OpenSim

—

Evaluating an Open Source Simulation Software for Human Movements

Technical Report
Nicolas Schwenck

Department of Mechanics
Kungliga Tekniska Högskolan

Stockholm, July 2009
Preface

This thesis mainly results from my work at the Department of Mechanics from August till December 2008.

I thank Professor Anders Eriksson who helped me to get in touch with the field of biomechanics and gave me the opportunity to write this thesis at KTH and also for always offering me great advice and support.

I thank Lanie Gutierrez-Farewik for the contact to the Karolinska University Hospital, for supervising the work in the motion laboratory and the medical part guidance.

I thank Olesya for the very productive discussions and the corporate work in the motion laboratory.

I thank Professor J.F. Wagner for the support at the University of Stuttgart.

I also would also like to thank my parents who have given me the financial possibility for staying in Stockholm and who always have supported me.
Abstract

The open source software OpenSim is a biomechanical simulation software. It was specially developed at Stanford University for the analysis of human movements. Its functions were analysed and tested by means of some examples. In this project the focus of the examples lays in manipulating motions of the human gait.

Forward dynamic simulations were done by altering muscle excitation patterns and computing the resulting motions. Backward dynamic simulations were done by scaling motions or creating new movements, computing the needed muscle forces and excitations.

The compatibility to an important commercial software (SIYM) on which OpenSim is based was tested.

This thesis consequently shows the feasibility of the methods mentioned above to manipulate motions and its applicability to real cases and states the prospects of OpenSim in the field of biomechanical simulation software for analysing human movements.

The test results show that OpenSim can fulfil the requirements within the small range of the test cases. It also shows that some modules are not stable enough and that documentation and the error message system have to be improved. The tests show that further verification and validation is necessary to confirm the used algorithms. The usage as commercial software replacement is not recommended at present, but it seems to be a cheap and useful alternative for teaching purposes.
Contents

Preface iii

Abstract iv

Contents vi

List of Figures ix

1 Introduction 1
   1.1 Biomechanics 1
   1.2 Simulations of Human Movement 1
       1.2.1 Motion Laboratory 2
   1.3 OpenSim and SIMM 2
   1.4 Outline of this Thesis 3

2 Fundamentals 5
   2.1 Multi Body Mechanics 5
   2.2 Numerics 6
       2.2.1 Runge-Kutta 6
       2.2.2 Optimisation 8

3 OpenSim 11
   3.1 Example Operating Sequence 11
       3.1.1 Scaling 11
       3.1.2 Inverse Kinematics 11
       3.1.3 Inverse Dynamics 12
       3.1.4 Forward Dynamics and Analysis 12
   3.2 Muscle Force Computation 13
       3.2.1 Correlation Between Muscle Force and Excitation Level 14
   3.3 Special Features 15
       3.3.1 Residual Reducing Algorithm 15
       3.3.2 Computing Muscle Control 17

4 Results 19
   4.1 Motion Laboratory results 19
   4.2 Manipulate Motions 21
       4.2.1 Musculoskeletal Geometry 22
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.3.3 Movements Without Ground Reactions</td>
<td>63</td>
</tr>
<tr>
<td>A.4 Source Codes</td>
<td>64</td>
</tr>
<tr>
<td>A.4.1 Motion Laboratory</td>
<td>64</td>
</tr>
<tr>
<td>A.4.2 Scale Motions</td>
<td>67</td>
</tr>
</tbody>
</table>
List of Figures

3.1 Dimensionless model of muscle and tendon .......................................................... 14
4.1 Spatial coordinate system of the human body .......................................................... 20
4.2 Location of the body-segmental reference frames, the lower extremities ................... 22
4.3 Location of the body-segmental reference frames, the foot .................................... 23
4.4 Comparison of original and changed excitations of left knee flexors ...................... 25
4.5 Comparison of original and changed excitations of left knee extensors .................... 25
4.6 Left Knee Flexion/Extension ................................................................................. 26
4.7 Flexion/Extension of the right hip joint ................................................................. 27
4.8 Comparison between CMC calculated excitation of right Gluteus Maximus ............ 28
4.9 Comparison between CMC calculated excitation of right Psoas and Iliacus ............ 28
4.10 Prescribed values for the hip/knee during the kick motion .................................... 29
4.11 Position of the left foot and knee joint during the kick motion ............................... 30
4.12 Summed main flexors/extensors muscle forces of the hip/knee during knee bending 31
4.13 Summed main flexors/extensors muscle forces of the hip/knee during a kick movement ........................................................................................................................................ 31
5.1 Extension and flexion of the left and right knee and hip joint .................................. 36
5.2 CMC calculated excitation of the Gastrocnemius ....................................................... 37
5.3 CMC calculated excitation of the Vastus Intermedius ............................................... 37
5.4 Angle values for subtalar and ankle joint of the left leg during a Forward Dynamics computation ......................................................................................................................... 38
5.5 Flexion/Extension of the right hip joint .................................................................... 39
5.6 Comparison between CMC calculated excitation of right Gluteus Maximus for a longer period of time ................................................................................................................ 39
5.7 Comparison between CMC calculated excitation of right Psoas and Iliacus for a longer period of time ................................................................................................................ 40
A.1 Marker positions for the motion lab recordings ......................................................... 52
A.2 Hip Flexion/Extension - Hip transverse plane ......................................................... 53
A.3 Hip Flexion/Extension - Knee transverse plane ....................................................... 53
A.4 Hip Flexion/Extension - Hip sagittal plane .............................................................. 53
A.5 Hip Flexion/Extension - Knee sagittal plane ............................................................ 53
A.6 Hip Flexion/Extension - Hip frontal plane ............................................................... 53
A.7 Hip Flexion/Extension - Knee frontal plane ............................................................ 53
A.8 Hip Ab-/Adduction - Hip transverse plane ............................................................... 54
A.9 Hip Ab-/Adduction - Knee transverse plane ............................................................ 54
A.10 Hip Ab-/Adduction - Hip sagittal plane ........................................ 54
A.11 Hip Ab-/Adduction - Knee sagittal plane ...................................... 54
A.12 Hip Ab-/Adduction - Hip frontal plane ........................................ 54
A.13 Hip Ab-/Adduction - Knee frontal plane ....................................... 54
A.14 Hip Rotation - Hip transverse plane ........................................... 55
A.15 Hip Rotation - Knee transverse plane ......................................... 55
A.16 Hip Rotation - Hip sagittal plane ............................................... 55
A.17 Hip Rotation - Knee sagittal plane ............................................. 55
A.18 Hip Rotation - Hip frontal plane ................................................ 55
A.19 Hip Rotation - Knee frontal plane .............................................. 55
A.20 Knee Flexion/Extension - Hip transverse plane ............................. 56
A.21 Knee Flexion/Extension - Knee transverse plane ........................... 56
A.22 Knee Flexion/Extension - Hip sagittal plane .................................. 56
A.23 Knee Flexion/Extension - Knee sagittal plane ................................ 56
A.24 Knee Flexion/Extension - Hip frontal plane ................................... 56
A.25 Knee Flexion/Extension - Knee frontal plane ................................ 56
A.26 Comparison of original and changed excitations of right hip extensors (Iliopsoas) 57
A.27 Right knee flexion/extension ...................................................... 57
A.28 Comparison of CMC calculated excitation of right Quadriceps Femoris 59
A.29 Comparison of CMC calculated excitation of right Plantaflexors .......... 59
A.30 Separate muscle forces of the main flexors/extensors of the hip/knee during knee
       bending ................................................................. 60
A.31 Separate muscle forces of the main flexors/extensors of the hip/knee during a
       kick movement .......................................................... 60
A.32 Summation of the CMC calculated excitations of the main knee flexors .... 61
A.33 Summation of the CMC calculated excitations of the main knee extensors ... 61
A.34 Comparison of CMC calculated excitation of right Quadriceps Femoris for a
       longer period of time .................................................... 62
A.35 Comparison of CMC calculated excitation of right Plantaflexors for a longer pe-
       riod of time ............................................................. 62
A.36 CMC computed motion without ground reaction forces ...................... 63
A.37 Inverse Dynamics computed motion without ground reaction forces ........ 63
1 Introduction

1.1 Biomechanics

Biomechanics of human movement is an interdisciplinary field which combines the field of engineering mechanics with the medical field of the anatomy and physiology of the living including the human body. It describes, analyses and assesses different parts of the living with the methodology of mechanical engineering.

In this context mechanical engineering mainly includes the applied mechanics: thermodynamics, fluid mechanics and solid mechanics. By applying the laws and concepts of physics, biomechanical mechanisms and structures can be simulated and studied. Biomechanical mechanisms and structures comprise different dimensions of observation. The smallest is an inner cell observation (molecular dimension), followed by other inner body observations (muscles, tendons, organs, other tissue) and the greatest dimension is the observation of whole bodies.

Nature has always developed a solution for every problem. Engineers working in the field of biomechanics research natural mechanisms and try to apply the results to new products. This can for example mean developing swim-suits with a surface similar to shark skin or building the supporting structure of skyscrapers similar to special grasses or reed.

The analysis of the body (mainly the human body, but also of other species) is very important for medical professionals. Many medical principles can be explained by mechanical concepts, for example the heart as a pump or muscles as actuators and movements as multi body mechanics, or with the help of the Finite Element Analysis.

This knowledge about nature is very often useful to get a better understanding of athletic performance, too. Athletes for instance are able to optimise their training and achieve better results in competitions.

In this thesis simulation of human movements is seen as a part of biomechanics.

1.2 Simulations of Human Movement

Simulation instead of experiments comes naturally with advantages as well as disadvantages. The disadvantages are the occurrence of numerical errors and a lack of verification if there is no comparison to experimental data. The computational simulation is normally cheaper and faster. The resulting data is available for the whole model and not only for specific measurement points. There can be data which can only be computed through simulation and cannot be measured. There is no feedback from the measurement to the model. The model and the setup can
be changed more easily. Model and setup failures/errors can happen in both simulation and experiments.

In the field of biomechanics the used software and its functions are highly relevant. Medical engineers support physiotherapists, orthopaedists and surgeons by simulating the human body. It facilitates the work of health professionals, improves the quality of the patient treatment, leads to new methods of treatment and medical procedures and helps athletes improving their training.

Simulating motions may have different objectives, but there are two main reasons. On the one hand there is a medical interest in motions and on the other hand there is a sportive interest.

The medical interest in simulating movements is to find the weakness or abnormality in it or to simulate the outcome of a medical treatment (e.g. physiotherapy, surgery). The sportive interest is to analyse the athletes course of motions to improve his training and to lead him to better results in competitions.

1.2.1 Motion Laboratory

A motion laboratory is the place where the data for biomechanic simulations is usually collected. In such a motion laboratory there are several tracking cameras (at least two) which record the motion. The test object must be equipped with trackers. These trackers are the only things the cameras can register. They convert the recorded motion into coordinates at user specified time intervals. The trackers are placed at significant places on the test object, which makes it possible to simulate the needed bones as rigid links each with its own coordinate system.

Normally there are also force plates in a motion laboratory. These force plates measure the forces and the centre of pressure at the specified time steps with the help of pressure sensors, so that they can be put into the simulation, too.

Sometimes electromyographs (EMG) are used to measure muscle excitations. EMG measures the electrical potential produced by the selected muscle during contraction and resting. These EMG results can be compared qualitatively with the excitation levels computed by \textit{CMC}. EMG results can not be associated directly to muscle forces, [10]. For that purpose an additional calibration would be necessary.

1.3 OpenSim and SIMM

OpenSim is a freely available software package which offers the possibilities to build, exchange and analyse computer models of the musculoskeletal system and run dynamic simulations of movement. It is an open source project which started at the Stanford University. It is based on the commercial software SIMM (MusculoGraphics Inc, Stanford, USA, [24]), [4, 25, 26].

SIMM is used by hundreds of biomechanics researchers to create computer models of musculoskeletal structures and to simulate movements such as walking, cycling, running and stair climbing.
OpenSim was developed as a supplement to SIMM. There are some features which are only or better available in SIMM (e.g. model creation, motion import) and some features significant for OpenSim (residual reducing, cf. chapter 3.3.1; computing muscle excitation, cf. chapter 3.3.2).

Open source software comes naturally with advantages and disadvantages. The possibilities for every user to develop their own extensions, enhance the base code and share these innovations with the community are advantages. At the moment, however, the OpenSim source code is not public, only the usage is for free. Because open source software is free, it is perfectly suited for users with low budgets or e.g. teaching, because the use of commercial software normally means buying a license for each student. The lack of documentation can cause problems and can force the user to invest much time to understand the ideas and program codes of other users.

The developers' intention and proposal is to use SIMM and OpenSim together as complement, so the strengths of both software packages can be used.

1.4 Outline of this Thesis

In this project OpenSim should be tested for the Biomechanics group of the Department of Mechanics at Kungliga Tekniska Högskolan, Stockholm. A documentation of main software functions and the testing experiences is needed. Thereby, the main focus lays on motion handling, scaling and manipulating. Furthermore, the compatibility of SIMM and SIMM-models (e.g. import/export) should be tested and important functions in OpenSim should be compared to the SIMM functions.

Chapter two contains the mechanical and mathematical fundamentals on which some parts of this thesis are based. The mechanical fundamentals are multi-body mechanics. The main mathematical fundamentals are numerical integration and optimisation.

Chapter three contains basic explanations of the OpenSim methods and how they work. It describes an exemplary operating sequence and compares the muscle force computation in SIMM and OpenSim.

In chapter four the results of testing are outlined in respect to the main focus manipulating motions, scaling motions and compatibility to SIMM.

Chapter five results from additional work with OpenSim after version 1.8.1 was released.

Concluding, in chapter six the author’s opinions are summarised.
2 Fundamentals

2.1 Multi Body Mechanics

Today there are two most used types of simulation software in mechanical engineering simulation of solid bodies. On the one hand there are Finite Element Analysis (FEA) simulation tools and on the other hand there are Multi Body Analysis (MBA) simulation tools.

In a MBS (Multi Body System) the solid body is fragmented and seen as several rigid elements linked by joints, springs, etc. The fragmentation results from design-engineering conditions. The separate parts can revolve or move (possibly a high magnitude) comparatively to each other. It is often used for dynamic analysis. The key feature of a system that makes it suitable for multibody treatment is the observation that its motion is localised, i.e., it is well-described as a set of independently identifiable parts which undergo large motions with respect to one another, but are themselves rigid or nearly rigid, [23].

A FE-Model (a system consisting of elements using the Finite Element Analysis) is mainly used for simulating cases in which deformations are important. The solid body is discretised by many comparatively small elements (bar-shaped, triangular, quadrangular, etc.) forming a mesh with distributed mass and elasticity. The element size depends on the problem and the required accuracy. Today a number of several million elements is not unusual. It has considerably more degrees of freedom (DoF), [27].

Table 2.1 shows a comparison of the different characteristics of the physical models. FE models and MBS are discontinua models. "COS" means continuous system.

Because the resources of OpenSim to allow model creation are very limited, most of the models used in OpenSim were created by using a different software package, for example SIMM.

In OpenSim two multi body mechanics engines are used. The SIMM Kinematics engine supports

<table>
<thead>
<tr>
<th>Model</th>
<th>MBS</th>
<th>FEA</th>
<th>COS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part</td>
<td>rigid, elastic if applicable</td>
<td>elastic</td>
<td>elastic</td>
</tr>
<tr>
<td>Geometry of the System</td>
<td>complex</td>
<td>complex</td>
<td>simple</td>
</tr>
<tr>
<td>Number of DoF</td>
<td>restricted</td>
<td>high</td>
<td>infinite</td>
</tr>
<tr>
<td>Coverage of Deformations</td>
<td>restricted</td>
<td>always</td>
<td>always</td>
</tr>
<tr>
<td>Forces, Torques</td>
<td>discrete</td>
<td>distributed and discrete</td>
<td>distr. and discr.</td>
</tr>
<tr>
<td>Computed Eigenfrequency</td>
<td>rather too low</td>
<td>rather too high</td>
<td>exact</td>
</tr>
</tbody>
</table>

Table 2.1: Characteristics of Physical Models [27]
all SIMM models. With this engine active, every model created in SIMM can be shown in OpenSim. The SIMM Kinematics engine, however, does not support the dynamic modules of OpenSim like RRA, CMC etc.

The other multibody mechanics engine used in OpenSim is SimBody. SimBody is designed especially for biosimulations. It provides the multibody system formulation and the call of different numeric methods, but not the numeric methods themselves. It is also part of the open source community SimTK and that is also the source where SimBody gets the numerical methods from.

In OpenSim and SIMM a model consists of bodies and actuators. Bodies represent the bones and the skeletal structure of the human body. Bodies are rigid. They can not be deformed. This is a feasible assumption because the occurring forces during normal (healthy) movements are very low and the Young’s modulus of the bones of an adult is comparatively high, so that deformations of the bones are not to be expected. If abnormal movements should be analysed or the test person has abnormal bones or is simply a child (bones become harder during grow up) this must be taken into account for the choice of the model.

The bodies are connected through joints. There are two possible joints. On the one hand unprescribed joints which are also known as degrees of freedom, on the other hand prescribed joints, which lead to motions between two or more bodies in respect to each other.

Actuators represent the muscle-tendon structure. They can create active (muscle contraction) and passive forces (tightened tendons, represented through a spring element) and are necessary for creating motions.

### 2.2 Numerics

#### 2.2.1 Runge-Kutta

Mathematically speaking simulation means solving problem specific differential equations by numerical integration. The software in this thesis uses Runge-Kutta methods to integrate these specific equations.

Ordinary differential equations can be seen as:

<table>
<thead>
<tr>
<th>$c_2$</th>
<th>$a_{21}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_3$</td>
<td>$a_{31} \ a_{32}$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$c_s$</td>
<td>$a_{s1} \ a_{s2} \ ... \ a_{s,s-1}$</td>
</tr>
</tbody>
</table>

| $b_1 \ b_2 \ ... \ b_{s-1} \ b_s$ |

| $\hat{b}_1 \ \hat{b}_2 \ \ ... \ \hat{b}_{s-1} \ \hat{b}_s$ |
### Fundamentals

#### Table 2.4: Butcher’s array for Runge-Kutta-Fehlberg 4(5) [9]

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1/4</th>
<th>3/8</th>
<th>12/13</th>
<th>1</th>
<th>1/2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>h</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
<td>e</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>9/32</td>
<td>1932/2197</td>
<td>439/216</td>
<td>-8/27</td>
</tr>
<tr>
<td>1/4</td>
<td>9/32</td>
<td>2197/2197</td>
<td>2197/2197</td>
<td>1097/2197</td>
<td>2197/2197</td>
<td>7296/2197</td>
</tr>
<tr>
<td>3/8</td>
<td>439/216</td>
<td>-8</td>
<td>3680/513</td>
<td>-845/4104</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12/13</td>
<td>439/216</td>
<td>-8</td>
<td>3680/513</td>
<td>-845/4104</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>16/135</td>
<td>0</td>
<td>6656/12825</td>
<td>28561/56430</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/2</td>
<td>-8/27</td>
<td>2</td>
<td>-3544/2565</td>
<td>1859/4104</td>
<td>-11/40</td>
<td></td>
</tr>
<tr>
<td>y1</td>
<td>25/216</td>
<td>0</td>
<td>1408/2565</td>
<td>2197/4104</td>
<td>-1/5</td>
<td></td>
</tr>
<tr>
<td>y1</td>
<td>25/216</td>
<td>0</td>
<td>1408/2565</td>
<td>2197/4104</td>
<td>-1/5</td>
<td></td>
</tr>
<tr>
<td>9/27</td>
<td>2</td>
<td>-3544/2565</td>
<td>1859/4104</td>
<td>-11/40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/27</td>
<td>2</td>
<td>-3544/2565</td>
<td>1859/4104</td>
<td>-11/40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9/50</td>
<td>2</td>
<td>-3544/2565</td>
<td>1859/4104</td>
<td>-11/40</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Runge-Kutta methods, which are one-step methods, construct an approximation of the solution $y_{n+1}^n$ at $x_{n+1} = x_n + h$ ($h$ is the step size) from the step before ($y_n$ at $x_n$) via the formulas, [8, 14]:

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i k_i \tag{2.2.1}$$

Here $k_i$ are internal stages which are computed for every step separately. The number of internal stages is $s$. The internal stages are given by:

$$k_i = f(x_i^n, y_i^n) \tag{2.2.2}$$

and $x_i^n$, $y_i^n$ are defined as:

$$x_i^n = x^n + hc_i \tag{2.2.3}$$

for $i = 1, \ldots, s$

and

$$y_i^n = y^n + h \sum_{j=1}^{s} a_{ij} k_j$$

The coefficients $a_{ij}$, $b_i$, $c_i$ determine the method and are usually shown in a so called Butcher’s array (cf. table 2.2).

If $a_{ij} = 0$ for $i \leq j$ it is possible to compute the internal stages $k_1^n, \ldots, k_s^n$ one after the other from equations 2.2.2 and 2.2.3 by explicit function evaluation. Such methods are called explicit. Otherwise, equation 2.2.2 constitutes a nonlinear system for the internal stages and the method is called implicit.
The Fehlberg method is an embedded Runge-Kutta formula, i.e., for every numerical computed step \( y^n \) it contains an expression \( \hat{y}_n \) of higher accuracy (order), which can be used for error and step size control and in particular makes step rejections, as in some extrapolation methods, less expensive, [9]. The coefficients then are usually shown in the extended Butcher’s array (cf. table 2.3).

The coefficients have to fulfill both:

\[
\begin{align*}
y^1 &= y^0 + h(b_1 k_1 + \cdots + b_s k_s) \\
\hat{y}^1 &= y^0 + h(\hat{b}_1 k_1 + \cdots + \hat{b}_s k_s)
\end{align*}
\]  

(2.2.4)

The approximation of the solution at step one is \( y^1 = f(x^1) \). The initial condition is \( y^0 = f(x^0) \). \( \hat{y} \) describes the higher order, respectively.

Fehlberg’s methods try to minimise the local error for the lower order, therefore they underestimate the local error. It becomes especially noticeable if a high accuracy is required. Furthermore, the local errors generally have very little in common with the global errors. The most popular Fehlberg method is the so called Runge-Kutta-Fehlberg 4(5) (RKF). The coefficients for the RKF 4(5) are shown in table 2.4.

Whenever an ordinary difference equation is to be solved, OpenSim uses the RKF4(5) integrator.

2.2.2 Optimisation

Optimisation in an engineering context means finding the best solution to a given problem. Therefore the problem has to have measurable parameters. With increasing complexity very often the real optimum can not be found without support of huge computing power. Optimisation examples in context of biomechanics are: increasing the possible length of a long jump with given muscle strength, reducing the metabolic energy during normal gait, improving the gait of a cerebral palsy patient through a tendon transfer surgery, fitting experimental (motion laboratory) and simulated data (OpenSim).

In this chapter the focus lays on mathematical optimisation problems. In this context optimisation problems occur when a mathematical model should be applied to observed data. In order to reduce the errors in the observations, more measurement data are taken than necessary (unknown parameters in the mathematical model). The optimisation problem is to solve an overdetermined system of equations, [1].

Considering a set of \( m \) data points, \( (x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m) \), and a curve (model function) \( y = f(x, \beta) \), that in addition to the variable \( x \) also depends on \( n \) parameters, \( \beta = (\beta_1, \beta_2, \ldots, \beta_n) \), with \( m \geq n \). It is desired to find the vector \( \beta \) of parameters such that the curve fits the given data in the least squares sense, [28], i.e., the sum of squares

\[
S = \sum_{i=1}^{m} r_i^2
\]  

(2.2.5)
is minimised, where the residuals (errors) $r_i$ are given by

$$r_i = y_i - f(x_i, \beta) \quad \text{for } i = 1, 2, \ldots, m$$

(2.2.6)

The minimum value of $S$ occurs when the gradient is zero. Since the model contains $n$ parameters there are $n$ gradient equations. In a non-linear system, the derivatives are functions of both the independent variable and the parameters, so these gradient equations do not have a closed solution. Instead, initial values must be chosen for the parameters. Then, the parameters are refined iteratively, that is, the values are obtained by successive approximation. By approximation of each iteration step with a first-order Taylor series expansion (linearisation) and after some rearrangements it will lead to, [29]:

$$(J^T J) \Delta \beta = J^T \Delta y.$$  \hspace{1cm} (2.2.7)

as normal equations. The Jacobian matrix, $J$, at this point is defined by

$$-J_{ij} = \frac{\partial r_i}{\partial \beta_j} \quad \text{for } j = 1, \ldots, n$$

The residuals are then given by

$$r_i = \Delta y_i - \sum_{j=1}^{n} J_{ij} \Delta \beta_j \quad \text{with} \quad \Delta y_i = y_i - f(x_i, \beta^k)$$

When the observations are not equally reliable, a weighted sum of squares may be minimised,

$$S = \sum_{i=1}^{m} W_{ii} r_i^2$$  \hspace{1cm} (2.2.8)

This leads to the weighted normal equations:

$$(J^T W J) \Delta \beta = J^T W \Delta y$$  \hspace{1cm} (2.2.9)

which can now be solved by numerical solvers (e.g., methods based on QR-decomposition or methods based on the Gaussian-elimination), [1].
3 OpenSim

Each test in this thesis was made with OpenSim 1.5.5 released on the 31th of July 2008, therefore each result, comment and conclusion refers to version 1.5.5 and the software packages current at that time.

OpenSim is an open source software developed to simulate and analyse the neuromusculoskeletal system, primarily of the human body[4]. There is a stable version available, but the development is still in progress. OpenSim does not offer a great possibility to generate, modify or change models or parts of models. It is necessary to have a generic model which can be imported in OpenSim, e.g. a SIMM model.

OpenSim offers the possibilities to view SIMM models and motions. It can handle and compute scaling, inverse kinematics, inverse and forward dynamics. It has two special features which are not implemented in SIMM: on the one hand a Residual Reducing Algorithm (RRA) (cf. chapter 3.3.1) and on the other hand a Computing Muscle Control (CMC) algorithm (cf. chapter 3.3.2). Furthermore, OpenSim offers GUIs for the users to easily manipulate, change or create data for muscle excitation and muscle parameters. Also several analysis and plot GUIs are available.

3.1 Example Operating Sequence

3.1.1 Scaling

The first step in OpenSim should be to scale the generic model to the concrete test case and the special anthropometric requirements. The anthropometric data can be changed either manually or automatically by importing measurement data from motion capturing data and its trackers. The weight can be changed manually.

The second step is to get the motion in the model. Therefore, the motion capturing data has to be imported (cf. chapters 1.2.1, 4.1) followed by the program routines Inverse Kinematics (IK) to create the generalised coordinates of the joint angles et cetera, and the Inverse Dynamics (ID) to generate the forces and torques.

3.1.2 Inverse Kinematics

IK goes through each time step (frame) of motion and computes generalised coordinate values which positions the model in a pose that best matches experimental marker and coordinate values for that time step. Mathematically, the best match is expressed as a weighted least
squares problem, whose solution aims to minimise both marker and coordinate errors, [7]. Basics of numerical solving least squares problems are described in chapter 2.2.2.

\[
\min_{\mathbf{q}} = \left[ \sum_{i \in \text{markers}} w_i \| \mathbf{x}_i^\text{exp} - \mathbf{x}_i(\mathbf{q}) \|^2 + \sum_{j \in \text{unprescribed coords}} \omega_j (q_j^\text{exp} - q_j)^2 \right] \tag{3.1.1}
\]

Equation 3.1.1 is the weighted least squares problem solved by IK. The vector \( \mathbf{q} \) contains the generalised coordinates being solved for, \( \mathbf{x}_i^\text{exp} \) is the experimental position of marker \( i \), \( \mathbf{x}_i(\mathbf{q}) \) is the position of the corresponding marker on the model (which depends on the coordinate values). \( q_j^\text{exp} \) stands for the experimental value for coordinate \( j \). Prescribed coordinates are set to their experimental values, i.e. \( q_j = q_j^\text{exp} \) for all prescribed coordinates \( j \), [7]. In what way this quadratic optimisation problem is solved is not specified in the User’s Guide. The marker weights \( (w_i) \) and coordinate weights \( (\omega_j) \) are specified by the user.

### 3.1.3 Inverse Dynamics

The idea of the Inverse Dynamics tool (ID) is to solve the classical equations of motion:

\[
\begin{align*}
\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{G}(\mathbf{q}) &= \mathbf{\tau} \\
\end{align*}
\tag{3.1.2}
\]

where \( \mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}} \in \mathbb{R}^N \) are the vectors of generalised positions, velocities and accelerations, respectively, and \( N \) is the number of degrees of freedom. The matrix \( \mathbf{M}(\mathbf{q}) \in \mathbb{R}^{N \times N} \) is the system mass matrix, whereas \( \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^N \) is the vector of Coriolis and centrifugal forces, \( \mathbf{G}(\mathbf{q}) \in \mathbb{R}^N \) is the vector of gravitational forces and \( \mathbf{\tau} \in \mathbb{R}^N \) is the vector of generalised forces to be solved.

The motion of the model is completely defined by the generalised positions, velocities and accelerations. Consequently, the remaining term on the right-hand side of the equations of motion is unknown. ID uses the known motion of the model to solve the equations of motion for the unknown generalised forces, [7].

Now the case specific model and motion is found. The next steps are the RRA module (cf. chapter 3.3.1) to make the model more consistent with the experimental data and the CMC module (cf. chapter 3.3.2) to compute the muscle excitation patterns and muscle forces which produce the movement.

### 3.1.4 Forward Dynamics and Analysis

After finishing CMC it is possible to do a Forward Dynamic simulation. This simulation will compute the generalised coordinates resulting from prescribed excitation. It is possible to do it without any alterations of the excitation to check how coincident the model with the experimental data is. Slightly changing some excitation patterns to investigate what effect it will have on the recorded motion is also possible. At this point (after finishing the first CMC run) the motion manipulating test cases (cf. chapter 4) begin.
Finally, OpenSim offers GUIs to easily change the excitation patterns of the muscles to create different motions (as described above), the muscle parameters themselves or to create graphs from the motion data. It has a *perturbation tool* which can cause artificial perturbations to compute for example general accelerations for specified body segments.

### 3.2 Muscle Force Computation

OpenSim was developed to complement SIMM. Some functions and methods are the same but SIMM and OpenSim have some different focuses (like RRA and CMC). Beside these differences both can compute muscle forces, but they do it in totally different ways.

The *Inverse Dynamics* module computes the generalised forces (i.e. forces applied to each degree of freedom). Normally these forces are not as interesting as the muscle forces. Because most musculoskeletal models contain more muscles than generalised coordinates, there is no unique solution to this problem. To solve this problem, OpenSim uses an optimisation algorithm (cf. chapter 3.3.2). In SIMM the *Dynamic Pipeline Module* does not offer an optimisation algorithm for solving the equations of motion, but SIMM offers an other muscle force computation method to compute the muscle force, [13].

The muscle models used in SIMM and OpenSim are based on a Hill-type muscle model (cf. figure 3.1). The model represents muscle properties by an active contractile element (CE) in parallel with a passive elastic element. The model is defined by five known muscle constants for each muscle: pennation angle (α), tendon slack length (l_T^S), optimal fibre length (l_M^0), maximal fibre velocity (V_{MAX}^M) and maximal muscle force (F_M^0). Additionally, four curves are defined for each muscle. These curves are defined by listed control points which are interpolated by cubic splines, [13] (cf. figure 3.1). The pennation angle (α) is the angle between the muscle fibres and the tendon. The forces in muscle and tendon are normalised by the peak isometric muscle force (F_M^0). Muscle-fibre length (l_M) and tendon length (l_T) are normalised by optimal muscle fibre length (l_M^0). Tendon slack length (l_T^S) is the length at which tendons begin to transmit force when stretched. Velocities are normalised by the maximum contraction velocity of muscle (V_{MAX}^M).[6]

SIMM does not compute the muscle forces by solving equations of motion or static equilibrium equations, but by an iterative method, [12, 11]. The inputs are muscle-tendon length (l_{MT}) and muscle velocity (V_{MT}). Both can be computed for each time step from the generalised coordinates given by an inverse kinematic analysis from the motion capturing data. Due to the relation $F_T = F_M \cos \alpha$ an iterative method, [5], to calculate the individual muscle force can be used without solving the optimisation problem.

The iterative muscle force computation method is described as:

1. Choose initial values for $l_M$, $l_T$, based on $l_M^0$ and $l_{MT}$
2. Compute the muscle force

The muscle force $F_M$ is the sum of the active muscle force (excited by the nervous system) and the passive muscle force (independent of excitation). $F_M$ depends on muscle fibre
length and velocity. With the middle graph in figure 3.1 and with $l^M$ as input, $F^M_{\text{act}}$ and $F^M_{\text{pass}}$ can be calculated. This value has to be scaled by activation effects (cf. chapter 3.2.1) and modified by force-velocity effects (right graph in figure 3.1).

The summation of $F^M_{\text{act}}$ and $F^M_{\text{pass}}$ results in $F^M$.

3. Compute $F^T$

The muscle is in series with the tendon, which is represented by a nonlinear elastic element. With the left graph in figure 3.1 and with $l^T$ as input, $F^T$ can be calculated.

4. Check the convergence criterion

The convergence criterion $F^T = F^M \cos \alpha$ has to be checked. Normally the required accuracy is reached in less than five iterations.

5. Start the next iteration pass

Before the next iteration pass starts again at step two, $l^M$, $l^T$ are adjusted based on the slope in the left and middle graphs of figure 3.1, respectively.

3.2.1 Correlation Between Muscle Force and Excitation Level

In OpenSim CMC gets the motion as input and computes activation level, excitation level and muscle force. Therefore, the equation of motion (cf. equation 3.1.2) is extended and in more detail $\tau$ can be written as:

$$\tau = R(q) \cdot F^M + E(q, \dot{q})$$ (3.2.1)
\( \mathbf{R} \) is a matrix of muscle moment arms, \( \mathbf{F}_M \) is a vector of muscle forces, and \( \mathbf{E} \) is a vector of generalised forces that characterises the interactions with the environment.\(^2\)

In this thesis the term excitation is reserved for the amplitude of the control signal going to the muscle. It is analogous to the firing level of the motor neurons that innervate the muscle. The term activation refers to the activity level of the muscle fibres themselves.\(^3\) Excitation and activation levels are allowed to vary continuously between zero (no excitation and activation) and one (full excitation and activation).

The correlation between activation and excitation (the process by which muscle–fibre calcium concentration is modulated by motor unit action potentials) is modelled by a first-order equation relating the time rate of change of muscle activation (\( \dot{a} \)) to muscle activation (\( a \)) and excitation (\( u \)):\(^5, 13, 26\)

\[
\dot{a} = \begin{cases} 
(u - a) \cdot \left[ \frac{u}{\tau_{\text{act}}} + \frac{1 - u}{\tau_{\text{deact}}} \right] & u \geq a \\
\frac{u - a}{\tau_{\text{deact}}} & u < a 
\end{cases}
\]  

(3.2.2)

where \( \tau_{\text{act}} \) and \( \tau_{\text{deact}} \) are the time constants for activation and deactivation, respectively. \(^2\)

Constant activation will lead to an activation asymptotically approaching the same excitation value.

The correlation between activation level and muscle force finally is given by:

\[
\mathbf{F}_M = (a \cdot f_{lv}(l_m, \dot{l}_m) + F_{\text{pass}}(l_m)) \cdot \cos(\alpha)
\]  

(3.2.3)

where \( \mathbf{F}_M \) (vector) is the combination of the individual muscle forces \( F^M \) and \( f_{lv} \) are their active force-length-velocities, \(^2\).

In SIMM there is no direct link between muscle force and activation level (cf. \(^5\)). The connection is made via the muscle model. There are about ten different models available. A programming code of these models can be found in the derivs.c file. SIMM also differs between excitation and activation in a similar way as OpenSim, \(^13\).

### 3.3 Special Features

#### 3.3.1 Residual Reducing Algorithm

\( \text{RRA} \) gets as input the accelerations computed by \( \text{IK} \) and tries to fit these with accelerations produced through actuators at the generalised coordinates by solving an optimisation problem.

In the \( \text{RRA} \) mode, the model has no muscles to apply forces to the skeleton. Instead, the model has torque actuators at each joint to apply forces to the skeleton. For each degree of freedom exactly one actuator is needed. \( \text{RRA} \) is only intended for gait or similar motions, i.e., movements like walking and running where the model is displaced relative to the ground affected by ground reaction forces and torques. \(^7\). There are also six additional artificial actuators (three translational and three rotational) at the artificial joint between the pelvis and...
the ground. These actuators are called residual actuators. They are necessary to move the model in space (simulating gait). Otherwise there has to be some kind of contact conditions between the ground and the feet. They are also necessary to correct dynamical inconsistency due to experimental errors and modelling assumptions.

The forces and motions measured for a subject do not satisfy Newton’s Second Law \( F = m \cdot a \), [4]. Instead, equation 3.3.1 has to be considered, where the residual forces are included.

\[
F + F_{\text{residual}} = m \cdot a \quad (3.3.1)
\]

An experimental error can be that the markers to record the motion on the human body are not at the same position as in the virtual model, due to the fact that they are above the muscle and the skin and not directly fixed to the bones. Because they are on the muscle and the skin they are not really fixed. They can move slightly in relation to the joints and each other. Another possible error could be that the model is not complete, or not detailed enough (e.g. gait is simulated without the upper extremities). Also the assumption of the artificial ground actuators itself leads to dynamical inconsistency. Of course, numerical errors are also possible.

First, RRA computes the joint torques for the actuators needed to drive a dynamic subject-specific skeletal model to follow a prescribed motion (normally computed by IK) in small time steps. These actuator forces are calculated by solving a weighted minimisation problem. The weights are user specified. The objective function to be minimised is, [7]:

\[
\min_{\phi} \phi = \sum_{i=1}^{n\phi} \phi_i^2 + \sum_{j=1}^{nq} w_j (\ddot{q}_j^* - \ddot{q}_j) \quad (3.3.2)
\]

where \( \phi_i \) are the actuator controls being solved for and \( \ddot{q}_j \) are the model accelerations calculated by IK. The desired accelerations \( (\ddot{q}_j^*) \) are calculated by an iterative process (equation 3.3.2 and solving the equation of motion for the RRA model with actuators at the generalised joints and additional artificial actuators).

This objective function produces very stable and good results. There is also another, faster and more unstable possibility of an objective function. The objective function is then more simple with more additional constraint conditions which have to be fulfilled. The exact mechanism is described by D.G. Thelen et al, [26].

At the end of RRA, the average value for each residual actuator is computed. The main goal of RRA is to adjust the torso mass centre to correct any “leaning” of the model which it is doing due to inaccuracies in the representation of the torso geometry. This adjusting should lead to lower residual forces.

Additionally, the average value of \( F_y \) can be used to compute the recommended mass changes for all of the body segments. After adjusting, a second pass of the RRA is possible. In this pass the residuals \( (\ddot{q}_j^* - \ddot{q}_j) \) are weighted \( (w_j) \) more heavily to make the optimiser choose smaller values for the residuals when minimising the objective function. Additionally, minimum and maximum limits are placed on the residual values, [7].
3.3.2 Computing Muscle Control

There are mainly two ways of analysing muscle driven motions. On the one hand there are tracking approaches and on the other hand there are performance-based dynamic optimisation codes. Tracking algorithms determine muscle excitations that closely replicate an observed movement. In contrast to this, performance-based dynamic optimisation is capable of generating novel movement based on a quantifiable objective (e.g. minimisation of metabolic energy during gait). CMC falls in the category of tracking approaches.[25]

Although the tracking approach is not able to generate new motions by solving optimisation problems, it has an important role in current simulation environments. Normally it produces results faster. Therefore, it is often used to deliver initial estimates for large-scale musculoskeletal models, which are needed due to the high non-linearity of those. Furthermore, the tracking approach is a feasible, supportive, fast alternative in problems which do not necessarily need optimisation such as finding the underlying cause of a movement disorder or in treatment planning.

The CMC module computes muscle excitation at user-specified time intervals during a simulation, [7, 20, 32, 21]. These excitation levels represent actuators that will drive the generalised coordinates (e.g., joint angles) of a dynamic musculoskeletal model towards the desired kinematic trajectory. The working of CMC can be described with the following four steps:

1. Initial states for the model are computed
   The states comprise the generalised coordinates (joint angles), generalised speeds (joint angular velocities), plus any muscle states (e.g. muscle activation levels and fibre lengths). While the initial values of the generalised coordinates and speeds can be taken from the desired kinematics, the initial values of the muscle states are generally unknown. Because of the need of initial states computation the first few time steps are not valid and must not be interpreted together with the rest of the simulation.

2. Compute the desired accelerations with the following PD law
   The computation of a set of desired accelerations is done by using the following proportional-derivative (PD) control law:

   \[
   \ddot{q}^*(t + T) = \ddot{q}^{\text{exp}}(t + T)k_v \cdot [\dot{q}^{\text{exp}}(t) - \dot{q}(t)] + k_p [q(t)^{\text{exp}} - q(t)]
   \]

   where \(k_v\) and \(k_p\) are the feedback gains on the velocity and position errors, respectively. \(\ddot{q}^*\) is the set of desired accelerations which will drive the model coordinates, \(q\), toward the experimentally-derived coordinates, \(q^{\text{exp}}\). Because the forces that muscles apply to the body cannot change instantaneously, the desired accelerations are computed for any small time \(T\) in the future. For musculoskeletal models, \(T\) is typically chosen to be about 0.010 seconds. This time interval is short enough to allow adequate control, but long enough to allow muscle forces to change. If these desired accelerations are achieved, errors between the model coordinates and experimentally-derived coordinates will be driven to zero. To drive these errors to zero
in a critically damped fashion (i.e. without over-shooting or over-damping), the velocity gains can be chosen from the relation:

\[ k_v = 2\sqrt{k_p} \]

For musculoskeletal models, it works well when the error gains are chosen to drive any errors slowly to zero. The error gains \( k_v = 20 \) and \( k_p = 100 \) will cut down tracking errors.

3. Static optimisation

This step computes the actuator controls, \( \mathbf{x} \), that will achieve the desired accelerations, \( \ddot{\mathbf{q}}^* (t + T) \), and distributes the load across the actuators. It is called static optimisation because the performance criterion (i.e. the cost index) is confined to quantities that can be computed at any instant in time during a simulation. Using criteria as for instance jump height or total metabolic energy over a gait cycle, are not possible because these require simulating until the body leaves the ground or until the end of the gait cycle is reached. The static optimisation (see chapter 2.2.2 for optimisation basics) problem in CMC is solved in the same way as in RRA (chapter 3.3.1, equation 3.3.2). The usual performance criterion is the minimisation of the activation levels.

The static optimisation step provides an activation level, an excitation level and the muscle force for each muscle. The correlation between muscle force, muscle excitation level and activation level is described in chapter 3.2.1.

4. Forward dynamic simulation

The fourth and last step is a forward dynamic simulation for a small, user-defined time step, so that the CMC routine can advance to the next time step and begin with step two again. Once CMC finishes execution, one can compare the computed muscle excitation patterns with prototypical or electromyographical measurements.

The excitation patterns can easily be modified with a graphical editor or by editing the contraction-time curves in a table directly in a control file where the excitation patterns are saved.
4 Results

4.1 Motion Laboratory results

The spatial coordinate system used in this thesis to describe the human movement is shown in fig. 4.1. The direction of progression (anterior - posterior) is X, the vertical direction is Y and the sideways direction (medial - lateral) is Z [31]. The origin is defined as the centre of mass (CoM) for the global coordinate system.

The names of the muscles are given in this thesis in a short form, i.e. without the leading “musculus” respectively “musculi”, e.g., “Musculus Quadriceps Femoris” is only called “Quadriceps Femoris”.

The marker positions are given in figure A.1. The exact position are described in table A.1, where “ASIS” means “Anterior Superior Iliac Spine” and “PSIS” the “Posterior Superior Iliac Spine” respectively.

In this experiment four different motions were recorded. The first motion is a forward-backward swing of the leg (cf. A.2 - A.7). The second motion is a sideward swing of the leg (cf. A.8 - A.13). The third motion is a leg turning (cf. A.14 - A.19) and the fourth is a swing movement just of the lower leg (cf. A.20 - A.25). The calculation of the angles is shown in list A.1 by means of the hip flexion/extension calculation in MATLAB.

In a perfect world each motion would show a significant curve just in its associated graph. The other curves would be always zero. However, the experiments are not perfect, due to several conditions. The markers are placed on soft tissue (skin, muscle); therefore, they can move slightly in respect to each other. This is not possible in the model, because the model is build under the assumption that the markers are fixed on the bones which are rigid links between the joints.

The proband is human. His motions are not perfect. It is not easily possible to move a part of the body just in exact one reference plane.

The markers are placed and positioned by humans, so they could be placed wrong. The mechanical and also the mathematical model may differ from the reality. The joint centres do not suit the real joint centres at all times and therefore the joint model axes do not exactly represent the real joint axes. Numerical errors are also possible, but in this case probably so small that they can be neglected.

All this has to be considered, when motions in a simulation environment like OpenSim are imported, analysed and used for further experiments and testing.
Results

The forward-backward swing of the leg is a hip flexion/extension movement. The main character of this motion is shown in figure A.4. The knee remains during this motion nearly straight (cf. A.3). Also the rotation of the hip (A.2) can be seen as zero. During the backward swing an additional abduction (cf. A.6, A.7) for the whole leg is recorded. Also a slight rotation of the lower leg can be seen in figure A.4. These results seem reasonable as a healthy human movement.

The sideward swing of the leg is a hip ab-/adduction movement. The main character of this motion is shown in figure A.12. The recorded motions in figures A.8, A.10, A.9 and A.11 can be seen as constant. In figure A.13 an adduction of the lower leg near the maximum abduction of the hip can be seen. This is probably caused by an additional adduction of the foot. This also an reasonable result. This motion is called supination.

The leg turning movement is a rotation of the leg. The main character of this motion is shown in figures A.14 and A.14. It can be seen that the external rotation of the foot is greater than the rotation of the knee. The motions visible in the other planes are nearly constant. This motion shows a very clear result without bigger interference.

The swing movement of the lower leg is similar to the forward-backward swing of the whole leg. The main character of this motion is shown in figure A.23. The motions shown in figures A.20, A.22, A.24 and A.25 are nearly constant. In figure A.21 a rotational movement of the lower leg is shown. This is not usual for this kind of motion. It is probably cause by the unstable standing position of the proband and the unnatural movement.
4.2 Manipulate Motions

The main scope of the tests in this project is to manipulate and handle motions. There are different possibilities to reach this goal. Altering the motion directly by scaling or creating new artificial motions by prescribing degrees of freedom and computing the needed muscle forces and excitation is one possible way. Another possibility is to change or prescribe excitation patterns (and muscle forces respectively) and compute a new motion by means of forward dynamic simulations. Taking the long jump as an example, this leads to augmenting the step length right before the jump, augmenting the jump length itself, altering the jump trajectory and computing the needed muscle forces. Varying the used muscles and computing the resulting jump refers to the second possibility. These variations are tested in this thesis. Also increasing the running velocity or muscle strength are possible alternatives, which are though not covered in this project.

The following sections describe the test cases and results of the OpenSim testing. The main test case is based on a test case of the human healthy gait (gait2354) which is included in the software package. This is the first choice, because then there is no problem in the model setup and no disturbances by other parameters than the test parameters. In all test cases, the example files are used to load the model, scale it, run IK, run ID, use RRA and finally use CMC. After creating the motion file and the other motion data like the excitation patterns through CMC, different tests are made. The observed and used part of the test case gait2354 includes motion data for nearly the whole right stance phase (and therefore a part of the left swing phase, the period of double limb support is not included) of an average human healthy gait cycle.

All the graphs show the normalised time on the x-axis, i.e. zero is the start of the simulation and one is the end. If a simulation did not run stable, the curve finishes before one.

The motion graphs show the flexion/extension of the related degree of freedom (joint) on the y-axis. There is no general convention for the algebraic sign (positive/negative) for flexion or extension. Therefore, the direction of flexion or extension is specified separately in each graph.

Because of the width and thickness of some muscles, it is not easy to define one discrete contact point. Therefore, some muscles (e.g. Gluteus Maximus) are split into several parts, with different contact points to represent one big real muscle. The CMC computed muscle activation is normalised for each muscle in the model.

The excitation graphs show the summation of important muscles in the model for the respective motion on the y-axis, so that the maximum is higher than one. The y-axis scaling is therefore not comparable among different muscle groups, because the maximum depends on the number of muscles in the summation. It is possible to connect the normalised excitation directly to the muscle force. This is due to how CMC works (cf. chapter 3.3.2) and in what way muscle forces are computed (cf. chapter 3.2) in OpenSim.
4.2.1 Musculoskeletal Geometry

The computer model gait2354 represents a normal, adult male and the anthropometric data is based on parameters that were compiled in several different experimental studies. To acquire the bone surface data, the surfaces of the bones were marked with a mesh of polygons and then the coordinates of the vertices were determined with a three-dimensional digitizer (Polhemus Navigation Sciences, Colchester, Vermont). Based on the anatomical landmarks of these bone surface models, the paths (i.e., the lines of action) of forty-three muscle-tendon actuators were defined. Origin and insertion are necessary landmarks and, in some cases, are sufficient for describing the muscle path. In other cases, where the muscle wraps over bone or is constrained by retinacula, intermediate “via points” were introduced to represent the muscle path more accurately. By visually comparing the muscle paths with paths defined by a commonly used set of muscle coordinates the anatomical accuracy was verified. The muscle parameters can be found in table A.2. The muscle path coordinates can be found in Appendix A of [3].

The lower extremity is modelled as seven rigid body elements: pelvis, femur, patella, tibia/fibula, talus, foot (comprising the calcaneus, navicular, cuboid, cuneiforms and metatarsals) and toes (phalanges), with reference frames fixed in each segment. Figures 4.2 and 4.3 show the joint models and the location of the body-segmental reference frames. The annotation can be found in table 4.1. The relative motion of these segments is defined by models of the hip, knee, ankle, subtalar and metatarsophalangeal joints. A detailed description of these joints can be found in Appendix B of [3].
Figure 4.3: Location of the body-segmental reference frames, the foot [3]

Table 4.1: Location of the body-segmental reference frames [3]

<table>
<thead>
<tr>
<th>Reference Frame</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pelvis</td>
<td>The pelvic reference frame is fixed at the midpoint of the line connecting the two anterior superior iliac spines.</td>
</tr>
<tr>
<td>Femur</td>
<td>The femoral frame is fixed at the center of the femoral head.</td>
</tr>
<tr>
<td>Tibia</td>
<td>The tibial frame is located at the midpoint of the line between the medial and lateral femoral epicondyles.</td>
</tr>
<tr>
<td>Patella</td>
<td>The patellar frame is located at the most distal point of the patella.</td>
</tr>
<tr>
<td>Talar</td>
<td>The talar frame is located at the midpoint of the line between the apices of the medial and lateral malleoli.</td>
</tr>
<tr>
<td>Calcaneus</td>
<td>The calcaneal frame is located at the most inferior, lateral point on the posterior surface of the calcaneus.</td>
</tr>
<tr>
<td>Toe</td>
<td>The toe frame is located at the base of the second metatarsal.</td>
</tr>
</tbody>
</table>
A possibility to alter or manipulate motions is to change the excitation patterns. The excitation patterns can be shown with the “excitation editor" GUI in OpenSim for each muscle. It is possible to alter the excitation by using this GUI and editing the excitation curves graphically. It is also possible to change the excitation values in an XML file by editing the tabulated excitation data on which the interpolated curves in the GUI are based. After changing the excitation patterns, a new forward dynamic simulation is made, so that the original motion and the motion based on the altered excitation patterns can be compared.

In this test case the knee flexion during the observed gait cycle interval should be changed. The left leg is in the swing phase and at the beginning of the motion the knee is flexed (∼60°). During the motion the knee would be straightened (∼0°) so that the stance phase could begin (after the here simulated part of the gait cycle).

Assuming that the patient could not straighten the knee because of too weak functional muscles for this movement, the following simulation is made. From the medical point of view the result then indicates which muscles are mainly used and cause the pathological dysfunction, so that an optimal treatment plan can be simulated, e.g. should the doctor prefer a surgery to increase the moment arms of the weak muscles or a pharmacological strengthening of these muscles.

To simulate that the left knee could not be straightened, the excitation of two muscle groups has to be changed. The knee flexors (in the model represented through Biceps Femoris long head, Biceps Femoris short head, Gracilis, Gastrocnemius and Sartorius) excitation is changed to nearly constantly active (the muscles are contracted) so that this muscle group produces forces to bend the knee or hold the knee in a bent position. The knee extensors (in the model represented through Rectus Femoris and Vastus Intermedius) excitation, which straightens the knee in the unmodified motion, is changed to nearly constant passive, i.e., these muscles do not try to straighten the knee. Figure 4.4 shows a comparison between CMC calculated excitation of the left knee flexors and the artificially changed excitation. Figure 4.5 shows the same comparison for the knee extensors, respectively.

In figure 4.6 the original motion and the motion based on the changed excitation patterns are shown. This shows that the knee remains more flexed, because of the reduction of the knee extensors excitation and the increasing of the knee flexors excitation.

The result indicates that a pharmacological treatment to relax the knee flexors is a good idea to start with, but not sufficient. The knee extensors have to be strengthened, too, either through a pharmacological or a surgical treatment.

Another test case, in which only the hip extensors (Psoas and Iliacus) excitation is changed (nearly constant low, i.e. no extension force produced here) shows that this change leads from an original 5° flexed knee in the unmodified motion to a nearly 20° extended knee in the modified motion (cf. figures A.26, A.27). This shows that changing the excitation patterns of a muscle group (here the hip extensors) affect the whole model and can lead to a complete new motion.
Figure 4.4: Comparison of original and changed excitations of left knee flexors for the test case described in chapter 4.2.2

Figure 4.5: Comparison of original and changed excitations of left knee extensors for the test case described in chapter 4.2.2
Results

Figure 4.6: Left Knee Flexion/Extension resulting from original and changed excitations shown in figures 4.4, 4.5 and described in chapter 4.2.2

4.2.3 Scale Motions

In this test case the scope is to scale a motion and then compute the needed muscle excitation patterns using CMC. The motion scaling is done by scaling the step length through scaling the angle of the degree of freedom of the hip flexion/extension by five percent (cf. figure 4.7). It is done with a Perl-Skript (A.2).

Scaling a motion is interesting for example from the athletic point of view. Running faster normally means to run with an increased step length. To compute which muscles are necessary to reach this bigger step length CMC can be used after the scaling.

As in the test case above the motion and the muscle excitations are given (cf. chapter 4.2). The motion is stored in a motion file in a table – for each time step the values of each degree of freedom.

After scaling the motion, CMC is used and the computed muscle excitations before and after the motion scaling are compared. The actuators (muscles) have a higher normalised excitation level in certain time periods of the simulation cycle. The main flexor, Gluteus Maximus, has a higher contraction during the first part of the observed part of the gait cycle (flexion) (cf. figure 4.8) and the main extensors (Psoas and Iliacus) have higher contraction during the second part (extension) (cf. figure 4.9). It is also noticeable that for example the knee extensor (Quadriceps Femoris, in this model represented through Rectus Femoris and Vastus Intermedius) or the muscle group for the plantarflexion (in this model represented through Gastrocnemius, Soleus and Tibialis Posterior) have nearly the same excitations after the scaling as before (cf. figures A.28, A.29).
This result shows that the typical used muscles have to produce a higher muscle force. For the athlete does this mean that he has to train the same muscles even harder and not to change or extend the training to other muscles.

4.2.4 Create New Motions

Another possibility to get muscle excitation patterns is through creating a motion file by formulating values for each degree of freedom. In this case the motion wanted is a backward kick where the final velocity of the foot is in negative x-direction.

The medical interest in creating new motions is for example to simulate motions for which no motion laboratory data is available. The backward kick is just a semi-complex example to proof that this kind of simulation is also possible with OpenSim.

The motion is divided into two parts. First the knee is bent from 0° to 90°. The alteration of the knee angle is linear over time. During the second part the knee is extended while the hip is also extended. Under the assumption that the distances between hip and knee joints and knee and ankle joints are the same, the following relation between hip extension angle $\psi_{hip}$ and knee extension angle $\psi_{knee}$ can be found, so that the movement of the foot in the sagittal plane is just in negative x-direction:

$$\psi_{knee} = \arccos(1 - \cos(\psi_{hip})) - \psi_{hip} \quad (4.2.1)$$

Figure 4.10 shows that relation.

The origin of the coordinate system is in this case chosen at the hip joint centre. The coordinate
Results

Figure 4.8: Comparison of CMC calculated excitation of right Gluteus Maximus before and after scaling the motion, figure 4.7, and described in chapter 4.2.3

Figure 4.9: Comparison of CMC calculated excitation of right Psoas and Iliacus before and after scaling the motion, figure 4.7, and described in chapter 4.2.3
values are normalised by the length of the thigh (or shank, cf. assumptions above). The trajectories of the second part of the motion (the kick) are shown in figure 4.11.

Due to the complexity of human movements, the used example model and the instability of $CMC$, the desired kick movements are applied to the left leg during the normal gait motion, i.e. the example gait motion is used and only the values of the needed degrees of freedom are altered (hip angle and knee angle). The right leg further performs the gait movement. Also the ground reaction forces stay the same. These together are a lot of assumptions. Because the simulations can not be verified with test data, the results should be considered carefully.

Because of the instability of $CMC$, only a very small time interval of the second part of the desired kick motion can be simulated. Nevertheless, the resulted muscle force shows the most important muscles and in which ratio they have to be activated.

Due to the fact that some muscles have more than one exact function, e.g. Rectus Femoris can extend the knee and flex the hip, their value is divided equally between their functions. The main knee flexors are here: Gastrocnemius, Biceps Femoris short head and Biceps Femoris long head, the main knee extensors are Vastus Intermedius and Rectus Femoris, the main hip flexors are Psoas Major, Iliacus, Rectus Femoris, the main hip extensors are Gluteus Maximus, Biceps Femoris long head and the posterior part of the Gluteus Medius. The muscle force is normalised by the maximal occurring force during each motion part, respectively. The time is normalised by the simulation time (cf. figures A.30, A.31).

In figures 4.12 and 4.13, the knee and the hip flexor and extensor forces are summed, respectively. Thereby only the resulting, dominant flexion or extension is shown. During the knee bending the dominant muscle force is produced by the knee flexors. The hip flexion/extension is insignificant. In figure A.30 however is observable that both the hip flexion and extension forces increase while
Results

Figure 4.11: Position of the left foot and knee joint during the kick motion described in chapter 4.2.4

the knee is bent increasingly.

During the kick movement the dominant knee flexor force decreases while the hip extensor force increases. This is exactly as expected by observing the prescribed values for the knee and hip angle.

Medically these results show that for a simple motion (like bending the knee) the number of used muscles is small and the result is very clear. The observation of a more complex motion (like the backward kick) leads to very many different muscle activities which have to be evaluated by an experienced medical professional. It is not easy to see which muscle is used for which part of the motion. Also the entire muscle forces are increasing with the increase of complexity of the motion. There are probably more “fighting forces” which can lead the model (or the proband respectively) to go out of balance.

4.3 Compatibility to SIMM (Import/Export)

OpenSim offers the possibility to import SIMM models directly. During the import process the user can choose which underlying multi-body engine should be used. Due to compatibility problems with SIMM models and the OpenSim multi-body engine SimBody, OpenSim offers the possibility to choose the so called SIMM kinematics engine as an underlying multi-body engine. When an OpenSim model is built out of a SIMM model with the SIMM kinematics engine, it can not be used for most of the dynamic simulation tools offered by OpenSim, but it can be viewed.

The problem of two multi-body engines available and the problem of handling all SIMM models
Figure 4.12: Summed main flexors/extensors muscle forces of the hip/knee during knee bending described in chapter 4.2.4, the separate muscle forces are shown in figure A.30

Figure 4.13: Summed main flexors/extensors muscle forces of the hip/knee during the kick movement described in chapter 4.2.4, the separate muscle forces are shown in figure A.31
Results

with SimBody is still present in version 1.5.5 of OpenSim. There is an unofficial announcement made in August 2008 that the problem should be solved with version 1.6. Then, SimBody should handle all SIMM models and the SIMM kinematics engine should be obsolete for the usage in OpenSim.

OpenSim offers a function to export OpenSim models to SIMM. This is just possible for models based on the SIMM kinematics engine. Models based on the SimBody engine can not be easily exported to SIMM.

4.4 Conclusions

Testing results, except for the ones mentioned above, are summarised consecutively.

The test case (gait2354) simulates only a small part of the whole gait cycle (a part of the right stance phase). The motion and ground reaction force data are however recorded and thus available for several gait cycles. Within testing and trying to run the simulations for a longer time period, it emerged that some modules in OpenSim are still very unstable.

RRA and CMC very often simply break down without a traceable reason. According to other user statements on the users’ forum this is a very common problem. This also seems to be a platform and resource independent problem. The functions do not work without reasonable ground reaction forces either. The official statement is that the problem is recognised, but a solution is not expected soon.

Forward Dynamics breaks down when the initial state file does not offer values for each time step. The official statement is that it should run with just one initial state, but during the testing in this project it did not. Other users reported similar problems. There has not been made a further official statement yet.

The test case in which excitation patterns are changed shows that it is possible to change motions by altering the excitation patterns and finding a new motion by using the forward dynamics module. It is also noticeable that anatomically incorrect results could be produced. In the test case, where the excitation of the Iliopsoas is reduced to nearly zero (cf. figure A.27), the knee is finally overextended by up to 20% which normally can not occur (cf. figure A.26) without deformation of the bones or injuries of the soft tissues (muscles, tendons, et cetera).

The second test case in which the step length is scaled shows that it is also possible to create new motions through scaling. During the testing it emerged that the step length could be scaled up to plus 30% before the CMC module became unstable and did not produce any result.

The third test case in which a new motion was created, again showed the instability of CMC. It also showed that a stable run leads to analyseable results.
5 Extension – Comparison v1.5.5 vs. v1.8.1

This chapter contains additional test cases and testing results, which are based on my work at the Institut für Statik und Dynamik der Luft- und Raumfahrtkonstruktionenm (ISD) at the University of Stuttgart from April till June 2009.

I thank Professor J.F. Wagner for the support at the University of Stuttgart and Professor Anders Eriksson and Lanie Gutierrez-Farewik for the further support and guidance via email from Stockholm.

In March 2009 the new release v1.8.1 of OpenSim was published. The following chapter mainly compares the two versions of OpenSim, v1.5.5 and v1.8.1.

Because of the importance of optimisation, the algorithms supported by OpenSim, are introduced. The compatibility to SIMM is tested and the User’s Guides are compared. The test cases from chapter 4 are repeated and the results are summarised. One additional test case about a motion without ground contact is introduced and discussed. The chapter ends finally with a project proposal for further investigation.

The chapter 6 is extended with a personal conclusion of the author to the new version 1.8.1.

5.1 Optimisers

An essential part of RRA and CMC is the numerical optimiser. OpenSim uses for RRA the lapack algorithm by solving it without bounds on the controls and then checking if the solved controls fell within the required bounds. Most of the time this is the case, and if it is not, RRA falls back on IPOPT.[16]

For CMC OpenSim uses regularly the IPOPT (Interior Point OPTimizer). IPOPT is a software library for large scale nonlinear optimisation of continuous systems. It is written in Fortran and C and is released under the CPL (common public licence), so that is free available. IPOPT is designed to exploit 1st and 2nd derivative (Hessians) information if provided. If no Hessians are provided, IPOPT will approximate them using a quasi-Newton methods, specifically a BFGS (Broyden-Fletcher-Goldfarb-Shanno method) update.[30]

OpenSim also supports the CFSQP algorithm. It is a feasible sequential quadratic programming optimiser developed at the University of Maryland. CFSQP is not free, but if the user has a licence for it separately it can be used together with OpenSim. CFSQP is the best choice (if it is available), because it is the fastest of the tested algorithms and should give a 2x speedup compared to IPOPT.[2][17]
5.2 Compatibility to SIMM

One of the major problems in version 1.5.5 was that most of the SIMM models could not be imported. In version 1.8.1 the multi body engine SimBody, which is integrated in the SimTKCore project, has been updated. The SIMM Kinematics Engine is now obsolete and SimBody supports constraints and welds. As a result, the support for SDFast is dropped.

In general it is now possible to import all SIMM models. In the test cases there occurred no problems.

5.3 Comparison of the User’s Guides

The tutorials are the same in both User’s Guides as the whole Guides itself. There is one additional chapter about the marker editor, because this is a new function in version 1.8.1.

The User’s Guide is still not up to date because the RRA is not described precisely enough. RRA1 and RRA2 are described as two runs but in the example task file and the GUI there is no possibility to differ between those two runs (cf. chapter 5.4.1 and chapter 6.2).

5.4 Stability

The tests were made on a Windows 64-bit system with an Intel Core 2Duo 2.5 GHz processor and 4 GB RAM.

5.4.1 RRA

In chapter 3.3.1 it is written that normally the RRA consists of more than one run and two runs are enough to reach a certain accuracy. In the User’s Guide ([7]) it is written that the RRA always consists of two runs, but this part of the User’s Guide seems to be obsolete, because in the user interface of the recent version there is no possibility to differ between the two different RRA run versions. Therefore it is not obvious if the RRA algorithm has been changed and the User’s Guide is not up to date or if the user interface is not able to handle all possible variations of all the parameters of RRA1 and RRA2 as described in the User’s Guide.

With the version v1.8.1 the RRA computes the states for the whole gait data (15 seconds) with the fast target option in about 40 minutes. The slow target option does not work and the RRA does not finish.

In OpenSim project wiki ([15]) there is an official statement, that for RRA the slow target option and for CMC the fast target option has to be used. It is not obvious to what this advice should lead. It is not conform with the User’s Guide which says that for RRA and CMC the fast and the slow target option are available.

During the tests there often appeared a warning: \textit{WARN- a desired points file was not specified.}
This warning does not affect the stability in the test cases. Because it occurs even in the standard cases there is probably a problem with this warning message and not necessarily with the data files or the algorithms.

### 5.4.2 CMC

*CMC* now works stable for longer time periods. The use of the slow target option really leads to greater stability. This stability gain is paid by computing speed (cf. chapter 5.5.1).

An advantage in v1.8.1 is that OpenSim does not break completely down if *CMC* fails. Now there is a error message in the message window and the user can change some parameters and start *CMC* again.

A problem that can occur with *CMC* is that the muscles do not relax as they would in reality. This can lead to an increase of force in the model. The official statement is: “It’s also known that the *CMC* optimiser is more likely to activate an antagonist muscle rather than deactivate a muscle (because deactivation time is longer).”[19]

### 5.4.3 Forward Dynamics

Similar to *CMC* the *Forward Dynamics* module does force OpenSim to completely break down anymore if it fails and produces an error message instead.

The *Forward Dynamics* sometimes produces unreasonable result if the simulated time is too long. This is different to *CMC* because it works “in the loop” (cf. chapter 3.3.2) and the *Forward Dynamics* does not.

The usage of *Forward Dynamics* still causes OpenSim sometimes to completely break down without obvious reasons.

There is an unofficial announcement in the users' forum that the next release should be more robust.

### 5.5 Comparison of the Test Cases

#### 5.5.1 Change Excitation Patterns

*CMC* does not compute the test cases stable with the use of the fast target option, but with the slow target option. The run with the fast target option fails with the following error message

```
SimTK Exception thrown at InteriorPointOptimizer.cpp:249: Optimizer failed: Ipopt:
Restoration failed (status -2). OPTIMIZATION FAILED...
```

It is not known, what this message means in relation to OpenSim and the *CMC* algorithm, because there is no reference guide for the error numbers or anything similar.
The computation of the muscle force distribution for the first five seconds of the normal gait takes about 15 hours.

In figure 5.1 the values of the generalised coordinates at the left and right knee and hip joint respectively are shown. This motion is periodical and contains four complete gait cycles. Because of the more complex motion it is not as easy as in chapter 4.2.2 to interpret the motion from the summed main knee and hip flexors and extensors respectively. Figures A.32 and A.33 show a high noise and some unrealistic peaks. With a closer look at some specific muscles, in this case the Gastrocnemius for the knee flexion (cf. figure 5.2) and the Vastus Intermedius for the knee extension (5.3), the periodic motion and the relation to the whole motion can be seen. The noise and the unrealistic peaks has to be investigated and the CMC results altogether should be validated by comparison with experimental data.

The next step in chapter 4.2.2 is to slightly change the excitation and then run a forward dynamics simulation. Forward Dynamics is though not able to compute a reasonable motion from the CMC output with unchanged muscle excitations for a whole gait cycle. It is possible to compute the Forward Dynamics for nearly half of a gait cycle. In the following example computation the results become unreasonable during the change from the swing phase of one leg to the stance phase. This can be seen exemplified in figure 5.4 with the angle values for the subtalar and the ankle joint. The results seem reasonable up to second 1.1 and after that the simulation fails. Absolute values up to 1000 degrees (i.e. the foot revolts nearly three times around the ankle, subtalar respectively, axis) are reached and after second 1.5 the simulation breaks down.
Figure 5.2: CMC calculated excitation of the Gastrocnemius

Figure 5.3: CMC calculated excitation of the Vastus Intermedius
Within this test case there was no way to produce reasonable and stable *Forward Dynamics* results for longer time periods than in chapter 4.2.2. Under this conditions another change of the excitation patterns is not expedient.

After second 1.7 no stable run of *Forward Dynamics* was possible. All this shows that there is either a problem with the *Forward Dynamics* or the CMC results are not correct.

### 5.5.2 Scale Motions

In chapter 4.2.3 the angle of the degree of freedom of the hip flexion/extension is scaled by five percent. In this test case the same scaling is done for a longer period of time (cf. figure 5.5).

In figures 5.6 and 5.7 the computed muscle excitations before and after the motion scaling are compared.

The results seem reasonable and are very similar to those in chapter 4.2.3. The scaled excitations seem a little bit smoothed compared against the unchanged excitations, this can especially be seen in the figure for the excitations of the Gluteus Maximus.

The results for excitations which should not affected by the scaling seem as reasonable as in chapter 4.2.3, cf. figures A.35.

The excitation of the right Quadriceps Femoris has not changed, as expected, but the excitation of the left Quadriceps Femoris has changed slightly after the scaling (cf. figure A.34). This is not a result as expected because the change of the motion is for the left side and the right side equal. This can not be explained easily. It has to be evaluated if this computation is possibly...
Figure 5.5: Flexion/Extension of the right hip joint; original and scaled as described in chapters 4.2.3 and 5.5.2.

Figure 5.6: Comparison of CMC calculated excitation of right Gluteus Maximus for a longer period of time before and after scaling the motion, figure 5.5, and described in chapters 4.2.3 and 5.5.2.
5.5.3 Create New Motions

Creating new motions is still a point of interest. The investigation of the kick motion is satisfactory done in chapter 4.2.4, so that test case is not repeated and instead a new test case is introduced in chapter 5.6.

At the Department of Mechanics at the KTH, Stockholm, there is currently a project to investigate jump motions and muscle force distributions during jumps with a performance based algorithm. Therefore this test case is chosen instead to the previous kick motion.

5.6 Movements Without Ground Reactions

This test case is made to test the stability and accuracy of CMC for movements without ground contact, like jumps. This is a special case, because OpenSim does not use contact conditions between the subject and the ground. OpenSim uses residual forces normally applied to the pelvis. If there are no ground reactions, there are also no residual forces and RRA cannot be applied.

The test case is the normal gait example (2354) without the ground reaction forces. This motion is then similar to walking through the air, like the leg movements of an athlete doing a long jump.
The first step is to run the Inverse Dynamics without ground reaction forces to get the correct forces and torques at the generalised coordinates. The next step normally is the RRA but because of the lack of ground reactions there are no residual forces applied to the pelvis and therefore there is no need to run it. Now CMC computes the muscle forces.

The fast target always fails so that the slow target option has to be used. The run with the slow target option finishes and produces results. It seems that the subject is falling. It moves freely and not along the prescribed coordinates recorded in the motion lab and computed with Inverse Kinematics. Also the other degrees of freedom are not even similar to their prescribed values. In figure A.36 are the y coordinate of the pelvis and the progress of the values for the right hip flexion and the lumbar extension shown exemplified. The original motion data (before running CMC) can be seen in figure A.37. It is obvious that these results are not similar to the reality. The main question here is: Why does CMC compute a new motion, where it actually should compute muscle force distributions?

5.7 Project Proposal – CMC and NMT, Two Muscle Force Computation Approaches

The CMC algorithm of OpenSim is a new and progressive algorithm to compute muscle force distributions. It has his strengths and weaknesses. A good opportunity to really test its limits is to compare it to an other algorithm. The suggestion here is the NMT algorithm, because it seems to fit in the gaps that CMC leaves.

5.7.1 Introduction

OpenSim is a freely available software package which offers the possibilities to build, exchange and analyse computer models of the musculoskeletal system and run dynamic simulations of movements.[4] Especially it offers a module to compute muscle forces and excitations for given motions. This module is called CMC (Computing Muscle Control).

In general there are two ways of analysing muscle driven motions. On the one hand there are tracking approaches and on the other hand there are performance-based dynamic optimisation codes. Tracking algorithms determine muscle excitations that closely replicate an observed movement. In contrast to this, performance-based dynamic optimisation is capable of generating novel movement based on a quantifiable objective (e.g. minimisation of metabolic energy during gait). CMC falls in the category of tracking approaches.[25] A recent example for a fast performance-based algorithm is the NMT (neuromusculoskeletal tracking) method.[22]

In this project it should be demonstrated that the use of simulations can improve treatment outcomes for individuals with movement disorders. For that purpose the NMT method should be integrated into OpenSim and compared to the CMC module.

Several case studies should be made to determine if general principles for treatment planning can be elucidated from the insights gained from analysing simulations. Studies that retrospec-
tively compare predictions from subject-specific simulations to the subject’s actual outcomes should also be made to evaluate whether the existing musculoskeletal models are sufficiently accurate, and to establish the conditions under which the results of simulations are applicable and which method (NMT, CMC) should be best used or if a combination of both is necessary and possible.[4]

5.7.2 Methods that Could Be Used

The determination of finding general principles for treatment planning requires many case studies. In this project three different cases should be examined and compared. The case study used for the validation for CMC[26], pedalling (ground contact during the whole simulation), should be tested with NMT, the case study coming from the validation for NMT[22], height-jump (long time without ground contact), should be tested with CMC even if RRA (Residual Reducing Algorithm) cannot be applied, and a third case, healthy gait (ground reaction forces changing strongly over the simulated cycle) should be applied to both algorithms.

5.7.3 Significance of this Study

Even in a time where the computing power increases rapidly and more and faster supercomputers become available the efficiency of algorithms is still very important for highly complex models in human movement simulations. This study should compare two of the most recent muscle force computation approaches integrated in an open-source software package and make suggestions for what cases and under which conditions they should be applied best. It should result in a fast, efficient, accurate and stable way to compute muscle forces even in very complex musculoskeletal models. It also should be applicable to all kind of motions (with and without ground reaction conditions).
6 Experiences and Future Work

In this chapter the author’s opinion about the testing and the whole project is described.

6.1 Version 1.5.5

This chapter contains conclusions referring to chapter 4 and the OpenSim version 1.5.5.

All three different test cases could be handled by OpenSim. The stability was different and therefore the way to work with OpenSim, too. Nevertheless, the results seemed to be reasonable all the times and applicable to real cases as far as this can be evaluated without the comparison to experiments.

Working in a motion lab requires good anatomical and technical knowledge. If the experimental data is not collected precisely, the following simulation and simulation results are not feasible.

The testing experiences show that changing excitation patterns (cf. chapter 4.2.2) is not a generally applicable procedure to create more complex movements. The user has to be very experienced in anatomy and human movements to change the right muscle excitations; otherwise the computed movement will not be realistic or predictable at all. Due to the complexity and instability of human movement, just very small changes are possible. Manipulating excitation patterns can be useful to slightly vary known motions or check in which direction a forward simulation will lead, after a tendon transfer or other surgeries.

Scaling motions (cf. chapter 4.2.3) or producing a wholly new motion (cf. chapter 4.2.4) and after that computing the needed muscle forces is an applicable way to analyse motions. The problem is to obtain the reasonable ground reaction forces, which are needed for CMC to run. It is thus not possible to use this analysis method to produce reasonable results without great experience or preceding testing.

The choice of multi-body mechanics to analyse and simulate the human gait is very often used and in most cases suitable. Developing an open source software (like OpenSim) is a very ambitious and great project which can only succeed, if the software is accepted and used by the scientific community. It is necessary for the developers that the user community reports bugs, problems with actual modules and makes suggestions for useful new tools or modules. It seems that the OpenSim developers try to reach these goals in different ways. At Stanford (where the main developer team works) beta versions and release candidates are tested extensively. The OpenSim users’ forum includes a feedback and bug-report section on the SimTK homepage, which is the easiest way to communicate with OpenSim users worldwide. The quality of such bulletin boards depends highly on the community itself and not only on the moderators. In the
Experiences and Future Work

users' forum, some questions are answered within two days and some never. Because the user does not have the advantage of a real support team (like in the commercial software package SIMM) he can try to get help from the community, but should never rely on it and always be prepared for finding a solution to the problem on his own.

On 21 August 2008 an OpenSim User's Guide was published, [7]. In this manual, the basic functions of OpenSim and its underlying mathematics are described. The most of the documentation is very useful and understandable. In chapter 16 of the User’s Guide the RRA is described (chapter 17 for CMC respectively). As mentioned in chapter 3.3.1 the documentation is not sufficient to understand the basic principles of those modules. The reference quoted have to be consulted.

The modules IK and ID are working stably and produce comprehensible results. The documentation is sufficiently described in detail in the User's Guide. Those modules are necessary basic features for a biomechanical simulation software. OpenSim has a stable basis.

The modules RRA and CMC are working very unstably, as described in chapter 4.4. The crash report files contain no relevant information, so that the user can not fix the problem on his own and has to wait for an official statement or update. This is a major problem, because these two modules are unique features of OpenSim and should distinguish this software from others like SIMM. OpenSim cannot provide this if it does not work stably enough.

It is suggested to run the RRA module more than one time. Normally, more than two passes are not necessary. The adjusting values after the second pass should be very small. If they are not small enough, there is perhaps a problem existing in the model or with the ground reaction forces.

OpenSim offers the possibility to do Forward Dynamics. This is absolutely correct from a numerical and mathematical point of view. However, a lot of experimental data to compute the muscle excitation is needed and prior Inverse Kinematics and Inverse Dynamics have to be done, so that for most of the users and application areas it is not a real forward simulation but could be called a semi-forward simulation. The documentation and the official statement to the instability of Forward Dynamics (cf. chapter 4.4) is inconsistent with the test experiences. This leads to a major problem. Forward Dynamics is a necessary module for biomechanical simulation software. If the user can not use it and the user's guide does not give suitable explanations, the user can not count on that feature and is forced to use another software with greater reliability.

The most important feature is, due to the fact that OpenSim is not designed for creating or making major model changes, the import function. Because the SIMM import module does not yet work for most of the SIMM models, the software is very limited in its range of application. This problem was recognised by the development team and will hopefully lead to a practicable solution soon.

The idea and intention of OpenSim is great. In the version 1.5.5 the basic functions are working fine. Some major problems are recognised and the developers are working them up. The documentation is a minor problem and it is to be expected that it will become better, with more users working with it and the longer stable versions of all modules are available.
Experiences and Future Work

Working with OpenSim is not recommendable at the moment to users expecting great reliability. It is a good alternative for experimental usage and showing some examples of biomechanical engineering in teaching.

6.2 Version 1.8.1

This chapter contains conclusions referring to chapter 5 and the OpenSim version 1.8.1. Unfortunately there is still only a Windows 32-Bit version available. There is no support for other OS like Linux or MacOS and also no support for any 64-Bit systems. I think that should be one of the next steps, but it is not totally up to the development team of OpenSim but rather of the SimTK-group, which is responsible for the numerical background and the multi-body core engine.

In the test cases all SIMM models could be imported. This leads to a greater variety of possible applications of OpenSim even in running projects which are based on SIMM models. It does not lead to better results of OpenSim or to extension of what OpenSim is able to do. The SIMM compatibility is a very basic feature and absolutely necessary for OpenSim to compete in the market for biomechanical simulation software tools.

The User's Guides and the support in an open source project are always one of the biggest problems. From version 1.5.5 to v1.8.1 there is no big improvement of the User's Guides especially the chapter about the unique features RRA and CMC is still not sufficient and additional papers has to be consulted to understand these mechanisms. In the OpenSim project wiki there is an additional information document to RRA and CMC. As written in chapter 5.4.1 it is not possible for the user to fully understand how RRA works, because of a discrepancy between the User's Guide and the GUI. Although it is possible to use features which are working but without feasible explanations, it is often not satisfying and it is even more a problem if the user wants to change details or works with non standard cases, so it has to be checked if the specific algorithm can be used and in what way it computes its results.

In most of the cases in which OpenSim cannot find a solution or has a problem it does not completely break down anymore but instead it produces an error message with an error number. The messages are normally very short so that it is sometimes hard for the user to find out what exactly went wrong. The error numbers are not helpful because there is no explanation for them. The documentation of those error numbers has to be published.

Another documentation problem is that there are some options which are not available but hidden in the XML control files and can only be activated by editing those control files. It is confusing if the User's Guide does not mention them or, even worse, if it describes them in a wrong way. The problem exists the other way round, too: in the XML files are some options which do not have any effect on the computation or the output.

The stability of CMC has increased significantly. With now longer test cases and more stable simulations possible this module has become more interesting. Some results of the test cases seem reasonable and some simply do not. There is no real advance if the simulations are more
Experiences and Future Work

stable but do not provide good results, so that the results have to be validated and compared with measurements in more than one case study.

The main idea of inventing $CMC$ (and not using an old muscle force computation algorithm) was to make muscle force computation faster and possible for users even with lower level computing resources. The slow target option seems to be really slow, so that possibly that scope is not reached. A performance study would be interesting and should provide additional insight into stability and computing speed.

The new test case shows that the new stability of $CMC$ is fading away for motions without ground reaction forces. There is no official statement that $CMC$ is not suitable for such motions and there are so far no other user reports in the users’ forum which include positive statement about such motions. The main question which is left here is: Is it possible to run $CMC$ for motions without ground reactions?

All these results about $CMC$ mainly lead to the project proposal in chapter 5.7.

The instability of Forward Dynamics, as described in chapter 4.4, is still a problem and no further official announcements has been made. Due to this fact and the complexity of the musculoskeletal problems, OpenSim and the Forward Dynamics function should only be used in the way suggested in the User’s Guide: The first step is to track a motion and do the calculation steps including the computation of the muscle forces and excitations with $CMC$, the second step is to vary slightly the muscle excitations to do in a third step a forward dynamic simulation.

My general conclusion for the comparison between version 1.5.5 and 1.8.1 is that some modules has become considerably more stable, but slow and maybe the results are not always as good as with the older version.

This shows that OpenSim is still in an early stage of development. The announced improvement of the forward dynamics module gives reason for hope. OpenSim has a stable basis and a great potential, the newest version is one big step in the right direction.
Bibliography


Bibliography


[27] J. Wagner and H.P. Mlejnek: Dynamik 1 Vorlesungsunterlagen. Institut für Statik und Dynamik der Luft- und Raumfahrtkonstruktionen, Universität Stuttgart, Germany,
2008.


A Appendix

This appendix contains additional figures resulting of the test cases discussed in chapter 4 and 5 and source codes that were used to handle and modify motion data files.
A.1 Motion Laboratory

Table A.1: Description of the marker positions for the motion lab experiments

<table>
<thead>
<tr>
<th>Marker</th>
<th>Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>PL1</td>
<td>Pelvis Left ASIS</td>
</tr>
<tr>
<td>PR1</td>
<td>Pelvis Right ASIS</td>
</tr>
<tr>
<td>PR2</td>
<td>Pelvis Right PSIS</td>
</tr>
<tr>
<td>RF1</td>
<td>Right Greater Trochanter</td>
</tr>
<tr>
<td>RF2</td>
<td>Right Medial Femural Condyle</td>
</tr>
<tr>
<td>RF3</td>
<td>Right Lateral Femural Condyle</td>
</tr>
<tr>
<td>RT1</td>
<td>Right Medial Tibia</td>
</tr>
<tr>
<td>RT2</td>
<td>Right Lateral Tibia/Fibular Head</td>
</tr>
<tr>
<td>RT3</td>
<td>Right Medial Malleolus</td>
</tr>
<tr>
<td>RT4</td>
<td>Right Lateral Malleolus</td>
</tr>
</tbody>
</table>

Figure A.1: Marker positions for the motion lab recordings
Appendix

A.1.1 Hip Flexion/Extension

Figure A.2: Hip Flexion/Extension - Hip transverse plane

Figure A.3: Hip Flexion/Extension - Knee transverse plane

Figure A.4: Hip Flexion/Extension - Hip sagittal plane

Figure A.5: Hip Flexion/Extension - Knee sagittal plane

Figure A.6: Hip Flexion/Extension - Hip frontal plane

Figure A.7: Hip Flexion/Extension - Knee frontal plane
A.1.2 Hip Ab-/Adduction

Figure A.8: Hip Ab-/Adduction - Hip transverse plane

Figure A.9: Hip Ab-/Adduction - Knee transverse plane

Figure A.10: Hip Ab-/Adduction - Hip sagittal plane

Figure A.11: Hip Ab-/Adduction - Knee sagittal plane

Figure A.12: Hip Ab-/Adduction - Hip frontal plane

Figure A.13: Hip Ab-/Adduction - Knee frontal plane
A.1.3 Hip Rotation

Figure A.14: Hip Rotation - Hip transverse plane

Figure A.15: Hip Rotation - Knee transverse plane

Figure A.16: Hip Rotation - Hip sagittal plane

Figure A.17: Hip Rotation - Knee sagittal plane

Figure A.18: Hip Rotation - Hip frontal plane

Figure A.19: Hip Rotation - Knee frontal plane
A.1.4 Knee Flexion/Extension

Figure A.20: Knee Flexion/Extension - Hip transverse plane

Figure A.21: Knee Flexion/Extension - Knee transverse plane

Figure A.22: Knee Flexion/Extension - Hip sagittal plane

Figure A.23: Knee Flexion/Extension - Knee sagittal plane

Figure A.24: Knee Flexion/Extension - Hip frontal plane

Figure A.25: Knee Flexion/Extension - Knee frontal plane
A.2.2 Change Excitation Patterns

Figure A.26: Comparison of original and changed excitations of right hip extensors (Iliopsoas) for the test case described in chapter 4.2.2

Figure A.27: Right knee flexion/extension resulting from original and changed excitations shown in figure A.26 and described in chapter 4.2.2


A.2 Manipulate Motions

A.2.1 Musculoskeletal Geometry

Table A.2: Muscle-tendon Parameters for 43 Lower-Limb Muscles [3]

<table>
<thead>
<tr>
<th>muscle</th>
<th>peak force ($F_{m0}^p$)(N)</th>
<th>optimal fibre length ($l_{m0}^f$)(cm)</th>
<th>pennation angle $(\alpha)$(degrees)</th>
<th>tendon slack length ($l_{TS}$)(cm)</th>
<th>tendon length/fibre length $(l_{TS}^f) / (l_{m0}^f)$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>hip muscles</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>gluteus medius1</td>
<td>550</td>
<td>5.4</td>
<td>8</td>
<td>7.8</td>
<td>1.4</td>
</tr>
<tr>
<td>gluteus medius2</td>
<td>380</td>
<td>8.4</td>
<td>0</td>
<td>5.3</td>
<td>0.6</td>
</tr>
<tr>
<td>gluteus medius3</td>
<td>435</td>
<td>6.5</td>
<td>19</td>
<td>5.3</td>
<td>0.8</td>
</tr>
<tr>
<td>gluteus minimus1</td>
<td>180</td>
<td>6.8</td>
<td>10</td>
<td>1.6</td>
<td>0.2</td>
</tr>
<tr>
<td>gluteus minimus2</td>
<td>190</td>
<td>5.6</td>
<td>0</td>
<td>2.6</td>
<td>0.5</td>
</tr>
<tr>
<td>gluteus minimus3</td>
<td>215</td>
<td>3.8</td>
<td>1</td>
<td>5.1</td>
<td>1.3</td>
</tr>
<tr>
<td>gluteus maximus1</td>
<td>380</td>
<td>14.2</td>
<td>5</td>
<td>12.5</td>
<td>0.9</td>
</tr>
<tr>
<td>gluteus maximus2</td>
<td>550</td>
<td>14.7</td>
<td>0</td>
<td>12.7</td>
<td>0.9</td>
</tr>
<tr>
<td>gluteus maximus3</td>
<td>370</td>
<td>14.4</td>
<td>5</td>
<td>14.5</td>
<td>1.0</td>
</tr>
<tr>
<td>adductor magnus1</td>
<td>345</td>
<td>8.7</td>
<td>5</td>
<td>6.0</td>
<td>0.7</td>
</tr>
<tr>
<td>adductor magnus2</td>
<td>310</td>
<td>12.1</td>
<td>3</td>
<td>13.0</td>
<td>1.0</td>
</tr>
<tr>
<td>adductor magnus3</td>
<td>445</td>
<td>13.1</td>
<td>5</td>
<td>20.0</td>
<td>2.0</td>
</tr>
<tr>
<td>adductor longus</td>
<td>420</td>
<td>13.8</td>
<td>6</td>
<td>11.0</td>
<td>0.8</td>
</tr>
<tr>
<td>adductor brevis</td>
<td>285</td>
<td>13.3</td>
<td>0</td>
<td>2.0</td>
<td>0.2</td>
</tr>
<tr>
<td>pectineus</td>
<td>175</td>
<td>13.3</td>
<td>0</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>iliacus</td>
<td>430</td>
<td>10.0</td>
<td>7</td>
<td>9.0</td>
<td>0.9</td>
</tr>
<tr>
<td>psoas</td>
<td>370</td>
<td>10.4</td>
<td>8</td>
<td>13.0</td>
<td>1.3</td>
</tr>
<tr>
<td>quadratus femoris</td>
<td>225</td>
<td>5.4</td>
<td>0</td>
<td>2.4</td>
<td>0.4</td>
</tr>
<tr>
<td>pectineus</td>
<td>110</td>
<td>2.4</td>
<td>0</td>
<td>3.9</td>
<td>1.6</td>
</tr>
<tr>
<td>piriiformis</td>
<td>205</td>
<td>2.6</td>
<td>10</td>
<td>11.5</td>
<td>4.4</td>
</tr>
<tr>
<td><strong>hip and knee muscles</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rectus femoris</td>
<td>780</td>
<td>8.4</td>
<td>5</td>
<td>34.6</td>
<td>4.0</td>
</tr>
<tr>
<td>semimembranosus</td>
<td>1030</td>
<td>8.0</td>
<td>15</td>
<td>35.9</td>
<td>4.5</td>
</tr>
<tr>
<td>semitendinosus</td>
<td>330</td>
<td>20.1</td>
<td>5</td>
<td>28.2</td>
<td>1.3</td>
</tr>
<tr>
<td>biceps femoris (lh)</td>
<td>720</td>
<td>10.9</td>
<td>0</td>
<td>34.1</td>
<td>3.1</td>
</tr>
<tr>
<td>gracilis</td>
<td>110</td>
<td>35.2</td>
<td>3</td>
<td>14.0</td>
<td>0.4</td>
</tr>
<tr>
<td>sartorius</td>
<td>105</td>
<td>57.9</td>
<td>0</td>
<td>4.0</td>
<td>0.1</td>
</tr>
<tr>
<td>tensor fasciae latae</td>
<td>155</td>
<td>9.5</td>
<td>3</td>
<td>42.5</td>
<td>4.5</td>
</tr>
<tr>
<td><strong>knee muscles</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vastus medialis</td>
<td>1235</td>
<td>8.9</td>
<td>5</td>
<td>12.6</td>
<td>1.4</td>
</tr>
<tr>
<td>vastus intermedius</td>
<td>1235</td>
<td>8.7</td>
<td>3</td>
<td>13.6</td>
<td>1.6</td>
</tr>
<tr>
<td>vastus lateralis</td>
<td>1870</td>
<td>8.4</td>
<td>5</td>
<td>15.7</td>
<td>1.9</td>
</tr>
<tr>
<td>biceps femoris (sh)</td>
<td>400</td>
<td>17.3</td>
<td>23</td>
<td>10.0</td>
<td>0.6</td>
</tr>
<tr>
<td><strong>knee and ankle muscles</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>medial gastrocnemius</td>
<td>1115</td>
<td>4.5</td>
<td>17</td>
<td>40.8</td>
<td>9.0</td>
</tr>
<tr>
<td>lateral gastrocnemius</td>
<td>140</td>
<td>6.4</td>
<td>8</td>
<td>38.5</td>
<td>6.0</td>
</tr>
<tr>
<td><strong>ankle muscles</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>soleus</td>
<td>2830</td>
<td>3.0</td>
<td>25</td>
<td>28.8</td>
<td>8.9</td>
</tr>
<tr>
<td>tibialis posterior</td>
<td>1270</td>
<td>3.1</td>
<td>12</td>
<td>31.0</td>
<td>10.0</td>
</tr>
<tr>
<td>fl. digitorum longus</td>
<td>310</td>
<td>3.4</td>
<td>7</td>
<td>40.0</td>
<td>11.8</td>
</tr>
<tr>
<td>fl. hallucis longus</td>
<td>350</td>
<td>4.3</td>
<td>10</td>
<td>38.0</td>
<td>8.8</td>
</tr>
<tr>
<td>peroneus brevis</td>
<td>350</td>
<td>5.0</td>
<td>5</td>
<td>16.1</td>
<td>3.2</td>
</tr>
<tr>
<td>peroneus longus</td>
<td>755</td>
<td>4.9</td>
<td>10</td>
<td>34.5</td>
<td>7.0</td>
</tr>
<tr>
<td>tibialis anterior</td>
<td>600</td>
<td>9.8</td>
<td>5</td>
<td>22.3</td>
<td>2.2</td>
</tr>
<tr>
<td>peroneus tertius</td>
<td>90</td>
<td>7.9</td>
<td>13</td>
<td>10.0</td>
<td>1.3</td>
</tr>
<tr>
<td>ext. digitorum longus</td>
<td>340</td>
<td>10.2</td>
<td>8</td>
<td>34.5</td>
<td>3.4</td>
</tr>
<tr>
<td>ext. hallucis longus</td>
<td>110</td>
<td>11.1</td>
<td>6</td>
<td>30.5</td>
<td>2.6</td>
</tr>
</tbody>
</table>
A.2.3 Scale Motions

Figure A.28: Comparison of CMC calculated excitation of right Quadriceps Femoris before and after scaling the motion, figure 4.7, and described in chapter 4.2.3

Figure A.29: Comparison of CMC calculated excitation of right Plantaflexors before and after scaling the motion, figure 4.7, and described in chapter 4.2.3
A.2.4 Create New Motions

Figure A.30: Separate muscle forces of the main flexors/extensors of the hip/knee during knee bending described in chapter 4.2.4, the separate muscle forces are shown in figure 4.12

Figure A.31: Separate muscle forces of the main flexors/extensors of the hip/knee during the kick movement described in chapter 4.2.4, the summed muscle forces are shown in figure 4.13
A.3 Extension

A.3.1 Change Excitation Patterns

Figure A.32: Summation of the CMC calculated excitations of the main knee flexors

Figure A.33: Summation of the CMC calculated excitations of the main knee extensors
A.3.2 Scale Motions

![Graph showing comparison of CMC calculated excitation of right Quadriceps femoris for a longer period of time before and after scaling the motion, figure 5.5, and described in chapters 4.2.3 and 5.5.2.]

**Figure A.34:** Comparison of CMC calculated excitation of right Quadriceps femoris for a longer period of time before and after scaling the motion, figure 5.5, and described in chapters 4.2.3 and 5.5.2

![Graph showing comparison of CMC calculated excitation of right Plantaflexors for a longer period of time before and after scaling the motion, figure 5.5, and described in chapters 4.2.3 and 5.5.2.]

**Figure A.35:** Comparison of CMC calculated excitation of right Plantaflexors for a longer period of time before and after scaling the motion, figure 5.5, and described in chapters 4.2.3 and 5.5.2
Appendix

A.3.3 Movements Without Ground Reactions

Figure A.36: CMC computed motion without ground reaction forces: y coordinate of the pelvis, angles of the right hip flexion and the lumbar extension.

Figure A.37: Inverse Dynamics computed motion without ground reaction forces: y coordinate of the pelvis, angles of the right hip flexion and the lumbar extension.
Listing A.1: MATLAB source code for calculating joint angles (of hip flexion/extension)

```matlab
import file HipFlexionExtension.txt;

% allocating matrices
PHI_p = zeros(length(data),3);
PHI_h = zeros(length(data),3);
PHI_k = zeros(length(data),3);
KJC = zeros(length(data),3); % knee joint centre
FC = zeros(length(data),3); %"femur centre"
TC = zeros(length(data),3); %"tibia centre"
AJC = zeros(length(data),3); %ankle joint centre
HJC = zeros(length(data),3); %hip joint centre
PO = zeros(length(data),3); %pelvis cos origin

for i = 1:length(data)
    % definition of the pelvis cos
    PR1 = [data(i,18) data(i,19) data(i,20)]';
    PR1_all(i,1:3)=PR1;
    PL1 = [data(i,24) data(i,25) data(i,26)]';
    ey_p = (PR1 -PL1)/norm(PR1 -PL1);
    PR2 = [data(i,21) data(i,22) data(i,23)]';
    lambda = ((PR2 -PR1)'*ey_p)/(ey_p'*ey_p);
    ex_p = PR2 -PR1 - lambda * ey_p; %not normalised
    ex_p = ex_p / norm(ex_p); %normalised
    ez_p=cross(ex_p,ey_p); %not normalised
    ez_p=ez_p/norm(ez_p); %normalised
    T_p = [ex_p ey_p ez_p];

    %Transformation matrix world into pelvis at timestep i
    EX_p(i,1:3)=ex_p;
    EY_p(i,1:3)=ey_p;
    EZ_p(i,1:3)=ez_p;

    % set the origin of the pelvis coordinate system in the middle between
    % PR1 and PL1
    po=0.5* (PR1 + PL1);
    PO(i,1:3) = po';

    % the hip joint centre is assumed to be (-50, 0, -50)
    % from PR1 in the pelvis coordinate system
    PR1_p=T_p*PR1-po;
    hjc_p=PR1_p + [-50 0 -50]';
    hjc = T_p(hjc_p-po);
    HJC(i,1:3) = hjc';

    % calculate the ankle joint centre
    RT3 = [data(i,27) data(i,28) data(i,29)]';
    RT4 = [data(i,30) data(i,31) data(i,32)]';
```
ajc = 0.5* (RT4+RT3);
AJC(i,1:3)=ajc;

% calculating assistance points fc, tc:
% calculating the centre of RT1, RT2
RT1 = [data(i,12) data(i,13) data(i,14)]';
RT2 = [data(i,9) data(i,10) data(i,11)]';
tc = 0.5* (RT2-RT1);
TC(i,1:3)=tc';

% calculating the centre of RF2, RF3
RF2 = [data(i,6) data(i,7) data(i,8)]';
RF3 = [data(i,3) data(i,4) data(i,5)]';
f = 0.5* (RF3-RF2);
FC(i,1:3)=f';

% calculate the knee joint centre under the assumption, that the
% intersection of the vector from ajc to tc and the vector
% from hjc to f defines it
templ=[(hjc(1)-f(1)) (tc(1)-ajc(1));
(hjc(2)-f(2)) (tc(2)-ajc(2))]
[kjc= hjc+ templ(1)* (hjc-fc);
KJC(i,1:3)=kjc';

% definition of the hip cos, origin is kjc
%ez_h = (kjc - hjc)/norm(kjc - hjc);
ey_h = (fc - hjc)/norm(fc - hjc);
ex_h = (RF3-RF2)/norm(RF3-RF2);
ex_h = cross(ey_h,ez_h); % not normalised
ex_h = ex_h /norm(ex_h); % normalised
T_h = [ex_h ey_h ez_h];

% Transformation matrix world into hip at timestep i
T_p= T_h/T_p;
% Transformation matrix pelvis into hip at timestep i
EX_h(i,1:3)=ex_h;
EY_h(i,1:3)=ey_h;
EZ_h(i,1:3)=ez_h;

% definition of the knee cos, origin is ajc
%ez_k = (ajc - kjc)/norm(ajc - kjc);
ey_k = (ajc - tc)/norm(ajc - tc);
ex_k = (RT4-RT3)/norm(RT4-RT3);
ex_k = cross(ey_k,ez_k); % not normalised
ex_k = ex_k /norm(ex_k); % normalised
T_k = [ex_k ey_k ez_k];

% Transformation matrix world into hip at timestep i
T_k= T_k/T_h;
% Transformation matrix hip into knee at timestep i
EX_k(i,1:3)=ex_k;
EY_k(i,1:3)=ey_k;
EZ_k(i,1:3)=ez_k;
calculating the Euler angles with convention zyx (Tait-Bryan)

% pelvis in respect to world
% phi = transverse, sagittal, frontal

phi_p = atan(T_p(1,2)/T_p(1,1)) asin(-T_p(1,3)) atan(T_p(2,3)/T_p(3,3));
PHI_p(i,1:3) = phi_p*180/pi;

phi_h = [-atan(T_ph(1,2)/T_ph(1,1)) asin(-T_ph(1,3)) atan(T_ph(2,3)/T_ph(3,3))];
PHI_h(i,1:3) = phi_h*180/pi;

phi_k = [-atan(T_hk(1,2)/T_hk(1,1)) asin(-T_hk(1,3)) atan(T_hk(2,3)/T_hk(3,3))];
PHI_k(i,1:3) = phi_k*180/pi;
### A.4.2 Scale Motions

**Listing A.2: Perl source code for modifying motion files**

```perl
#!/usr/bin/perl

# read *.mot file ------------------------------------------

$file = 'sourcefile.sto';

# READIN: print "File:";
#$file = <STDIN>;

if (open(MOTION, $file))
{
    @lines = <MOTION>;
    close(MOTION);
}
else
{
    print "Error, try again! \n";
    goto READIN;
}
# end read in-----------------------------------------------

# clean file and sort header out------------------------

foreach $each (@lines)
{
    if ($each =~ /\s+\d+/)
    {
        $each =~ s/\s+//;
        $each =~ s/\n/\n\eol/g;
        $each =~ s/\s+//g;
        @temp = split(/\n/,$each);
        push(@numbers,@temp);
    } else
    {
        push (@header,$each);
    }
}

$last=@header;
while ($header[$last -1] =~ /\s+/)
{
    pop (@header);
    $last=@header;
}

$temp2 = $header[$last -1];
chomp(@caption);
@caption = split(/\s+/, $temp2);
# end of file cleaning---------------------------------------

# choose the column(s) to modify---------------------------

MODIFY: $n=0;
print "\n";

foreach $m (@caption)
{
```
print "$n \t $m \n";
++$n;
}

print "Choose the parameter you wish to modify (number):";
$i = <STDIN>;

#define the modifying function-----------------------------
# print "function:";
#$function = <STDIN>;
$function = 1.05;
# end of function definig -----------------------------

#define modify partial data-------------------------------
$anzahl=@numbers;
$increment=@caption+1;

for ($i; $i<$anzahl; $i += $increment)
{
    print "old: $numbers[$i] \n";
    $numbers[$i] += $function;
    #$numbers[$i] += $function;
    print "new: $numbers[$i] \n";
}

print "Do you want to modify another parameter (y/n)?";
chomp ($yesno = <STDIN>);
if ($yesno eq "y")
{
    goto MODIFY;
}

# end of data modifying ---------------------------------

# write output data--------------------------------------
open (OUTPUT, ">output.mot");
print OUTPUT @header;
foreach $item (@numbers)
{
    if ($item eq "eol")
    {
        print OUTPUT "\n";
    }
    else
    {
        print OUTPUT "$item\t";
    }
}

close (OUTPUT);

# end of writing-----------------------------------------