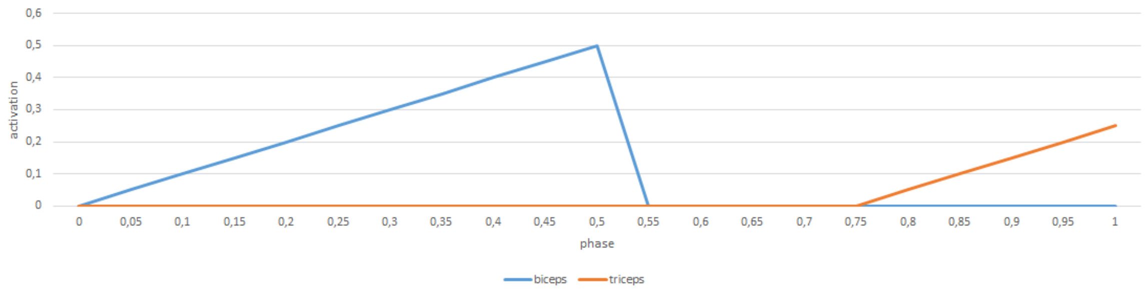
The aim of the optimization is to minimize the objective function which is defined as $\sum_{j \in joints} ||\tilde{\tau}_j||$ in order to avoid both configurations where PD-gains are too low and hence the forces are too weak to follow the target poses, and those where PD-gains are too high and the forces are too high and overshoot the target pose, which can result in shaking and loss of stability. Other possible objective functions include that of [Wang et al. 2012] which uses the total rate, over all MTUs, of metabolic expenditure, or in other words the work done and the heat created by the muscles. [Cruz Ruiz et al. 2016] give an overview of other objective functions that can minimize overall effort or fatigue.

Similarly to [Geijtenbeek et al. 2013] we consider that a run has failed and can be aborted without evaluation of the objective function, if before the end of the run, the COM of the biped has dropped below a threshold (0.9 times the initial COM height) or if the heading has deviated from the target heading (over 45 degrees). Further possible criteria for failure include self-collision and leg-crossing.

7 Experiments

The first milestone of this internship was, after the first month, the implementation of the musculoskeletal model, which allowed us to observe muscle actuated motion, calculated from a raw input of activation signals. We started by testing this on the elbow joints because these have few influences and dependencies on the overall balance of the biped and on adjacent joints. A set of six muscles was placed on the arms according to a model available on OpenSim, 3 for the biceps and 3 for the triceps. The application of a linear activation function, with a different offset for the biceps and triceps, resulted in a stable regular folding and unfolding of the right elbow (see figure 9).

The next phase was the implementation of the controller in charge of converting the desired torques (calculated by the existing controller) to excitation signals. Here, we used a simplified model to connect this controller to the musculoskeletal model in order to complete the system loop shown in figure 5. This simplification served as a starting point to investigate the motion generated by different muscles on the arms and legs, considering the different steps of the loop one by one and their implications. Finally, a set of muscles was placed on the entire body, and the full system, both controller and model, were subjected to offline optimization. The SimBiCon framework provides a few gaits that are torques-actuated, some walking and some jogging. These were used as starting points for the muscle-actuated gaits and served as aims throughout the experiments.



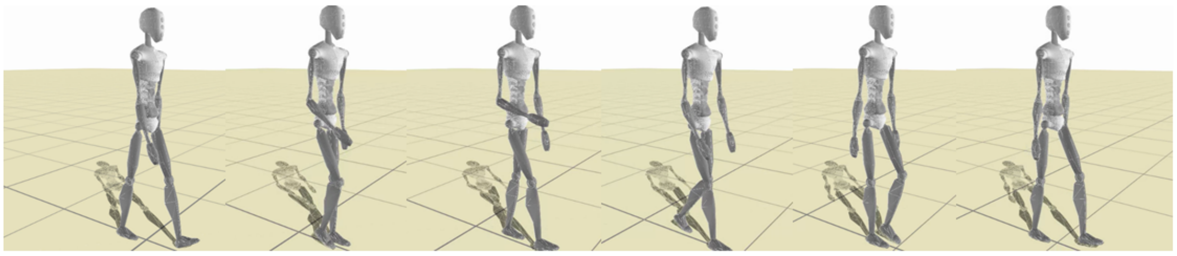


Figure 9: *Example of a raw input activation for three biceps muscles (in blue) and three triceps muscles (in orange), over the time phase of one step, with the resulting motion of the right arm.*

7.1 Simplified Model

For a simplified starting point of the complete control system, we used a model where only the elbow joints are muscle actuated. We use only one pair of antagonistic muscles (a biceps and a triceps), placed so that the directions of the moment arms exactly match the directions of the DOFs. Finally, we simplify the system loop by directly applying the desired forces to the muscles, temporarily ignoring activation and contraction dynamics. We hence consider the following steps for a joint j, covered by two muscles in the set $M_j \ni m$, for which the controller outputs a desired torque $\tilde{\tau}_j$:

1. $\tilde{T}_{j,m} = \frac{r_{j,m}}{||r_{j,m}||} \cdot \tilde{\tau}_j$
2. $\tilde{F}_{j,m} = \frac{\tilde{T}_{j,m}}{||r_{j,m}||}$
3. $F_{j,m} = \tilde{F}_{j,m}$
4. $\tau_{j,m} = F_{j,m} r_{j,m}$
5. $\tau_j = \sum_{m \in M_j} \tau_{j,m}$

Due to the simplifications of the model, $\tilde{\tau}_j$ and τ_j are expected to be exactly equal, and hence the motion of the elbows is expected to be identical to the reference torques actuated motion.

In the cases of joints that are able to bend to very narrow angles, as is the case with the elbows and the knees, it needs to be ensured that the muscles are placed correctly at all possible angle configurations. For example if the elbow is bent to its maximum and the triceps' via points are in the middle of the upper and lower arm, the muscle segment will be on the wrong side of the bones and its moment arm's direction will coincide with that of the

biceps, hence making it impossible to unfold the arm by contracting the triceps.
In OpenSim, conditional via points, called "wrapping points", are added only when the joint is bent above a certain threshold, in order to make the muscle wrap around the bone instead of going through it. The additions of these points has the disadvantage of causing discontinuities of the moment arm dynamics, as the muscle segment that overlaps the joint will change. Here, no wrapping points were added but the muscles were placed such that even at maximum bend, no muscle segment intersects or traverses the bones, avoiding these situations.

This simplified model provided the expected results, namely a motion identical to the torques actuated reference, which we verified quantitatively by comparing the respective values of the torques and the joint angles, which were identical. On this basis, the missing steps of the calculations were then added individually. These steps are detailed in the three following sections. We studied their impacts, one by one in an isolated manner at first, on the muscle-actuated motion of the arms, before putting them all together to obtain a full model.

7.2 Moment Arm Orientation

As mentioned earlier, the torques calculated for the directions of the moment arms are then projected onto the DOFs within the physics engine. This projection takes the form of a scalar product and can lead to force losses. If the moment arm orientation matches that of the DOF, all force is preserved and the motion matches that obtained with direct torques applications. However, the more the angle between the two vectors increases towards a right angle, the more force is lost, therefore the smaller are the torques converted from the muscle force, and the weaker are the movements.

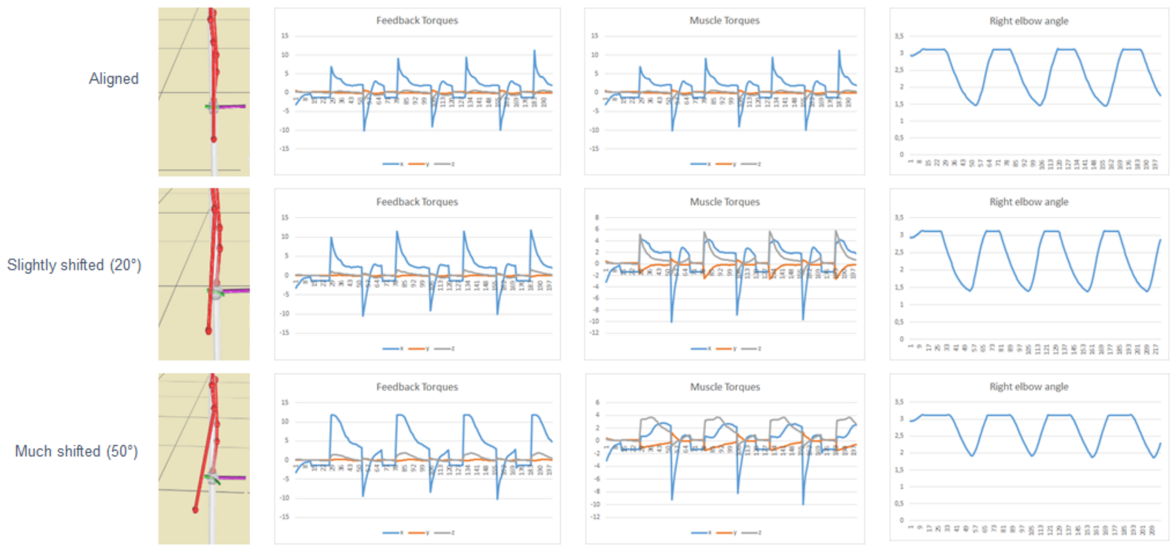


Figure 10: *Comparison of three different configurations of the biceps moment arm and the elbow's x-axis DOF. From left to right: the placement of the muscle (in red) with the DOF in purple and the moment arm in green, the Feedback Torques computed by the controller, the torques generated by the muscles and the angles in radians reached by the elbow. The recording frequency is 50 Hz.*

Some of these results are shown on figure 10. When the shift is greater, force is lost and the torques are smaller, therefore the controller computes higher Feedback Torques to reduce the larger error between the current and target poses. However, even with higher Feedback Torques, the forces that the muscles are able to generate are not high enough to compensate the loss and therefore produce smaller torques. Hence the angle reached by the elbow decreases, meaning that the arm bends less and resembles the target pose less.

In the case of the model of [Wang et al. 2012], the muscles are placed in a manner that the moment arms are mostly not aligned well with the joints' DOFs. Furthermore, these muscles are intended to actuate only some of the DOFs of the joints of the lower body, the remaining being torque-actuated in the case of [Wang et al. 2012], and actuated by passive springs in that of [Geijtenbeek et al. 2013]. Therefore here, muscles were added and via points were moved for a better alignment of the moment arms and DOFs, to have a chance of obtaining results comparable to those obtained by direct application of the desired torques.

7.3 Biarticular Muscles

A biarticular muscle covers two joints and hence outputs force calculated from the average of the two different forces needed to produce the desired torques on each of them. To observe this in a simple configuration, we added joints for the wrists in order to have two consecutive hinge joints with few constraints. We placed a single elongated muscle for elbow and wrist flexion and another for elbow and wrist extension, and observed how the forces are distributed over both joints.

The results differ depending on the synchronization of the target angles of both joints. Good synchronization for these muscles requires that the elbow and wrists flex at the same time and extend at the same time. When the target motion is shifted or opposed (e.g. the wrist must flex while the elbow must extend), a muscle would theoretically have to contract to produce a positive force for one joint and extend to produce a negative force for the other. An example is given on figure 11.

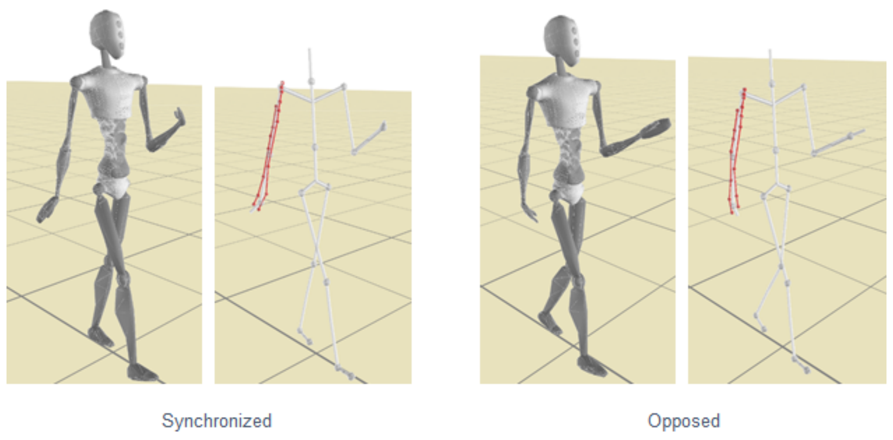


Figure 11: *Examples of synchronized and opposed wrist and elbow flexion*

To compute the force the muscle outputs in such cases, in the literature, some take only the positive forces for the calculation of the average, some leave the negative values in the calculation, and others threshold them to zero. These three calculations change the way in which the average force is reduced in the case of non-synchronized motion. We tried all three possibilities and observed the resulting motion in the elbow and wrist. Figure 12 shows a result of the method of leaving the negative forces in the calculation, that compares the angles of the right wrist and elbow with joint and with muscle actuation, on three different synchronizations of both joints. Considering the complications implied, and the need of additional parametrization of the controller for a good synchronization of the motion, we decided to start by adding only uniarticular muscles to the lower body.

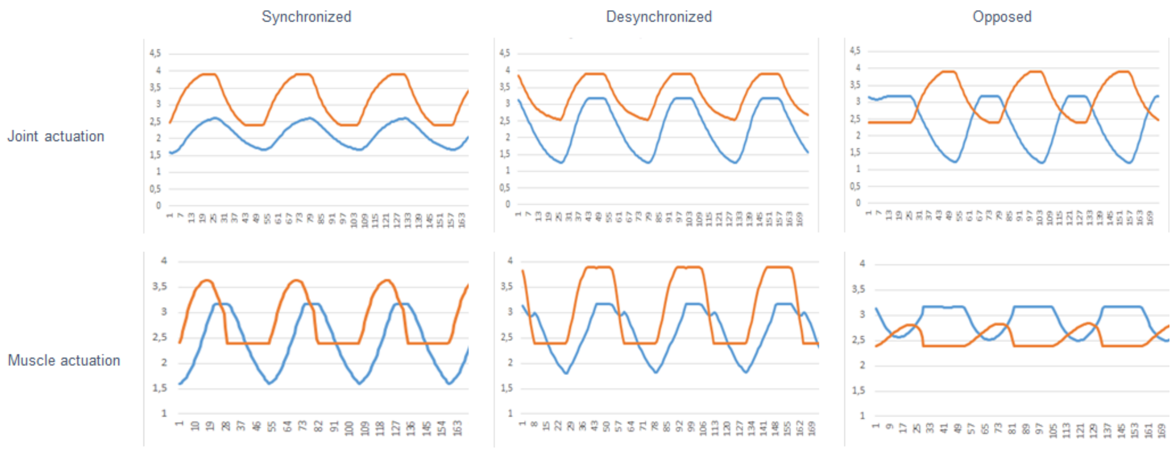


Figure 12: *Angles of the right elbow in blue and wrist in orange given in radians over time with a frequency of 50Hz. Comparison between the results from joint actuation and muscle actuation, on three different gaits: The synchronized and opposed motion shown on figure 11 and an intermediate desynchronized motion.*

7.4 Muscle Groups

In the models used by [Wang et al. 2012] or [Geijtenbeek et al. 2013], often more than one muscle cover the same joint DOF (e.g. both the HAM and GAS muscles allow knee flexion, see figure 3). In those cases, the torques from each muscles are summed together, therefore the resulting joint torques are greater. In the reference gaits we use for comparison, with full torque actuation, the target poses are hardly ever reached but just aimed at. A similar motion is obtained when only one muscle and its antagonist actuate one DOF; when there are more, the movements are stronger, the elbows bend more and get closer to the target pose, but further from the desired comparative gait.

7.5 Full model

In view of the factors discussed in the previous sections, we worked on a model with the complete control loop and with full muscle-actuation. For this, we placed one uniarticular muscle and its antagonist for every DOF of the biped, thus limiting the complications of moment arm orientation, biarticular muscles and muscle groups. In many cases, we used uniarticular muscles that were already existent in the other models ([Wang et al. 2012], [Geijtenbeek et al. 2013], OpenSim) and their via points were shifted a little for better alignment of DOF and moment arm. Some biarticular muscles were simply cut in two. For other DOFs, usually no muscles are modeled, such as for the ankle pronation (internal rotation) and supination (external rotation). For such cases, new muscles were added and their characteristics subjected to optimization.

We started with the four joints of the lower body, which are of greater interest in the biomechanical study of gaits. These can be seen on figure 13.

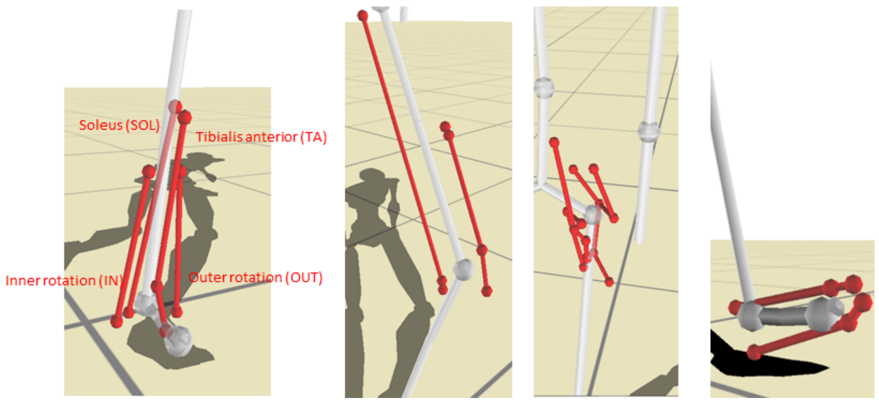


Figure 13: *muscles of the ankles, knees, hips and toes*

In order to obtain a gait with full muscle-actuation of these joints we used a simple forward walking gait present in SimBiCon called "fWalk" and changed the PD-gains first by hand and later by optimization, with the aim of obtaining a stable, resembling result. This turned out to be difficult, particularly for the ankle and hip joints.

In general, for the calibration of the controller, when a joint's PD-gains are too high, the error between current and target angles is over-corrected in one direction and then the opposite, which results in a trembling effect of the child rigid body. This was particularly observed on the ankles and hips, which required very low PD-gains to obtain a motion allowing the biped to remain upright. Figure 14 shows the angles that are taken by the left ankle joint during one full step phase of the "fWalk" gait, and the associated feedback torques. We see that to keep the ankle at a constant angle, which occurs during stance phase, the torques are very discontinuous. This is problematic to achieve with muscle actuation due to the fact that the activation signals are continuous. This is illustrated on figure 15 for the example of the four muscles of the left ankle. The desired activation represents what would achieve the torques of figure 14, the excitation takes the same values shifted in time by the neural delay, and the activation is the result of equation 10, i.e. the linear activation signal actually applied to the muscles.

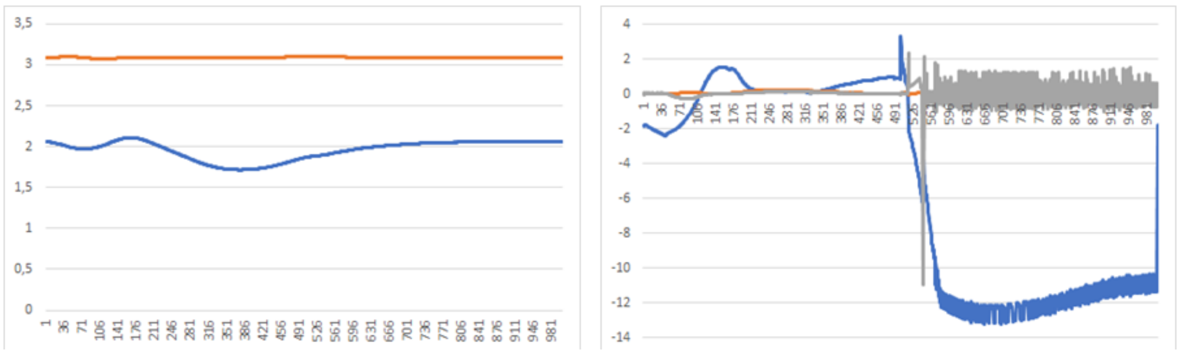


Figure 14: *Left ankle angles (plotted in radians, flexion in blue and rotation in orange) and feedback torques (x in blue, y in orange, z in grey) in time (1 KHz), over one cycle of the "fWalk" gait.*

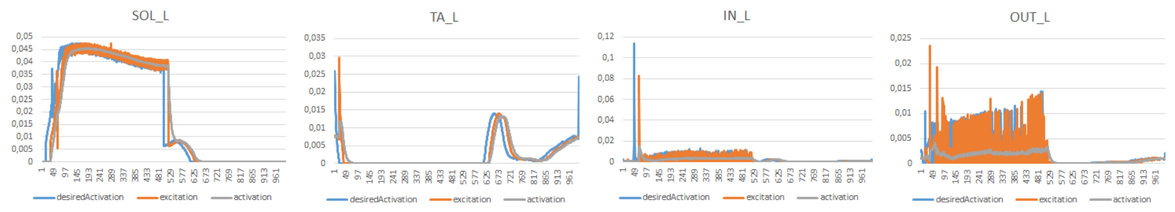


Figure 15: *Desired activation, excitation and (actual) activation of the four muscles of the left ankle in time (1000Hz), over one cycle of the "fWalk" gait.*

This resulted in very weak ankles that couldn't initiate the steps strongly enough. The toe joints are not present in the other models studied for this project. Similarly to the ankles, the toes seemed too weak, lagging behind the foot during the foot drop state of the gait. Yet the most problematic parametrization was to obtain a stable gait with both ankle and hip muscle-actuation. It was indeed very difficult to find parameters for the hips with which the biped could even perform a single step.

For the upper body, some muscles needed calibration of their constant parameters and PD-gains but it was easy to obtain results that were stable and comparable with the torques-based gait. Figure 16 shows the biped with the resulting 56 uniarticular muscles (covering the biped's 28 DOFs).

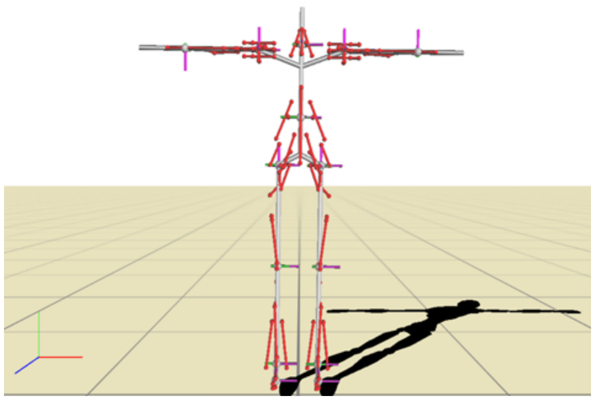


Figure 16: *Placement of a full muscle model.*

Two almost fully muscle-actuated gaits, that can be seen on figure 17, were used for offline optimization. In the first, all joints except the hips were muscle-actuated, which resulted in a stable gait with very short steps, slight slipping of the feet on the ground, and lagging toes. In the second, all joints except the back joint (between pelvis and torso) were muscle-actuated, which gave us a less stable gait with unnecessary outward movements of the legs.

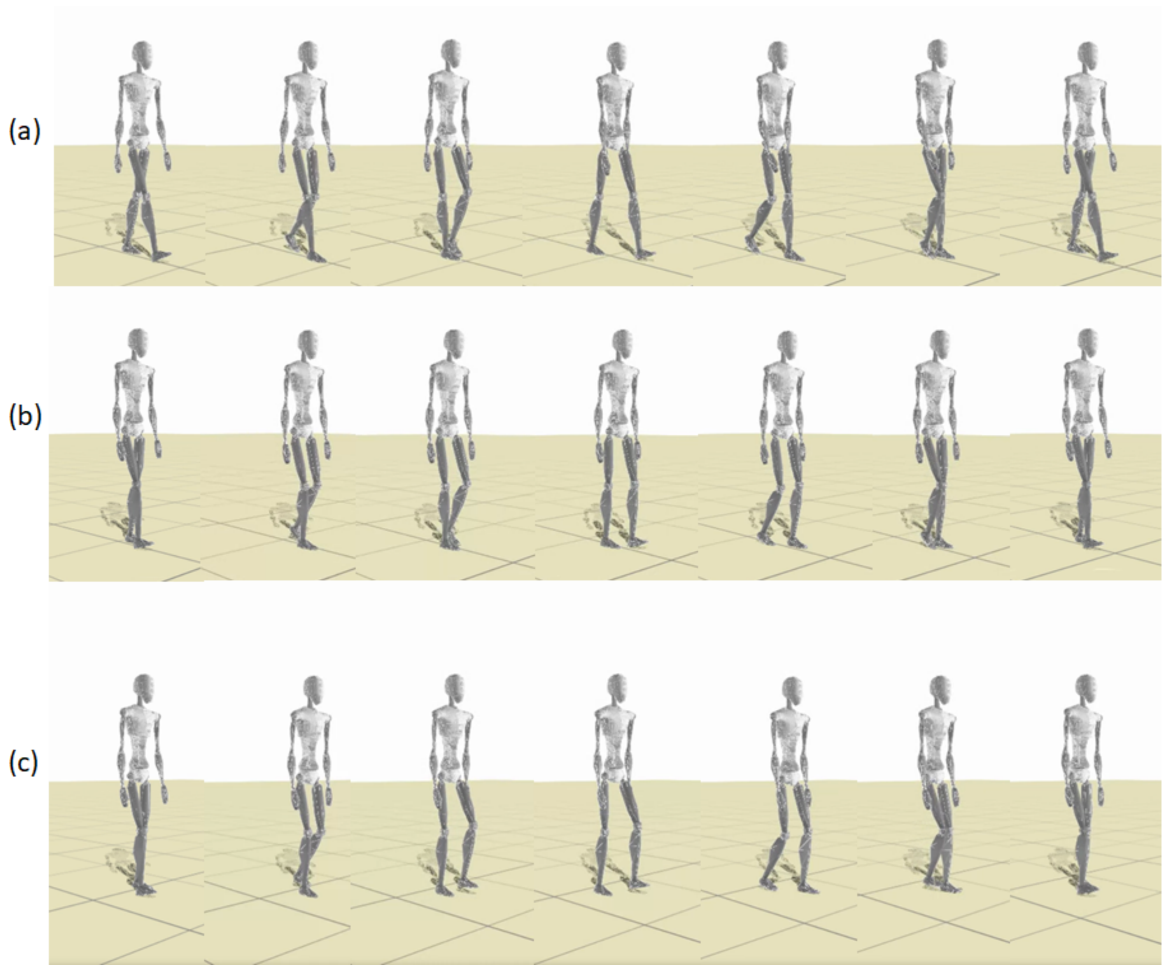


Figure 17: *(a) the reference "fWalk" gait with torques-actuation, (b) the gait with muscles on all joints except the hips, (c) the gait with muscles on all joints except the back. (b) and (c) were subject to optimization with the hopes of approaching gait (a).*

In the offline optimization, we modified the parameters of the controller (the gains of the PD-controller k_p and k_d) and model (F_{max}, l_{SE}^{slack} and l_{CE}^{opt}) using the objective function discussed in section 6, i.e. minimizing the sum of the magnitudes of the feedback torques.

8 Conclusion

During this internship I took up a framework of a biped controller for physics-based animations and added the concept of muscles to the model and the dynamics, converting the system from torques-based to muscle-based. The two main references were the works presented in [Wang et al. 2012] and [Geijtenbeek et al. 2013]. The desire of physical realism, real-time interactivity and stylized motion and bipeds confirmed the established complexity of the task of physical bipedal locomotion control.

The SimBiCon version used as a starting point provided the possibility to create different walking and running controllers and apply them on different bipeds, with the ability to create or modify gaits in real-time or to use motion capture data. However, under many aspects the controllers were unstable to changes, and some features such as stance foot contact control, were only added in later versions [Carensac 2015] and would probably have made it easier to achieve better results with the new muscle model.

As compared to the torque-based control method used in SimBiCon, muscle-based control provides a more detailed, lower-level model for human motion, and much more detailed than that of kinematic models of motion. This implies the disadvantage of needing an even larger set of parameters that have to be modeled, calibrated by optimization or identified from data, which adds to the already high complexity of forward physical biped walking controllers. However, for the need of medical applications, as is the case here, muscle-based actuation has the potential to create more realistic models of motion, because the constraints imparted by the muscles are implicit in the resulting motions.

With the aim of obtaining results similar to the torques-based control, several different complications were observed, that could each be subject to a closer study, like for example the moment arm orientation in relation to muscle force loss. The simplified model was able to reenact the torques-based motions of the upper body, and of the full body model with uniarticular muscles (given on figure 16). Yet the complete system loop for full body muscle-actuation seems to rely greatly on offline optimization for quality results. It was indeed difficult to obtain a stable full-body model with prior calibration by hand, due to the discontinuity of the required muscle activation. This is particularly the case for the hip and back joints that are crucial for overall balance, as well as for the toes and ankles, due to the complexity of ground collision management.

The greatest difficulty in this project was the question that often came up of why the obtained motion does not resemble the expected target. It is often complicated to know whether the results are after all justified by some of the muscle's features or if there are errors within the implementation, or if the calibration of some parameters is flawed. Similarly, for post-optimization results it is difficult to know whether the reason for unsatisfying results include implementation errors, the random nature of optimization method, the calculation of the objective function, the model of the biped, or limitations of the control architecture.

After optimization on the two models based on the same gait, with different muscles and prior calibration, the results still lacked realism and stability. Ways that are likely to improve these results include additional failing conditions in the case of self-collision or leg-crossing, or a more elaborate objective function, as discussed in section 6, that could benefit from case-specific tuning. For example, a higher weighting of the error of the hip torques could be used for the gait showing abnormal hip abduction. Another alternative would be to put in place additional feedback system within the control [Wang et al. 2012]. Furthermore, the use of an algorithm like the popular CMA would provide better odds in finding the optimal solutions as compared to the random selection process used here, even though a risk of finding a local minimum remains.

8.1 Future Work

In the PhD project following this internship, the SimBiCon framework with muscle actuation will be used by the biomecanists from Hannover Medical School to obtain predictive simulations using patient-specific data. In order to start interfacing the work from both labs, we sent a stable version of the framework (available here) containing one muscle-actuated gait (the modified "fWalk", without hip muscles and before optimization) as well as some torque-actuated gaits, along with a user-guide and some explanations on the use of data. During the OMEGA project, the framework will need to be adapted to the patient-specific data in its available format and to the specific needs of the experiments. For this it is likely that new models for muscle placement will be required depending on the desired accuracy and on which joints are of interest for particular simulations.

New controllers for specific existing gaits and the according new biped models will probably need further offline optimization for more stability and realism. The optimization process put in place should allow fairly easy additions or modifications of the parameters that are to be changed, as well as the objective function and the failing conditions. These factors can vary widely depending on the criteria specific to each case of study. One could for example chose to fully adapt a gait controller to a fixed biped model in order to obtain a result matching recorded data.

Depending on the realism of the muscle placement, it could also be of interest to add the mechanism of wrapping points, particularly for muscles of the knees, elbows and shoulders. This can either be done by adding conditional via points that exist only depending on the angle formed by the joint, or by checking for collisions between the muscles and the bones, and adding new points close to the surface of the bone. The latter can be done with the current simplifications that assume that muscles and bones are mass-less segments, or it could further add to the realism of the model to include 3D volume models of the bones and muscles. It has been done in another internship on SimBiCon to replace the existing meshes of the biped's rigid bodies by meshes of bones resulting from scanners of real patients. As for the muscles, the addition of the notions of mass, muscle width and pennation angle would add to physical realism and add a factor to the calculations of the output force (see equation 1). Finally, the optimisation of muscle routing done by [Geijtenbeek et al. 2013] could be useful here, as it might even provide predictive post-surgery information.

Furthermore, to better match the practical needs of the application, it could be interesting to replace the actuation of some specific DOFs by passive spring dampers or direct torque actuation. Specifically, based on [Geijtenbeek et al. 2013], muscle actuation could be omitted for spine joints and hips in the axial direction, as well as for planar rotation of the ankles. Additional axial rotation for the ankles could also be added to the model, with simplified actuation. For now, it is only possible to chose for all of one joint's DOFs to be either muscle or torques-actuated, therefore adding the possibility of a different management for DOFs of the same joint could bring some simplifications to the overall system.

8.2 Personal Experience

For the aims of this internship, I had to acquire the necessary background knowledge in biomechanics and animation physics, including many notions I wasn't familiar with. I very much enjoyed discovering the medical aspects and applications of this project, and also of other current projects of the SAARA team. Before this internship, I had only a brief overview on how the study of areas such as animation, geometrical modeling, meshes and algorithmic geometry can apply to medicine, and had no experience at all with the fields of physical animations and biomechanics.
Other than the fact that I have particularly enjoyed the theoretical documentation and practical applications of this internship, it also gave me an insight on the work atmosphere in a research laboratory and on a smaller scale in the team my project was associated with. It provided me with a better idea of the work routine of PhD students and professors.

Acknowledgments

I would like to thank all the people who contributed in making this internship an instructive experience. Firstly I express my thanks to Nicolas Pronost who allowed me to be a part of this project and continuously supervised my progress. I very much appreciate his kind help in acquiring the necessary theoretical background knowledge and the balance between support and autonomy I was able to work under.
I'm also thankful toward Raphaëlle Chaine, head of the Master in Computer Graphics, 3D Technology and Development, and Saïda Bouakaz, leader of the SAARA team, for granting me the opportunity of this internship.
Finally, I would like to thank the other intern students who worked in the same premises of the LIRIS laboratory as me during this internship, for creating a very positive, productive and supportive work environment.

References

[1] S. Carensac. "Contrôle du mouvement de marche de personnages virtuels en milieu liquide". In: (2015).

[2] S. Coros and P. Beaudoin. "Generalized biped walking control". In: *ACM Trans. Graph. SIGGRAPH* 29, 4 (2010).

[3] S. Coros, P. Beaudoin, and M. van de Panne. "Robust Task-based Control Policies for Physics-based Characters". In: *ACM Trans. on Graphics* 28, 5 (2009).

[4] S. Coros, P. Beaudoin, and M. van de Panne. *Simbicon. Controller Editor Framework.* Version 1.5. 2009. URL: `www.cs.ubc.ca/~van/simbicon_cef`.

[5] A.L. Cruz Ruiz et al. "Muscle-Based Control For Character Animation". In: *Computer Graphics Forum* (2016).

[6] S.L. Delp et al. "OpenSim: Open-source software to create and analyze dynamic simulations of movement." In: *IEEE Transactions on Biomedical Engineering* (Nov. 2007).

[7] T. Geijtenbeek, M. van de Panne, and F. van der Stappen. "Flexible Muscle-Based Locomotion for Bipedal Creatures". In: *ACM Trans. Graph.* 32, 6.206:1–206:11 (2013).

[8] T. Geijtenbeek and N. Pronost. "Interactive Character Animation Using Simulated Physics: A State-of-the-Art Review". In: *Comp. Graph. Forum* 31, 8.2492–2515 (2012).

[9] H. Geyer and H. 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, 3.263–73 (2010).

[10] Y. Lee et al. "Locomotion Control for Many-Muscle Humanoids". In: *ACM Trans. Graph.* 33, 6 (Nov. 2014).

[11] *The AnyBody^{TM} Modeling System.* 2013. URL: `http://www.anybodytech.com`.

[12] D. Thelen, F. Anderson, and S. Delp. "Generating dynamic simulations of movement using computed muscle control." In: *Journal of Biomechanics* 36, 321–328 (2003).

[13] J. Wang et al. "Optimizing locomotion controllers using biologically-based actuators and objectives." In: *ACM Trans. on Graphics* 31, 4, 25 (2012).

[14] K. Yin, K. Loken, and M. van de Panne. "SIMBICON: simple biped locomotion control." In: *ACM Trans. Graph.* 26, 3 (2007).

[15] F.E. Zajac. “Muscle and tendon: properties, models, scaling, and application to biomechanics and motor control”. In: *Critical Reviews Biomedical Engineering* 359-411 (1989).